



## 第十二讲：图像压缩编码

任课教师：寇旗旗

计算机科学与技术学院



- 为减少视频信号的数据量，实现有效的传输和存储，因此，需要关注压缩编码技术。
- 利用图像固有的统计特性，以及视觉生理、心理学特性，或者记录设备和显示设备等的特性，从原始图像中经过压缩编码提取有效信息，尽量去除无用的或用处不大的冗余信息，以便高效率的进行图像的数字传输或存储，而且在复原时仍能够获得与原始图像差不多的复原图像。



# 主要内容

- 1 图像压缩与编码基本概念
- 2 编码方法
  - 2.1 哈夫曼编码
  - 2.2 香农-范诺编码
  - 2.3 行程编码
  - 2.4 算术编码
  - 2.5 变换编码
- 3 静止图像压缩编码标准-JPEG
- 4 运动图像压缩编码标准-MPEG

# 12.1 图像压缩与编码基本概念



- 为什么要进行图像压缩
- 图像数据压缩的可能性
- 常见的图像数据冗余
- 图像压缩的目的
- 图像数据压缩技术的重要指标

# 12.1.1为什么要进行图像压缩？



数字图像通常要求很大的比特数，这给图像的传输和存储带来相当大的困难。要占用很多的资源，花很高的费用。

如一幅512 x 512的灰度图象的比特数为

$$512 \times 512 \times 8 = 256k$$

一张A4(210mm×297mm) 大小的照片，若用中等分辨率(300dpi)的扫描仪按真彩色扫描，其数据量为多少？（注：dpi表示每英寸像素，1英寸=25.4mm）

若按每像素3个字节计算，上述结果为约26M




# 12.1.1为什么要进行图像压缩？



- 如一部**90分钟**的彩色电影，每秒放映**24帧**。  
把它数字化，每帧**512×512**像素，每像素的**R、G、B**三分量分别占**8b**，则总比特数为  
$$90 \times 60 \times 24 \times 3 \times 512 \times 512 \times 8 \text{bit} = \mathbf{97,200 \text{MB}}$$
- 如一张**CD**光盘可存**600MB**数据，存储这部电影光图像（还有声音）就需要**160**张**CD**光盘。
- 因此，传输带宽、速度、存储器容量的限制使得对图象数据进行压缩显得非常必要。

# 12.1.2. 图像数据压缩的可能性



-  一般原始图像中存在很大的冗余度。
-  用户通常允许图像失真。
-  用户对原始图像的信号不全都感兴趣，可用特征提取和图像识别的方法，丢掉大量无用的信息。提取有用的信息，使必须传输和存储的图像数据大大减少。



# 12.1.3. 常见的图像数据冗余

## 图像编码的基本理论

图像压缩所要解决的问题是尽量减少表示数字图像时需要的数据量，而减少数据量的基本原理是去除其中的冗余数据。

### ■ 数字图像中的冗余？

----- 编码冗余，像素间冗余，心理视觉冗余。

如果能减少或消除其中的一种或多种冗余，就能取得数据压缩效果。

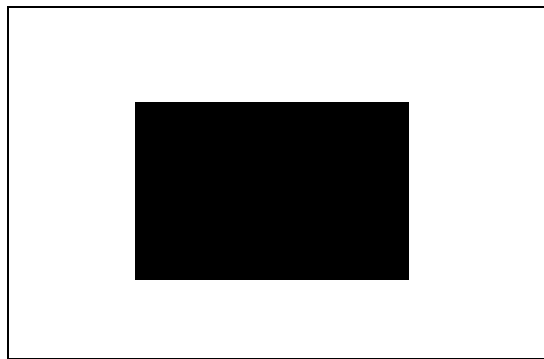


# 12.1.3. 常见的图像数据冗余



- (1) 编码冗余:

编码冗余，又称为信息熵冗余、如果一幅图像的灰度级编码，使用了多于实际需要的编码位数，就称该图像包含了编码冗余。



例：如果用8位表示该图像的像素，我们就说该图像存在着编码冗余，因为该图像的像素只有两个灰度，用一位即可表示。

# 12.1.3. 常见的图像数据冗余



## 图像编码的基本理论

### (1) 编码冗余

在大多数图像中，图像像素的灰度值分布是不均匀的，因此若对图像的灰度值直接进行自然二进制编码（等长编码），则会对有最大和最小概率可能性的值分配相同比特数，而产生了编码冗余。

# 12.1.3. 常见的图像数据冗余

## 图像编码的基本理论

### (1) 编码冗余



162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
164	164	158	156	161	160	159	160
161	161	163	158	160	162	159	156
159	159	156	157	159	159	156	157

1010 0001



# 12.1.3. 常见的图像数据冗余

## 图像编码的基本理论

### (1) 编码冗余

用自然二进制编码时没有考虑像素灰度值出现的概率。只有按概率分配编码长度，才是最精减的编码方法。

162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
162	162	162	161	162	157	163	161
164	164	158	156	161	160	159	160
161	161	163	158	160	162	159	156
159	159	156	157	159	159	156	157

灰度级出现概率大的用短码表示，出现概率小的灰度级用长码表示，则有可能使编码总长度下降。

# 12.1.3. 常见的图像数据冗余



- (2) 像素间冗余:

任何给定的像素值，原理上都可以通过它的邻域内的像素值预测到，单个像素携带的信息相对是少的。

对于一幅图像，很多单个像素对视觉的贡献是冗余的。这是建立在对邻域内像素预测的基础上。

原始图像越有规则，各像素之间的相关性越强，它可能压缩的数据就越多。

# 12.1.3. 常见的图像数据冗余



## 图像编码的基本理论

### (2) 像素间冗余

对应图像目标的像素之间一般具有相关性。  
因此，图像中一般存在与像素间相关性直接联系着的数据冗余——像素相关冗余。





# 12.1.3. 常见的图像数据冗余



## 图像编码的基本理论

### (2) 像素间冗余

- 空间冗余。邻近像素灰度分布的相关性很强。
- 频间冗余。多谱段图像中各谱段图像对应像素之间灰度相关性很强。
- 时间冗余。序列图像帧间画面对应像素灰度的相关性很强。
- 结构冗余。有些图像存在较强的纹理结构或自相似性，如墙纸、草席等图像。
- 知识冗余。有些图像中包含与某些先验知识相关的信息

# 12.1.3. 常见的图像数据冗余

## (3) 心理视觉冗余

## 图像编码的基本理论

人的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，这些信息可以认为是心理视觉冗余的。







## 12.1.4. 图像压缩的目的

图像数据压缩的目的是在满足一定图像质量条件下，用尽可能少的比特数来表示原始图像，以提高图像传输的效率和减少图像存储的容量。在信息论中称为信源编码。

图像从结构上大体上可分为两大类，一类是具有一定图形特征的结构，另一类是具有一定概率统计特性的结构。

基于不同的图像结构特性，应采用不同的压缩编码方法。

# 12.1.5 图像数据压缩技术的重要指标



(1) **压缩比**：图像压缩前后所需的信息存储量之比，压缩比越大越好。

(2) **压缩方法**：利用不同的编码方式，实现对图像的数据压缩。

(3) **失真性**：压缩前后图像存在的误差大小。

## 12.1.5 图像数据压缩技术的重要指标



- 一幅灰度级为 $K$ 的图像，第 $k$ 个灰度级出现的概率为 $p_k$ ， $B_k$ 为编码后第 $k$ 个灰度级对应的比特数，图像大小为 $M \times N$ ，编码前每个像素用 $d$ 比特表示，每两帧图像间隔 $\Delta t$ ，则数字图像的熵 $H$ :

$$H = - \sum_{k=1}^K p_k \log_2 p_k$$

# 12.1.5 图像数据压缩技术的重要指标



- 图像的平均码字长度 $R$ 为：

$$R = \sum_{k=1}^K B_k p_k$$

- 编码效率伊塔 $\eta$ 定义为：
- 

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

## 12.1.5 图像数据压缩技术的重要指标



- 信息冗余度为：

$$\nu = 1 - \eta$$

- 每秒钟所需的传输比特数bps为：

$$bps = \frac{M \times N \times R}{\Delta t}$$

- 压缩比r为：

$$r = \frac{d}{R}$$

# 12.1.5 图像数据压缩技术的重要指标



全面评价一种编码方法的优劣，除了看它的**编码效率**、**实时性**和**失真度**以外，还要看它的**设备复杂程度**，是否**经济与实用**。

常采用混合编码的方案，以求在性能和经济上取得折衷。

随着计算方法的发展，使许多高效而又比较复杂的编码方法在工程上有实现的可能。

# 12.1.6 图像编码中的保真度准则



图像信号在编码和传输过程中会产生误差，尤其是在有损压缩编码中，产生的误差应在允许的范围之内。在这种情况下，保真度准则可以用来衡量编码方法或系统质量的优劣。通常，这种衡量的尺度可分为客观保真度准则和主观保真度准则。



# 12.1.6 图像编码中的保真度准则

## (1) 客观保真度准则

通常使用的客观保真度准则有输入图像和输出图像的**均方根误差**；输入图像和输出图像的**均方根信噪比**两种。

**均方根误差**：设输入图像是由 $N \times N$ 个像素组成，令其为 $f(x, y)$ ，其中 $x, y=0, 1, 2, \dots, N-1$ 。这样一幅图像经过压缩编码处理后，送至受信端，再经译码处理，重建原来图像，这里令重建图像为 $g(x, y)$ 。它同样包含 $N \times N$ 个像素，并且 $x, y=0, 1, 2, \dots, N-1$ 。





# 12.1.6 图像编码中的保真度准则

在 $0, 1, 2, \dots, N-1$ 范围内 $x, y$ 的任意值, 输入像素和对应的输出图像之间的误差可用下式表示:

$$e(x, y) = g(x, y) - f(x, y)$$

而包含 $N \times N$ 像素的图像之均方误差为:

$$\begin{aligned} \overline{e^2} &= \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} e^2(x, y) \\ &= \frac{1}{N^2} \sum_{N=0}^{N-1} \sum_{N=0}^{N-1} [g(x, y) - f(x, y)]^2 \end{aligned}$$

由式可得到均方根误差



如果把输入、输出图像间的误差看作是噪声，那么，重建图像 $g(x,y)$ 可由下式表示：

$$g(x,y) = f(x,y) + e(x,y)$$

在这种情况下，另一个客观保真度准则——重建图像的均方信噪比如下式表示：

$$\begin{aligned} \left(\frac{S}{N}\right)_{ms} &= \frac{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} g^2(x,y)}{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} e^2(x,y)} \\ &= \frac{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} g^2(x,y)}{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} [g(x,y) - f(x,y)]^2} \end{aligned}$$



# 12.1.6 图像编码中的保真度准则

均方根信噪比为：

$$\left(\frac{S}{N}\right)_{rms} = \left\{ \frac{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} g^2(x, y)}{\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} [g(x, y) - f(x, y)]^2} \right\}^{\frac{1}{2}}$$

# 12.1.6 图像编码中的保真度准则



## (2) 主观保真度准则

图像处理的结果,大多是给人观看,由研究人员来解释的,因此,图像质量的好坏,既与图像本身的客观质量有关,也与视觉系统的特性有关。

有时候,客观保真度完全一样的两幅图像可能会有完全不同的视觉质量,所以又规定了主观保真度准则,这种方法是把图像显示给观察者,然后把评价结果加以平均,以此来评价一幅图像的主观质量。

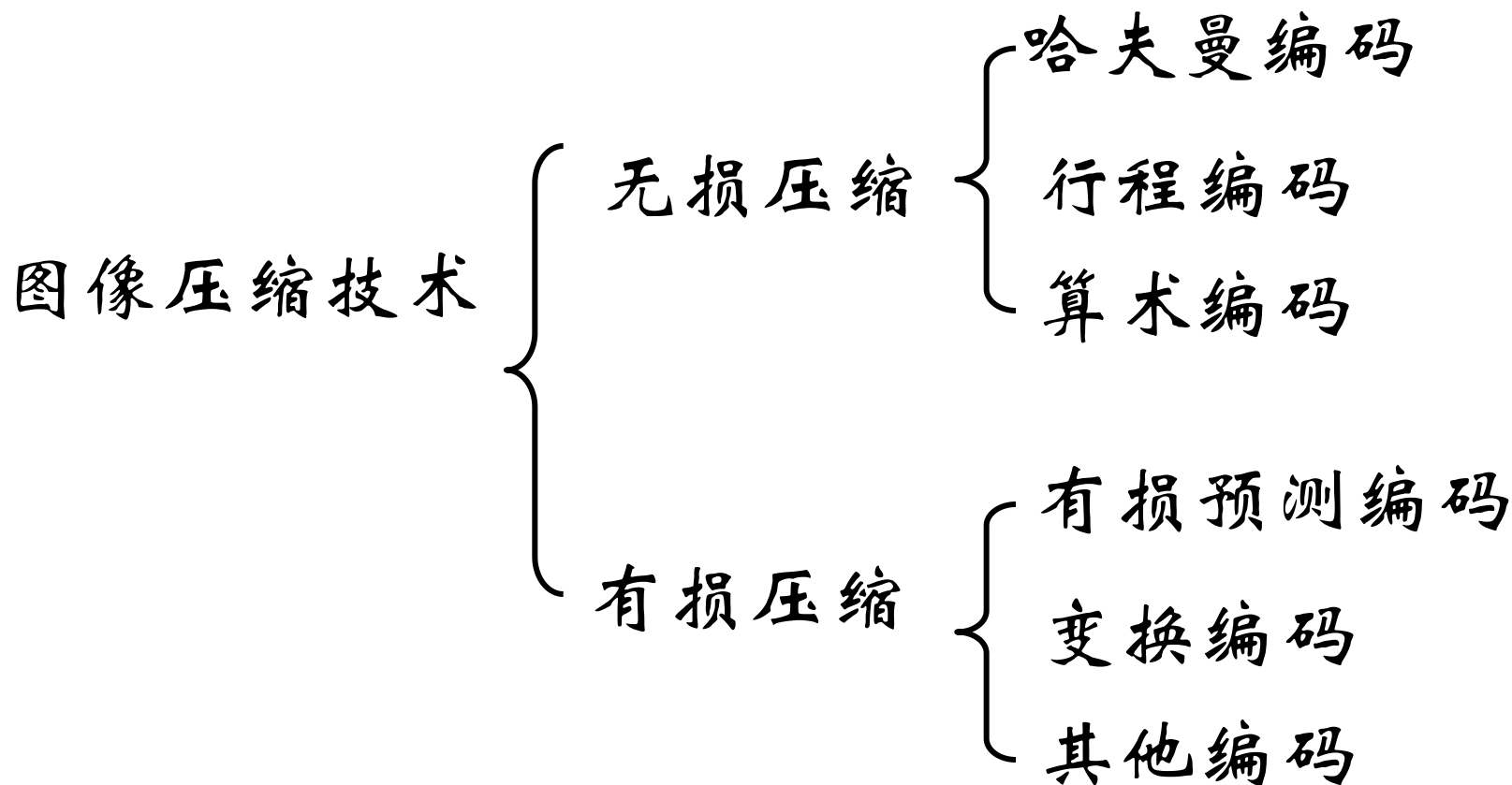
另外一种方法是规定一种绝对尺度,如:

# 12.1.6 图像编码中的保真度准则



评 分	评 价	说 明
1	优秀	图像质量非常好，如同人能想象出的最好质量。
2	良好	图像质量高，观看舒服，有干扰但不影响观看。
3	可用	图像质量可接受，有干扰但不太影响观看。
4	刚可看	图像质量差，有些干扰妨碍观看，观察者希望改进。
5	差	图像质量很差，妨碍观看的干扰始终存在，几乎无法观看
6	不能用	图像质量极差，不能观看

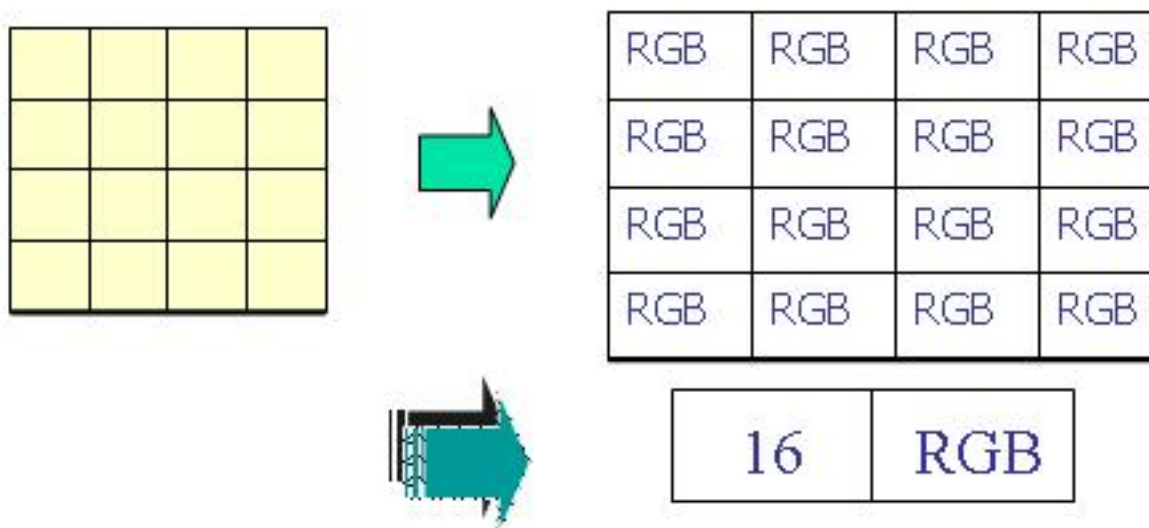
## 12.2. 常用的压缩编码方法



## 12.2. 常用的压缩编码方法

※ **无损压缩算法**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，无损压缩的压缩比很少有能超过3:1的，常用于要求高的场合。

