图像压缩

叶志鹏

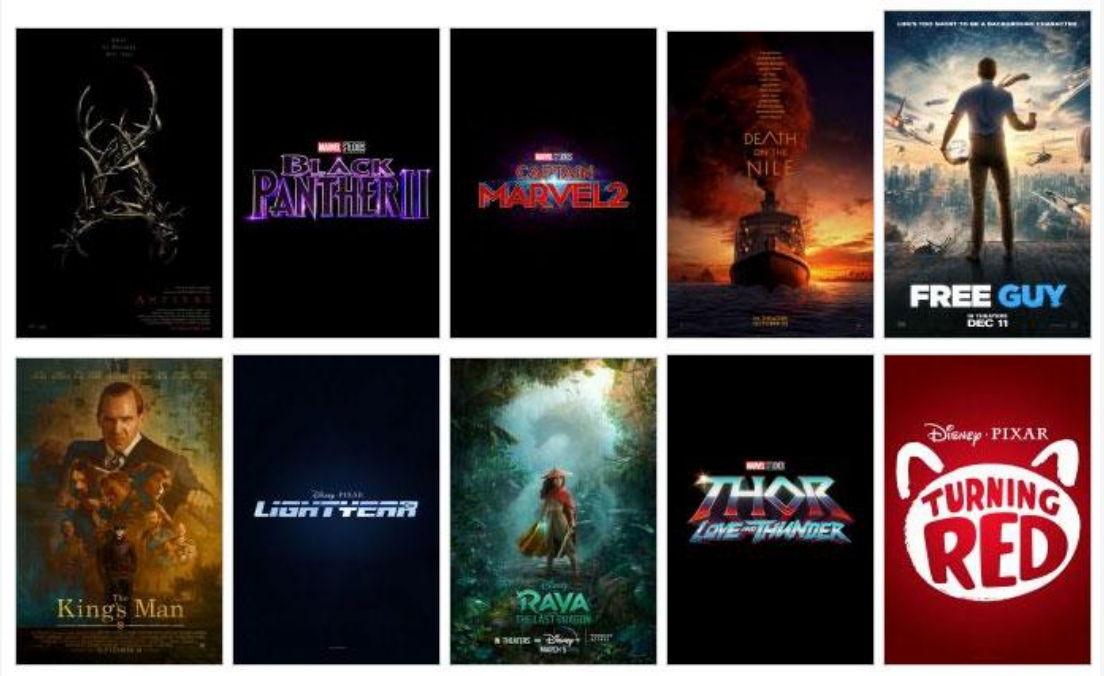
图像压缩

- 图像压缩的基本概念
- 有损压缩与无损压缩的概念
- 三种图像冗余
  - 编码冗余 （固定长度编码/变长编码）
  - 像素间冗余 （差分预测编码）
  - 心理视觉冗余 （变换域编码）
- JPEG图像格式的压缩步骤与原理

为什么我们需要图像压缩技术

- 考虑如下场景，使用 720P分辨率（1280*720 pixels, standard HD），24bit 颜色深度播放 2小时的电影，视频以 30 frame/s 速度播放。求该电影的文件大小?

$$1280 * 720 * 24 * 30 * 60 * 60 * 2$$
$$= 4777574400000 \text{ bit}(b)$$
$$= 597196800000\ byte(B) = 597196800\ KB$$
$$= 597196.8 \text{ MB} \approx 597GB$$



1KB = 1,000 Byte
1MB = 1,000 KB
1GB = 1,000,000 KB
1TB = 1,000,000,000 KB

1KiB = 1,024Byte
1MiB = 1,024KiB
1GiB = 1,024MiB = 1,048,576 KiB
1TiB = 1,024GiB = 1,073,741,824 KiB

为什么我们需要图像压缩技术

- 1080P(Full High Definition), 大概需要 1.3TB 大小
- 2K (Quad HD)，大概需要2.4TB
- 4K(Ultra HD)，大概需要5.4TB
- 8K(Full Ultra HD), 大概需要21.5TB

……很可怕

怎么存储?

怎么传输?

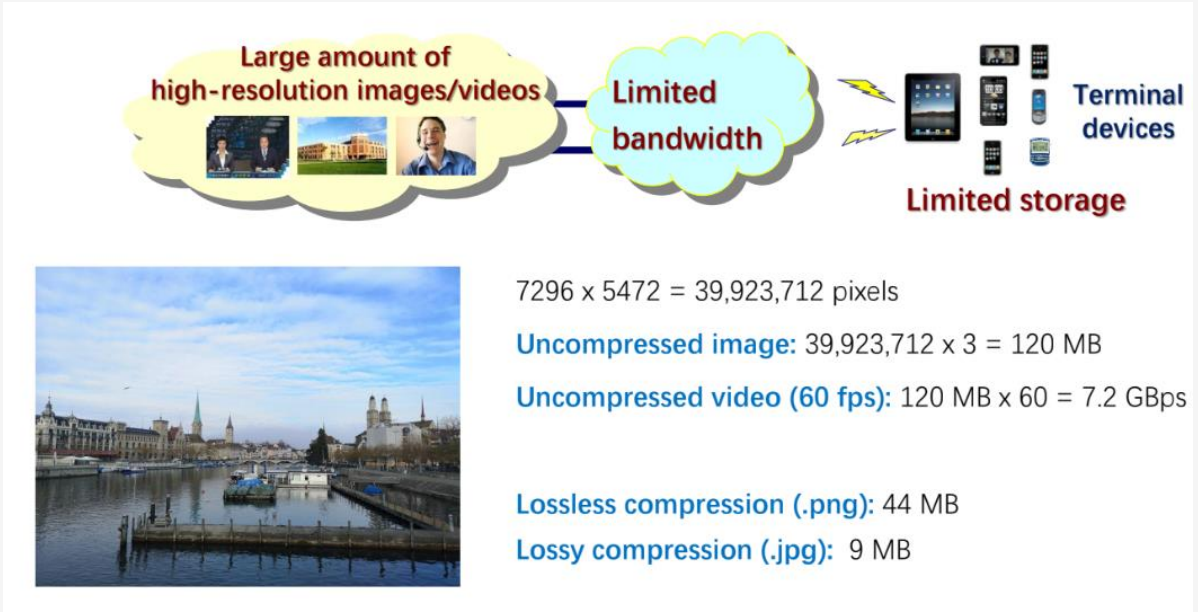
bilibili

YouTube

NETFLIX prime video hulu Disney+

什么是图像压缩

- 图像压缩的基本任务是减少数据量来表达信息。
- 图像压缩算法的设计方向
  - 编码冗余
  - 时间与空间冗余
  - 不相关信息（人类视觉心理冗余信息）
- 图像压缩的好处：
  - 减少数据存储大小
  - 减少数据传输时间和消耗
  - 减少数据运算所需的内存



什么是数据（Data），什么是信息(Information)?

- 数据是信息的载体
- 数据是信息的具体表现形式（符号、文字、数字、语音、图像、视频）

开心

- 信息论给出了信息的数学定义：消除事件不确定性的物理量。
- 香农是与冯·诺伊曼、图灵、维纳齐名的计算机相关领域科学家。
- 信息论广泛应用于通信，密码，人工智能领域。

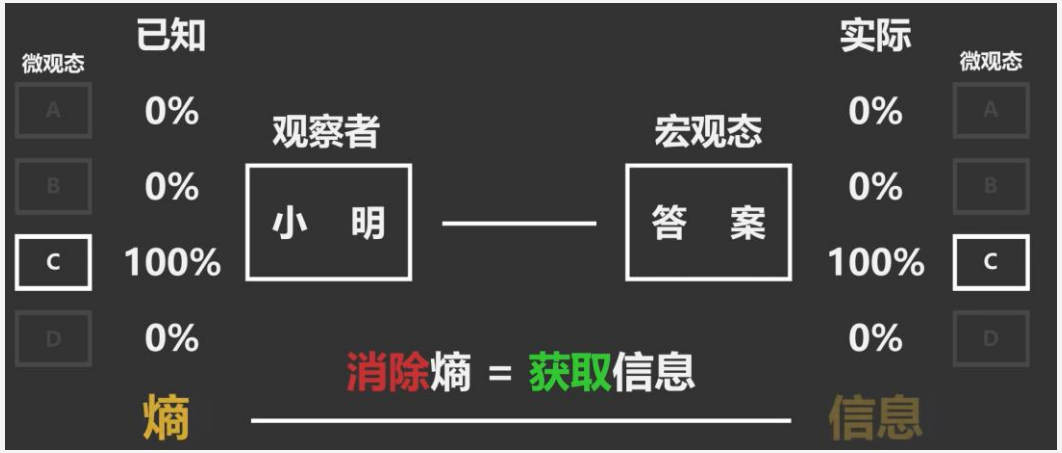
Claude Elwood Shannon (April 30, 1916 – February 24, 2001) was an American mathematician, electrical engineer, and cryptographer known as a "father of information theory"

信息熵 Entropy （重点）

- 随机变量 X 的不确定性，又叫信息熵，定义如下：

$$H(X) = -\sum_{k=1}^{n} p_k \log_2 p_k = \sum_{k=1}^{n} p_k \log_2 \frac{1}{p_k}$$

- 例如，小明不会做某数学选择题，有4种可能选项 A, B, C, D。 此时随机变量X不确定性大小为（$4 * \frac{1}{4}\log_2 \frac{1}{4} = 2\ bit$）
  - 隔壁小强告诉小明正确答案选A,那么概率分布律就变成了$100\%, 0\%, 0\%, 0\%$。此时不确定性为0，小强提供了 2 bit 信息，让小明知道了答案，消除了不确定性。



- 隔壁小强告诉小明正确选项在A,B两项之间
- 隔壁小强告诉小明 C是错误选项时，提供了多少信息?
- 隔壁小强告诉小明，答案在A,B,C,D中，提供了多少信息?

