数字图像处理

压缩编码

背景

- 在计算机图像处理系统中，图像的最大特点和难点就是海量数据的表示与传输，因此如何有效快速地存储和传输这些图像数据成为当今信息社会的迫切需求。
- 图像数据的压缩技术从总体上来说就是利用图像数据固有的冗余性和相关性，将一个大的图像数据文件转换成较小的同性质的文件。

图像的特点

- 数据量大，为其存储、传输带来困难，需压缩

- 例子：电话线传输速率一般为56kbit/s（波特率）
  一幅彩色图像640×480×24bit = 7Mbit大小
  - 传输一幅图像：时间约2分钟左右；
  - 实时传送：640×480×24bit×25帧/s=175Mbit/s，时间为50min左右；
  - 如压缩20倍，传一幅图6s左右，可以接受，实用

图像压缩编码的必要性

- 图像的数据量特别大，同时现在对图像需求的增长超过了网络带宽的限制，所以压缩是图像传输和存储的一个关键技术
- 由于图像压缩的巨大商业潜力，激励着人们提高现有的技术或发现新的技术

图像压缩编码的必要性

视频大小：1280×720
帧率：29.915fps
持续时间：30.854s
总帧数：923f
原始码流：661672kbit/s
视频平均码流： 4644kbit/s
压缩比：142.5

- AVI（Audio Video Interleave）
  - 同样是以AVI为后缀的视频文件，其采用的压缩算法可能不同，需要相应的解压软件才能识别和回放该AVI文件
  - AVI文件目前主要应用在多媒体光盘上，用来保存电影、电视等各种影像信息，有时也出现在Internet上，供用户下载、欣赏新影片的精彩片断。

图像压缩编码的必要性

视频大小：640×480
帧率：25fps
持续时间：32.04s
总帧数：801f
原始码流：184320kbit/s
视频平均码流： 847kbit/s
压缩比：217.6

- ASF
  - ADVANCED STEAMING FORMAT的缩写。
  - 它是一种采用流式传输方式在Internet播放的媒体格式，它可以将整个媒体文件分压成一个个的数据包，再由视频服务器向用户计算机进行连续、实时的传送。

图像压缩编码的必要性

视频大小：640×480
帧率：23.976fps
持续时间：40.04s
总帧数：960f
原始码流：176770kbit/s
视频平均码流： 828kbit/s
压缩比：213.5

- WMV
  - ASF格式升级延伸来得。在同等视频质量下，WMV格式的体积非常小，因此很适合在网上播放和传输。
  - WMV格式的主要优点包括：本地或网络回放、可扩充的媒体类型、部件下载、可伸缩的媒体类型、流的优先级化、多语言支持、环境独立性、丰富的流间关系以及扩展性等 。

图像压缩编码的必要性

- RM格式：
  - Real Networks公司所制定的音频视频压缩规范称为Real Media，主要用来在低速率的广域网上实时传输活动视频影像，可以根据网络数据传输速率的不同而采用不同的压缩比率，从而实现影像数据的实时传送和实时播放。
- 优点：可以把比较大的电影压缩成比较小的文件。
- 缺点：文件模糊不清。

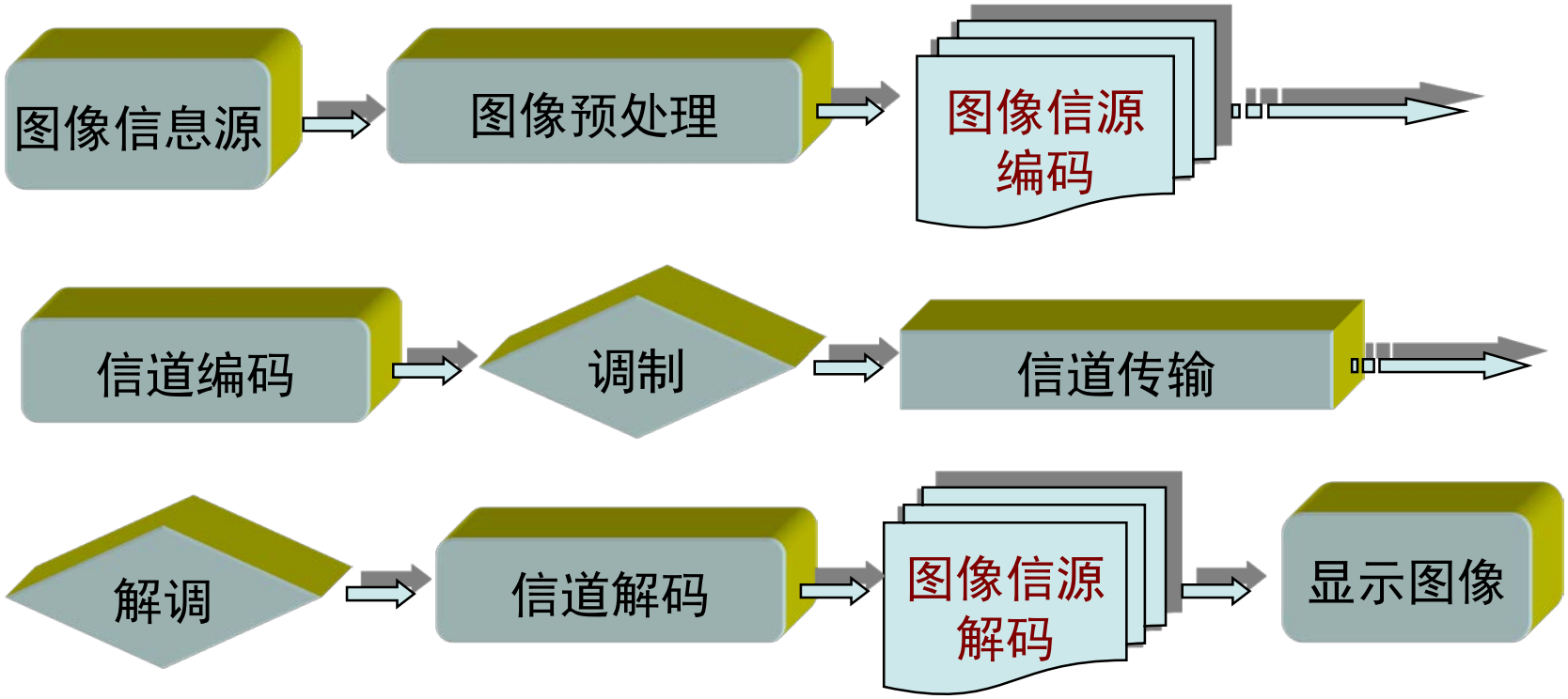
图像压缩编码的必要性

- RMVB格式：
  - 由RM视频格式升级延伸出的新视频格式，比RM多了VB两字，在这里VB是VBR（Variable Bit Rate--可变比特率）的缩写。
  - 打破了原先RM格式那种平均压缩采样的方式，在保证平均压缩比的基础上合理利用比特率资源，在保证了静止画面质量的前提下，大幅地提高了运动图像的画面质量。
  - 要想播放这种视频格式，可以使用RealOne Player2.0或RealPlayer8.0加RealVideo9.0以上版本的解码器形式进行播放。

图像压缩编码的必要性

- MOV格式：
  - MOV是美国Apple公司开发的一种视频格式，播放器是QuickTime Player。具有较高的压缩比率和较完美的视频清晰度等特点。
  - MOV也可以作为一种流文件格式。QuickTime能够通过Internet提供实时的数字化信息流、工作流与文件回放功能

图像通信系统模型



图像通信系统模型

图像通信系统模型

- 图像压缩编码系统：
  - 图像编码: 对图像信息进行压缩和编码，在存储、处理和传输前进行，也称图像压缩；
  - 图像解码：对压缩图像进行解压以重建原图像或其近似图像。

图像压缩编码的必要性

- 大数据量的图像信息会给存储器的存储容量、通信干线信道的带宽以及计算机的处理速度增加极大的压力。
- 单纯靠增加存储器容量，提高信道带宽以及计算机的处理速度等方法来解决这个问题是不现实的，这时就要考虑压缩。
- 因此，图像数据在传输和存储中，数据的压缩都是必不可少的。

图像压缩编码的必要性

- 数据压缩的研究内容包括数据的表示、传输、变换和编码方法，目的是减少存储数据所需的空间和传输所用的时间。
- 图像编码与压缩就是对图像数据按一定的规则进行变换和组合，达到以尽可能少的代码（符号）来表示尽可能多的图像信息。

图像压缩编码的可能性

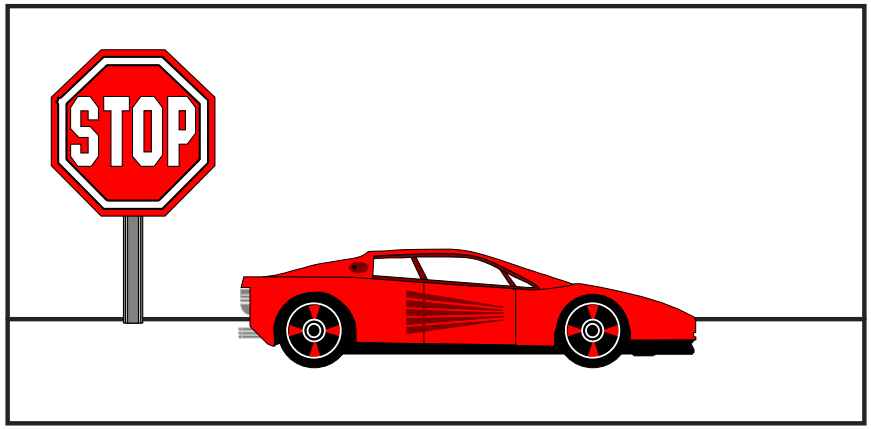
- 数字图像本身的特征带来的数据压缩的可能性
- 图像数据是高度相关的， 或者说存在冗余信息，去掉这些冗余信息后可以有效压缩图像， 同时又不会损害图像的有效信息。
- 数字图像的冗余主要表现为以下几种形式：
  - 空间冗余
  - 时间冗余
  - 视觉冗余
  - 信息熵冗余
  - 结构冗余等。

数字图像的冗余

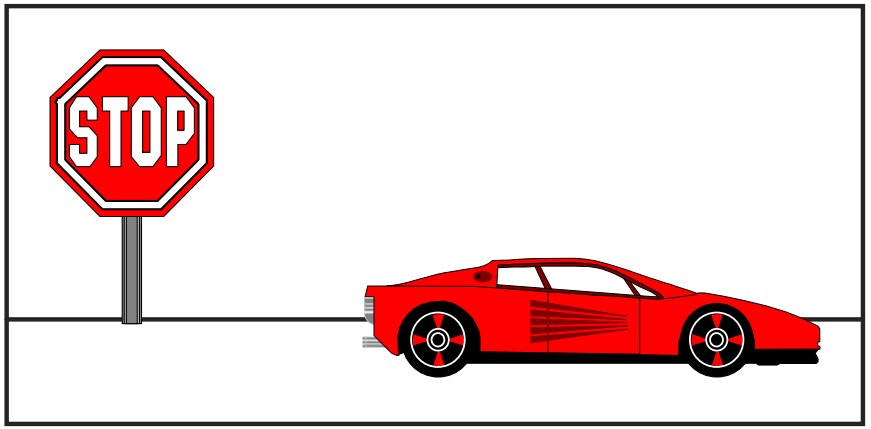
- 空间冗余（像素间冗余、几何冗余）
  - 这是图像数据中所经常存在的一种冗余。
  - 在同一幅图像中，规则物体和规则背景（所谓规则是指表面是有序的而不是完全杂乱无章的排列）的表面物理特性具有相关性，这些相关性的光成像结果在数字化图像中就表现为数据冗余。
- 时间冗余
  - 在序列图像（电视图像、运动图像）中，相邻两帧图像之间有较大的相关性。

时间冗余

- 时间冗余示例



(a) F1帧



(b) F2帧

如图所示，F1帧中有一个小汽车和一个路标，在时间T后的F2图像中仍包含以上两个物体，只是小车向前行驶了一段路程，此时F1和F2中的路标和背景都是时间相关的，小车也是时间相关的，因而F2和F1具有时间冗余。

数字图像的冗余

- 知识冗余
  - 有许多图像的理解与某些基础知识有相当大的相关性，例如人脸的图像有固定的结构，比如说嘴的上方有鼻子，鼻子的上方有眼睛，鼻子位于正脸图像的中线上等等，这类规律性的结构可由先验知识和背景知识得到，称此类冗余为知识冗余。

数字图像的冗余

- 信息熵冗余
  - 也称为编码冗余，如果图像中平均每个像素使用的比特数大于该图像的信息熵，则图像中存在冗余，称为信息熵冗余。
- 结构冗余
  - 有些图像存在较强的纹理结构，如墙纸、草席等图像，称其存在结构冗余。

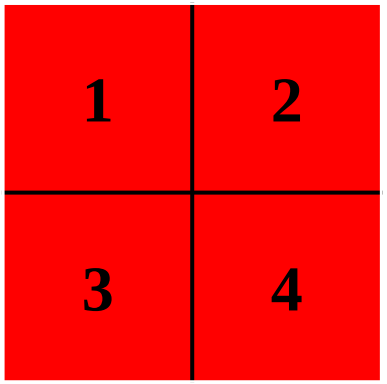
数字图像的冗余

- 心理视觉冗余
  - 人类的视觉系统对于图像场的注意是非均匀和非线性的，特别是视觉系统并不是对于图像场的任何变化都能感知，即眼睛并不是对所有信息都有相同的敏感度，有些信息在通常的视觉感觉过程中与另外一些信息相比来说并不那么重要，这些信息可认为是心理视觉冗余的，去除这些信息并不会明显地降低所感受到的图像的质量。
  - 心理视觉冗余的存在是与人观察图像的方式有关的，人在观察图像时主要是寻找某些比较明显的目标特征，而不是定量地分析图像中每个像素的亮度，或至少不是对每个像素等同地分析，人通过在脑子里分析这些特征并与先验知识结合以完成对图像的解释过程，由于每个人具有的先验知识不同，对同一幅图像的心理视觉冗余也就因人而异。

图像压缩编码的可能性

- 图像能量在变换域内分布的不均匀性
  - 例如：大部分能量集中在低频部分，而小部分能量集中在高和较高的频率部分。
  - 应用环境允许图像有一定程度失真
    - 接受端图像设备分辨率较低，则可降低图像分辨率。
    - 根据人的视觉特性对不敏感区进行降低分辨率编码。
    - 用户所关心的图像区域有限，可对其余部分图像采用空间和灰度级上的粗化。

图像压缩编码的可能性

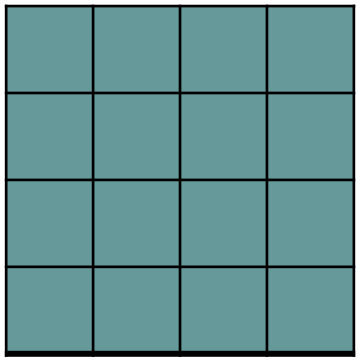


“这是一幅2×2的图像，图像的第一个像素是红的，第二个像素是红的，第三个像素是红的，第四个像素是红的”。

“这是一幅2×2的图像，整幅图都是红色的”。

图像压缩编码的可能性

- 图像无损压缩的原理



RGB	RGB	RGB	RGB
RGB	RGB	RGB	RGB
RGB	RGB	RGB	RGB
RGB	RGB	RGB	RGB

16	RGB

从原来的16×3×8bit=284bit
压缩为：(1+3)×8bit=32bit

图像压缩编码的可能性

- 图像有损压缩的原理

36	35	34	34	34
34	34	32	34	34
33	37	30	34	34
34	34	34	34	34
34	35	34	34	31

34	34	34	34	34
34	34	34	34	34
34	34	34	34	34
34	34	34	34	34
34	34	34	34	34

25	34

图像压缩编码的分类

- 根据解压重建后的图像和原始图像之间是否具有误差，图像编码压缩分为无损（亦称无失真、无误差、信息保持型）编码和有损（有失真、有误差、信息非保持型）编码两大类。
  - 无损压缩：这类压缩算法中删除的仅仅是图像数据中冗余的信息，因此在解压缩时能精确恢复原图像。无损压缩用于要求重建后图像严格地和原始图像保持相同的场合，例如复制、保存十分珍贵的历史、文物图像等。
  - 有损压缩：这类算法把不相干的信息也删除了，因此在解压缩时只能对原始图像进行近似的重建，而不能精确的复原，有损压缩适合大多数用于存储数字化了的模拟数据。

图像压缩编码的分类

- 根据编码原理，图像编码分为
  - 熵编码
  - 预测编码
  - 变换编码
  - 混合编码
  - ……

图像压缩编码的分类

- 熵编码：这是纯粹基于信号统计特性的编码技术，是一种无损编码。熵编码的基本原理是给出现概率较大的符号赋予一个短码字，而给出现概率较小的符号赋予一个长码字，从而使最终的平均码长很小。常见的熵编码方法有哈夫曼编码、算术编码和行程编码。
- 预测编码：它是基于图像数据的空间或时间冗余特性，用相邻的已知像素（或像素块）来预测当前像素（或像素块）的取值，然后再对预测误差进行量化和编码。预测编码可分为帧内预测和帧间预测，常用的预测编码有差分脉码调制（Differential Pulse Code Modulation， DPCM）和运动补偿法。

图像压缩编码的分类

- 变换编码：通常是将空间域上的图像经过正交变换映射到另一变换域上，使变换后的系数之间的相关性降低。图像变换本身并不能压缩数据，但变换后图像的大部分能量只集中到少数几个变换系数上，再采用适当的量化和熵编码就可以有效压缩图像。
- 混合编码：是指综合了熵编码、变换编码或预测编码的编码方法，如JPEG标准和MPEG标准。

图像压缩编码的分类

- 从图像的光谱特征出发，将压缩编码分为
  - 单色图像编码
  - 彩色图像编码
  - 多光谱图像编码

- 从图像的灰度层次上，压缩编码可分为
  - 多灰度编码
  - 二值图像编码

图像压缩编码的系统评价

- 图像编码的目标
  - 在图像编码中，编码质量是一个非常重要的概念，怎样以尽可能少的比特数来存储或传输一幅图像，同时又让接收者感到满意，这是图像编码的目标。
- 图像编码的质量评价
  - 基于压缩编码参数的评价
  - 基于保真度（逼真度）准则的评价
  - 算法的适用范围
  - 算法的复杂度

主观保真度准则

- 一种常用的方法是让一组（不少于20人）观察者观察图像并给该图像评分，将他们对该图像的评分取平均，作为这幅图像的质量。
  - 损伤程度：不能察觉、刚察觉、不讨厌、有点讨厌、很讨厌，不能用；
  - 质量：优、良、中、次、劣；
  - 比较：+2，好的多；+1，好；0，同；-1，坏；-2 坏的多。

