多媒体数据压缩技术（一）

内容提纲

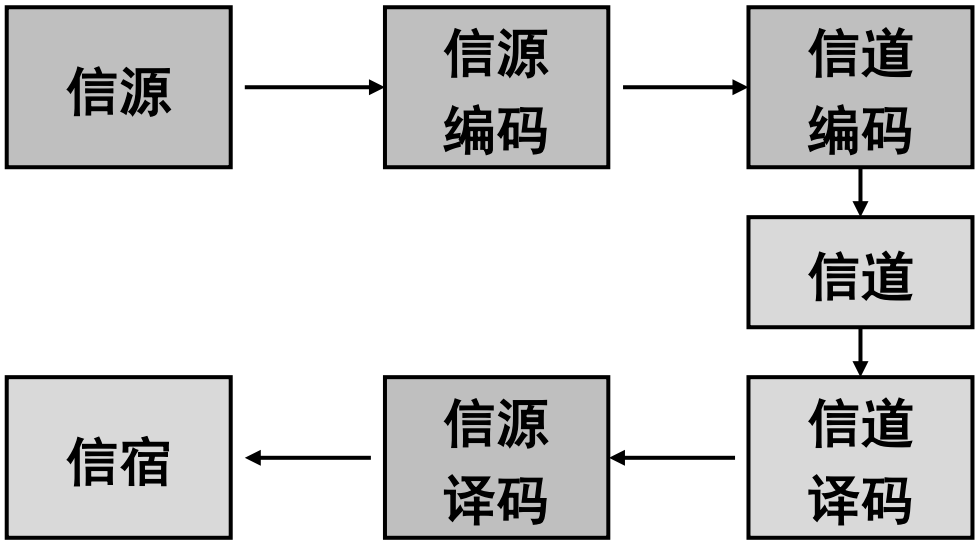
- 数据压缩的重要性
- 数据压缩方法分类
- 数据压缩基础

内容提纲

- **数据压缩的重要性**
- 数据压缩方法分类
- 数据压缩基础

多媒体数据压缩的重要性

- 多媒体信息的数据海量性，与当前硬件技术所能提供的计算机存储资源和网络带宽之间有很大差距
- 目前，数字化的媒体信息数据以压缩形式存储和传输仍将是唯一选择



多媒体数据压缩的重要性

- 几个未经压缩的数字化信息的例子
  - 一页在B5（约180mm×255mm）纸上的文件，若以中等分辨率（300 dpi）扫描仪进行数字化，其数据量约6.61MB/页。一片650MB的CD-ROM，可存98页
  - 双通道立体声激光唱盘（CD-A），采样频率44.1KHz，采样精度16位/样本，一个650MB的CD-ROM，可存约1小时的音乐

多媒体数据压缩的重要性

- PAL制的数字电视图像，4:4:4采样格式
  - 每帧数据量 720×576×3 = 1.24 MB
  - 每秒数据量 1.24×25 = 31.3 MB/s
  - 一片CD-ROM可存帧数

    650 / 1.24 = 0.524 千帧/片
  - 一片CD-ROM可存节目时间

    650 / 31.1 = 20.9 秒/片

多媒体数据压缩的重要性

- 遥感图像：陆地卫星（LandSat-3），其水平、垂直分辨率分别为2340和3240，四波段、采样精度7位
  - 一幅图像的数据量为：

$$2340 \times 3240 \times 7 \times 4 = 212 \text{ Mb}$$

  - 按每天30幅计算，每天的数据量为

$$212 \times 30 = 6.36 \text{ Gb}$$

  - 每年的数据量高达 2300 Gb

多媒体数据压缩的重要性

- 庞大的数据量是多媒体技术发展中的一个非常棘手的**瓶颈问题**
- 通过数据压缩手段把信息数据量压下来，以压缩形式存储和传输
  - **紧缩节约了存储空间**
  - **提高了通信干线的传输效率**
  - **使计算机实时处理音频、视频信息，以保证播放出高质量的视频、音频节目成为可能**

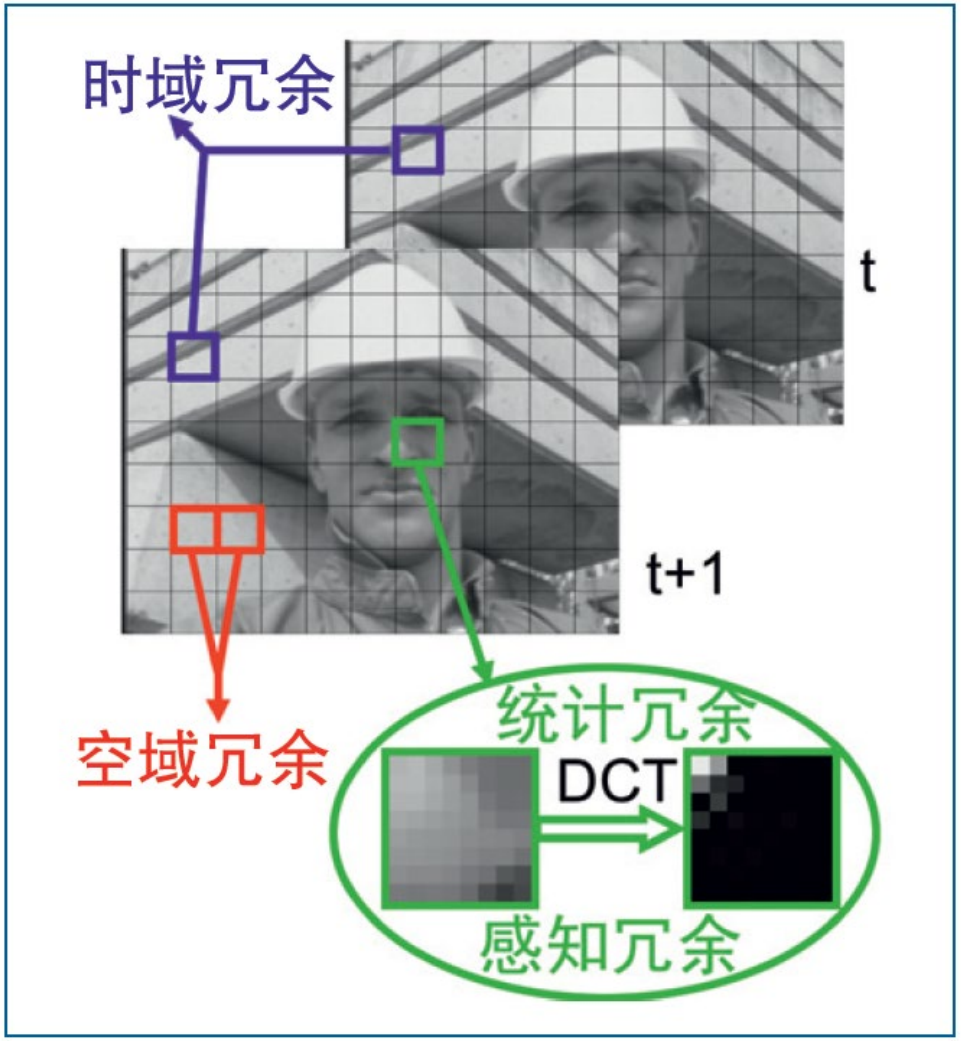
多媒体数据压缩的可能性

- 人们研究发现，图像数据表示中存在着大量的**冗余（redundancy）**
  - 通过去除那些冗余数据可以使原始图像数据极大地减少。**图像数据压缩技术就是研究如何利用图像数据的冗余性来减少图像数据量的方法**
  - 随着对人类视觉系统和图像模型的进一步研究，人们可能会发现更多的冗余性，使图像数据压缩编码的可能性越来越大，从而推动图像压缩技术的进一步发展

多媒体数据压缩的可能性

空域冗余：图像在空间上存在很大相关性，如：相邻像素值重复或非常接近。

时域冗余：视频图像在时间上存在很大相关性。



多媒体数据压缩的可能性

- **视觉冗余（感知冗余）**：人眼感受不到图像中的一些复杂细节信息，丢弃这些信息不会影响视觉感受
  - 视觉系统对图像的亮度和色彩度的敏感性相差很大（YUV、YIQ颜色空间）
  - 人眼对高频信息不太敏感，对纹理丰富的内容区域进行较粗的量化，以提升编码效率

多媒体数据压缩的可能性

Lena.bmp (原图)

Lena.jpg (压缩率18.4)

多媒体数据压缩方法的应用

- 多媒体数据压缩算法及标准广泛应用于：
  - 多媒体计算机、多媒体数据库、常规的电视数字化、高清晰度电视(HDTV)以及交互式电视(ITV)系统中

- 目前正在开展的应用项目有：
  - 可视电话、视频会议、多媒体电子邮件、电子出版物、家庭卫星广播业务(BSS)、地面数字电视广播(DTTB)、电子影院(EC)、电子新闻采集系统(ENG)、个人通信(IPC)、网络数据库(NDB)、家庭电视剧场(HTT)、遥控监视(RVS)以及电视点播系统(VOD)等

内容提纲

- 数据压缩的重要性
- **数据压缩方法分类**
- 数据压缩基础

多媒体数据压缩方法的分类

- 质量有无损失
  - 有损失编码、无损失编码
- 作用域
  - 空间方法、变换方法、混合方法
- 是否自适应
  - 自适应性编码、非自适应性编码

多媒体数据压缩方法的分类

- **脉冲编码调制（PCM）**
  - 它实际上是连续模拟信号的数字采样表示。PCM编码器和解码器位于一个图像编码系统的起点和终点，它们实际上分别是A/D和D/A转换器

- 其它编码方法通常都是在多媒体数据模拟信号经过PCM编码后再进行的

多媒体数据压缩方法的分类

- **预测编码**
  - 编码器记录与传输的不是样本的真实值，而是它与预测值的差
  - 由于空间的相关性，真实值和预测值的差值变化范围远远小于真实值的变化范围，因而可以采用较少的位数来表示
  - 差分脉冲编码调制（DPCM）

多媒体数据压缩方法的分类

- **变换编码**
  - 主要思想：利用图像块内像素值之间的相关性，把图像变换到一组新的基上，使得能量集中到少数几个变换系数上，通过存储这些系数达到压缩的目的
  - 由于对整幅图像进行变换的计算量太大，所以一般把原始图像分成许多个矩形区域子图像独立进行变换
  - KLT、DCT、DFT 等

多媒体数据压缩方法的分类

- **统计编码**
  - 对于出现频率大的符号用较少的位数表示，而对于出现频率小的符号用较多的位数来表示
  - 编码效率取决于信源的概率分布
  - Huffman编码、算术编码、游程编码
- **混合编码**
  - 指合并变换和预测技术的编码方法
  - 例如：对动态图像而言，空域的二维变换再加上时间方向上的DPCM预测