图像的信息熵

- 计算灰度频率（直方图），然后计算出信息熵。

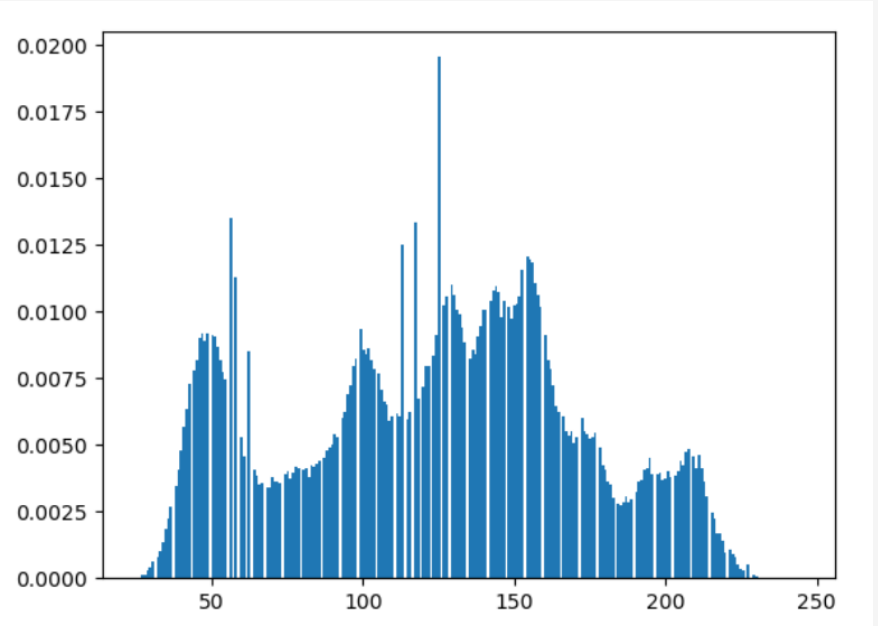
```
from skimage import io, measure

image1 = io.imread('boundary_extr.png')
image2 = io.imread('lenna512.bmp')

image1_entropy = measure.shannon_entropy(image1)
image2_entropy = measure.shannon_entropy(image2)

print(f'image 1 entropy:{image1_entropy}, image 2
entropy:{image2_entropy}')
```

代码



$$H(X) = -\sum_{k=1}^{n} p_k \log_2 p_k = \sum_{k=1}^{n} p_k \log_2 \frac{1}{p_k}$$

图像的信息熵（重点）

- 计算图像的信息熵?

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 7 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 7 & 5 & 5 & 5 & 5 & 2 & 2 \\ 0 & 7 & 0 & 0 & 0 & 0 & 2 & 2 \\ 0 & 7 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

符号 Symbols	频率
0	½
2	¼
1	1/8
5	1/16
7	1/16

压缩效率的衡量指标

- CR (Compression ratio) 衡量了压缩前后的比值，CR越大，压缩程度越高。公式定义如下：

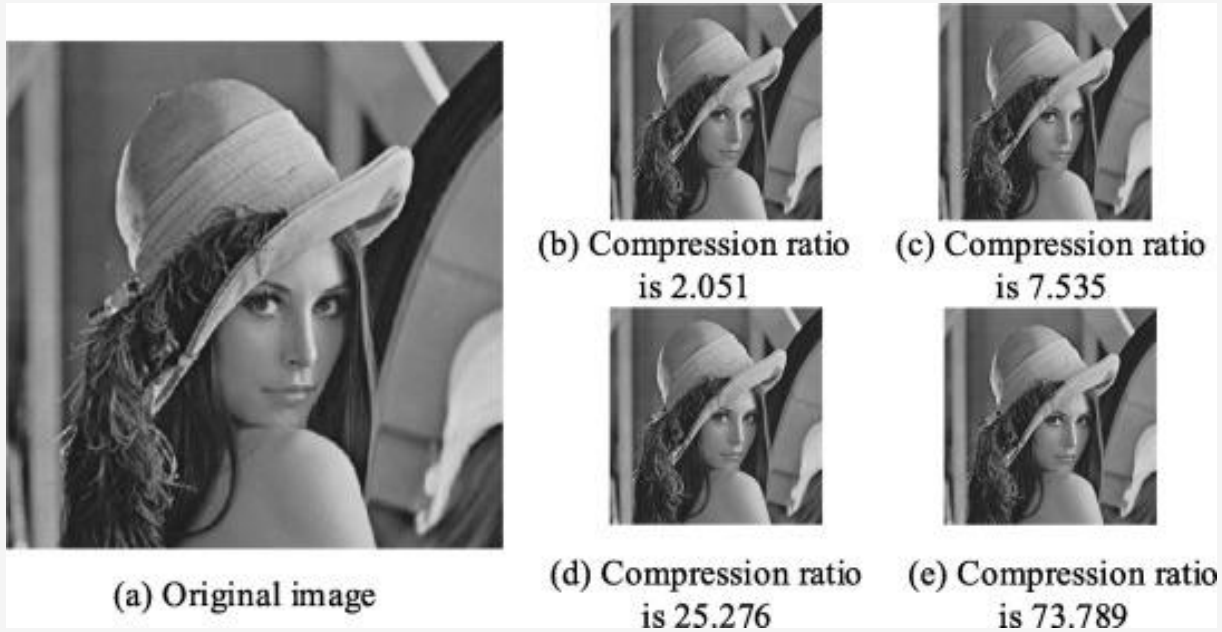
$$\mathrm{CR} = \frac{\text{原图像}}{\text{压缩后图像}} > 1$$

- Compression rate （非正式表达，压缩率）

$$compression\ rate = \frac{\text{压缩后图像}}{\text{原图像}} < 1$$

- 相对数据冗余R，描述了有多少比例数据被压缩掉了，或者是相对冗余的。定义如下：

$$\mathrm{R} = 1 - \frac{1}{CR} = 1 - compression\ rate$$



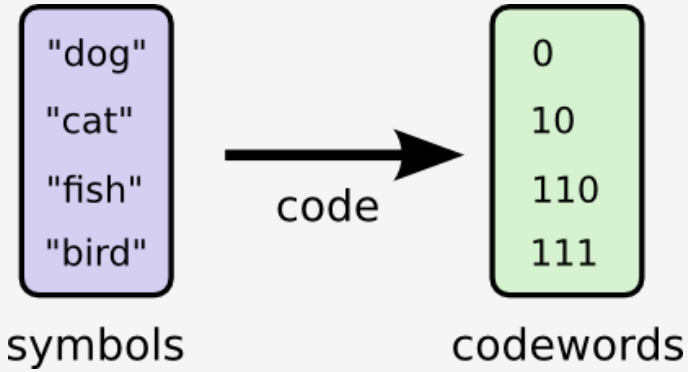
压缩类型（重点）

- 图像压缩可以分为无损压缩和有损压缩。

区别	无损压缩	有损压缩
解压方式	解压后能够得到原文件	解压后只能接近原文件质量
瓶颈	> 信息熵 Entropy	可以小于信息熵
经验 CR	CR < 10	CR ≥ 10
应用场景	医疗，军事，航空	多媒体应用，电视

无损压缩

- 无损压缩的设计目标是最小化平均编码长度，单位 bit。
- 无损压缩通过挖掘数据的统计属性：
  - 灰度的概率分布
  - 编码冗余（编码冗余是用于表示信息实体的符号系统。每个符号赋予一个编码符号的序列，称为码字。）