主观保真度准则

- 主观（人判别）专家投票的方法，实用方法：
  - 人的视觉的主观亮度是光强的对数函数。
  - 人眼对黑暗区误差比明亮区更敏感。
  - 人眼对灰度突变边缘比较敏感。

基于压缩编码参数的评价

- 图像的熵与平均码字长度
  - 令图像像素灰度级集合为$\{l_1, l_2, \ldots, l_m\}$，其对应的概率分别为$p(l_1), p(l_2), \ldots, p(l_m)$，图像熵定义为：

$$H = -\sum_{i=1}^{m} p(l_i)\log_2 p(l_i)$$

单位为比特/字符，图像熵表示图像灰度级集合的比特数均值，或者说描述了图像信源的平均信息量。

基于压缩编码参数的评价

- 图像的熵与平均码字长度
  - 当灰度级集合$\{l_1, l_2, \ldots, l_m\}$中$l_i$出现的概率相等，都为$2^{-L}$时，熵$H$最大，等于$L$比特；只有当$l_i$出现的概率不相等时，熵$H$才会小于$L$比特。
  - 香农信息论已经证明：信源熵是进行无失真编码的理论极限，低于此极限的无失真编码方法是不存在的，这是熵编码的理论基础。
  - 平均码长定义为：

$$R = \sum_{i=1}^{m} n_i \, p(l_i)$$

其中，n_i为灰度级l_i所对应的码字长度，平均码长的单位也是比特 / 字符。

编码效率

- 编码效率定义为：

$$\eta = \frac{H}{R}$$

  - 如果R和H相等，编码效果最佳；
  - 如果R和H接近，编码效果为佳；
  - 如果R远大于H，则编码效果差。

编码效率

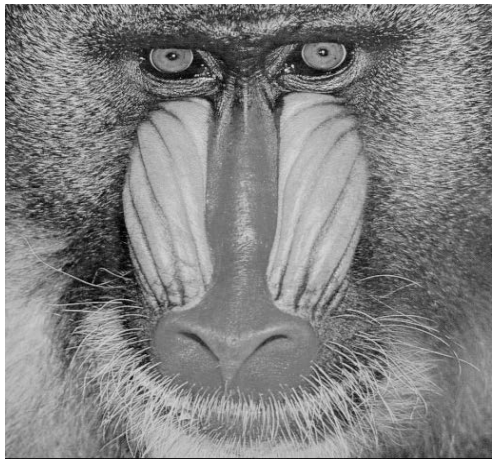
- 由于同一图像压缩编码算法对不同图像的编码效率往往不同。为了公平地衡量图像压缩编码算法的效率，常常需要定义一些所谓的“标准图像”。
- 通过测量不同图像编码算法在同一组“标准图像”上的性能来评价各图像压缩算法的编码效率。

编码效率

- 下图给出了国际上流行的三幅图像Lena、Barbara和Mandril。
  - 图(a)头发部分高频数据含量丰富，背景含低频数据，肩部亮度过渡平滑；
  - 图(b)低频区域含量适中，但物体边缘丰富，头巾、裤子及桌布上有极细腻的条纹；
  - 图(c)高频数据极为丰富，特别是脸部毛发部分，主要用于评价图像编码算法对高频区域数据的处理性能。

(a) Lena

(b) Barbara



(c) Mandrill

压缩比

- 压缩比是衡量数据压缩程度的指标之一，到目前为止，尚无压缩比的统一定义，目前常用的压缩比P_r定义为：

$$P_r = \frac{L_s - L_d}{L_s} \times 100\%$$

其中：L_s为源代码长度；L_d为压缩后的代码长度。

- 压缩比的物理意义是被压缩掉的数据占源数据的百分比。
  - 一般来讲，压缩比大，则说明被压缩掉的数据量多；当压缩比接近100%时，压缩效率最理想。

冗余度

- 如果编码效率$\eta \neq 100\%$，就说明还有冗余度，冗余度r定义为：

$$r = 1 - \eta$$

  - 冗余度r越小，说明可压缩的余地越小。
- 总之，一个编码系统要研究的问题是设法减小编码平均长度R，使编码效率尽量趋于1，而冗余度尽量趋于0。

例1：有一信源符号为6的信源，其编码如下表所示。求信源熵、平均码长、编码效率及冗余度。

符号	×1	×2	×3	×4	×5	×6
概率	0.25	0.25	0.20	0.15	0.10	0.05
码字	01	10	11	000	0010	0011

解答

$$H(x) = -0.25\log_2^{0.25} - 0.25\log_2^{0.25} - 0.20\log_2^{0.20}$$
$$-0.15\log_2^{0.15} - 0.10\log_2^{0.10} - 0.05\log_2^{0.05} = 2.42\text{比特/符号}$$

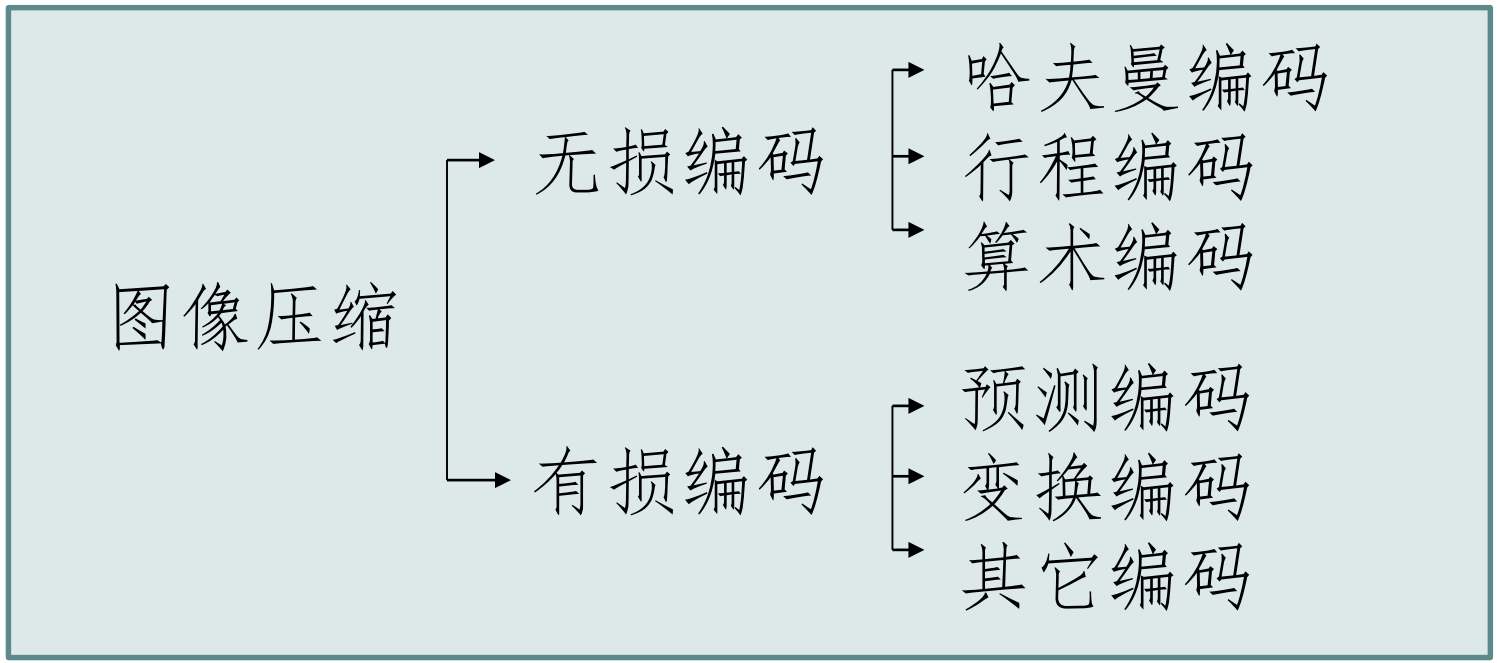
$$R = 2\times0.25 + 2\times0.25 + 2\times0.20$$
$$+3\times0.15 + 4\times0.10 + 4\times0.05 = 2.45\text{比特/符号}$$

$$\eta = \frac{H}{R}\times100\% = \frac{2.42}{2.45}\times100\% = 98.8\%$$

$$r = 1 - \eta = 1 - 98.8\% = 1.2\%$$

图像压缩编码的分类

- 根据解压重建后的图像和原始图像之间是否具有误差，图像编码压缩分为无误差（亦称无失真、无损、信息保持）编码和有误差（有失真或有损）编码两大类。



哈夫曼编码

- 哈夫曼编码的理论基础
  - 变长最佳编码定理

> 在变长编码中，对出现概率大的信息符号赋予短码字，而对于出现概率小的信息符号赋予长码字。如果码字长度严格按照所对应符号出现概率大小逆序排列，则编码结果平均码字长度一定小于任何其它排列方式。即平均码字长度为最小.

  - 该定理是哈夫曼编码的理论基础。

哈夫曼编码算法

哈夫曼编码算法	
步骤1：	统计信源中各符号出现的概率， 按符号出现的概率从大到小排序；
步骤2：	把最小的两个概率相加合并成新的概率， 与剩余的概率组成新的概率集合；
步骤3：	对新的概率集合重新排序， 再次把其中最小的两个概率相加， 组成新的概率集合。如此重复进行， 直到最后两个概率的和为1；
步骤4：	分配码字。码字分配从最后一步开始反向进行， 对每次相加的两个概率，给大的赋“0”，小的赋“1”如果两个概率相等，则任选一个赋“0”，另一个赋“1”。

例2 设一幅灰度级为8（分别用S_0、S_1、S_2、S_3、S_4、S_5、 S_6、S_7表示）的图像中，各灰度所对应的概率分别为0.40、0.18、 0.10、 0.10、 0.07、 0.06、 0.05、0.04。现对其进行哈夫曼编码。

具体步骤如下：

(1) 首先对信源概率从大到小排序，选出最小的两个概率（0.04和0.05），相加得0.09，与其他概率组成新的概率集合{0.40，0.18，0.10， 0.10，0.07，0.06，0.09}。

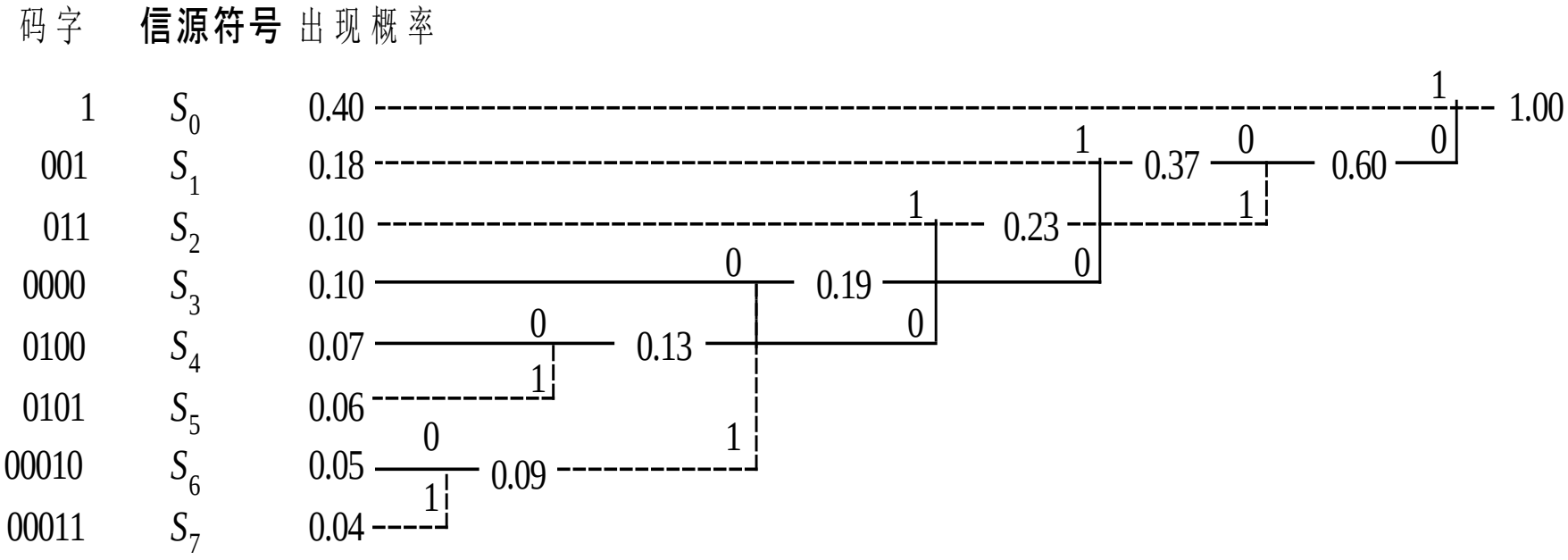
(2) 对新的概率集合重新排序，选出最小的两个概率（0.06和0.07），相加得0.13，直到最后两个概率（0.60和0.40）相加和为1。

例2 设一幅灰度级为8（分别用S_0、S_1、S_2、S_3、S_4、S_5、 S_6、S_7表示）的图像中，各灰度所对应的概率分别为0.40、0.18、 0.10、 0.10、 0.07、 0.06、 0.05、0.04。现对其进行哈夫曼编码。

具体步骤如下：

(3) 分配码字。从最后一步反向进行， 首先给最后相加的两个概率（0.60和0.40）分配码字，由于0.60大于0.40，于是给0.60赋“0”，给0.40赋“1”。如此依次给每次相加的两个概率分配码字。

(4) 最后写出每个符号的哈夫曼编码。


码字
信源符号
出现概率
1
001
011
0000
0100
0101
00010
00011
S_0
S_1
S_2
S_3
S_4
S_5
S_6
S_7
0.40
0.18
0.10
0.10
0.07
0.06
0.05
0.04
0.09
0.13
0.19
0.23
0.37
0.60
1.00

哈夫曼编码的编码效率

- 平均码长R为

$$R = \sum_{k=1}^{N} B_k P_k$$

$$= 1\times 0.40 + 3\times 0.18 + 3\times 0.10 + 4\times 0.10 + 4\times 0.07$$

$$+ 4\times 0.06 + 5\times 0.05 + 5\times 0.04$$

$$= 2.61$$

哈夫曼编码的编码效率

- 数字图像的熵H为

$$H = -\sum_{k=1}^{N} P_k 1bP_k$$

$$= -(0.4 \times 1b0.4 + 0.18 \times 1b0.18 + 0.1 \times 1b0.1 + 0.1 \times 1b0.1$$

$$+ 0.07 \times 1b0.4 + 0.06 \times 1b0.06 + 0.05 \times 1b0.05 + 0.04 \times 1b0.04)$$

$$= 2.55$$

则哈夫曼编码的编码效率η为

$$\eta = \frac{H}{R} \times 100\% = \frac{2.55}{2.61} \times 100\% = 97.8\%$$

由此可见，哈夫曼编码的编码效率是相当高的，其冗余度只有2. 2%。如果采用等长编码，由于有8种灰度级，则每种灰度级别至少需要3比特来表示，对于例中的图像而言，其编码的平均码长为3，编码效率为85%。

练习1：设有一个信源数据序列，包括A、B、C、D、E五个符号，它们出现的概率分别是0.40，0.18，0.15，0.15和0.12，对其进行哈夫曼编码，并求编码效率。

哈夫曼编码的编码效率

- 哈夫曼编码在不同概率分布下的编码效果对比

信源符号	概率分布为不均匀			概率分布为 2 的负幂次方			概率分布为均匀分布		
	出现概率	码字	码长	出现概率	码字	码长	出现概率	码字	码长
l_1	0.40	1	1	2^{-1}	1	1	2^{-3}	111	3
l_2	0.18	001	3	2^{-2}	01	2	2^{-3}	110	3
l_3	0.10	011	3	2^{-3}	001	3	2^{-3}	101	3
l_4	0.10	0000	4	2^{-4}	0001	4	2^{-3}	100	3
l_5	0.07	0100	4	2^{-5}	00001	5	2^{-3}	011	3
l_6	0.06	0101	4	2^{-6}	000001	6	2^{-3}	010	3
l_7	0.05	00010	5	2^{-7}	0000001	7	2^{-3}	001	3
l_8	0.04	00011	5	2^{-7}	0000000	7	2^{-3}	000	3
三个参数	H=2.55 R=2.61 η=97.8%			H=1.984 R=1.984 η=100%			H=3 R=3 η=100%		

香农-范诺编码

- 香农-范诺编码：
  - 香农-范诺编码也是一种常见的可变字长编码。
  - 基础是符号的码字长度N_i完全由该符号出现的概率来决定，即

$$-\log_2 P_i \le N_i \le -\log_2 P_i + 1$$

香农-范诺编码的步骤

香农-范诺编码算法

步骤1：将信源符号按其出现概率从大到小排序；

步骤2：计算出各概率对应的码字长度N_i；

步骤3：计算累加概率A_i，即

$$A_i = A_{i-1} + P_{i-1}, i = 1, 2, \ldots, N-1; \quad A_0 = 0$$

步骤4：把各个累加概率A_i由十进制转化为二进制，取该二进制数的前N_i位作为对应信源符号的码字。

例3. 设一幅灰度级为8（分别用S_0、S_1、S_2、S_3、S_4、S_5、 S_6、S_7表示）的图像中，各灰度所对应的概率分别为0.40、0.18、0.10、 0.10、 0.07、 0.06、 0.05、0.04。现对其进行香农-范诺编码。

P排序	0.4	0.18	0.1	0.1	0.07	0.06	0.05	0.04
码字N_i	2	3	4	4	4	5	5	5
P累加	0	0.4	0.58	0.68	0.78	0.85	0.91	0.96
二进制	0	11	1001	1010	1100	11011	11101	11110

香农—范诺编码

信源符号	出现概率 P_i	码字长度 N_i	累加概率 A_i	转换为二进制	分配码字 B_i
l_1	0.40	2	0	0	00
l_2	0.18	3	0.40	0110	011
l_3	0.10	4	0.58	10010	1001
l_4	0.10	4	0.68	10100	1010
l_5	0.07	4	0.78	11000	1100
l_6	0.06	5	0.85	110110	11011
l_7	0.05	5	0.91	111010	11101
l_8	0.04	5	0.96	111101	11110
平均码长 R=3.17		图像熵 H=2.55		编码效率 η = 80.4%	

二分法香农-范诺编码

二分法香农-范诺编码算法
步骤1：首先统计出每个符号出现的概率；
步骤2：从左到右对上述概率从大到小排序；
步骤3：将概率集合从某个位置分为两个子集合，尽量使两个子集合的概率和近似相等，前面子集合赋值为0，后面为1；
步骤4：重复步骤3，直到各个子集合中只有一个元素为止；
步骤5：将每个元素所属的子集合的值依次串起来即可。