内容提纲

- 数据压缩的重要性
- 数据压缩方法分类
- **数据压缩基础**

统计编码

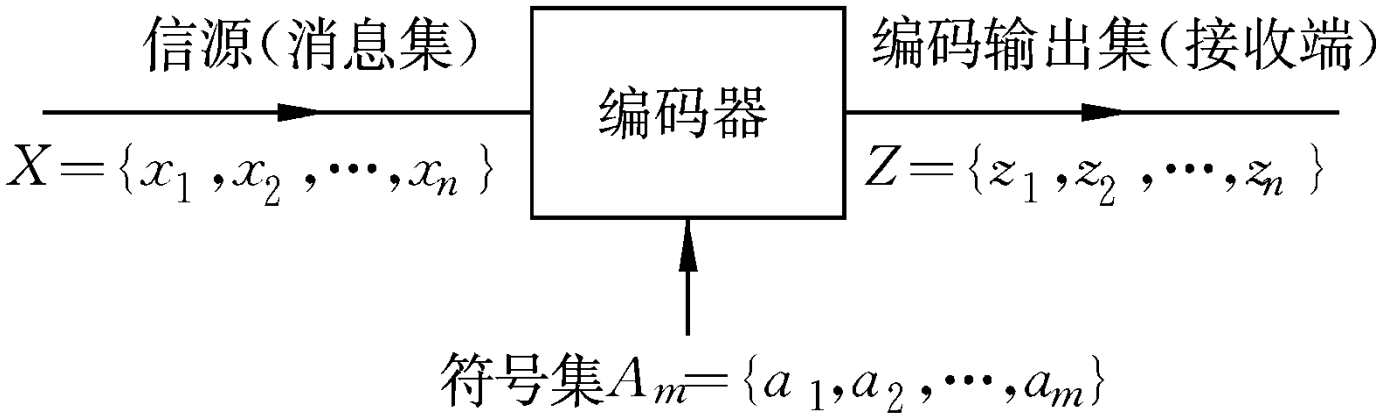
统计编码

- **数据压缩的理论基础是信息论**。根据信息论原理，可以找到最佳数据压缩编码方法，数据压缩的理论极限是信息熵

- 无失真编码（熵编码）
  - 编码过程中不丢失信息，即要求保存信息熵
- 有失真编码
  - 可允许信息部分损失以换取较高的压缩比

信息量和信息熵

- **编码器模型图**



- X是消息集，由几个信号单元x_j构成（j=1,2,…,n）
- Z是输出集，由几个码子z_j构成（ j=1,2,…,n ），z_j与x_j 一一对应
- A_m是符号集，由m个码元a_i构成（i=1,2,…,m），码元组成输出码字
- 问题：当信源发出某个随机事件（消息） x_j后，接收端收到一个相应的码子z_j，所收到的码子中包含多大的信息量？

信息量和信息熵

- 信息是用不确定性的度量定义的
  - 一个消息的可能性愈小，其信息愈多
  - 而消息的可能性愈大，则其信息愈少

- 信息论定义了一种度量信息量的方法

$$I(x_j) = -\log_a P(x_j) \qquad j=1,2,\ldots n$$

  - $P(x_j)$：信源X发出x_j的先验概率
  - $I(x_j)$：x_j发生后的自信息量，其含义是信源X发出x_j这个消息（随机事件）后，接收端收到信息量的度量

信息量和信息熵

- **信源的熵H(X)**定义为：信源X发出的n个随机事件 x_j (j=1,2,…,n) 的自信息量的统计平均，

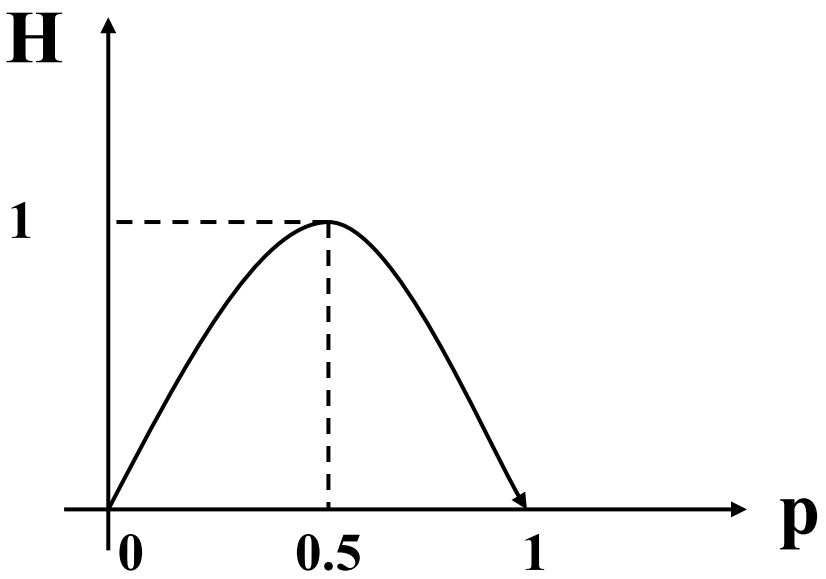
$$H(X) = E\{I(x_j)\} = \sum_{j=1}^{n} P(x_j) I(x_j)$$

$$= -\sum_{j=1}^{n} P(x_j) \log_a P(x_j)$$

  - 信源的熵H(X)：信源X发出任意一个随机变量的平均信息量
  - 当a取2时，H(X)的单位为比特(bit)

信息量和信息熵

- 熵的大小与信源的概率分布有着密切联系
- **等概率事件的熵最大**：当信源中各事件为等概率分布时，熵具有极大值$\log_2 n$。n为信源中事件个数



信息量和信息熵

- 以n=8为例，当$p(x_1)=p(x_2)=\ldots=p(x_8)=1/8$时，

$$H(X) = -\sum_{j=1}^{8} \frac{1}{8} \log_2 \frac{1}{8} = 3\ bits$$

- 当$p(x_1)=1$，必然有：$p(x_2)=p(x_3)=\ldots=p(x_8)=0$，

$$H(X) = -p(x_1) \log_2\ p(x_1) = 0$$

- 熵的范围为：$0 \le H(X) \le \log_2 N$

信息量和信息熵

- 在编码中用熵衡量是否为最佳编码。令$\tilde{N}$表示编码器输出码子的平均码长：

$$\tilde{N} = \sum_{j=1}^{n} p_j * N_j$$

- **熵值是平均码长的下限：**
  - 当分配给x_j的比特数$N_j = I(x_j) = -\log_2 p(x_j)$时，平均码长取极小值，即达到压缩极限。
  - 当 $\tilde{N} >> H(x)$，有冗余，不是最佳
  - 当 $\tilde{N} \approx H(x)$，最佳编码（$\tilde{N}$ 稍大于H(x)）
  - 当 $\tilde{N} < H(x)$，不可能

Huffman编码

- **最佳编码定理**
  - 在变字长码中，对于出现概率大的消息符号编以短字长的码，对于出现概率小的消息符号编以长字长的码，如果码子长度严格按照符号概率的大小的相反顺序排列，则平均码子长度一定小于按任何其他符号顺序排列方式得到的码子长度

Huffman编码

- **Huffman编码的具体步骤：**
  - 1、概率统计，得到n个不同概率的消息符号
  - 2、将n个信源消息符号的n个概率，按概率大小排序
  - 3、将n个概率中，最后两个小概率相加，这时候概率个数减为n-1个
  - 4、将n-1个概率，按大小重新排序
  - 5、重复3，将新排序后的最后两个小概率再相加，相加和与其余概率再排序
  - 6、如此反复重复n-2次，得到只剩两个概率序列
  - 7、以二进制码元(0,1)赋值，构成Huffman码子。结束

Huffman编码

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

Huffman编码

Original source			Source reduction									
Sym.	Prob.	Code	1		2		3		4			
a_2	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0		
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1		
a_1	0.1	011	0.1	011	0.2	010	0.3	01				
a_4	0.1	0100	0.1	0100	0.1	011						
a_3	0.06	01010	0.1	0101								
a_5	0.04	01011										

$\tilde{N}$ = (0.4)(1)+(0.3)(2)+(0.1)(3)+(0.1)(4)+(0.06)(5)+ (0.04)(5) = 2.2 bit/symbols

H = (0.4)log2(1/0.4)+(0.3)log2(1/0.3)+(0.1)log2(1/0.1)+(0.1)log2(1/0.1)
+(0.06)log2(1/0.06)+ (0.04)log2(1/0.04) = 2.1435 bit/symbols

0 1 0 1 0 0 1 1 1 1 0 0 ⟶ $a_3 a_1 a_2 a_2 a_6$

Huffman编码

- **总结如下特点**：
  - 平均码长 $\tilde{N} > H$（熵）
  - 平均码长 $\tilde{N} < 3$ bits（等长码需要的比特数）
  - 保证解码的唯一性，短码字不构成长码子的前缀
  - 在接收端需保存一个与发送端相同的Huffman码表（输入与Huffman的对应表）

Huffman编码

符号	S1	S2	S3	S4
出现概率	1/2	1/4	1/8	1/8
等长编码	00	01	10	11
霍夫曼	0	10	110	111

- $H(X) = 1.75$
- $\tilde{N}_1 = 2$
- $\tilde{N}_2 = 1.75$

Huffman编码

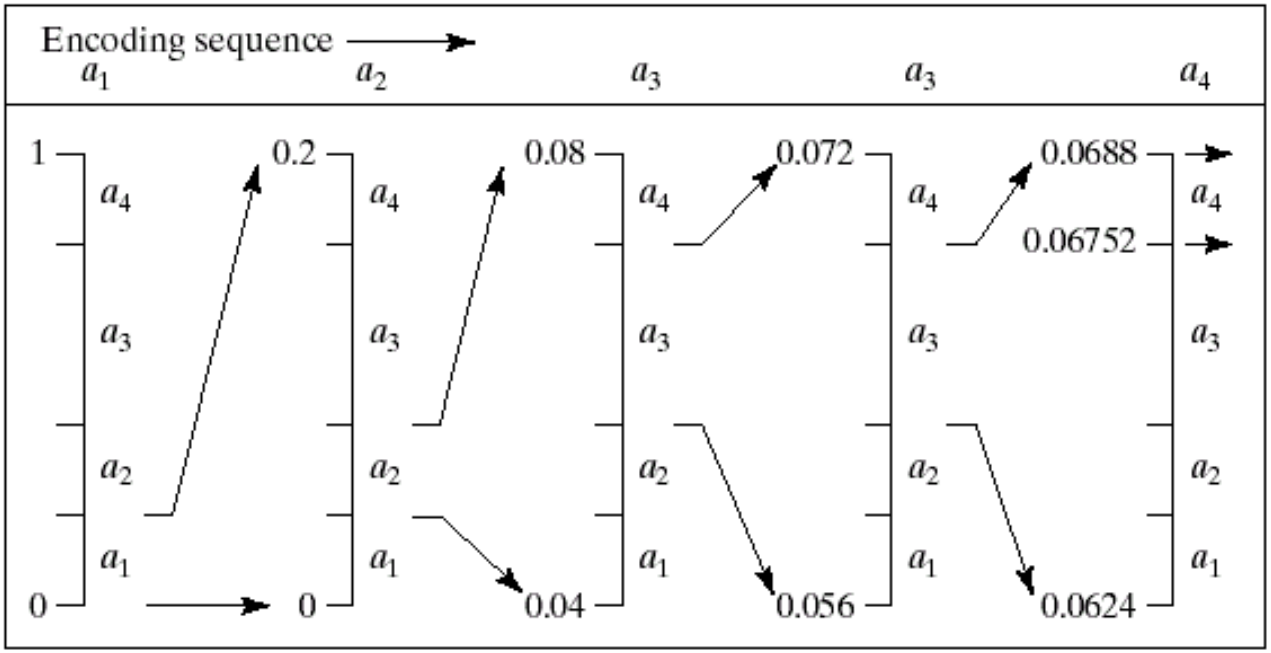
- **Huffman编码的局限**
  - 如果信源事件的概率分布不是2负n次方形式，编码是否可以达到最佳？原因…
  - 假设某个字符的出现概率为80%，该字符事实上只需要 -log2(0.8) = 0.322 位编码，但 Huffman 编码一定会为其分配一位 0 或一位 1 的编码
    - 可以想象，整个信息的 80% 在压缩后几乎相当于理想长度的 3 倍左右，压缩效果可想而知。

