

- 视频压缩简介
- 基于运动补偿的视频压缩
- 运动矢量搜索
 - H.261
 - H.263

多媒体基础——基本视频压缩技术（2021年春季）2

1 引言

- 为什么我们需要视频压缩？
 - 未压缩的视频数据可能极其庞大
 - 给网络通信带来一些问题
- 高清电视 (HDTV)
 - 1920x1080
 - 每秒30帧 (全动态)
 - 每种三原色8位
 - 总计每秒1.5吉比特!
 - 每个电缆通道为 6MHz
 - 最大数据速率为 19.2Mb/sec
 - 包含音频和控制信号时降至 18Mb/sec ...
 - 压缩率必须为83:1!

多媒体基础——基本视频压缩技术（2021年春季）4

1. 视频压缩简介

舍弃一些信息

- 冗余的空间信息?
- 颜色信息?
- 冗余的时间信息?

多媒体基础——基本视频压缩技术（2021年春季）5

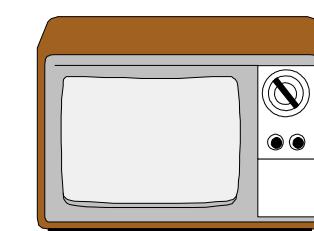
空间信息

- 欧洲广播电视标准



417行

- 分辨率降低
 - 至 352 (宽) × 288 (高) 像素
 - 源输入格式 (SIF)



288像素

多媒体基础——基本视频压缩技术（2021年春季）6

颜色

- 人类感知对亮度变化最为敏感
- 颜色的重要性较低，例如黑白照片仍可辨认
- RGB编码效率低——人类感知对较差的色彩也能接受。
- 使用YUV格式，仅对每个方向上半分辨率 (176×144, 即四分之一SIF) 的色度 (UV) 进行编码。与 Y 相比，这使得 U 和 V 的数据量仅为 Y 的 0.25 倍。

多媒体基础——基本视频压缩技术（2021年春季）7

舍弃色彩信息

- 舍弃所有这些信息后，我们的数据速率仍为（假设每个YUV分量为8位）：
 - $Y = (352 * 288) * 25 * 8 = 20.3\text{Mb/s}$
 - $U = (352/2 * 288/2) * 25 * 8 = 5.07 \text{ Mb/s}$
 - $V = (352/2 * 288/2) * 25 * 8 = 5.07\text{Mb/s}$
 - 总计（视频）= 30.45 兆比特/秒
 - MPEG 1 音频以 128Kb/s 运行
 - 视频光盘 - 目标是 1.5 兆比特/秒
 - 视频空间 = $1.5 - 0.128\text{Mb/s} = 1.372 \text{ 兆比特/秒}$
 - 所以现在使用压缩技术以实现 22:1 的节省率

