

# 第八章 图像压缩

本章主要参考盛斌副教授的幻灯片，少量参考书中内容。

## 8.1 图像压缩基本概念

**数据压缩：**减少表示给定信息量所需数据量的处理。数据是信息传递的手段，因为不同数量的数据可以用来表示相同数量的信息，包含不相关或重复信息的表示被认为包含冗余数据，若令  $b$  与  $b'$  表示代表相同信息的两种表示中的比特数（或信息携带单元），则用  $b$  比特表示的相对数据冗余  $R$  是：

$$R = 1 - \frac{1}{C}$$

其中， $C$  通常称为压缩率，定义为  $C = b/b'$ 。如果  $C = 10$  或写作  $10 : 1$ ，则较大表示有 10 比特数据，对于较少表示中的每 1 比特数据，较大表示中对应的相对数据冗余为 0.9，即 90% 的数据是冗余的。

**数据冗余：**图像数据中存在着三种数据冗余：编码冗余、空间和时间冗余（像素冗余）和不相关的信息（视觉心理冗余）。

- **编码冗余：**如果一个图像的灰度级编码，使用了多于实际需要的编码符号，就称该图像包含了编码冗余。在多数二维灰度阵列中，用于表示灰度的 8 比特编码所包含的比特数，要比表示该灰度所需要的比特数多。
- **空间和时间冗余（像素冗余）（像素间相关）：**任何给定位置像素值，原理上均与相邻像素相关，都可以通过它的邻居预测到，单个像素携带的信息相对较少。对于一幅图像，大量单个像素对视觉的贡献是冗余的。这是建立在对邻居值的预测值上。在视频序列中，时间相关的像素（即类似于或取决于相邻帧像素的那些像素）也是重复信息。
- **不相关的信息（视觉心理冗余）：**多数二维灰度阵列中包含有一些被人类视觉系统忽略或与用途无关的信息。从未被利用的角度看，它是冗余的。如人眼对区域亮度的感觉不仅取决于该区域的反射光，而且取决于周围的环境；视觉残留现象；人眼的亮度适应能力。

我们可以利用图像数据的冗余开展压缩。因为有编码冗余和像素冗余，当我们把图像信息的描述方式改变之后，可以压缩掉这些冗余，进行“无损”压缩；因为有主观视觉冗余，当我们忽略一些视觉不太明显的微小差异，可以进行“有损”压缩。

**客观保真度准则：**如果信息损失的程度，可以表示为压缩处理的输入和输出的数学函数时，则称其是以客观保真度准则为基础的。一个例子是两幅图像的均方根误差。令  $\hat{f}(x, y)$  为  $f(x, y)$  的近似，则误差  $e(x, y) = \hat{f}(x, y) - f(x, y)$ ；于是可以得到均方根误差

$$e_{rms} = \sqrt{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

均方信噪比为：

$$SNR_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}^2(x, y)}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

在实际应用中，进一步采用更为简便的客观准则，信噪比 SNR 和峰值信噪比 PSNR 定义如下：

$$SNR = -10 \log_{10} \frac{\sigma^2}{D}, \quad PSNR = -10 \log_{10} \frac{255}{D}$$

式中,

$$\sigma^2 = \frac{1}{N} \sum_{i=0}^{N-1} (x_i - \bar{x})^2, \quad D = \frac{1}{N} \sum_{i=0}^{N-1} (x_i - \hat{x}_i)^2, \quad \bar{x} = \frac{1}{N} \sum_{i=0}^{N-1} x_i$$

**主观保真度准则:** 通过视觉比较两个图像, 给出一个定型的评价, 如粗、很粗、较粗、相同、较好、很好等, 这种评价被称为**主观保真度准则**。

## 8.2 图像压缩模型

**图像压缩系统的一般构成:** 信源编码, 信道编码, 信道, 信道解码, 信源解码。

- 信源编码: 完成原始数据的编码与压缩;
- 信道编码: 为抗干扰, 增加一些容错、校验位, 实际上是有规律地增加传输数据的冗余, 以便于消除传输过程中加入的随机信号;
- 信道: 传送数据(信息)的手段。

**信源编码模型:** 信源依次输入映射器、量化器、符号编码器得到编码结果;

- 映射器: 减少像素冗余(空间和时间冗余), 如使用字典编码, 或进行图像变换;
- 量化器: 减少视觉心理冗余(不相关的信息), 仅用于有损压缩;
- 符号编码器: 减少编码冗余, 如使用哈夫曼编码。

**信源解码模型:** 编码后的信源头依次输入符号解码器、反向映射器得到原始信源。

## 8.3 图像压缩中的信息论观点

### 信息论

- (狭义) 关于通讯技术的理论, 以数学方法研究通讯技术中关于信息的传输和变换规律的一门科学。
- (广义) 以各种系统、各门科学中的信息为对象, 广泛地研究信息的本质和特点, 以及信息的获取、计量、传输、储存、处理、控制和利用的一般规律。
- 1948年, 香农发表文章——通讯中的数学原理, 奠定了信息基本理论的基础, 标志着狭义信息论的创立。文章中, 严格定义了信息的单位——“熵”的概念。在此基础上, 定义了信道容量的概念, 并给出在不同噪音情况下无失真通信的极限传输速率。
- **信息熵:** 表示一条信息中真正需要编码的信息量, 与通讯讯号复杂程度成正比例。

**信息的测量:** 信息的产生可以被模拟为一个概率过程, 假设某随机事件  $E$  出现概率为  $P(E)$ , 则其包含信息为

$$I(E) = \log \frac{1}{P(E)} = -\log P(E)$$

- $I(E)$  称为  $E$  的自信息, 如果  $P(E) = 1$ , 则  $I(E) = 0$ , 没有信息。
- 对数底数决定衡量信息的单位。底数为  $m$ , 称为  $m$  元单位, 底数为 2, 则信息的单元为比特。

**信源的模型化:** 信源被抽象为一个随机变量序列(随机过程)。

- 如果信源输出的随机变量取值于某一连续区间, 就叫做**连续信源**, 如语音信号  $X(t)$ ; 如果信源输

出的随机变量取值于某一离散符号集合，就叫做离散信源，如平面图像  $X(x, y)$ 。

- 如果随机序列中各个变量具有相同的概率分布，则称为离散平稳信源；
- 如果离散平稳信源的输出序列中各个变量是相互独立的，即前一个符号的出现不影响以后任何一个符号出现的概率，则称为离散无记忆平稳信源，否则称为离散有记忆平稳信源。

**离散信源：**假定信源输出的符号系列为  $\{a_1, a_2, \dots, a_j\}$ ，信源产生符号  $a_i$  概率为  $P(a_i)$ ，并有

$$\sum_{i=1}^j P(a_i) = 1$$

则该信源所有输出符号的集合

$$A = \{a_j\}$$

及其所有输出符号的概率的集合同量

$$z = [P(a_1), P(a_2), \dots, P(a_j)]^T$$

共同构成了信源的有限总体集合  $(A, z)$ ，该有限总体集合完全描述了信源。对于符号  $a_i$ ，其概率  $P(a_i)$ ，因此自信息为  $-\log P(a_i)$ ，于是信源输出平均信息为：

$$H(z) = -\sum_{i=1}^j P(a_i) \log P(a_i)$$

又称为信源的不定度或一阶熵。在符号出现之前，熵表示符号集中的符号出现的不确定性；在符号出现之后，熵代表接受一个符号所获得的平均信息量。信源的上反映了信源输出平均信息量的大小，符号概率分布越不均匀，信源的熵越小；符号概率平坦分布，即当所有符号等概率分布时，则熵最大。

**最大离散熵定理：**当与信源对应的字符集中的各个字符为等概率分布时，熵具有极大值  $\log_2 m$ ，其中  $m$  为字符集中字符个数。

- 应用：对于同一个信源，其总信息量不变，如果能够通过某种变换（编码）使信源尽量等概率分布，则每个输出符号所独立携带的信息量增大，那么传送相同信息量所需要的序列长度就越短。
- 离散无记忆信源的冗余度隐含在信源符号的非等概率分布之中。只要  $H(X)$  小于  $\log_2 m$ ，就存在数据压缩的可能。

**离散无记忆信源的编码：**如果对字符  $a_j$  的编码长度为  $L_j$ ，则  $X$  的平均码长为：

$$\bar{L} = \sum_{j=1}^m p_j L_j$$

于是有

$$\bar{L} = \sum_{j=1}^m p_j L_j \geq H(X) = -\sum_{j=1}^m p_j \log_2 p_j$$

于是， $L_j = -\log_2 p_j$ ，平均码长取得极小值。

**香农第一定理：**对信源输出  $A = \{a_j\}$ ，其信息熵为

$$H(z) = -\log_{i=1}^J P(a_i) \log P(a_i)$$

当信源输出符号具有统计上独立的特性时，这类信源编码所需的平均码字长度下限为其信息熵。

## 离散无记忆信源的其他结论

- 一阶熵即为离散无记忆平稳信源的压缩极限（基本极限）；
- 只要信源不是等概率分布，就存在着数据压缩的可能性；
- **数据压缩的基本途径之一：**使各字符的编码长度尽量等于字符的信息量。