算术编码

- 基本思想：**算术编码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用来缩短这个区间。消息序列中元素越多，所得到的区间就越小，表示这个区间所需的位数就更多
  - 采用算术编码每个符号的平均编码长度可以为小数，因此可以更加接近无损压缩的熵极限

算术编码

Source Symbol	Probability	Initial Subinterval
a_1	0.2	[0.0, 0.2)
a_2	0.2	[0.2, 0.4)
a_3	0.4	[0.4, 0.8)
a_4	0.2	[0.8, 1.0)



如何编码：

$a_3\ a_3\ a_1\ a_2\ a_4$

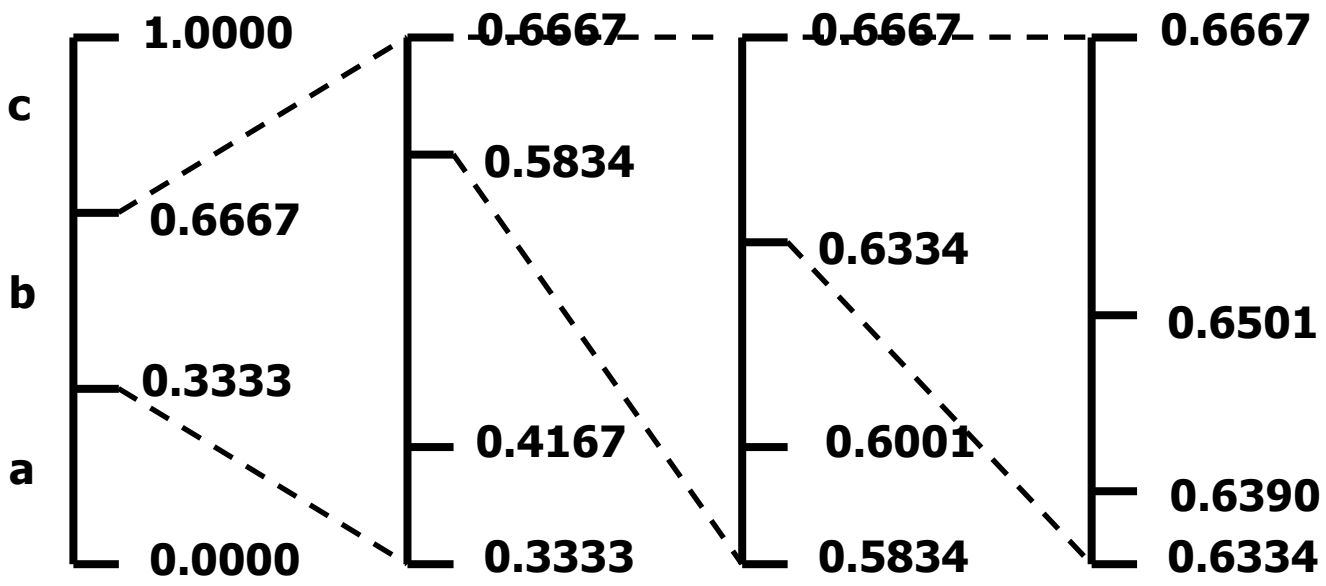
自适应算术编码

输入序列为：bcc……

自适应概率模型

假设编码之前我们并不知道信源的概率模型

先假设所有信源符号出现的概率相等，然后再根据输入符号自适应地调整概率模型



c	1/3	1/4	2/5	3/6
b	1/3	2/4	2/5	2/6
a	1/3	1/4	1/5	1/6

游程编码

- 现实中有许多这样的图像，在一幅图像中具有许多颜色相同的图块。在这些图块中，许多行上都具有相同的颜色，或者在一行上有许多连续的象素都具有相同的颜色值

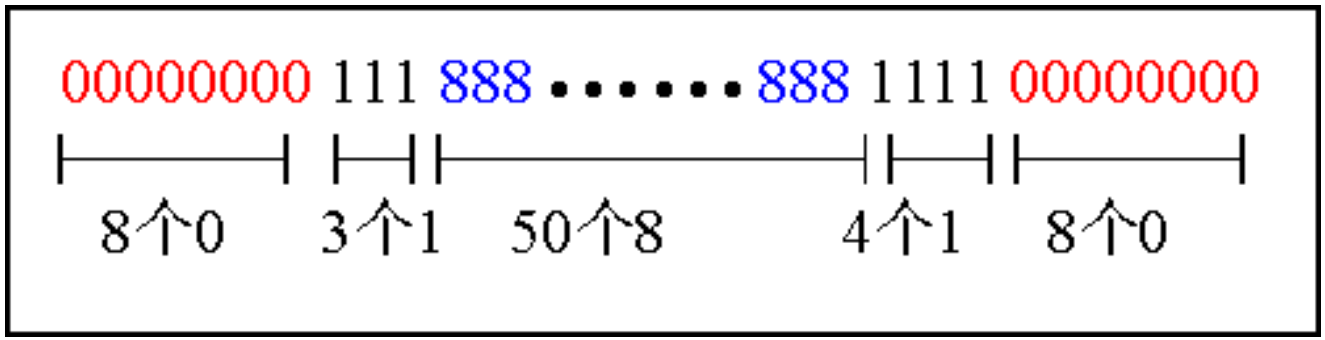
- 游程编码（Run-Length Encoding）：它通过将信源中相同符号序列转换成一个计数字段再加上一个重复字符标志实现压缩

游程编码

- 用RLE编码方法得到的代码为：

8 0 3 1 50 8 4 1 8 0

- 代码中用粗体表示的数字是游程长度，其后的数字代表像素的颜色值。例如**50**代表有连续50个像素具有相同的颜色值，它的颜色值是8
- 压缩比约为：7 : 1



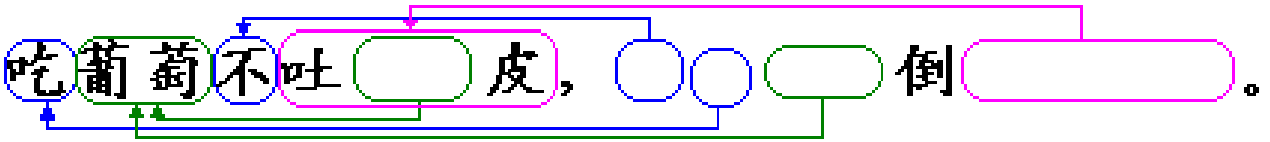
游程编码

- RLE所能获得的压缩比有多大，这主要是取决于图像本身的特点
  - 如果图像中具有相同颜色的图像块越大，图像块数目越少，获得的压缩比就越高。反之，压缩比就越小

- **RLE通常需要和其他压缩编码技术联合应用**

词典编码

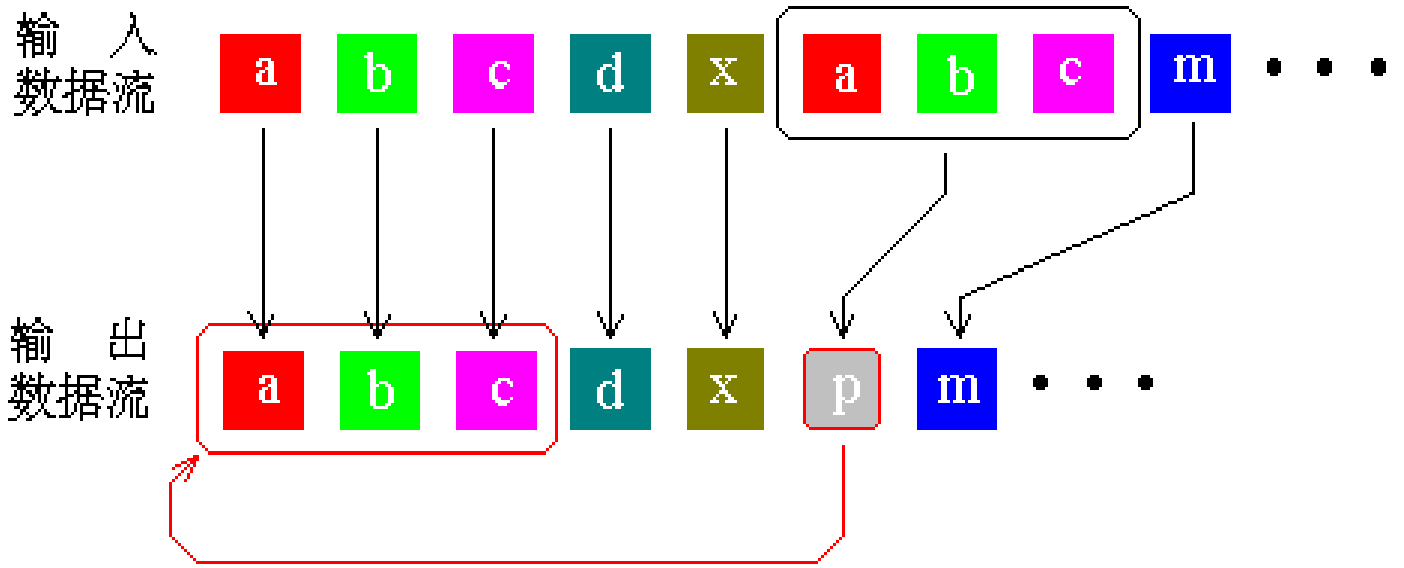
- 词典编码主要利用数据本身包含许多重复代码的特性（文本文件和光栅图像）
  - 例如：吃葡萄不吐葡萄皮，不吃葡萄倒吐葡萄皮。我们如果用一些简单的代号代替这些字符串，就可以实现压缩，实际上就是利用了信源符号之间的相关性。字符串与代号的对应表就是词典