多媒体基础——基本视频压缩技术（2021年春季）8

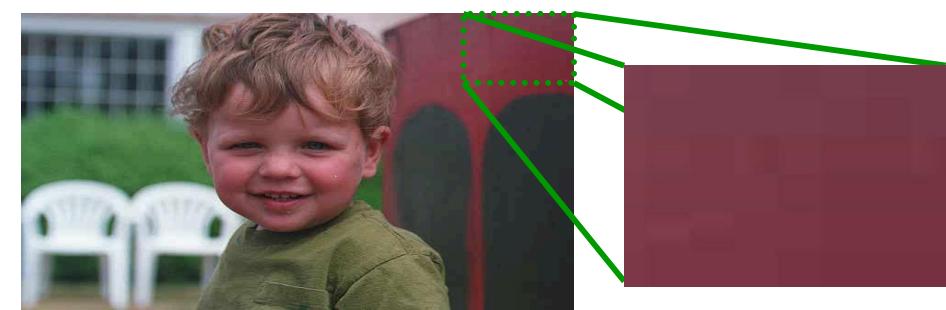
通过JPEG进行空间压缩

- 视频是一系列图像，而图像可以被压缩
- JPEG采用有损压缩，典型的压缩比为10:1至20:1
- 我们可以直接压缩图像并传输这些图像

1 引言

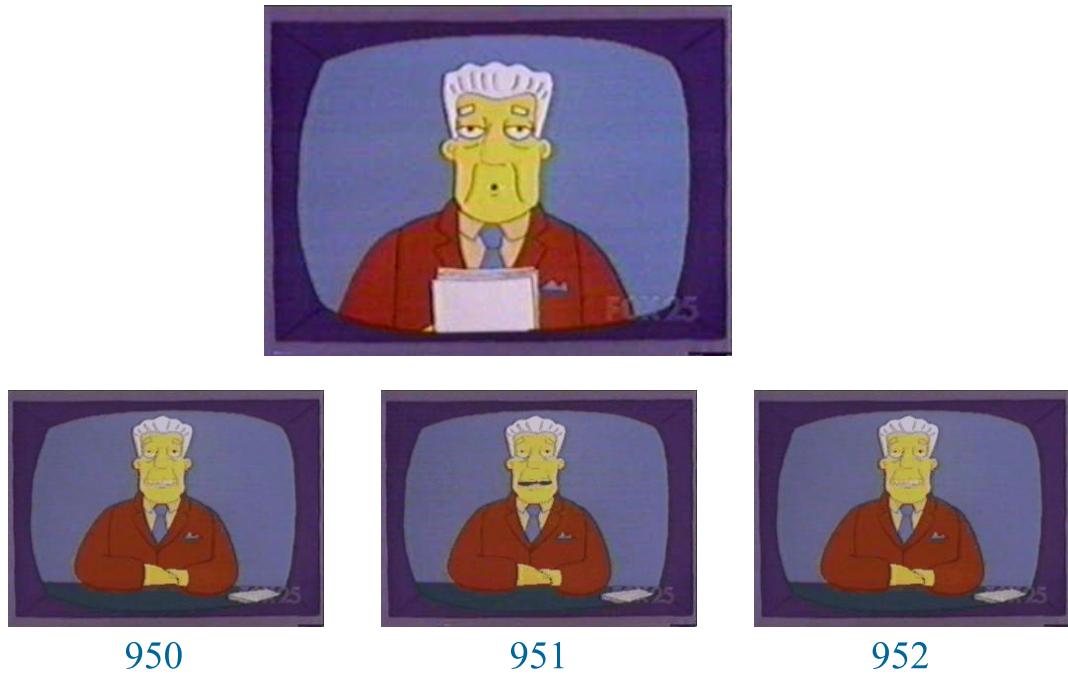
• 视频压缩的可行性

- 同一场景中的帧非常相似，因此视频数据存在时间冗余
- 即使是静态图像也能以高压缩比进行压缩，更不用说视频了



空间冗余

1 引言

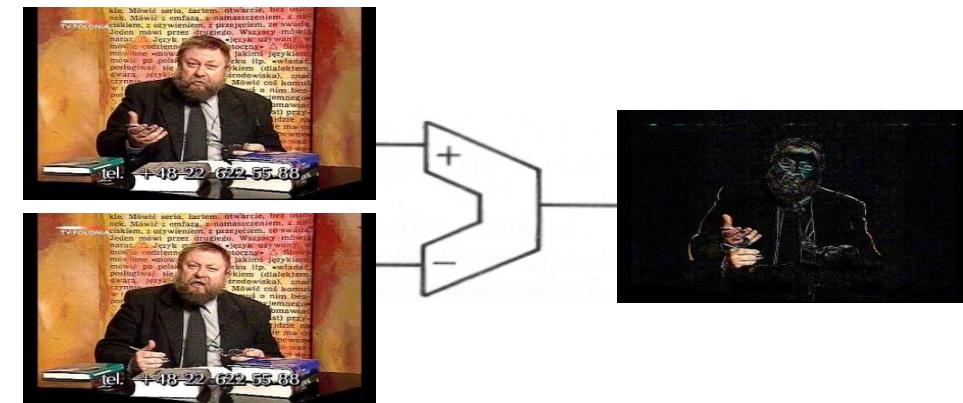


时间冗余

1 引言

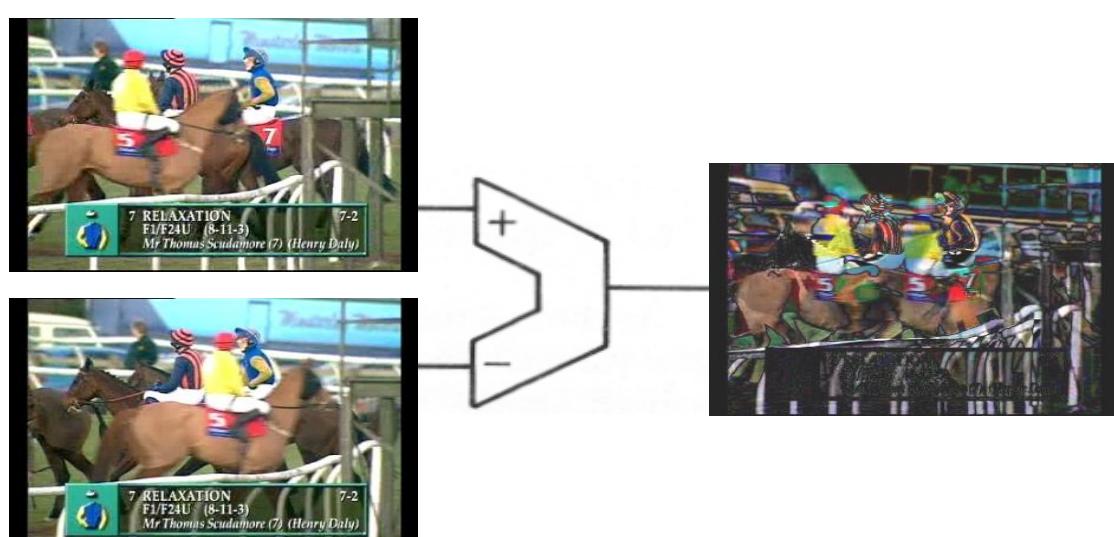
• 视频是在时间维度上堆叠的一系列图像

- 最简单的方法：预测编码
 - 按时间顺序相减图像
 - 对残差进行编码



1 引言

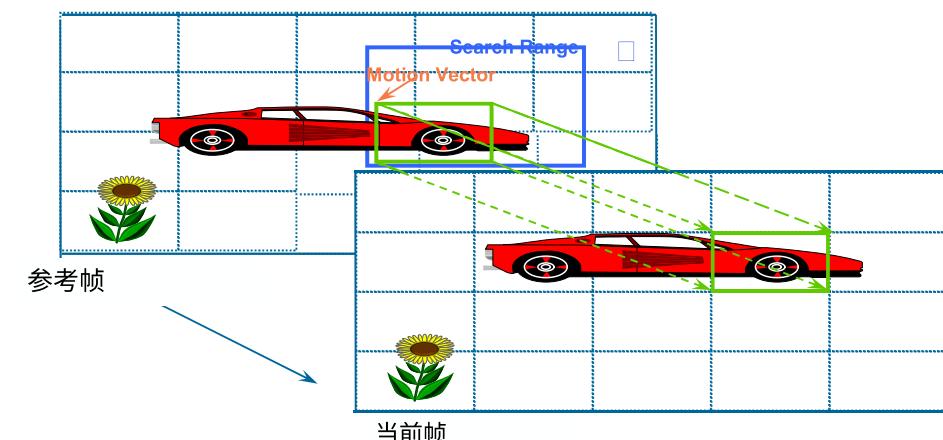
- 差分编码效果不错，但通常对象在帧与帧之间只是改变了位置。
- 对于“稀疏”差分图像，离散余弦变换（DCT）编码效果不如其他方法。



1 引言

• 更好的方法

- 寻找图像中合适的部分进行相减操作前一帧。
- 运动估计
- 运动补偿



2.1 时间冗余

• 视频：时间维度上的一系列图像

• 连续帧通常相似

- 视频具有显著的时间冗余
- 并非每一帧都独立编码
- 对相邻帧之间的差异进行编码

• 帧间差异的主要原因

- 相机或物体的运动

• 运动产生因素可以被补偿

- 检测相应像素的位移或区域
- 测量它们的差异（运动补偿MC）

2. 基于运动补偿的视频压缩

2.1 时间冗余——动态连续画面中同时存在空间冗余和时间冗余



- 动态画面编码原理：减少空间冗余和时间冗余——帧内编码：与JPEG类似——帧间编码：基于运动预测和补偿——P帧、B帧——多帧参考（H.264）

2.2 运动补偿

• 三个主要步骤

- 运动估计：运动矢量搜索
- 基于运动补偿的预测
- 预测误差的推导

- 每个图像被划分为大小为 $N \times N$ 的宏块。默认情况下，亮度图像的宏块大小为 $N = 16$ 。对于色度图像，如果采用4:2:0色度子采样，则宏块大小为 $N = 8$ 。

2.2 运动补偿

- 运动补偿在宏块级别执行。

- 当前图像帧称为目标帧。
- 在目标帧中的宏块与前一帧和/或未来帧（称为参考帧）中最相似的宏块之间进行匹配搜索。
- 参考宏块相对于目标宏块的位移称为运动矢量MV。
- 图10.1展示了前向预测的情况，其中参考帧为前一帧。

2.2 运动补偿

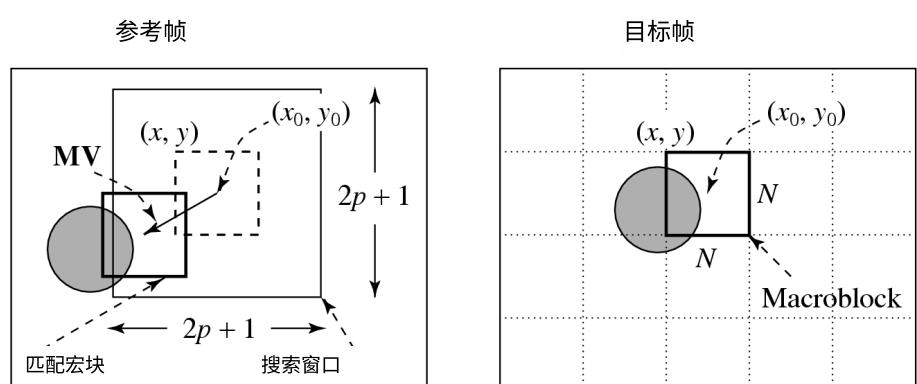
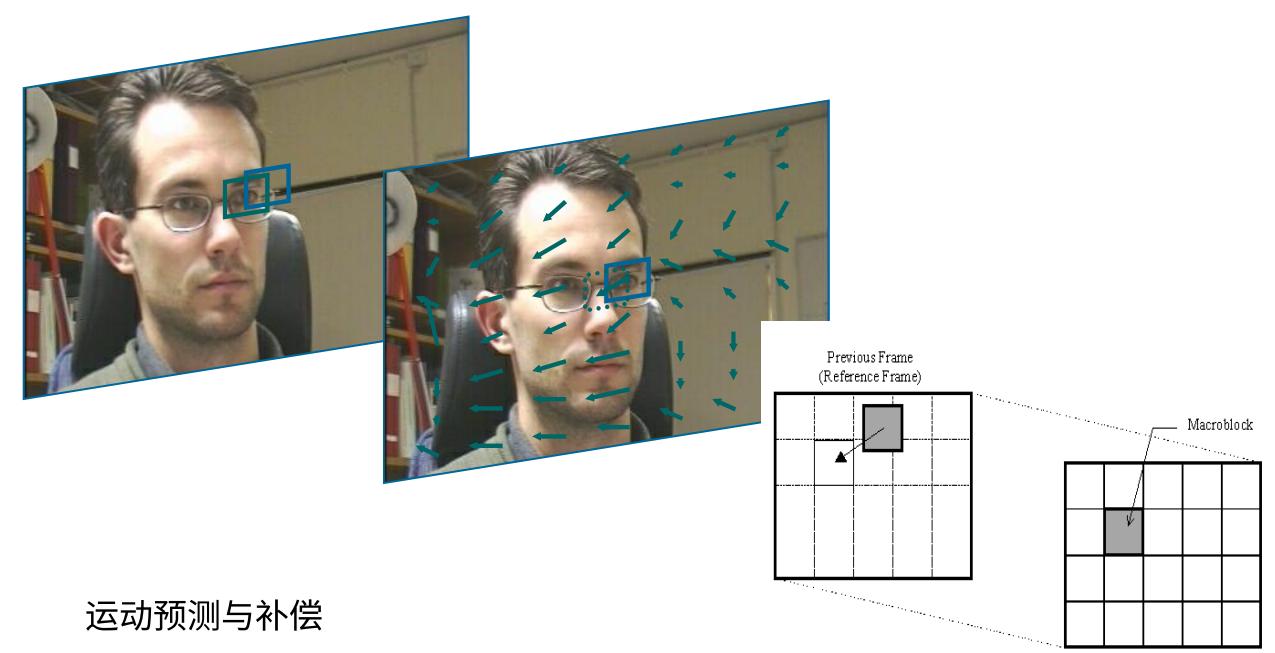