**离散有记忆信源编码：**用条件熵  $H(X_n | X_1, X_2, \dots, X_{n-1})$  表示信息量，可以证明  $H(X|Y) \leq H(X)$ 。联合熵与其可能达到的最大值之间的差值反映了该有记忆信源所含有的冗余度，这种冗余是由于随机变量序列之间相关性造成的。

- 离散有记忆信源的压缩极限为  $\lim_{n \rightarrow \infty} H(X_n | X_1, X_2, \dots, X_{n-1})$ ；
- **数据压缩的基本途径之二：**尽量去除各分量之间的相关性，再对各分量进行独立编码；
- **数据压缩的基本途径之三：**可利用条件概率进行编码，阶越高越有利；
- **数据压缩的基本途径之四：**可将多个分量合并为向量，利用其联合概率进行编码，联合的分量越多越有利。

**信息系统模型：**将前述信源输出通过信道传输到接收端，信息接收者接收到符号集合  $B = \{b_k\}$ ，其概率  $P(b_k)$  与信源输出  $z$  的关系为

$$P(b_k) = \sum_{i=1}^J P(b_k | a_i) P(a_i)$$

信宿收到的可能的符号集合概率向量为  $v = [P(b_1), P(b_2), \dots, P(b_k)]$ ，则有限集合  $(B, v)$  完整地描述了用户所收到的信息。而  $P(b_k | a_i)$  反映了从信源到信宿之间的信息传递关系。考虑到所有符号的传递，可构成  $K \times J$  转移矩阵  $Q$ ，即**信道矩阵**。于是，我们形成了完整的信息系统模型（信源  $(A, z)$  → 信道  $Q$  → 信宿  $(B, v)$ ）。

## 8.4 图像压缩编码的基本方法：无损压缩

### 图像压缩编码的基本发展

- **第一代压缩编码：**八十年代之前，主要是根据传统理论进行的信源编码方法。包括像素编码（行程编码、算术编码、熵编码）、预测编码（增量调制、DPCM 调制）、变换编码（DCT 变换）、其他编码（位平面编码）。
- **第二代压缩编码：**八十年代之后，突破信源编码理论，结合分形、模型基、神经网络、小波变换等数学工具，充分利用视觉生理心理特性和图像信源的各种特性。包括子带编码、分层编码、分形编码、模型编码。

**数据压缩的分类：**数据压缩分为无损压缩与有损压缩；

- **无损压缩：**包括熵编码、无损预测编码、字典编码；
- **有损压缩：**包括有损预测编码、变换编码、小波编码。

**熵编码：**利用香农理论，从统计的角度，设法找到一种编码使得平均码长达到熵极限。一般对出现概率较大的符号取较短的码长，而对出现概率较小的符号取较长的码长。主要包括香农-范诺编码、霍夫曼编码、算术编码。

- **Huffman 编码**
  - 步骤：每次合并概率最小的两个事件，构造霍夫曼树。

- 存在的问题：对一幅图像进行编码时，如果图像尺寸大于 256，这幅图像的不同码字就有可能很大，例如极限为 256 个不同码字，这样会让小分布的灰度值具有很长的编码。可能达不到压缩的效果反而使数据量加大。

解决方法：将图像分割成若干小块，对每块进行独立 Huffman 编码。如分成  $8 \times 8$  子块降低不同灰度值的个数。

- 分类：**1** 静态 Huffman 编码：在压缩之前就建立好一个概率统计表和编码树，算法速度快但压缩效果不是最好；**2** 动态 Huffman 编码：对每一个图像临时建立统计表和编码树，算法速度慢但压缩效果较好。
- 局限性：利用 Huffman 编码，每个符号的编码长度只能为整数，如果源符号集的概率分布不是  $2^{-n}$  形式，则无法达到熵极限；输入符号数首先与可实现的码表尺寸；译码复杂；需要提前知道输入符号集的概率分布；没有错误保护功能。

### ● 香农-费诺编码

- 步骤：将编码字符集中的字符按照出现频度与概率进行排序；用递归的方法分成两部分，使两个部分的概率和接近于相等，直至不可再分，即每一个叶子对应一个字符；然后编码。
- 例：如图 61 所示。

### ● 算术编码

- 基本思想：考虑到 Huffman 编码的局限性是编码使用整数个二进制位对符号进行编码，这种方法在许多情况下无法得到最优压缩效果。我们不将单个信源符号映射成一个码字，而是把整个信源表示为实数  $[0, 1]$  上的一个区间，其长度等于该序列的概率。再在该区间内选择一个代表性的小区间，转化为二进制作为实际的编码输出；消息序列中每个元素都要用来缩短这个区间；消息序列中元素越多，所得到的区间就越小，当区间变小时，就需要更多位数来表示这一区间。
- 特点：采用算术编码，每个符号的平均编码长度可以为小数。解决了包括 Huffman 编码等其他编码方式中，每个符号的码长为整数的局限性。
- 局限性：算术编码器对整个消息只产生一个码字，这个码字是在间隔  $[0, 1)$  中的一个实数。因此，译码器在接收到表示这个实数的所有位之前不能进行译码；由于实际的计算机精度不可能无限长，运算中可能溢出；同时也是对错误敏感的方法，有一位发生错误就会导致整个消息译码错误。
- 例：如图 62~63 所示，假设 00, 01, 10, 11 出现概率分别为 0.1, 0.4, 0.2, 0.3，则将其分别编码在区间  $[0, 0.1)$ ,  $[0.1, 0.5)$ ,  $[0.5, 0.7)$ ,  $[0.7, 1)$  上。则编码过程如图 62 所示，解码过程如图 63 所示。

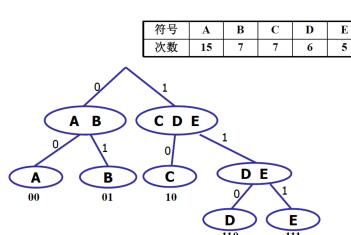


图 61

步 骤	输入 符 号	编 码 时 间 隔	编 码 判 决
1	10	[0.5, 0.7)	符号的间隔范围 [0.5, 0.7)
2	00	[0.5, 0.52)	[0.5, 0.7) 间隔的第一个 1/10
3	11	[0.514, 0.52)	[0.5, 0.52) 间隔的最后 3 个 1/10
4	00	[0.514, 0.5146)	[0.514, 0.52) 间隔的第一个 1/10
5	10	[0.5143, 0.51442)	[0.514, 0.5146) 间隔的第五个 1/10 开始，二个 1/10
6	11	[0.514384, 0.51442)	[0.5143, 0.51442) 间隔的最后 3 个 1/10
7	01	[0.514384, 0.51442)	[0.514384, 0.51442) 间隔的第 4 个 1/10，从第 1 个 1/10 开始
8			从 [0.5143876, 0.514402] 中选择一个数作为输出：0.5143876

图 62

步 骤	间 隔	译 码 符 号	译 码 判 决
1	[0.5, 0.7)	10	0.51439 在间隔 [0.5, 0.7)
2	[0.5, 0.52)	00	0.51439 在间隔 [0.5, 0.7) 的第 1 个 1/10
3	[0.514, 0.52)	11	0.51439 在间隔 [0.5, 0.52) 的第 7 个 1/10
4	[0.514, 0.5146)	00	0.51439 在间隔 [0.514, 0.52) 的第 1 个 1/10
5	[0.5143, 0.51442)	10	0.51439 在间隔 [0.514, 0.5146) 的第 5 个 1/10
6	[0.514384, 0.51442)	11	0.51439 在间隔 [0.5143, 0.51442) 的第 7 个 1/10
7	[0.51439, 0.5143948)	01	0.51439 在间隔 [0.51439, 0.5143948) 的第 1 个 1/10
8			译码的消息：10 00 11 00 10 11 01

图 63

基于字典的压缩：压缩方法的思路完全不同于上述传统思路。编码开始时不知道要编码数据的统计特性，也不一定允许实现知道他们的统计特性，而是主要利用数据本身包含有重复代码这个特性。形成字典编码算法的核心是：如何动态形成字典、如何选择输出格式减小冗余。字典编码大致有两类：RLE 编码（行程编码）以及 LZW 编码。

- **RLE 编码** (行程编码)

- 行程：具有相同灰度值的像素序列；
- 编码思想：去除像素冗余，用行程的灰度和行程的长度代替行程本身。
- 基本原理：通过改变图像的描述方式来实现压缩。
- 例：如信息 `aaaabbccdeeeeefffff` 可以编码为 `4a3b2c1d5e7f`。

- **LZW 编码**：LZW 编码在 LZ77、LZ78 基础上改进而成，它们垄断了通用数据压缩领域的如 `gz`、`zip`、`arj` 等的算法。继承了其压缩速度好、速度快的优点，并且容易被人接受，实现也比较简单。在标准图像数据格式中得到广泛应用，如 `GIF`、`TIFF` 等。