例3 设一幅灰度级为8（分别用S_0、S_1、S_2、S_3、S_4、S_5、S_6、S_7表示）的图像中，各灰度所对应的概率分别为0.40、0.18、0.10、0.10、0.07、0.06、0.05、0.04。现对其进行二分香农-范诺编码。

码字	符号	出现概率				
00	S_0	0.40	0.58(0)	0.40(0)		
01	S_1	0.18		0.18(1)		
100	S_2	0.10	0.42(1)	0.20(0)	0.10(0)	
101	S_3	0.10			0.10(1)	
1100	S_4	0.07		0.21(1)	0.13(0)	0.07(0)
1101	S_5	0.06				0.06(1)
1110	S_6	0.05			0.09(1)	0.05(0)
1111	S_7	0.04				0.04(1)
平均码长 $R=2.64$		图像熵 $H=2.55$		编码效率 $\eta=96.59\%$		

练习题

- 设有一个信源数据序列，包括A、B、C、D、E五个符号，它们出现的概率分别是0.40，0.18，0.15，0.15和0.12，对其进行香农-范诺编码，并求编码效率。

算术编码

- 算术编码是80年代发展起来的一种熵编码方法，
  - 这种方法不是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线上的0到1之间的一个区间，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数，转化为二进制作为实际的编码输出。
  - 消息序列中的每个元素都要缩短为一个区间，消息序列中元素越多，所得到的区间就越小。
  - 当区间变小时，就需要更多的数位来表示这个区间，采用算术编码，每个符号的平均编码长度可以为小数。

算术编码

- 算术编码有两种模式
  - 一种是基于信源概率统计特性的固定编码模式；
  - 另一种是针对未知信源概率模型的自适应模式。
    - 自适应模式中各个符号的概率初始值都相同，它们依据出现的符号而相应地改变。只要编码器和解码器都使用相同的初始值和相同的改变值的方法，那么它们的概率模型将保持一致。
  - 上述两种形式的算术编码均可用硬件实现，其中自适应模式适用于不进行概率统计的场合。有关实验数据表明，在未知信源概率分布的情况下，算术编码一般要优于Huffman编码。在JPEG扩展系统中，就用算术编码取代了哈夫曼编码。

下面结合一个实例来阐述固定模式的算术编码的具体方法。

例：设一待编码的数据序列（即信源）为“cadacdb”，信源中各符号出现的概率依次为 $P(a)=0.1$，$P(b)=0.4$，$P(c)=0.2$，$P(d)=0.3$。

首先，数据序列中的各数据符号在区间$[0,1]$内的间隔（赋值范围）设定为：

$$a=[0,0.1),b=[0.1,0.5),c=[0.5,0.7),d=[0.7,1.0)$$

算术编码所依据的公式为：

$$\begin{cases}\text{Start}_N=\text{Start}_B+L\times\text{Left}_C\\ \text{End}_N=\text{Start}_B+L\times\text{Right}_C\end{cases}$$

$Start_N$、End_N分别表示新间隔（或称之为区间）的起始位置和结束位置，$Start_B$表示前一间隔的起始位置，L为前一间隔的长度，$Left_C$、$Right_C$分别表示当前编码符号的初始区间的左端和右端。

初始化时，$Start_B = 0$，$L = 1.0 - 0 = 0$。

①对第一个信源符号c编码：

$$Start_N = Start_B + L \times Left_C = 0 + 1 \times 0.5 = 0.5$$

$$End_N = Start_B + L \times Right_C = 0 + 1 \times 0.7 = 0.7$$

信源符号c将区间$[0,1) \rightarrow [0.5,0.7)$；

下一个信源的范围为：$L = End_N - Start_N = 0.7 - 0.5 = 0.2$

②对第二个信源符号a 编码：

$$Start_N = Start_B + L \times Left_C = 0.5 + 0.2 \times 0 = 0.5$$

$$End_N = Start_B + L \times Right_C = 0.5 + 0.2 \times 0.1 = 0.52$$

信源符号a将区间[0.5,0.7)→[0.5,0.52)；

下一个信源的范围为：$L = End_N - Start_N = 0.52 - 0.5 = 0.02$

③对第三个信源符号d 编码：

$$Start_N = Start_B + L \times Left_C = 0.5 + 0.02 \times 0.7 = 0.514$$

$$End_N = Start_B + L \times Right_C = 0.5 + 0.02 \times 1 = 0.52$$

信源符号d将区间[0.5,0.52)→[0.514,0.52)；

下一个信源的范围为：$L = End_N - Start_N = 0.52 - 0.514 = 0.006$

④对第四个信源符号a 编码：

$$\mathrm{Start_N} = \mathrm{Start_B} + \mathrm{L} \times \mathrm{Left_C} = 0.514 + 0.006 \times 0 = 0.514$$

$$\mathrm{End_N} = \mathrm{Start_B} + \mathrm{L} \times \mathrm{Right_C} = 0.514 + 0.006 \times 0.1 = 0.5146$$

信源符号a将区间[0.514,0.52)→[0.514,0.5146)；

下一个信源的范围为：

$$L = End_N - Start_N = 0.5146 - 0.514 = 0.0006$$

⑤对第五个信源符号c编码：

$$\mathrm{Start_N} = \mathrm{Start_B} + \mathrm{L} \times \mathrm{Left_C} = 0.514 + 0.0006 \times 0.5 = 0.5143$$

$$\mathrm{End_N} = \mathrm{Start_B} + \mathrm{L} \times \mathrm{Right_C} = 0.514 + 0.0006 \times 0.7 = 0.51442$$

信源符号c将区间[0.514,0.5146)→[0.5143,0.51442)；

下一个信源的范围为：

$$L = End_N - Start_N = 0.51442 - 0.5143 = 0.00012$$

⑥对第六个信源符号d 编码：

$$Start_N = Start_B + L \times Left_C = 0.5143 + 0.00012 \times 0.7 = 0.514384$$

$$End_N = Start_B + L \times Right_C = 0.5143 + 0.00012 \times 1 = 0.51442$$

信源符号d将区间[0.5143,0.51442)→[0.514384,0.51442)；

下一个信源的范围为：

$$L = End_N - Start_N = 0.51442 - 0.514384 = 0.000036$$

⑦对第七个信源符号b 编码：

$$Start_N = Start_B + L \times Left_C = 0.514384 + 0.000036 \times 0.1 = 0.5143876$$

$$End_N = Start_B + L \times Right_C = 0.514384 + 0.000036 \times 0.5 = 0.514402$$

信源符号b将区间[0.514384,0.51442)→[0.5143876,0.514402)；

最后，从[0.5143876,0.514402)中选择一个数作为编码输出，

选择0.5143876。

解码是编码的逆过程，通过编码最后的下标界值0.5143876得到信源“cadacdb”是唯一的编码。

由于0.5143876在［0.5，0.7］区间，所以可知第一个信源符号为c。

得到信源符号c以后，由于已知信源符号c的上界和下界，利用编码的可逆性，减去信源符号c的下界0.5，得0.0143876，再用信源符号c的范围0.2去除，得到0.071938，由于已知0.071938落在信源符号a的区间，所以得到第二个信源符号为a；同样，再减去信源符号a的下界0，除以信源a的范围0.1，得到0.71938，已知0.71938落在信源符号d的区间，所以得到第三个信源符号为d，……，同理操作下去，直至解码结束。

具体解码操作过程如下：

$$\frac{0.5143876-0}{1}=0.5143876\Rightarrow c$$

$$\frac{0.0646-0}{0.1}=0.646\Rightarrow c$$

$$\frac{0.5143876-0.5}{0.2}=0.071938\Rightarrow a$$

$$\frac{0.646-0.5}{0.2}=0.73\Rightarrow d$$

$$\frac{0.071938-0}{0.1}=0.71938\Rightarrow d$$

$$\frac{0.73-0.7}{0.3}=0.1\Rightarrow b$$

$$\frac{0.71938-0.7}{0.3}=0.0646\Rightarrow a$$

$$\frac{0.1-0.1}{0.4}=0\Rightarrow \text{结束}$$

行程编码

- 行程编码（Run-length Coding）是相对简单的编码技术，主要思路是将一个相同值的连续串用一个代表值和串长来代替。
  - 例如有一个字符串“aaabccddddd”，经过行程编码后可以用“3a1b2c5d”来表示。对图像编码来说，可以定义沿特定方向上具有相同灰度值的相邻像素为一轮，其延续长度称之为延续的行程，简称为行程或游程。
  - 例如，若沿水平方向有一串M个像素具有相同的灰度N，则行程编码后，只传递两个值(N, M)就可以代替M个像素的M个灰度值N 。

行程编码

- 行程编码分为定长行程编码和变长行程编码两种。
  - 定长行程编码：编码的行程所使用的二进制位数固定，如果灰度连续相等的个数超过了固定二进制位数所能表示的最大值，则进行下一轮行程编码。
  - 变长行程编码：对不同范围的行程使用不同位数的二进制位数进行编码，需要增加标志位来表明所使用的二进制位数。

行程编码

- 行程编码一般不直接应用于多灰度图像，但比较适合于二值图像的编码。为了达到较好的压缩效果，有时行程编码和其他一些编码方法混合使用。
  - 例如，在JPEG中，行程编码和DCT及哈夫曼方法一起使用，先对图像分块处理，然后对分块进行DCT，量化后的频域图像数据作Z形扫描，然后作行程编码，对行程编码的结果再进行哈夫曼编码。
- 行程编码对传输差错很敏感，一位符号出错就会改变行程编码的长度，从而使整个图像出现偏移，因此一般要用行同步、列同步的方法把差错控制在一行一列之内。

预测编码

- 预测编码是建立在信号（语音、图像等）数据的相关性之上，根据某一模型利用以往的样本值对新样本进行预测，减少数据在时间和空间上的相关性，以达到压缩数据的目的。
- 但实际利用预测器时，并不是利用数据源的某种确定型数学模型，而是基于估计理论、现代统计学理论设计预测器。
- 预测方法有多种，其中差分脉冲编码调制(Differential Pulse Code Modulation)，简称DPCM，是一种具有代表性的编码方法。

DPCM基本原理

- 差值图像的统计特性
  - 由图像的统计特性可知，相邻像素之间有着较强的相关性，即相邻像素的灰度值相同或相近。因此，某像素的值可根据以前已知的几个像素值来估计、来猜测，正是由于像素间的相关性，才使预测成为可能。
  - 图像在扫描行方向（或称水平方向）相邻两像素的相关性是指:如果某像素的灰度值为h，则相邻它的上一个像素的灰度值可能性最大的也为h或$h+\Delta$，Δ为一个小量（如灰度级为256时，$\Delta=1$）。

DPCM基本原理

- 差值图像的统计特性
  - 一般来说相邻两像素灰度值突变的概率小，水平方向是如此，在图像的垂直方向也是如此。如果取一幅图像第i行的第j列像素的亮度离散值为$f(i,j)$，则：

$$\Delta_{水平} = f(i,j) - f(i,j-1)$$

$$\Delta_{垂直} = f(i,j) - f(i-1,j)$$

Δ称为差值信号。

DPCM基本原理

- 这表现在一幅图像中都含有亮度值恒定或变化很小的大面积区域，从而使差值信号的80%-90%落在约16-18个量化层（量化层总数为256时）中。
- 因此利用图像水平方向（或垂直方向）两个像素真实的离散幅度相减而得到它们的差值，然后对差值进行编码、传送就能达到压缩图像数据的目的，预测法的图像压缩编码就是在这基础上发展起来的。

DPCM基本原理

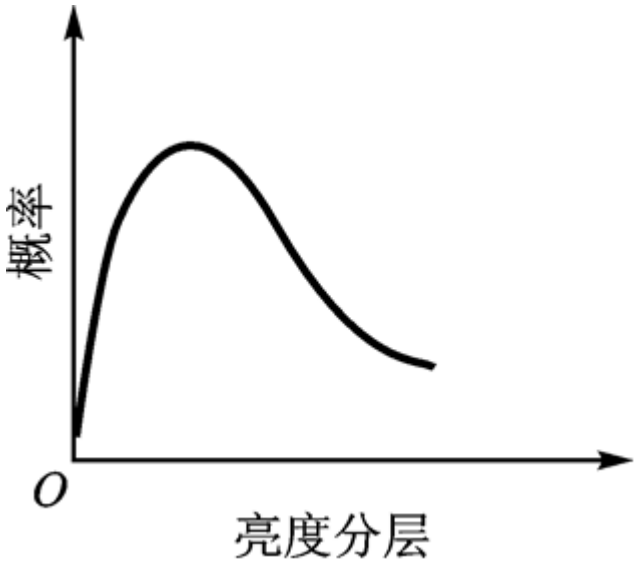


图1 自然景物及人物图像的直方图

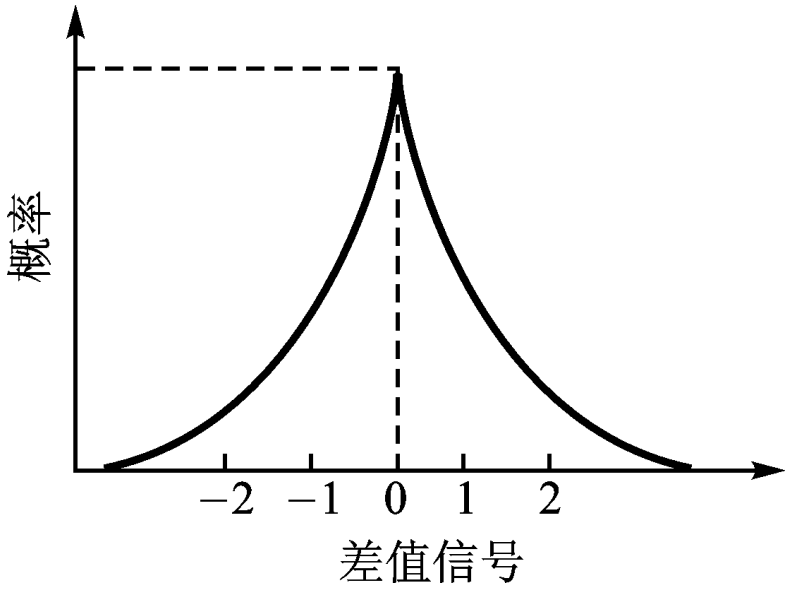


图2 图像差值信号的概率分布

从对大量的自然景物、人物图像的统计分析看出，亮度层次的的概率如图1所示。

经过对大量图像的差值信号统计，其概率分布有如图2所示形式。幅度差值愈大的差值信号出现的概率愈小，而零值或接近零值的差值信号出现的概率最大。

预测编码的基本原理

- 预测编码的基本思想是通过仅提取每个像素中的新信息并对它们编码来消除像素间的冗余，这里一个像素的新信息定义为该像素的当前或现实值与预测值的差，即如果已知图像一个像素离散幅度的真实值，利用其相邻像素的相关性，预测它的下一个像素（水平方向的或垂直方向的）的可能数值，再求其两者差，或者说利用这种具有预测性质的差值，再量化、编码、传输，其效果更佳，这一方法就称为DPCM法。
- 因此在预测法编码中，编码和传输的并不是像素取样值本身，而是这个取样值的预测值（也称估计值）与其实际值之间的差值。DPCM系统原理框图见图3。

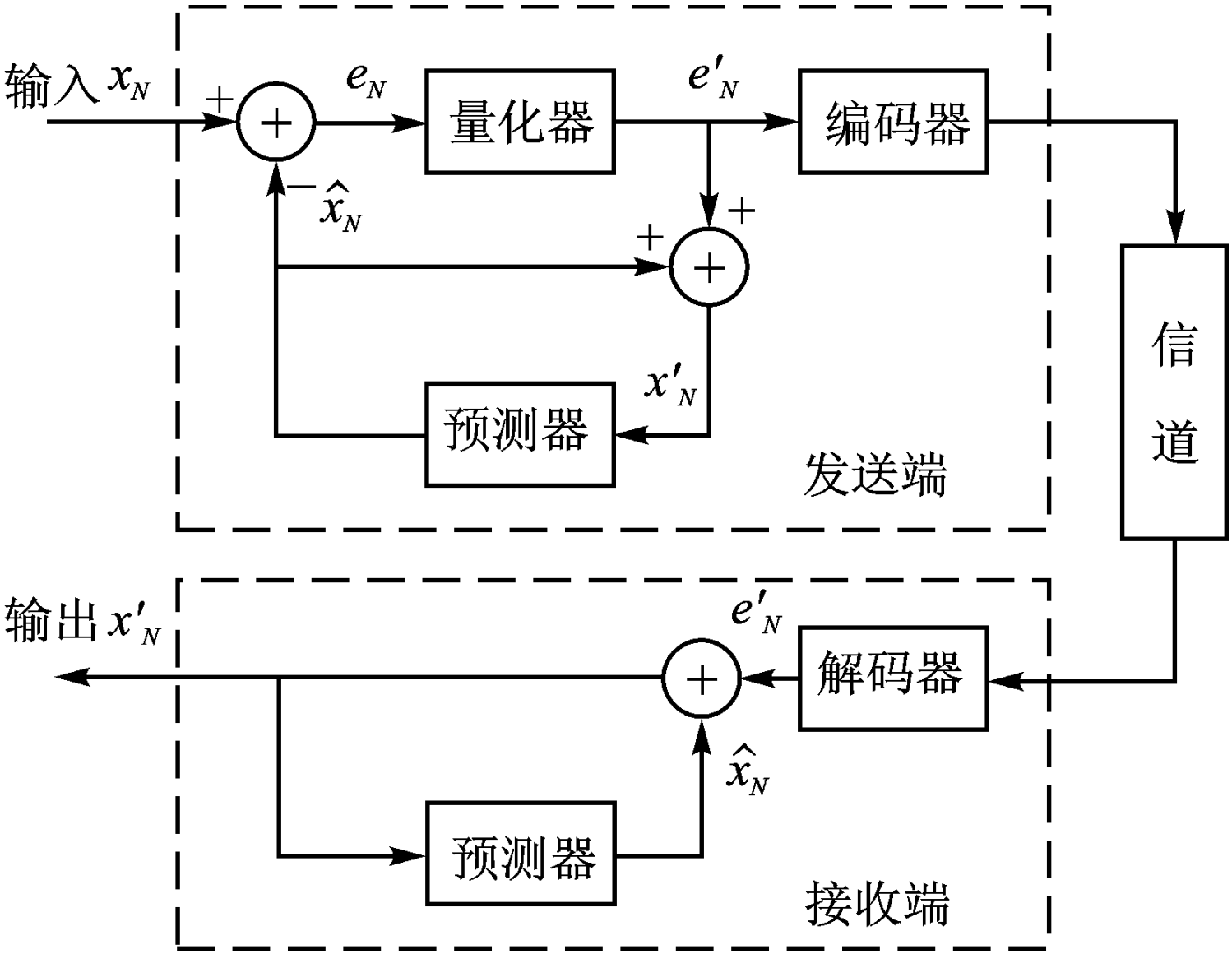


图3 DPCM系统原理框图

DPCM系统中的误差

- 设x_N为t_N时刻输入信号的亮度取样值，$\hat{x}_N$为根据时刻以前已知的像素亮度取样值$x_1, x_2, \ldots, x_{N-1}$对$x_N$所作的预测值；
  - e_N为差值信号，也称误差信号

$$e_N = x_N - \hat{x}_N$$

  - q_N为量化器的量化误差，e'_N为量化器输出信号

$$q_N = e_N - e'_N$$

  - 接收端输出为x'_N

$$x'_N = \hat{x}_N + e'_N$$

- 那么在接收端复原的像素值与发送端的原输入像素值之间的误差为

$$\begin{aligned} x_N - x'_N &= x_N - (\hat{x}_N + e'_N) \\ &= (x_N - \hat{x}_N) - e'_N \\ &= e_N - e'_N = q_N. \end{aligned}$$