图10.1：视频压缩中的宏块和运动矢量。

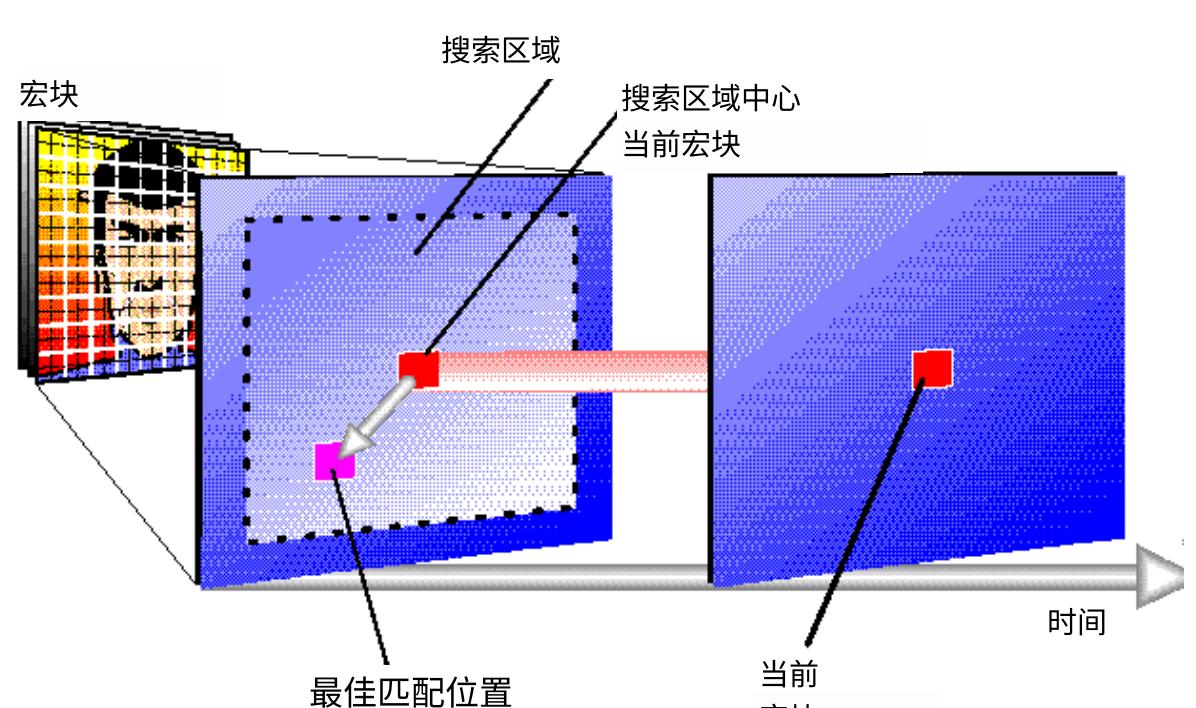
- 运动矢量搜索通常局限于一个较小的紧邻区域——水平和垂直位移范围均在 $[-p, p]$ 内。这形成了一个大小为 $(2p + 1) \times (2p + 1)$ 的搜索窗口。

2.2 运动补偿



运动预测与补偿

运动矢量搜索



3. 运动矢量搜索

3.1 匹配准则

- 运动矢量 (MV) 搜索：一个匹配问题，称为对应问题
- 水平和垂直位移 i, j 在范围 $[-p, p]$ 内，搜索窗口大小为 $(2p + 1) \times (2p + 1)$
- 目标：找到 (i, j) 使两个宏块之间的距离最小化

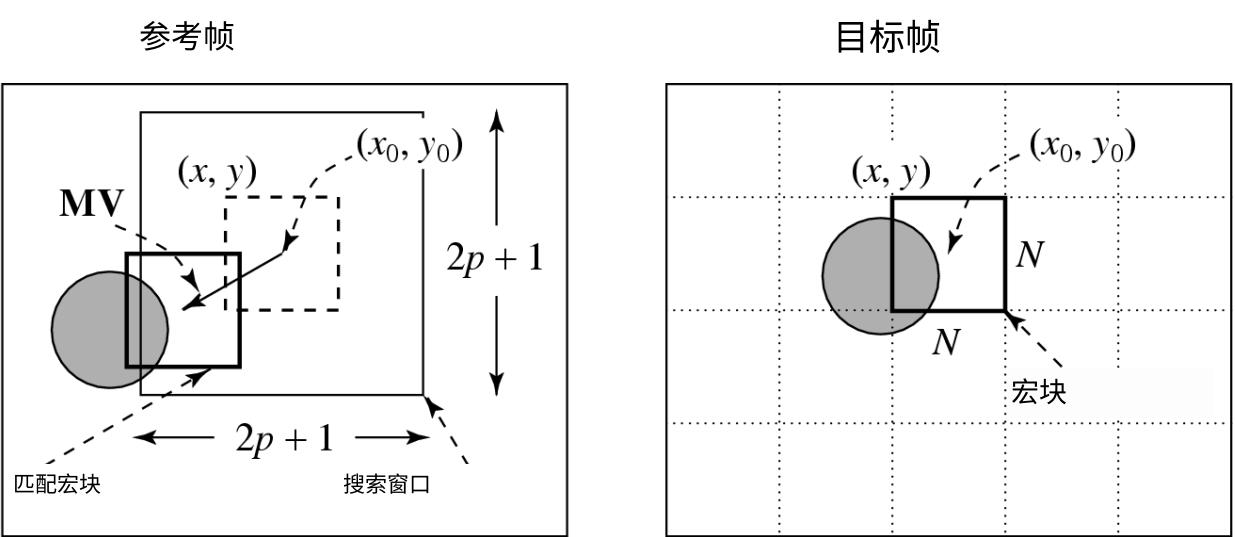
$C(x \square k, y \square l)$ ： t 目标帧宏块中的像素

$R(x \square i \square k, y \square j \square l)$ ：参考宏块中的像素

当运动矢量为 (i, j) 时

$$\text{MAD}(i, j) = \frac{1}{N^2} \sum_{k=0}^{N-1} |C(x \square k, y \square l) - R(x \square i \square k, y \square j \square l)|$$
$$(u, v) = (i, j) \mid \text{MAD}(i, j) \text{ is minimum}, i \in [-p, p], j \in [-p, p]$$