  - 实用的词典编码算法的核心就是如何动态地形成词典，以及如何选择输出格式以减小冗余

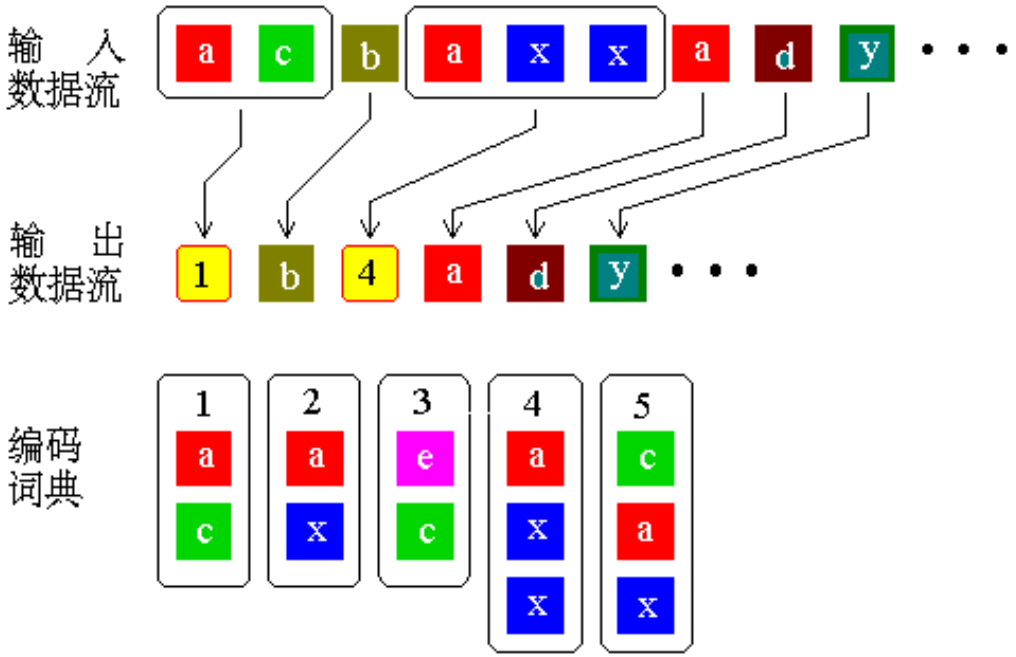
词典编码

- 第一类词典法编码的思想：
  - 企图查找正在压缩的字符序列是否在以前输入的数据中出现过，然后用已经出现过的字符串替代重复的部分，输出仅仅是指向早期出现过的字符串的“指针”



词典编码

- 第二类词典法编码的思想：
  - 企图从输入的数据中**创建一个“短语词典”**，这种短语可以是任意字符的组合。编码数据过程中当遇到已经在词典中出现的“短语”时，编码器就输出这个短语的“索引号”，而不是短语本身

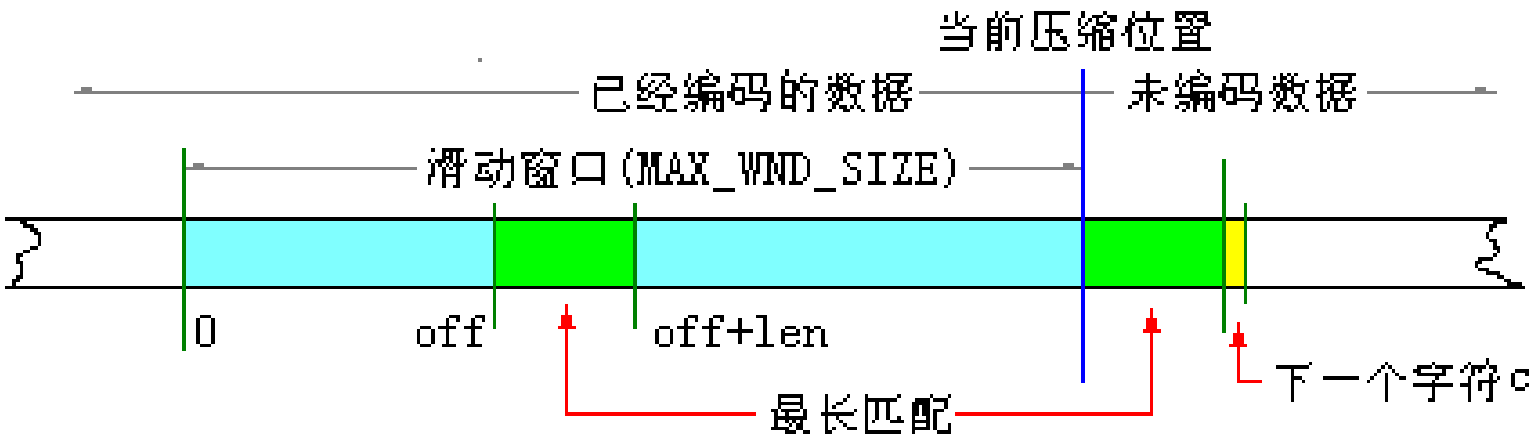


词典编码

- 第一类词典法
  - LZ77算法：1977年，A. Lempel和J. Ziv开发的算法
  - LZSS算法：1982年，由Storer和Szymanski改进的算法
- 第二类词典法
  - LZ78算法：1978年，J. Ziv和A. Lempel首次发表了介绍这种编码方法的文章
  - LZW算法：1984年，Terry A. Weltch发表了改进这种编码算法的文章

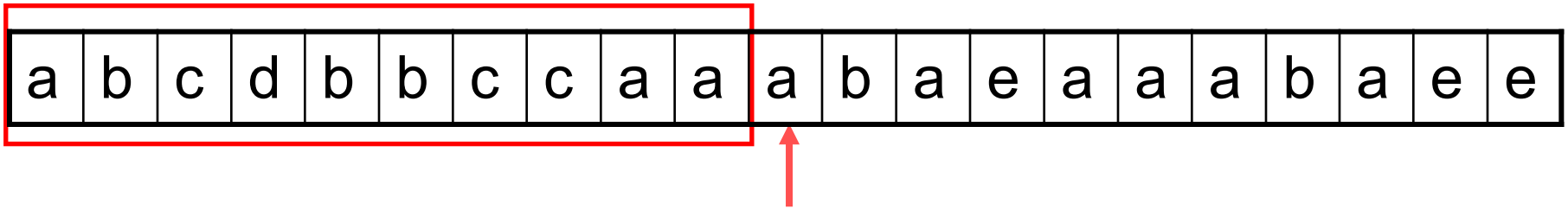
LZ77算法

- **LZ77编码算法步骤：**
  - Step 1：从当前压缩位置开始，考察未编码的数据，并试图在滑动窗口中找出最长的匹配字符串，如果找到，则进行步骤 2，否则进行步骤 3
  - Step 2：输出**三元符号组 (off, len, c)**。其中 off 为窗口中匹配字符串相对窗口边界的偏移，len 为可匹配的长度，c 为下一个字符。然后将窗口向后滑动 len + 1 个字符，继续步骤 1
  - Step 3：输出三元符号组 (0, 0, c)。其中 c 为下一个字符。然后将窗口向后滑动 len + 1 个字符，继续步骤 1



LZ77算法

- 假设窗口的大小为 10 个字符，刚编码过的 10 个字符是 abcdbbccaa，即将编码的字符是abaeaaabaee



步骤	位置	匹配串	输出
1	1	ab	（0,2） a
2	4	——	（0, 0） e
3	5	aaabae	（4, 6） e

- 解压过程：只要像压缩时那样维护好滑动的窗口，随着三元组的不断输入，在窗口中找到相应的匹配串，缀上后继字符 c 输出（如果 off 和 len 都为 0 则只输出后继字符 c ）即可还原出原始数据。

LZSS算法

- LZ77通过输出真实字符解决窗口中没有匹配串的问题，但这个解决方案包含冗余信息
  - 冗余信息表现在两个方面，一是空指针，二是编码器可能输出额外的字符，这种字符是指可能包含在下一个匹配串中的字符

- **LZSS算法的思想**
  - 如果匹配串的长度比指针本身的长度长就输出指针(匹配串长度≥MIN_LENGTH)，否则就输出真实字符。另外要输出额外的标志位区分是指针还是字符

LZSS算法

- **LZSS算法步骤：**
  - Step 1：把编码位置置于输入数据流的开始位置。
  - Step 2：在前向缓冲存储器中查找与窗口中最长的匹配串
    - ① Pointer ：=匹配串指针。
    - ② Length ：=匹配串长度。
  - Step 3：判断匹配串长度Length是否大于等于最小匹配串长度(Length >= MIN_LENGTH)
    - 如果“是”：输出指针，然后把编码位置向前移动Length个字符
    - 如果“否”：输出前向缓冲存储器中的第1个字符，然后把编码位置向前移动一个字符。
  - Step 4：如果前向缓冲存储器不是空的，就返回到步骤 2

LZSS算法

输入数据流：

位置	1	2	3	4	5	6	7	8	9	10	11
字符	A	A	B	B	C	B	B	A	A	B	C

编码过程

MIN_LEN=2

步骤	位置	匹配串	输出
1	1	——	A
2	2	A	A
3	3	——	B
4	4	B	B
5	5	——	C
6	6	BB	（3，2）
7	8	AAB	（7，3）
8	11	C	C

LZSS算法

- LZSS算法比LZ77可获得更高的压缩比，而译码同样简单
  - 许多后来开发的文档压缩程序都使用了LZSS的思想。例如，PKZip, GZip, ARJ, LHArc和ZOO等等，其差别仅仅是指针的长短和窗口的大小等有所不同

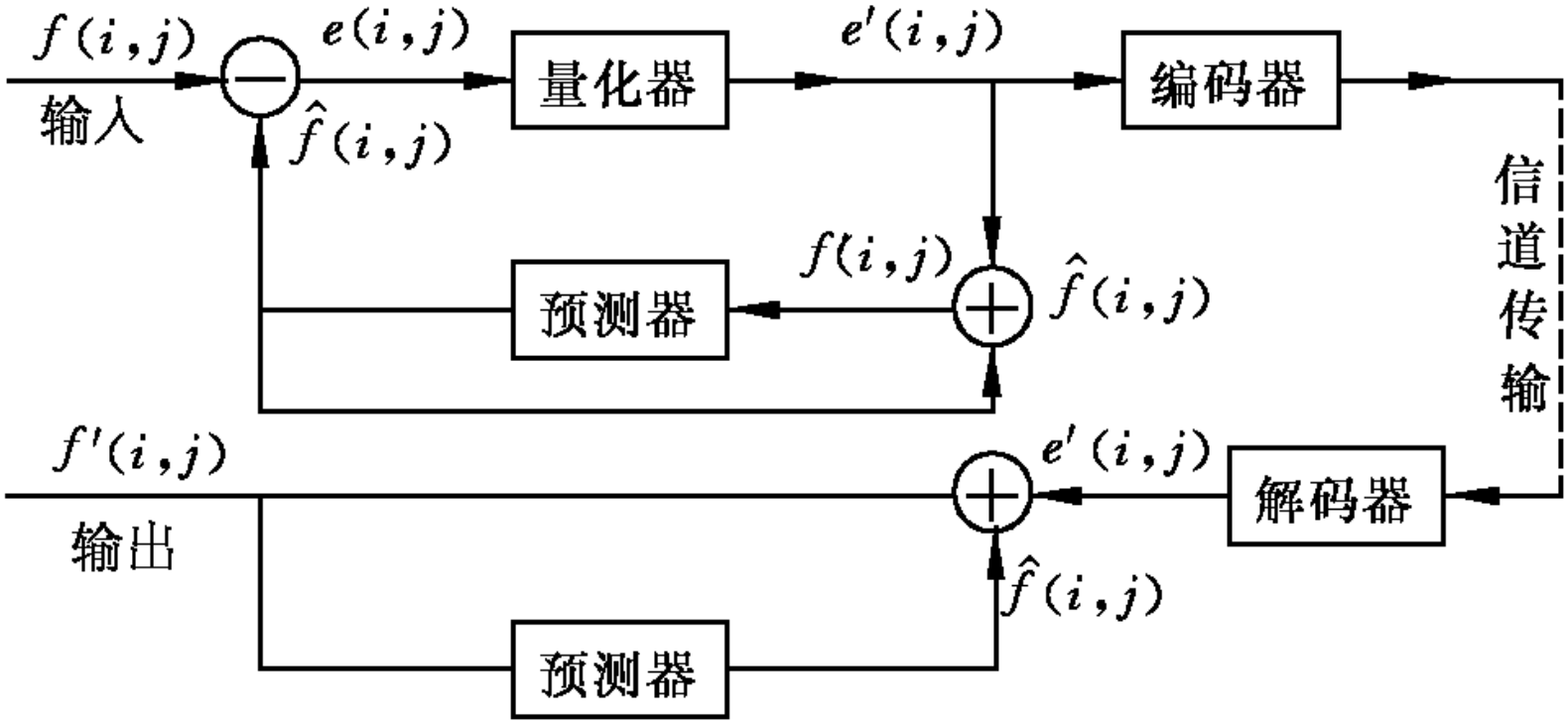
- LZSS同样可以和熵编码联合使用
  - 例如ARJ就与霍夫曼编码联用，而PKZip则与Shannon-Fano联用，它的后续版本也采用霍夫曼编码

