H264解码详解

H264是新一代的编码标准，以高压缩高质量和支持多种网络的流媒体传输著称，在编码方面，我理解的他的理论依据是：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。所以对于一段变化不大图像画面，我们可以先编码出一个完整的图像帧A，随后的B帧就不编码全部图像，只写入与A帧的差别，这样B帧的大小就只有完整帧的1/10或更小！B帧之后的C帧如果变化不大，我们可以继续以参考B的方式编码C帧，这样循环下去。这段图像我们称为一个序列（序列就是有相同特点的一段数据），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A1，随后的图像就参考A1生成，只写入与A1的差别内容。

在H264协议里定义了三种帧，完整编码的帧叫I帧，参考之前的I帧生成的只包含差异部分编码的帧叫P帧，还有一种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。

序列的说明

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。

***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。**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。*

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。

三种帧的说明

I帧:帧内编码帧，I帧表示关键帧，你可以理解为这一帧画面的完整保留；解码时只需要本帧数据就可以完成（因为包含完整画面）

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。

注: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。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。

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),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（Spatialcompression）。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（Temporalcompression），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（Framedifferencing）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（Lossyless）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。

H264层次构成

H264标准是由JVT（Joint Video Team，视频联合工作组）组织提出的新一代数字视频编码标准。JVT于2001年12月在泰国Pattaya成立。它由ITU-T的VCEG（视频编码专家组）和ISO/IEC的MPEG（活动图像编码专家组）两个国际标准化组织的专家联合组成。JVT的工作目标是制定一个新的视频编码标准，以实现视频的高压缩比、高图像质量、良好的网络适应性等目标H264标准。H264标准将作为MPEG-4标准的一个新的部分（MPEG-4 part.10）而获得批准，是一个面向未来IP和无线环境下的新数字视频压缩编码标准。

H264标准的主要特点如下：

1．更高的编码效率：同H.263等标准的特率效率相比，能够平均节省大于50％的码率。

2．高质量的视频画面：H.264能够在低码率情况下提供高质量的视频图像，在较低带宽上提供高质量的图像传输是H.264的应用亮点。

3．提高网络适应能力：H.264可以工作在实时通信应用（如视频会议）低延时模式下，也可以工作在没有延时的视频存储或视频流服务器中。

4．采用混合编码结构：同H.263相同，H.264也使用采用DCT变换编码加DPCM的差分编码的混合编码结构，还增加了如多模式运动估计、帧内预测、多帧预测、基于内容的变长编码、4x4二维整数变换等新的编码方式，提高了编码效率。

5．H.264的编码选项较少：在H.263中编码时往往需要设置相当多选项，增加了编码的难度，而H.264做到了力求简洁的“回归基本”，降低了编码时复杂度。

6．H.264可以应用在不同场合：H.264可以根据不同的环境使用不同的传输和播放速率，并且提供了丰富的错误处理工具，可以很好的控制或消除丢包和误码。

7．错误恢复功能：H.264提供了解决网络传输包丢失的问题的工具，适用于在高误码率传输的无线网络中传输视频数据。

8．较高的复杂度：264性能的改进是以增加复杂性为代价而获得的。据估计，H.264编码的计算复杂度大约相当于H.263的3倍，解码复杂度大约相当于H.263的2倍。

H264标准各主要部分有Access Unit delimiter（访问单元分割符），SEI（附加增强信息），primary coded picture（基本图像编码），Redundant Coded Picture（冗余图像编码）。还有Instantaneous Decoding Refresh（IDR，即时解码刷新）、Hypothetical Reference Decoder（HRD，假想码流调度器）、Hypothetical Stream Scheduler（HSS，假想参考解码）。

主要部分结构如图3.18所示：

H.264的目标应用涵盖了目前大部分的视频服务，如有线电视远程监控、交互媒体、数字电视、视频会议、视频点播、流媒体服务等。H.264为解决不同应用中的网络传输的差异。定义了两层：视频编码层（VCL：Video Coding Layer）负责高效的视频内容表示，网络提取层（NAL：Network Abstraction Layer）负责以网络所要求的恰当的方式对数据进行打包和传送。如图3.19所示。

