Matlab图像处理编程实践初步

无损图像压缩

肖俊

浙江大学计算机学院

2025

内容提要

- 信息论
- 差分编码
- 无损图像压缩

1.3 Basics of Information Theory

- The entropy of an information source
  - With alphabet S = {s1, s2, …, sn}
  - $\eta = H(S) = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i} = -\sum_{i=1}^{n} p_i \log_2 p_i$
  - p_i is the probability that symbol s_i in S will occur

- $\log_2 {}^{1}\!/_{P_i}$ indicate the amount of information contained in characters (Self-information)

1.3 Basics of Information Theory

- For example： the probability of ***n*** in a manuscript is 1/32， so
  - The amount of information is 5 bits
  - A character sting nnn require 15 bits to code

- What is entropy?
  - A measure of the disorder of a system
  - The more entropy, the more disorder

1.3 Basics of Information Theory

- Examples:
  - Suppose a system has 4 states outcome, each outcome has probability 1/4 :

$$4\times\frac{1}{4}\times\log_2 1/\frac{1}{4}=2bits$$

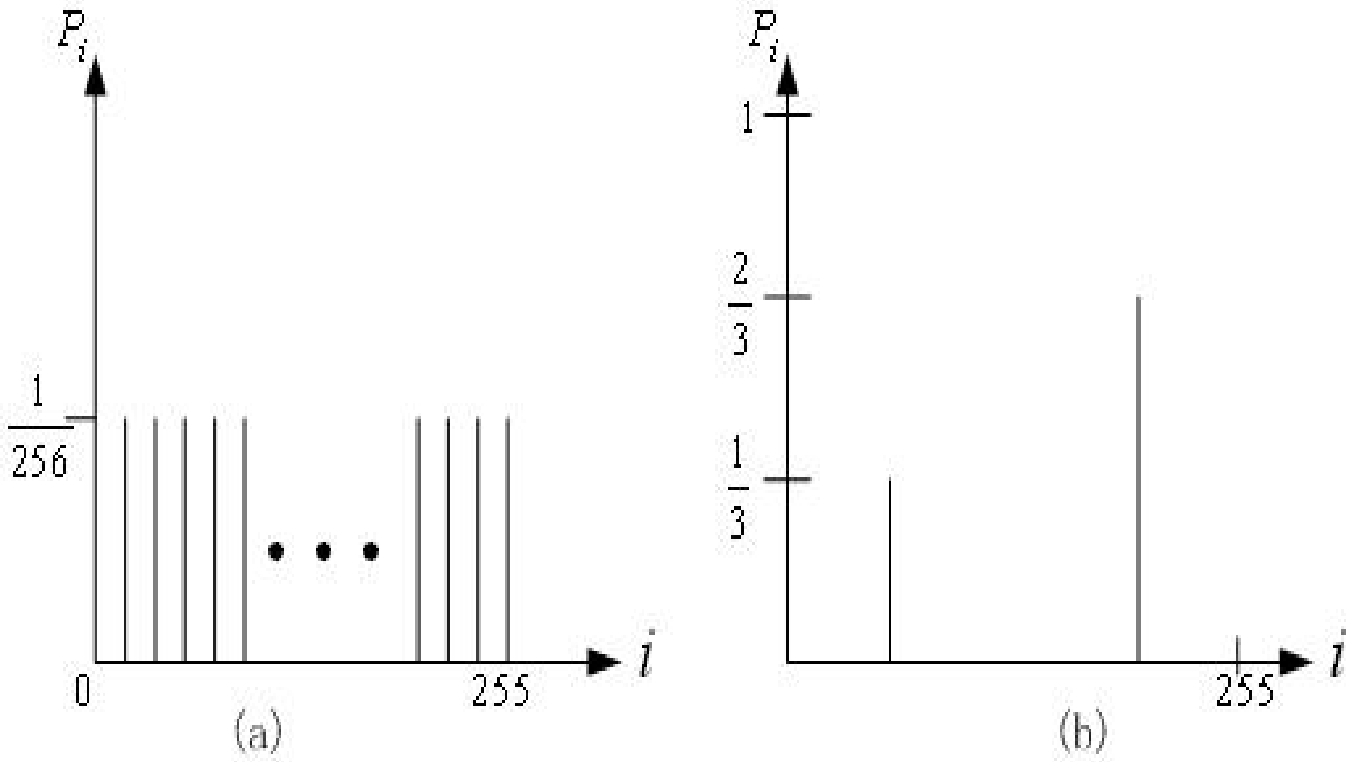
  - If one state had probability ½,the other three had probability 1/6:

$$\frac{1}{2}\times\log_2^2+3\times\frac{1}{6}\times\log_2^6=1.795<2bits$$

  - The most-occurring one means fewer bits to send

- The definition of entropy -- identifying often-occurring symbols as short *codewords*
  - Variable-Length coding

1.3 Basics of Information Theory



Histograms for two gray-level images

1.3 Basics of Information Theory

- The entropy of the two above images
  - The entropy of image a is:
  -
  - $\eta = \sum_{i=0}^{255} \frac{1}{256} \cdot \log_2^{256} = 8$
  - The entropy of image b is:

$$\eta = \frac{1}{3} \cdot \log_2^{\frac{3}{2}} + \frac{2}{3} \cdot \log_2^{\frac{3}{2}}$$

$$= 0.33 \times 1.59 + 0.67 \times 0.59 = 0.92$$

- The entropy is greater when the probability is flat and smaller when it is more peaked.

1.3 Basics of Information Theory

- As can be seen in Eq. (7.3): the entropy η is a weighted-sum of terms $\log_2 \frac{1}{p_i}$; hence it represents the *average* amount of information contained per symbol in the source S.

- The entropy η specifies the lower bound for the average number of bits to code each symbol in S, i.e.,

$$\eta \leq \bar{l} \tag{7.5}$$

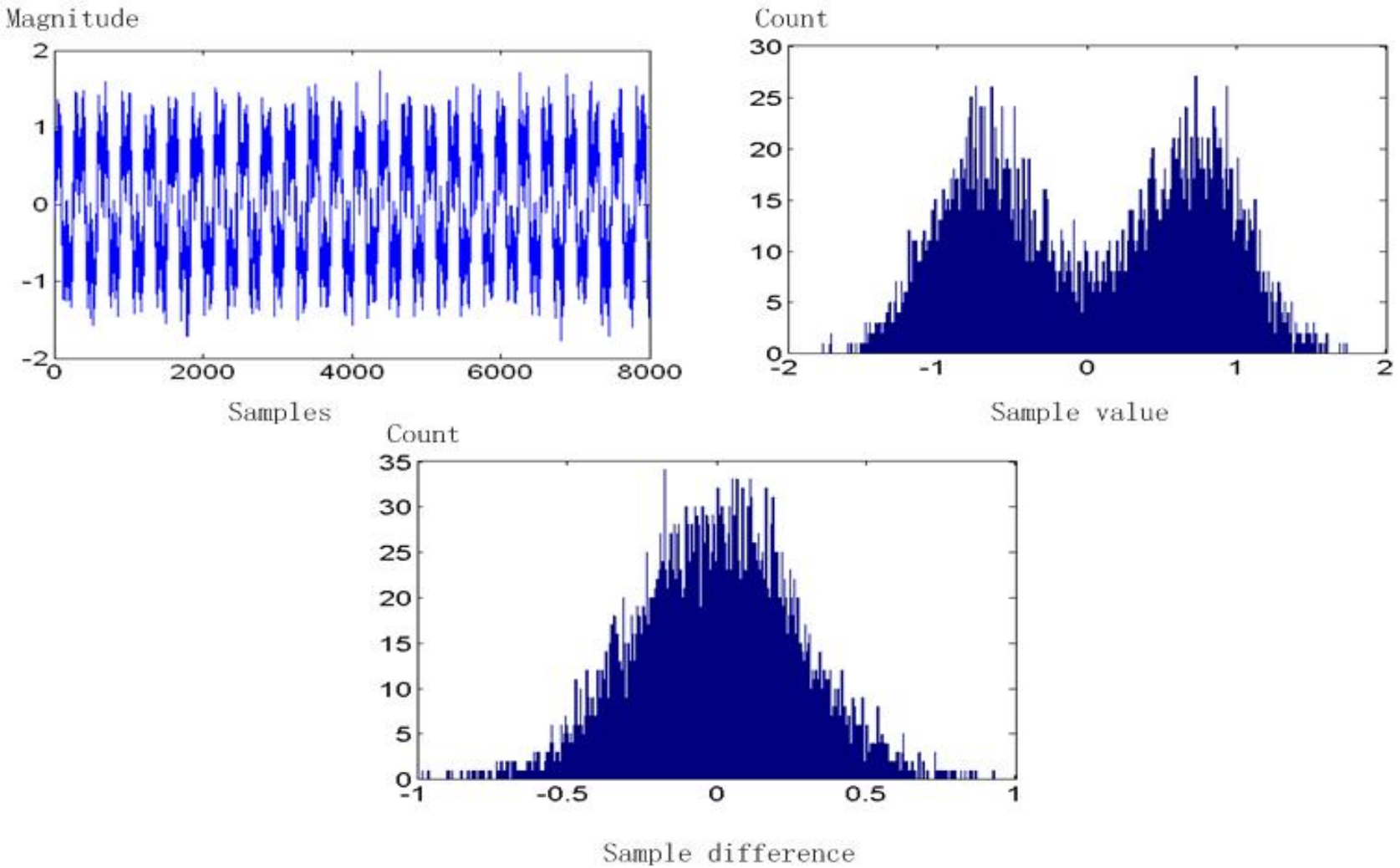
$\bar{l}$ - the average length (measured in bits) of the codewords produced by the encoder.