3.1 匹配准则



3.2 顺序搜索

- 顺序搜索：按顺序搜索参考帧中的整个 $(2p + 1) \times (2p + 1)$ 窗口（也称为全搜索）。
- 将窗口内每个位置为中心的宏块与目标帧中的宏块逐像素进行比较，然后使用公式(10.1)得出它们各自的平均绝对差 (MAD)。
- 提供最小 MAD 的向量 (i, j) 被指定为目标帧中宏块的运动向量 (MV) (u, v) 。
- 顺序搜索方法成本很高——假设每个像素比较需要三个操作（减法、绝对值加法），为单个宏块获取运动向量的成本是 $(2p + 1)(2p + 1)N^2 3 O(p^2 N^2)$ 。

3.2 顺序搜索

```
开始
min_MAD = 大数; /* 初始化 */
从 i = -p 到 p
  从 j = -p 到 p
  {
    当前 MAD = MAD(i, j);
    如果当前平均绝对偏差 (MAD) 小于最小平均绝对偏差 (min_MAD)
      最小平均绝对偏差 (min_MAD) = 当前平均绝对偏差 (MAD);
    }
  u = i; /* 获取MV的坐标。 *l
  v = j
}
结束
```

3.3 二维对数搜索

- 对数搜索：一种成本较低的版本，虽非最优但通常仍有效。
- 二维运动矢量对数搜索过程需要多次迭代，类似于二分搜索：
 - 如图10.2所示，最初仅将搜索窗口中的九个位置用作基于平均绝对差 (MAD) 搜索的种子；它们被标记为“1”。
 - 找到产生最小 MAD 的位置后，将新搜索区域的中心移至该位置，并将步长（“偏移量”）减小一半。
 - 在下一次迭代中，九个新位置被标记为“2”，依此类推。

3.3 二维对数搜索

```
开始
偏移量 = ;
在参考帧的搜索窗口内指定九个宏块，它们以  $(x_0, y_0)$  为
中心，水平和/或垂直方向上相隔偏移量;
当最后一个 ≠ 为真时
{
  找到九个指定宏块中使最小的一个；若偏移量 = 1，则最后一个为真;
  MAD;
  若偏移量 = 1，则最后一个为真;
  偏移量 = 偏移量/2;
  用新的偏移量和找到的新中心形成一个搜索区域;
}
end
```

3.3 二维对数搜索

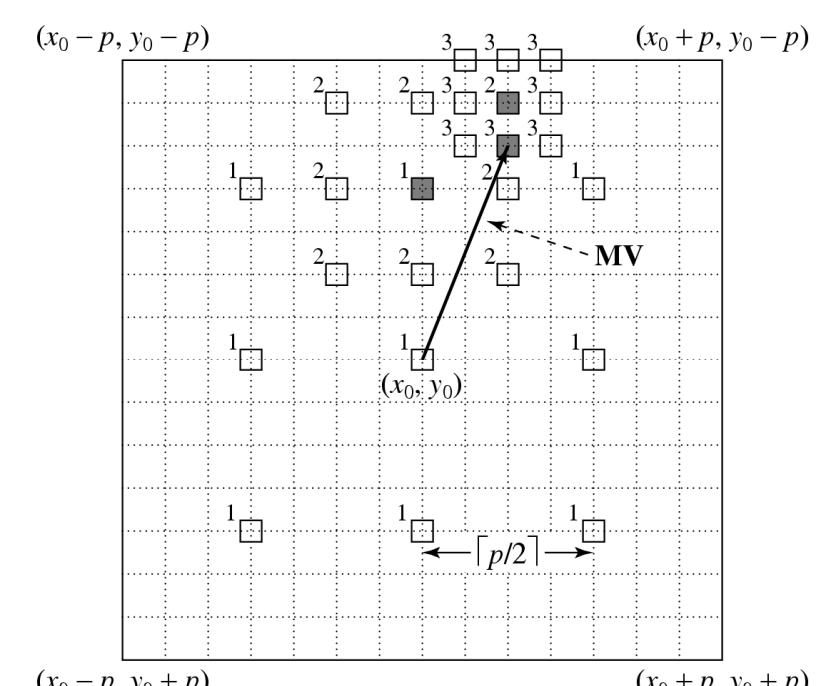


图10.2：运动矢量的二维对数搜索。

3.3 二维对数搜索

- 使用与上一小节相同的示例，每秒的总操作次数降至：

$$\begin{aligned} OPS_{\text{per-second}} &= (8 \cdot (\lceil \log_2 p \rceil + 1) + 1) \cdot N^2 \cdot 3 \cdot \frac{720 \times 480}{N \cdot N} \cdot 30 \\ &= (8 \cdot \lceil \log_2 15 \rceil + 9) \times 16^2 \times 3 \times \frac{720 \times 480}{16 \times 16} \times 30 \\ &\approx 1.25 \times 10^9 \end{aligned}$$

3.4 分层搜索

• 搜索可以采用分层（多分辨率）方法，在该方法中，可以从分辨率显著降低的图像中获得运动矢量的初始估计。

• 图10.3：三级分层搜索，其中原始图像位于第0级，第1级和第2级的图像是通过将前一级图像下采样2倍得到的，初始搜索在第2级进行。

由于宏块的尺寸较小，并且 p 也可以按比例减小，因此所需的操作数量大大减少。

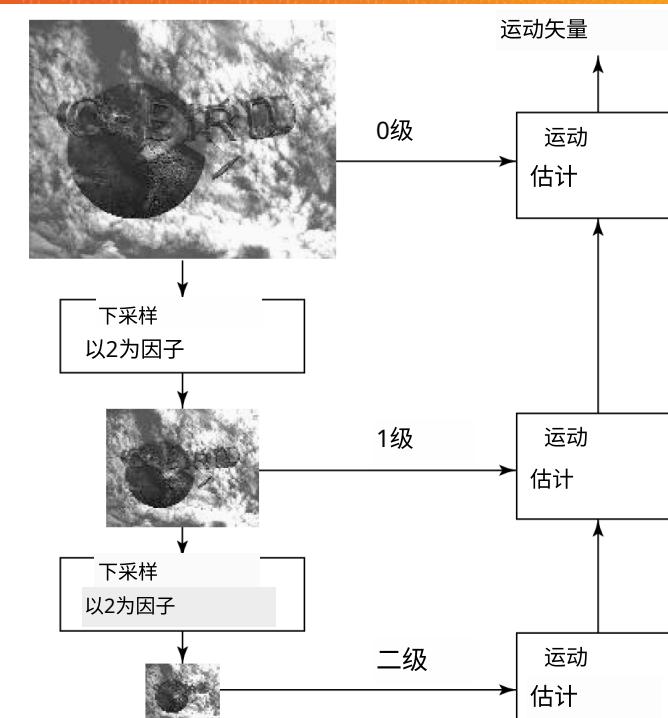


图10.3：运动矢量的三级分层搜索

3.4 分层搜索

• 给定第 k 层的估计运动矢量 (u^k, v^k) ，在第 $k - 1$ 层以 $(2 \cdot u^k, 2 \cdot v^k)$ 为中心的 3×3 邻域中搜索细化后的运动矢量。

• 细化过程使得在第 $k - 1$ 层，运动向量 (u^{k-1}, v^{k-1}) 满足：

$$(2u^k - 1 \leq u^{k-1} \leq 2u^k + 1, 2v^k - 1 \leq v^{k-1} \leq 2v^k + 1)$$

• 设 (x_0^k, y_0^k) 表示目标帧中第 k 层宏块的中心。目标帧中以 (x_0^k, y_0^k) 为圆心的宏块的分层运动向量搜索过程可概述如下：

过程 10.3 运动向量：分层搜索

```
开始
    // 获取最低分辨率第 k 层的宏块中心位置
     $x_0^k = x_0^0/2^k; y_0^k = y_0^0/2^k;$ 
    使用顺序（或二维对数）搜索方法在层级上获取初始估计值 MV  $(u^k, v^k)$ 
    k;
    当最后一个 ≠ 为真时
    {
        找到以层级 k - 1 为中心的九个宏块中产生最小 MAD 的一个
        at
         $(2(x_0^k + u^k) - 1 \leq x \leq 2(x_0^k + u^k) + 1; 2(y_0^k + v^k) - 1 \leq y \leq 2(y_0^k + v^k) + 1);$ 
        如果 k = 1 成立，则 last = 真;
        k = k - 1
        用新的中心位置和运动矢量 (MV) 赋值  $(x_0^k; y_0^k)$  和  $(u^k, v^k)$ ;
    }
结束
```