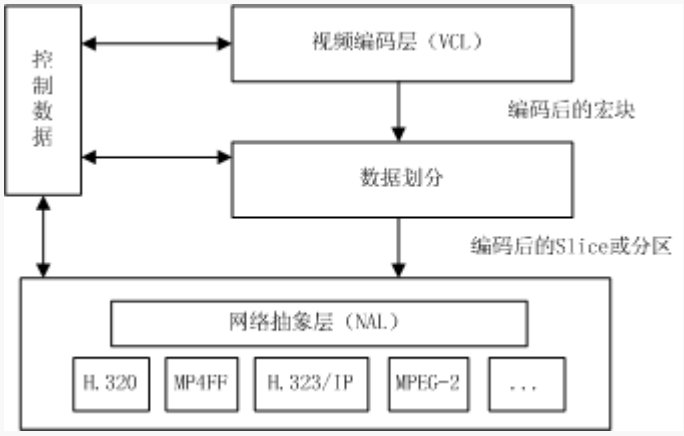

图3.19 标准的整体框架

基本层次（Baseline Profile）：该层次使用了H.264的除了B-Slices，CABAC以及交织编码模式外所有的特性。该层次主要使用于低时延的实时应用场合。

主要层次（Main Profile）：包含Baseline profile的所有特性，并包括了B-slices，CABAC以及交织编码模式。它主要针对对时延要求不高，当压缩率和质量要求较高的场合。

扩展层次(Profile X)：支持所有Baseline profile的特性，但不支持CABAC以及基于宏块的自适应帧场编码。该层次主要针对的时各种网络视频流传输方面的应用。

CABAC

CABAC是基于内容的自适应二进制算术编码，当参数entropy_coding_mode设置为1时，一个算术系统被用来编码和解码H.264的语法元素。

H.264采用两种方法进行熵编码：CAVLC编码和CABAC编码算法。采用基于上下文的自适应二进制算术编码算法（CABAC），能够充分利用上下文信息和算术编码的优点，使得编码后的平均码长更逼近图像的信息熵，达到最佳的编码效率。采用CABAC算法进行编码，可以提高大约10%的编码率

具体编码步骤：

1二值化：CABAC使用二进制算术编码，所以要将数据先转换为二进制数据，这些原始数据包括变换系数和运动矢量等。转换后二进制数据为可变长编码的数据，并且还要将这些数据进行算术编码。

2内容模式选择：内容模式是针对二进制数据进行统计的概率模型，这个模式根据之前编码的一些数据符号的统计特性从一些可选模式中选出。内容模式存储了每一位“1”或“0”的概率。

3算术编码：算术编码器根据选择的内容模式对每一位进行编码。

4概率校正：被选择的内容模式根据实际被编码的值进行校正，例如，如果数据比特流中有数值“1”，就将“1”的概率统计值加1。

DCT变换

H.264仍然采用对残差信号进行变换在量化后进行熵编码的模式来压缩空间冗余信息。使用了类似于4x4离散余弦变换DCT的整数变换而不是象MPEG4那样采用8x8DCT的浮点数变换。最终使用那种变换方式还用根据残余数据类型的不同来选择，帧内编码宏块的亮度DC系数（仅对16x16预测模式有效）采用4x4的矩阵，色度DC系数采用2x2的矩阵，对于其他的都采用4X4的块来变换。

使用以整数为基础的空间变换可以提高计算速度（只使用加法和位移运算），但是使用整数变换要以不失精确度为前提；整数变换的反变换过程中不会出现较大的误差，并且缩放矩阵的乘法集成到了量化中，降低了乘法的总次数。

（1）4×4亮度分量的直流系数变换

如果宏块被编码为16×16帧内模式，则每个4×4残差块首先用前面叙述的变换进行变换，然后对于每个4×4的变换后的直流（DC）系数进行4×4的二次变换，采用Hadamard变换。
正变换为：

$$Y_D = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} & W_D & \\ & & \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}_1 \right) / 2$$

$$W_{QD} = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} & Z_D & \\ & & \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}_1 \right) / 2$$