DPCM系统中的误差

- 由此可见：在DPCM系统中的误差来源是发送端的量化器，而与接收端无关。
  - 若去掉量化器，那么$e_N = e'_N$，则$q_N = 0$，$x_N - x'_N = 0.$这样就可以完全不失真地恢复输入信号x_N，从而实现信息保持型编码；
  - 若$q_N \neq 0$，那么输入信号x_N和复原信号x'_N输出之间就一定存在误差，从而产生图像质量的某种降质，这样的DPCM系统实现的是保真度编码，在这样的DPCM系统中就存在一个如何能使误差尽可能减少的问题。

预测编码的类型

- 若t_N时刻之前的已知样值与预测值之间的关系呈现某种函数形式，该函数一般分为线性和非线性两种，所以预测编码器也就有线性预测器和非线性预测编码器两种。
- 若估计值$\hat{x}_N$与$x_1, x_2, \ldots, x_{N-1}$样值之间呈现为：

$$\hat{x}_N = \sum_{i=1}^{N-1} a_i x_i$$

其中$a_i\ (i = 1,2,\ldots,N-1)$为常量，则称之为线性预测，$a_1, a_2, \ldots, a_{N-1}$为预测系数。

- 若t_N时刻的信号样本值x_N与t_N时刻之前的已知样本值$x_1, x_2, \ldots, x_{N-1}$不是如上式的线性组合关系，而是非线性关系，则称之为非线性预测。

线性预测方案

- 在图像数据压缩中，常用如下几种线性预测方案：
  - 前值预测：即$\hat{x}_N = ax_{N-1}$
  - 一维预测：即用x_N的同一扫描行中的前面已知的几个采样值x_N预测，其预测公式为

$$\hat{x}_N = \sum_{i=1}^{N-1} a_i x_i$$

  - 二维预测：即不但用的同一扫描行以前的几个采样值(x_1, x_5)，如图4所示，还要用x_N的以前几行中的采样值(x_2, x_3, x_4)一起来预测。例如

$$\hat{x}_N = a_1 x_1 + a_2 x_2 + a_3 x_3 + a_4 x_4$$

- 以上都是一幅图像中像素点之间的预测，统称为帧内预测。

线性预测方案

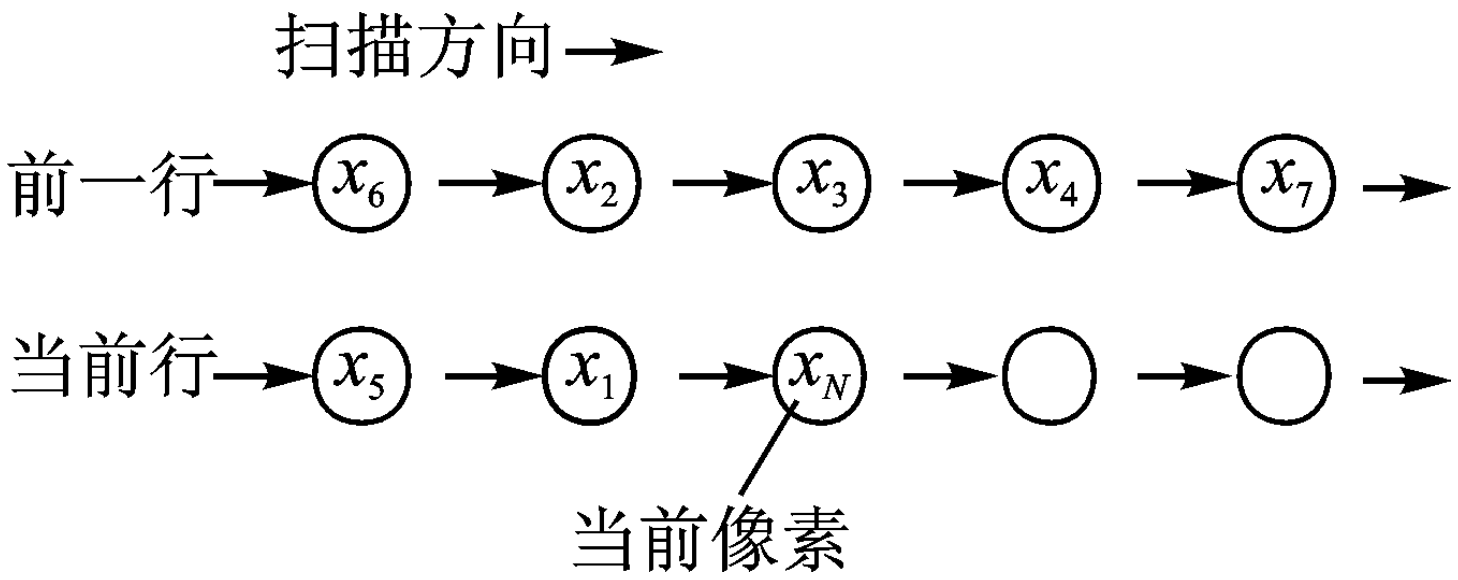


图4 二维预测示意图

线性预测方案

- 在图像数据压缩中，常用如下几种线性预测方案：
  - 三维预测（帧间预测）：即取用已知像素不但是前几行的而且还包括前几帧的来预测x_N。通常相邻帧间细节的变化是很少的，即相对应像素的灰度变化较小，存在极强的相关性，利用预测编码去除帧间的相关性，可以获得更大的压缩比。

- 最佳线性预测编码方法
  - 最佳线性预测就是按照均方误差最小准测，选择线性预测系数a_i，使得预测的偏差值$e_N = x_N - \hat{x}_N$为最小。

最佳线性预测编码方法

- 假定二维图像信号$x(t)$是一个均值为零，方差为σ^2的平稳随机过程，$x(t)$在$t_1, t_2, \ldots, t_{N-1}$时刻的抽样值集合为$x_1, x_2, \ldots, x_{N-1}$
- 由下式可以得到t_N时刻抽样值的线性预测值：

$$\hat{x}_N = \sum_{i=1}^{N-1} a_i x_i = a_1 x_1 + a_2 x_2 + \ldots + a_{N-1} x_{N-1}$$

式中，a_i为预测系数。

- 根据线性预测定义，$\hat{x}_N$必须十分逼近x_N，这就要求$a_1, a_2, \ldots, a_{N-1}$为最佳系数。采用均方误差最小化准则，可得到最佳的$a_i$。

均方误差最小化准则

设x_N的均方误差为：

$$\begin{aligned}\mathbb{E}\{\|e_N\|^2\} &= \mathbb{E}\{\|x_N - \hat{x}_N\|^2\} \\ &= \mathbb{E}\{\|x_N - (a_1x_1 + a_2x_2 + \ldots + a_{N-1}x_{N-1})\|^2\}\end{aligned}$$

为使$\mathbb{E}\{\|e_N\|^2\}$最小，在上式中对a_i求微分，即：

$$\begin{aligned}\frac{\partial}{\partial a_i}\mathbb{E}\{\|e_N\|^2\} &= \frac{\partial}{\partial a_i}\mathbb{E}\{\|x_N - (a_1x_1 + a_2x_2 + \ldots + a_{N-1}x_{N-1})\|^2\} \\ &= -2\mathbb{E}\{[x_N - (a_1x_1 + a_2x_2 + \ldots + a_{N-1}x_{N-1})]x_i\}\end{aligned}$$

其中，$i = 1, 2, \ldots, N-1.$

均方误差最小化准则

根据极值定义，得到$(N-1)$个方程组成的方程组：

$$\left.\begin{aligned}
&\mathbb{E}\{[x_N-(a_1x_1+a_2x_2+\ldots+a_{N-1}x_{N-1})]x_1\}=0\\
&\mathbb{E}\{[x_N-(a_1x_1+a_2x_2+\ldots+a_{N-1}x_{N-1})]x_2\}=0\\
&\cdots\cdots\\
&\mathbb{E}\{[x_N-(a_1x_1+a_2x_2+\ldots+a_{N-1}x_{N-1})]x_{N-1}\}=0
\end{aligned}\right\}$$

简记为：

$$\mathbb{E}\{[x_N-(a_1x_1+a_2x_2+\ldots+a_{N-1}x_{N-1})]x_i\}=0$$

其中，$i=1,2,\ldots,N-1.$

均方误差最小化准则

- 在实用中，对每幅图像都按公式计算a_i显得太麻烦，这时可参照前人已得的数据选择使用。
- 在静止图像的国际标准JPEG方案中，给出了静止图像的一个完整的二维预测器设计方案，它只考虑临近三点x_1, x_2, x_3，它们的位置关系如右图所示。第一行或第一列均采用同一行或同一列的前值预测，其它各点基本采用临近三点预测。对任意一点可采用下述预测公式之一：

x_1

x_2

x_3

$x_1 + ((x_3 - x_2)/2)$

$x_1 + x_3 - x_2$

$x_3 + ((x_1 - x_2)/2)$

$(x_1 + x_3)/2$

自适应预测编码方法

- 线性预测编码的基础是假设图像全域为平稳的随机过程，自相关系数与像素在域中的位置无关。
- 实际上，图像的起伏始终是存在的，被描述像素和周围像素之间，各有多种多样的关系。线性预测系数a_1是一种近似条件下的常数，忽略了像素的个性，存在以下缺点，影响图像质量，如图5所示。
  - 对灰度有突变的地方，会有较大的预测误差，致使重建图像的边缘模糊，分辨率降低。
  - 对灰度变化缓慢区域，其差值信号应为零，但因其预测值偏大而使重构图像有颗粒噪声。

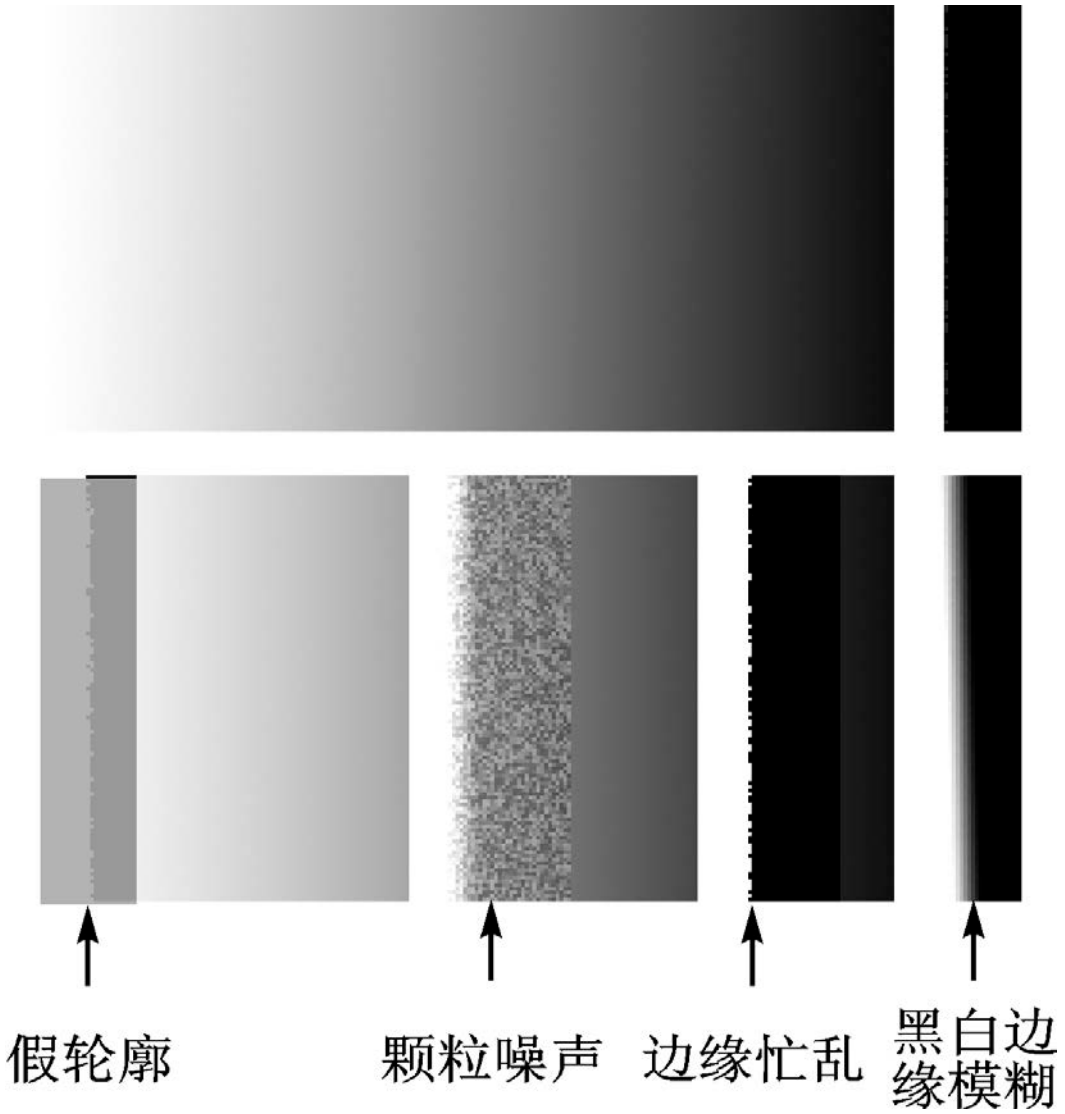


图5 预测编码降质图像

自适应预测编码方法

- 为了改善图像质量，克服上述预测编码带来的缺点，非线性预测充分考虑了图像的统计特性和个别变化，预测器的预测系数不固定，随图像的局部特性而有所变化。
- 尽量使预测系数与图像所处的局部特性相匹配，即预测系数随预测环境而变，从而得到较为理想的输出，故称为自适应预测编码。

自适应预测编码方法

- 自适应方法是很多的，1977年Yamada提出了一个二维DPCM的自适应预测方案：预测函数为

$$\hat{x}_N = K\left(a_1 x_1 + a_4 x_4\right)$$

  - 其预测系数$a_1 = 0.75$、$a_4 = 0.25$，K为自适应预测参数，定义如下：

$$K = \begin{cases} 1.0 + 0.125 & \left|e'_{N-1}\right| = e_K \\ 1.0 & e_1 < \left|e'_{N-1}\right| < e_K \\ 1.0 - 0.125 & \left|e'_{N-1}\right| = e_1 \end{cases}$$

其中e_k是最大量化输出正电平，e_1是最小量化输出的正电平；e'_{N-1}是第$N-1$个采样值的量化输出电平。

自适应预测编码方法

$$K=\begin{cases}1.0+0.125 & \left|e'_{N-1}\right|=e_K \\ 1.0 & e_1<\left|e'_{N-1}\right|<e_K \\ 1.0-0.125 & \left|e'_{N-1}\right|=e_1\end{cases}$$

- 当图像的第$N-1$个量化输出电平$|e'_{N-1}|$在e_1和e_k之间时，此时取自适应参数$K=1$，则第N个预测值将按$\hat{x}_N=0.75x_1+0.25x_4$输出。
- 当图像的第$N-1$个量化输出电平$|e'_{N-1}|=e_k$时，预测值$\hat{x}_N$自动增大12.5%，即自适应参数$K=1.125$。这对于缓减$\hat{x}_N$和$\hat{x}_{N+1}$等几个相邻像素出现斜率过载，增强图像边缘，减轻其模糊现象是有效的。
- 当$|e'_{N-1}|=e_1$时，取自适应系数$K=0.875$，预测值自动减小12.5%，在图像中减少颗粒噪声是有作用的。

变换编码

- 变换编码的基本概念就是将原来在空间域上描述的图像等信号，通过一种数学变换（常用二维正交变换如傅立叶变换、离散余弦变换、沃尔什变换等），变换到变换域中进行描述，达到改变能量分布的目的。
- 变换编码将图像能量在空间域的分散分布变为在变换域的能量的相对集中分布，达到去除相关的目的，再经过适当的方式量化编码，进一步压缩图像。