3.3 Differential coding of audio

- Audio is often stored not in simple PCM but instead in a form that exploits differences — which are generally smaller numbers, so offer the possibility of using fewer bits to store.

- If a time-dependent signal has some consistency over time ("temporal redundancy"), the difference signal, subtracting the current sample from the previous one, will have a more peaked histogram, with a maximum around zero.

3.3 Differential coding of audio



3.4 Lossless Predictive Coding

- Predictive coding: simply means transmitting differences — predict the next sample as being equal to the current sample; send not the sample itself but the difference between previous and next.

  (a) Predictive coding consists of finding differences, and transmitting these using a PCM system.

  (b) Note that differences of integers will be integers. Denote the integer input signal as the set of values f_n. Then we predict values $\hat{f}_n$ as simply the previous value, and define the error e_n as the difference between the actual and the predicted signal:

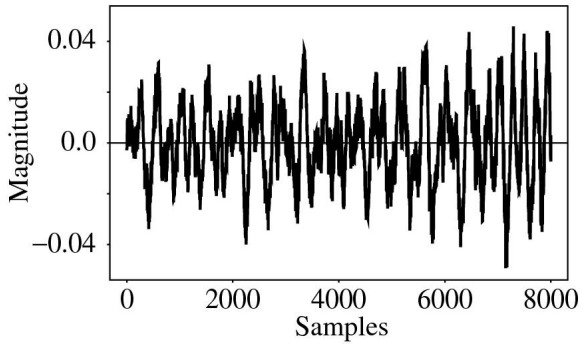
$$\hat{f}_n = f_{n-1} \qquad e_n = f_n - \hat{f}_n \tag{6.12}$$

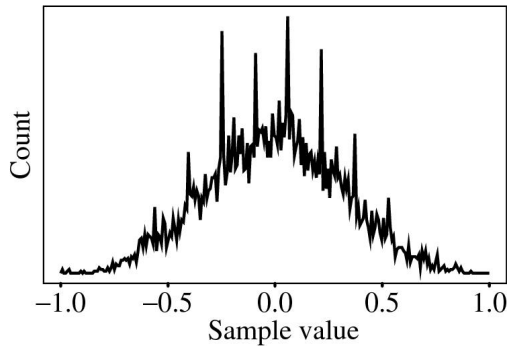
3.4 Lossless Predictive Coding

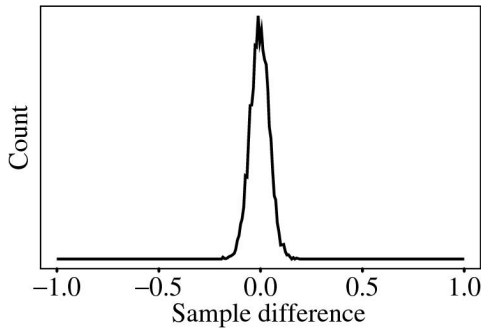
(c) But it is often the case that some function of a few of the previous values, fn−1, fn−2, fn−3, etc., provides a better prediction. Typically, a linear predictor function is used:

$$\hat{f}_n = \sum_{k=1}^{2to4} a_{n-k} f_{n-k}$$

The idea of forming differences is to make the histogram of sample values more peaked.







3.4 Lossless Predictive Coding

- One problem: suppose our integer sample values are in the range 0..255. Then differences could be as much as -255..255 —we've increased our dynamic range (ratio of maximum to minimum) by a factor of two → need more bits to transmit some differences.
  - (a) A clever solution for this: define two new codes, denoted SU and SD, standing for Shift-Up and Shift-Down. Some special code values will be reserved for these.
  - (b) Then we can use codewords for only a limited set of signal differences, say only the range −15..16. Differences which lie in the limited range can be coded as is, but with the extra two values for SU, SD, a value outside the range −15..16 can be transmitted as a series of shifts, followed by a value that is indeed inside the range −15..16.
  - (c) For example, 100 is transmitted as: SU, SU, SU, 4, where (the codes for) SU and for 4 are what are transmitted (or stored).

3.4 Lossless Predictive Coding

- Lossless predictive coding — the decoder produces the same signals as the original. As a simple example, suppose we devise a predictor for $\hat{f}_n$ as follows:

$$\hat{f}_n = \left\lfloor \frac{1}{2}(f_{n-1} + f_{n-2}) \right\rfloor$$

$$e_n = f_n - \hat{f}_n$$

3.4 Lossless Predictive Coding

- Let's consider an explicit example. Suppose we wish to code the sequence f1, f2, f3, f4, f5 = 21, 22, 27, 25, 22. For the purposes of the predictor, we'll invent an extra signal value f0, equal to f1 = 21, and first transmit this initial value, uncoded:

$$\hat{f}_2 = 21, e_2 = 22 - 21 = 1;$$

$$\hat{f}_3 = \left\lfloor \frac{1}{2}(f_2 + f_1) \right\rfloor = \left\lfloor \frac{1}{2}(22 + 21) \right\rfloor = 21,$$

$$e_3 = 27 - 21 = 6;$$

$$\hat{f}_4 = \left\lfloor \frac{1}{2}(f_3 + f_2) \right\rfloor = \left\lfloor \frac{1}{2}(27 + 22) \right\rfloor = 24,$$