表10.1 基于示例的运动矢量搜索计算成本比较

搜索方法	30帧每秒时 720×480 的每秒操作数	
	$p = 15$	$p = 7$
顺序搜索	29.89×10^9	7.00×10^9
二维对数搜索	1.25×10^9	0.78×10^9
三级分层搜索	0.51×10^9	0.40×10^9

3.5 其他方法

• 运动矢量搜索是视频压缩的主要步骤之一，人们提出了许多方法来提高效率。

• 以下列出了一些方法：

- 三步搜索法：TSS
- 共轭方向搜索法：CDS
- 十字搜索算法：CSA
- 新型三步搜索法：NTSS
- 四步搜索法：FSS
- 菱形搜索算法：DS
- 自适应块匹配算法：ABMA

4.1 H.261概述

• H.261：一种早期的数字视频压缩标准（1990年制定），其基于运动补偿的压缩原理被所有后续视频压缩标准所保留。

- 该标准专为通过综合业务数字网（ISDN）的可视电话、视频会议和其他视听服务而设计
- 视频编解码器支持 $p \times 64$ 千比特每秒的比特率，其中 p 的范围是1到30（因此也被称为 $p^* 64$ ）。
- 要求视频编码器的延迟小于150毫秒，以便视频可用于实时双向视频会议。

4. H.261

4.1 H.261概述

表10.2 H.261支持的视频格式

视频格式	亮度图像分辨率	色度图像分辨率	比特率 (Mbps) (若为30帧/秒且未压缩)	H.261支持
QCIF	176 × 144	88 × 72	9.1	必需的
CIF	352 × 288	176 × 144	36.5	可选的

4.1 H.261概述

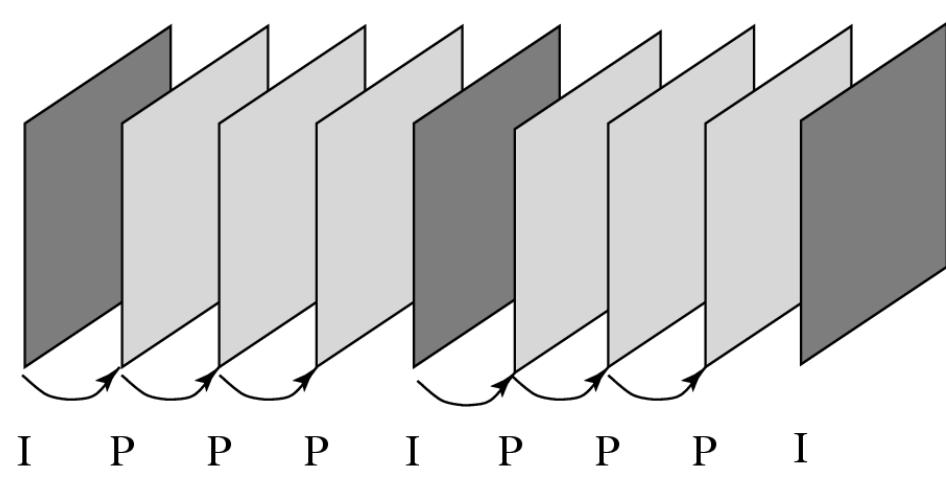


图10.4: H.261帧序列。

4.1 H.261概述

- 定义了两种类型的图像帧：帧内帧（I帧）和帧间帧（P帧）：
 - I帧被视为独立图像。每个I帧内应用类似于JPEG的变换编码方法，因此称为“帧内”。
 - P帧不是独立的：通过前向预测编码方法进行编码（允许从前一个P帧进行预测，而不仅仅是从前一个I帧）。
 - P帧编码中包含时间冗余去除，而I帧编码仅执行空间冗余去除。
 - 为避免编码错误的传播，通常在视频的每秒中发送几次I帧。
 - H.261中的运动矢量始终以整像素为单位进行测量，并且它们的范围有限，为±15像素，即 $p = 15$ 。

4.2 帧内编码

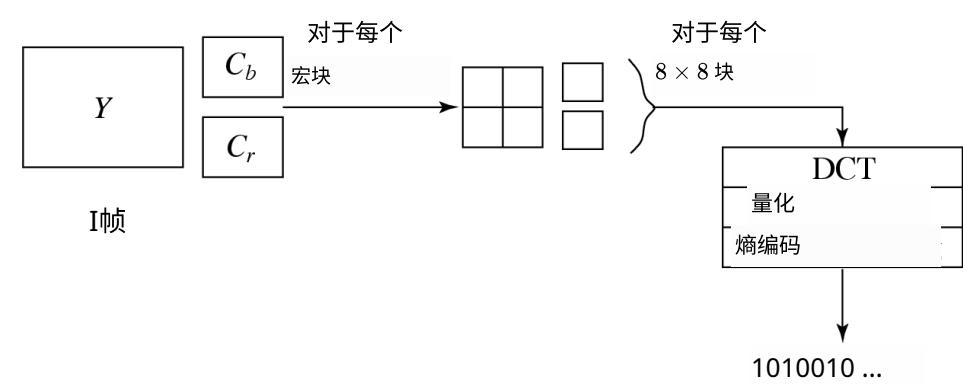
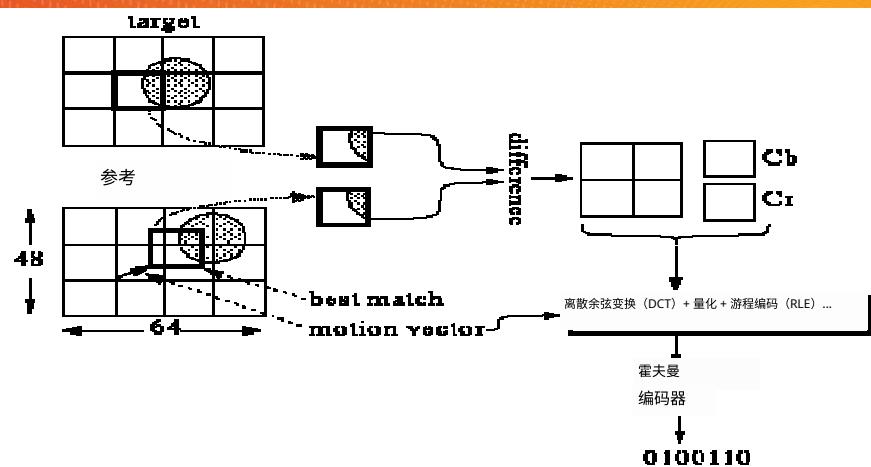


图10.5: I帧编码。

• 由于采用了4:2:0色度采样，对于Y帧，宏块大小为16×16像素，对于Cb和Cr帧，宏块大小为8×8。一个宏块由四个Y、一个Cb和一个Cr8×8块组成。

• 对每个8×8块应用离散余弦变换（DCT），然后DCT系数经过量化、之字形扫描和熵编码。

4.3 帧间预测编码



- 对于目标帧中的每个宏块，通过前面讨论的搜索方法之一分配一个运动矢量。
- 预测完成后，会导出一个差值宏块来衡量预测误差。
- 这些8×8块中的每一个都要经过离散余弦变换（DCT）、量化、之字形扫描和熵编码过程。