- 思路：类似“成语字典”，用数据中出现的字符序列形成字典，给出字典的索引作为压缩结果。
- 两种方式：
  - 1 (LZ77) 企图查找正在压缩的字符序列是否在以前输入的数据中出现过，然后用已经出现过的字符串替代重复的部分，它的输出仅是指向早期出现过小字符串的“指针”；
  - 2 企图从输入的数据中创建一个“短语词典”，短语可以是任意字母组合。编码诗句的过程中遇到已经在字典中出现的短语即可直接输出字典中短语的“索引号”，而不是短语本身。
- 基本思想：去除像素冗余。在压缩过程中动态地形成一个字符序列表；每当压缩扫描图像发现一个字典中没有的字符序列，就把该序列存到字典中；用字典的地址（编码）作为字符序列的代码并替换原图像中的字符序列；下次再碰到相同的字符序列，就用字典的地址代替；压缩的结果，除了压缩图像外，不需要保留压缩过程中形成的字典，而在解压缩时可以临时恢复之。
- 转换表（字典）：字典存放前缀字符序列，并且为每个表项分配一个码字，也称为序号；实际上是把 8 位 ASCII 字符集合进行扩充，增加的符号用来表示在文本或图像中出现的可变长度 ASCII 字符串。扩充后的代码可用 9、10、11、12 位甚至更多的位来表示（LZW 中为 12 位，12 位转换表有 4096 个表项，其中 256 个表项存放已定义的字符，剩下 3840 个表项存放前缀）。
- 编码时语法分析算法（贪婪分析算法）：串行检查来自数据流的字符串，从中分解出已经识别的最长的字符串，也就是已经在字典中出现的最长前缀 `prefix`。用已知的 `prefix` 加上下一个输入字符 `c` 也就是当前字符作为该前缀的扩展字符，形成新的扩展字符串“缀-符”串 `prefix.c`。换句话说，检查字典中是否存在与它相同的“缀-符”串；若有，则将其变为前缀 `prefix`，继续输入新字符；否则就将其写入字典中生成新前缀，并给一个代码。
- 编码算法：
  - 开始时，字典包含所有可能的字符串，`P` 为空；
  - 取当前字符 `c`（字符流中的下一个字符）；
  - 判断 `P + c`（缀-符）是否在字典中：若是，则 `P = P+c`；若否，则把当前前缀 `P` 的码字输出到码字流；把 `P + c` 添加到字典并令 `P = c`。
  - 判断码字流中是否还有码字要译：若是，则返回步骤 2；否则，把代表当前前缀 `P` 的码字输出到码字流，结束。
- 译码算法：令 `cw` 表示当前码字，`string.cw` 表示当前“缀-符”串；`pw` 表示先于当前码字的码字，`string.pw` 表示先前“缀-符”串。LZW 译码算法开始时，译码字典与编码字典相同，包含所有可能的前缀根。LZW 算法在译码过程中会记住先前码字 `pw`，从码字流读入当前码字 `cw` 之后输出当前“缀-符”串 `string.cw`，然后用 `cw` 的第一个字符扩展先前的“缀-符”串 `string.pw` 添加到字典中。具体步骤如下：
  - 开始译码时，字典包含所有可能的字符串，`pw` 为空；
  - 提取当前字符 `cw` 为码字流中的第一个码字；

- 输出当前“缀-符”串 `string.cw`；
- 令 `pW = cw`；同时令当前码字 `cw` 为码字流的下一个码字。
- 判断当前码字 `cw` 是否在字典中：
  - 如果是，则输出 `string.cw` 到输出符号流；同时，先前前缀 `P` 为先前“缀-符”串 `string.pw`，当前字符 `C` 为当前“缀-符”串 `string.cw` 第一个字符，把 `P+C` 加入字典中；
  - 如果否，则先前前缀 `P` 为先前“缀-符”串 `string.pw`，当前字符 `C` 为先前“缀-符”串的 `string.pw` 第一个字符，把 `P+C` 加入字典中；
- 判断码字流是否还有码字要译，若有返回第 4 步。

○ 例：编码如图 64 所示，译码如图 65 所示。

<p><b>■ 编码实例</b></p> <p>输入字符串：</p> <table border="1" style="margin-left: 20px; margin-bottom: 10px;"> <tr><td>位置</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td></tr> <tr><td>字符</td><td>A</td><td>B</td><td>B</td><td>A</td><td>B</td><td>A</td><td>B</td><td>A</td><td>C</td></tr> </table> <p>输出码字串：</p> <p>122473</p>	位置	1	2	3	4	5	6	7	8	9	字符	A	B	B	A	B	A	B	A	C	<p><b>■ 译码实例</b></p> <p>输入码字串：</p> <p>122473</p> <p>输出字符串：</p> <p>ABBABABAC</p>	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>步骤</th> <th>位置</th> <th>字典</th> <th>输出</th> </tr> </thead> <tbody> <tr><td></td><td></td><td>(1) A</td><td></td></tr> <tr><td></td><td></td><td>(2) B</td><td></td></tr> <tr><td></td><td></td><td>(3) C</td><td></td></tr> <tr><td>1</td><td>1</td><td>(4) AB (1)</td><td></td></tr> <tr><td>2</td><td>2</td><td>(5) BB (2)</td><td></td></tr> <tr><td>3</td><td>3</td><td>(6) BA (2)</td><td></td></tr> <tr><td>4</td><td>4</td><td>(7) AB A (4)</td><td></td></tr> <tr><td>5</td><td>6</td><td>(8) AB AC (7)</td><td></td></tr> <tr><td>6</td><td>--</td><td>-- -- (3)</td><td></td></tr> </tbody> </table> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>步骤</th> <th>代码</th> <th>字典</th> <th>输出</th> </tr> </thead> <tbody> <tr><td></td><td></td><td>(1) A</td><td></td></tr> <tr><td></td><td></td><td>(2) B</td><td></td></tr> <tr><td></td><td></td><td>(3) C</td><td></td></tr> <tr><td>1</td><td>(1)</td><td>--</td><td>A</td></tr> <tr><td>2</td><td>(2)</td><td>(4) A B</td><td>B</td></tr> <tr><td>3</td><td>(2)</td><td>(5) B B</td><td>B</td></tr> <tr><td>4</td><td>(4)</td><td>(6) B A</td><td>A B</td></tr> <tr><td>5</td><td>(7)</td><td>(7) A B A</td><td>A B A</td></tr> <tr><td>6</td><td>(3)</td><td>(8) A B A C</td><td>C</td></tr> </tbody> </table>	步骤	位置	字典	输出			(1) A				(2) B				(3) C		1	1	(4) AB (1)		2	2	(5) BB (2)		3	3	(6) BA (2)		4	4	(7) AB A (4)		5	6	(8) AB AC (7)		6	--	-- -- (3)		步骤	代码	字典	输出			(1) A				(2) B				(3) C		1	(1)	--	A	2	(2)	(4) A B	B	3	(2)	(5) B B	B	4	(4)	(6) B A	A B	5	(7)	(7) A B A	A B A	6	(3)	(8) A B A C	C
位置	1	2	3	4	5	6	7	8	9																																																																																													
字符	A	B	B	A	B	A	B	A	C																																																																																													
步骤	位置	字典	输出																																																																																																			
		(1) A																																																																																																				
		(2) B																																																																																																				
		(3) C																																																																																																				
1	1	(4) AB (1)																																																																																																				
2	2	(5) BB (2)																																																																																																				
3	3	(6) BA (2)																																																																																																				
4	4	(7) AB A (4)																																																																																																				
5	6	(8) AB AC (7)																																																																																																				
6	--	-- -- (3)																																																																																																				
步骤	代码	字典	输出																																																																																																			
		(1) A																																																																																																				
		(2) B																																																																																																				
		(3) C																																																																																																				
1	(1)	--	A																																																																																																			
2	(2)	(4) A B	B																																																																																																			
3	(2)	(5) B B	B																																																																																																			
4	(4)	(6) B A	A B																																																																																																			
5	(7)	(7) A B A	A B A																																																																																																			
6	(3)	(8) A B A C	C																																																																																																			
图 64	图 65																																																																																																					

## 8.5 图像压缩编码的基本方法——有损压缩

### 有损压缩

- 有损压缩是利用人类的视觉心理冗余，通过忽略不引起广泛关注的细节信息，实现图像数据高倍率的压缩。另一方面，通过牺牲图像的部分准确率，亦可达到加大压缩率的目的。如果我们容忍解压缩的结果中有一定的误差，那么压缩率将显著提高；
- 通常，有损压缩方法在图像压缩比大于 50:1 时仍然能够有效地重构图像，现代方法甚至可以达到 100:1 以上；而如果压缩比为 10:1 或 30:1，重构的图像与原图在视觉上几乎没有差别。对一般图像，无损压缩的压缩比很少有超过 3:1 的，这两种压缩方法的根本差别在于对图像数据的量化上。
- **分类：**预测编码（如 DPCM、DM 编码），变换编码（如 DCT 变换编码，小波变换编码等）。