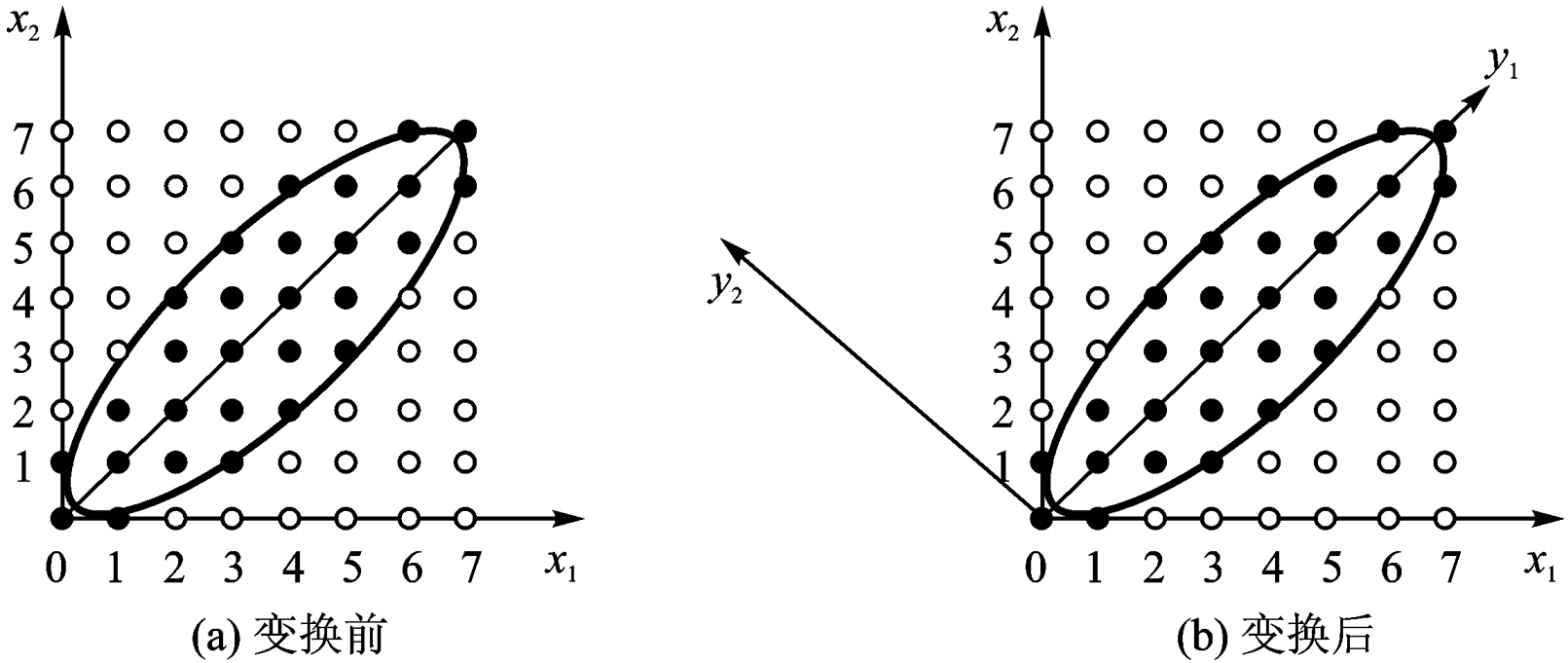
变换编码的数理解释

先从一个最简单的实例来看，一个域的数据变换到另一个域中以后其分布是如何改变的。

- 假设把一个$n \times n$像素的子图像看成为n^2维坐标系中的一个坐标点，也就是说，在这个坐标系中每一个坐标点对应于n^2个像素。这个坐标点各维的数值是其对应的n^2个像素的灰度组合。
- 以1×2像素构成的子图像（即相邻两个像素组成的子图像）为例，设每个像素取8个灰度级（3比特量化）。以图(a)中的x_1轴表示第一个像素可能取的8个量化层，x_2轴表示第二个像素可能取的8个量化层。由x_1, x_2组成的二维坐标系中每一个坐标点对应于一个1×2子图像，该点数值由两个像素的灰度所组成。
- 因此总共有8×8=64种可能的灰度组合，由图(a)中64个坐标点表示。

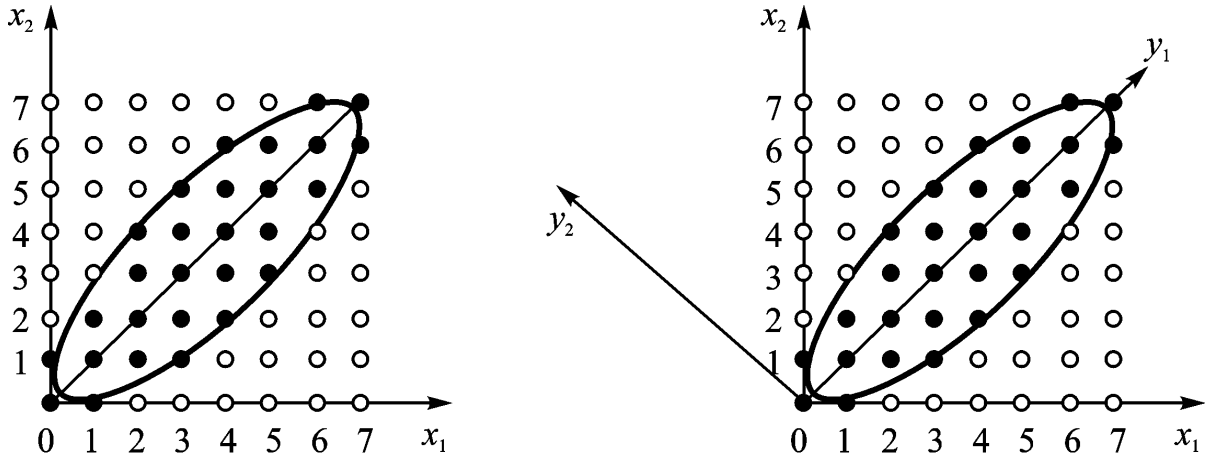
变换编码的数理解释

对一般图像而言，因相邻像素之间存在很强的相关性，绝大多数的子图像中相邻两像素其灰度级相等或很接近，也就是说在直线$x_1 = x_2$附近出现的概率大。



(a) 变换前　　(b) 变换后

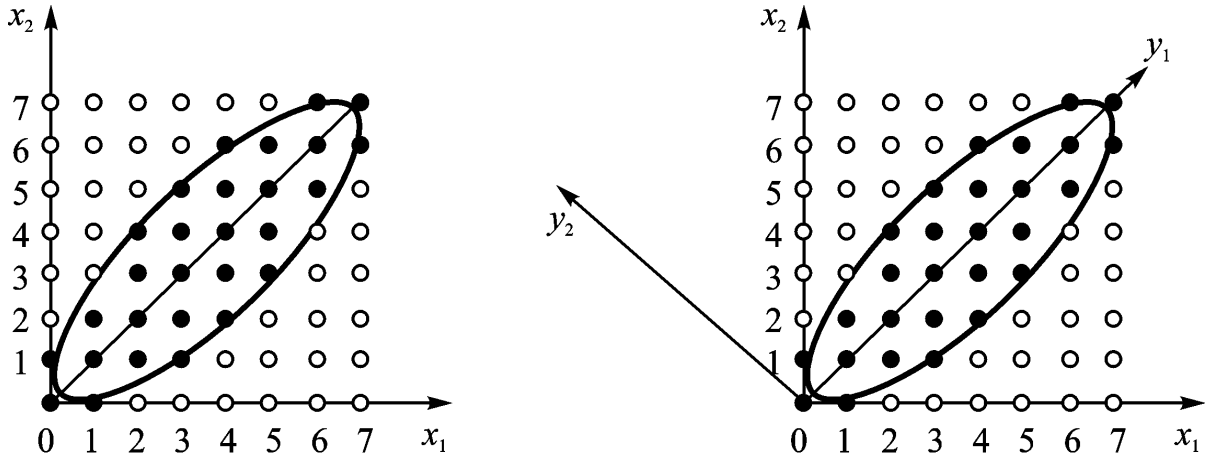
正交变换的物理概念



(a) 变换前　　(b) 变换后

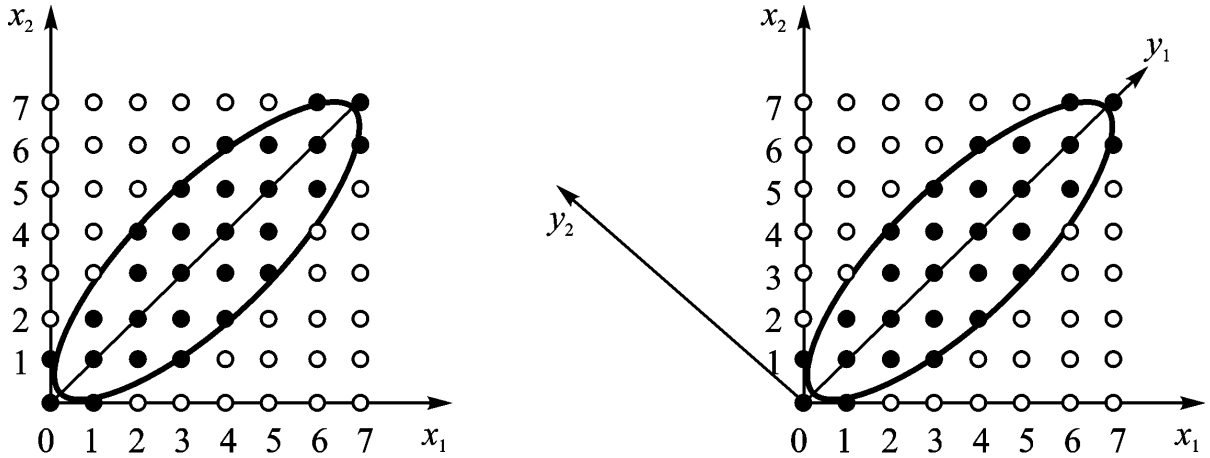
- 现在将坐标系逆时针旋转45度，如图(b)所示。在新的坐标系y_1, y_2中，概率大的子图像区位于y_1轴附近。由此表明变量y_1, y_2之间的联系比变量x_1, x_2之间的联系在统计上更加独立，而且方差也重新分布。

- 在原来坐标系中子图像的两个像素具有较大的相关性，能量的分布比较分散，两者具有大致相同的方差$\sigma_{x_1}^2 \approx \sigma_{x_2}^2$；而在变换后的坐标系中，子图像的两个像素之间的相关性大大减弱，能量的发布向y_1轴集中，且y_1的方差也远大于y_2，即$\sigma_{y_1}^2 \gg \sigma_{y_2}^2$.



(a) 变换前　　(b) 变换后

- 从这个简单的例子可以看出，这种变换后坐标轴上方差不均匀分布正是正交变换编码能够实现图像数据压缩的理论根据。
- 若按照人的视觉特性，只保留方差较大的那些变换系数分量，就可以获得更大的数据压缩比，这就是所谓视觉心理编码的方法之一。



(a) 变换前　　(b) 变换后

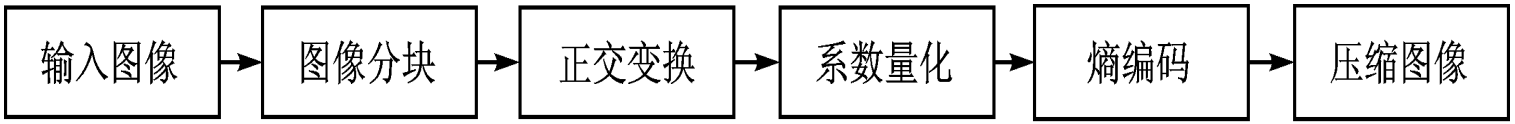
- 将上述简单的实例推广到一般的$n \times n$的图像的变换，图像在n^2维变换域中，相关性大大下降。因此用变换后的系数进行编码，将比直接使用原图像数据编码获得更大的数据压缩。

正交变换编码的基本原理

- 信息论的研究表明：正交变换不改变信源的熵值，变换前后图像的信息量并无损失，完全可以通过反变换得到原来的图像值。
- 但是，统计分析表明，图像经过正交变换后，把原来分散在原空间的图像数据在新的坐标空间中得到集中，对于大多数图像，大量的变换系数很小，只要删除接近于0的系数，并且对较小的系数进行粗量化，而保留包含图像主要信息的系数，以此进行压缩编码。

正交变换编码的基本原理

- 在重建图像进行解码（逆变换）时，所损失的将是一些不重要的信息，几乎不会引起图像的失真，图像的变换编码就是利用这些来压缩图像的，这种方法可得到很高的压缩比。
- 一个典型的变换编码系统如下图所示，编码器执行四个步骤：图像分块、变换、量化和编码。



(a) 编码器方框图

(b) 解码器方框图

正交变换编码的基本原理

- 一个典型的变换编码系统如下图所示，编码器执行四个步骤：图像分块、变换、量化和编码。
  - 首先分块，将一幅$N \times N$的图像分成$(N/n)^2$个子图像；
  - 然后对子图像进行变换，解除子图像像素间的相关性，达到用少量的变换系数包含尽可能多的图像信息目的；
  - 接下来的量化步骤是有选择的消除或粗量化带有很少信息的变换系数，因为它们对重建图像的质量影响很小；
  - 最后是编码，一般用变长码对量化后的系数进行编码。解码是编码的逆操作，由于量化是不可逆的，所以在解码中没有对应的模块，要注意的是压缩并不是在变换步骤中取得的，而是在量化变换系数和编码时取得的。

变换的选择

- 许多图像变换都可以用于变换编码。
  - 在理论上，K–L变换是最优的正交变换，它能完全消除子图像块内像素间的线性相关性，经K–L变换后各变换系数在统计上不相关，其协方差矩阵为对角阵，因而大大减少了原数据的冗余度，如果丢弃特征值较小的一些变换系数，那么所造成的均方误差是所有正交变换中最小的。
  - 由于K–L变换是取原图各子图像块协方差矩阵的特征向量作为变换后的基向量，因此K–L变换的基对不同图像是不同的，与编码对象的统计特性有关，这种不确定性使得K-L变换使用起来非常不方便，所以尽管K–L变换具有上述优点，一般只将它作为理论上的比较标准。

变换的选择

- 许多图像变换都可以用于变换编码。
  - 在目前常用的正交变换中，DCT变换其性能接近最佳，仅次于K-L变换，所以DCT变换被认为是一种准最佳变换。
  - 另一方面，DCT变换矩阵与图像内容无关，而且由于它是构造成对称的数据序列，从而避免了子图像边界处的跳跃和不连续现象，并且也有快速算法（FDCT）。
  - 所以在图像编码的应用中，往往都采用二维DCT。在JPEG基本系统中，就是采用二维DCT的算法作为压缩的基本方法。

变换的选择

- 许多图像变换都可以用于变换编码。
  - 傅立叶变换是应用最早的变换之一，也有快速算法，但它的不足之处在于子图像的变换系数在边界处的不连续而造成恢复的子图像在其边界也不连续，于是由各恢复子图像构成的整幅图像将呈现隐约可见的以子图像的方块状结构，影响图像质量。
  - 沃尔什变换与DCT变换相比，其算法简单（只有加法和减法），因而运算速度快，适用于高速实时系统，而且也容易硬件实现，但性能比DCT变换要差—些。

子图像尺寸的选择

- 正交变换是一种数据处理手段，它将被处理的数据按照某种变换规则映射到另一个域中去处理。
- 正交变换有一维、二维和多维不同的处理方式，由于图像可以看成二维数据矩阵，所以在图像编码中多采用二维正交变换的方式。
- 如果将一幅图像作为一个二维矩阵，则其正交变换的计算量也太大，难以实现，所以在实用中变换编码并不是对整幅图像进行变换和编码，而是将图像分成若干个$n \times n$的子图像后分别处理，原因如下：
  - 小块图像的变换计算容易；
  - 距离较远的像素之间的相关性比距离较近的像素之间的相关性小。

子图像尺寸的选择

- 实践证明，子图像取4×4、8×8、16×16适合图像的压缩，这是因为：
  - 如果子图像尺寸取得太小，虽然计算速度快，实现简单，但压缩能力有限。
  - 如果子图像尺寸取得太大，虽然去相关效果好，因为DFT、DCT等正弦类变换均渐近最佳性，但也渐趋饱和；若尺寸太大，由于图像本身的相关性很小，反而使其压缩效果不明显，而且增加了计算的复杂性。

变换系数的选择

- 对子图像经过变换后，变换后的系数保留哪些系数用作编码和传输将直接影响信号恢复的质量，变换系数的选择原则是保留能量集中的、方差大的系数。
- 系数选择通常有变换区域编码和变换阈值编码两种方法。

变换系数的选择

- 变换区域编码
  - 变换区域编码就是对设定形状的区域内的变换系数进行量化编码，区域外的系数就被舍去。一般来说，变换后的系数值较大的都会集中在区域的左上部，即低频率分量都集中在此部分，保留的也是这一部分。其他部分的系数被舍去，在恢复信号时再对它们补以零。这样，由于保留了大部分图像信号能量，在恢复信号后，其质量不会产生显著变化。
  - 变换区域编码的明显缺陷，就是高频分量完全丢失。反映在恢复图像上将是轮廓及细节模糊。克服这一缺陷的方法，可以预先设定几个区域，根据实际系数分布自动选取能量最大的区域。

变换系数的选择

- 变换阈值编码
  - 变换阈值编码就是根据实际情况设定某一大小幅度的阈值，若变换系数超过该阈值，则保留这些系数进行编码传输，其余的补以零。这样，多数低频成分被编码输出，而且少数超过阈值的高频成分也将被保留下来进行编码输出，这在一定程度上弥补了区域法的不足。
  - 但这种选择系数的方法有两个问题需要解决：
    - 一个是被保留下来进行编码的系数在矩阵中的位置是不确定的，因此尚需增加“地址”编码比特数，其码率相对地要高一些；
    - 另一个问题是“阈值”需要通过实验来确定，当然也可以根据总比特数，进行自适应阈值选择，但需要一定的技术，将增加编码的复杂程度。

图像压缩编码国际标准

- 静止图像压缩标准JPEG
  - JPEG（Joint Photographic Experts Group）是联合图像专家小组的缩写，所谓联合是指国际标准化组织（ISO）和国际电报电话咨询委员会（CCITT）的联合，联合图像专家小组1986年成立，任务是开发研制出连续色调、多级灰度、静止图像的数字图像压缩编码标准，使之满足以下要求：
    - 必须将图像质量控制在可视保真度高的范围内，同时编码器可被参数化，允许用户设置压缩或质量水平；
    - 压缩标准可以应用于任何一类连续色调数字图像，并不应受到维数、颜色、画面尺寸、内容、影调的限制；
    - 压缩标准必须从完全无损到有损范围内可选，以适应不同的存储、CPU和显示要求。

图像压缩编码国际标准

- 静止图像压缩标准JPEG
  - 此外，JPEG标准是为连续色调图像的压缩提供的公共标准，连续色调图像并不局限于单色调图像，该标准可适用于各种多媒体存储和通信应用所使用的灰度图像、摄影图像及静止视频压缩文件。
  - JPEG标准包括图像编码和解码过程以及压缩图像数据的编码表示，它提供了三种压缩算法：基本系统（Baseline System）、扩展系统（Extended System）和无失真压缩（Lossless），所有的JPEG编码器和解码器必须支持基本系统，另外两种压缩算法适用于特定的应用。

图像压缩编码国际标准

- JPEG支持两种图像建立模式：
  - 顺序型（Sequential）和渐进型（Progressive），以满足用户对应用的不同需求。
- JPEG压缩算法分为两大类：
  - 无失真压缩算法：将源图像数据转变为压缩数据，该压缩数据经对应的解压缩算法处理后可获得与源图像完全一致的重建图像。
  - 有失真压缩算法：基于离散余弦变换，所生成的压缩图像数据经解压缩生成的重建图像与源图像在视觉上保持一致。：