其中A是变换核矩阵
a=1/2

$$b = \sqrt{1/2}\cos(\pi/8)$$

$$c = \sqrt{1/2}\cos(3\pi/8)$$

$$Y = AXA^T = \begin{bmatrix} a & a & a & a \\ b & c & -a & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} \begin{bmatrix} a & b & a & c \\ a & c & -a & -b \\ a & -c & -a & b \\ a & -b & a & -c \end{bmatrix}$$

$$Y = (CXC^T) \otimes E =$$

$$\left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & d & -d & -1 \\ 1 & -1 & -1 & 1 \\ d & -1 & 1 & -d \end{bmatrix} \begin{bmatrix} x_{11} & x_{12} & x_{13} & x_{14} \\ x_{21} & x_{22} & x_{23} & x_{24} \\ x_{31} & x_{32} & x_{33} & x_{34} \\ x_{41} & x_{42} & x_{43} & x_{44} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & d \\ 1 & d & -1 & -1 \\ 1 & -d & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \right) \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix}$$

$$Y = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \begin{bmatrix} x_{00} & x_{01} & x_{02} & x_{03} \\ x_{10} & x_{11} & x_{12} & x_{13} \\ x_{20} & x_{21} & x_{22} & x_{23} \\ x_{30} & x_{31} & x_{32} & x_{33} \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 1 & -1 \end{bmatrix}$$

（2）2×2色度块的DC系数变换

每个宏块内的4个4×4色度块经过变换后，每个块的DC系数构成了一个2×2的块WD，对其进行2×2的Hadamard变换。

正变换的公式为：

$$Y_D = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} W_D & \\ & \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

反变换公式为：

$$W_{QD}=\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}Z_D & \\ & \end{bmatrix}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$$

（3）如图3.18所示，展示了宏块中的变换块及其传送顺序。编号为-1的块在采用Intra16x16模式编码时0-15号4x4子块经整数DCT变换后的DC系数在经4x4的哈达变换的结果。块16、17是色度块的DC系数进行2x2哈达码变换的结果。其余的24块则进行4x4整数变换。

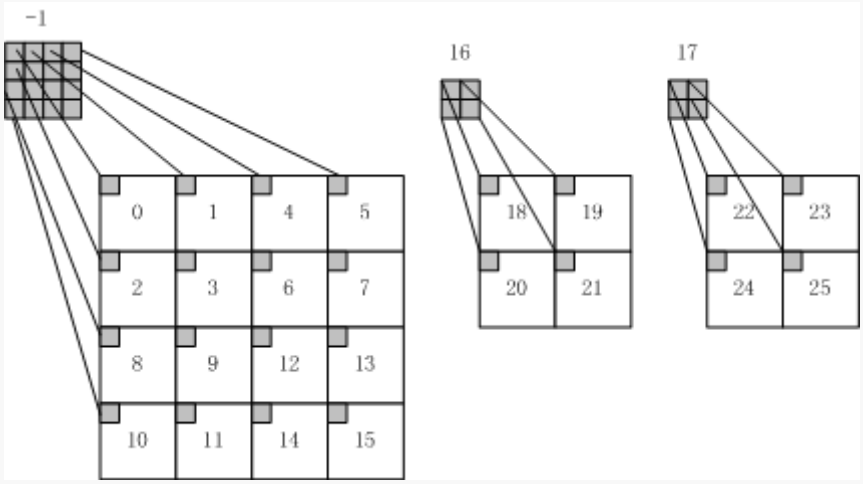

图3.20 宏块中的变换及其传送顺序

多种运动补偿块

有7种形状的运动补偿可供选用，这7种块是：INTER16x16，INTER16x8，INTER8x16，INTER8x8，INTER8x4，INTER4x8，INTER4x4。根据运动补偿采用的块尺寸的不同，宏块的编码模式分为四种，前三种模式分别按照一个16x16块、两个16x8块和两个8x16块来进行运动补偿；最后一种模式记作P8x8，在P8x8模式下，一个宏块被分为4个8x8的子块，而每一个子块又有4种可能的子模式，分别按照一个8x8块、两个8x4块、两个4x8块及四个4x4块进行运动补偿，如图3.19所示，第一行是宏块四种模式，第二行是子块四种模式。