### ■ 图像冗余无损压缩的原理

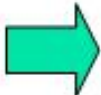


## 12.2. 常用的压缩编码方法

※ **有损压缩**是通过牺牲图像的准确率以实现较大的压缩率，如果容许解压图像有一定的误差，则压缩率可显著提高。有损压缩在压缩比大于30:1时仍然可重构图像，而如果压缩比为10:1到20:1，则重构的图像与原图几乎没有差别

### ■ 图像冗余有损压缩的原理

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

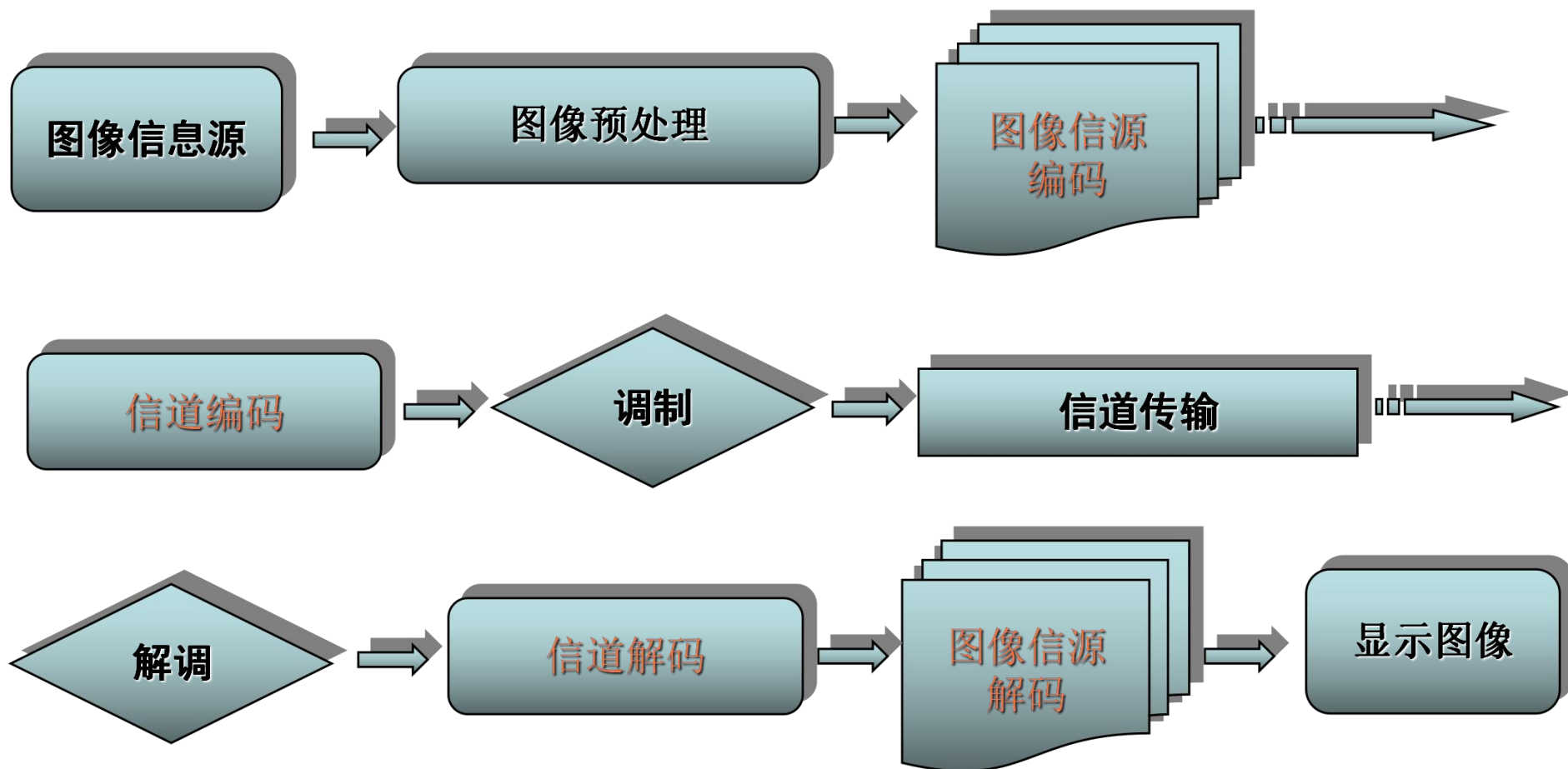


25	34
----	----

[返回](#)



# 12.2.1. 图像的压缩模型

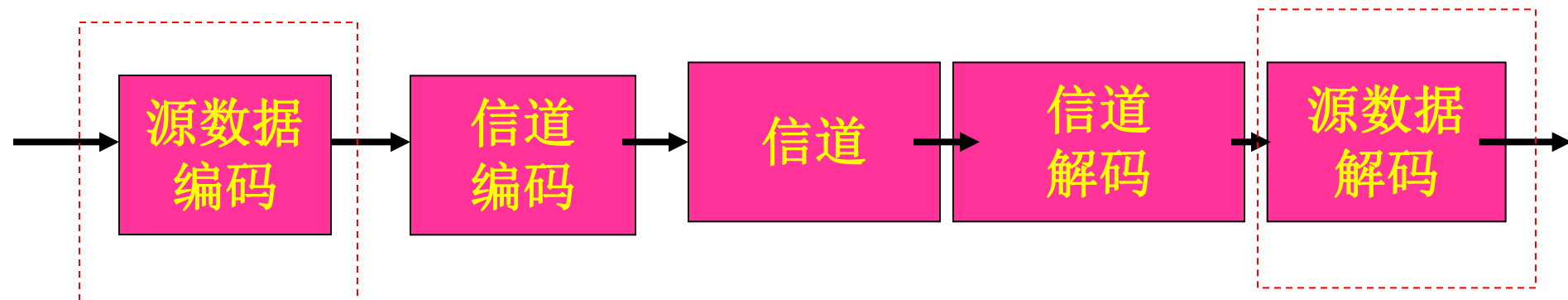


# 12.2.1. 图像的压缩模型

源数据编码：完成原数据的压缩。

信道编码：为了抗干扰，增加一些容错、校验位，实际上是增加冗余。

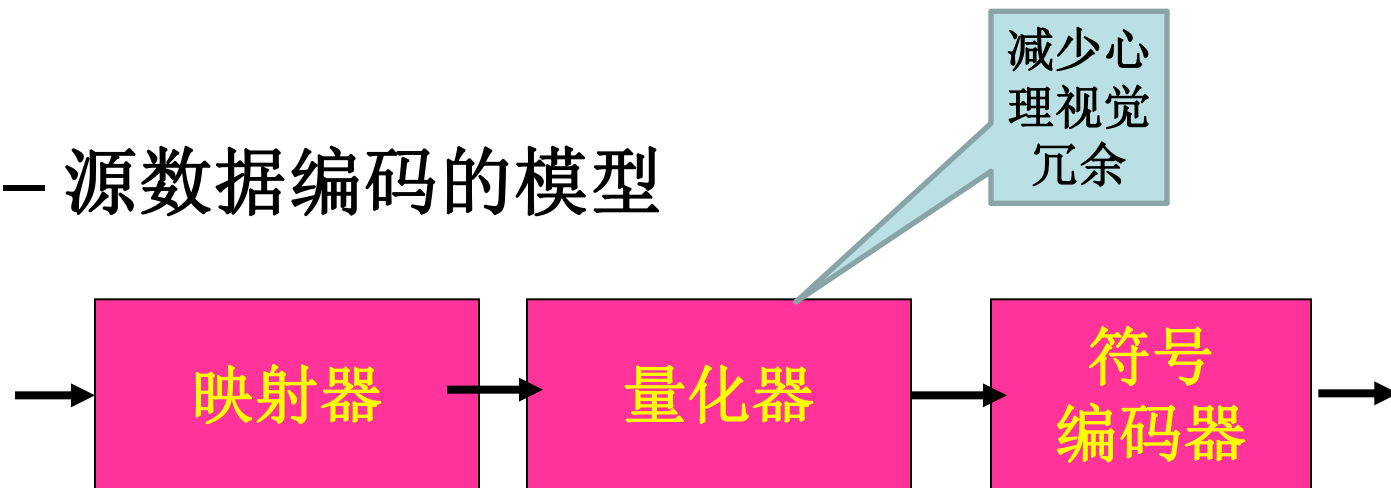
信道：如Internet、广播、通讯、可移动介质



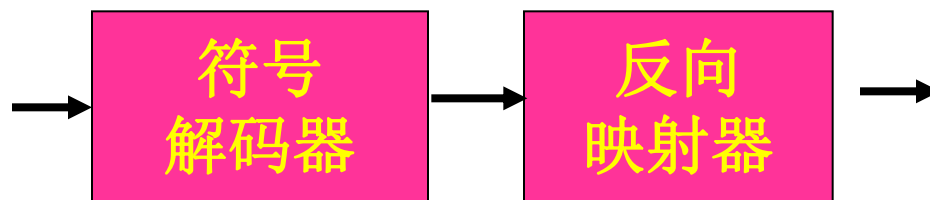
# 12.2.1. 图像的压缩模型

- 源数据编码与解码的模型

- 源数据编码的模型



- 源数据解码的模型



# 12.2.1. 图像的压缩模型

- 源数据编码与解码的模型

- 映射器：减少像素间冗余，如使用RLE编码。或进行图像变换
- 量化器：减少视觉心理冗余，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



## 12.2.2. 哈夫曼编码

哈夫曼编码是一种利用信息符号概率分布特性的变字长的编码方法。对于出现概率大的信息符号编以短字长的码，对于出现概率小的信息符号编以长字长的码。



## 12.2.2. 哈夫曼编码

- i. 将信源符号按出现概率从大到小排成一列，然后把最末两个符号的概率相加，合成一个概率，形成新的符号。
- ii. 把这个新符号的概率与其余符号的概率按从大到小排列，然后再把最末两个符号的概率加起来，合成一个概率，形成新的符号。
- iii. 重复上述做法，直到最后合成1个符号。
- iv. 从最后一步开始逐步向前进行编码。每步只需对两个分支各分配一个二进制码，如对概率大的分配0，对概率小的分配1。
- v. 搜索每个符号被合并到1的路径，反方向写出

## 12.2.2. 哈夫曼编码



### Huffman 编码

输入      输入概率

$S_1$       0.4

$S_2$       0.3

$S_3$       0.1

$S_4$       0.1

$S_5$       0.06

$S_6$       0.04

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入      输入概率    第一步

$S_1$             0.4            0.4

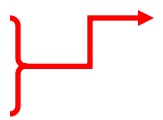
$S_2$             0.3            0.3

$S_3$             0.1            0.1

$S_4$             0.1            0.1

$S_5$             0.06          0.1

$S_6$             0.04





# 12.2.2. 哈夫曼编码



## Huffman 编码

输入 输入概率 第一步 第二步

$S_1$	0.4	0.4	0.4
$S_2$	0.3	0.3	0.3
$S_3$	0.1	0.1	0.2
$S_4$	0.1	0.1	0.1
$S_5$	0.06	0.1	
$S_6$	0.04		

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入	输入概率	第一步	第二步	第三步
$S_1$	0.4	0.4	0.4	0.4
$S_2$	0.3	0.3	0.3	0.3
$S_3$	0.1	0.1	0.2	0.3
$S_4$	0.1	0.1	0.1	
$S_5$	0.06	0.1		
$S_6$	0.04			

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入	输入概率	第一步	第二步	第三步	第四步	最后
$S_1$	0.4	0.4	0.4	0.4	0.6	1
$S_2$	0.3	0.3	0.3	0.3	0.4	
$S_3$	0.1	0.1	0.2	0.3		
$S_4$	0.1	0.1	0.1			
$S_5$	0.06	0.1				
$S_6$	0.04					

# 12.2.2. 哈夫曼编码



## Huffman 编码

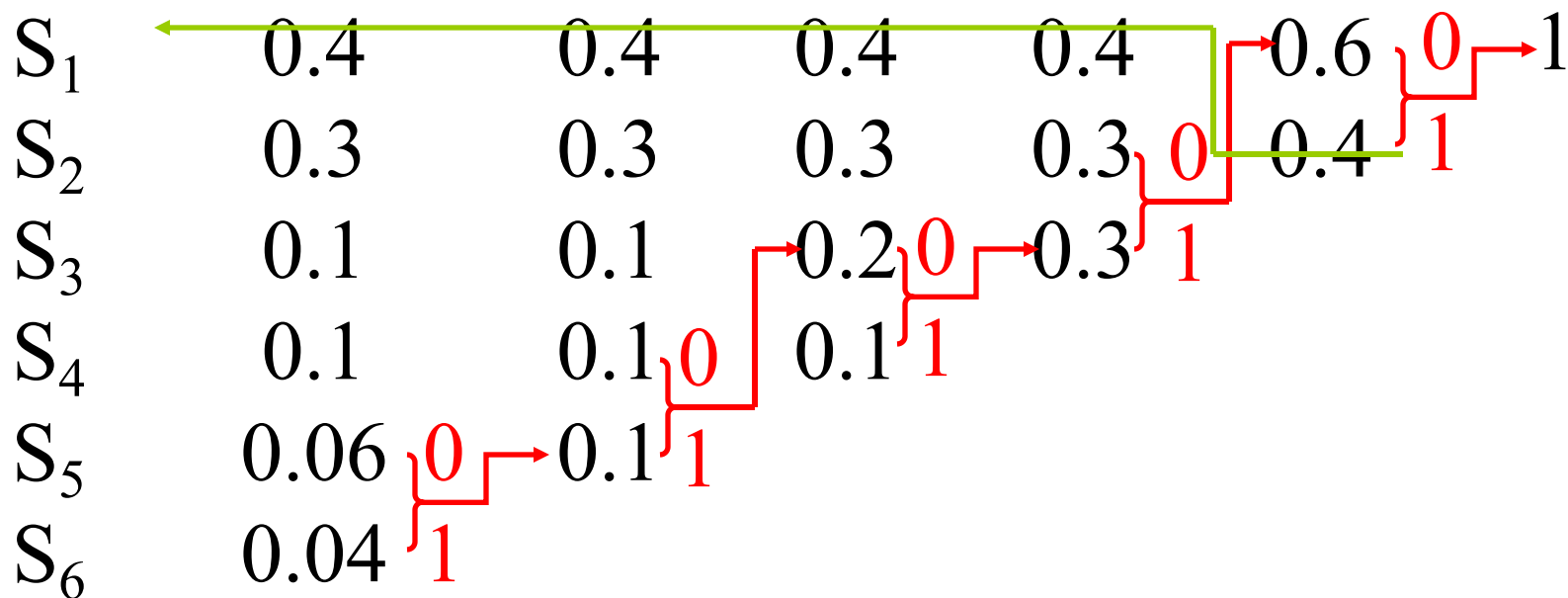
输入	输入概率	第一步	第二步	第三步	第四步	最后
$S_1$	0.4	0.4	0.4	0.4	0.6	1
$S_2$	0.3	0.3	0.3	0.3	0.4	1
$S_3$	0.1	0.1	0.2	0.3		
$S_4$	0.1	0.1	0.1			
$S_5$	0.06	0.1				
$S_6$	0.04					