预测编码

预测编码基本思想

- 基本思想：
  - 根据某一模型利用以往的样本值对于新的样本值进行预测，然后将样本的实际值与其预测值相减得到一个误差值，对这一误差值进行编码
  - 如果模型足够好且样本序列在时（空）间上的相关性较强，误差信号的幅度将远远小于原始信号，因此可使用较少的位数对其量化

DPCM的基本原理



预测误差：　　$e(i,j) = f(i,j) - \hat{f}(i,j)$

DPCM的基本原理

- 由像素(i, j)的3个相邻像素得到的三点预测，定义为：

$$\hat{f}(i,j) = a_1 f(i,j-1) + a_2 f(i-1,j-1) + a_3 f(i-1,j)$$

构成三阶预测器。其中a_1, a_2, a_3称预测系数，都是待定参数。预测误差为：

$$\begin{aligned} e(i,j) &= f(i,j) - \hat{f}(i,j) \\ &= f(i,j) - [a_1 f(i,j-1) \\ &\quad + a_2 f(i-1,j-1) \\ &\quad + a_3 f(i-1,j)] \end{aligned}$$

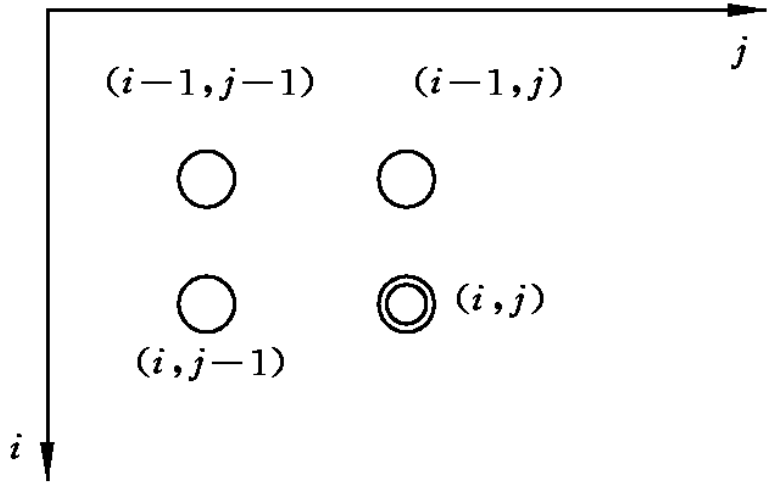
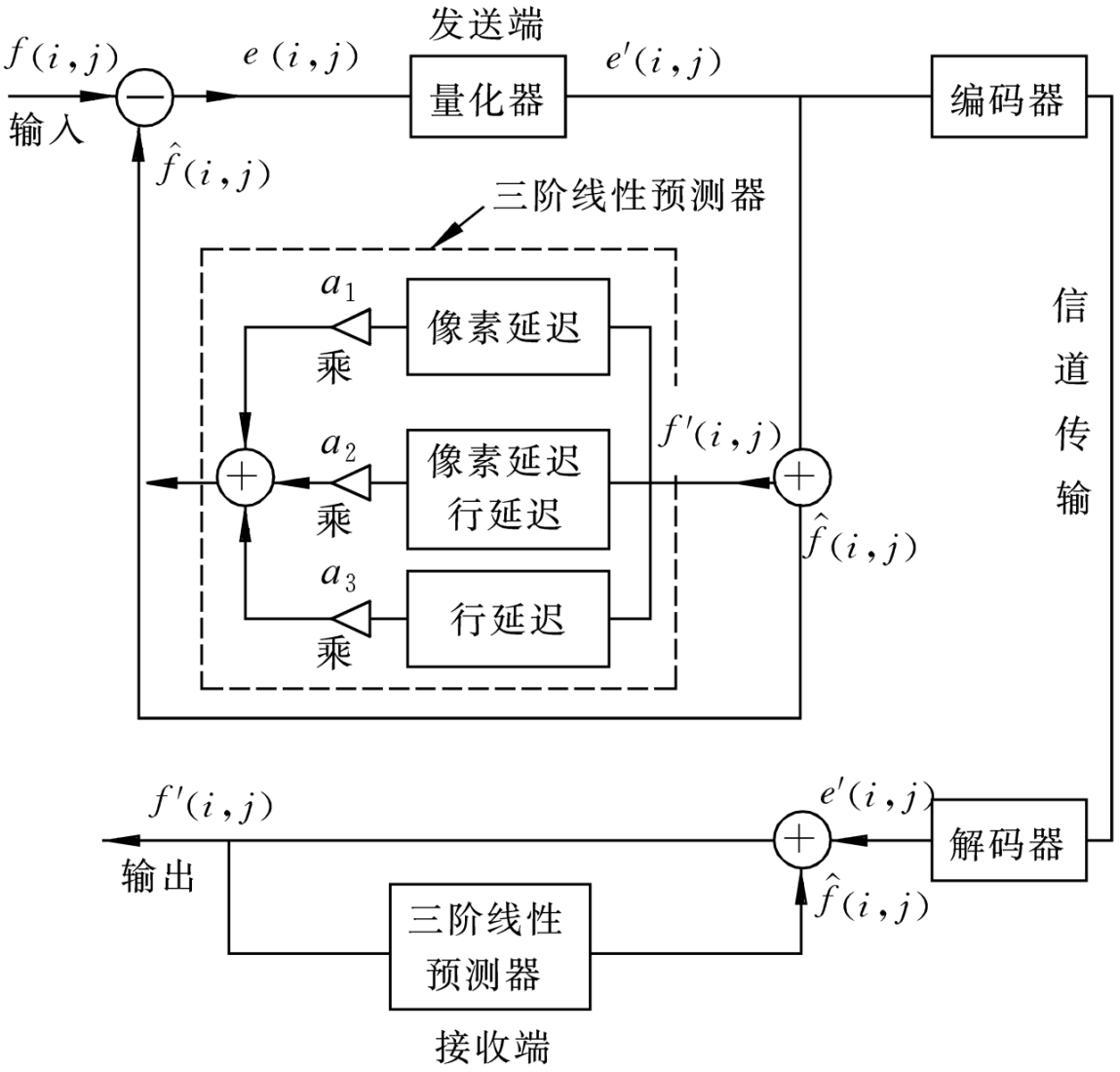


图 4.14　预测域

DPCM的基本原理

三阶DPCM线性预测



DPCM的基本原理

- **基于均方误差最小准则的最佳线性预测**
  - 均方误差的表达式为：

$$\bar{e}^2 = E\{e(i,j)^2\} = E\{[f(i,j) - \hat{f}(i,j)]^2\}$$
$$= E\{[f(i,j) - a_1 f(i,j-1) - a_2 f(i-1,j-1) - a_3 f(i-1,j)]^2\}$$

  - 将预测值与实际值之间的均方误差对a_1, a_2, a_3求偏导，令

$$\frac{\partial \bar{e}^2}{\partial a_1} = 0, \quad \frac{\partial \bar{e}^2}{\partial a_2} = 0, \quad \frac{\partial \bar{e}^2}{\partial a_3} = 0,$$

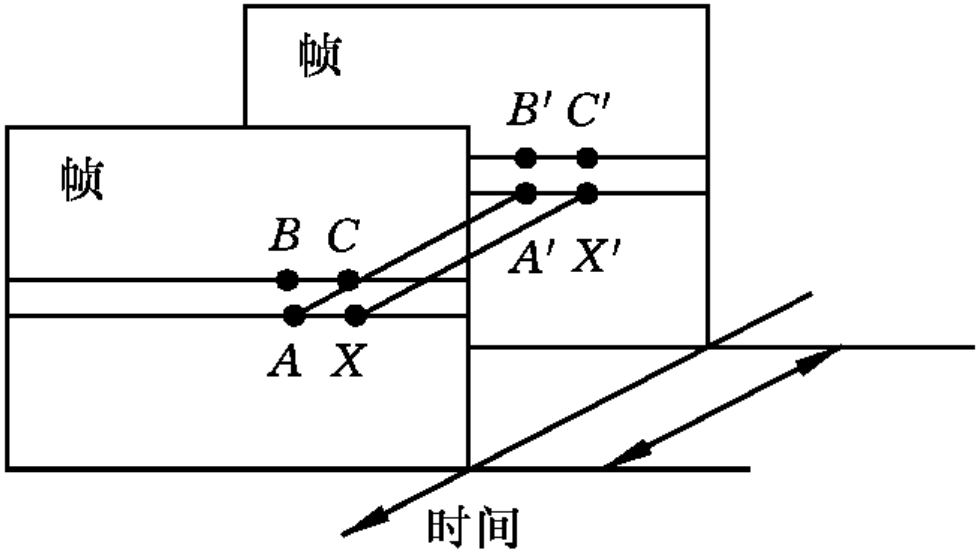
  - 解方程，得a_1, a_2, a_3，即为最佳线性预测系数

自适应预测编码

- **DPCM**
  - 当预测系数和量化器参数一次设计好后，整幅图都用这套参数，不再改变
- **自适应差分脉冲编码调制（ADPCM）**
  - 自适应预测和自适应量化：预测器的预测系数和量化器的量化参数，能够根据图像的局部区域分布特点而自动调整
  - 与DPCM相比，ADPCM不仅能改善图像的评测质量和视觉效果，同时还能进一步压缩数据

帧间预测编码

- 帧间编码技术处理的对象是序列图像（运动图像）
- 基于预测技术的帧间预测编码方法：
  - (1) 条件补充法
  - (2) 运动补偿技术

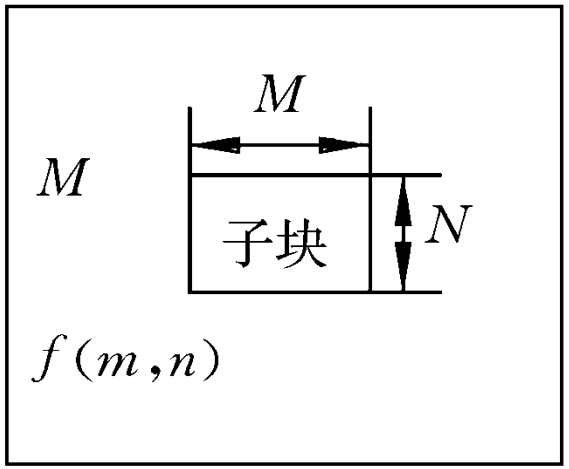


条件补充法

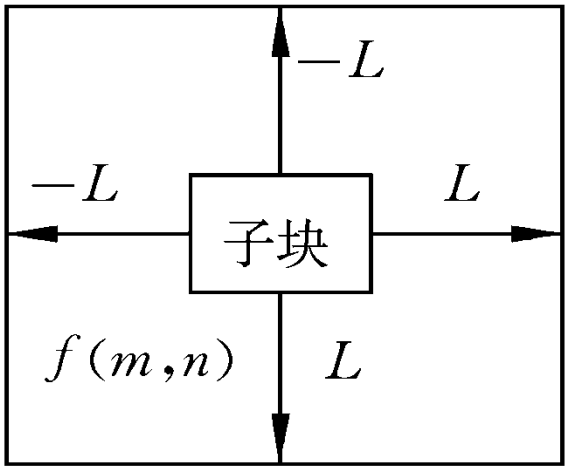
- **条件像素补充法**
  - 若帧间各对应像素的亮度差超过阈值，则把这些像素存入缓冲存储器，并以恒定的传输速度传送；而阈值以下的像素则不传送，在接收端用上一帧相应像素值代替
  - 一幅电视图像可能只传送其中较少部分的像素，且传送的只是帧间差值，压缩比较高
  - 可视电话中，只需传送6%左右的像素