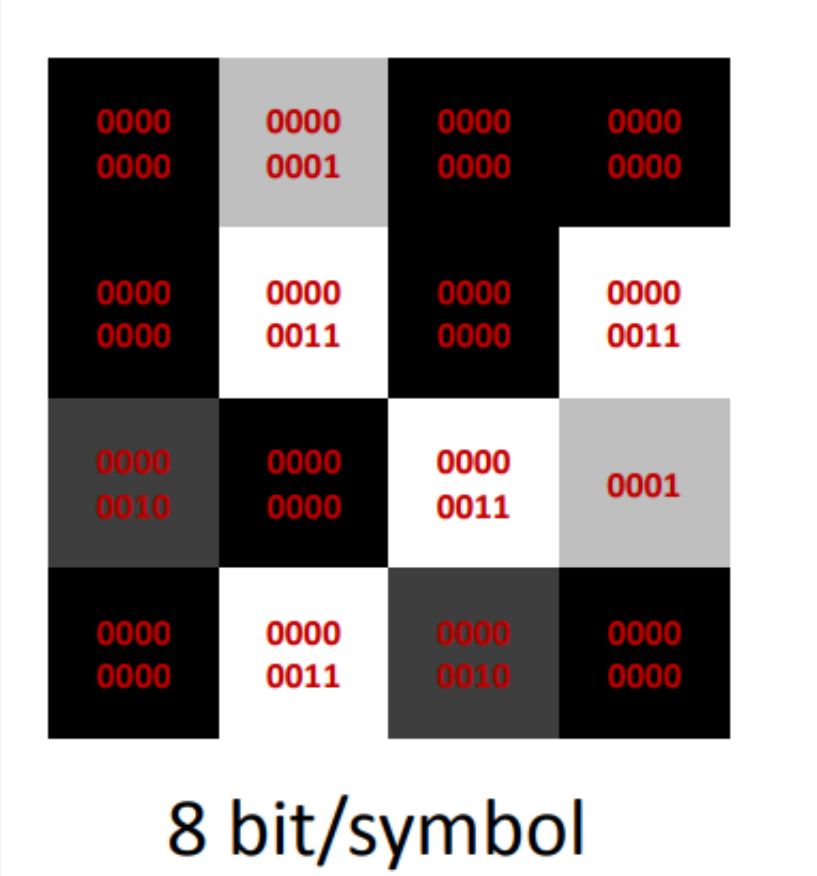
平均编码长度公式：

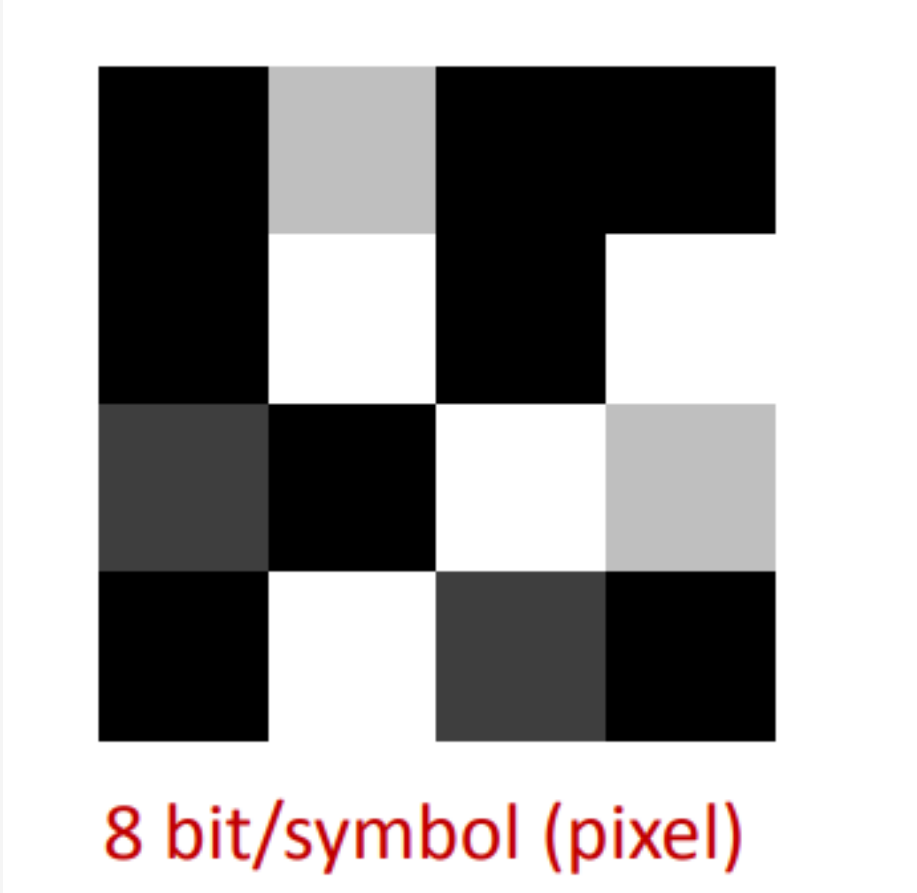
$$L_{avg} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k)$$

$l(r_k)$代表第k个灰度级别，对应的编码长度；$L-1$是灰度级别；$p_r(r_k)$表示该灰度级别对应的概率或者频率。

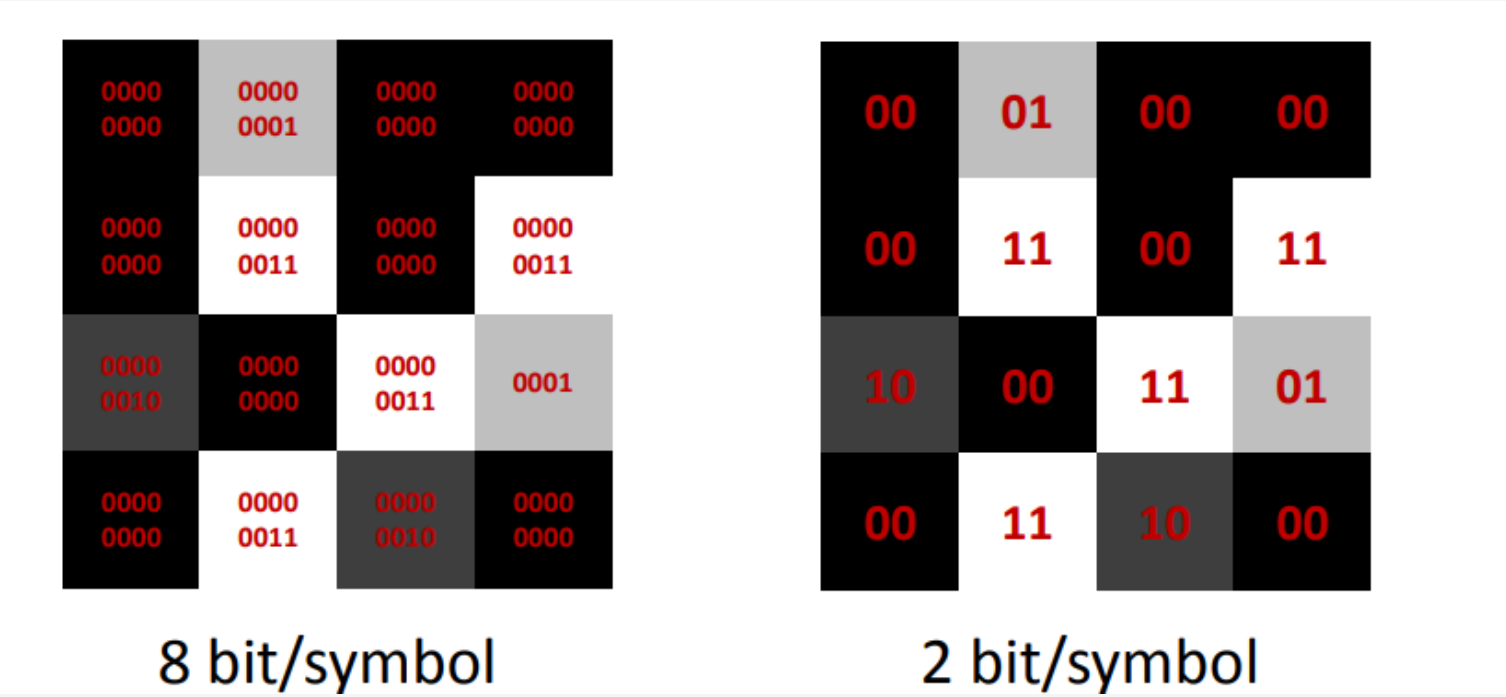
编码冗余与 Fixed-Length Code 固定长度编码

- 采用8 bit 表示灰度，将右图编码成二进制。





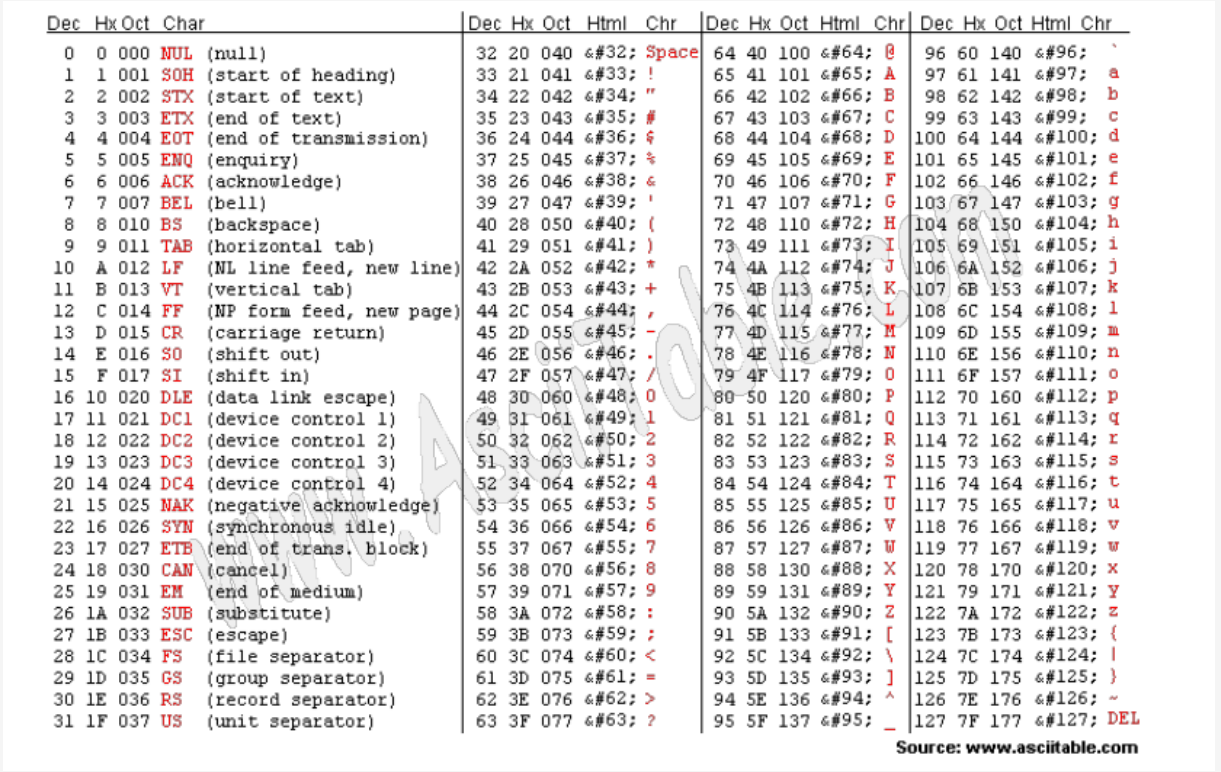
Fixed-Length Code 固定长度编码



- 文件大小?
- CR ?