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入 输入概率 第一步 第二步 第三步 第四步 最后



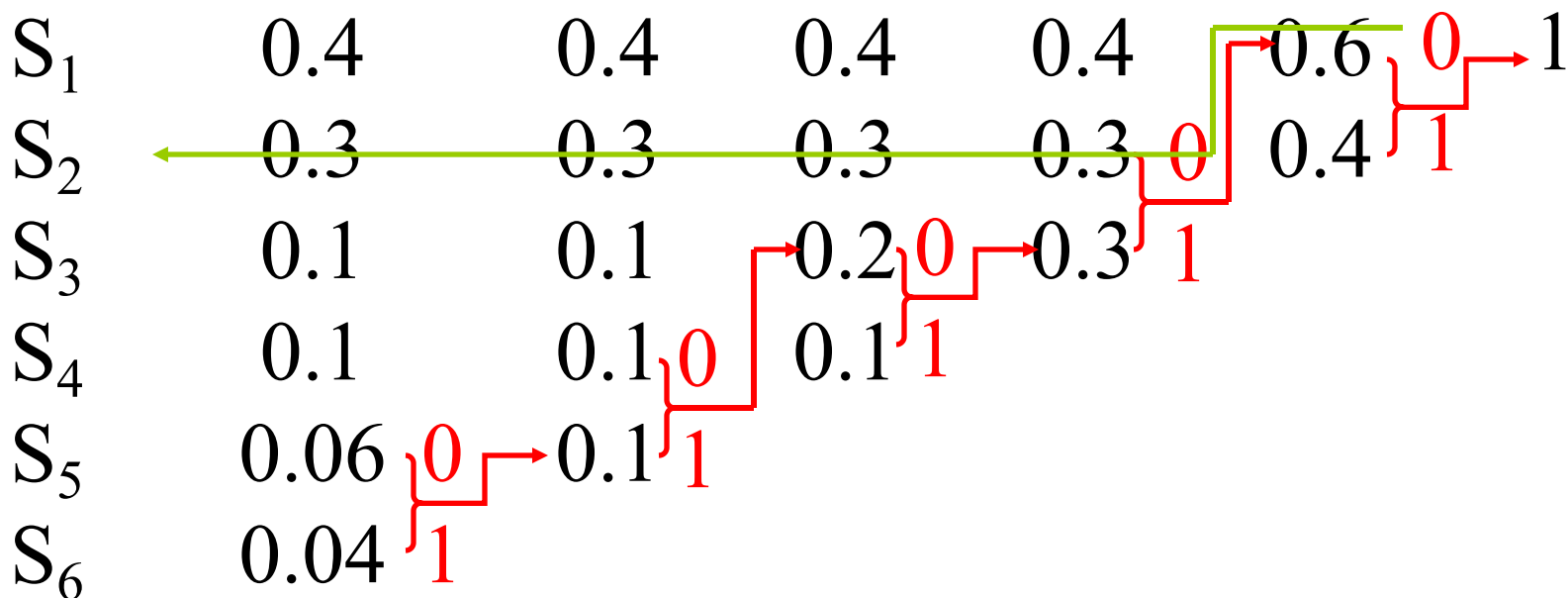
$$S_1=1$$

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入 输入概率 第一步 第二步 第三步 第四步 最后



$$S_2=00$$

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入 输入概率 第一步 第二步 第三步 第四步 最后

$S_1$	0.4	0.4	0.4	0.4	0.6	0	1
$S_2$	0.3	0.3	0.3	0.3	0.4	0	1
$S_3$	0.1	0.1	0.2	0.3	0.3	1	
$S_4$	0.1	0.1	0.1	0.1			
$S_5$	0.06	0.1					
$S_6$	0.04						

$$S_3 = 011$$

# 12.2.2. 哈夫曼编码



## Huffman 编码

输入 输入概率 第一步 第二步 第三步 第四步 最后

$S_1$	0.4	0.4	0.4	0.4	0.6	0	1
$S_2$	0.3	0.3	0.3	0.3	0.4	0	1
$S_3$	0.1	0.1	0.2	0.3	0.3	1	
$S_4$	0.1	0.1	0.1	0.1			
$S_5$	0.06	0.1					
$S_6$	0.04						

$$S_4 = 0100$$

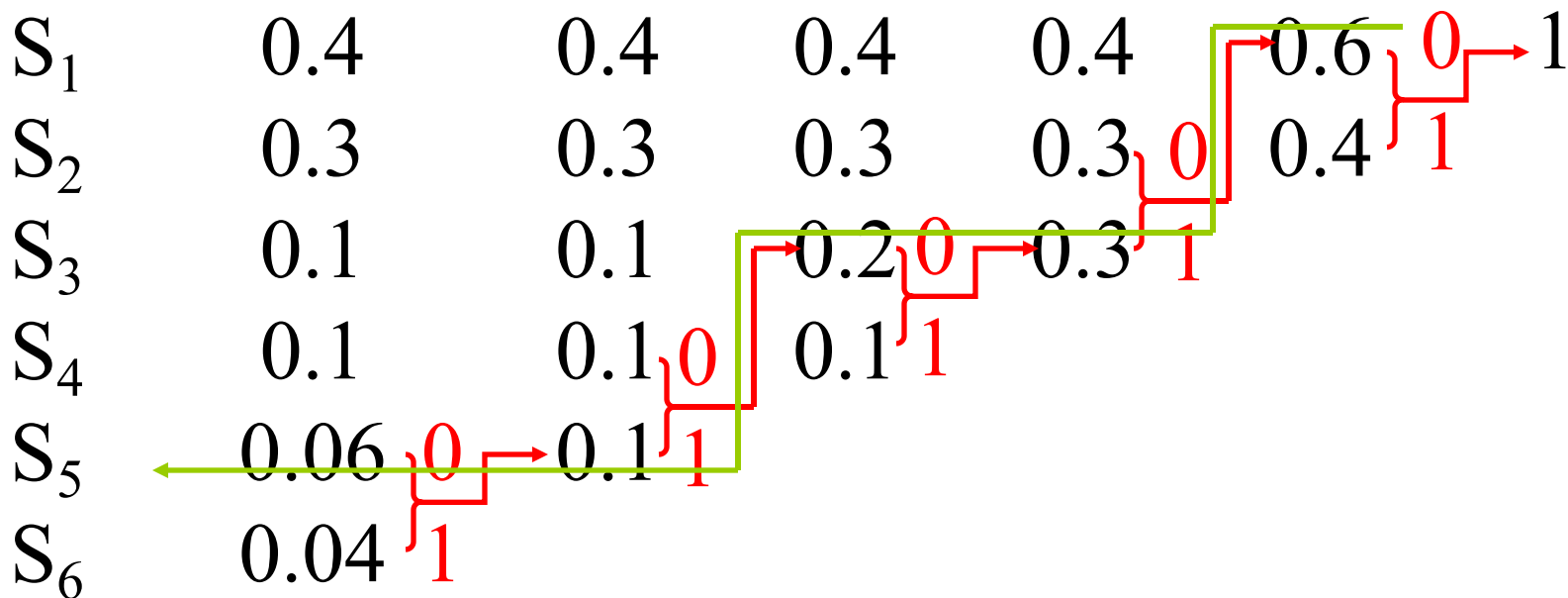


# 12.2.2. 哈夫曼编码



## Huffman 编码

输入 输入概率 第一步 第二步 第三步 第四步 最后



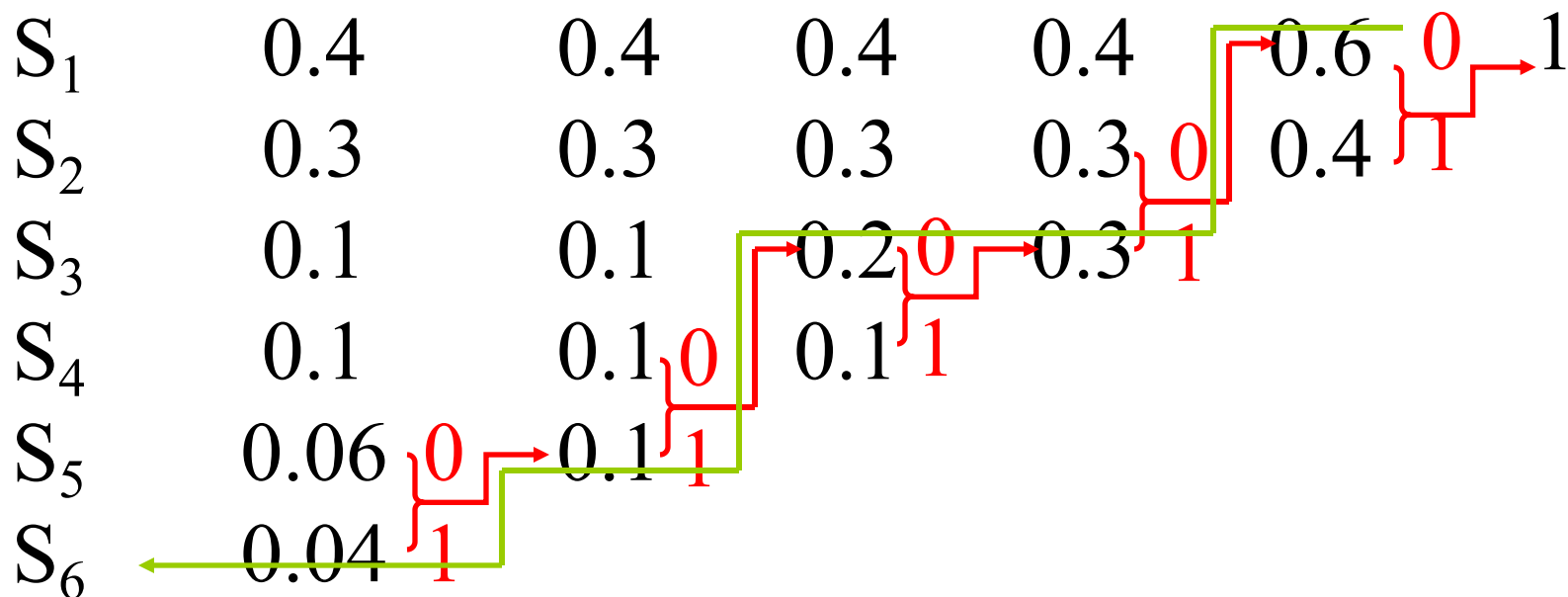
$$S_5 = 01010$$

# 12.2.2. 哈夫曼编码



## Huffman 编码

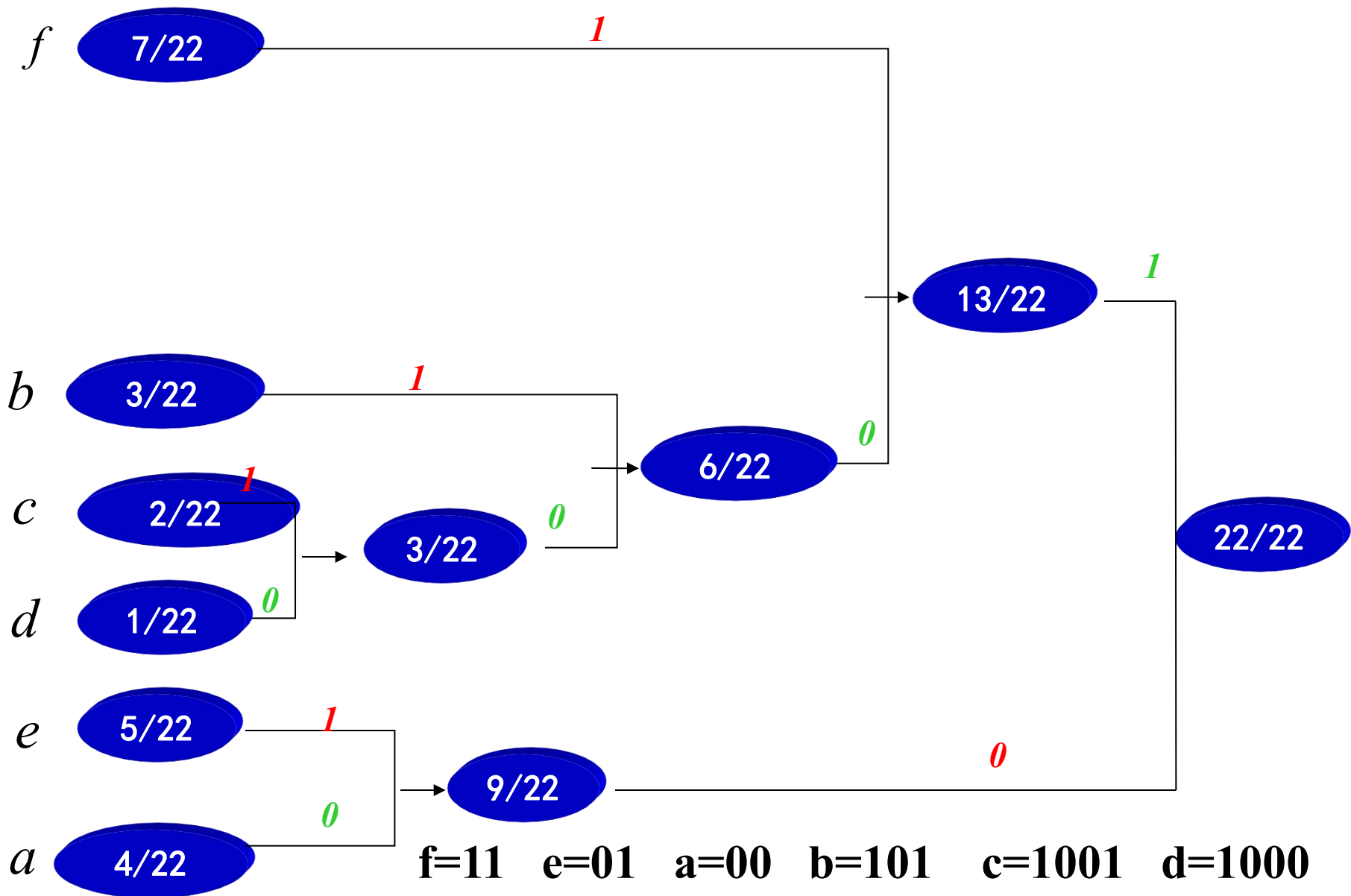
输入 输入概率 第一步 第二步 第三步 第四步 最后



$$S_6 = 01011$$



# 编码举例





## 12.2.2. 哈夫曼编码

概率分布为 2 的负幂次方				概率分布为均匀分布		
信源符号	出现概率	哈夫曼码字	码字长度	出现概率	哈夫曼码字	码字长度
$S_0$	$2^{-1}$	1	1	0.125	111	3
$S_1$	$2^{-2}$	01	2	0.125	110	3
$S_2$	$2^{-3}$	001	3	0.125	101	3
$S_3$	$2^{-4}$	0001	4	0.125	100	3
$S_4$	$2^{-5}$	00001	5	0.125	011	3
$S_5$	$2^{-6}$	000001	6	0.125	010	3
$S_6$	$2^{-7}$	0000001	7	0.125	001	3
$S_7$	$2^{-7}$	0000000	7	0.125	000	3
编码效率	$H=1.984\ 375$	$R=1.984\ 375$	$\eta=100\%$	$H=3$	$R=3$	$\eta=100\%$



## 12.2.2. 哈夫曼编码

对不同概率分布的信源符号，哈夫曼编码的编码效率有所差别。根据信息论中信源编码理论，对于二进制编码，当信源符号概率为2的负幂次方时，哈夫曼编码的编码效率可达100%，其平均码字长度也很短。信源符号概率为均匀分布时，其编码效果明显降低。在上表中，显然，第二种情况的概率分布也服从2的负幂次方，故其编码效率 $\eta$ 也可以达到100%，但由于服从均匀分布，其熵最大，平均编码长度很大，因此从其他指标看（如压缩比 $r$ ），其编码效果最低。在信源概率接近于均匀分布时，一般不使用哈夫曼编码。

## 12.2.2. 哈夫曼编码

1. 有如下信源 $X$ ,

$X =$

$u_1$	$u_2$	$u_3$	$u_4$	$u_5$			
$P_1$	$P_2$	$P_3$	$P_4$	$P_5$			

其中： $P_1 = 0.21$ ,  $P_2 = 0.48$ ,  $P_3 = 0.11$ ,  $P_4 = 0.13$ ,  
 $P_5 = 0.07$ 。

将该信源进行哈夫曼编码。



## 12.2.2. 哈夫曼编码

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

## 12.2.2. 香农—范诺编码



香农—范诺 (Shannon-Fannon) 编码也是一种典型的可变字长编码。与哈夫曼编码相似，当信源符号出现的概率正好为2的负幂次方时，香农—范诺编码的编码效率可以达到100%。





## 12.2.2. 香农—范诺编码

- 香农—范诺编码的理论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，对于二进制编码即有：

$$-\log_2 p_i + 1 \geq N_i \geq -\log_2 p_i \quad (10-1)$$



## 12.2.2. 香农—范诺编码

$$-\log_2 p_i + 1 \geq N_i \geq -\log_2 p_i$$

### 编码步骤

- (1) 将信源符号按其出现的概率由大到小顺序排列，若两个符号的概率相等，则相等概率的字符顺序可以任意排列；
- (2) 根据式(10-1)计算出各概率符号所对应的码字长度 $N_i$ ；
- (3) 将各符号的概率累加，计算累加概率 $P$ ，即：



## 12.2.2. 香农—范诺编码

$$\left\{ \begin{array}{l} P_0 = 0 \\ P_1 = p_0 \\ P_2 = p_0 + p_1 \\ P_3 = p_0 + p_1 + p_2 \\ \vdots \\ P_i = p_0 + p_1 + p_2 + \cdots + p_{i-1} = P_{i-1} + p_{i-1} \end{array} \right.$$



## 12.2.2. 香农—范诺编码

- (4) 把各个累加概率P由十进制转换为二进制;
- (5) 根据式(10-1)取二进制累加概率前 $N_i$ 位的数字, 并省去小数点前的“0.”字符, 即为对应信源符号的香农—范诺编码码字。

$$-\log_2 p_i + 1 \geq N_i \geq -\log_2 p_i$$

# 12.2.2. 香农—范诺编码



## 编码举例

例： 设一幅灰度级为8的图像中，各灰度级分别用 $S_0$ 、 $S_1$ 、 $S_2$ 、 $S_3$ 、 $S_4$ 、 $S_5$ 、 $S_6$ 、 $S_7$ 表示，对应的概率分别为0.40、0.18、0.10、0.10、0.07、0.06、0.05、0.04。现对其进行编码。