运动补偿技术

- **运动补偿技术**
  - 是MPEG标准中使用的主要技术之一。尤其适合于运动部分只占整个画面较小的电视会议、可视电话
  - 关键是运动向量的计算



第K帧（当前帧）



第$K-N_s$帧

运动补偿技术

- 块匹配算法
  - 1、判别两个子块匹配的准则

$$NCCF(i,j)=\frac{\sum_{m=1}^{M}\sum_{n=1}^{N}f_k(m,n)f_{k-N_s}(m+i,n+j)}{[\sum_{m=1}^{M}\sum_{n=1}^{N}f_k^2(m,n)]^{1/2}[\sum_{m=1}^{M}\sum_{n=1}^{N}f_{k-N_s}^2(m+i,n+j)]^{1/2}}$$

$$MSE(i,j)=\sum_{m=1}^{M}\sum_{n=1}^{N}[f_k(m,n)-f_{k-N_s}(m+i,n+j)]^2,\ (i,j)\in SR$$

$$MAD(i,j)=\sum_{m=1}^{M}\sum_{n=1}^{N}\left|f_k(m,n)-f_{k-N_s}(m+i,n+j)\right|,\ (i,j)\in SR$$

  - 2、计算量最小的搜索方法
    - 全搜索、二维对数法、三步法、共轭方向法、正交搜索法等

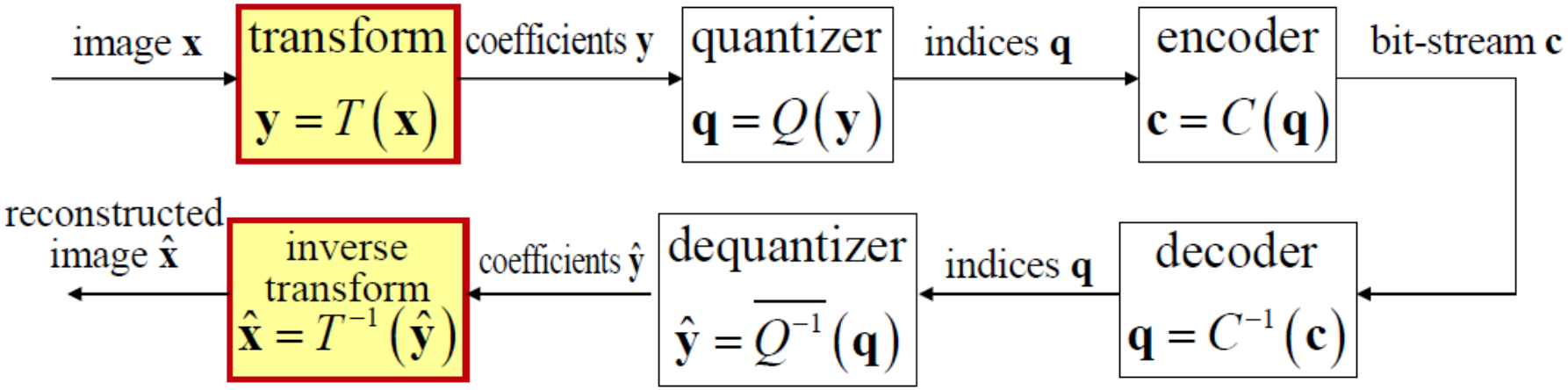
变换编码

变换编码基本思想

- **变换编码不是直接对空域图像信号编码，而是首先将空域图像信号映射变换到另一个正交矢量空间（变换域或频域），产生一批变换系数，然后对这些变换系数进行编码处理**

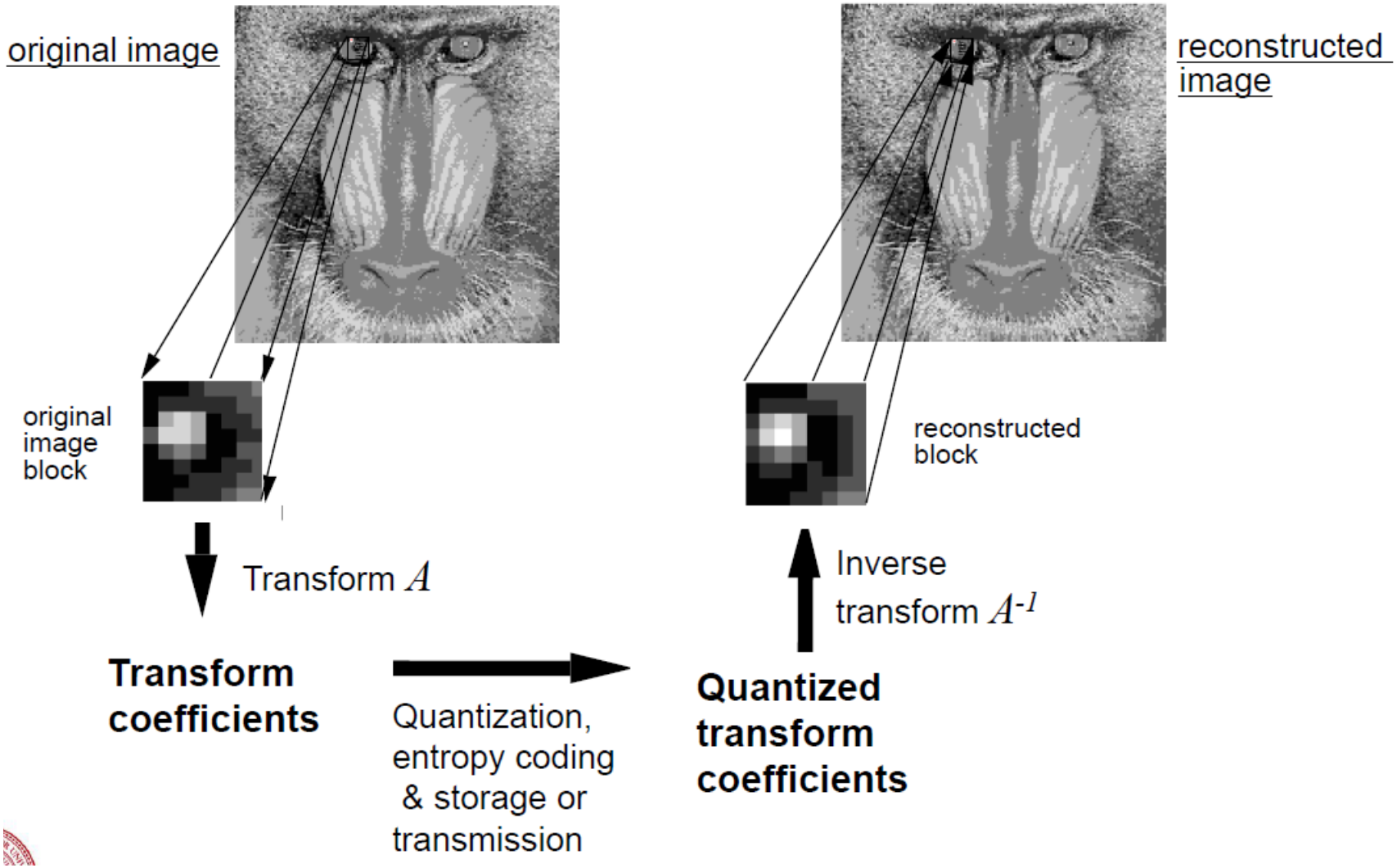
- 正交变换种类
  - 傅里叶(Fourier)变换
  - K-L (Karhunen-Loeve)变换
  - 余弦变换、正弦变换
  - 沃尔什(Walsh)变换
  - 哈尔(Haar)变换等

变换编码框图



- Transform $T(\boldsymbol{x})$ usually invertible
- Quantization $Q(\mathbf{y})$ not invertible, introduces distortion
- Combination of encoder $C(\mathbf{q})$ and decoder $C^{-1}(\mathbf{c})$ lossless

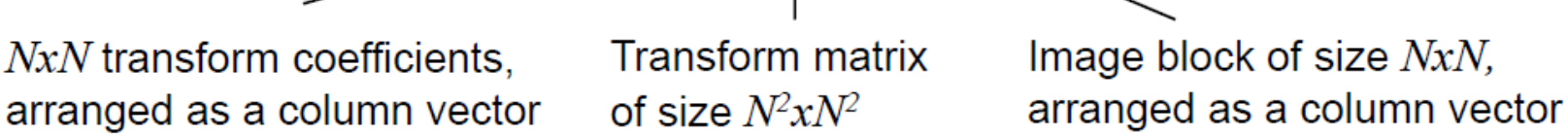
基于块的变换编码


original image
reconstructed image
original image block
reconstructed block
Transform A
Inverse transform A^{-1}
Transform coefficients
Quantization, entropy coding & storage or transmission
Quantized transform coefficients

正交变换性质

- Forward transform

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$



- Inverse transform

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{y} = \mathbf{A}^{\mathrm{T}}\mathbf{y}$$

- Linearity: $\mathbf{x}$ is represented as linear combination of "basis functions" (i.e., columns of $\mathbf{A}^{\mathrm{T}}$)

能量守恒

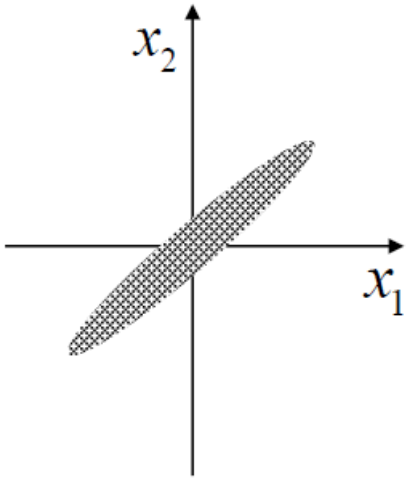
- For any orthonormal transform $\mathbf{y} = \mathbf{A}\mathbf{x}$

$$\|\mathbf{y}\|^2 = \mathbf{y}^T\mathbf{y} = \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} = \|\mathbf{x}\|^2$$