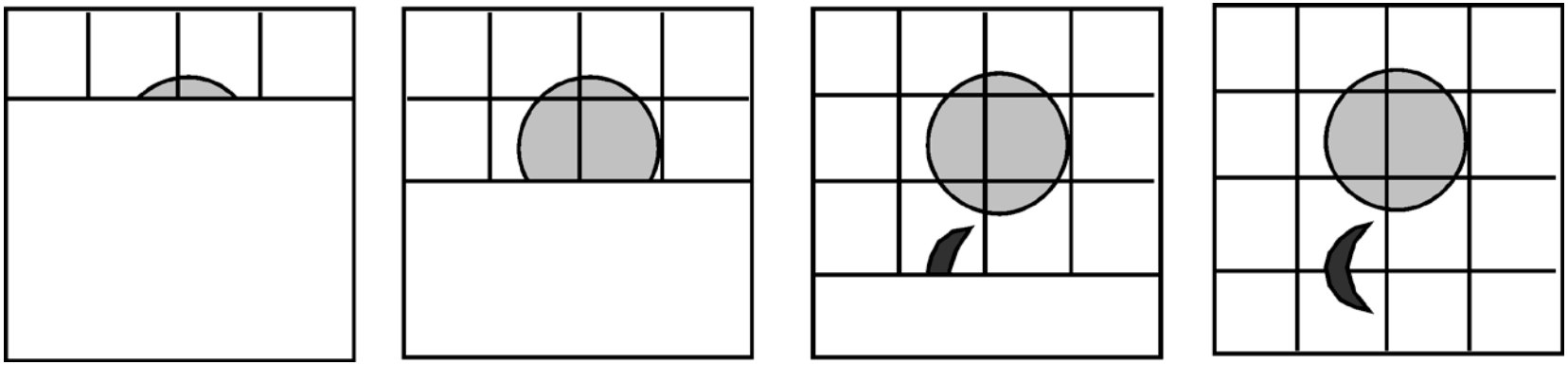
JPEG工作模式

- 一般来说，压缩比越大，视觉上的一致性越差。综合以上两点，JPEG总共有四种工作模式：
  - 顺序型编码工作模式
  - 渐进式DCT模式
  - 无失真编码工作模式
  - 分层方式

JPEG工作模式

- 顺序型编码工作模式

  图像的所有8×8像素的图像子块从左到右、从上到下依次输入。图像子块经DCT变换后形成8×8的DCT系数阵列，每一个系数阵列被量化后立即进行熵编码并作为压缩图像数据的一部分输出，从而尽可能地降低对系数存储要求。



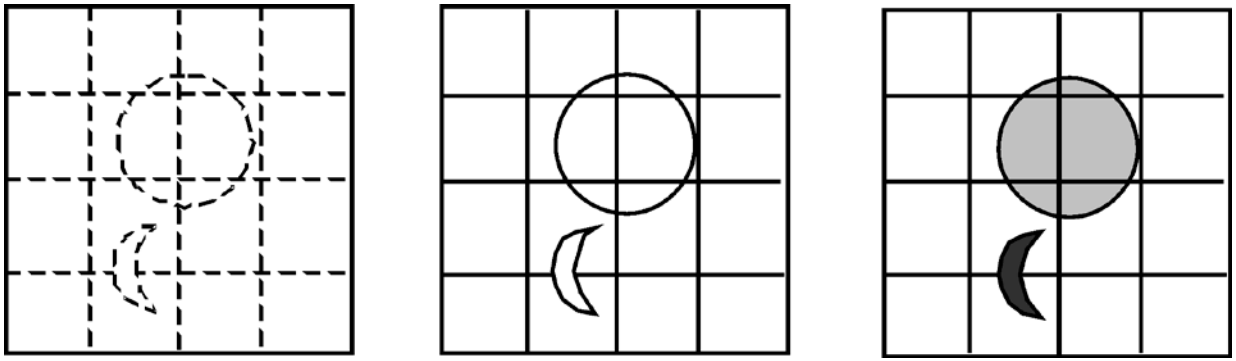
顺序型编码工作模式

JPEG工作模式

- 渐进式DCT模式

  基于DCT，对图像分层次进行处理，从模糊到清晰地传输图像。有两种实现方法：

  - 一种是频谱选择法，即按Z形扫描的序号将DCT量化序数分成几个频段，每个频段对应一次扫描，每块均先传送低频扫描数据，得到原图概貌，再依次传送高频扫描数据，使图像逐渐清晰；
  - 另一种是逐次逼近法，即每次扫描全部DCT量化序数，但每次的表示精度逐渐提高。



渐进型编码工作模式

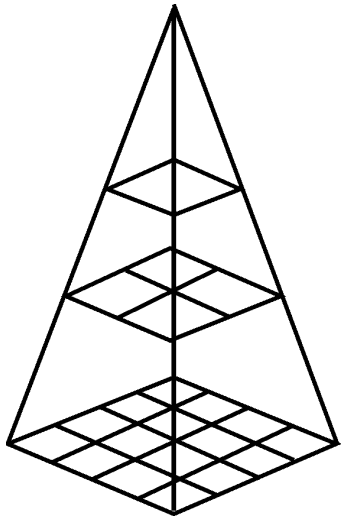
JPEG工作模式

- 无失真编码工作模式

  被编码的图像可以保证恢复到与源图像数据完全一致。

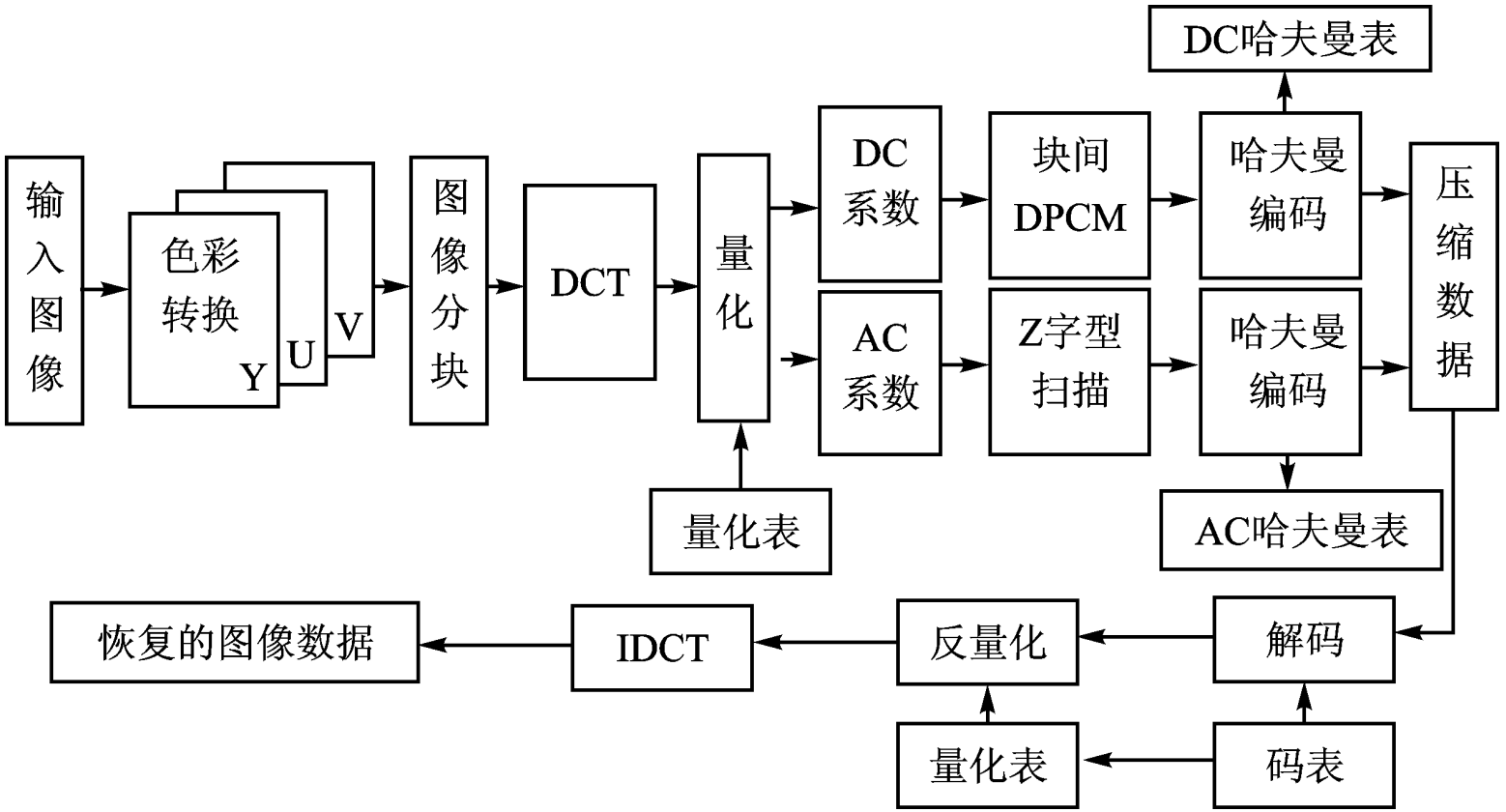
- 分层方式

  在空间域将源图像以不同的分辨率表示，每个分辨率对应一次扫描，处理时可以基于DCT或预测编码，可以是渐进式，也可以是顺序式，如图所示。



分层编码工作模式

JPEG基本系统的编解码



JPEG基本系统的编解码方框图

颜色空间转换

- 在彩色图像中，JPEG分别压缩图像的每个彩色分量。虽然JPEG可以压缩通常的红绿蓝分量，但在YCbCr空间的压缩效果会更好，这是因为人眼对色彩的变化不如对亮度的变化敏感，因而对色彩的编码可以比对亮度的编码粗糙些，这主要体现在不同的采样频率和量化精度上。
- 因此，编码前一般先将图像从RGB空间转换到YCbCr空间。

数据分块

- 在颜色空间转换完成之后，再将每个分量图像分割成不重叠的8×8像素块，每一个8×8像素块称为一个数据单元（DU）。

图像采样

- 在对图像采样时，可以采用不同的采样频率，这种技术称为二次采样。
  - 由于亮度比色彩更重要，因而对Y分量的采样频率可高于对Cb、Cr的采样频率，这样有利于节省存储空间。常用的采样方案有YUV422和YUV411。
  - 把采样频率最低的分量图像中一个DU所对应的像区上覆盖的所有各分量上的DU按顺序编组为一个最小编码单元（MCU）。
  - 对灰度图像而言，只有一个Y分量，MCU就是一个数据单元。而对彩色图像而言，以4:1:1的采样方案为例，则一个MCU由4个Y分量的DU、1个Cb分量的DU和1个Cr分量的DU组成。

离散余弦变换（DCT）

- 图像数据块分割后，即以MCU为单位顺序将DU进行二维离散余弦变换。对以无符号数表示的具有P位精度的输入数据，在DCT前要减去2^{P-1}，转换成有符号数，而在IDCT后，应加上2^{P-1}，转换成无符号数。对每个8×8的数据块DU进行DCT后， 得到的64个系数代表了该图像块的频率成分，其中低频分量集中在左上角， 高频分量分布在右下角。系数矩阵左上角的叫做直流（DC）系数，它代表了该数据块的平均值，其余63个叫交流（AC）系数。

系数量化

- 在DCT处理中得到的64个系数中，低频分量包含了图像亮度等主要信息。在从空间域到频域的变换中，图像中的缓慢变化比快速变化更易引起人眼的注意，所以在重建图像时，低频分量的重要性高于高频分量。
- 因而在编码时可以忽略高频分量， 从而达到压缩的目的，这也是量化的根据和目的。

系数量化

- 在JPEG标准中，用具有64个独立元素的量化表来规定DCT域中相应的64个系数的量化精度，使得对某个系数的具体量化阶取决于人眼对该频率分量的视觉敏感程度。
- 理论上，对不同的空间分辨率、数据精度等情况，应该有不同的量化表。不过，一般采用下图所示的量化表，可取得较好的视觉效果。
- 之所以用两张量化表，是因为Y分量比C_b和C_r更重要些，因而对Y采用细量化，而对C_b和C_r采用粗量化。量化就是用DCT变换后的系数除以量化表中相对应的量化阶后四舍五入取整。由于量化表中左上角的值较小，而右下角的值较大， 因而起到了保持低频分量、抑制高频分量的作用。

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

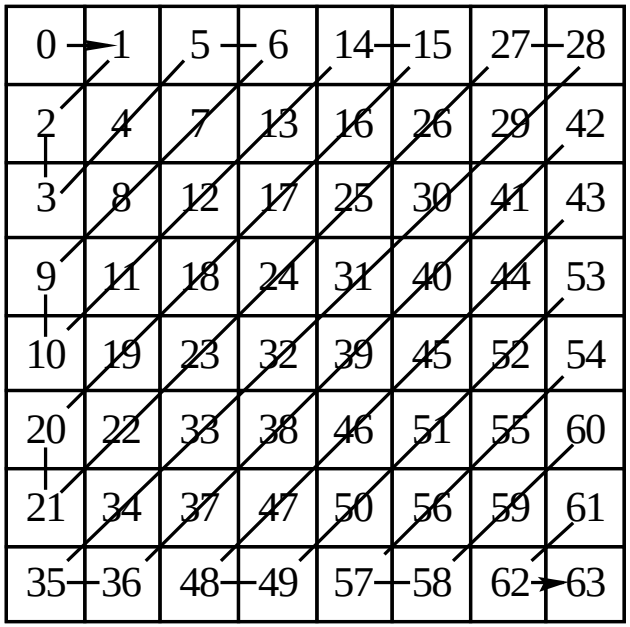
亮度量化表

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

色度量化表

Z形扫描

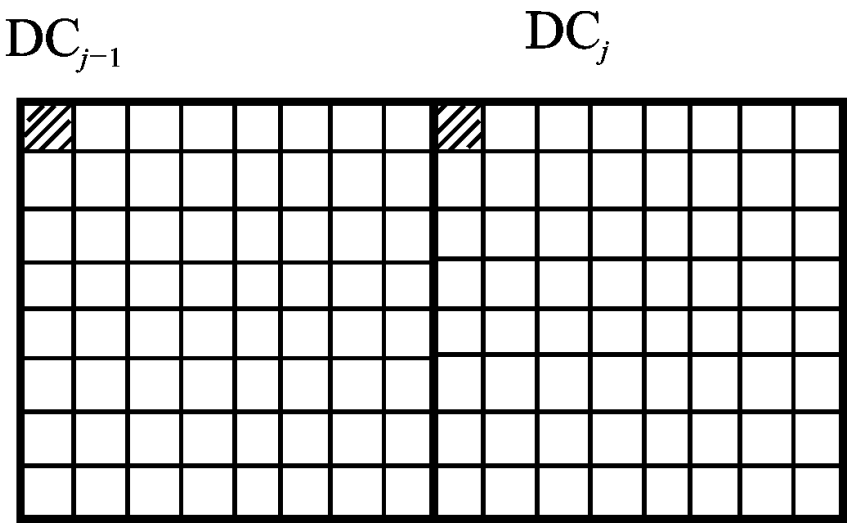
- DCT系数量化后，构成一个稀疏矩阵，用Z（Zigzag）形扫描将其变成一维数列，将有利于熵编码。Z形扫描的顺序如图所示。



DCT系数的Z形扫描顺序

DC系数编码

- DC系数反映了一个8×8数据块的平均亮度，一般与相邻块有较大的相关性。JPEG对DC系数作差分编码，即用前一数据块的同一分量的DC_{j-1}系数作为当前块的预测值，再对当前块的实际值DC_j与预测值DC_{j-1}的差值作哈夫曼编码。



DC系数差分编码

DC系数编码

- 若DC系数的动态范围为-1024～+1024，则差值的动态范围为-2047～+2047。如果为每个差值赋予一个码字，则码表过于庞大。
- 因此，JPEG对码表进行了简化，采用“前缀码（SSSS）+尾码”来表示。前缀码指明了尾码的有效位数B，可以根据DIFF从表5.5.1中查出前缀码对应的哈夫曼编码。尾码的取值取决于DC系数的差值和前缀码，如果DC系数的差值DIFF大于等于0，则尾码的码字为DIFF的B位原码；否则，取DIFF的B位反码。

图像分量为8位时DC系数差值的典型哈夫曼编码表

SSSS	DC 系数差值 DIFF	亮度码字	色度码字
0	0	00	00
1	－1，1	010	01
2	－3，－2，2，3	011	10
3	－7～－4，4～7	100	110
4	－15～－8，8～15	101	1110
5	－31～－16，16～31	110	11110
6	－63～－17，17～63	1110	111110
7	－127～－64，64～127	11110	1111110
8	－255～－128，128～255	111110	11111110
9	－511～－256，256～511	1111110	111111110
10	－1023～－512，512～1023	11111110	1111111110
11	－2047～－1023，1023～2047	111111110	11111111110

AC系数编码

- 经Z形排列后的AC系数，更有可能出现连续0组成的字符串，从而对其进行行程编码将有利于压缩数据。
  - JPEG将一个非零AC系数及其前面的0行程长度（连续0的个数）的组合称为一个事件。将每个事件编码表示为“NNNN/SSSS+尾码”，其中，NNNN为0行程的长度，SSSS表示尾码的有效位数B（即当前非0系数所占的比特数），如果非零AC系数大于等于0，则尾码的码字为该系数的B位原码，否则，取该系数的B位反码。
  - 由于只用4位表示0行程的长度，故在JPEG编码中，最大0行程只能等于15。当0行程长度大于16时，需要将其分开多次编码，即对前面的每16个0以“F/0”表示，对剩余的继续编码。
  - 由非零系数的数值可从表2中查出对应的SSSS，由NNNN / SSSS又可从表3或表4中查得其对应的哈夫曼表。

表2 AC系数的尾码位数表

SSSS	AC 系数的幅度
0	0
1	－1，1
2	－3，－2，2，3
3	－7～－4，4～7
4	－15～－8，8～15
5	－31～－16，16～31
6	－63～－17，17～63
7	－127～－64，64～127
8	－255～－128，128～255
9	－511～－256，256～511
10	－1023～－512，512～1023