编码步骤如下



## 12.2.2. 香农—范诺编码

### 编码举例

- (1) 将信源符号按其出现概率由大到小顺序排列，为0.40, 0.18, 0.10, 0.10, 0.07, 0.06, 0.05, 0.04;
- (2) 对于概率0.40对应的符号 $S_0$ ，根据(10-1)计算 $N_0=2$ ，将累加概率0.00转换位二进制小数为0.00，取前 $N_0=2$ 位，并去除小数点前的字符，即 $S_0$ 字符编码为00;
- (3) 对于概率0.18对应的符号 $S_1$ ，根据(10-1)计算 $N_1=3$ ，将累加概率0.40转换位二进制小数为0.0110，取前 $N_1=3$ 位，并去除小数点前的字符，即 $S_1$ 字符编码为011;

$$-\log_2 p_i + 1 \geq N_i \geq -\log_2 p_i$$



# 12.2.2. 香农—范诺编码

## 编码举例

- (4) 对于概率0.10对应的符号S2, 根据(10-1)计算 $N_2=4$ , 将累加概率0.58转换位二进制小数为0.10010, 取前 $N_2=4$ 位, 并去除小数点前的字符, 即S2字符编码为1001;
- (5) 对于概率0.10对应的符号S3, 根据(10-1)计算 $N_3=4$ , 将累加概率0.68转换位二进制小数为0.10100, 取前 $N_3=4$ 位, 并去除小数点前的字符, 即S3字符编码为1010;
- (6) 对于概率0.07对应的符号S4, 根据(10-1)计算 $N_4=4$ , 将累加概率0.78转换位二进制小数为0.11000, 取前 $N_4=4$ 位, 并去除小数点前的字符, 即S4字符编码为1100;

$$-\log_2 p_i + 1 \geq N_i \geq -\log_2 p_i$$



# 12.2.2. 香农—范诺编码

## 编码举例

- (7) 对于概率0.06对应的符号S5，根据(10-1)计算 $N_5=5$ ，将累加概率0.85转换位二进制小数为0.1101100，取前 $N_5=5$ 位，并去除小数点前的字符，即S5字符编码为11011；
- (8) 对于概率0.05对应的符号S6，根据(10-1)计算 $N_6=5$ ，将累加概率0.91转换位二进制小数为0.1110100，取前 $N_6=5$ 位，并去除小数点前的字符，即S6字符编码为11101；
- (9) 对于概率0.04对应的符号S7，根据(10-1)计算 $N_7=5$ ，将累加概率0.68转换位二进制小数为0.11110100，取前 $N_7=5$ 位，并去除小数点前的字符，即S7字符编码为11110；

$$-\log_2 p_i + 1 \geq N_i \geq -\log_2 p_i$$



## 12.2.2. 香农—范诺编码效能



- (1) 图像信息熵为

$$H = - \sum_{k=0}^7 p_k \log_2 p_k = 2.55$$

- (2) 平均码字长度为

$$R = \sum_{k=1}^K B_k p_k$$

## 12.2.2. 香农—范诺编码效能



$$\begin{aligned} &= 2 \times 0.40 + 3 \times 0.18 + 4 \times 0.10 + 4 \times 0.10 + \\ &4 \times 0.07 + 5 \times 0.06 + 5 \times 0.05 + 5 \times 0.04 \\ &= 3.17 \end{aligned}$$

效率为:

$$\eta = \frac{H}{R} \times 100\% = \frac{2.55}{3.17} = 80.4\%$$

信息冗余度为

$$\nu = 1 - \eta = 19.6\%$$



## 12.2.3. 行程编码

### RLE 编码——Run Length Encoding

#### — 概念:

- 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。

例: 设重复次数为  $iC$ , 重复像素值为  $iP$

编码为:  $iPiC \ iPiC \ iPiC$

编码前: aaaaaaabbbbbbcccccccc

编码后: a7b6c8



## 12.2.3. 行程编码

### □ 行程长度编码RLE (Run Length Encoding) :

一幅图像中可能有许多颜色相同的图像块，用一整数对存储一个像素的颜色值及相同颜色像素的数目（长度）。例如：

(G , L)

颜色  
值      长度

编码时采用从左到右，从上到下的排列，  
每当遇到一串相同数据时就用该数据及  
重复次数代替原来的数据串。

000000003333333333  
2222222222226666666  
111111111111111111  
111115555555555555  
888888888888888888  
5555555555555553333  
222222222222222222

(0,8) (3,10) (2,11) (6,7)  
(1,18) (1,6) (5,12) (8,18)  
(5,14) (3,4) (2,18)

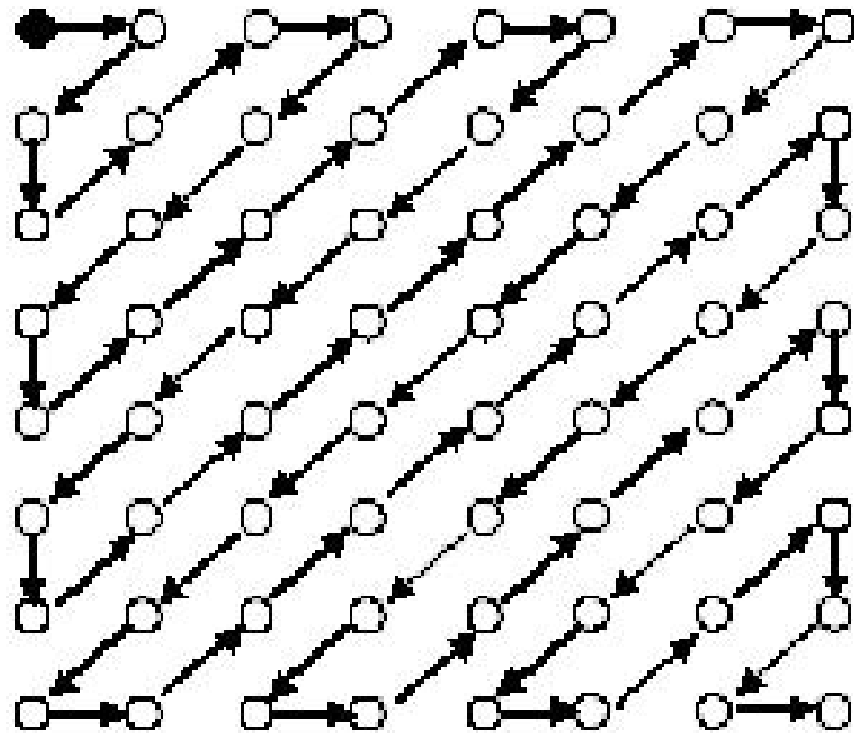
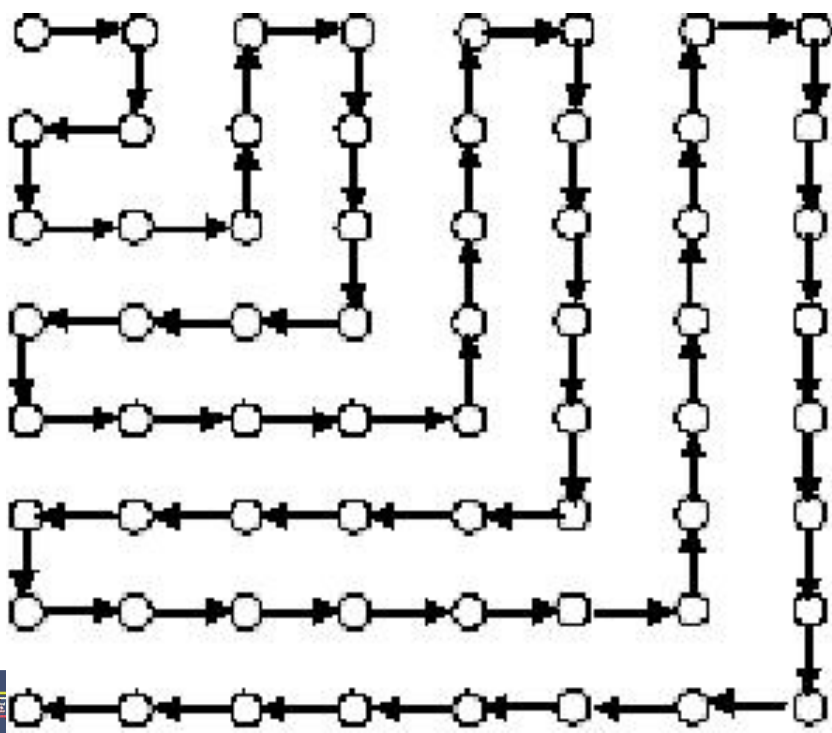
18\*7的像素颜色仅用11对数据

## 12.2.3. 行程编码



### 二维行程编码

- 二维行程编码要解决的核心问题是：将二维排列的像素，采用某种方式转化成一维排列的方式。之后按照一维行程编码方式进行编码
- 两种典型的二维行程编码的排列方式



## 12.2.3. 行程编码

### 二维行程编码

- 数据量:  $64 \times 8 = 512(\text{bit})$

$$f = \begin{bmatrix} 130 & 130 & 130 & 129 & 134 & 133 & 129 & 130 \\ 130 & 130 & 130 & 129 & 134 & 133 & 130 & 130 \\ 130 & 130 & 130 & 129 & 132 & 132 & 130 & 130 \\ 129 & 130 & 130 & 129 & 130 & 130 & 129 & 129 \\ 127 & 128 & 127 & 129 & 131 & 129 & 131 & 130 \\ 127 & 128 & 127 & 128 & 127 & 128 & 132 & 132 \\ 125 & 126 & 129 & 129 & 127 & 129 & 133 & 132 \\ 127 & 125 & 128 & 128 & 126 & 130 & 131 & 131 \end{bmatrix}$$



## 12.2.3. 行程编码

### 二维行程编码

如果按照方式(a)扫描的顺序排列的话，数据分布为：

130, 130, 130, 130, 130, 130, 130, 130, 130; 129, 129, 129, 129, 129, 130, 130, 129; 127, 128, 127, 129, 131, 130, 132, 134, 134; 133, 133, 132, 130, 129, 128, 127, 128, 127, 128, 127, 125, 126, 129, 129; 127, 129, 133, 132, 131, 129, 130, 130; 129, 130, 130, 130, 129, 130, 132, 132; 131, 131, 130, 126, 128, 128, 127, 127

行程编码为：

(7, 130), (2, 130), (4, 129), (2, 130), (1, 129); (1, 127), (1, 128), (1, 127), (1, 129), (1, 131), (1, 130), (1, 132), (2, 134), (2, 133), (1, 132), (1, 130), (1, 129), (1, 128), (1, 127), (1, 128), (1, 127), (1, 128), (1, 127), (1, 125), (1, 126), (2, 129), (1, 127), (1, 129), (1, 133), (1, 132), (1, 131), (1, 129), (2, 130), (1, 129), (3, 130), (1, 129), (1, 130), (2, 132), (2, 131), (1, 130), (1, 126), (2, 128), (2, 127)

数据量为： $43 * (3+8) = 473(\text{bit})$  (94.22%)

# 12.2.3. 行程编码



## 二维行程编码

- RLE 编码——Run Length Encoding

— 分析：

- 对于有大面积色块的图像，压缩效果很好
- 直观，经济，是一种无损压缩
- 对于复杂的图像，压缩效果不好，最坏情况下，会加倍图像的数据量

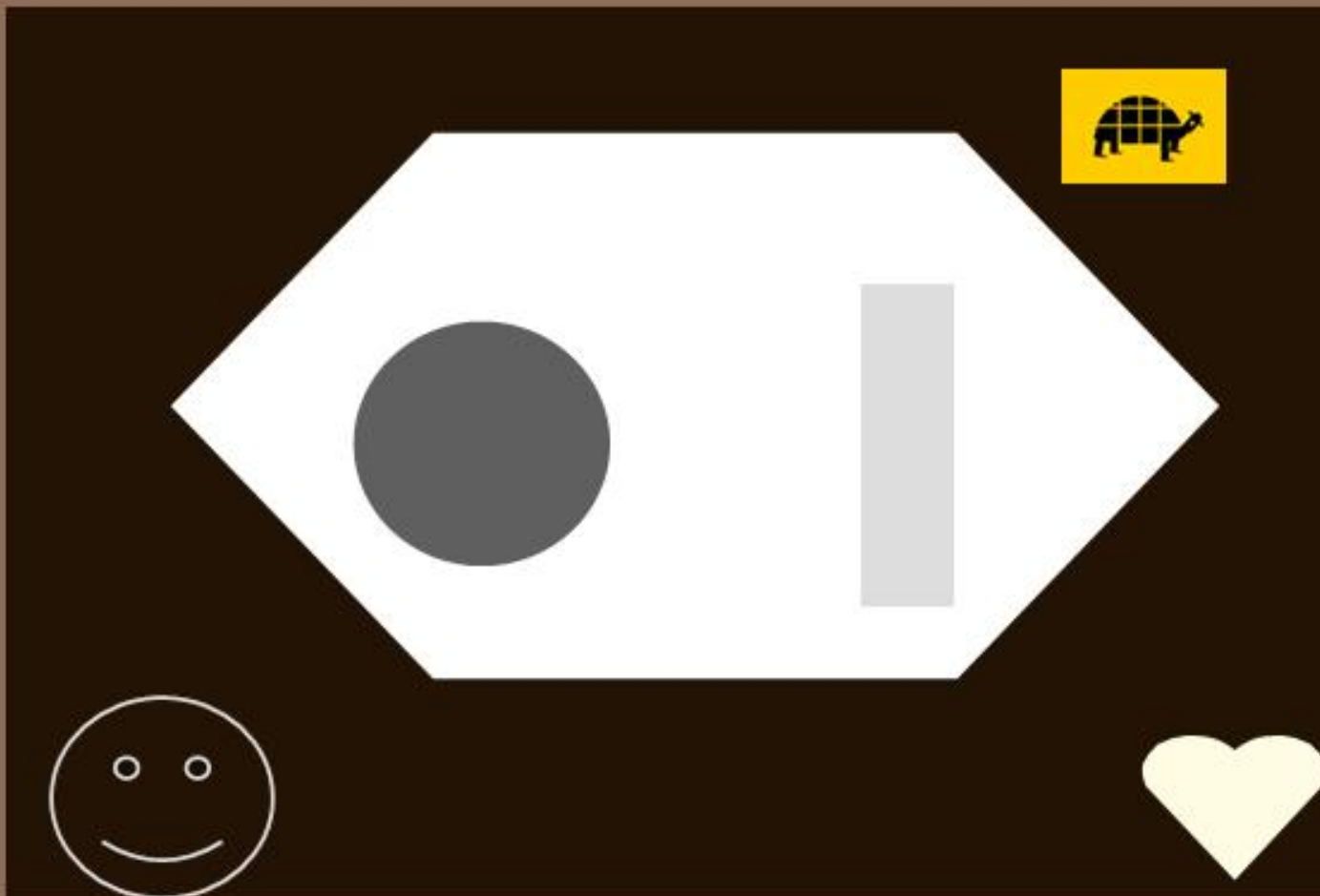


# 12.2.3. 行程编码



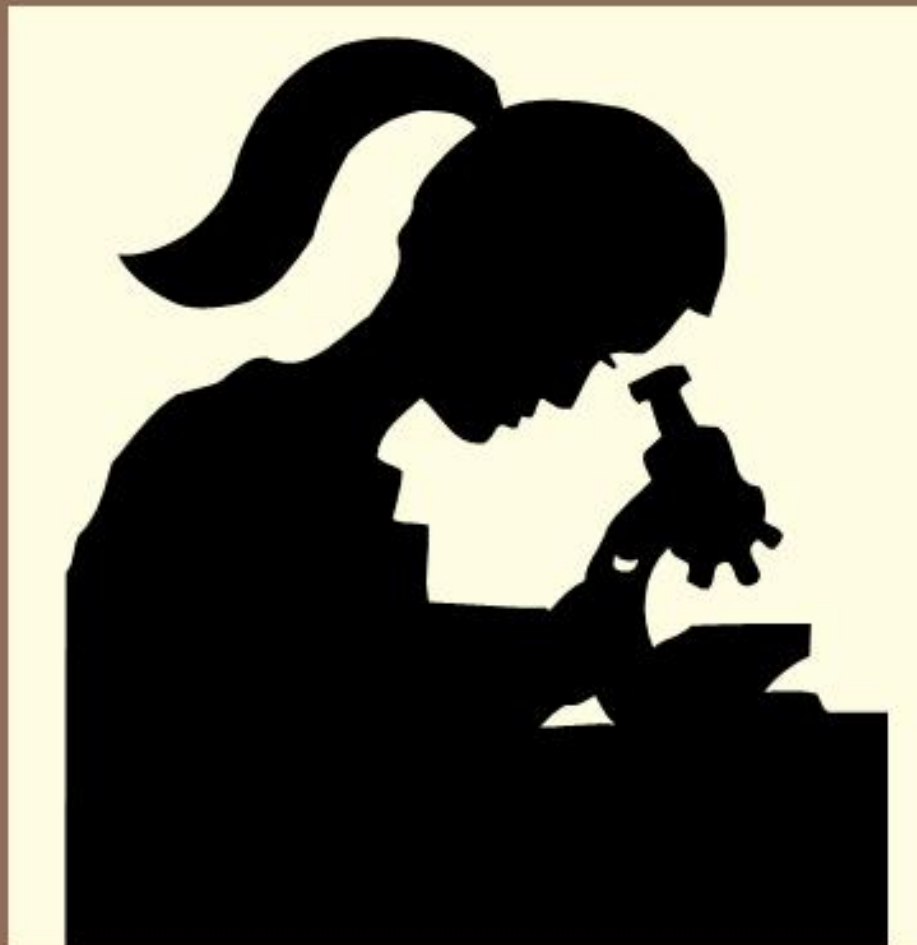
适合行程编码的图

## 12.2.3. 行程编码



适合行程编码的图

## 12.2.3. 行程编码



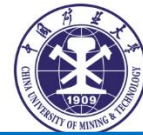
适合行程编码的图

## 12.2.3. 行程编码



适合行程编码的图

## 12.2.4. 算术编码



- 从理论上分析，采用哈夫曼编码可以获得最佳信源字符编码效果；
- 实际应用中，由于信源字符出现的概率并非满足2的负幂次方，因此往往无法达到理论上的编码效率和信息压缩比；