**预测编码：**根据离散信号之间存在一定相关性的特点，利用前面的一个或多个信号对下一个信号进行预测，然后用实际值和预测值的差进行编码。如果预测比较准确，那么误差信号就会很小，就可以用很少的码位编码，从而达到数据压缩的目的。第  $n$  个符号的熵满足

$$H(x_n) \geq H(x_n | x_{n-1}) \geq \dots \geq H(x_n | x_{n-1}x_{n-2}\dots x_1)$$

参与预测的符号越多，预测就越准确，该信源的不确定性就越小，数码率就可以降到更低。预测时所用的方程式称为**预测方程式**，其具有一般形式：

$$x'_k = f(x_1, x_2, \dots, x_{k-1}, k)$$

可以采用线性预测方法：

$$x'_k = \sum_{i=1}^{k-1} a_i(k) x_i$$

如果  $a_i$  是常数，则为时不变线性预测；否则为自适应线性预测（ADPCM）。一个最简单的预测方程即为  $x'_k = x_{k-1}$ 。

**最佳预测编码：**使误差函数  $E[(x_n - x'_n)^2]$  达到最小值的预测方程式叫做最佳预测。于是列方程组求最佳线性预测的各个参数  $a_i$ 。一般地，有如下方程组：

$$E[x_n x_i] = \sum_{l=1}^{n-1} a_l E[x_l x_i] \quad (i = 1, 2, \dots, n-1)$$

如果为一阶线性预测，可以求得

$$a_1 = \frac{E[x_n x_{n-1}]}{E[x_{n-1}^2]}$$

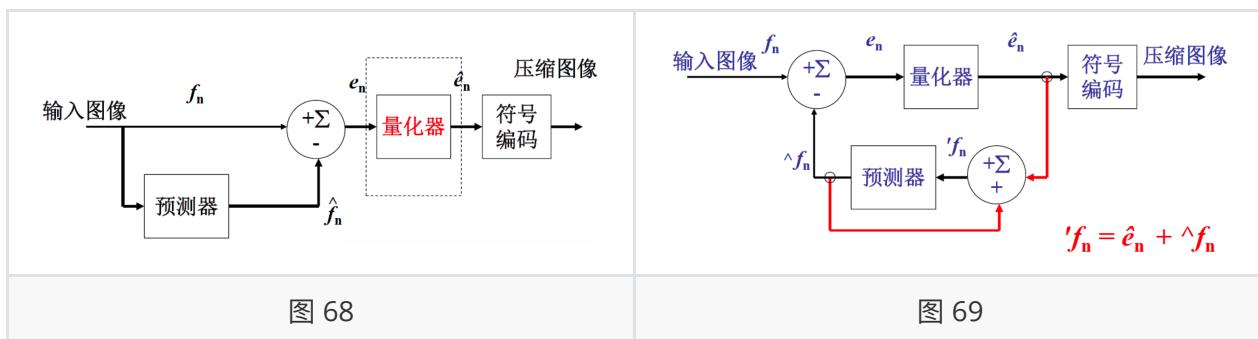
得到  $x'_n = a_1 x_{n-1}$ 。

### DPCM 编码

- **基本思想：**去除像素间相关性冗余；当前像素值的发生与以前像素值集合密切相关，因此可以用以前像素值来预测当前像素值。用当前时刻以前的像素值  $f_{n-1}$ ，通过预测方法得到一个预测值  $\hat{f}_n$ ，对当前值和预测值求差，对差值进行编码，作为输出压缩数据流中的下一个元素。由于差比原数据要小，因此编码可以较小，可用变长编码形成无损 DPCM 编码。大多数情况下， $f_n$  的预测是通过  $m$  个以前像素的线性组合来生成。
- DPCM 是有损还是无损主要取决于对差值  $e_n$  如何编码。
- **无损 DPCM：**编码如图 66 所示；解码如图 67 所示。

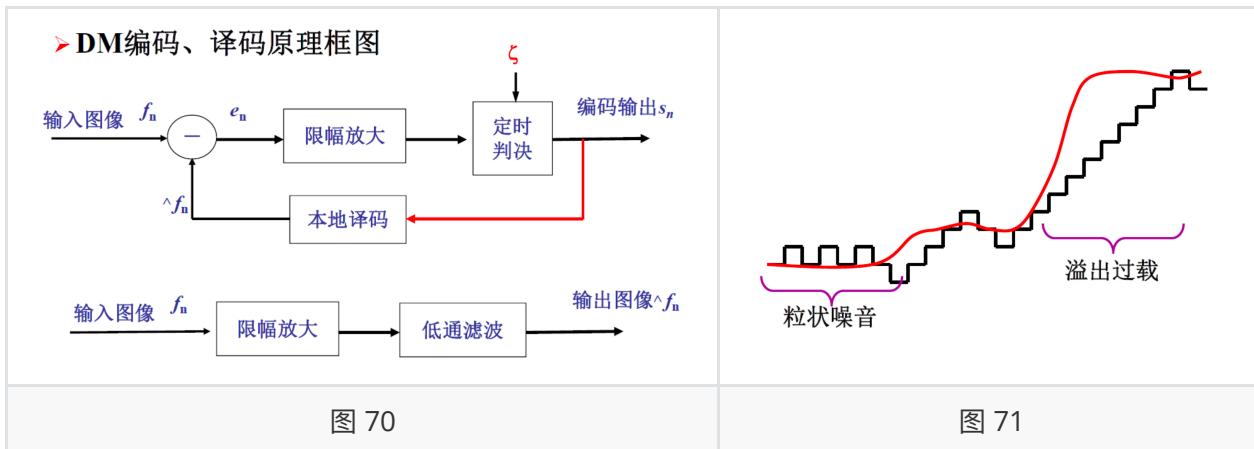


- **有损 DPCM：**对无损预测压缩的误差进行量化，通过消除视觉心理冗余达到对图像进一步压缩的目的，即令  $\hat{e}_n = Q(f_n - \hat{f}_n)$ ，在符号编码之前加入量化器即可，如图 68 所示。在有损 DPCM 解码中，编码时预测器的输入为  $f_n$ ，而解码时预测器的输入为  $'f_n = \hat{e}_n + \hat{f}_n$ ，理论上应该使用不同的预测器，要使用相同的预测器，编码方案要进行修改。
- **改进有损 DPCM 编码：**在编码时，将量化器放在符号编码之前，用  $\hat{e}_n + \hat{f}_n$  的结果反馈给预测器预测下一个值，如图 69 所示。



## DM 有损预测编码（Delta 调制）

- **基本原理：**通过恒定增量（减量）的变化，以数字方波模拟连续信号。相当于一个 1-bit 的预测编码器，用一位码字来控制误差。
- **量化器：**若  $e_n > 0$ ，量化为  $+\zeta$ ，否则量化为  $-\zeta$ ，其中  $\zeta$  为正常数； $\hat{e}_n$  用 1 位编码。
- **预测器：** $\hat{f}_n = f_{n-1} \alpha$ ,  $\alpha$  为小于 1 的预测系数。
- **编码器与解码器：**如图 70 所示。
- **算法分析：**在信号变化快的区域， $\zeta$  太小以至于不能表示输入的最大变化，发生一个被称为溢出过载的失真；在信号相对平滑的区域， $\zeta$  太大以至不能表示输入的最小变化，出现了粒状噪音；在大多数情况下，这两种现象导致对象边缘的钝化和平滑区域表面粒状失真，如图 71 所示。



**有损预测编码分析：**在所有有损预测压缩中都会出现误差，误差的严重程度取决于使用的量化方法和预测方法之间的相互作用。尽管存在相互作用，定义预测函数时仍然假定没有量化误差，而定义量化函数时仅是尽可能降低它自身的误差。量化函数和预测函数可分别定义与设计，分别寻求最佳设计方案。我们接下来寻找预测器的最优设计方案。

**最优预测器：**最优预测器需要满足条件：① 误差最小  $'f_n = \hat{e}_n + \hat{f}_n \approx e_n + \hat{f}_n = f_n$ ；② 仅用前面的值预测后面的值  $\hat{f}_n = \sum_{i=1}^m \alpha_i f_{n-i}$ 。其基本原理为：预测值可限制为前  $m$  个点的线性组合函数（不是必须的，但是能简化分析，同时减小计算复杂度）。预测编码的结果被称为差分调制脉冲码 (DPCM)。在以上条件下，最佳预测器的设计问题可归结为，如何选取  $m$  个预测系数，使得下式达到极小值：

$$E\{e_n^2\} = E\left\{\left[f_n - \sum_{i=1}^m \alpha_i f_{n-i}\right]^2\right\}$$

对上式微分后解方程组，有  $\alpha = R^{-1}r$ ，其中  $R$  为  $m \times m$  矩阵且  $R_{i,j} = E\{f_{n-i}f_{n-j}\}$ ， $r$  为  $m \times 1$  向量且  $r_i = E\{f_n f_{n-i}\}$ 。于是，对于任何一个输入图像，系数  $\alpha$  都可以通过矩阵运算得到，其仅依赖于原始图像中像素之间的关系，用这些最优系数产生的预测方差是

$$\sigma_e^2 = \sigma^2 - \alpha^T r = \sigma^2 - \sum_{i=1}^m E\{f_n f_{n-i}\} \alpha_i$$