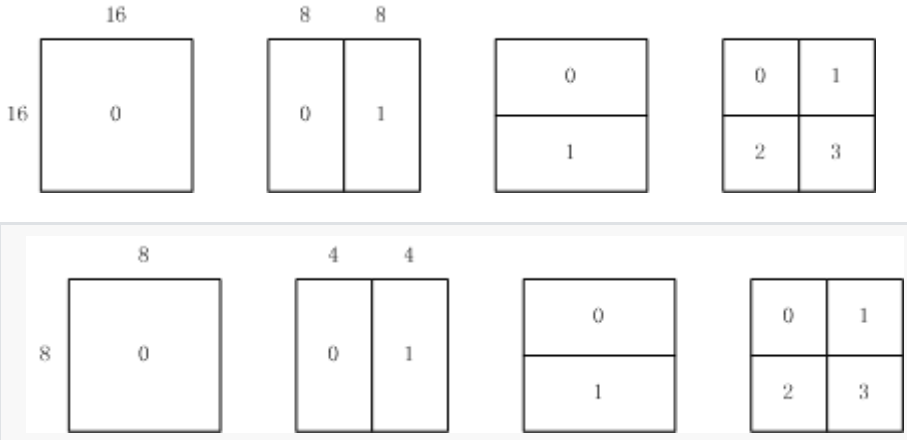

图3.21 宏块划分方式

块大小的选择是否合理对于压缩效果的好坏有很大的影响，通常来说，对于变化缓慢的部分采用较大分块效果比较好，对于包含较多细节的部分则应该采用较小的分块方式。

1/4像素精度运动估计

帧内编码宏块的每一分块都是由参考帧中相同大小的区域预测得到。这两个区域之间的偏移量即运动矢量。由于图像的运动不可能总是整像素的。因此引入了亚像素运动矢量。对亮度分量，运动矢量的分辨率为1/4像素。由于参考帧中本身不可能存在亚像素采样点，因此需要利用其临近像素内插产生亚像素采样点。亚像素采样点的内插产生过程，如图3.20所示

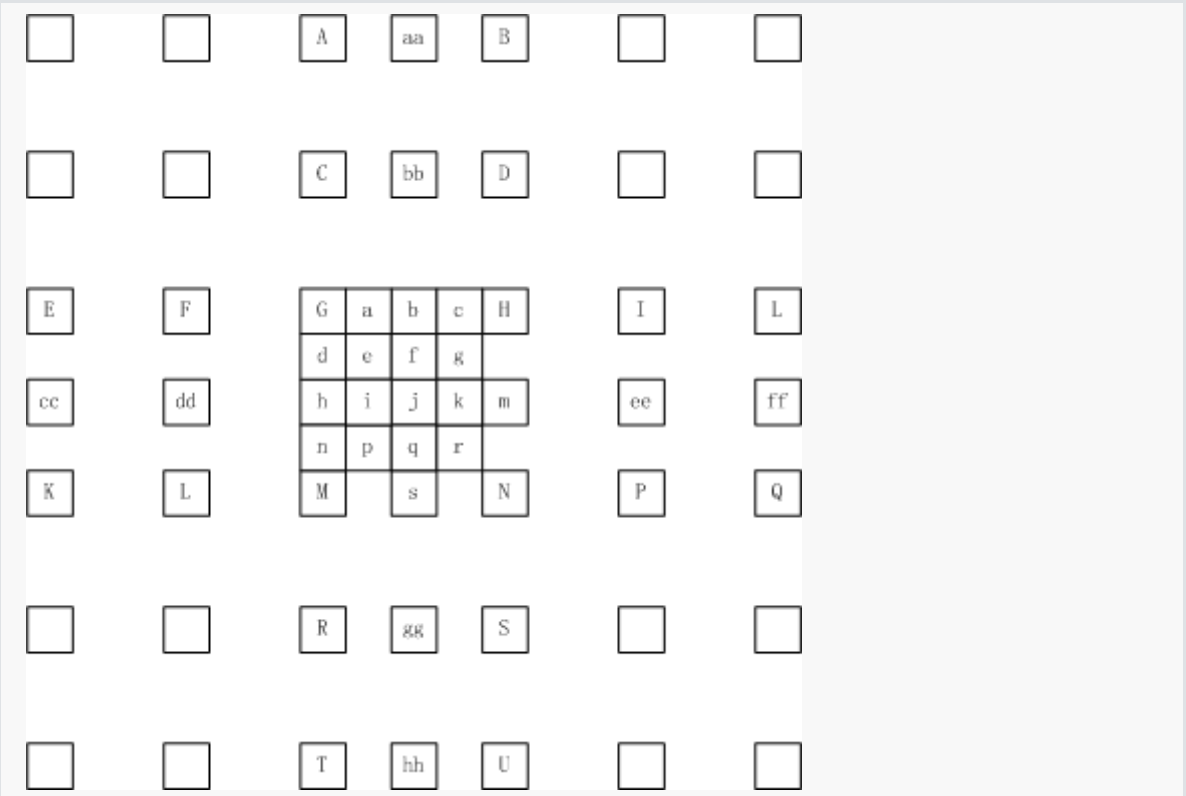

图3.22 亚像素采样点

半像素内插值分别由运动于水平和垂直方向的一维6阶滤波器产生。1/4像素值由整数像素和半像素点求均值取得。
例如：

```
b=round（（E-5F+20G+20H-5I+J）/32） a=round（（G+b）/2） e=round（（b+h）/2）
```