## 12.2.4. 算术编码



以信源字符序列  $\{x, y\}$  为例

- 设字符序列  $\{x, y\}$  对应的概率为  $\{1/3, 2/3\}$ ， $N_x$  和  $N_y$  分别表示字符  $x$  和  $y$  的最佳码长，则根据信息论有：

$$N_x = -\log_2\left(\frac{1}{3}\right) = 1.58$$

$$N_y = -\log_2\left(\frac{2}{3}\right) = 0.588$$

## 12.2.4. 算术编码



- 字符 $x$ 、 $y$ 的最佳码长分别为1.58bit和0.588bi;
- 这表明, 要获得最佳编码效果, 需要采用小数码字长度, 这是不可能实现的;
- 即采用哈夫曼方法对 $\{x, y\}$ 的码字分别为0和1, 也就是两个符号信息的编码长度都为1。对于出现概率大的字符 $y$ 并未能编码成较短的码字;
- 实际编码效果往往不能达到理论效率;
- 为提高编码效率, Elias等人提出了算术编码算法。

# 12.2.4. 算术编码



## 算术编码的特点

- 算术编码是信息保持型编码，它不像哈夫曼编码，无需为一个符号设定一个码字；
- 算术编码分为固定方式和自适应方式两种编码；
- 选择不同的编码方式，将直接影响到编码效率；
- 自适应算术编码的方式，无需先定义概率模型，适合于无法知道信源字符概率分布的情况；
- 当信源字符出现的概率比较接近时，算术编码效率高于哈夫曼编码的效率，在图像通信中常用它来取代哈夫曼编码；
- 实现算术编码算法的硬件比哈夫曼编码复杂。



## 算术编码的原理

- 算术编码方法是将被编码的信源消息表示成 $0\sim 1$ 之间的一个间隔，即小数区间，消息越长，编码表示它的间隔就越小；
- 以小数表示间隔，表示的间隔越小所需的二进制位数就越多，码字就越长。反之，间隔越大，编码所需的二进制位数就少，码字就短。
- 算术编码将被编码的图像数据看作是由多个符号组成的字符序列，对该序列递归地进行算术运算后，成为一个二进制数；
- 接收端解码过程也是算术运算，由二进制数重建图像符号序列。

## 12.2.4. 算术编码



### 算术编码的步骤

- 1 把当前区间定义为 $[0, 1)$
- 2 把当前区间分割为长度正比于符号概率的子区间
- 3 为输入的符号 $s$ 选择一个子区间，并将定义为当前区间
- 4 重复2，3直到输入的符号处理完毕，将最后的区间的一个数转化为二进制形式，取小数点后面的部分二进制作作为相应的编码

## 12.2.4. 算术编码



### 编码的举例

- 设图像信源编码可用a、b、c、d这4个符号来表示，若图像信源字符集为{dacba}，信源字符出现的概率分别如下表所示，采用算术编码对图像字符集编码。

信源字符	a	b	c	d
出现概率	0.4	0.2	0.2	0.2

## 12.2.4. 算术编码



### 算术编码的基本步骤

- (1) 根据已知条件和数据可知，信源各字符在区间 $[0, 1)$ 内的子区间间隔分别如下：

$$a = [0.0, 0.4)$$

$$b = [0.4, 0.6)$$

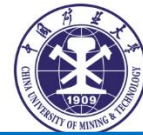
$$c = [0.6, 0.8)$$

$$d = [0.8, 1.0)$$

- (2) 计算中按如下公式产生新的子区间：

$$\begin{cases} Start_N = Start_B + Left_C \times L \\ End_N = Start_B + Right_C \times L \end{cases}$$

## 12.2.4. 算术编码



### 算术编码的基本步骤

- (3) 第1个被压缩的字符为“d”，其初始子区间为[0.8 , 1.0)
- (4) 第2个被压缩的字符为“a”，由于其前面的字符取值区间为[0.8 , 1.0)范围，因此，字符“a”应在前一字符区间间隔[0.8 , 1.0)的[0.0 , 0.4)子区间内，根据公式可得：

$$Start_N = 0.8 + 0.0 \times (1.0 - 0.8) = 0.8$$

$$End_N = 0.8 + 0.4 \times (1.0 - 0.8) = 0.88$$

$$\begin{cases} Start_N = Start_B + Left_C \times L \\ End_N = Start_B + Right_C \times L \end{cases}$$

### 算术编码的基本步骤

- (5) 第3个被压缩的字符为“c”，由于其前面的字符取值区间为 $[0.8, 0.88)$ 范围内，因此，字符“c”应在前一字符区间间隔 $[0.8, 0.88)$ 的 $[0.6, 0.8)$ 子区间内，根据可得：

$$Start_N = 0.8 + 0.6 \times (0.88 - 0.8) = 0.848$$

$$End_N = 0.8 + 0.8 \times (0.88 - 0.8) = 0.864$$

## 12.2.4. 算术编码



### 算术编码的基本步骤

- (6) 第4个被压缩的字符为“b”，由于其前面的字符取值区间为 $[0.848, 0.864)$ 范围内，因此，字符“b”应在前一字符区间间隔 $[0.848, 0.864)$ 的 $[0.4, 0.6)$ 子区间内，根据可得：

$$Start_N = 0.848 + 0.4 \times (0.864 - 0.848) = 0.8544$$

$$End_N = 0.848 + 0.6 \times (0.864 - 0.848) = 0.8576$$

## 12.2.4. 算术编码



### 算术编码的基本步骤

- (7) 第5个被压缩的字符为“a”，由于其前面的字符取值区间为 $[0.8544, 0.8)$ 范围内，因此，字符“a”应在前一字符区间间隔 $[0.8544, 0.8576)$ 的 $[0.0, 0.4)$ 子区间内，可得：

$$Start_N = 0.8544 + 0.0 \times (0.8576 - 0.8544) = 0.8544$$

$$End_N = 0.8544 + 0.4 \times (0.8576 - 0.8544) = 0.85568$$



## 12.2.4. 算术编码



### 算术编码的基本步骤

- 经过上述计算，字符集{dacba}被描述在实数 $[0.8544, 0.85568)$ 子区间内，即该区间内的任一实数值都唯一对应该符序列{dacba}；
- 因此，可以用 $[0.8544, 0.85568)$ 内的一个实数表示字符集{dacba}。

## 12.2.4. 算术编码



### 算术编码的基本步骤

- $[0.8544, 0.85568)$ 子区间的二进制表示形式为：  
 $[0.1101101010000110, 0.1101101100001101)$ ;
- 在该区间内的最短二进制代码为0.11011011，去掉小数点及其前的字符，从而得到该字符序列的算术编码为11011011。
- 算术编码可以通过硬件电路实现，在上述乘法运算，可以通过右移来实现，因此在算术编码算法中只有加法和移位运算。

## 12.2.4. 算术编码



### 算术编码的效能

- 根据上述运算结果，编码11011011惟一代表字符序列{dacba}，因此，平均码字长度为：

$$R = \frac{8}{5} = 1.6 \text{ bit/字符}$$

## 12.3. 变换编码



变换编码的基本原理是将空域中的图像信号，变换到另外一些正交空间中去，用变换系数来表示原始图像，并对变换系数进行编码。

一般来说在变换域里描述要比在空域简单，因为图像的相关性明显下降。尽管变换本身并不带来数据压缩，但变换图像的能量大部分只集中于少数几个变换系数上，采用量化和熵编码则可以有效地压缩图像的编码比特率。

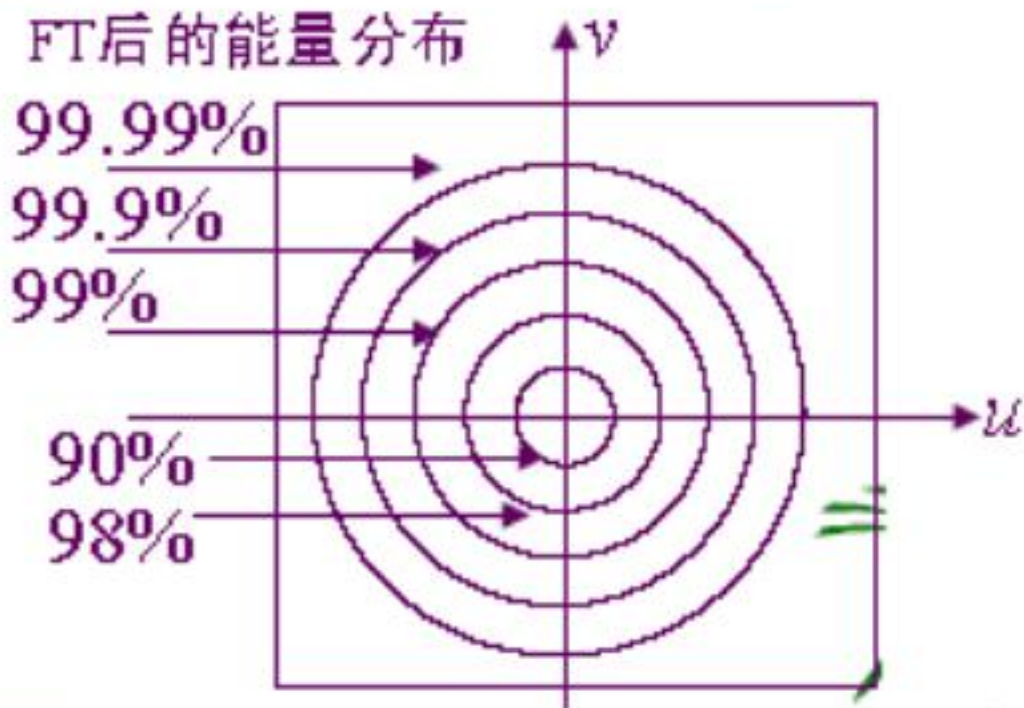
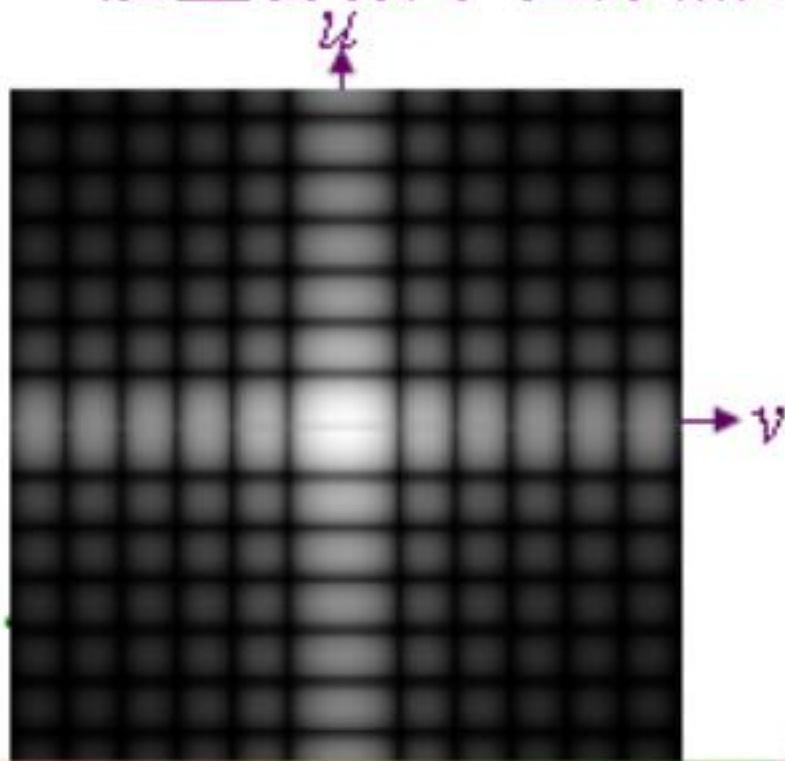
# 12.3.1 变换编码的特点



- 图像信息经过变换处理，相邻像素之间的相关性明显下降，有利于图像的编码压缩。
- 图像频谱中的变换系数，表示图像在不同空间频率上的相对幅度，而且某一空间频率所包含的信息来自整幅图像，频谱能量主要集中在低频部分，谱能量随频率的增加而迅速下降，
- 再次，变换编码受噪声干扰的影响较小。图像的变换编码，随着数字信号处理技术的发展，特别是快速变换的算法和大规模集成电路(LSI)的出现，使它具有实际应用的可能。

# 12.3.1 变换编码的特点

一般灰度图象，能量分布在整幅图上，FT后能量都集中在原点。另有  $F(0,0) = \bar{f}(x,y)$ 。



## 12.3.1 变换编码的特点

变换本身不能直接减少数据量，只有通过适当的编码，才能利用变换来压缩图像数据。

例，设一幅8x8的图像信息如下图

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 3 & 2 & 3 & 2 & 1 & 0 \\ 0 & 1 & 2 & 3 & 2 & 3 & 0 & 1 \\ 1 & 0 & 3 & 2 & 3 & 2 & 1 & 0 \\ 0 & 0 & 2 & 3 & 2 & 3 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 64 & 0 & -32 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -32 & 0 & 32 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 32 \end{bmatrix}$$

并对其进行二维Walsh变换

# 12.3.1 变换编码的特点



上面的例子说明，原始信号的能量分布是相当分散的，经过变换后却相当集中，而且主要集中在少数的频率谱上。对绝大部分区域来说，它的谱能量为零。为了达到数据的压缩，即选出能量集中的区域进行编码，而放弃不集中的区域。



# 12.3.1 变换编码的特点



## 变换编码的基本原理——举例

原始图像

相应的DCT系数

52	55	61	66	70	61	64	73	-415	-29	-62	25	55	-20	-1	3
63	59	66	90	109	85	69	72	7	-21	-62	9	11	-7	-6	6
62	59	68	113	144	104	66	73	-46	8	77	-25	-30	10	7	-5
63	58	71	122	154	106	70	69	-50	13	35	-15	-9	6	0	3
67	61	68	104	126	88	68	70	11	-8	-13	-2	-1	1	-4	1
79	65	60	70	77	68	58	75	-10	1	3	-3	-1	0	2	-1
85	71	64	59	55	61	65	83	-4	-1	2	-1	2	-3	1	-2
87	79	69	68	65	76	78	94	-1	-1	-1	-2	-1	-1	0	-1