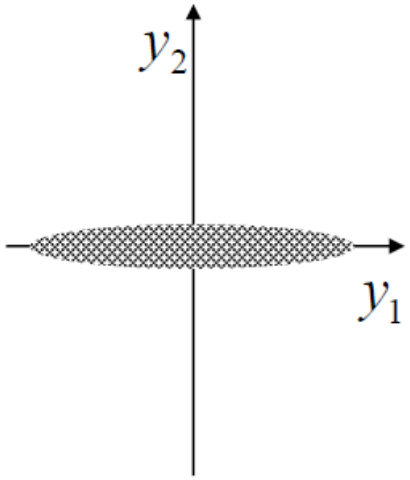
- Interpretation
  - Vector length („energies“) conserved
  - Orthonormal transform is a rotation of the coordinate system around the origin (plus possible sign flips)

二维正交变换

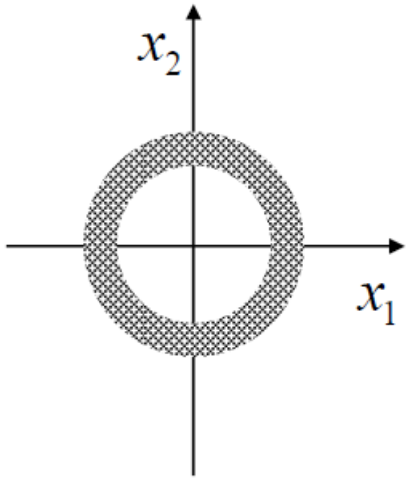
$$\mathbf{A} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$



Strongly correlated samples,
equal energies

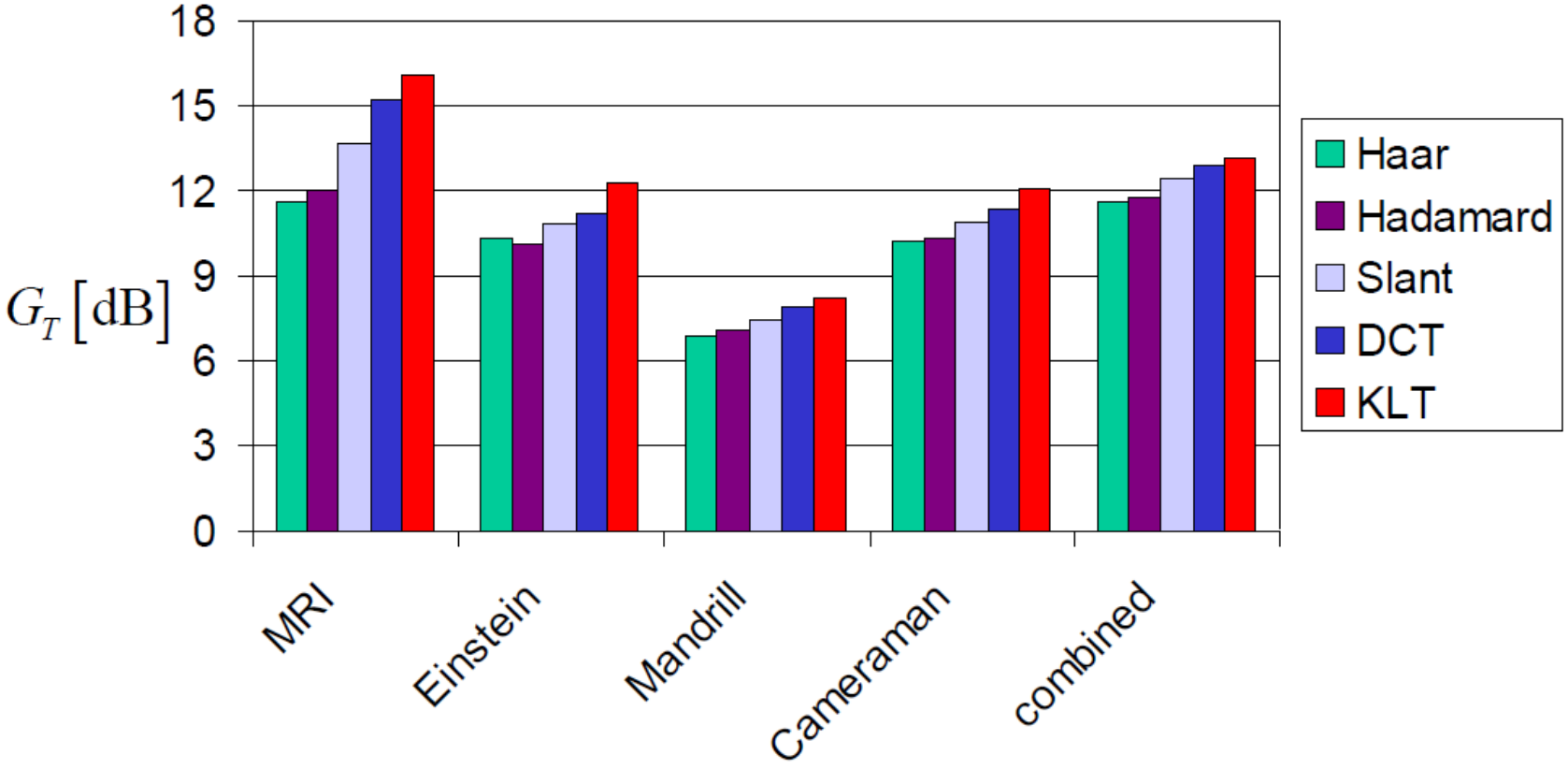


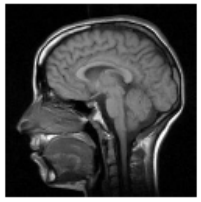
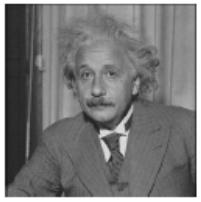
After transform:
uncorrelated samples, most of the energy in first coefficient



Despite statistical dependence, orthonormal transform won't help.

编码增益


18
15
12
9
6
3
0
G_T [dB]
MRI
Einstein
Mandrill
Cameraman
combined
Haar
Hadamard
Slant
DCT
KLT

离散余弦变换（DCT）

- Type II-DCT of blocksize NxN is defined by transform matrix A containing elements

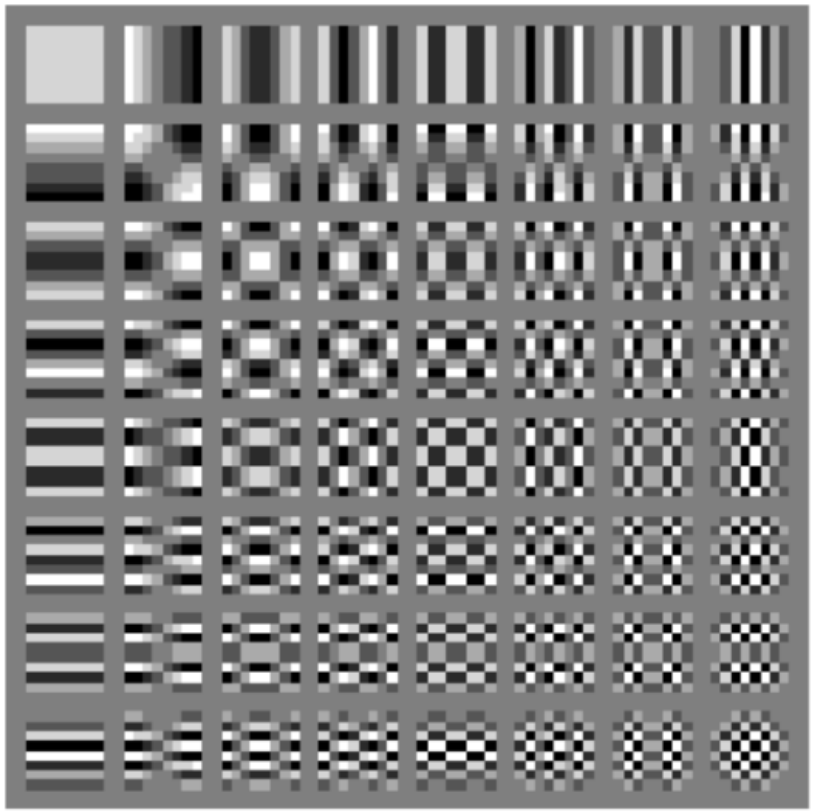
$$a_{ik} = \alpha_i \cos \frac{\pi(2k+1)i}{2N}$$

$$\text{for } i,k = 0,\ldots,N-1$$

$$\text{with } \alpha_0 = \sqrt{\frac{1}{N}}$$

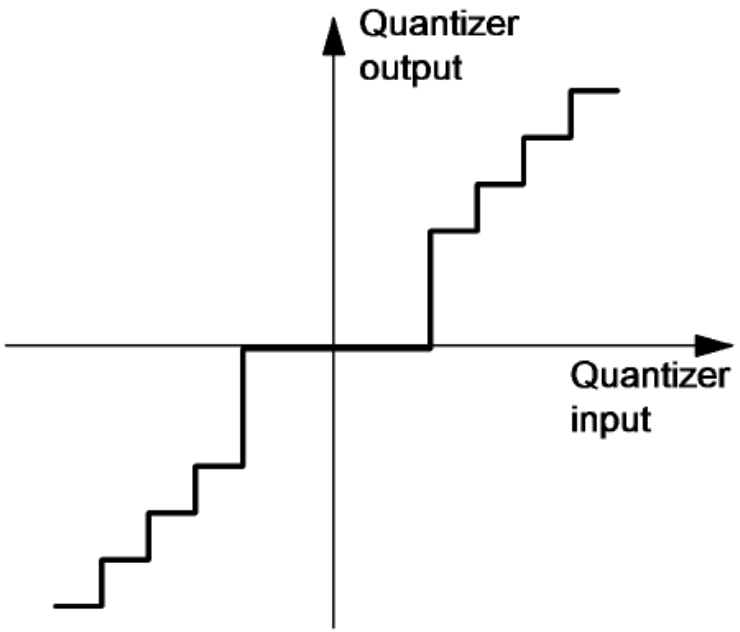
$$\alpha_i = \sqrt{\frac{2}{N}} \quad \forall i \neq 0$$

- 2D DCT basis functions:



对变换系数的编码

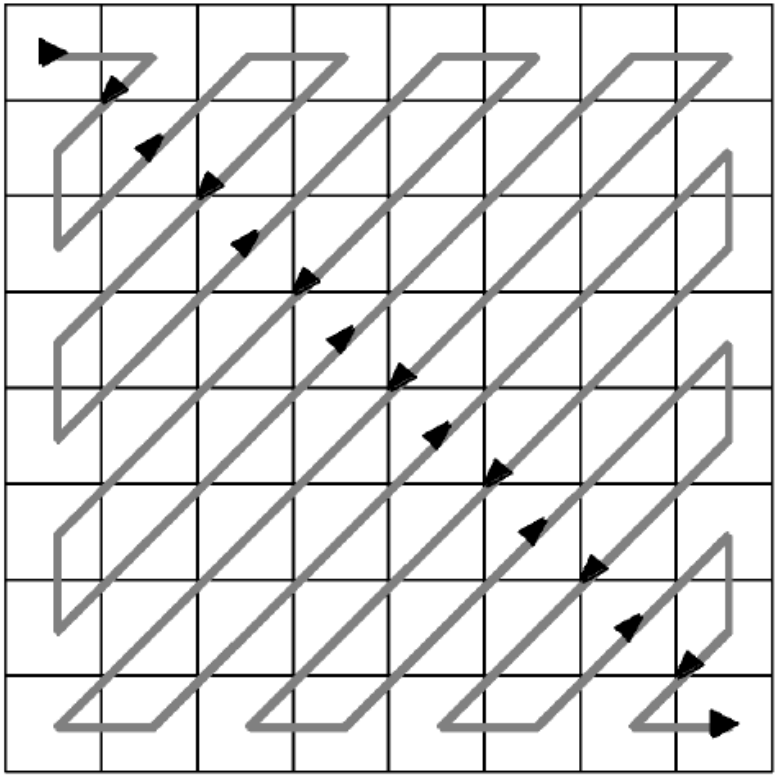
- Uniform deadzone quantizer: transform coefficients that fall below a threshold are discarded.