Fixed-Length Code 固定长度编码

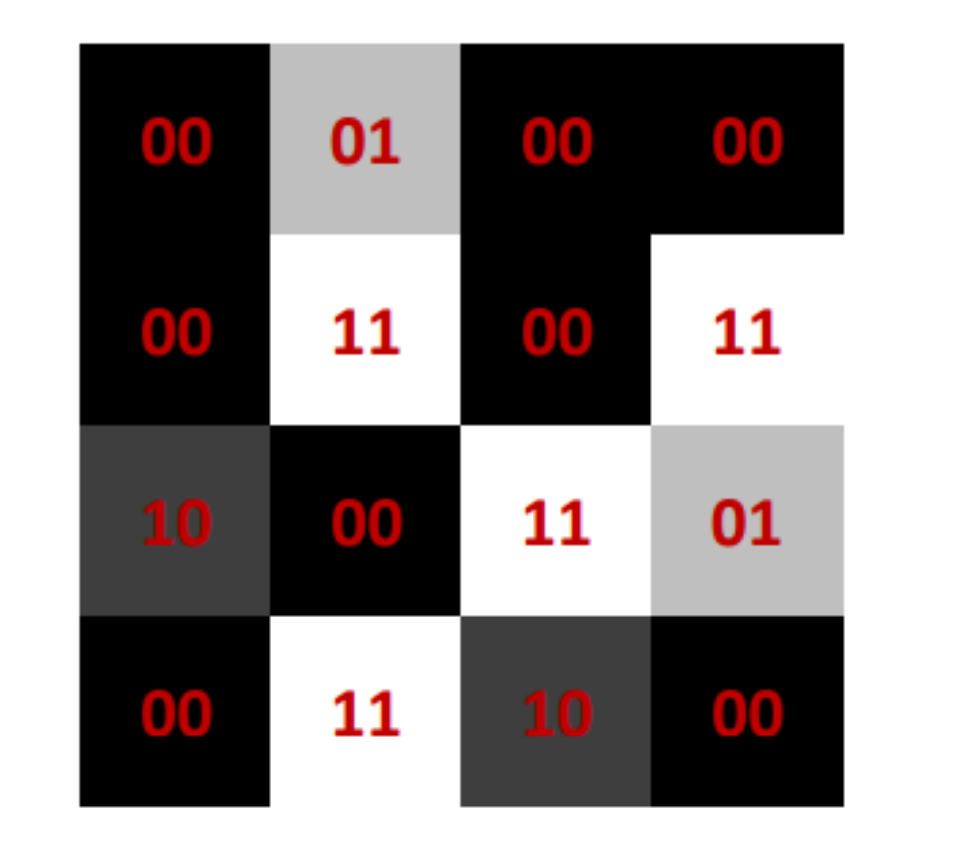
- 特征
  - 使用了相同长度的 bit 数量来表示所有可能的符号
  - 编码和解码过程简单
- 应用：
  - American Standard Code for Information Interchange (ASCII) code
  - 条形码 Universal Product Code (UPC) on products in stores



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.asciitable.com

Fixed-Length Code 固定长度编码



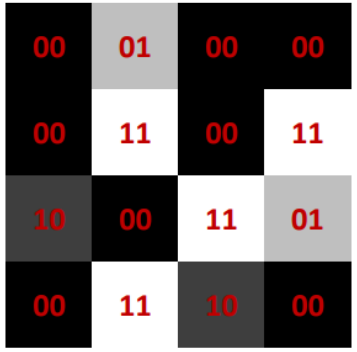
- 是否还有优化的空间?

Run-length coding 行程长度编码

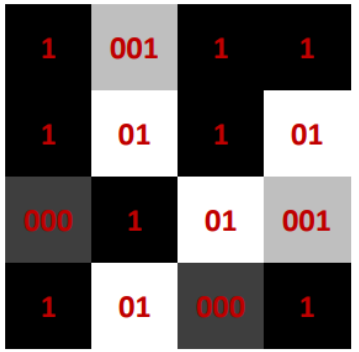
- 举例来说，一组字符串“AAAABBBCCDEEEE”，由4个A、3个B、2个C、1个D、4个E组成，经过行程长度编码可将字符串压缩为4A3B2C1D4E(由14个单位转成10个单位)。

Variable-Length Code 变长编码

- 变长编码的步骤(VLC)
  - 使用不同长度的码字进行编码
  - 给频率最高的符号赋值最短的码字
  - 给频率最低的符号赋值最长的码字
  - 压缩更加有效
- 主流的变长编码包括：
  - Morse code
  - Shannon-Fano code
  - Huffman code



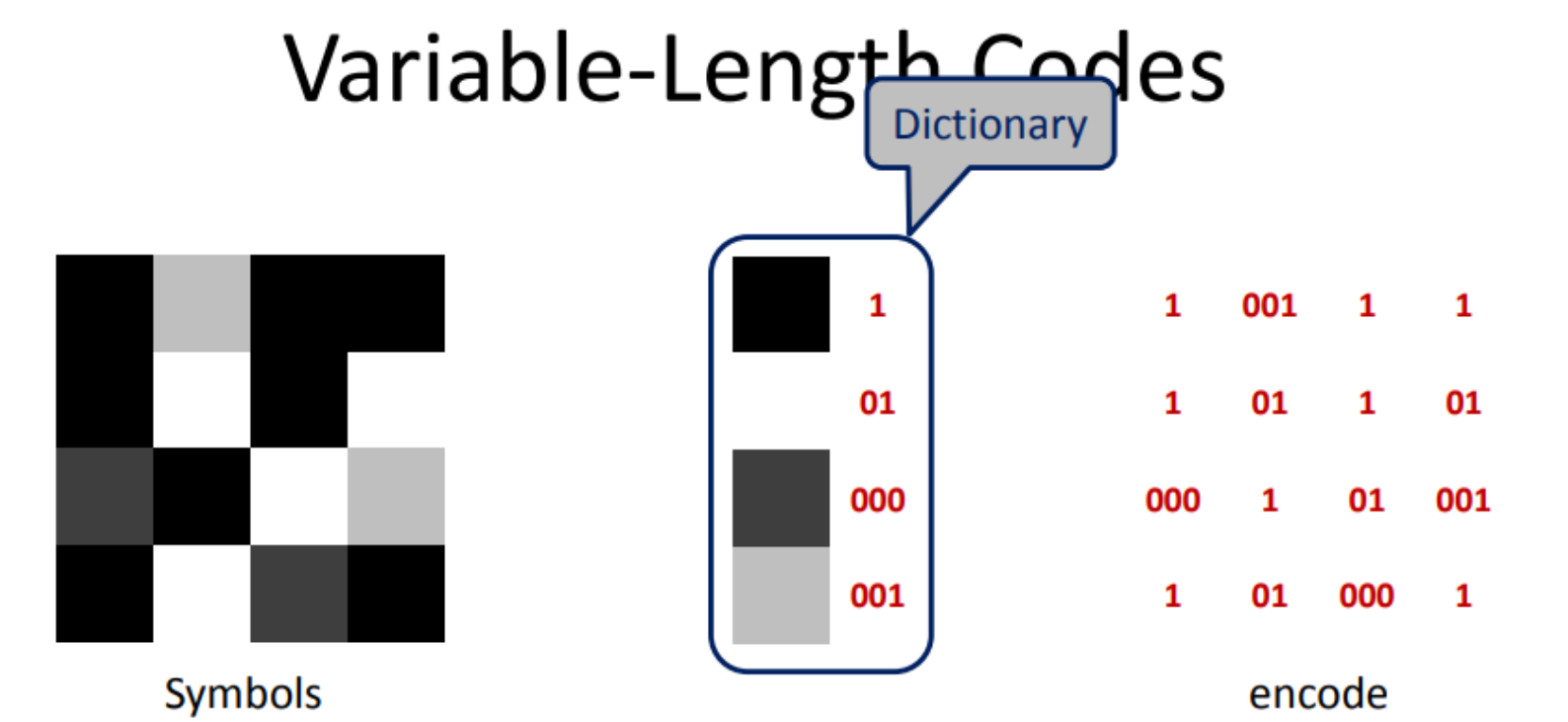
2 bit/symbol



(28/16) bit/symbol

$$L_{avg} = \frac{8 * 1 + 4 * 2 + 4 * 3}{16} = \frac{28}{16}\ bit/symbol$$

Variable-Length Code 的编码过程



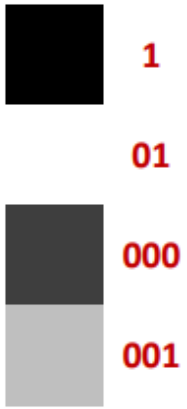
- 根据Symbol的发生频率，建立Symbol-codeword 的字典。
- 根据字典，将图像编码成二进制数据