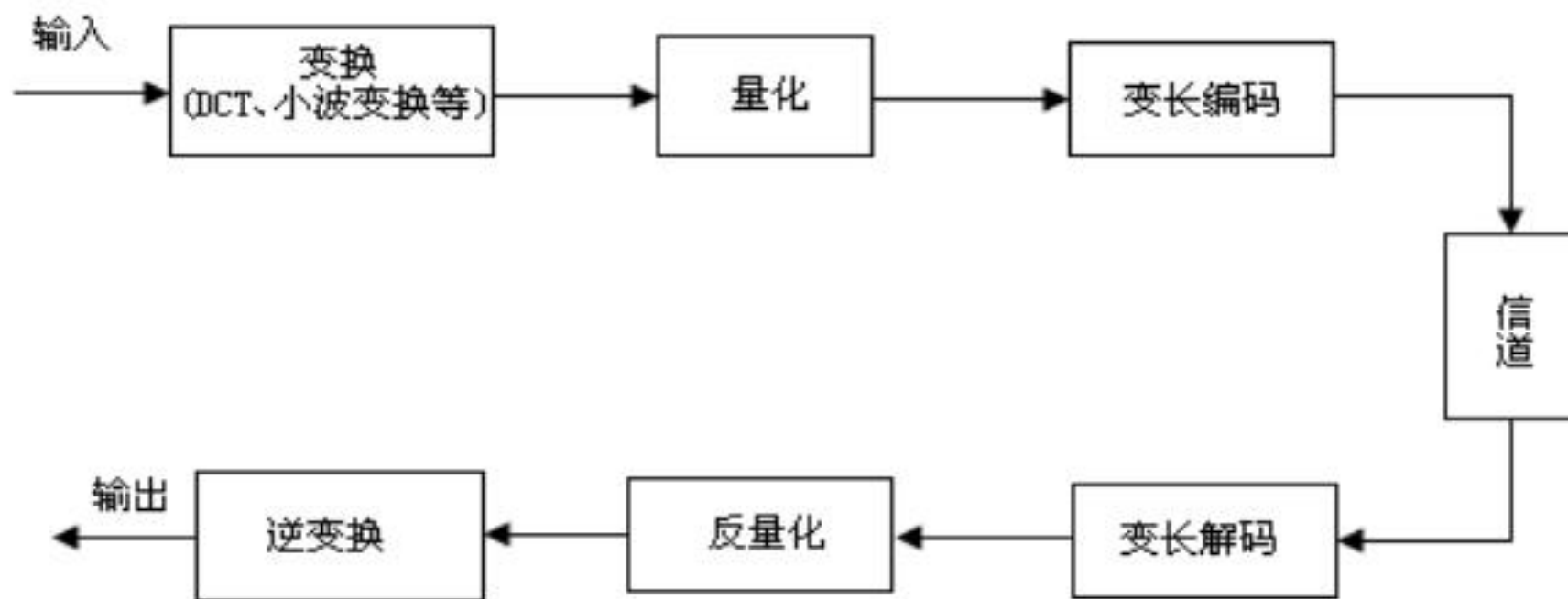
## 12.3.2 变换编码的基本步骤

- (1) 图像分块，用一个可逆线性变换（如傅立叶变换）把图像映射到变换系数集合。
- (2) 对该系数集合进行量化和编码。对于大多数图像，重要系数的数量是比较少，且图像失真较小。
- (3) 在接收端对接收到的码流进行解码，分离出各变换系数，且对舍去的系数用“0”来代替，然后求反变换，恢复各图像子块。

## 12.3.2 变换编码的基本步骤



### 编码、解码流程



# 12.3.2 变换编码的基本步骤



## ► 正交变换的在变换编码中的意义

图像数据正交变换后不改变信源的熵值，变换前后图像的信息量没有损失，完全可以通过对应的逆变换得到原来的图像数据。

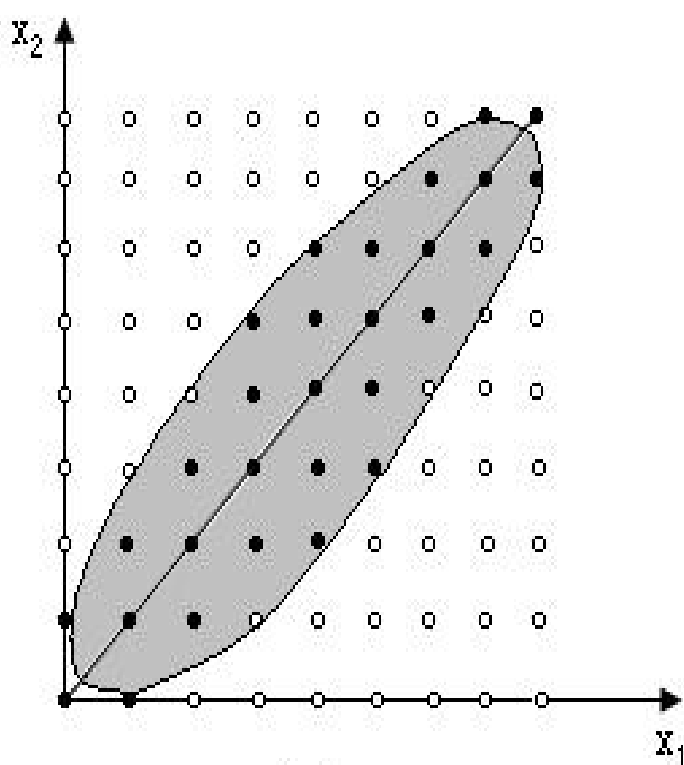
经过正交变换后，数据的分布规律发生了很大的改变，像素之间的相关性下降，变换系数向新坐标系中的少数坐标集中，一般集中于少数的直流或低频分量的坐标点。

变换编码将统计上高度相关的像素所构成的矩阵通过正交变换，变成统计上彼此较为独立、甚至达到完全独立的变换系数矩阵，以达到压缩数据的目的。

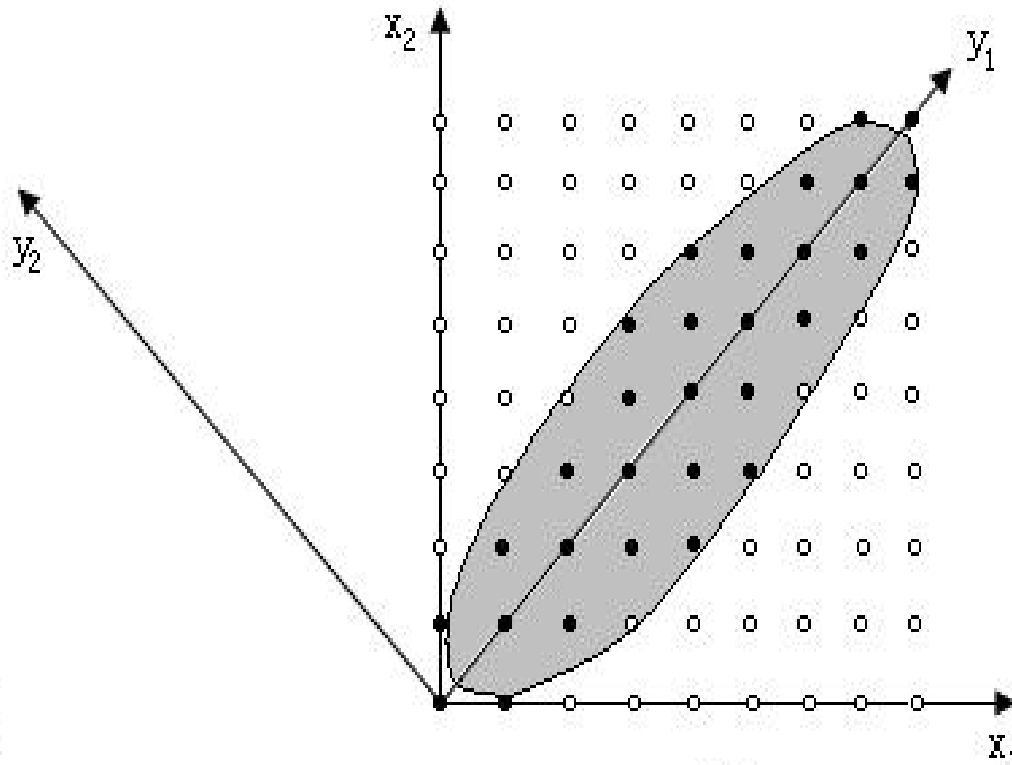
## 12.3.2 变换编码的基本步骤



### ► 正交变换的在变换编码中的几何意义



(a)



(b)

正交变换是保持图形形状和大小不变的几何变换，包含旋转，平移，轴对称及上述变换的复合

## 12.3.2 变换编码的基本步骤



图像变换应将整幅图像分成 $8 \times 8$ 或 $16 \times 16$ 的小块，然后分别进行变换。

子图像在变换中可能出现“边缘效应”，影响图像质量。

Fourier变换不仅在图像的增强，复原，重构，描述和图像序列分析中得到广泛应用，而且在图像变换编码中首先引起重视。

Fourier变换编码会出现“边缘效应”，若用余弦变换代替，由于其偶对称性，子图像在变换域中“边缘”效应几乎不存在。

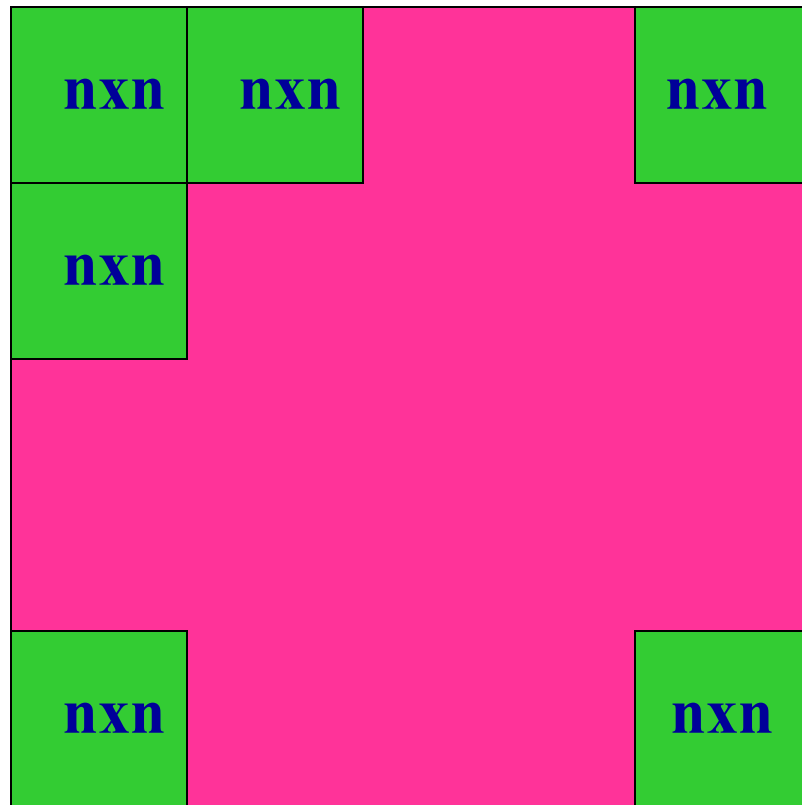
DCT是一种接近最佳的正交变换，在图像变换中具有重要的实用价值。JPEG标准也采用了它。

# 12.3.2 变换编码的基本步骤



-构造 $n \times n$ 的子图

$N \times N$

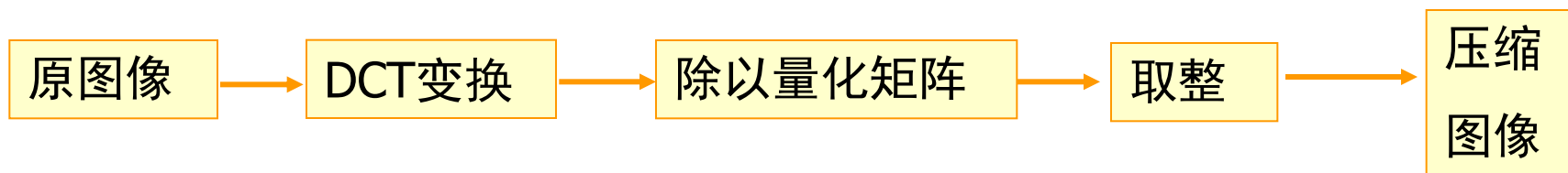


# 12.3.3 DCT变换编码的基本步骤

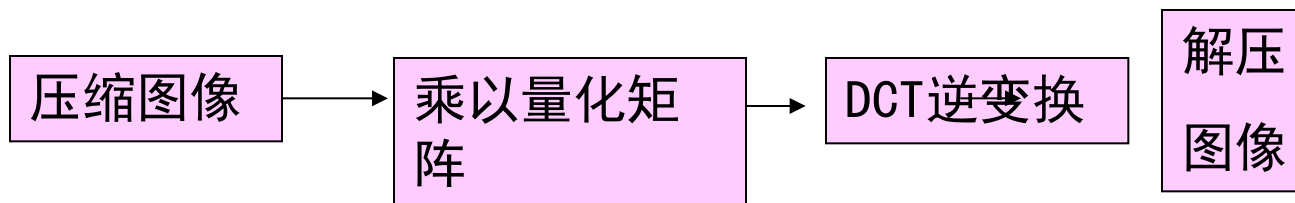


DCT变换编码方法：

1) 编码过程：



2) 解码过程：





# 12.3.3 DCT变换编码的基本步骤



例：

原图像为： $F = \begin{bmatrix} 59 & 60 & 58 & 57 \\ 61 & 59 & 59 & 57 \\ 62 & 59 & 60 & 58 \\ 59 & 61 & 60 & 56 \end{bmatrix}$

DCT变换

$D_1 = \begin{bmatrix} 120.5 & 119.5 & 118.5 & 114.0 \\ -0.27 & -0.65 & -1.58 & 0.38 \\ -2.50 & 1.50 & -0.50 & -1.00 \\ 0.65 & -0.27 & 0.11 & 0.92 \end{bmatrix}$

除以量化矩阵，取整

$C = \begin{bmatrix} 16 & 11 & 10 & 16 \\ 12 & 12 & 14 & 19 \\ 14 & 13 & 16 & 24 \\ 14 & 17 & 22 & 29 \end{bmatrix}$

$D = \begin{bmatrix} 8 & 11 & 12 & 7 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

# 12.3.3 DCT变换编码的基本步骤



## DCT变换编码



原图



解压图

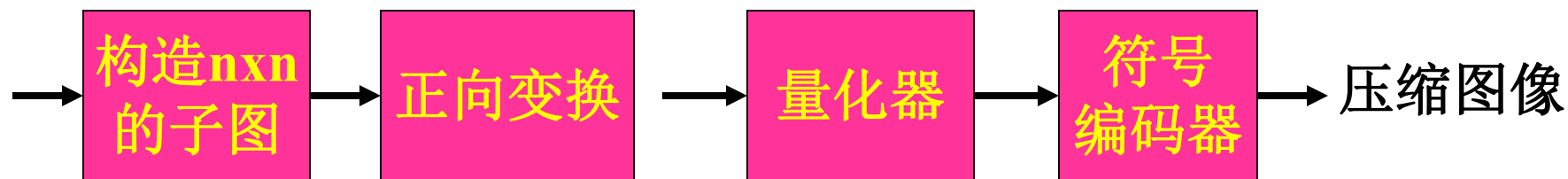
# 12.3.4 变换编码的核心问题



## 实现变换压缩算法的核心问题

- 变换方式的选择
- 子图尺寸的选择

输入  
图像  
 $N \times N$



# 12.3.4 变换编码的核心问题



## 主要问题一：变换方式的选择

### 1、可以选择的变换

- 1) K-L变换(KLT)
- 2) 离散傅立叶变换 (DFT)
- 3) 离散余弦变换 (DCT)
- 4) Walsh-Hadamard变换 (沃尔什-阿达玛WHT)
- 5) 小波变换

# 12.3.4 变换编码的核心问题



## 2、对变换的评价

➤按信息封装能力排序（由强到弱）：

$$K-L > DCT > DFT > WHT > HRT$$

➤若输入是广义平稳序列，则存在一种最佳的正交变换——卡洛变换。所谓最佳：

1.变换系数互不相关；

2.数值较大的方差出现在少数系数中，即能量高度集中。这样，可在允许的总的均方误差一定的条件下，将数据减到最少。

➤K-L变换的基图像是数据依赖的，所以一般没有快速算法，因此只宜于作理论分析和试验用。

➤DFT的块效应严重。

# 12.3.4 变换编码的核心问题



常用的是DCT，已被国际标准采纳，作成芯片。  
其优点：

1) 基本没有块效应。

2) 信息封装能力强，把最多的信息封装在最少的系数中。

# 12.3.4 变换编码的核心问题



## 主要问题二：子图尺寸的选择

子图尺寸的选择有两个原则：

- 1) 如果 $n$ 是子图的维数，为便于降低计算复杂度， $n$ 应该是2的整数次方。。
- 2)  $n$ 一般选为 $8 \times 8$ 或 $16 \times 16$ 。由实践得到：随着 $n$ 的增加，块效应相应减少。

# 12.3.4 变换编码的核心问题



一般来说，图像变换的编码压缩按下列步骤进行：

(1) 确定图像矩阵的尺寸

(2) 确定变换矩阵

(3) 计算变换域

(4) 保留较大的那些系数，构成压缩后的新矩阵

(5) 按新矩阵传输那些系数不为零的数值，并在接收

端用反变换求得原始图像的值。



# 12.3.4 变换编码的核心问题



- ✓ 在确定图像矩阵的尺寸后，选哪种变换矩阵，都必须从存储量的大小，计算速度，变换图像的质量，硬件实施等因素来综合考虑。
- ✓ 由于图像和客观景物的千变万化，为了达到较高的压缩比，还可以把变换编码同其它形式的编码（如预测编码）结合起来的编码，称为混合编码。

# 12.4. 静止图像压缩编码标准—JPEG



- 图像标准的制定：

ISO(International Standard Organization)和CCITT  
(国际电报电话咨询委员会) 联合制定

- 标准的类型：

— 连续图像压缩标准：

静止帧黑白、彩色压缩：(1)面向静止的单幅图像  
— JPEG)

连续帧黑白、彩色压缩：(2)面向连续的视频影像  
— MPEG)