4.3 帧间预测编码

- P帧编码对差值宏块（而非目标宏块本身）进行编码。
- 有时，无法找到良好的匹配，即预测误差超过了某个可接受的水平。
 - 然后对宏块本身进行编码（将其视为帧内宏块），在这种情况下，它被称为非运动补偿宏块。

- 对于运动矢量，差值MVD被发送进行熵编码：

$$MVD = MV_{\text{Preceding}} - MV_{\text{Current}} \quad (10.3)$$

4.4 H.261中的量化

- 量化对一个宏块中的所有DCT系数使用一个常量（步长），步长是2到62之间的偶数之一。
- 在帧内模式下，直流系数始终使用步长=8。

$$QDCT = \left\lceil \frac{DCT}{step_size} \right\rceil \left\lceil \frac{DCT}{8} \right\rceil$$

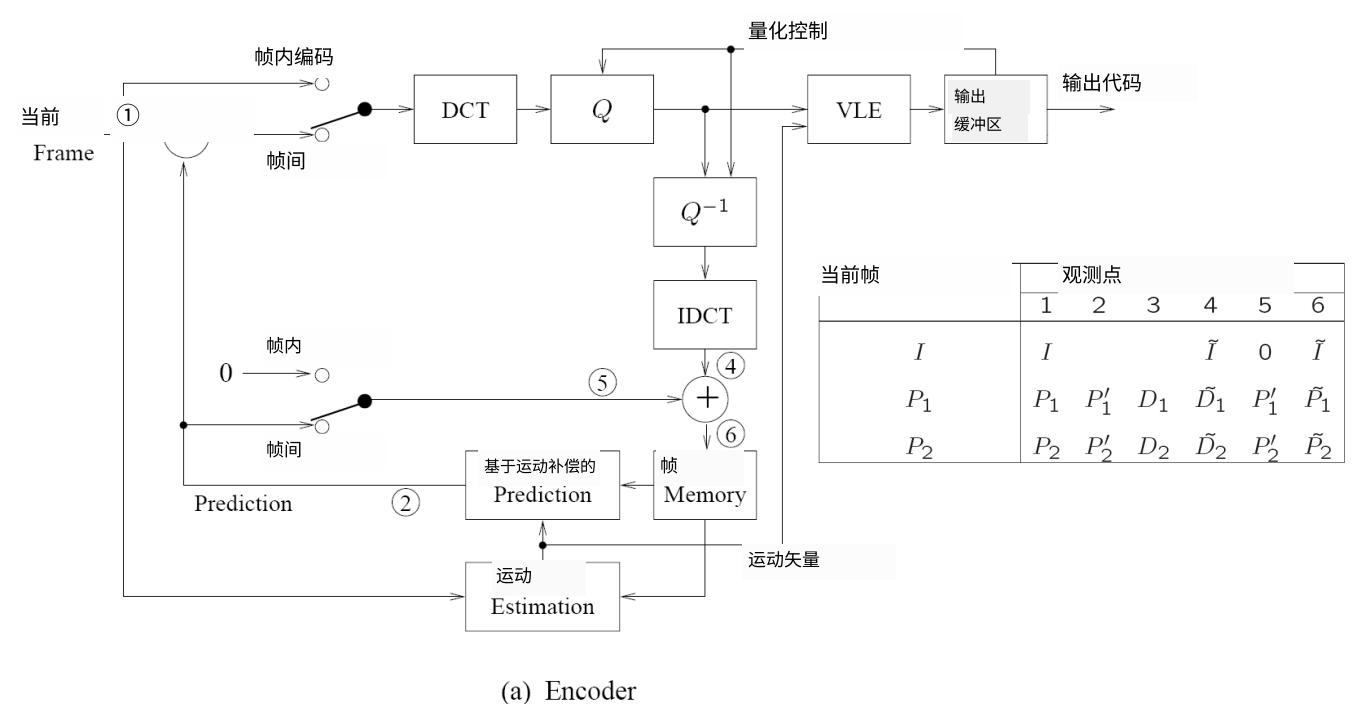
- 对于所有其他系数

$$QDCT = \left\lceil \frac{DCT}{step_size} \right\rceil \left\lceil \frac{DCT}{2 \cdot scale} \right\rceil$$