Variable-Length Code 的解码过程

Decoding VLC

- Suppose you received a file with these data (and with the dictionary):

1 1 0 0 0 1 0 0 1 0 1 1 0 1 1 1 0 1 0 0 0 1 0 1 0 0 1 1

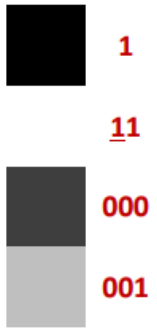


- To generate the information (decode)…

1 1 000 1 001 01 1 01 1 1 01 000 1 01 001 1

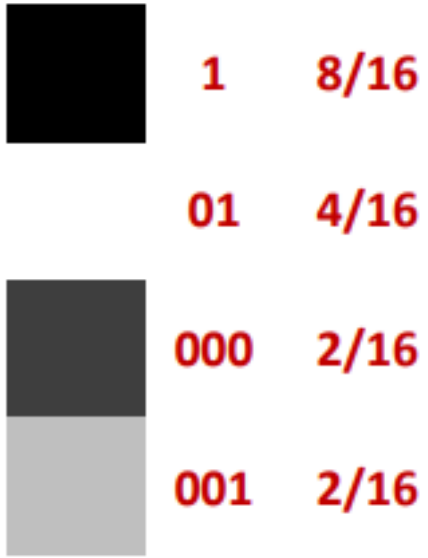
Variable-Length Code 的解码过程

Try decoding this string



- try it ..

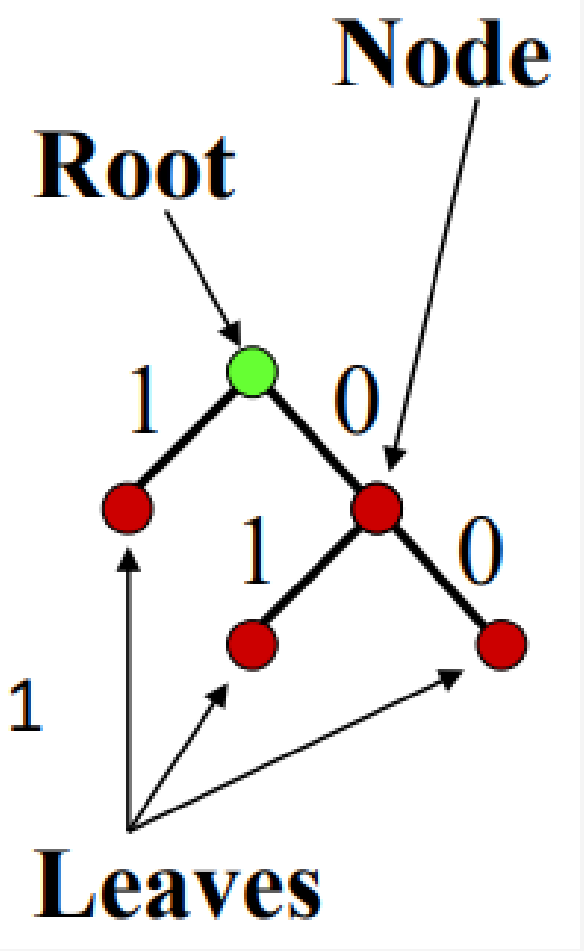
1 1 0 0 0 1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 0 0 1 1



- 前缀码是唯一可译的。
- 前缀码的判断规则，任意码字都不是其他字符编码的前缀。

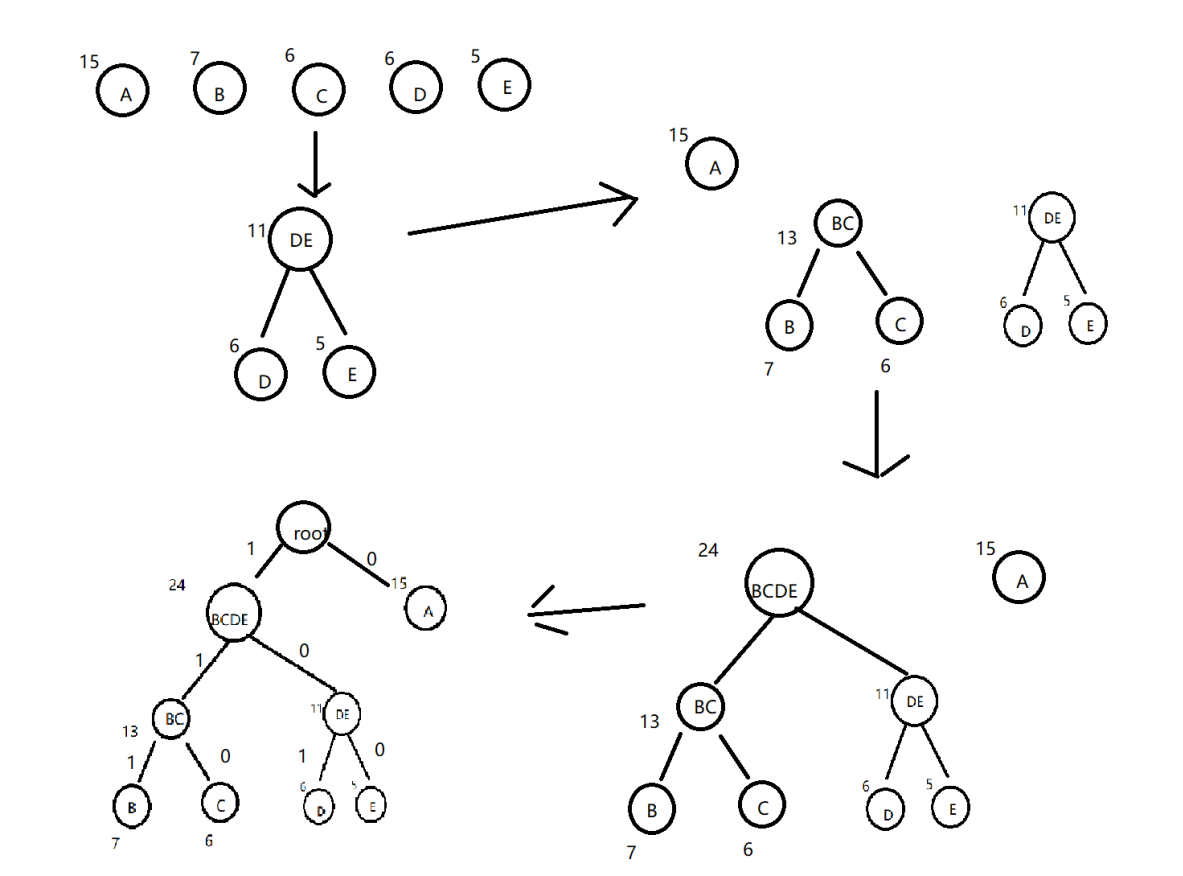
Huffman coding算法

- 霍夫曼编码（Huffman coding）是一种可变长的前缀码。
- 霍夫曼编码的算法过程：
  1. 根据符号出现的概率降序排序
  2. 从频率低的两个符号 x,y 开始，给最低的概率符号赋值0，次低的赋值1。
  3. 合并刚才的两个符号 x,y 组成新符号{x,y}
  4. 带着{x,y}以及加起来的概率，重新排序
  5. 重复上述操作，直至排序列表中只有两个符号。



Huffman coding举例

- 假设我们有A,B,C,D,E字符，频率依次是15，7，6，6，5。



Huffman code: A – 0 ; B – 111; C - 110;

D-101;E-100.

$$L_{avg}$$

$$\approx 1 * 0.385 + 3 * 0.1795 + 3 * 0.1538$$

$$+ 3 * 0.1538 + 3 * 0.1282$$

$$= 2.231\ bit/symbol$$

$$\text{entropy} = 2.152\ \text{bit}$$

Huffman coding例题

- 对下列图像进行Huffman编码，并计算平均编码长度和信息熵。

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 7 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 7 & 5 & 5 & 5 & 5 & 2 & 2 \\ 0 & 7 & 0 & 0 & 0 & 0 & 2 & 2 \\ 0 & 7 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

符号 Symbols	频次
0	32
2	16
1	8
5	4
7	4

无损压缩的下限

- 无损压缩的平均编码长度 ≥ 信息熵
- 编码效率 Coding efficiency公式定义如下：

$$\eta = \frac{H(x)}{L_{avg}}, \quad 0 < \eta \leq 1$$

显然， η 越大，效率越高。η 越小，效率越低。

$$x_i = p_i, y_i = \frac{r^{-l_i}}{K}$$

$$\begin{aligned} H_r(\mathcal{S}) &= \sum_{i=1}^{q} p_i \log_r \frac{1}{p_i} \\ &\leq \sum_{i=1}^{q} p_i \log_r (r^{l_i} K) \\ &= \sum_{i=1}^{q} p_i (l_i + \log_r K) \leq L(\mathcal{C}) \end{aligned}$$

Shannon 第一定律的数学证明

A Mathematical Theory of Communication

像素间冗余与DPCM压缩算法

- 如果一个像素能够合理地从它周围地像素中预测得到，则称该图像存在像素间冗余（Interpixel redundancy）因为多数二维阵列的像素是空间相关的（每个像素取决于向量像素）。在相关像素的表示中，信息没必要重复。
- 像素间冗余取决于图像的分辨率
  - 分辨率越高，像素间冗余越多
  - 分辨率越低，像素间冗余越少