由于亮度分量中的1/4像素精度运动矢量将在色度分量中产生1/8像素精度。因此，采用线性内插法产生1/8像素采样点。

```
a=round（（[（8-dx）.（8-dy）A+dx.（8-dy）B+（8-dx）.dyC+dx.dyD]/64）
```

图片分割

H.264支持slice结构的图片分割。一个slice有一帧图片内的若干宏块组成。编码器端对slice种包含的宏块数目没有限制。一个slice可以仅包含一个宏块也可以包含该帧中的所有宏块。然而，任何一个宏块都只能包含在某一个slice中，不允许重复出现（在冗余slice方法中例外）。
采用slice结构的主要动机是使编码的slice大小能适应不同的MTU大小。当它同时能应用于交叉打包等方法的实现方案中。

多参考帧选择

多参考帧选择在之前的一些视频编码标准中也可以得到应用。该方法尤其使用于具有反馈机制的系统中。但在时延要求较高的应用中意义不大。
与以往标准的P帧、B帧不同，H.264采用了前向与后向多个参考帧的预测

数据分快

通常，宏块中素有的码元都是被编码在单一的比特串中的。数据分块则为每一个slice创建多个比特串。在H.264中，使用了三种不同类型的数据分块。

头信息块，包括宏块类型，量化参数，运动矢量。这些信息是最重要的，因为离开他们，被的数据块种的码元都无法使用。该数据分块称为A类数据分块。

帧内编码信息数据块，称为B类数据分块。它包含帧内编码宏块类型，帧内编码系数。对应的slice来说，B类数据分块的可用性依赖于A类数据分块。和帧间编码信息数据块不通的是，帧内编码信息能防止进一步的偏差，因此比帧间编码信息更重要。

帧间编码信息数据块，称为C类数据分块。它包含帧间编码宏块类型，帧间编码系数。它通常是slice种最大的一部分。帧间编码信息数据块是不重要的一部分。它所包含的信息并不提供编解码器之间的同步。C类数据分块的可用性也依赖于A类数据分块，但于B类数据分块无关。

当采用数据分块方式的时候，源编码器将不通类型的码元放到三个不同的比特缓冲器种此外，slice大小也需要调整，以使最大数据分块不会大于最大的MTU尺寸。以此，对数据分块进行操作的是源编码器而不是NAL。
在解码器端，在开始正确解码之前必须获得所有数据分块信息。然而，如果帧间或帧内编码数据块信息丢失了，头信息仍然能够有效地应用于提高差错恢复效率。头信息种包含宏块类型，用动矢量等信息，因此能够据此较高质量地复制信息。而仅仅丢失了一些图像纹理信息。

参数集

序列参数集包括与一图片序列相关地所有信息。图像参数集包含与图像中所有slice相关地信息。在解码器端可以存储多个不同地序列和图片参数集。编码器可以选择适当地图片参数集，图片参数集本身又包含所引用地序列参数集信息。

参数集的创造性应用极大地提高了错误恢复性能。在容错环境中使用参数集地关键是确保参数集能可靠并及时地到达接受端解码器。一次可以用频带外可靠通讯控制协议传送参数集，并确保在解码器从实时通讯信道接收到第一个需要参考该参数集地slice数据之前送达。或者也可以在频带内传输，但必须采用一些应用层保护措施（例如传送一参数集地多个复制，以提高至少一个复制到底目的地地概率）。第三中方案是在编码器和解码器端预先放置一些参数集，编解码器都必须在其中选择参数集。

可变宏块排序

可变宏块排序（FMO，Flexible Macroblock Ordering）可以在Baseline和Ext4ended模式中使用，但不允许在Main模式重使用。可变宏块排序允许将宏块不按照扫描顺序分配给slice。具体地分配策略由一宏块分配映射图（MBAmap）规定。在slice内，宏块仍然按照正常地扫描顺序编码。