表3 亮度AC系数码表

行程/尺寸	码长	码 字	行程/尺寸	码长	码 字	行程/尺寸	码长	码 字
0/0 (EOB)	4	1010	5/4	16	1111111110011111	A/8	16	1111111111001101
0/1	2	00	5/5	16	1111111110100000	A/9	16	1111111111001110
0/2	2	01	5/6	16	1111111110100001	A/A	16	1111111111001111
0/3	3	100	5/7	16	1111111110100010	B/1	10	1111111001
0/4	4	1011	5/8	16	1111111110100011	B/2	16	1111111111010000
0/5	5	11010	5/9	16	1111111110100100	B/3	16	1111111111010001
0/6	7	1111000	5/A	16	1111111110100101	B/4	16	1111111111010010
0/7	8	11111000	6/1	7	1111011	B/5	16	1111111111010011
0/8	10	1111110110	6/2	12	111111110110	B/6	16	1111111111010100
0/9	16	1111111110000010	6/3	16	1111111110100110	B/7	16	1111111111010101
0/A	16	1111111110000011	6/4	16	1111111110100111	B/8	16	1111111111010110
1/1	4	1100	6/5	16	1111111110101000	B/9	16	1111111111010111
1/2	5	11011	6/6	16	1111111110101001	B/A	16	1111111111011000
1/3	7	1111001	6/7	16	1111111110101010	C/1	10	1111111010
1/4	9	111110110	6/8	16	1111111110101011	C/2	16	1111111111011001
1/5	11	11111110110	6/9	16	1111111110101100	C/3	16	1111111111011010
1/6	16	1111111110000100	6/A	16	1111111110101101	C/4	16	1111111111011011
1/7	16	1111111110000101	7/1	8	11111010	C/5	16	1111111111011100

续表

行程/尺寸	码长	码　字	行程/尺寸	码长	码　字	行程/尺寸	码长	码　字
1/8	16	1111111110000110	7/2	12	111111110111	C/6	16	1111111111011101
1/9	16	1111111110000111	7/3	16	1111111110101110	C/7	16	1111111111011110
1/A	16	1111111110001000	7/4	16	1111111110101111	C/8	16	1111111111011111
2/1	5	11100	7/5	16	1111111110110000	C/9	16	1111111111100000
2/2	8	11111001	7/6	16	1111111110110001	C/A	16	1111111111100001
2/3	10	1111110111	7/7	16	1111111110110010	D/1	11	11111111000
2/4	12	111111110100	7/8	16	1111111110110011	D/2	16	1111111111100010
2/5	16	1111111110001001	7/9	16	1111111110110100	D/3	16	1111111111100011
2/6	16	1111111110001010	7/A	16	1111111110110101	D/4	16	1111111111100100
2/7	16	1111111110001011	8/1	9	111111000	D/5	16	1111111111100101
2/8	16	1111111110001100	8/2	15	111111111000000	D/6	16	1111111111100110
2/9	16	1111111110001101	8/3	16	1111111110110110	D/7	16	1111111111100111
2/A	16	1111111110001110	8/4	16	1111111110110111	D/8	16	1111111111101000
3/1	6	111010	8/5	16	1111111110111000	D/9	16	1111111111101001
3/2	9	111110111	8/6	16	1111111110111001	D/A	16	1111111111101010
3/3	12	111111110101	8/7	16	1111111110111010	E/1	16	1111111111101011
3/4	16	1111111110001111	8/8	16	1111111110111011	E/2	16	1111111111101100

续表

3/5	16	1111111110010000	8/9	16	1111111110111100	E/3	16	1111111111101101
3/6	16	1111111110010001	8/A	16	1111111110111101	E/4	16	1111111111101110
3/7	16	1111111110010010	9/1	9	111111001	E/5	16	1111111111101111
3/8	16	1111111110010011	9/2	16	1111111110111110	E/6	16	1111111111110000
3/9	16	1111111110010100	9/3	16	1111111110111111	E/7	16	1111111111110001
3/A	16	1111111110010101	9/4	16	1111111111000000	E/8	16	1111111111110010
4/1	6	111011	9/5	16	1111111111000001	E/9	16	1111111111110011
4/2	10	1111111000	9/6	16	1111111111000010	E/A	16	1111111111110100
4/3	16	1111111110010110	9/7	16	1111111111000011	F/0	11	11111111001
4/4	16	1111111110010111	9/8	16	1111111111000100	F/1	16	1111111111110101
4/5	16	1111111110011000	9/9	16	1111111111000101	F/2	16	1111111111110110
4/6	16	1111111110011001	9/A	16	1111111111000110	F/3	16	1111111111110111
4/7	16	1111111110011010	A/1	9	111111010	F/4	16	1111111111111000
4/8	16	1111111110011011	A/2	16	1111111111000111	F/5	16	1111111111111001
4/9	16	1111111110011100	A/3	16	1111111111001000	F/6	16	1111111111111010
4/A	16	1111111110011101	A/4	16	1111111111001001	F/7	16	1111111111111011
5/1	7	1111010	A/5	16	1111111111001010	F/8	16	1111111111111100
5/2	11	11111110111	A/6	16	1111111111001011	F/9	16	1111111111111101
5/3	16	1111111110011110	A/7	16	1111111111001100	F/A	16	1111111111111111

表4 色差AC系数编码

行程/尺寸	码长	码字	行程/尺寸	码长	码字	行程/尺寸	码长	码字
0/0 (EOB)	2	00	5/4	16	1111111110100000	A/8	16	1111111111001111
0/1	2	01	5/5	16	1111111110100001	A/9	16	1111111111010000
0/2	3	100	5/6	16	1111111110100010	A/A	16	1111111111010001
0/3	4	1010	5/7	16	1111111110100011	B/1	9	111111001
0/4	5	11000	5/8	16	1111111110100100	B/2	16	1111111111010010
0/5	5	11001	5/9	16	1111111110100101	B/3	16	1111111111010011
0/6	6	111000	5/A	16	1111111110100110	B/4	16	1111111111010100
0/7	7	1111000	6/1	7	1111001	B/5	16	1111111111010101
0/8	9	111110100	6/2	11	11111110111	B/6	16	1111111111010110
0/9	10	1111110110	6/3	16	1111111110100111	B/7	16	1111111111010111
0/A	12	111111110100	6/4	16	1111111110101000	B/8	16	1111111111011000
1/1	4	1011	6/5	16	1111111110101001	B/9	16	1111111111011001
1/2	6	111001	6/6	16	1111111110101010	B/A	16	1111111111011010
1/3	8	11110110	6/7	16	1111111110101011	C/1	9	111111010
1/4	9	111110101	6/8	16	1111111110101100	C/2	16	1111111111011011
1/5	11	11111110110	6/9	16	1111111110101101	C/3	16	1111111111011100

续表

1/6	12	111111110101	6/A	16	1111111110101110	C/4	16	1111111111011101
1/7	16	1111111110001000	7/1	7	1111010	C/5	16	1111111111011110
1/8	16	1111111110001001	7/2	11	11111111000	C/6	16	1111111111011111
1/9	16	1111111110001010	7/3	16	1111111110101111	C/7	16	1111111111100000
1/A	16	1111111110001011	7/4	16	1111111110110000	C/8	16	1111111111100001
2/1	5	11010	7/5	16	1111111110110001	C/9	16	1111111111100010
2/2	8	11110111	7/6	16	1111111110110010	C/A	16	1111111111100011
2/3	10	1111110111	7/7	16	1111111110110011	D/1	11	11111111001
2/4	12	111111110110	7/8	16	1111111110110100	D/2	16	1111111111100100
2/5	15	111111111000010	7/9	16	1111111110110101	D/3	16	1111111111100101
2/6	16	1111111110001100	7/A	16	1111111110110110	D/4	16	1111111111100110
2/7	16	1111111110001101	8/1	8	11111001	D/5	16	1111111111100111
2/8	16	1111111110001110	8/2	16	1111111110110111	D/6	16	1111111111101000
2/9	16	1111111110001111	8/3	16	1111111110111000	D/7	16	1111111111101001
2/A	16	1111111110010000	8/4	16	1111111110111001	D/8	16	1111111111101010
3/1	5	11010	8/5	16	1111111110111010	D/9	16	1111111111101011
3/2	8	11110111	8/6	16	1111111110111011	D/A	16	1111111111101100
3/3	10	1111110111	8/7	16	1111111110111100	E/1	14	11111111100000
3/4	12	11111110110	8/8	16	1111111110111101	E/2	16	1111111111101101

续表

行程/尺寸	码长	码　字	行程/尺寸	码长	码　字	行程/尺寸	码长	码　字
3/5	16	1111111110010001	8/9	16	1111111110111110	E/3	16	1111111111101110
3/6	16	1111111110010010	8/A	16	1111111110111111	E/4	16	1111111111101111
3/7	16	1111111110010011	9/1	9	111110111	E/5	16	1111111111110000
3/8	16	1111111110010100	9/2	16	1111111111000000	E/6	16	1111111111110001
3/9	16	1111111110010101	9/3	16	1111111111000001	E/7	16	1111111111110010
3/A	16	1111111110010110	9/4	16	1111111111000010	E/8	16	1111111111110011
4/1	6	111010	9/5	16	1111111111000011	E/9	16	1111111111110100
4/2	9	111110110	9/6	16	1111111111000100	E/A	16	1111111111110101
4/3	16	1111111110010111	9/7	16	1111111111000101	F/0	10	1111111010
4/4	16	1111111110011000	9/8	16	1111111111000110	F/1	15	111111111000011
4/5	16	1111111110011001	9/9	16	1111111111000111	F/2	16	1111111111110110
4/6	16	1111111110011010	9/A	16	1111111111001000	F/3	16	1111111111110111
4/7	16	1111111110011011	A/1	9	111111000	F/4	16	1111111111111000
4/8	16	1111111110011100	A/2	16	1111111111001001	F/5	16	1111111111111001
4/9	16	1111111110011101	A/3	16	1111111111001010	F/6	16	1111111111111010
4/A	16	1111111110011110	A/4	16	1111111111001011	F/7	16	1111111111111011
5/1	6	111011	A/5	16	1111111111001100	F/8	16	1111111111111100
5/2	10	1111111001	A/6	16	1111111111001101	F/9	16	1111111111111101
5/3	16	1111111110011111	A/7	16	1111111111001110	F/A	16	1111111111111111

例 设有一图像（256灰度级）分成了很多8×8的不重叠的像素块，其中一个亮度数据块如图所示，请将其进行JPEG编码。

62	51	61	64	77	61	64	73
63	55	66	93	107	85	68	75
62	58	68	119	149	100	69	74
73	68	61	126	150	116	72	71
47	71	60	114	128	90	70	73
65	75	64	70	75	62	51	74
85	70	60	52	55	68	69	86
78	71	65	68	61	76	70	99

源图像亮度数据块

根据前面JPEG编码的步骤可知，在图像数据分块以后，应该以MCU为单位顺序将DU进行二维离散余弦变换，对以无符号数表示的具有P位精度的输入数据，在DCT变换前要减去2^{P-1}，上图所示的图像矩阵应该每个像素都减去2^7，即减去128，得：

$$\begin{bmatrix} -66 & -77 & -67 & -64 & -51 & -67 & -64 & -55 \\ -65 & -73 & -62 & -35 & -21 & -43 & -60 & -53 \\ -66 & -70 & -60 & -9 & 21 & -28 & -59 & -54 \\ -55 & -60 & -67 & -2 & 22 & -12 & -56 & -57 \\ -81 & -57 & -68 & -14 & 0 & -38 & -58 & -55 \\ -63 & -53 & -64 & -58 & -53 & -66 & -77 & -54 \\ -43 & -58 & -68 & -76 & -73 & -60 & -59 & -42 \\ -50 & -57 & -63 & -60 & -67 & -52 & -58 & -29 \end{bmatrix}$$

然后进行离散余弦变换，得变换系数矩阵为：

-413.6250	-35.9992	-64.3446	23.2097	56.6250	-25.7988	-5.4135	-7.5730
15.7499	-13.6739	-59.5102	14.7796	19.0568	1.4968	-6.3755	12.9149
-52.4762	6.6543	79.8631	-22.1357	-22.5934	14.7510	20.8451	7.7844
-49.8452	14.2227	30.6398	-18.3392	-16.1161	-1.1170	-2.8832	4.3218
12.3750	-9.7205	-12.0741	3.2943	2.6250	0.4443	-5.9580	-6.3068
-1.0142	6.4061	14.3106	12.8172	-0.3569	3.3340	5.0937	3.7301
-6.8117	4.8567	-4.4049	-8.7864	3.2700	-5.9359	0.1369	3.1459
0.2278	-9.1509	-6.8891	-3.5041	1.0817	0.0517	-6.5535	4.6791

可以看出，能量集中在少数低频系数上，用亮度量化表对系数矩阵量化后的结果如后图所示，对量化结果进行Z形扫描，并对扫描结果的DC及AC系数进行编码的结果见后表。

-26	-3	-6	1	2	-1	0	0
1	-1	-4	1	1	0	0	0
-4	1	5	-1	-1	0	0	0
-4	1	1	-1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

量化结果

表 Zigzag排列及行程编码与哈夫曼编码结果

序号 k	0	1	2	3	4	5	6
数据 ZZ(k)	－26	－3	1	－4	－1	－6	1
NNNN/SSSS		0/2	0/1	0/3	0/1	0/3	0/1
编码结果		0100	001	100011	000	100001	001
序号 k	7	8	9	10	11	12	13
数据 ZZ(k)	－4	1	－4	1	1	5	1
NNNN/SSSS	0/3	0/1	0/3	0/1	0/1	0/3	0/1
编码结果	100011	001	100011	001	001	100101	001
序号 k	14	15	16	17	18	19	20
数据 ZZ(k)	2	－1	1	－1	1	0	0
NNNN/SSSS	0/2	0/1	0/1	0/1	0/1		
编码结果	0110	000	001	000	001		
序号 k	21	22	23	24	25	26～63	
数据 ZZ(k)	0	0	0	－1	－1	0	
NNNN/SSSS				5/1	0/1		0/0
编码结果				11110100	000		1010

DC系数的编码说明

- 在量化系数矩阵的Z形扫描结果中，第一个系数为DC系数。
- 假设前一亮度数据块的DC系数为−30，则差值DIFF=−26−(−30)=4，因为4在（−7~−4，4~7）范围内，由表查得SSSS=3，其前缀码字为“100”，3位尾码即为4的二进制原码“100”，从而DC系数的编码为“100100”。
- 如果前一数据块的DC系数为−22，则DIFF=−4，由表查得SSSS及前缀码字同上，其3位尾码即−4的反码“011”，因此DC系数的编码为“100011”。

AC系数的编码说明

- 在Z形扫描结果中，第一个非零AC系数为−3。在它前面的连零个数为0，即NNNN=0。根据AC系数−3从表查得SSSS=2，由NNNN/SSSS=0/2从表查得其哈夫曼码字为“01”，而−3<0，所以尾码为−3的反码“00”，因此该AC系数的编码为“0100”。
- 其他非零AC系数的编码结果见上表。在Z形扫描结果中的末尾，除第25个系数非零外，其他系数全为零，故直接用一个结束块“EOB（0/0）”结束本块，由表查得其码字为“1010”。于是，该亮度块的编码为（假定其中差值为4）
  1001000100001100011000100001001100011001100011001001100101001011000000100000111110100000 1010。
- 共用了92位，而原始图像块需8×8×8＝512位，因此压缩比为5.565。

活动图像压缩标准MPEG简介

- MPEG标准
  - MPEG是Moving Picture Experts Group的英文缩写，其含义是“活动图像专家组”，它是对活动的视频图像压缩的国际标准的简称。
  - 该专家组成立于1988年，它的工作不仅局限于活动图像编码，还把伴音和图像的压缩联系在一起，并且根据不同的应用场合，定义了不同的标准。
  - MPEG-1是1993年8月正式通过的技术标准，其全称为“适用于约1.5Mbit/s以下数字存储媒体的运动图像及伴音的编码”。这里所指的数字存储媒体包括CD-ROM、DAT、硬盘、可写光盘等，同时利用该标准也可以在ISDN或局域网中进行远程通信。

活动图像压缩标准MPEG简介

- MPEG标准
  - MPEG-2是1994年11月发布的“活动图像及伴音通用编码”标准，该标准可以应用于2.048Mbit/s ~ 20Mbit/s的各种速率和各种分辨率的应用场合之中，如多媒体计算机、多媒体数据库、多媒体通信、常规数字电视、高清晰度电视以及交互电视等。

活动图像压缩标准MPEG简介

- MPEG标准
  - MPEG-4的情况是：1999年1月公布了该标准的V1.0版本，同年12月公布了V2.0版本。该标准主要应用于超低速系统之中，例如多媒体Internet、视频会议和视频电视等个人通信、交互式视频游戏和多媒体邮件、基于网络的数据业务、光盘等交互式存储媒体、远程视频监视及无线多媒体通信。特别是它能够满足基于内容的访问和检索的多媒体应用，且其编码系统是开放的，可随时加入新的有效算法模块。

活动图像压缩标准MPEG简介

- MPEG标准
  - MPEG-7是2000年11月颁布的称为“多媒体内容描述接口”的标准。定义该标准的目的是制定出一系列的标准描述符来描述各种媒体信息。这种描述与多媒体信息的内容有关，这样将便于用户进行基于内容和对象的视听信息的快速搜索。MPEG－7与其他MPEG标准的不同之处在于它提供了与内容有关的描述符，并不包括具体的视音频压缩算法，而且还未形成与内容提交有关的所有标准的总框架。
  - MPEG-21的全称为“多媒体框架”。该标准的目的在于为多媒体用户提供透明而有效的电子交易和使用环境。