• 缩放因子是一个范围在[1, 31]内的整数

4.5 H.261编码器和解码器

编码器与数据流

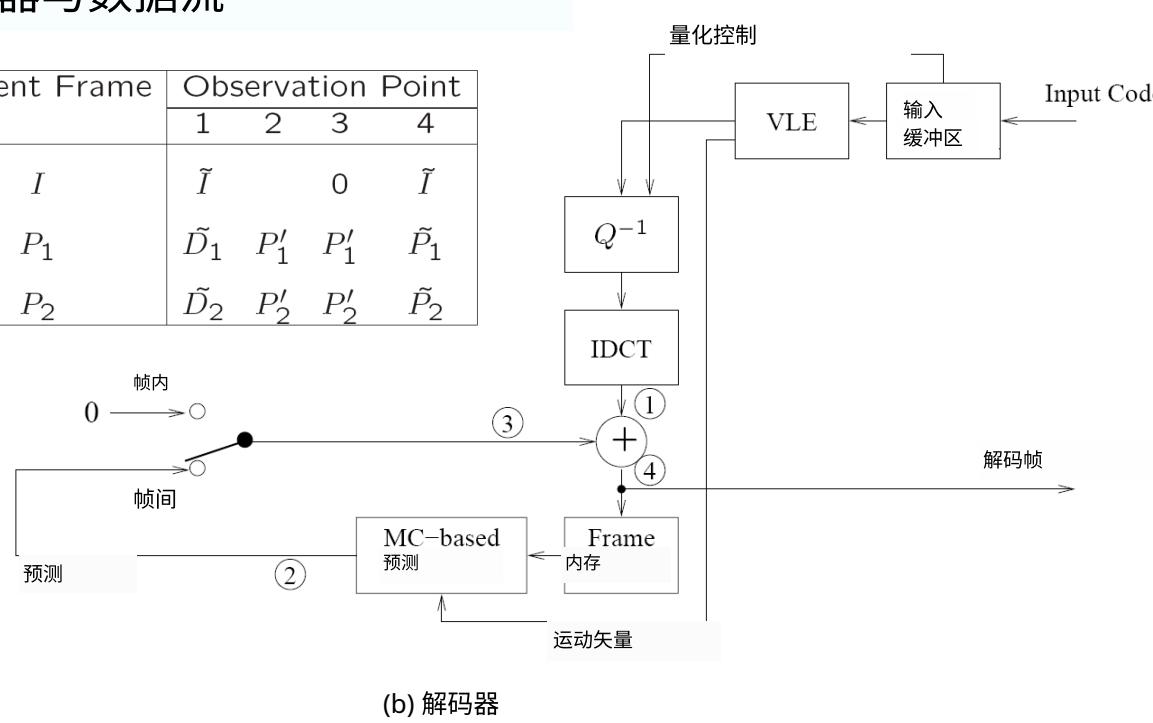


多媒体基础——基本视频压缩技术（2021年春季）

4.5 H.261编码器和解码器

解码器与数据流

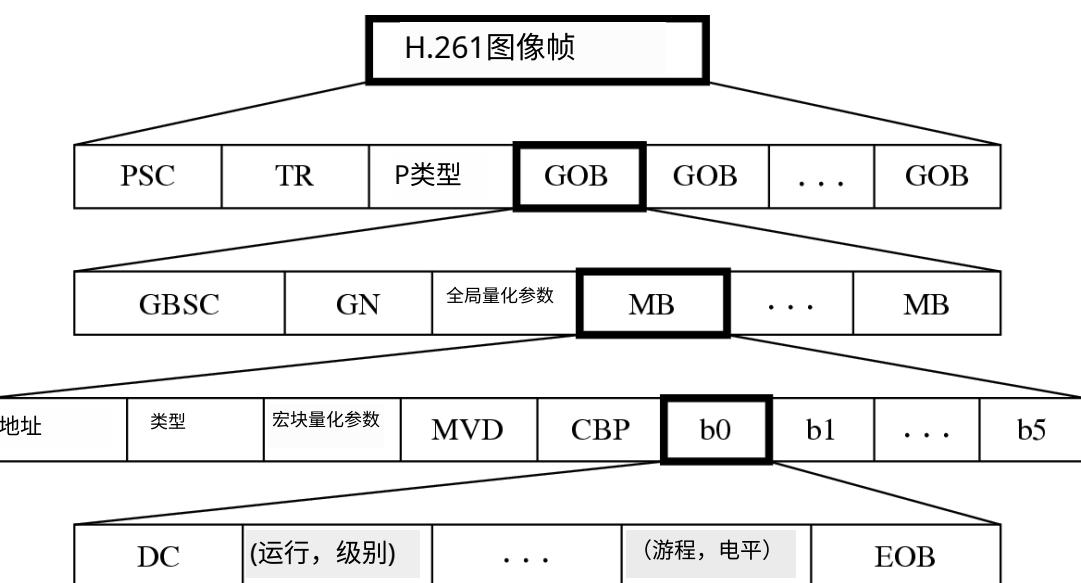
Current Frame	Observation Point			
	1	2	3	4
I	\tilde{I}	0	\tilde{I}	
P_1	$D\tilde{I}_1$	P'_1	P'_1	\tilde{P}_1
P_2	$D\tilde{I}_2$	P'_2	P'_2	\tilde{P}_2



4.6 H.261视频比特流语法

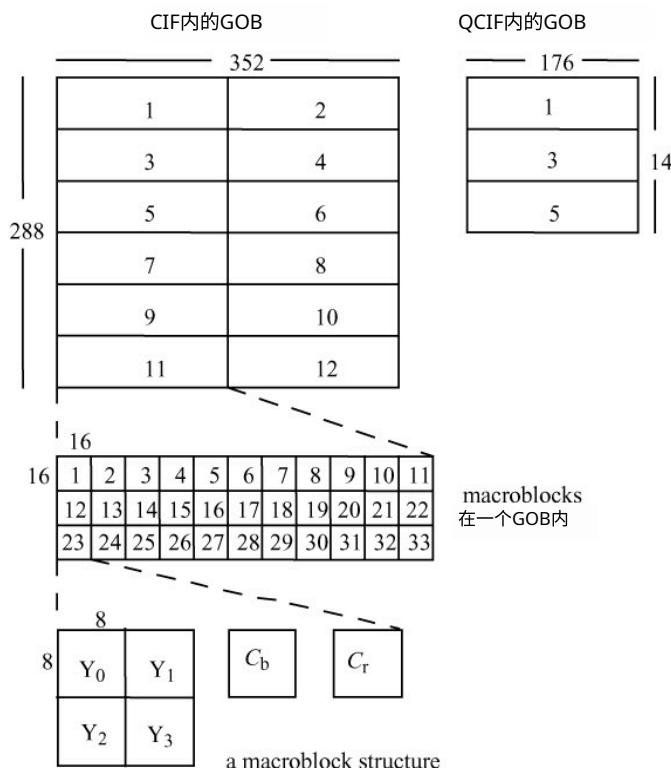
- H.261视频比特流语法：四层

- 图像、块组、宏块和块



4.6 H.261视频比特流语法

- 在H.261中，帧（图片）是最高层



4.6 H.261视频比特流语法

- 图片层

- PSC (图片起始码) 划定图片之间的边界。TR (时间参考) 为图片提供时间戳。



4.6 H.261视频比特流语法

- GOB层: H.261图像被划分为 11×3 个宏块区域，每个区域称为一个块组 (GOB)。

- 例如，CIF图像有 2×6 GOBs，对应其 352×288 像素的图像分辨率。每个GOB都有其起始码 (GBSC) 和组号 (GN)。
- 如果网络错误导致比特错误或某些比特丢失，H.261视频可以在下一个可识别的GOB处恢复并重新同步。
- GQuant表示GOB中要使用的量化器，除非它被任何后续的MQuant (宏块量化器) 覆盖。GQuant和MQuant在公式 (10.5) 中称为比例。



4.6 H.261视频比特流语法

- 宏块层

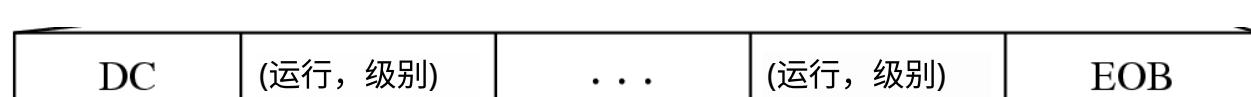
- 每个宏块 (MB) 都有自己的地址，用于指示其在 GOB中的位置、量化器 (MQuant) 和六个 8×8 图像块 ($4 Y, 1 Cb, 1 Cr$)。



4.6 H.261视频比特流语法

- 块层:

- 对于每个 8×8 块，比特流以直流 (DC) 值开始，接着是交流 (AC) 系数的零游程长度 (Run) 和后续非零值 (Level) 对，最后是块结束 (EOB) 码。Run 的范围是 $[0; 63]$ 。Level反映量化值，其范围是 $[-127, 127]$ 且 $Level \neq 0$ 。



4.6 H.261视频比特流语法

- H.261视频比特流的语法

- PSC (图像起始码)
- TR (时间参考)
- Ptype (图片类型)
- GOB (块组)
- GBSC (GOB起始码)
- Gquant (GOB量化器)
- GN (组编号)
- 宏块 (Macroblock)
- 宏块量化器 (MB Quantizer)
- 运动矢量数据 (Motion Vector Data)
- 编码块模式 (Coded Block Pattern)
- 块结束符 (End of Block)

5.1 H.263 概述

- 一种用于公共交换电话网 (PSTN) 上视频会议和其他视听服务的改进标准，于 1995 年被国际电信联盟电信标准化部门 (ITU - T) 第 15 研究组采纳
- 旨在实现比特率低于 64 kbps 的低比特率通信。

- H.263 支持子 QCIF、4CIF 和 16CIF，GOB 没有固定大小

5. H.263



视频格式	亮度图像分辨率	色度图像分辨率	比特率 (Mbps) (如果为 30 帧/秒且未压缩)	比特率 (kbps) BPP 最大 Kb (压缩后)
次 QCIF	128 × 96	64 × 48	4.4	64
QCIF	176 × 144	88 × 72	9.1	64
CIF	352 × 288	176 × 144	36.5	256
4CIF	704 × 576	352 × 288	146.0	512
16CIF	1,408 × 1,152	704 × 576	583.9	1024

Fundamentals of Multimedia — Basic Video Compression Techniques (2021 Spring)

58

5.2 块组 (GOB)

- 与 H.261 一样，H.263 标准也支持块组 (GOB) 的概念。
- 不同之处在于，H.263 中的 GOB 没有固定大小，并且它们总是在图像的左右边界开始和结束。
- 如图 10.10 所示，每个 QCIF 亮度图像由 9 个 GOBs 组成，每个 GOB 有 11×1 MBs (176×16 像素)，而每个 4CIF 亮度图像由 18 个 GOB 组成，每个 GOB 有 44×2 MBs (704×32 像素)。

Fundamentals of Multimedia — Basic Video Compression Techniques (2021 Spring)

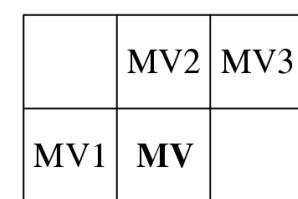
59

5.3 H.263 中的运动补偿

- MV 的水平和垂直分量分别根据来自“前一个”、“上方”和“上方右侧”宏块的 MV1、MV2、MV3 的水平和垂直分量的中值进行预测（见图 10.11 (a)）。