其中， $\sigma^2$  为标准方差（非最优时）。尽管计算方程组很简单，但是根据像素关系计算  $R$  与  $r$  事很复杂的事情，很少使用实时计算模型（对每个图像都计算一遍预测系数），大多数情况下， $R$  与  $r$  的计算通过一个简单的固定图像得到；利用固定模型计算得到的预测器，很难适应其他不同特性图像的预测要求。因此我们考虑自适应预测器，在预测过程中不断修改预测器参数，使之适应不同图像的统计特性。采用的图像统计模型不同将会形成不同的自适应预测编码算法，如线性自适应、非线性自适应、局部线性自适应、局部非线性自适应等。

**预测器误差模型：**预测误差的概率密度函数一般来说在 0 点是高尖的，且具有变化范围较小的特征。上述预测误差密度函数，经常通过 0 平均无关拉普拉斯概率密度来模型化，即

$$p(e) = \frac{1}{\sqrt{2}\sigma_e} e^{-\sqrt{2}|e|/\sigma_e}$$

其中， $\sigma_e$  是输入值  $e$  的标准方差。结合上述预测误差模型，进一步设计最优量化函数，可以达到更好的编码效果。

**活动图像的帧间预测编码：**视频信号的冗余度主要体现在空间相关性（帧内）、时间相关性（帧间）以及色度空间表示上的相关性。对于每秒 25 (30) 帧的电视信号，相继帧之间存在极强的相关性。据统计 256 级灰度的黑白图像序列，帧间差值超过 3 的像素不超过 4%，所以在活动图像序列中可以利用前面的帧预测后面的帧来实现数据压缩。该技术被广泛用于 H.261, H.263, MPEG-1 与 MPEG-2 等视频压缩标准之中。

### 变换编码

- **基本思想：**将原始数据变换到另一个表示空间，使得数据在新的空间上尽可能相互独立而能量更集中。
- **实现思路：**用一个可逆线性变换（如傅立叶变换），把图像映射到变换空间，将图像像素集合转化为变换系数的集合，对系数集合进行量化和编码；对大部分自然图像，重要系数数量较少，因此可仅以较小的图像失真为代价，进行量化或完全抛弃。
- **正交变换：**经常采用正交变换进行变换编码，因为正交变换具有熵保持特性、正交变化具有能量保持特性、正交变换可以重新分配集中能量以及其具有去相关特性。我们的目的是选取正交矩阵  $T$ ，使得变换  $Y = TX$ 、截取  $Y' = [Y_0, Y_1, \dots, Y_M]$ 、再反变换  $X' = T^{-1}Y'$  得到的失真尽可能小，同时压缩率尽可能大。
- **最佳变换：**基本条件为 ① 全部消除变换系数之间的相关性；② 变换系数的能量高度集中。考虑正交变化输出向量  $Y$  的协方差矩阵  $C_Y = TC_X T$ ，则实现最佳变换的矩阵应使  $C_Y$  称为对角线矩阵的同时，能量集中在前  $M$  个元素上。
- **编码解码基本思想：**分别如图 72、73 所示。其中， $n \times n$  的子图将  $N \times N$  原图分为  $(N/n)^2$  个部分。



图 72

图 73

- **变换的选择：**均方误差准则下最佳统计变换为 **K-L 变换** (Karhunen-Loeve 变换, KLT)，理论上对一阶马尔可夫图像信源可实现最佳变换，但计算复杂，需动态调整变换矩阵；均方误差准则下的准最佳统计变换使得变换系数的协方差矩阵接近对角线，同时具有快速计算能力，包括：DFT (离散傅立叶变换)、DCT (离散余弦变换)、WHT (Walsh-Hadamard 变换)、HT (哈尔变换)、小波变换等。
- **对变换的评价：**按照信息封装能力排序，可以得到 KLT、DCT、DFT、WHT、HT (强->弱)。由于 KLT 的基图是数据以来的，每次都需要重新计算，因而很少使用。常用的是 **DCT**，已被国际标准采纳，优点有：基本没有快效应、信息封装能力强（把最多的信息封装在最少的系数中）。
- **子图尺寸的选择：**两个原则如下
  - 为便于降低计算复杂度，子图维数  $n$  应该是 2 的正整数次方。
  - $n$  一般选取 8 或 16，由实践得到。需要注意的是，随着  $n$  的增加，块效应相应减少。

- 截取、量化与编码：截取和量化一般有两种方法：

- 区域编码（子带编码）

- 基本思想：找出具有最大方差的  $m$  个系数的位置，同时确定系数的坐标  $(u, v)$ ，这  $m$  个系数的  $Y$  值保留，其余  $Y$  值抛弃；对所有子图像使用相同的编码模板。因为大部分的信息应该包含在最大方差的变换系数中，每一个 DCT 变换系数可以认为是一个随机变量，该变量的分布可以在所有变换子图像的集合上进行计算。
- 构建变换系数截取模板函数： $m(u, v) = [Y(u, v) \text{ 不满足截断标准}]$ 。于是可以得到  $X' = T^{-1}MY$ ， $M$  中的原则用来消去对等式总和贡献最小的基。有两个原则：① 根据最大方差的分布情况得到系数截取模板；② 方差最大的地方置 1，其余置 0。
- 最大方差的计算：方差本身可直接由  $(N/n)^2$  个变换子图像数组集合计算得到；或基于一个假想图像模型得到。
- 算法实现：① 计算模板，方差最大的地方置 1，其余置 0；② 量化系数，线性、非线性或自适应量化器；③ 结果编码：有两种分配二进制位的编码方法：系数被赋予相同数量的二进制位以及系数之间固定分配一定的二进制位。

- 门限编码（阈值编码、自适应编码）

- 基本思想：没有一个消去系数的固定模板，不同的子图保留不同的系数，通过一个阈值  $T$  来决定每个系数的去留，即如果系数大于阈值，则保留，否则消去。由于其简单性，阈值编码是实际应用中更常使用的编码方法。
- 理论依据：取值最大的变换系数，在重构子图质量中起的作用也最重要；最大系数的分布随子图的不同而不同。
- 阈值选取：常有三种取法：① 所有子图使用同一个全局阈值：压缩率的大小随图像不同而不同，由超过全局阈值的系数个数所决定；② 对每个子图取不同阈值：每个子图保留的系数个数事先确定，即总是保留  $N$  个最大的，称为 N-最大化编码；对于每个子图同样多的系数被丢弃，因此，每个子图压缩率是相同的，并且也是预先知道的；③ 阈值作为子图系数位置的函数：所有子图使用同一个全局阈值模板，但阈值的取值与系数的位置相关，阈值模板给出了不同位置上系数的相应阈值。
- 对系数的编码：将系数按 45 度对角展开成序列（Zig-Zag 排序），得到一个一维数据序列，用 RLE 编码（行程编码）对上述序列编码。

## 小波变换图像压缩

- 基本算法步骤

- 按照图像特性，选择合适的小波基；
- 对图像进行小波变换；
- 确定度过一个阈值，超过该阈值的所有系数具有的能量总和大于某个界限；
- 对超过阈值的系数进行编码。

- 小波基的选择：是决定压缩性能的关键。对称性的小波基压缩效果较好，常用的小波基有 Harr 小波、Daubechies 小波、Coiflet 小波（Daubechies 小波的改进，比其更为对称）、Symlets 小波（同样为 Daubechies 小波为改善对称性的改进）。

- 相比于 DCT 变换小波变换的优点：

- 小波变换具有熵保持特性，能够有效地改变图像的能量分布，同时不损伤原始图像所包含的信息；
- 小波分解后大部分能量集中在低频子图的少量系数上；而大量的高频子图系数普遍较小，且存在明显相关性，有利于获得较高的编码效益；
- 小波变换作用于图像的整体，既能去除图像的全局相关性，又可以将量化误差分散到整个图像中，避免了方块效应的产生。

- 多级分解后行程的不同分辨率和频率特性的子带信号，便于在失真编码中综合考虑视觉特性，同时有利于图像的渐进传输。
- 压缩效果：**具有对称性的小波基压缩效果较好。

## 8.6 图像压缩标准

### 图像标准的制定

- ITU-T 标准（建议）：以字母排序，如 H.261 等，主要用于实时视频通信；
- ISO/IEC 标准：按序号排列，如 MPEG-1 对应 11172 等等，MPEG 标准主要用于广播电视、DVD 与视频流媒体。
- 两个标准组织独立制定不同的标准，但在许多方面有共同之处，如 H.262 标准与 MPEG-2 视频编码标准基本上是同一个标准等等，如图 74 所示。