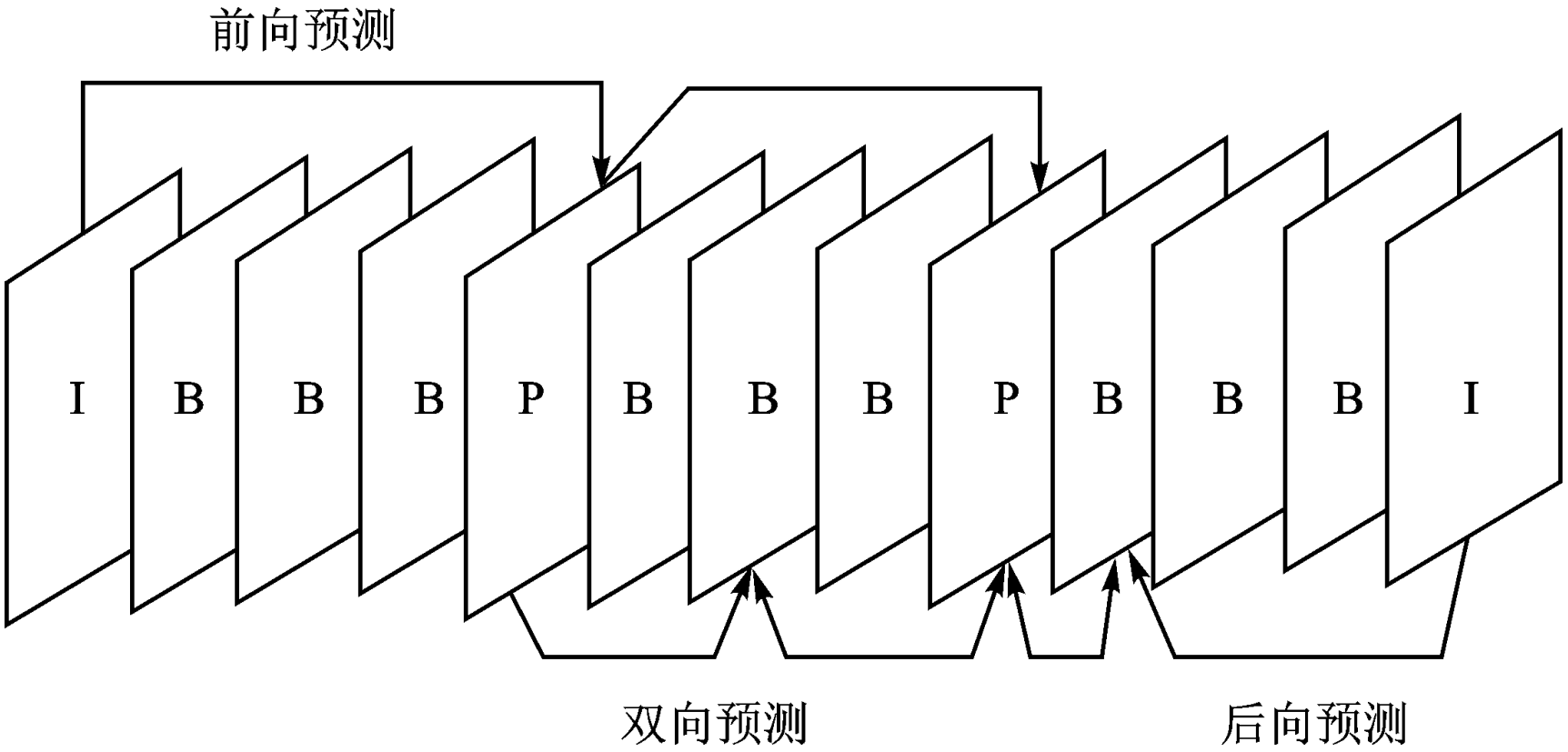
MPEG视频压缩方法

- 运动图像的压缩包括两个方面：
  - 帧内压缩：删除空间的数据冗余；
  - 帧间压缩：删除帧与帧之间的时间冗余。
- MPEG标准在空间域的压缩，类似于JPEG标准，每一帧被作为独立的图像获取，且压缩步骤与JPEG标准的步骤一样。
  - 帧间编码的基本思想是仅存储运动图像从一帧到下一帧的变化部分，而不是存储全部图像数据。这样做能极大地减少运动图像数据的存储量，达到帧间压缩的目的，这是通过把帧序列划分为I帧、P帧和B帧，使用参照帧及运动补偿技术来实现的。

帧的类型

- I帧：在编码时，无需参照任何其他帧的帧称为I帧，它是利用自身的相关性进行帧内压缩编码。
- P帧：在帧编码时，仅使用最近的前一帧（I帧或P帧）作为参照帧，该帧称为P帧或称为预测帧。
- B帧：在帧编码时，要使用前、后帧作为参照帧时，该帧称为B帧或双向预测帧。

下图给出了这3种不同图像类型的相互关系。



I、P和B三种不同图像类型的相互关系

运动补偿技术

- 在帧间编码中，运动补偿技术是提高帧间压缩的有效方法。运动补偿技术主要用于消除P帧和B帧在时间上的冗余。
- 在对P帧或B帧进行编码时，以宏块为基本编码单位，一个宏块的大小一般可定义为16×16。
  - 对于B帧，每一16×16宏块有4种类型，分别是帧内宏块（简称I块）、前向预测宏块（简称F块）、后向预测宏块（简称B块）、平均宏块（简称A块）。
  - 对于P帧，每一16×16宏块只有I块和F块两种。
  - 无论B帧还是P帧，I块编码与I帧编码技术一致。F块、B块、A块都是采用基于块的运动补偿技术。

运动补偿技术

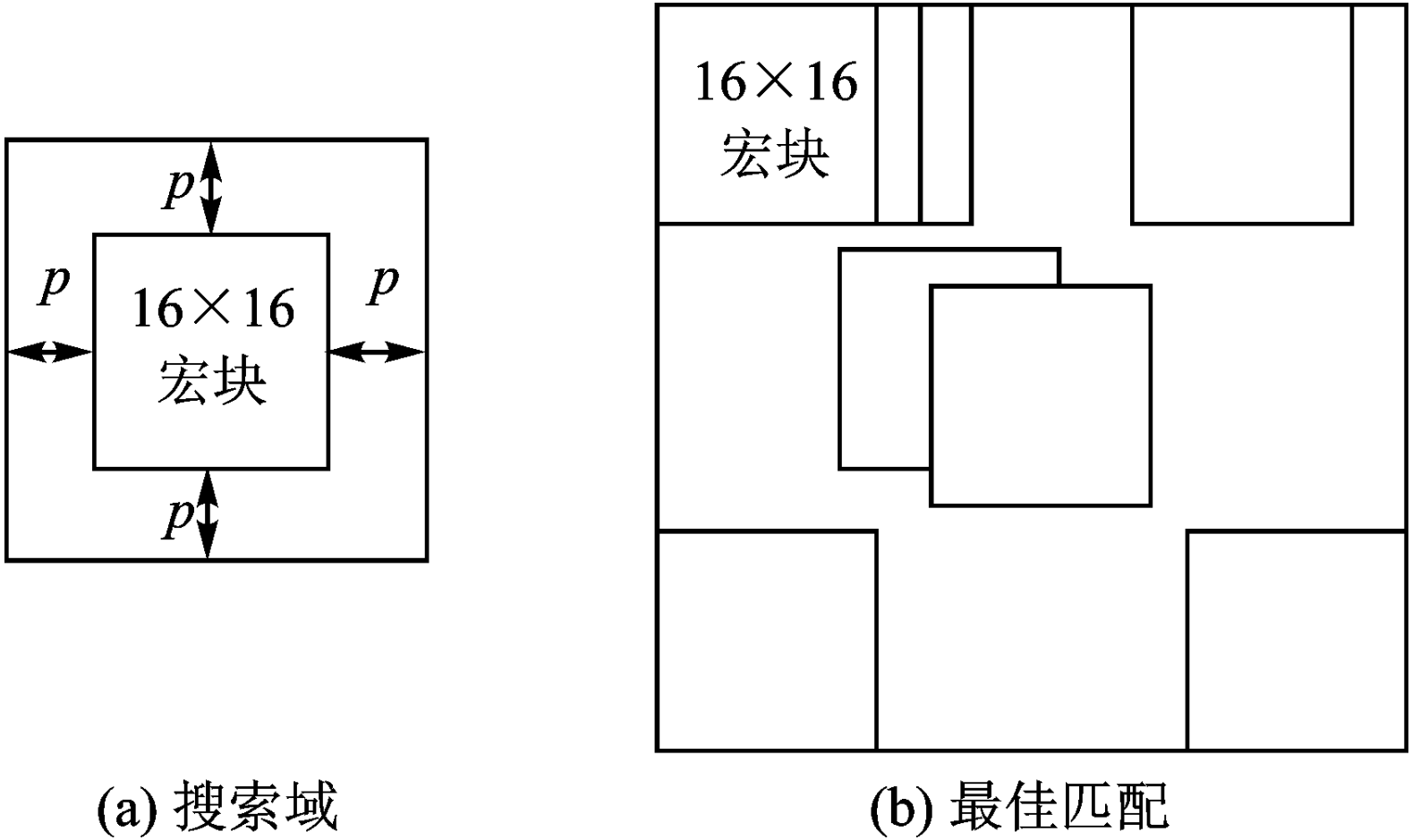
- 基于块的运动补偿技术是在参照帧中寻找与当前编码块最佳匹配的宏块。
  - 所谓最佳匹配是指这两个宏块之间差值最小，通常可以用绝对值AE最小作为匹配依据。

$$AE = \sum_{i=0}^{15}\sum_{j=0}^{15}\left| f(i,j) - g(i-dx, j-dy)\right|$$

式中，f是参照帧宏块；g为当前编码宏块；dx、dy是参照宏块在x和y方向上的运动矢量，它反映了从一帧到另一帧时，宏块仅仅是位置发生了改变，而内容并没有改变。

寻找最佳匹配块

- 在参照帧中寻找最佳匹配块的方法有块匹配法、梯度匹配法和相位匹配法。
  - 下图表示的是最简单的块匹配搜索方式，其中，搜索域为$(16+2p)\times(16+2p)$，p的取值范围可视运动图像的运动快慢而定，一般取$p=6$。
  - 搜索的基本方法是：每走一步，判断当前块与参照帧中的块是否完全相同或基本相同，若找到了基本匹配块，根据原来位置与新位置就可以得到运动矢量$(\mathrm{d}x,\mathrm{d}y)$。即首先在参照帧中找到与当前块相匹配的最佳宏块，并将宏块之差进行编码，同时对两宏块之间产生的运动矢量进行编码；解码时，根据运动矢量再借助参照帧的内容还原当前宏块。



(a) 搜索域　　(b) 最佳匹配

搜索域与最佳匹配

图像压缩编码的MATLAB实现

- 算术编码

```
% Main函数（主程序）
clear all
format long e;
symbol=['abcd'];
ps=[0.4 0.2 0.1 0.3];
inseq=('dacab');
codeword=suanshubianma(symbol,ps
,inseq)
outseq=suanshujiema(symbol,ps,co
deword,length(inseq))
```

```
%% 算术编码函数 arithmetic_encode
function
acode=arithmetic_encode(symbol,ps,inseq)
    high_range=[];
    for k=1:length(ps)
        high_range=[high_range sum(ps(1:k))];
    end
    low_range=[0 high_range(1:length(ps-1))];
    sbidx=zeros(size(inseq));
    for i=1:length(inseq)
        sbidx(i)=find(symbol==inseq(i));
    end
    low=0;
    high=1;
    for i=1:length(inseq)
        range=high-low;
        high=low+range*high_range(sbidx(i));
        low=low+range*low_range(sbidx(i));
    end
    acode=low;
end
```

```
%% 算术编码函数 arithmetic_decode
function symbos=arithmetic_decode(symbol,ps,codeword,symlen)
    format long e
    high_range=[];
    for k=1:length(ps)
        high_range=[high_range sum(ps(1:k))];
    end
    low_range=[0 high_range(1:length(ps)-1)];
    psmin=min(ps);
    symbos=[];
    for i=1:symlen
        idx=max(find(low_range<=codeword));
        codeword=codeword-low_range(idx);
        if abs(codeword-ps(idx))<0.01*psmin
            idx=idx+1;
            codeword=0;
        end
        symbos=[symbos symbol(idx)];
        codeword=codeword/ps(idx);
        if abs(codeword)<0.01*psmin
            i=symlen+1;
        end
    end
end
```

运行结果为：

```
codeword = 7.739200000000001e-01
outseq = dacab
```

行程编码

将图像cameraman.tif，转化为二值图像，然后对二值图像进行行程编码。

```
IY=imread('cameraman.tif');
I=im2bw(IY,0.35);
% 调用travel_encode进行编码
[zipped,info] = travel_encode(I);
% 调用travel_decode进行解码
Unzipped = travel_decode(zipped,info);
imshow(IY);
title('原始灰度图像')
figure;imshow(I);                    %显示二值图像
title('二值图像')
figure;imshow(unzipped);             %显示解码图像
title('解码图像')
unzipped=uint8(unzipped);
% 计算均方根误差得erms=0，表示行程编码是无失真编码。
erms=jfwucha(I(:),unzipped(:))
% 显示压缩比
cr=info.ratio
whos IY I unzipped zipped
```

```
%% 行程编码函数 travel_encode
function[zipped,info] = travel_encode(vector)
    [m,n]  = size(vector);
    vector = vector(:)';
    vector = uint8(vector(:));
    L = length(vector);
    c = vector(1);
    e(1,1) = c;
    e(1,2) = 0;
    t1 = 1;
    for j = 1:L
        if((vector(j) == c))
            e(t1,2) = double(e(t1,2))+1;
        else
            c  = vector(j);
            t1 = t1+1;
            e(t1,1) = c;
            e(t1,2) = 1;
        end
    end
    zipped = e;
    info.rows = m;
    info.cols = n;
    [m,n] = size(e);
    info.ratio = m*n/(info.rows*info.cols);
end
```

```
%% 行程解码函数 travel_decode
function unzipped = travel_decode(zip,info)
    zip = uint8(zip);
    [m,n] = size(zip);
    unzipped = [];
    for i = 1:m
        section = repmat(zip(i,1),1,double(zip(i,2)));
        unzipped = [unzipped section];
    end
    unzipped = reshape(unzipped,info.rows,info.cols);
    unzipped = double(unzipped);
end
```

```
%% 计算均方根误差函数 square_err
function erms = square_err(f1,f2)
    e = double(f1)-double(f2);
    [m,n] = size(e);
    erms = sqrt(sum(e(:).^2)/(m*n));
    if erms ~= 0
        emax = max(abs(e(:)));
        [h,x] = hist(e(:));
        if length(h) >= 1
            figure(4);
            bar(x,h,'k');
            e = mat2gray(e,[-emax,emax]);
            figure(5);
            imshow(e);
        end
    end
end
```

运行结果为:

```
erms = 0
cr = 0.1081
Name            Size            Bytes              Class
I               256x256         65536          logical array
IY              256x256         65536          uint8 array
unzipped        256x256         65536          uint8 array
zipped          3543x2          7086           uint8 array
Grand total is 203694 elements using 203694 bytes
```

(a) 原始图像

(b) 二值图像

(c) 解码图像

行程编码程序结果图

预测编码

对peppers.bmp图像进行无损一阶预测编码，采用前值预测。

```
% Main函数（主程序）
X=imread('peppers.bmp','bmp');
imshow(X)
title('原始图像')
X=double(X);
Y=Yucebianma(X);
XX=Yucejiema(Y);
figure,imshow(mat2gray(Y));
title('预测误差图像')
e=double(X)-double(XX);
[m,n]=size(e);
erms=sqrt(sum(e(:).^2)/(m*n));
[h,x]=hist(X(:));
figure;
bar(x,h,'k');
title('原图直方图')
[h,x]=hist(Y(:));
figure;
bar(x,h,'k');
title('预测误差直方图')
XX=uint8(XX);
figure;
imshow(XX);
title('解码图像')
```

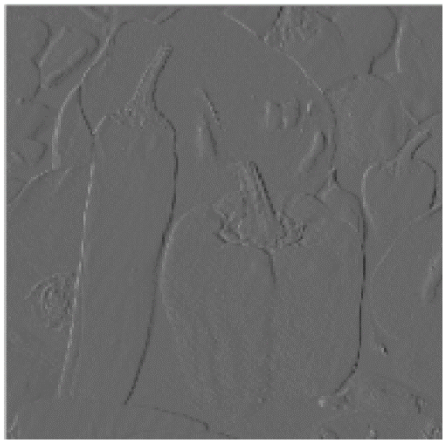
```
%% Yucebianma预测编码函数
% 一维无损预测编码压缩图像x，f为预测系数，如果f默认，则f=1，即为前值预测。
function Y=Yucebianma(x,f)
    error(nargchk(1,2,nargin))
    if nargin<2
        f=1;
    end
    x=double(x);
    [m,n]=size(x);
    p=zeros(m,n);
    xs=x;
    zc=zeros(m,1);
    for j=1:length(f)
        xs=[zc xs(:,1:end-1)];
        p=p+f(j)*xs;
    end
    Y=x-round(p);
end
```

```
%% Yucejiema是解码程序，与编码程序用的是同一个预测器。

function x = Yucejiema(Y,f)
    error(nargchk(1,2,nargin));
    if nargin < 2
        f = 1;
    end
    f = f(end:-1:1);
    [m,n] = size(Y);
    order = length(f);
    f = repmat(f,m,1);
    x = zeros(m,n+order);
    for j = 1:n
        jj = j + order;
        x(:,jj) = Y(:,j)+round(sum(f(:,order:-1:1).*x(:,(jj-
1):-1:(jj-order)),2));
    end
    x = x(:,order+1:end);
end
```

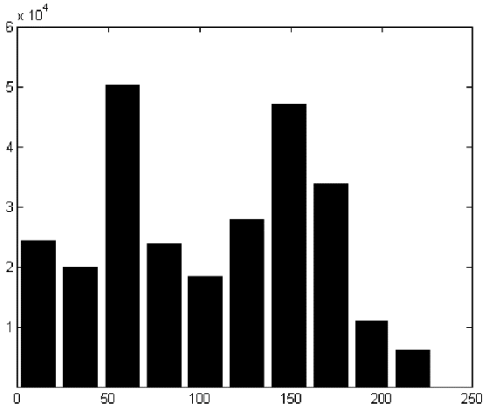
运行结果为：

(a) 原始图像

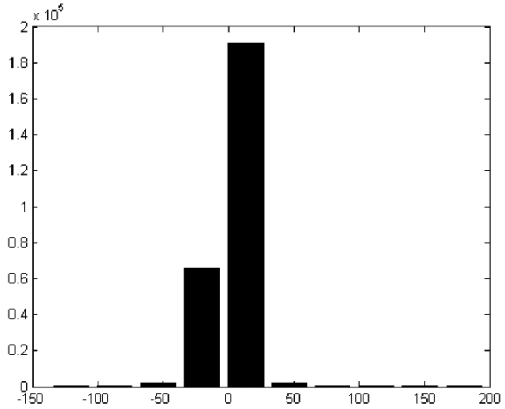


(b) 预测误差图像

(c) 解码图像



(d) 原图直方图



(e) 预测误差直方图

预测编码程序运行结果图

思考题

1. 简述图像压缩编码的必要性和可能性。

2. 编写一个实现哈夫曼编码的程序，要求用实际图像作为例子对其进行编码压缩，并计算熵、平均码长和编码效率。

3. 现有8个待编码的符号$\{l_1, l_2, \ldots, l_8\}$，这些符号出现的概率分别为{0.38，0.20，0.14，0.11，0.08，0.04，0.03，0.02}，请用香农－范诺编码方法对这8个符号进行编码。

思考题

4. 假设信源符号为$\{a, b, c, d\}$，假设这些符号出现的概率分别为$\{0.2, 0.2, 0.4, 0.2\}$，写出对信源符号$bdadc$进行算术编码和解码的过程。

5. 简述预测编码的基本原理。

6. 正交变换编码的基本原理是什么？

7. 什么是变换区域编码？什么是变换阈值编码？

8. 图像编码有哪些国际标准？其基本的应用对象是什么？