# 12.4. 静止图像压缩编码标准—JPEG



- JPEG标准简述
- JPEG压缩流程
- JPEG压缩算法的实现
- JPEG压缩举例

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG标准简述

由ISO/IEC与CCITT联合发起的联合图像专家组于20世纪90年代初制定了静止图像(包括8bit/像素的灰度图像与24bit/像素的彩色图像)的编码标准。

**JPEG标准**在较低的计算复杂度下，能提供较高的压缩比与保真度。在视觉效果不受到严重损失的前提下，算法可以达到15到20的压缩比。如果在图像质量上稍微牺牲一点的话，可以达到40:1或更高的压缩比。

# 12.4. 静止图像压缩编码标准—JPEG



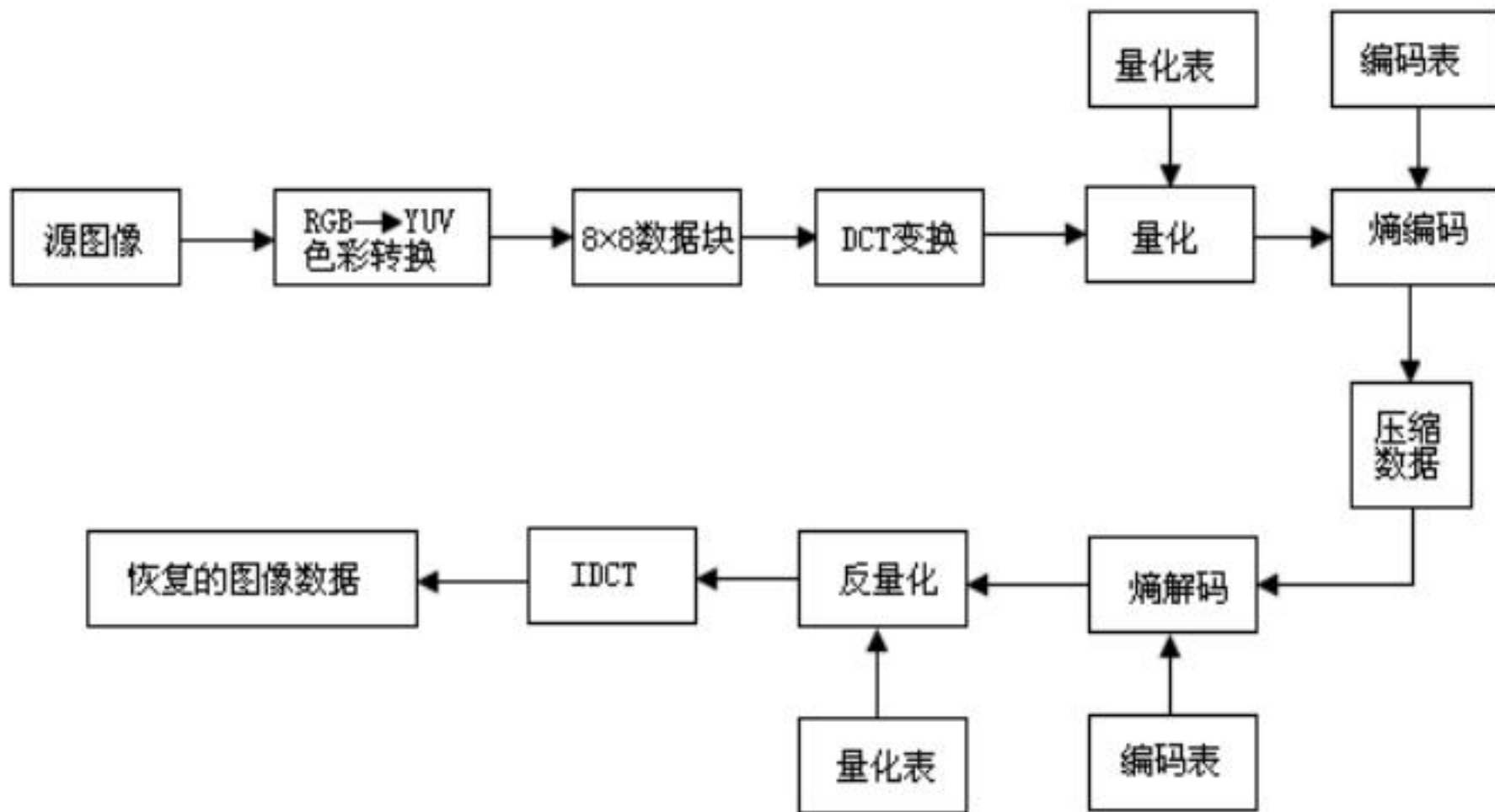
JPEG定义了一个基本系统，一个符合JPEG标准的编解码器至少要满足基本系统的技术指标。JPEG基本系统其核心属于变换编码。JPG编码时，对原始图像的每一个分量首先分割成互不重叠的 $8 \times 8$ 像素块，然后对每个像素块的编码过程可分为二维DCT变换。

根据图像信号的特点，对图像块进行二维DCT变换可以消除像素间的相关性。自然图像的像素块经DCT变换后，图像信号的能量主要集中在块的左上角，即图像的低频成分中。DCT变换后得到的系数矩阵中包括左上角的一个直流(DC)系数与63个交流(AC)系数，从左到右，水平频率增高，从上到下竖直频率增高。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程



# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 1. 颜色空间转换

人眼对亮度更敏感，提取亮度特征，将RGB转换为 $YCbCr$ 模型，编码时对亮度采用特殊编码：

$$\left\{ \begin{array}{l} Y = 0.299R + 0.5870G + 0.1140B \\ C_b = -0.1787R - 0.3313G + 0.5000B + 128 \\ C_r = 0.5000R - 0.4187G - 0.0813B + 128 \end{array} \right.$$

颜色解码：

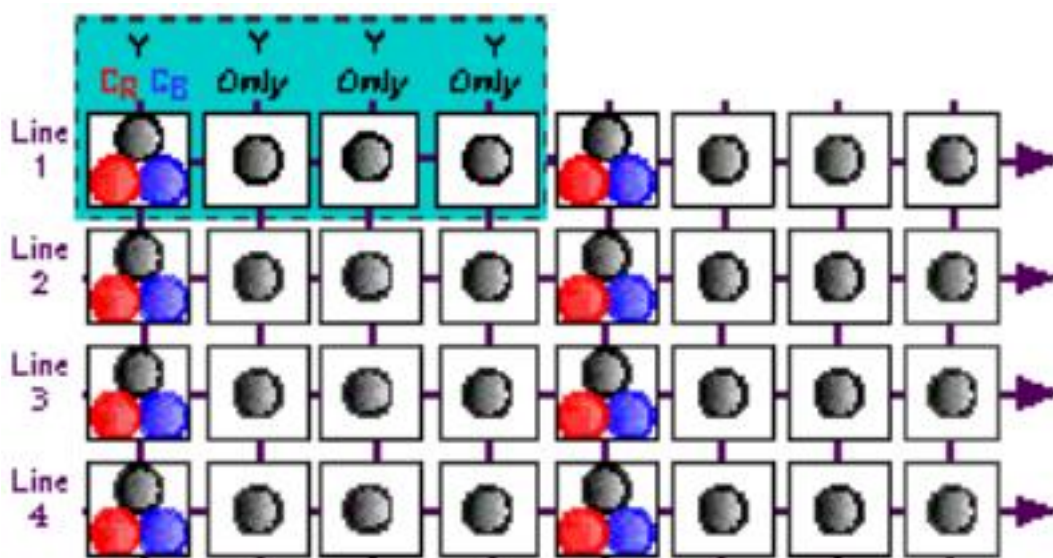
$$\left\{ \begin{array}{l} R = Y + 1.40200(C_r - 128) \\ G = Y - 0.34414(C_b - 128) - 0.71414(C_r - 128) \\ B = Y + 1.77200(C_b - 128) \end{array} \right.$$

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 1. 颜色空间采样



**4:1:1** 4:1:1则是指在水平方向上对色度信号进行4:1取样

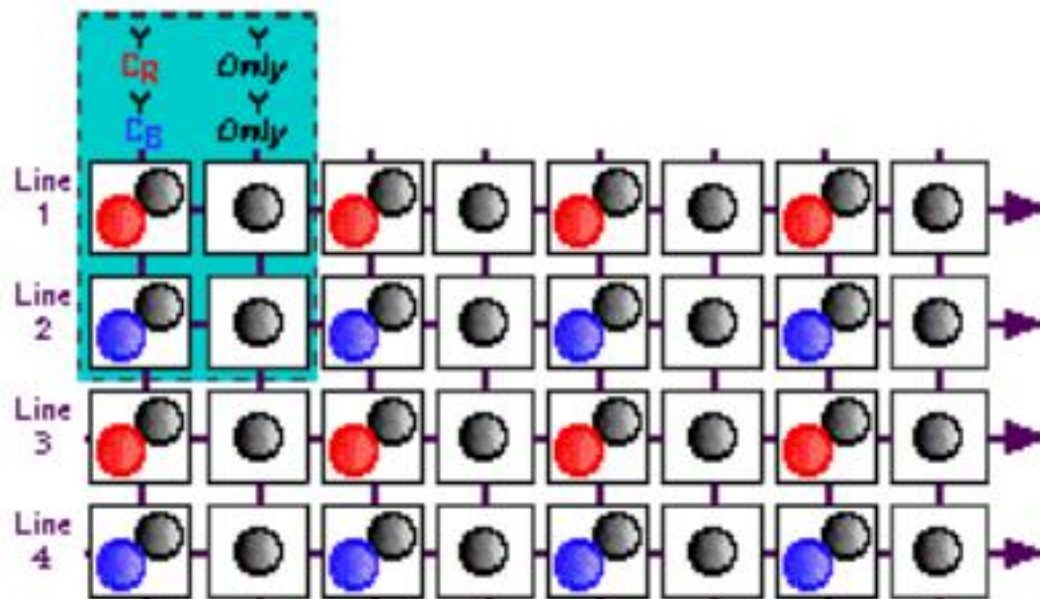


# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 1. 颜色空间采样



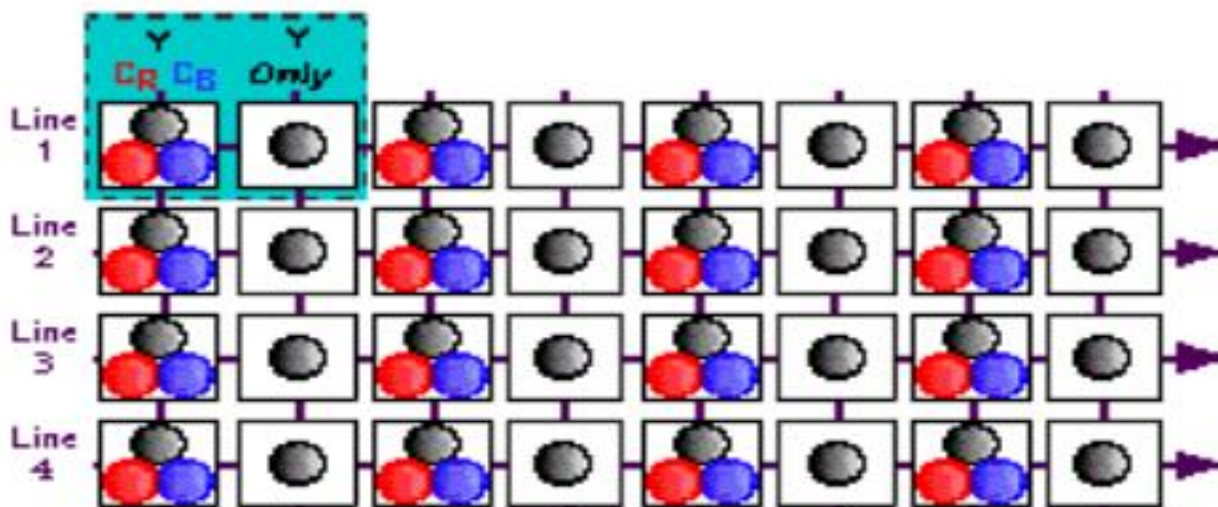
**4:2:0** 4:2:0是指对每行扫描线来说，只有一种色度分量以2:1的取样率存储，它并不意味着只有Y/Cb而无Cr分量

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 1. 颜色空间采样



4:2:2是指将色度信号的取样率选为亮度信号的1/2，其每个色差通道的取样率是亮度通道的1/2，用于YUV/YCbCr等色彩空间中，这种方式对图像质量的要求较高，广泛应用于电视演播室等领域。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 2. 图像分块

构造子图像：

子图像尺寸：8 x 8

### 3. DCT变换

- 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的值域，通过减去128，转换为值域在-128~127之间的值
- 目的：使像素的绝对值出现3位10进制的概率大大减少

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

用8x8的JPEG基线标准，压缩并重构下列子图

52	55	61	66	70	61	64	73
63	59	66	90	109	85	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	68	65	76	78	94

# 12.4. 静止图像压缩编码标准—JPEG



0偏置转换后

-76	-73	-67	-62	-58	-67	-64	-55
-65	-69	-62	-38	-19	-43	-59	-56
-66	-69	-60	-15	16	-24	-62	-55
-65	-70	-57	-6	26	-22	-58	-59
-61	-67	-60	-24	-2	-40	-60	-58
-49	-63	-68	-58	-51	-65	-70	-53
-43	-57	-64	-69	-73	-67	-63	-45
-41	-49	-59	-60	-63	-52	-50	-34

- 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），其余的63个系数称为交流系数（AC系数）。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

正向DCT变换 ( $N = 8$ ) 后变成

-415	-29	-62	25	55	-20	-1	3
7	-21	-62	9	11	-7	-6	6
-46	8	77	-25	-30	10	7	-5
-50	13	35	-15	-9	6	0	3
11	-8	-13	-2	-1	1	-4	1
-10	1	3	-3	-1	0	2	-1
-4	-1	2	-1	2	-3	1	-2
-1	-1	-1	-2	-1	-1	0	-1

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 4. 系数量化

1) 正向量化:

$$Sq_{uv} = \text{round}(S_{uv}/Q_{uv})$$

其中:  $S_{uv}$  是DCT系数,  $Q_{uv}$  量化模板系数

2) 逆向量化:

$$R_{uv} = Sq_{uv} Q_{uv}$$

$$\begin{aligned} \text{例: } Sq(0,0) &= \text{round}[-415/16] \\ &= \text{round}[-25.9] = -26 \end{aligned}$$

$$R_{uv}(0,0) = -26 * 16 = -416$$

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

对于亮度和颜色使用不同的量化阈值模板，并取整亮度的量化模板系数

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



# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

颜色的量化模板系数

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

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

量化变换后的数组，比例化并消去系数

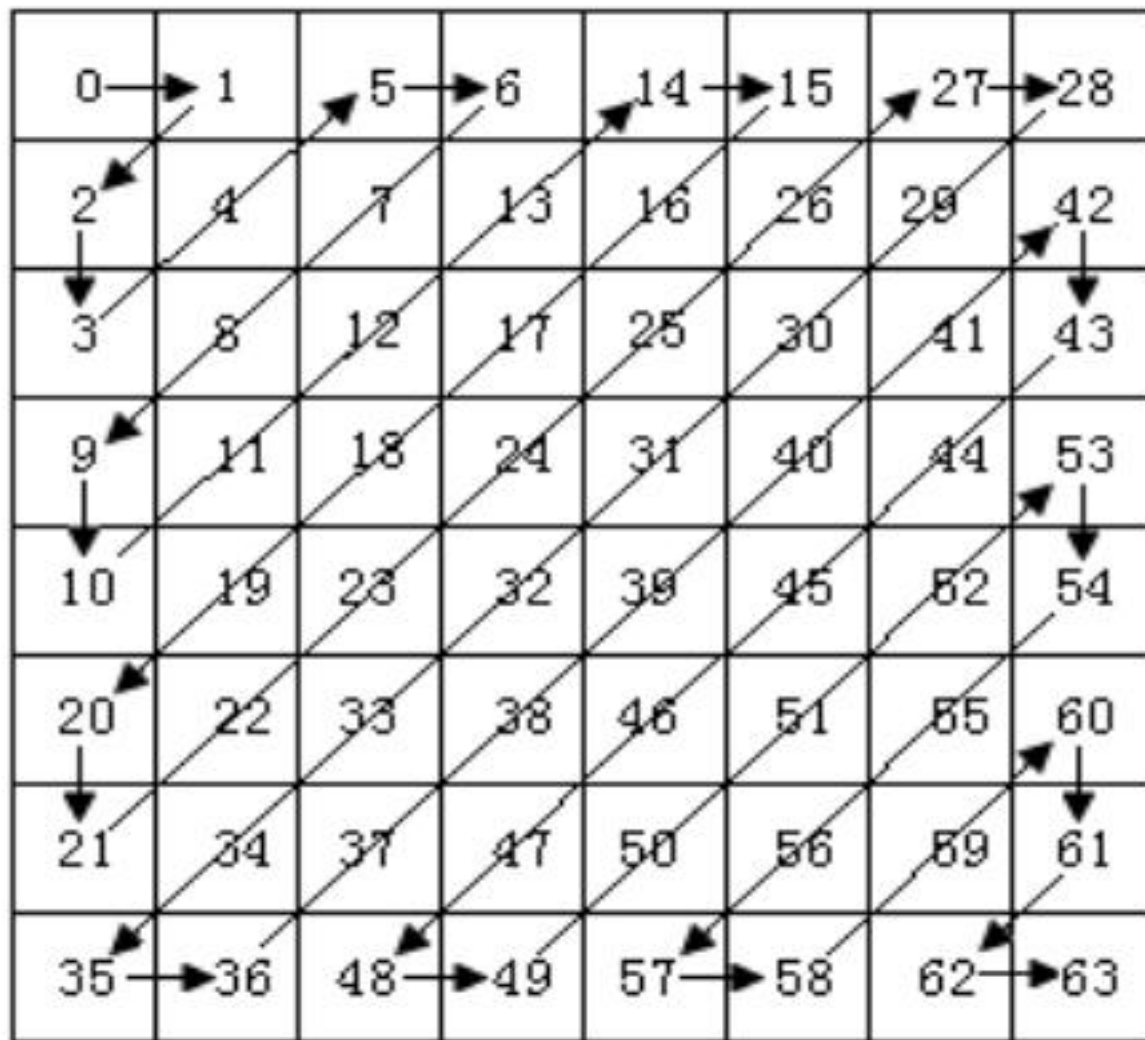
-26	-3	-6	2	2	0	0	0
1	-2	-4	0	0	0	0	0
-3	1	5	-1	-1	0	0	0
-4	1	2	-1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