该特性提供了一种将一帧图像中的宏块分配到多个slice中的模式，每个slice都是一个独立的编码单位，无论是帧间还是帧内编码都不能越界，如果在传输过程中出现数据丢失的情况，可以利用已接收到的宏块数据来对丢失的宏块数据进行恢复。

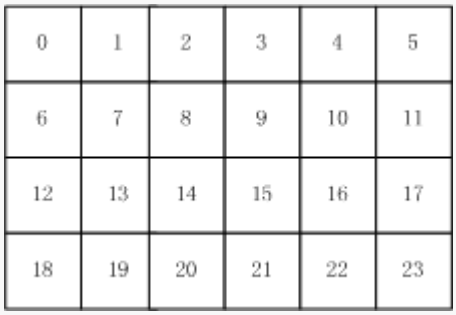

图3.23 可变宏块编码顺序

slice

slice是一个类似于H.263中图像组（GOP）的概念，一个slice是由一系列按光栅扫描顺序排列的宏块组成。一般情况下每个宏块均包含一个16×16 的亮度阵列，当视频格式不是单色时，还包含和两个相应的色度阵列。如果没有使用宏块自适应帧/场解码，每个宏块代表图像中的一个空间矩形区域。例如，如图3.22所示，一幅图像被分为两个条带。

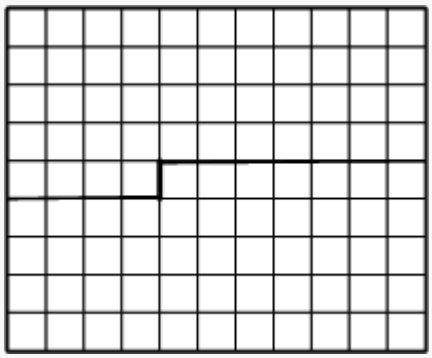

图3.24 slice对象

每个slice都是一个独立的编码单位，无论是帧间还是帧内编码都不能越界。冗余slice允许编码器在同一数据流中嵌入同一slice中宏块地一个或多个冗余表示。这种做法和传输层冗余技术，例如包复制等，关键区别是在冗余slice中宏块地冗余表示可以使用不同地编码参数编码。例如，首先要表示可以使用相对较低的量化系数以获得较低的图像质量，而在冗余表示中可以用相对较高的量化系数以减少比特数。当解码器正确接受到首要表示时，将冗余表示丢弃。而如果首要表示由于包丢失等原因无法正确获得，能够用冗余表示中地信息将相应slice数据恢复。冗余slice 最初是为支持高差错无线通信环境而引入的，但在基于IP的环境中同样有效。

通过块匹配估计运动的方法

完全抵消所有运动的运动补偿器将产生非常好的预测帧，以至于实际上在差别图片中不会存在任何功率。我们需要相对较多的数据以详细描述运动，但是只需要相对教少的数据，以描述差别帧。无可否认，甚至使用艺术技术也不可能从一般的帧源中识别和测量任何对象的运动。我们不得不满足于简化图片模型，例如经常使用的块匹配技术。除了次优的运动补偿之外，差别图片所需的数据速率比没有运动补偿所需的速率要小很多。进一步而言，我们的优势是特别简单，因而节省描述运动所需的位数。这在部分程度哂纳感弥补了差别图片的信号功率的不足，这种信号没有完全最小化。

使用块匹配技术的运动估计器

在数据压缩中，块匹配运动估计器可以任意处理每个新帧，使其用大小相同的直接相邻的对象进行传送。另外，对象仅仅能在2维平面上在一个方向上统一地移动。因而，被传输的帧被分割为一系列矩形图案块，它们是连续产生的。运动预测器假设图案块仅仅能在x和y方向上移动一个最大值。对于每个图案块，存在一个搜索区域，根据基本模型，在先前帧的这个区域内可以找到那个图案块。在使用等长步长的情况下，图案块逐渐移动通过搜索区域内的连续位置，并且每个位置都和旧图片进行比较。
位置变换也称为位移，如果某个位移达到了最佳的相似性或匹配结果，则它称为搜索后运动。然后，运动补偿帧的块将填充属于先前帧的块的内容，这将和前面搜索的图案块产生最佳的匹配。通过这种方式，运动补偿帧可以和瞬态帧尽可能地接近。