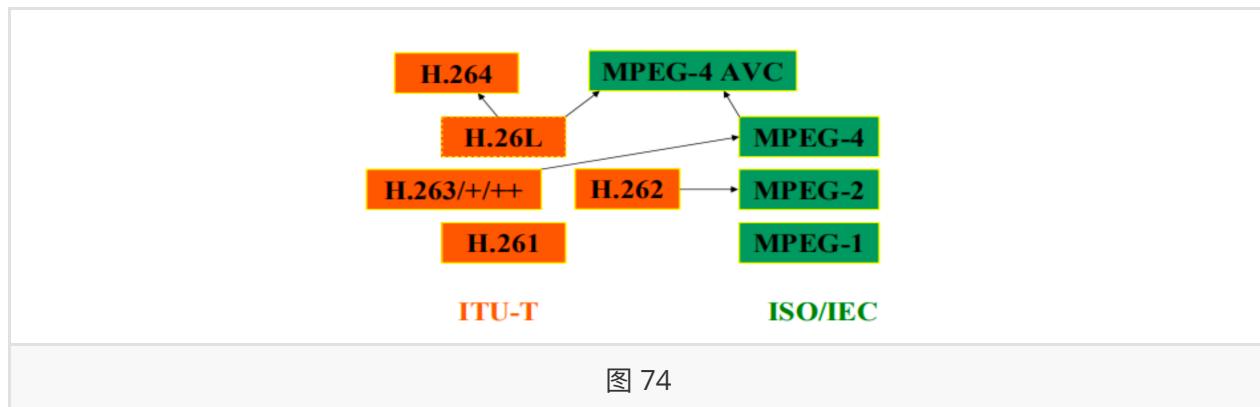


图 74

- 两个标准组织经常形成联合工作组，共同制定一些视频编码标准。如 JVT 标准由 ISO/IEC-MPEG 与 ITU-T-VCEG 联合成立的视频工作组 JVT 制定，致力于新一代数字视频压缩标准的制定；后更名 MPEG-4 AVC 标准 / H.264 标准（不同组织的命名标准）
- 中国参与了国际标准，提出 MPEG-China，接着开发具有独立知识产权的 AVS 技术。

**图像压缩标准类型：**① 包括二值图像压缩标准（面向传真设计）、② 静止帧连续调（黑白/彩色）图像压缩标准（面向静止的单幅图像）；③ 连续帧连续调（黑白/彩色）图像压缩标准（面向连续的视频影像）。

**JPEG 标准：**属于静止帧黑白/彩色图像压缩标准。

- 三种压缩系统：**基线编码系统（面向有损压缩的应用，采用 DCT 变换压缩）、扩展编码系统（面向递进式应用，从低分辨率到高分辨率逐步递进传递）、独立编码系统（面向无损压缩的应用，采用无损预测压缩，符号编码采用哈夫曼或算术编码）。
- 【注】**一个产品或系统必须包括对基线系统的支持。
- 标准 JPEG 压缩流程：**如图 75 所示。
  - 构造子图像：子图像尺寸为  $8 \times 8$ ；
  - 颜色空间转换：人眼对亮度更敏感，因此提取亮度特征，将 RGB 转化为 YCbCr 模型，编码时对亮度采用特殊编码，赋予更多的码速率，而对色差分量可给予较少的码速率。编码与解码可以参考后文【参考资料】RGB 与 YCbCr 转换一节。
  - 零偏置转换：（DCT 前操作）对于灰度级是  $2^n$  的像素，通过减去  $2^{n-1}$ ，替换像素本身。对于  $n = 8$ ，即将  $[0, 255]$  值域映射到  $[-128, 127]$ ，为了使像素绝对值出现 3 位十进制的概率大大减少。

- 频域变换：频域变换产生 64 个系数，第一个系数称为直流系数（DC 系数），其余 63 个系数称为交流系数（AC 系数）。
- 系数量化：采用阈值作为子图系数位置函数的量化方式，所有子图使用同一个全局阈值模板，但阈值的取值，与系数的位置有关，阈值模板给出了不同位置上系数的相应阈值。同时，对亮度和颜色使用不同的量化阈值模板，并取整。
  - 正向量化： $S_q(u, v) = \text{round}(S(u, v)/Q(u, v))$ ;
  - 逆向量化： $R(u, v) = \text{round}(S_q(u, v)Q(u, v))$ 。
- 符号编码：将量化后的系数按之字形排列成向量，全零结尾用特殊符号 EOB；DC 和 AC 用不同的方式分别编码。
  - DC 编码方式：预测+统计，由两部分区间号编码（SSSS，又表示 DC 项有意义的数位个数）和系数预测误差本身编码（VVVV）组成。区间表与区间 DC 的 Huffman 编码表见图 算法步骤如下：

- 求 DPCM，即用当前 DC 减去前一个子图的 DC 得到 VVVV（即 DIFF），即  $\text{DIFF} = DC - DC_{pre}$ ；
- 根据 DIFF 得到区间号 SSSS（可以查区间编号得出区间号 SSSS，再根据 SSSS 查哈夫曼编码表得其 Huffman 编码）；
- 对 VVVV 编码，正数是子集，负数用绝对值二进制按位取反。

这种编号下，解码时如果 VVVV 部分首位为 0，则为负数。

**【例】**  $DC = -26$ ,  $DC(\text{PRE}) = -17$ ，于是  $\text{DIFF} = -26 - (-17) = -9$ ，区间号 SSSS = 4，查表得到 Huffman 编码为 101；VVVV = -9 二进制编码为 0110，因此最后的编码为 1010110。

- AC 编码方式：编码由两部分组成，区间号编码（RRRR/SSSS）与系数本身（VVVV）；SSSS 表示区间号，RRRR 表示该系数前值为 0 的系数的个数；VVVV 表示系数本身编码。区间 AC 与区间 AC Huffman 编码表（一部分）见图 77、图 79。

**【例】** 对 07 编号有 RRRR = 1，查表有 SSSS = 3，于是由 RRRR/SSSS = 1/3 查表得到 1111001；VVVV = -7 二进制编码为 000，因此最后编码为 1111001000。

- 标准 JPEG 解压缩流程：如图 76 所示。

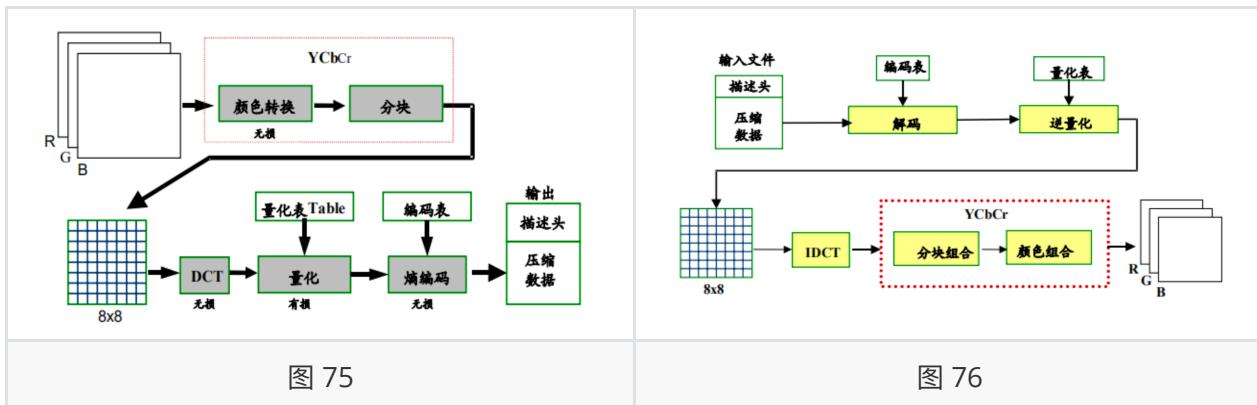


图 75

图 76

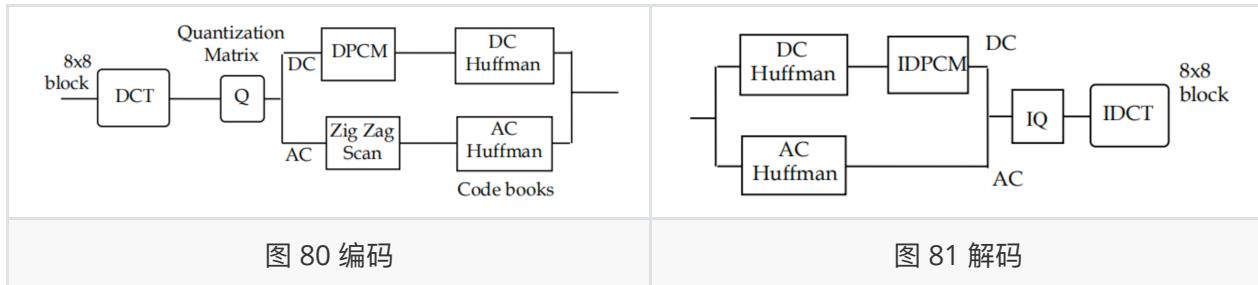
范围	DC差区间	AC区间	区间	编码	长度	区间	编码	长度	行程/区间	编码	长度
0	0	N/A	0	010	3	6	1110	10	0/0	1010(=EOB)	4
-1, 1	1	1	1	011	4	7	11110	12	0/1	00	3
-3, -2, 2, 3	2	2	2	100	5	8	111110	14	0/2	01	4
-7,...,-4, 4,...,7	3	3	3	00	5	9	1111110	16	0/3	100	6
-15,...,-8, 8,...,15	4	4	4	101	7	A	11111110	18	0/4	1011	8
-31,...,-16, 16,...,31	5	5	5	110	8	B	111111110	20	0/5	11010	10
-63,...,-32, 32,...,63	6	6							1/1	1100	5