- 对于具有 $MV(u, v)$ 的宏块：

$$u_p = median(u_1, u_2, u_3), \\ v_p = median(v_1, v_2, v_3).$$



不直接对 $MV(u, v)$ 本身进行编码，而是对误差向量($\delta u, \delta v$)进行编码，其中 $\delta u = u - u_p$ 且 $\delta v = v - v_p$

Fundamentals of Multimedia — Basic Video Compression Techniques (2021 Spring)

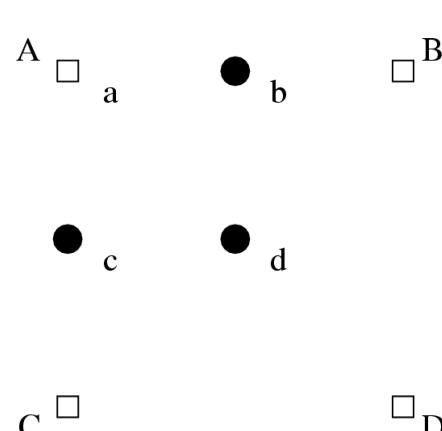
61

5.3 H.263 中的运动补偿

- 为了减少预测误差，H.263 支持半像素精度，而 H.261 仅支持全像素精度。

- $MV(u, v)$ 的水平和垂直分量 u 和 v 的默认范围现在是 [-16, 15.5]。

- 半像素位置所需的像素值通过简单的双线性插值方法生成，如图 10.12 所示。



□ 全像素位置

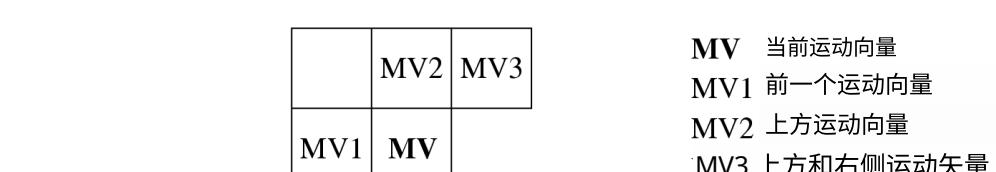
● 半像素位置

$$a = A$$

$$b = (A + B + 1) / 2$$

$$c = (A + C + 1) / 2$$

$$d = (A + B + C + D + 2) / 4$$



(a)

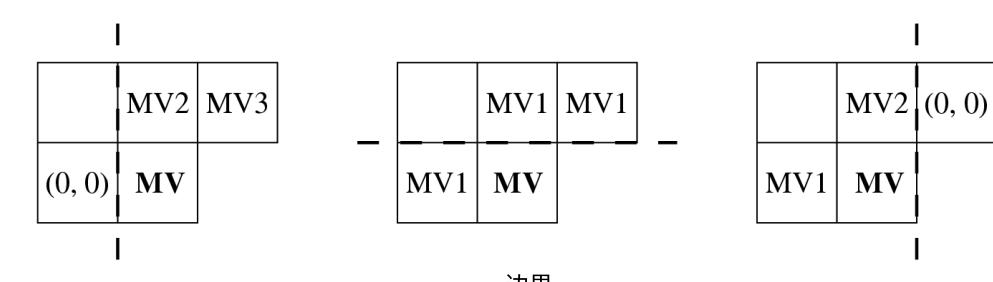


图 10.11 H.263 中运动矢量的预测。

Fundamentals of Multimedia — Basic Video Compression Techniques (2021 Spring)

62

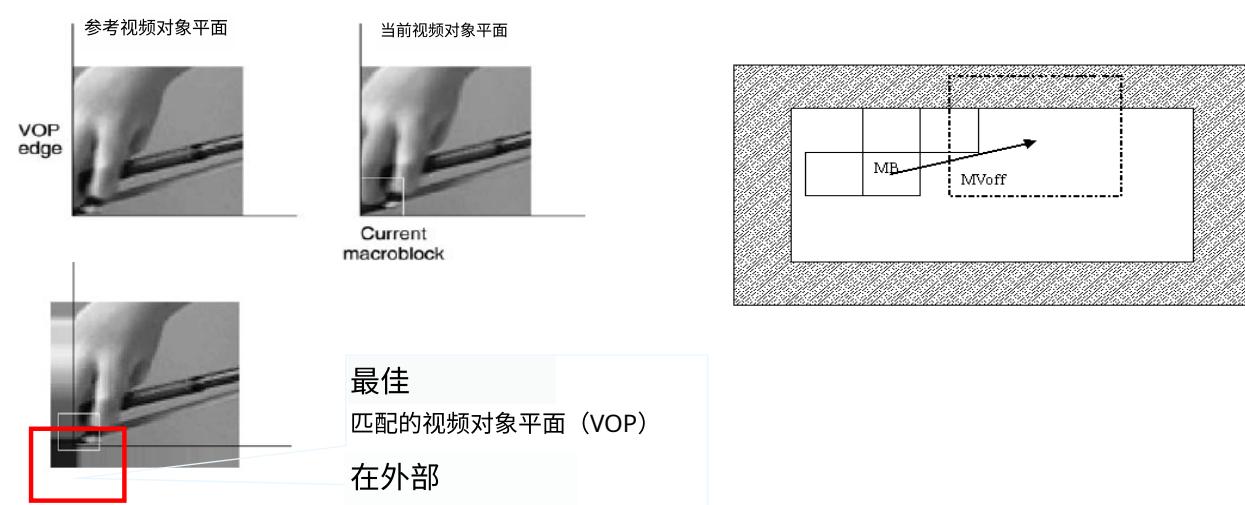
5.4 可选的 H.263 编码模式

- 除核心算法外的可协商选项：

- 无限制运动矢量模式。
- 基于语法的算术编码模式
- 高级预测模式（宏块的 4MV）
- PB 帧模式

5.4 可选的H.263编码模式

- 无限制运动矢量
- 参考不受图像边界限制
- 通过将编码图像边缘扩大一定尺寸（例如宏块大小）来实现



5.4 可选的H.263编码模式

2. 基于语法的算术编码模式：

- 与H.261一样，H.263默认使用可变长度编码（VLC）作为离散余弦变换（DCT）系数的编码方法。
- 与H.261类似，H.263的语法也采用四层分级结构。每层编码都结合使用了定长码和变长码。

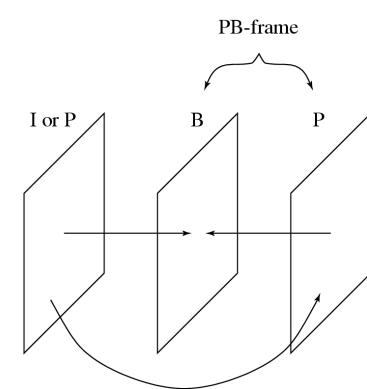
3. 高级预测模式：

- 在此模式下，运动补偿（MC）的宏块大小从 16×16 减小到 8×8 。
- 对于亮度图像中的每个宏块，会生成四个运动矢量（来自每个 8×8 块）。

5.4 H.263可选编码模式

4. PB帧模式：

- 在H.263中，PB帧由作为一个单元进行编码的两幅图像组成，如图10.13所示。
- PB帧模式的使用在PTYPE中指明。
- PB帧模式对于中等运动的视频能产生令人满意的效果。
- 在大幅度运动情况下，PB帧的压缩效果不如B帧，并且H.263第2版中开发了一种改进的新模式。



5.5 H.263+和H.263++

•H.263的第二个版本

- 重新定义无限制运动矢量模式
- 使用片结构替代GOB
- 实现时间、信噪比和空间可扩展性
- 支持改进的PB帧模式
- 使用去块滤波器减少块效应

•H.263++：新扩展

- 增强参考图像选择（ERPS）
- 数据分割片（DPS）
- 额外的补充增强信息

结束

谢谢！
邮箱：junx@cs.zju.edu.cn