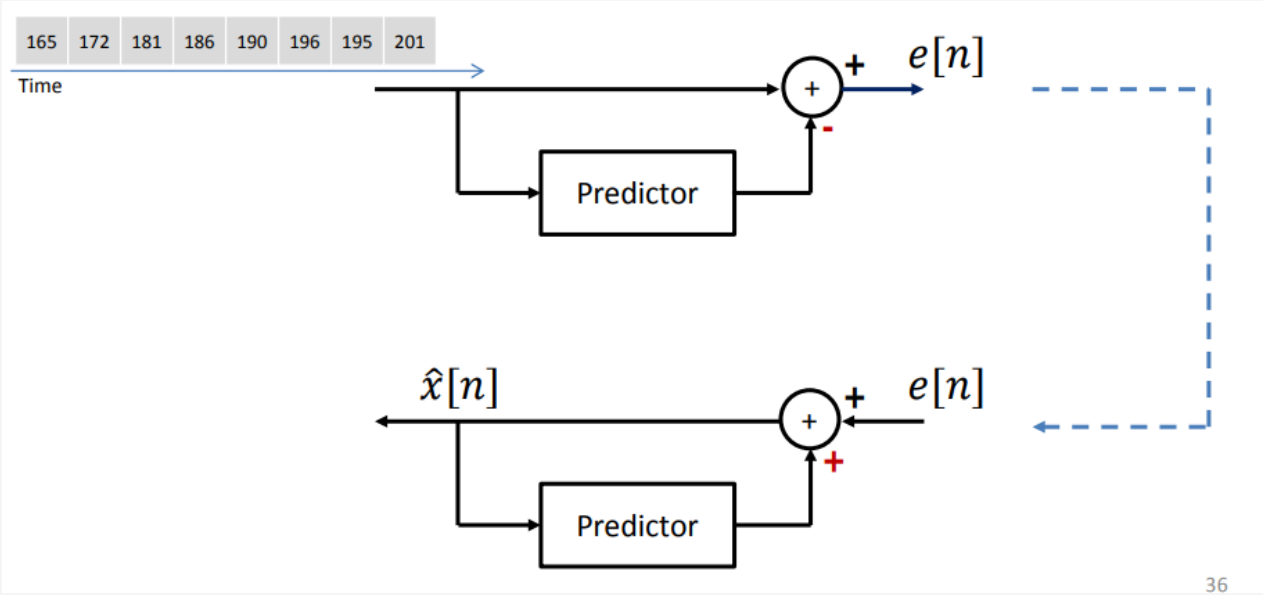
- Example

Images are smooth

165	172	181	186	190	196	195	201
169	176	184	187	192	193	194	195
169	173	182	187	190	193	189	190
173	177	182	185	191	189	189	188
168	173	179	182	189	187	188	190
169	170	175	180	183	184	185	189
166	169	173	176	181	180	186	184
171	168	167	176	176	180	177	181

像素间冗余与DPCM压缩算法

- 差分预测编码调制 (Differential Pulse code modulation，DPCM)能够充分利用像素间的冗余来对图像进行压缩，既可以是无损压缩也可以是有损压缩。(DPCM+量化+Huffman Coding)
- 差分预测编码器原理如右图所示：
  - $[x_0, x_1, x_2, \dots, x_n]$ 是原序列, 通过编码器得到 $e[n] = [x_1 - x_0, x_2 - x_1, \dots, x_n - x_{n-1}]$, 并将$x_0$ 与e$[n]$ 一并保存或发生。
  - 解码器，通过$x_0, e[n]$ 复原信号。



差分预测编码调制
(Differential Pulse code modulation，DPCM)

DPCM压缩算法的有效性

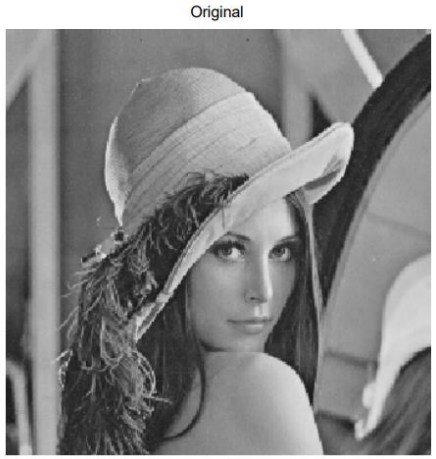
- Entropy(原序列) = 3.0 bit
- Entropy(现序列， $[x_0, e[n]]$) = 2.75 bit
- 信息熵变小了！
- 后续我们还可以对$e[n]$再进行变长编码（Huffman coding）
- 再举个例子 [3,4,5,6,7,8,9…,100]，输出序列多少？信息熵前后多少？
- DPCM算法为什么有效？

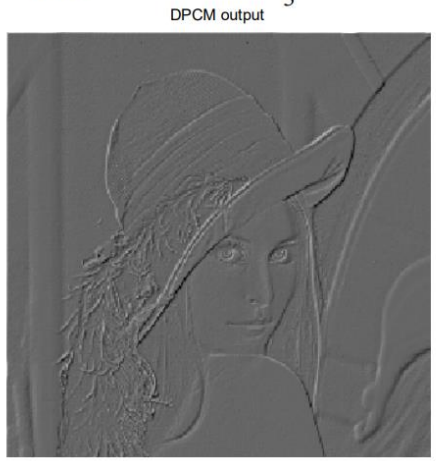
- Entropy(原序列) = 3.0 bit
- Entropy(现序列， $[x_0, e[n]]$) = 2.75 bit
- 信息熵变小了！
- 后续我们还可以对$e[n]$再进行变长编码（Huffman coding）
- 再举个例子 [3,4,5,6,7,8,9…,100]，输出序列多少？信息熵前后多少？
- DPCM算法为什么有效？

差分预测编码调制
(Differential Pulse code modulation，DPCM)

Raster-scan DPCM coding

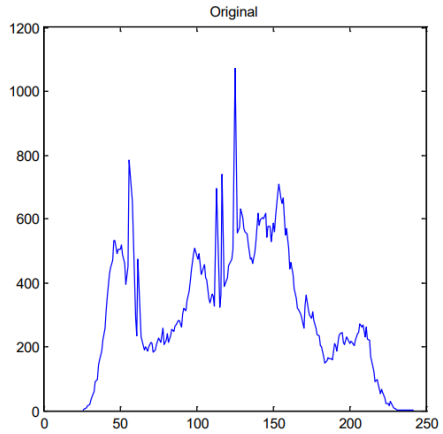
$$e(r,c) = x(r,c) - \frac{x(r,c-1) + x(r-1,c-1) + x(r-1,c)}{3}$$

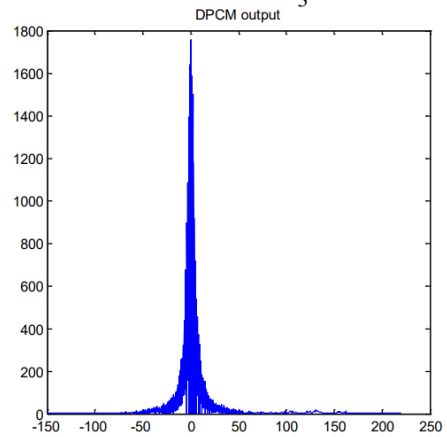




Raster-scan DPCM coding

$$e(r,c) = x(r,c) - \frac{x(r,c-1) + x(r-1,c-1) + x(r-1,c)}{3}$$





心理视觉冗余与有损压缩算法

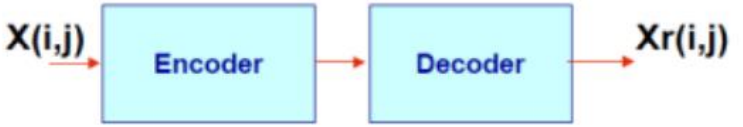
- 人类的视觉系统并不是对所有光的色彩和频率都比较敏感，而且因人而异。我们可以通过减少图像上的一些信息，来达到压缩的目的。比如高频信号。
- 有损压缩算法定义: 无法通过解码得到原信号
  - 优点：更高的压缩率 Compression ratio
  - 缺点：无法通过解码得到原信号，信息有一定损失

- Speech/Audio:
  - GSM/UMTS/WCDMA (multi-rate) speech compression
  - MP3 audio
- Image compression:
  - JPEG
  - JPEG2000
- Video compression
  - H.264/AVC , H.264/SVC
  - H.265 (HEVC)