图 77

图 78

图 79

- **JPEG 编解码系统总结：**见上文“符号编码”部分，这里用图 80~81 进行总结。



- **基于 DCT 变换的标准 JPEG 压缩的缺陷**

- 存在严重的方块效应；
- 抗干扰能力差。

#### 【参考资料】RGB 与 YCbCr 转换

$$\begin{aligned}
 Y &= 0.2990R + 0.5870G + 0.1140B \\
 C_b &= -0.1787R - 0.3313G + 0.5000B + 128 \\
 C_r &= 0.5000R - 0.4187G - 0.0813B + 128
 \end{aligned}$$

$$\begin{aligned}
 R &= Y + 1.40200(C_r - 128) \\
 G &= Y - 0.34414(C_b - 128) - 0.71414(C_r - 128) \\
 B &= Y + 1.77200(C_b - 128)
 \end{aligned}$$

**JPEG2000 标准：**克服标准 JPEG 算法的缺陷，寻求更大的压缩率。最终选择小波变换进行图像压缩，2000 年形成技术整体文本，2001 年形成标准，称为 JPEG2000 标准。

- **特征：**

- 高压缩率：JPEG2000 图片压缩比较 JPEG 提高 30%，同时压缩后图像显得更加细腻平滑；
- 无损压缩和有损压缩：JPEG2000 提供无损和有损两种压缩方式，允许从有损到无损的渐进解压；
- 渐进传输：JPEG2000 格式支持渐进传输；
- 感兴趣区域压缩：可以制定图片上感兴趣区域的压缩质量，或在恢复时制定某些区域的解压缩要求；
- 码流的随机访问和处理：用户可在图像中随机定义感兴趣区域，使得这一区域的图像质量高于其他图像区域；码流的随机处理允许用户进行旋转、移动、滤波和特征提取等操作；
- 具有容错能力；
- 开放的框架结构：编码器只实现核心的工具算法和码流的解析，解码器可以要求数据源发送未知的工具算法。
- 基于内容的描述：JPEG2000 压缩系统特性之一。

- **编码器与解码器：**如图 82~83 所示。

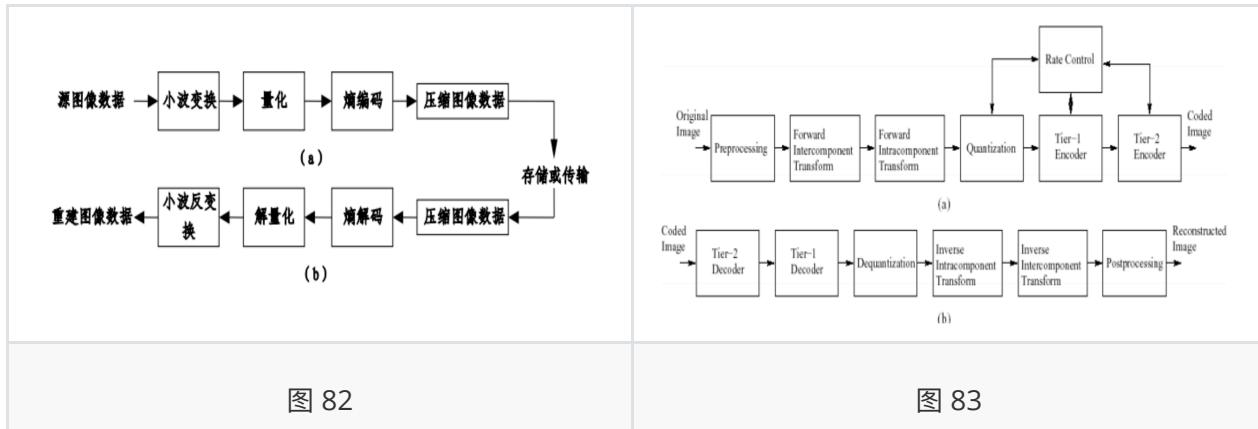


图 82

图 83

- **主要思想：**借助于小波变换进行多级图像变换后，只有很少部分的系数需要进行编码；使用具有最佳截断的嵌入式分块编码算法 (EBCOT)；采取比特平面编码方式。

### JPEG 与 JPEG2000 性能比较

标准	JPEG	JPEG2000
主要编码技术	离散余弦变换 DCT、知觉量化、Zig-zag 扫描、Huffman 编码、算术编码	离散小波变化 DWT、EBCOT 核心算法、ROI 编码、空间可扩展编码、质量可扩展编码、面向对象编码、位图形状编码、容错编码、TCQ、零数扫描
压缩比	2~30	2~50
算法效率	30:1 以上急剧下降	100:1 以上急剧衰减
速率失真特性		比 JPEG 提高 30%

**连续帧图像：**由多幅尺寸相同的精致图像组成的图像序列，被称为连续帧图像。与静止帧图像相比，连续帧图像多了一个时间轴，成为三维信号，因此连续帧图像也被称为三维图像。典型的连续帧图像是视频图像。

### 连续帧图像压缩的基本思想

- 基于基本假设：在各连续帧之间存在简单的相对平移运动，帧与帧之间相互存在极大的相关性；
- 通过减少帧间图像数据冗余，来达到减少数据量、压缩连续帧图像体积的；

- 一个画面上的像素量值可根据同帧附近像素加以预测，被称为帧内编码技术；也可以根据附近帧中的像素来加以预测，被称为帧间编码技术。
- 将连续帧图像序列，分为参考帧与预测帧，参考帧用精致图像压缩方法进行压缩，预测帧用帧差图像进行压缩。由于帧差图像的数据量大大小于参考帧，从而可以达到很高的压缩比。

### 帧间预测编码技术

- 向前预测：用前一帧预测当前帧；
- 双向预测：用前一帧和下一帧预测当前帧。
- 一些定义（如图 84 所示）
  - I 帧：不进行预测、进行帧内编码的编码帧（参考帧）；
  - P 帧：通过向前预测得到的误差编码帧；
  - B 帧：通过双向预测得到的误差编码帧，因图像序列存放在存储器中，所以可以使用下一帧。

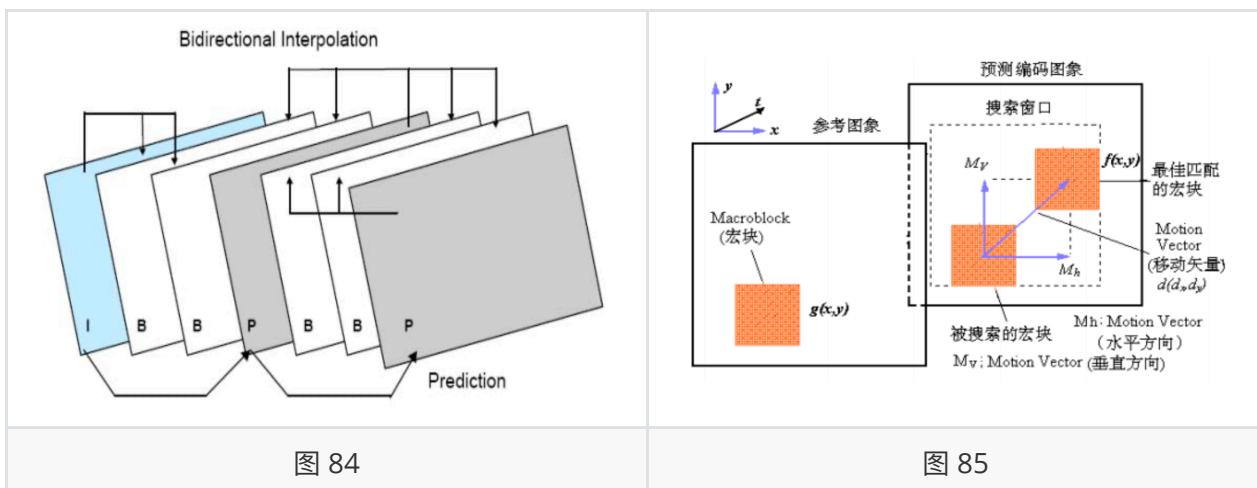
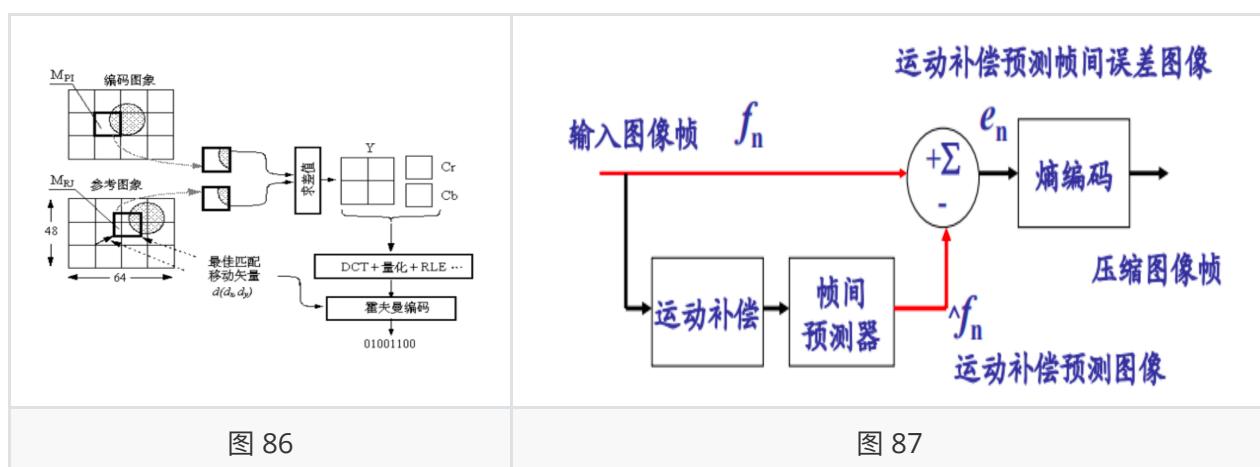