位移中的x和y成分通过侧向通道而传送到接受器，目的是可以从旧帧中构造运动补偿帧。对先前帧的内容执行这个操作，从而对已知图片进行这个操作，这就是这种编码技术的本质优点。

向量的数据速率取决于查找区域的带，从而取决于最大的位移，以及期望的向量的精确程度。对象的轮廓没有必要传送，原因是所有的对象具有相同的矩形。

P图像的VLC编码

VLC是可变长编码，VLC是统计编码技术，它的基本思想是：对出现频率较高的数值分配比特数较少的码字，而对出现频率较低的数值分配比特数较多的码字，因此从总的效果看，数据量比用均匀分配比特数的数据量要少。可变长编码是对Huffman编码的改进

P图像是参考过去的帧内图像或者过去预测得到得图像用运动补偿预测技术进行编码，P图像得编码也是以图像宏块为基本编码单元。预测编码得 基础是运动估值，它将直接影响到整个系统得编码效率和压缩性能，因此希望找到一种预测精度高同时计算量又小得运动估值算法。

正如I画面一样，每一幅P画面被分为一片或多片，每一片又被划分为若干宏块。对P画面的编码要比I画面复杂的多，因为要构造运动补偿宏块。运动补偿宏块与当前宏块的差值被一个二维的DCT变换为8x8的变换系数矩阵，这些系数在被量化成一组量化系数，最后，对量化后的系数采用行程长度技术编码。表3.11和3.12分别给出了P画面和B画面中所支持的宏块类型及VLC编码。

表3.11 P画面中的宏块类型及VLC编码

宏块类型	VLC码	INTRA	MOTION FORWARD	CODED PATTERN	QUANT
pred_mc	1	0	1	1	0
pred_c	01	0	0	1	0
pred_m	001	0	1	0	0
intra_d	0001 1	1	0	0	0
pred_mcq	0001 0	0	1	1	1
pred_cq	0000 1	0	0	1	1
intra_q	0000 01	1	0	0	1
skipped	无				

表3.12 B画面中的宏块类型及VLC编码

宏块类型	VLC码	INTRA	MOTION FORWARD	MOTION BACKWARD	CODED PATTERN	QUANT
pred_I	10	0	1	1	0	0
pred_ic	11	0	1	1	1	0
pred_b	010	0	0	1	0	0
pred_bc	011	0	0	1	1	0
pred_f	0010	0	1	0	0	0
pred_fc	0011	0	1	0	1	0
intra_d	00011	1	0	0	0	0
pred_icq	00010	0	1	1	0	1
pred_fcq	000011	0	1	0	0	1
pred_bcq	000010	1	0	1	1	1
intra_q	000001	1	0	0	0	1
skippde	无					

每一帧B画面被划分成一片或多片，每一片又被划分为若干宏块。由于要构造几种类型的运动补偿宏块：前向、后向、插播，所以对B画面的编码要比对P画面复杂的多。首先用一个二维DCT将运动补偿宏块与当前块之间的差值变换为8x8的变换系数矩阵，然后对着些系数进行量化，产生一组量化的系数，最后对这些量化后的系数用行程长度技术进行编码。

编码器不需要存储解码的B画面，因为B画面不用于运动补偿。

B画面宏块比P画面多了 几种类型，如果仅有前向运动矢量，则像P画面那样，从前面的一帧画面种构造运动补偿宏块。如果仅有后向运动矢量，则从后面的一帧画面种构造运动补偿宏块。如果既有前向也有后向运动矢量，则从前面以及后面的画面种构造运动补偿宏块，对结果求平均，用以形成插补宏块。

如同需要存储I画面一样，编码器也需要存储解了码的P画面，一位该P画面很可能会作为运动补偿的开始点。因此，编码器将要从量化系数种重构该画面的图像。
H.264所支持的帧编码模式如表3.13所示。

表3.13 帧编码模式

帧类型	描述	支持的框架
I(Intra)	只包含帧内预测的宏块(I)	全部
P(Predicted)	包含帧间预测宏块(P)和I型宏块	全部
B(Bi-Predictive)	包含帧间双向预测宏块(B)和I型宏块	扩展和主
SP(Switching P)	利于在编码的比特流中切换,包括I和P宏块	扩展
SI(Switching I)	利于在编码的比特流中切换,包含SI宏块(一种特殊的帧内编码宏块)	扩展