有损压缩质量的评价指标

- 有损压缩不仅要看压缩效率，还要评价压缩过后的图像质量
- 常见的图像质量评价方式有：
  - 客观数值： SNR, PSNR （越大越好）
  - 主观评价

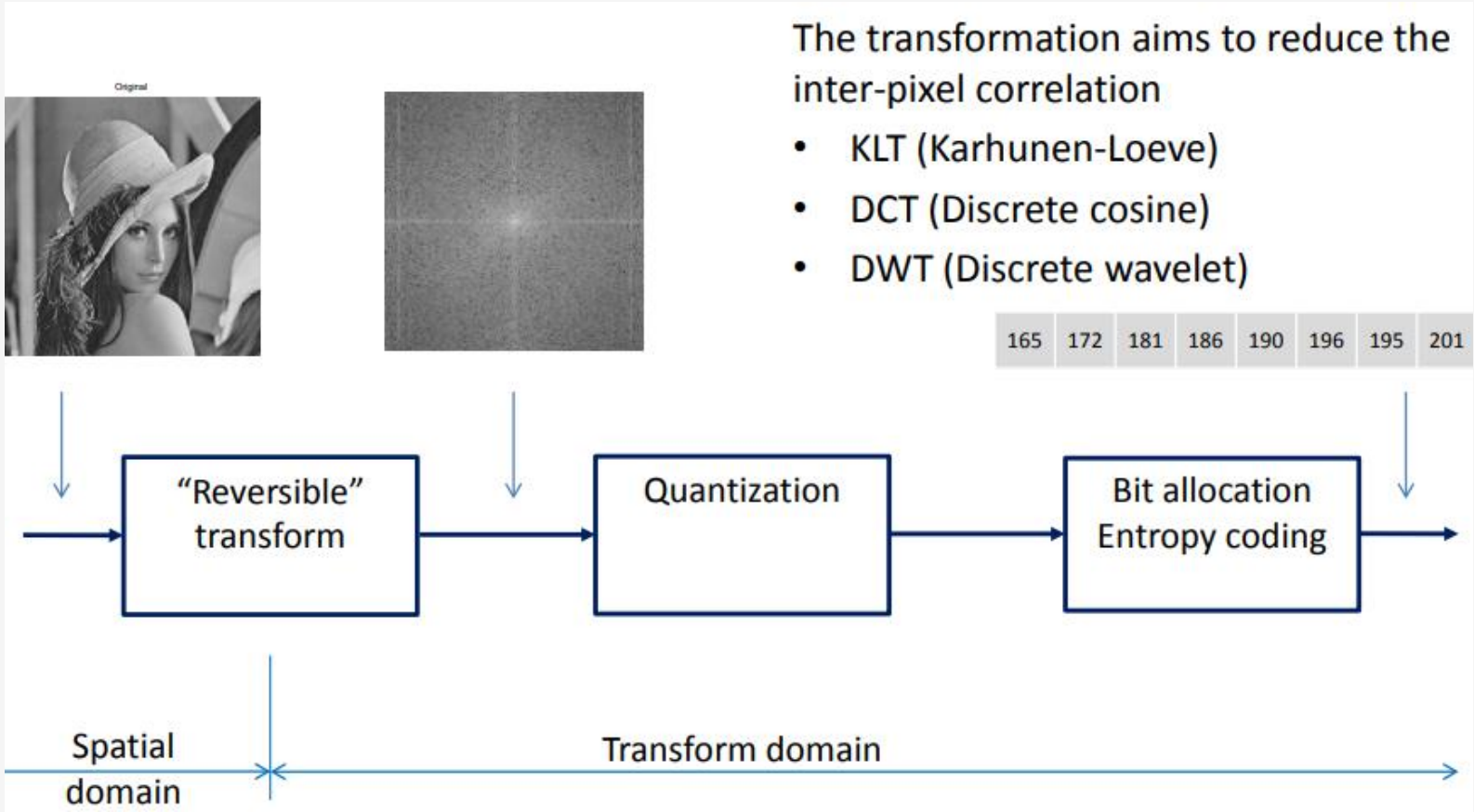
SNR & PSNR



$$SNR = 10\log_{10}\frac{\frac{1}{NM}\sum_{i,j}|X(i,j)|^2}{\frac{1}{NM}\sum_{i,j}|X(i,j)-X_R(i,j)|^2} = 10\log_{10}\frac{\frac{1}{NM}\sum_{i,j}|X(i,j)|^2}{MSE}$$

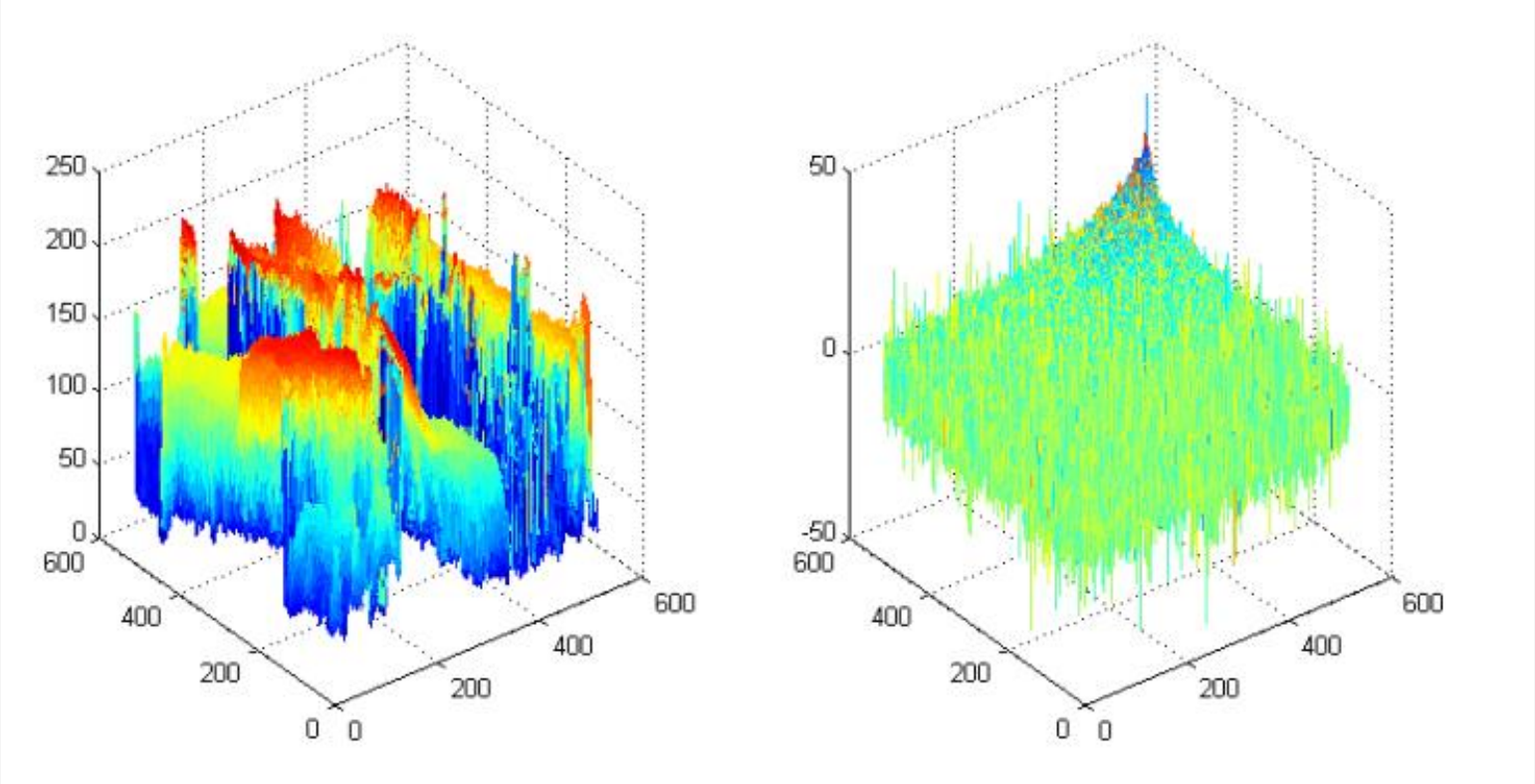
$$PSNR = 10\log_{10}\frac{255^2}{\frac{1}{NM}\sum_{i,j}|X(i,j)-X_R(i,j)|^2}$$

变换域编码 transform coding



- JPEG格式就是采用的变换域压缩编码方式

变换域编码 transform coding 的有效性



- 一幅图像的时域与DCT变换后的频域，可以看到大部分信息存在低频位置。

JPEG格式与背景知识

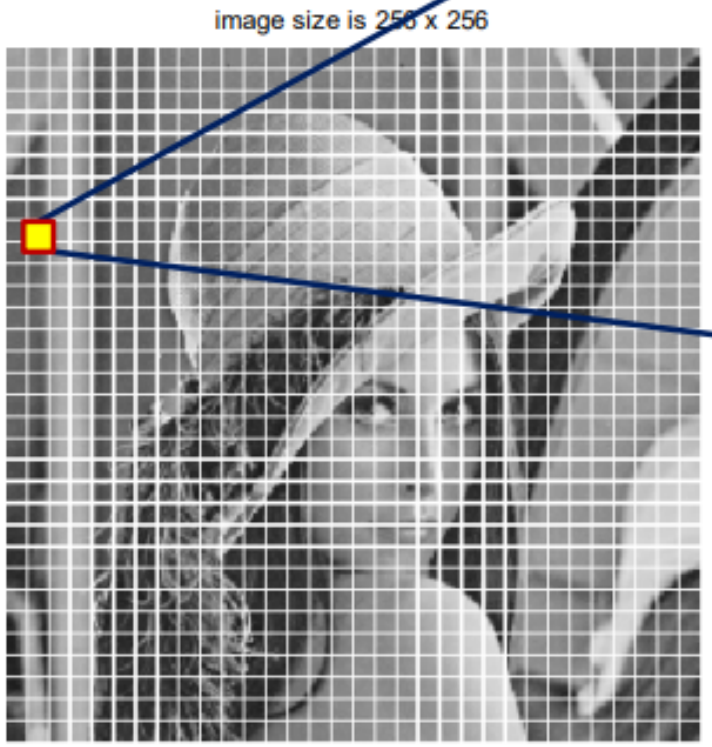
- JPEG（Joint Photographic Experts Group，联合图像专家小组）
- 此团队创立于1986年，其于1992年发布的 JPEG 标准在1994年获得了 ISO 10918-1 的认定，成为了图片压缩标准。