5. Z形扫描：将量化后的系数按Z字形扫描



# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 6. DC编码

- DCT变换的64个系数经量化后，其中 $F(0,0)$ 为直流系数DC，其余的63个为交流系数AC，DC的大小反映了一个 $8 \times 8$ 数据块的平均亮度。
- $8 \times 8$ 相邻子块之间DC系数有很强的相关性，所以JPEG对DC系数采用差分编码。
- 以前一数据块的同一分量的DC系数作为当前块的预测值，再对当前块的实际值与预测值的差值进行哈夫曼编码或算术编码。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

- 若DC系数的动态范围为-1024~1023，则差值的动态范围可达-2047~+2047;
- 由于差分值范围太大，JPEG没有采用对每一个差分值赋予一个码字，而是对码表进行了简化，采用“前缀码（SSSS）+尾码”表示;
- 前缀码指明了尾码的有效位数B，可以根据差分值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DC系数的差分值（D）大于等于0，则尾码的码字为差分值的原码；否则，取差分值的B位反码。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### DC系数编码步骤

- (1) 对差分值进行判断

如果差分值大于0，将差分值转换为二进制表示，并获得差值所占的位数。如差值为5时，其二进制表示为101，差分值得位数为3；

如果差分值小于0，则取绝对值后转换为二进制码表示，再获得该编码值的反码。如差值为-5时，最后得反码为010，差分值的位数为3。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

- (2) 根据“前缀码（SSSS）+尾码”的表示方法，以差分值所占的位数为索引值查表获得该差分值的编码。

表为亮度系数（DC）的哈夫曼编码表。

查表可获得差分为5时编码为100，结合前缀码（SSSS）和尾码，最终的编码值为100101；差值为-5时的最终编码为100010。

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG压缩流程（共七步）

### 7. AC编码

JPEG基本系统对63个AC系数采用行程编码，采用Z字形扫描是为了增加0的行程长度，从而更有利于压缩数据。

AC系数编码可采用哈夫曼编码或算术编码。



# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG图像压缩算法

- ❑ JPEG 是有损压缩算法
- ❑ JPEG 核心是离散余弦变换(DCT)

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG图像压缩算法

DCT变换的输入是8位的像素值（0~255，JPEG实现时将其减去128，范围变成-128~127），量化即通过整除运算减少输出值的存储位数。

使用量化矩阵（Quantization Matrix）来实现量化。

量化公式为：

$$\text{量化后的值}(i, j) = \text{ROUND}(\text{DCT}(i, j) / \text{量子}(i, j))$$

逆量化公式为：

$$\text{DCT}(i, j) = \text{量化后的值}(i, j) * \text{量子}(i, j)$$

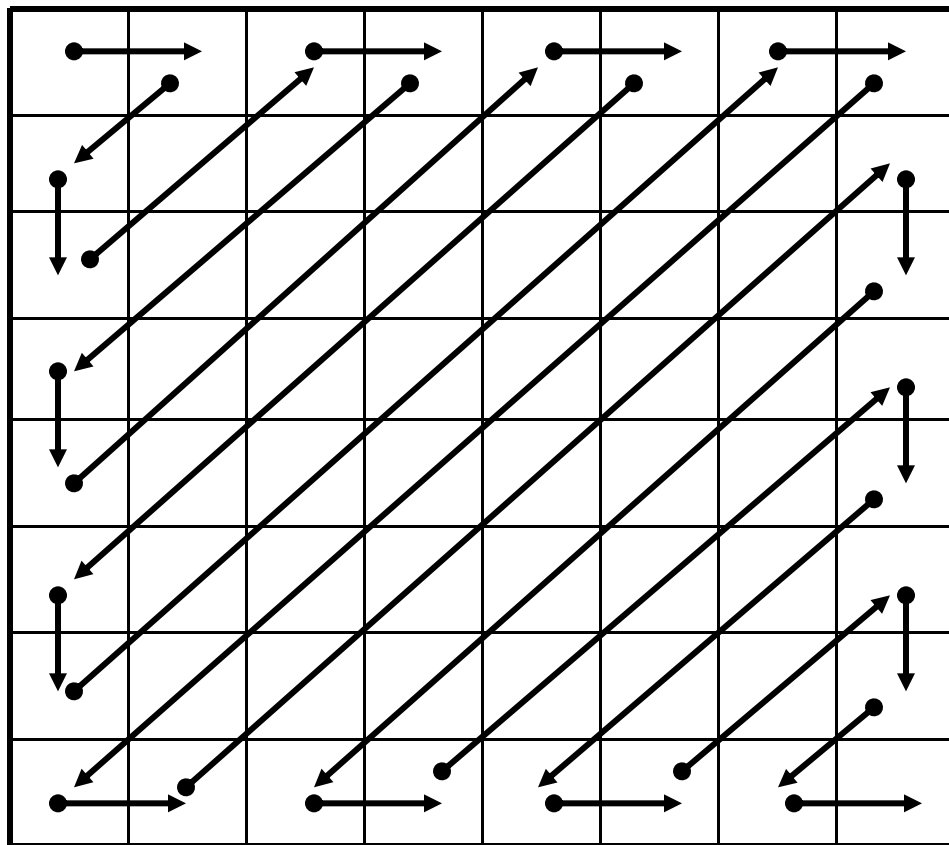
量化是JPEG算法中损失图像精度的根源，也是产生压缩效果的源泉

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG图像压缩算法

### Zig-Zag编码



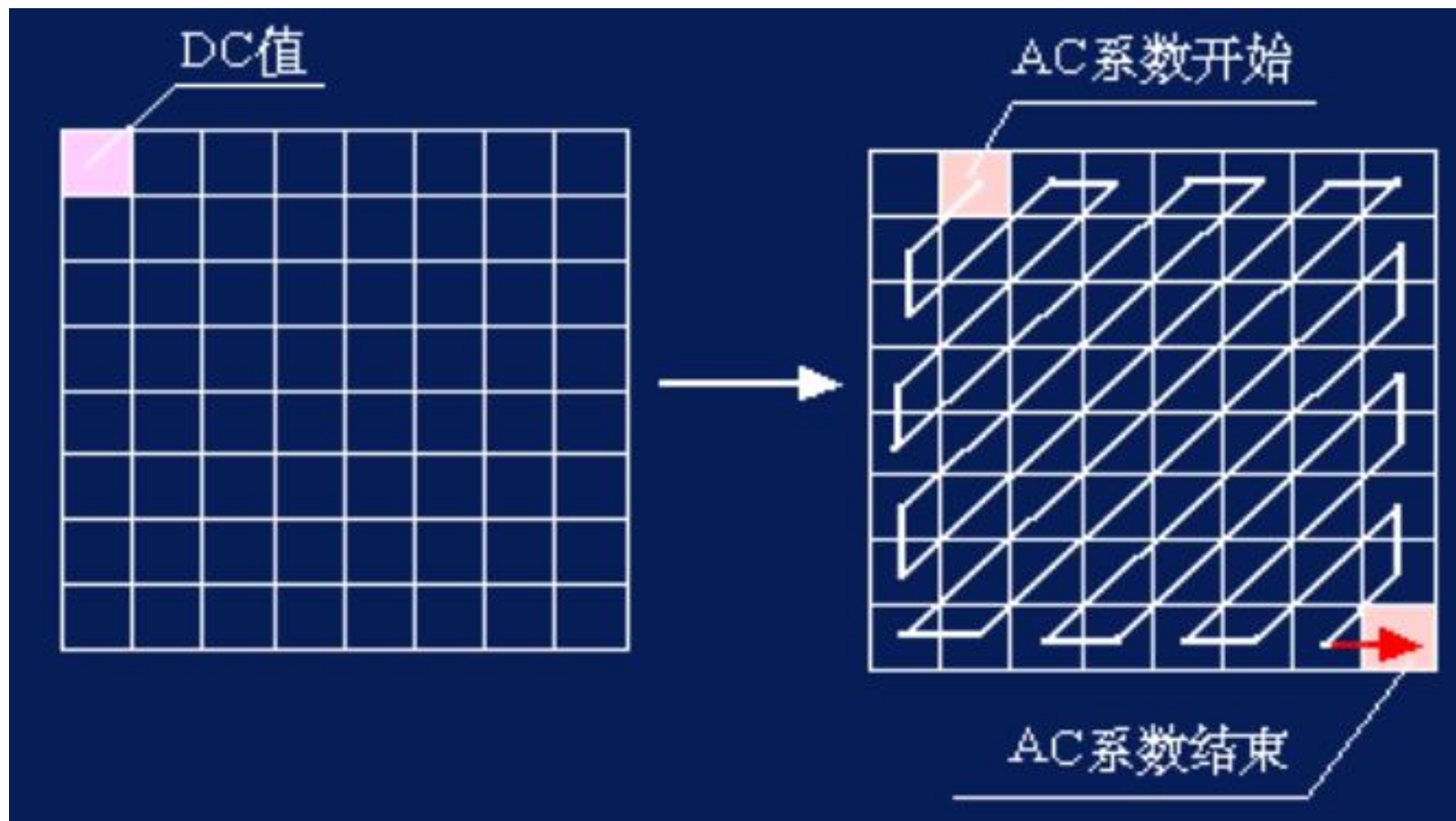
$(0, 0) \rightarrow (0, 1) \rightarrow (1, 0) \rightarrow (2, 0) \rightarrow \dots$

- ❑ 将量子化的矩阵按 Zig-Zag 顺序排列
- ❑ 将原始数列转换为差值数列
- ❑ 对差值数列进行编码，可以使用 Huffman 编码、算术编码或熵编码等方法

# 12.4. 静止图像压缩编码标准—JPEG



## ➤ JPEG图像压缩算法





# 一个真实的编码和解码过程

139	144	149	153	155	155	155	155	235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3	16	11	10	16	24	40	51	61
144	151	153	156	159	156	156	156	-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2	12	12	14	19	26	58	60	55
150	155	160	163	158	156	156	156	-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1	14	13	16	24	40	57	69	56
159	161	162	160	160	159	159	159	-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3	14	17	22	29	51	87	80	62
159	160	161	162	162	155	155	155	-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3	18	22	37	56	68	109	103	77
161	161	161	161	160	157	157	157	1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0	24	35	55	64	81	104	113	92
162	162	161	163	162	157	157	157	-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8	49	64	78	87	103	121	120	101
162	162	161	161	163	158	158	158	-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4	72	92	95	98	112	100	103	99

(a) 8X8的原始图像块

(b) DCT离散余弦变换

(c) 量子表

15	0	-1	0	0	0	0	0	240	0	-10	0	0	0	0	0	144	146	149	152	154	156	156	156
-2	-1	0	0	0	0	0	0	-24	-12	0	0	0	0	0	0	148	150	152	154	156	156	156	156
-1	-1	0	0	0	0	0	0	-14	-13	0	0	0	0	0	0	155	156	157	158	158	157	156	155
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160	161	161	162	161	159	157	155
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	163	163	164	163	162	160	158	156
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	163	164	164	164	162	160	158	157
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160	161	162	162	162	161	159	158
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	158	159	161	161	162	161	159	158

(d) 量子化

(e) 反向量子化

(f) 反向DCT变换

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

➤ 连续帧图像的定义

➤ 连续帧图像压缩的基本思想

➤ 帧间运动补偿预测编码技术

➤ MPEG1/2/4标准

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### ➤ 连续帧图像的定义

- 由多幅尺寸相同的静止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。





# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### 连续帧图像压缩的基本思想

基于如下基本假设：

- 在各连续帧之间存在简单的相关性平移运动。
- 一个特定画面上的像素量值：
  - 1) 可以根据同帧附近像素来加以预测，被称为：  
帧内编码技术
  - 2) 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



# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

- ◆ 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目的。
- ◆ 将连续帧图像序列，分为参考帧和预测帧，参考帧用静止图像压缩方法进行压缩，预测帧对帧差图像进行压缩。由于帧差图像的数据量大大小于参考帧的数据量，从而可以达到很高的压缩比。

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (1) MPEG1标准

MPEG-1标准是由国际标准化组织ISO与国际电工委员会IEC共同制定的，标准的编号是ISO/IEC/11172，标准的题目是“码流速率约为1.5Mb/s时，用于数字存储媒体的活动图像及其伴音的编码”。

**应用范围：**视频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

霍夫曼编码

算术编码

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (2) MPEG2标准

为了在高速网络的环境下（如ATM）提供高比特率、高质量的视频应用，ISO下属的MPEG委员会在1994年又发布了MPEG2, 它是一种高质量视频的编码标准, 也称为广播电视的视频编码标准。ISO和IEC在制定MPEG-1标准时, 已经开始考虑MPFG2和MPEG3.

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (2) MPEG2标准

MPEG2是作为一个传输标准制定的,同时也是MPEG-1的兼容扩展,它能广泛应用于各种速率(2~20Mb/s)和各种分辨率.同MPEG1相比,在完全引用了MPEG1基于DCT变换和运动补偿帧间双向预测的基本结构的基础上,作了许多扩展.人们对MPEG-2标准在质量与应用方面提出了许多要求,希望能包括视频通信的各个领域.如:

# 12.5. 运动图像压缩编码标准—MPEG



## (2) MPEG2标准

- (1) MPEG-2的视频图像格式达到 $720 \times 480$ ，码率达到10Mb/s。
- (2) 支持多点电视会议 (Multipoint Video Conferencing)
- (3) 支持工作站视窗显示 (Window Display on Workstations)。
- (4) 支持基于ATM的视频通信 (Video Communications on ATM Networks)。
- (5) 支持嵌入式标准电视的HDTV (HDTV with Embedded Standard TV)。

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (2) MPEG2标准

- 应用范围：数字电视、高质量视频、有线电视、视频编辑、视频存储
- 主要编码技术：
  - DCT变换
  - 前向、双向运动补偿预测
  - Zig-zag排序
  - 霍夫曼编码、算术编码

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (3) MPEG4标准

MPEG-4是ISO的MPEG委员会制定的关于低于32kb/s作传输速率的适用于可视电话的运动图像编码标准，MPEG-4旨在建立一种能被窄带网络、宽带网络、无线网络、多媒体数据库等各种存储传输设备所广泛支持的通用音频、视频数据格式，最终的MPEG-4标准在1998年底完成，第一版和第二版分别于1999年和2000年颁布。

# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (3) MPEG4标准

MPEG-4标准较之MPEG-1、MPEG-2的一个较大的改进就是提出了任意形状视频目标的编码方案。在这新的编码技术中，较大的扩展了MPEG-1、MPEG-2中所采用的传统方块变换编码方案。对任意形状的目标编码，包括对目标形状的轮廓编码以及对目标本身的编码。



# 12.5. 运动图像压缩编码标准—MPEG



## ➤ MPEG图像压缩算法

### (3) MPEG4标准

- 应用范围：互联网、交互视频、移动通信
- 主要编码技术：
  - DCT变换、小波变换
  - 前向、双向运动补偿预测
  - Zig-zag排序
  - 脸部动画、背影编码
  - 霍夫曼编码、算术编码

# 本章重点



- 常见的数据冗余;
- 数据压缩的评价指标;
- 哈夫曼编码的计算;
- 香农-范诺编码;
- 行程编码;
- 算术编码;
- 变换编码的概念;
- JPEG和MPEG.

■ 12.1 试对图像  $f =$ 

0	0	0	0	1	1	1	2
0	0	0	0	1	1	2	3
1	1	1	1	1	2	2	3
2	2	2	2	2	2	2	3
3	3	3	3	3	3	3	3
3	3	3	3	3	4	4	5
4	4	4	4	4	4	4	5
6	6	6	6	7	7	5	5

 进行 Huffman

编码，并求其平均码长。（要求写出 Huffman 编码过程，对图像中不同字符赋予的码字）

■ 12.2 已知信源符号集  $X = \{a_1, a_2\} = \{0, 1\}$ ，符号产生概率为  $p(a_1) = 1/4$ ， $p(a_2) = 3/4$ ，试对序列 1011 进行算术编码。