图 84

图 85



### 帧间预测编码技术

- 编码中的运动补偿
  - 运动补偿概念是对帧间运动的估算为基础的，若物体在空间上有一位移，那么用有限的运动参数来对帧间的运动加以描述，如对于像素的平移运动，可用运动矢量来描述。
  - 一个来自前一编码帧的运动补偿预测像素，就能给出一个当前像素的最佳预测。预测误差和运动矢量一同参与编码。
  - 由于一些运动矢量之间的空间相关性通常较高，因此，一个像素的运动矢量，可以代表一个相邻像素块的运动；

- 实现中，画面一般划分成一些不连接的像素块，对于每一个这样的像素块（在 MPEG1/2 中一个像素块为  $16 \times 16$  像素），只估算一个运动矢量。
- 常用的基于块的运动估算和补偿——**块匹配法**。主要思想是在搜索窗口内找到最佳匹配的宏块，如图 85 所示。
- 一般来说，只需要传送运动矢量和差值帧即可对连续帧进行编码，如图 86 所示。
- 帧间预测编码：**如图 87 所示。

**H261/H263 标准：**由 CCITT (ITU-T) 制定的标准。

- H261 标准：**应用于 ISDN 视频会议，主要编码技术为 DCT 变换，向前运动补偿预测、Zig-zag 排序、Huffman 编码，帧间编码模式为 IPPPPPIPPP.....。
- H263 标准：**应用于可视电话，主要编码技术为 DCT 变换，双向运动补偿预测、Zig-zag 排序、Huffman 编码、帧间编码模式 IBBPBBPBBIBBP.....。

**MPEG 标准：**MPEG 是 ISO/IEC 联合成立专家组制定的标准。MPEG 压缩标准是针对运动图像而设计的，平均压缩比可以达到 50:1，压缩率比较高，又有统一的格式，兼容性好。MPEG 标准阐明了声音和电视图像的编码和解码问题，严格规定了声音和图像数据编码后组成比特数据流的句法，提供了解码器的测试方法等，但没有对所有内容作严格规定，尤其是压缩和解压缩的算法。这样既保证了解码器能对符合 MPEG 标准的声音数据和电视图像数据进行正确解码，又给 MPEG 标准的实现留有很大余地。人们可以不断改进编码解码算法，提高声音和电视图像的质量以及编码效率。

- 编码流程：**图像序列 → RGB 转 YUV → 运动估计 → DCT → 量化 → RLE 编码 → 输出。
- MPEG1 标准：**数字电视标准，针对具有 1.5Mbps 以下数据传输率的存储或传输媒体的视频及其伴音编码解码的国际标准。视频编码采用标准交换格式，压缩率约 26:1，图像质量接近家用录像系统 VHS；音频解码器可以达到 CD 质量。
  - 应用范围：视频/音频 CD-ROM 存储、视频消费；
  - 主要编码技术：DCT 变换、前向/双向运动补偿预测、Zig-zag 排序、Huffman 编码、算术编码，每 15 帧至少要有一个 I 帧，即 IBBPBBPBBIBBP.....
  - MP3 指：MPEG1 Level 3。
- MPEG2 标准：**数字电视标准，得到一个通用标准，能够在很宽范围内对不同分辨率和不同输出比特率的图像信号进行压缩；
  - 应用范围：主要用于高画质的动画、DVD、数字卫星通信及数字电视广播。
  - 主要编码技术：DCT 变换、前向/双向运动补偿预测、Zig-zag 排序、Huffman 编码、算术编码，每 15 帧至少要有一个 I 帧，即 IBBPBBPBBIBBP.....
  - 与 MPEG1 兼容。
- MPEG3 标准：**用途 HDTV 与 EDTV，但是被 MPEG2 标准合并。
- MPEG4 标准：**多媒体应用标准。制定一个通用的低码率压缩标准，并采取第二代压缩算法，以有效支持甚低码率应用，同时实现内容的压缩编码，提高可靠性。同时将自然和合成物品进行编码，同时压缩率可以达到接近 100 倍。
  - 为支持对视频内容的访问，提出了 AVO 的概念，与物件可以有互动关系；同时加入了场景的逻辑结构，可以对场景结构进行基于物体的压缩编码；
  - 应用范围：互联网、交互视频、移动通信；
  - 主要编码技术：DCT 变换、前向/双向运动补偿预测、Zig-zag 排序、脸部动画、背影编码、Huffman 编码、算术编码，每 15 帧至少要有一个 I 帧，即 IBBPBBPBBIBBP.....
  - MPEG 在 MPEG1/2/4 中压缩能力最强，数据大小小，影响品质很好，网络传输最佳化。
- MPEG7 标准：**重点在于影音系统的描述和定义，形成跨媒体的统一的多媒体检索、获取与过滤。

- **MPEG21 标准**: 将不同协议、标准、技术结合在一起，制定新的标准，并将不同标准集成在一起，实际上就是一些关键技术的集成。
- **MPEG 家族总结**

- MPEG1/2 提供对于视频已频频的压缩方法与技术；
- MPEG4 解决对图像中具体内容的表示；
- MPEG7 实现对各种具体内容的描述；
- MPEG21 提供一种交互式多媒体操作框架。
- 当前正在进行制作的技术：MPEG-I 标准（沉浸式）

**AVS 标准**: 面向高清晰度电视、高密度光存储，针对 MPEG-2 进行改进。

- **重点技术**

- 帧间编码：可变块大小的运动补偿预测技术；
- 帧内编码：多方向的空间预测技术；
- 环内滤波器：去除块效应；
- $8 \times 8$  的整数正交变换及相应的量化策略；
- 改进的运动矢量预测编码；
- 更加高效的熵编码器；
- 基于率失真的编码优化技术。

- **主要技术对比**: 如图 88 所示。

- **目前成果**: 编码效率高；与 MPEG-4 AVC 与 H264 效率相当或更高，并一定程度兼容；
- **优势**: 方案简洁，复杂度低；与 H264/MPEG-4 AVC 相比，解码器复杂度降低到 70%，编码器复杂度降低到 30%；知识产权清晰（融合公共知识与新技术、必要专利数量不到相应国际标准的一半、大多数专利由中国会员贡献）。

编码工具	AVS	H.264	MPEG-2
帧内预测	基于 $8 \times 8$ 块，5种亮度预测模式，4种色度预测模式	基于 $4 \times 4$ 块，9种亮度预测模式，4种色度预测模式	只在频域内进行DC系数差分预测
多参考帧预测	最多2帧	最多16帧	只有1帧
变块大小运动补偿	$16 \times 16, 16 \times 8, 8 \times 16$ $8 \times 8$	$16 \times 16, 16 \times 8, 8 \times 16$ $8 \times 8, 8 \times 4, 4 \times 8, 4 \times 4$	$16 \times 16$ , $16 \times 8$ (场编码)
B帧宏块直接编码模式	时域空域相结合	独立的空域或时域预测模式	无
B帧宏块双向预测模式	对称预测模式，只编码一个前向运动矢量，后向运动矢量由前向导出	编码前后两个运动矢量	编码前后两个运动矢量

图 88

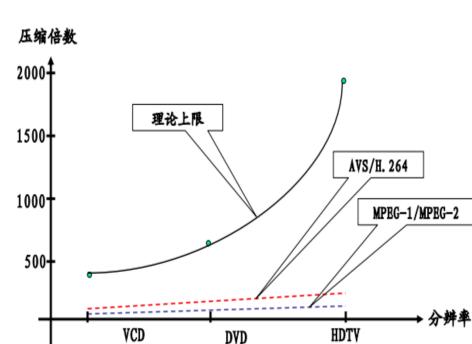


图 89

编码算法的未来发展：进一步提高压缩倍数，如图 89 所示。