

H.264 视频编码原理

李睿远

Oct 29, 2026

在数字视频时代，原始视频数据所包含的巨大信息量对存储与传输构成了严峻挑战。未经压缩的 1080p 视频每秒可能超过 1.5 Gbit，这意味着即使是目前主流的千兆网络也难以流畅传输。视频编码正是通过去除空间冗余与时间冗余，将信息压缩至可接受的码率，同时保持人眼可接受的画质。H.264/AVC 正是这一领域的标志性成果，它于 2003 年正式发布，至今仍是互联网流媒体、数字电视与蓝光光盘的主流格式。

本文将沿着「分层结构—预测—变换—熵编码—网络」的路线，逐层剖析 H.264 的核心机制。阅读完本文，读者能够理解宏块划分、帧内/帧间预测、整数变换与 CABAC 等关键概念，并对后续标准 HEVC、VVC 产生直观认识。

1 H.264 核心概念与分层结构

H.264 把编码器输出划分为两层：视频编码层（VCL）负责像素级信号处理，网络抽象层（NAL）则将 VCL 数据封装为适合不同网络环境的单元。每个 NAL 单元以一个字节的头部开始，随后紧跟原始字节序列载荷（RBSP）。这种清晰的边界让路由器或中间件无需解码即可丢弃损坏单元，从而提升了流媒体的鲁棒性。

在像素层面，H.264 以 16×16 的宏块为基本处理单元，并可进一步划分为 8×8、4×4 子块。较小的块尺寸允许编码器捕捉画面中的细微纹理，但同时也增加了模式判决与运动矢量编码的开销。因此，编码器通常采用率失真优化（RDO）在编码质量与码率之间寻找平衡点。

视频序列被组织成不同的帧类型：I 帧仅使用帧内预测，可独立解码；P 帧利用先前参考帧进行单向运动补偿；B 帧则可双向参考，压缩效率最高；SI/SP 帧专为码流切换设计，可在不同码率之间实现无缝跳转。

2 帧内预测

当场景切换或运动剧烈时，帧间预测失效，帧内预测便成为唯一选择。H.264 为 4×4 亮度块定义了 9 种预测模式，包括垂直、水平、DC 及 6 种角度模式；16×16 块仅保留 4 种模式，以降低开销；8×8 模式则在 High Profile 中引入。色度块通常使用水平、垂直、DC 与平面四种模式。

预测残差通过如下公式计算：

$$\text{Residual}(x, y) = \text{Original}(x, y) - \text{Pred}(x, y)$$

残差随后进入 4×4 整数 DCT 变换。若当前宏块使用 16×16 预测，还需对 16 个 DC 系数再做 Hadamard 变换，以进一步集中能量。整数 DCT 的核心矩阵如下：

$$C = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

该矩阵仅含 ± 1 、 ± 2 ，便于硬件用移位与加法实现。变换后的系数再进行标量量化：

$$Z = \text{round} \left(\frac{Y \times \text{MF}}{2^{q\text{bits}}} \right)$$

其中 MF 为量化矩阵，qbits 与 QP 相关。量化步长随 QP 每增加 6 而翻倍，恰好对应 1 bit 动态范围。

3 帧间预测

帧间预测通过运动估计寻找当前块在参考帧中的最佳匹配位置。H.264 支持 1/4 像素精度，亮度分量需经过 6 抽头滤波器生成半像素，再线性插值得到 1/4 像素。色度则直接 1/8 精度线性插值。插值公式示例如下：

$$a = \text{clip} \left(\frac{-5A + 20B + 20C - 5D + 16}{32} \right)$$

其中 A、B、C、D 为整数像素。亚像素精度使运动矢量编码长度增加，但残差能量显著降低，总体码率仍可下降 20% 以上。

为利用时间相关性，H.264 允许最多 16 个参考帧。参考帧管理采用短期与长期列表，编码器通过滑动窗口或显式信令更新列表。B 帧可采用双向预测、加权预测或直接模式。直接模式下，运动矢量由相邻块或参考列表推导，无需额外编码。

4 变换与量化

H.264 将 4×4 残差块变换为频域系数后，采用 CAVLC 或 CABAC 编码。CAVLC 利用游程与幅值分离编码，码表根据相邻块非零系数自适应切换；CABAC 则利用上下文模型与二进制算术编码，在高码率场景下可节省 10% - 15% 码率。

量化参数 QP 取值 0 - 51，每增加 1 步长，量化步长提升约 12%。编码器可通过以下伪代码自适应调整 QP：

```

1 if (bits_budget < target * 0.9) {
    QP = max(QP - 1, 0);
3 } else if (bits_budget > target * 1.1) {
    QP = min(QP + 1, 51);
5 }

```

上述代码首先比较当前帧已编码比特与目标比特；若低于 90%，则降低 QP 以提升质量；若高于 110%，则提高 QP 以控制码率。实际编码器还会引入平滑因子，避免 QP 剧烈波动导致主观质量闪烁。

5 熵编码与码率控制

CAVLC 与 CABAC 的核心差异在于符号概率建模。CAVLC 采用固定码表与简单上下文，硬件实现简单；CABAC 通过上下文模型迭代更新概率区间，编码效率更高，但串行依赖性强，难以并行。CABAC 的核心操作可用如下伪代码描述：

```
1 while (range < 256) {  
    if (LPS < (range >> 1)) {  
3         bit = 0;  
        range = range - LPS;  
5    } else {  
        bit = 1;  
7        range = LPS;  
    }  
9    renormalize();  
}
```

其中 LPS 表示「最小概率符号」区间长度，renormalize 用于将 range 放大至 256 以上，保证算术精度。

码率控制遵循 HRD（假设参考解码器）与 VBV（视频缓冲验证器）模型。编码器维护一个虚拟缓冲区，模拟解码端缓冲状态。若缓冲区即将溢出，则强制提高 QP；若即将下溢，则降低 QP。VBV 缓冲区大小通常以「标准帧时间」为单位，例如 1 秒视频对应的比特数。

6 容错与网络友好性

无线或拥塞网络中，数据包可能丢失。H.264 通过切片组（FMO）将一帧划分为多个独立解码单元。即使某个切片丢失，相邻切片仍可提供空间相关信息用于错误掩盖。此外，冗余片（redundant slice）可在主片损坏时提供备份。

SP/SI 帧用于不同码率之间的无缝切换。SP 帧编码方式与 P 帧类似，但量化残差时参考另一 SP 帧的重构图像，从而保证切换点像素值一致。SI 帧则相当于 I 帧，可在任意位置插入，实现「快速频道切换」。

H.264 在压缩效率、容错能力与网络适配性之间取得了优异平衡，其 4×4 整数变换、1/4 像素运动补偿与 CABAC 熵编码至今仍是后续标准的基础。然而，面对 4K/8K 与 360° 视频，H.264 的块划分与预测模式显得力不从心。HEVC 通过 64×64 编码树单元与 35 种角度模式，将码率再降 50%；VVC 更进一步引入仿射运动模型与自适应分辨率变换，面向 8K 与沉浸式媒体。理解 H.264 的原理，有助于我们更好地把握视频编码的演进脉络。