- Positions of non-zero transform coefficients are transmitted in addition to their amplitude values.

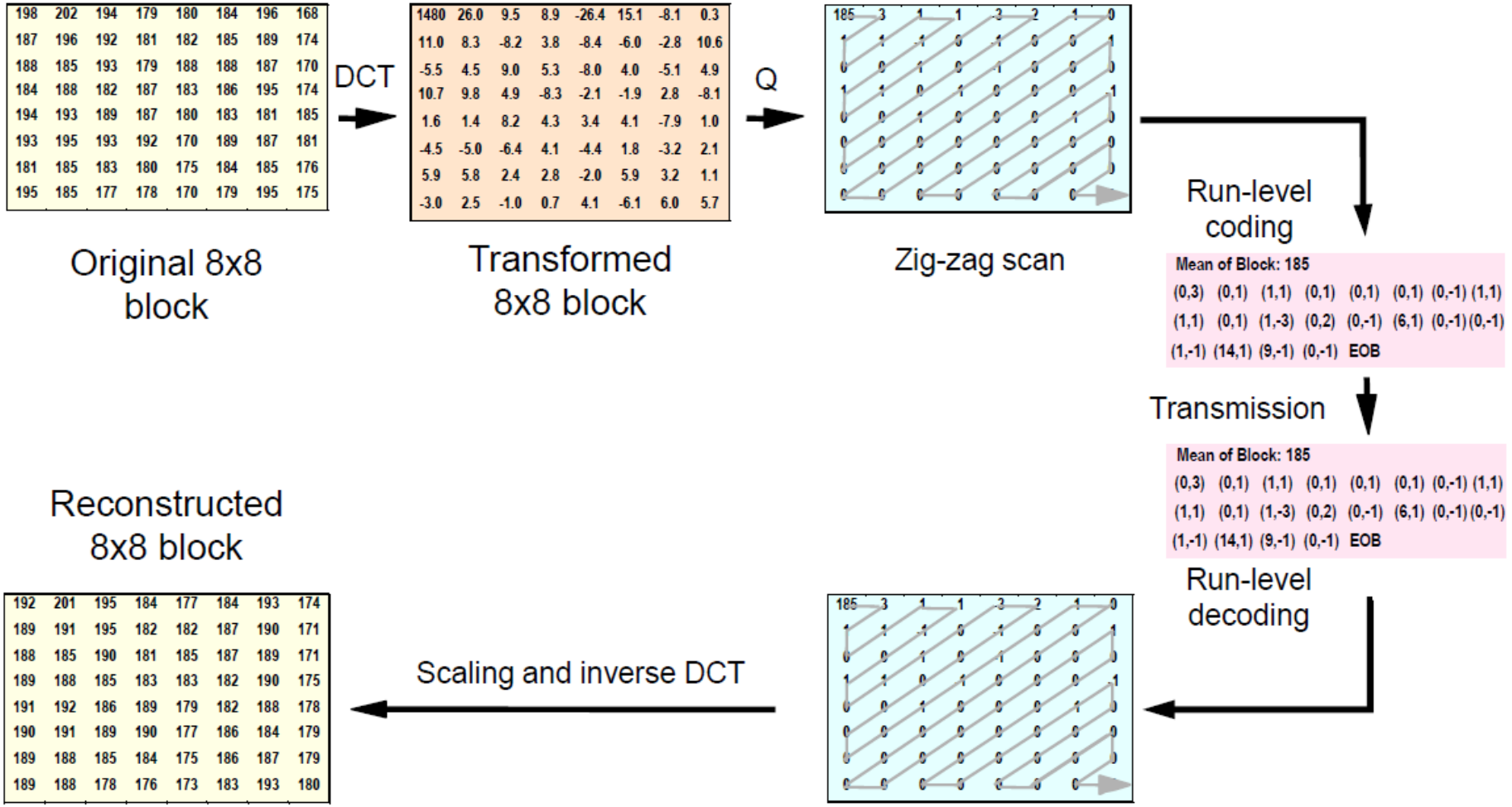
对变换系数的编码

- Efficient encoding of the position of non-zero transform coefficients: zig-zag-scan + run-level-coding



ordering of the transform coefficients by zig-zag-scan

对变换系数的编码


198 202 194 179 180 184 196 168
187 196 192 181 182 185 189 174
188 185 193 179 188 188 187 170
184 188 182 187 183 186 195 174
194 193 189 187 180 183 181 185
193 195 193 192 170 189 187 181
181 185 183 180 175 184 185 176
195 185 177 178 170 179 195 175
Original 8x8 block
DCT
1480 26.0 9.5 8.9 -26.4 15.1 -8.1 0.3
11.0 8.3 -8.2 3.8 -8.4 -6.0 -2.8 10.6
-5.5 4.5 9.0 5.3 -8.0 4.0 -5.1 4.9
10.7 9.8 4.9 -8.3 -2.1 -1.9 2.8 -8.1
1.6 1.4 8.2 4.3 3.4 4.1 -7.9 1.0
-4.5 -5.0 -6.4 4.1 -4.4 1.8 -3.2 2.1
5.9 5.8 2.4 2.8 -2.0 5.9 3.2 1.1
-3.0 2.5 -1.0 0.7 4.1 -6.1 6.0 5.7
Transformed 8x8 block
Q
Zig-zag scan
Run-level coding
Mean of Block: 185
(0,3) (0,1) (1,1) (0,1) (0,1) (0,1) (0,-1) (1,1)
(1,1) (0,1) (1,-3) (0,2) (0,-1) (6,1) (0,-1) (0,-1)
(1,-1) (14,1) (9,-1) (0,-1) EOB
Transmission
Mean of Block: 185
(0,3) (0,1) (1,1) (0,1) (0,1) (0,1) (0,-1) (1,1)
(1,1) (0,1) (1,-3) (0,2) (0,-1) (6,1) (0,-1) (0,-1)
(1,-1) (14,1) (9,-1) (0,-1) EOB
Run-level decoding
Scaling and inverse DCT
Reconstructed 8x8 block
192 201 195 184 177 184 193 174
189 191 195 182 182 187 190 171
188 185 190 181 185 187 189 171
189 188 185 183 183 182 190 175
191 192 186 189 179 182 188 178
190 191 189 190 177 186 184 179
189 188 185 184 175 186 187 179
189 188 178 176 173 183 193 180

DCT编码失真

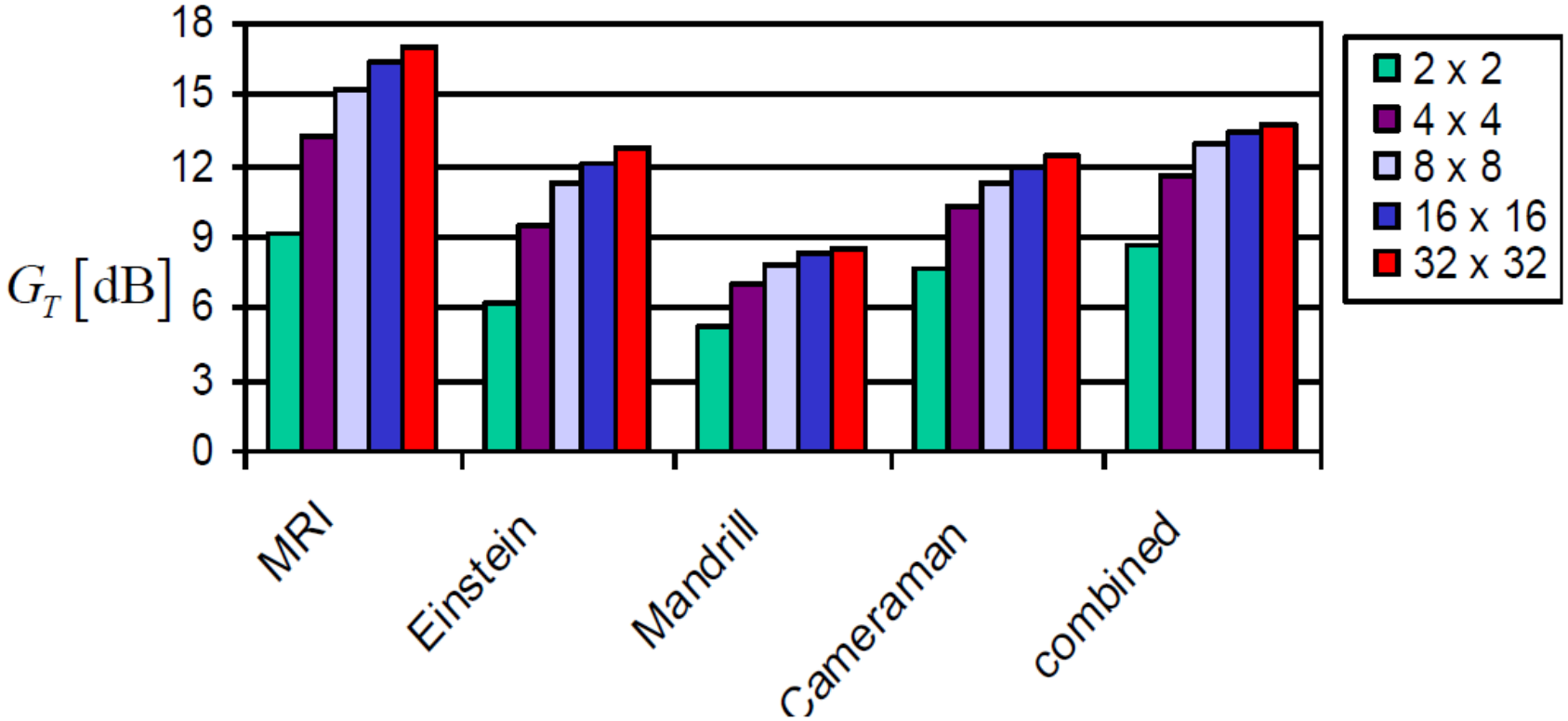
DCT coding with increasingly coarse quantization, block size 8x8

quantizer stepsize
for AC coefficients: 25

quantizer stepsize
for AC coefficients: 100

quantizer stepsize
for AC coefficients: 200

图像块尺寸的影响


18
15
12
9
6
3
0
G_T [dB]
MRI
Einstein
Mandrill
Cameraman
combined
2 x 2
4 x 4
8 x 8
16 x 16
32 x 32

变换编码小结

- Orthonormal transform: rotation of coordinate system in signal space
- Purpose of transform: decorrelation, energy concentration
- Bit allocation proportional to logarithm of variance, equal distortion
- KLT is optimum, but signal dependent and, hence, without a fast algorithm
- DCT shows reduced blocking artifacts compared to DFT
- 8x8 block size, uniform quantization, zig-zag-scan + run-level coding is widely used today (e.g. JPEG, MPEG, ITU-T H.261, H.263)