JPEG格式灰度图像的压缩步骤

1. 通过DCT变换将图像转换到频域
2. 量化 Quantization
3. 排列 zig-zag reading
4. 分别对DC 与 AC 信号做无损编码 lossless coding

DCT变换将图像转换到频域

- 8x8 DCT transform



818.0	-46.6	44.4	-26.7	12.0	-3.0	-1.9	-6.0
-9.1	1.7	-5.1	-4.0	3.4	4.5	-2.0	2.2
2.8	-0.7	4.2	-0.3	-3.9	1.3	1.2	-6.7
-2.5	1.2	0.7	-3.7	-3.1	0.2	1.4	-0.2
-3.8	1.7	6.4	-1.0	-4.3	-1.7	4.9	0.2
-1.9	3.0	2.7	-4.5	-3.9	-1.4	-0.9	2.7
-2.5	0.7	3.4	2.3	-4.1	1.8	2.1	0.5
-4.9	3.0	-1.5	1.5	-1.6	-0.8	-0.2	3.4

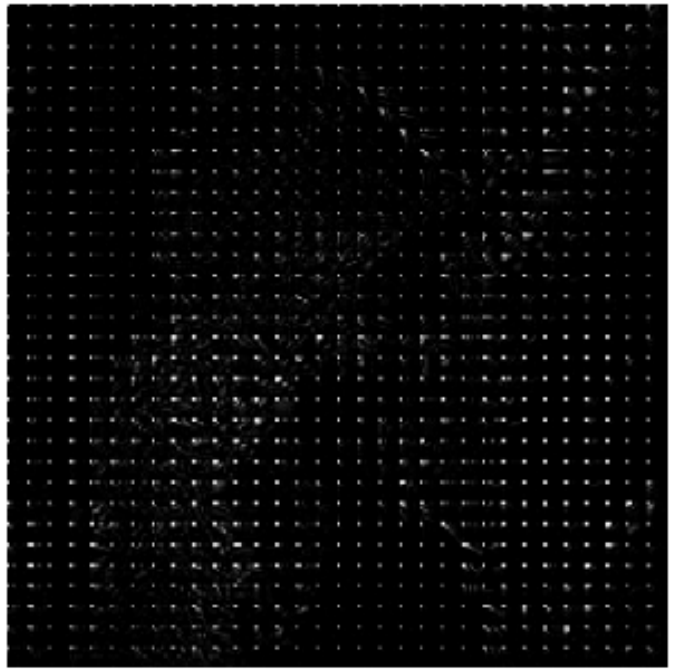
- The DCT coefficients are real valued, so forward and inverse DCT are limited by computer precision
- No truly lossless compression is possible if the non-integer DCT is used

8*8 DCT的特性

- 8x8 DCT transform

This format is not used in JPEG, it is just illustrative, however, it shows that the DC values are still correlated

the 8 x 8 2D DCT of the image (Natural order)

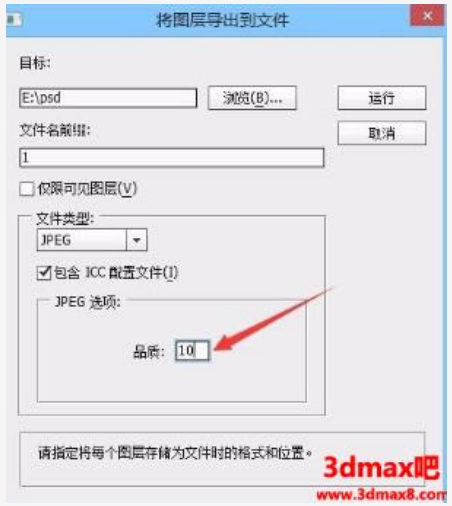


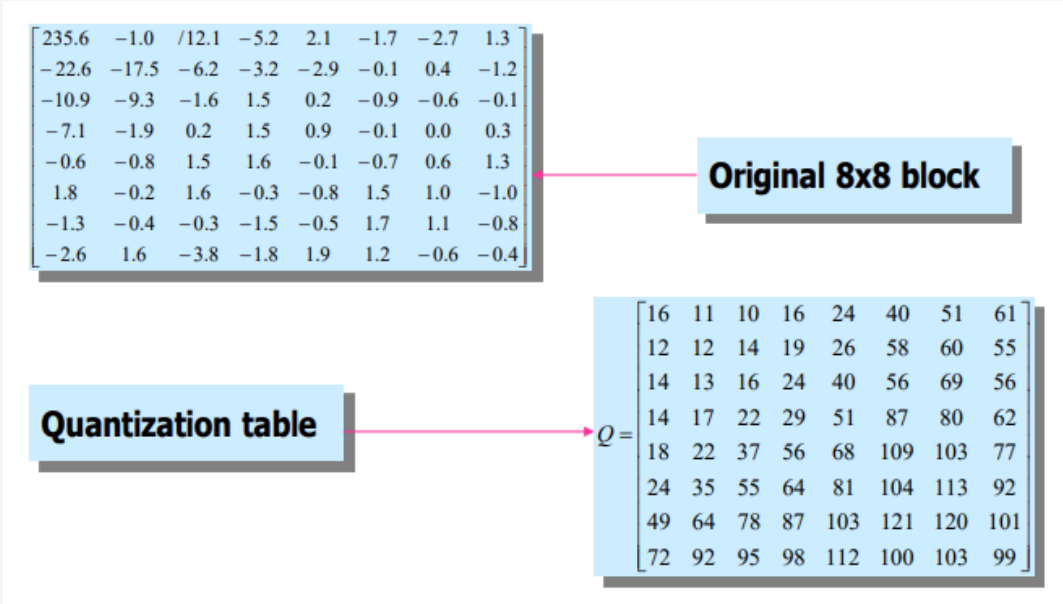
Natural order

Reordered (DC coeff. grouped)

量化 Quantization

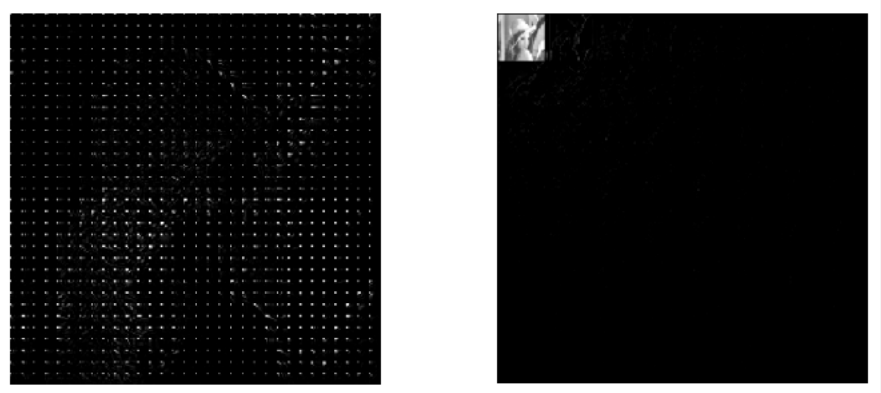
- 量化是 JPEG 压缩的主要有损部分，将8*8DCT块除以量化模板。
- 量化模板的作用是保留低频信息，减小高频特征
- 运算遇到小数时，四舍五入
- 此外，我们还能认为的增加量化模板的权重，称为Quality factor 质量因素。



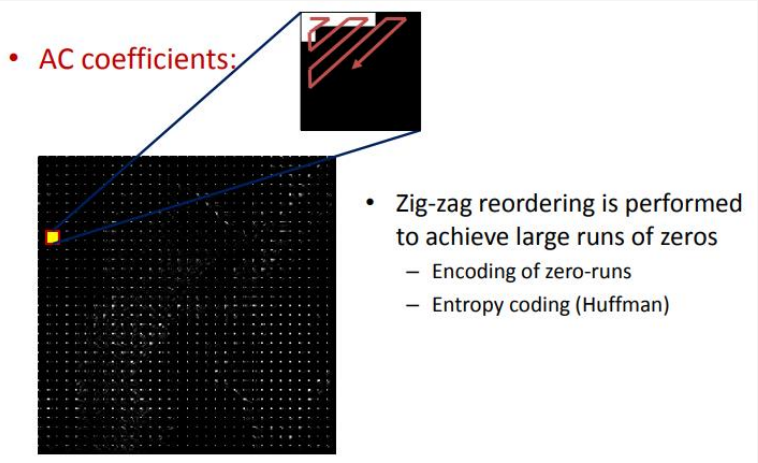


Zig-zag 与 无损编码 lossless coding

- DC （低频部分）部分采用 DPCM 差分预测编码调制压缩，减少像素间的冗余
- 再用Zig-zag将图像展开成一维
- Run-length 编码
- Huffman 编码



DC 低频部分



AC 高频部分

JPEG的解压过程

- Huffman 编码解压
- Run-length 编码解压
- 系数反量化（乘以量化模板）
  得到8*8DCT块
- IDCT 反变换得到空间域图像

JPEG的图像质量

Quality max - Size: 61k

Quality med - Size: 14k

Quality low - Size: 4k

JPEG的图像质量

Quality 95/100
3.926 bits per pixel (bpp)
CR = 24/3.926 = 6.1

Quality 25/100
0.705 bits per pixel (bpp)
CR = 34.0

Quality 50/100
1.067 bits per pixel (bpp)
CR = 22.5

Quality 5/100 (min.useful)
0.291 bits per pixel (bpp)
CR = 82.5

THANKS,希望大家实验课玩的愉快！