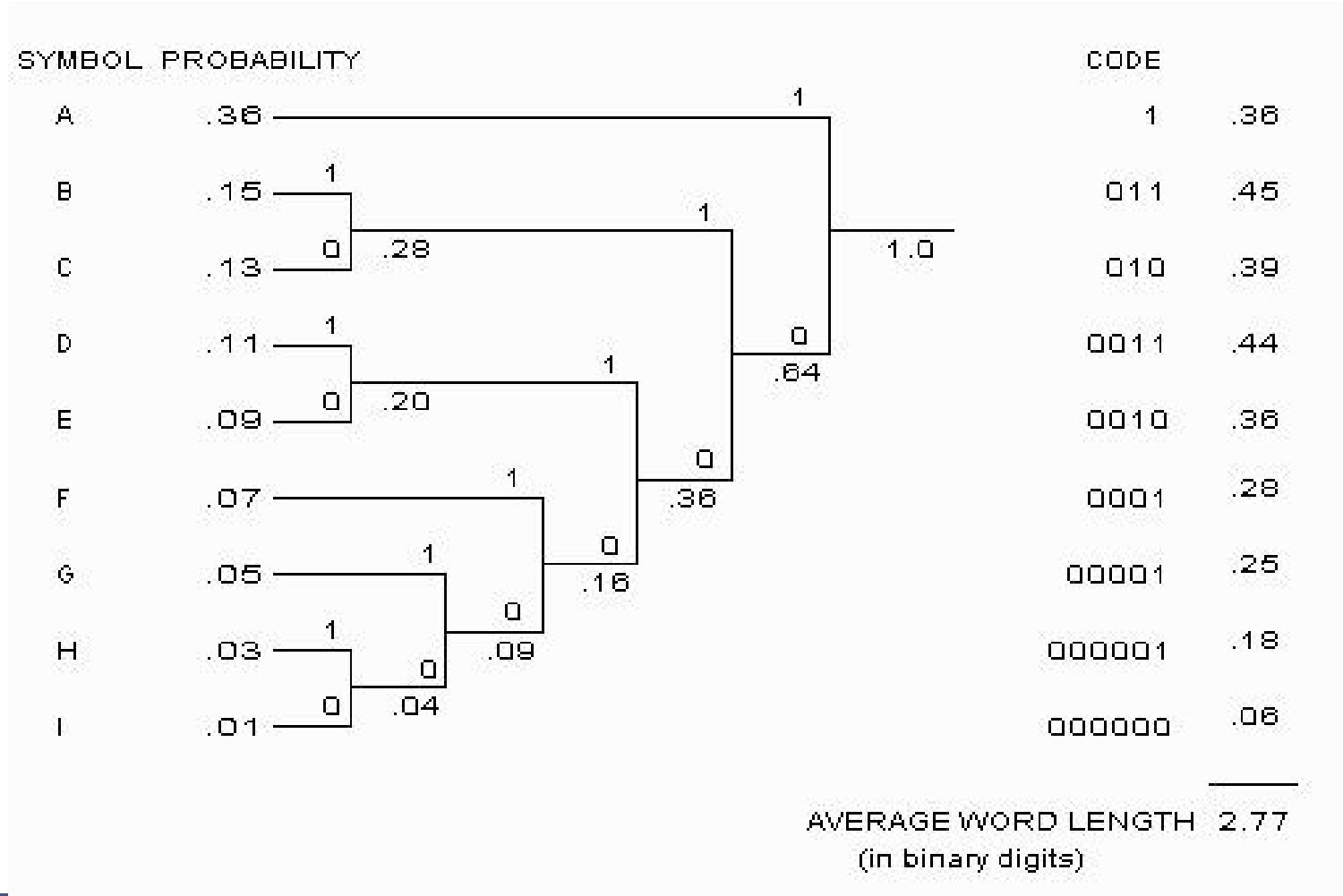
$$e_4 = 25 - 24 = 1;$$

$$\hat{f}_5 = \left\lfloor \frac{1}{2}(f_4 + f_3) \right\rfloor = \left\lfloor \frac{1}{2}(25 + 27) \right\rfloor = 26,$$

$$e_5 = 22 - 26 = -4$$

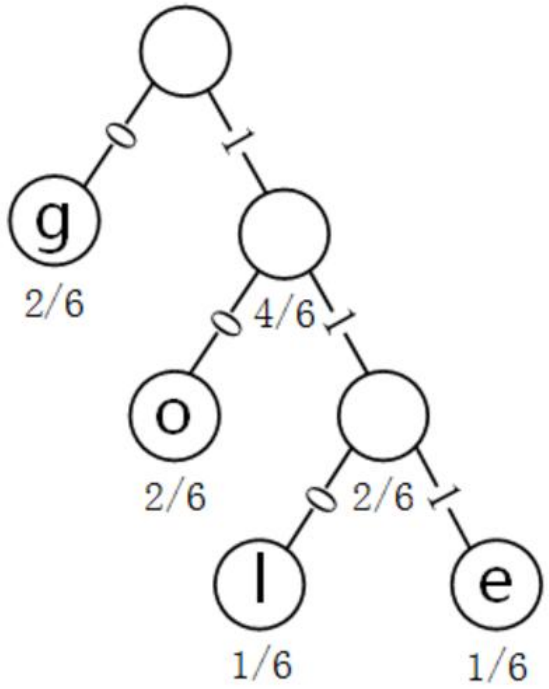
4.2 可变长编码

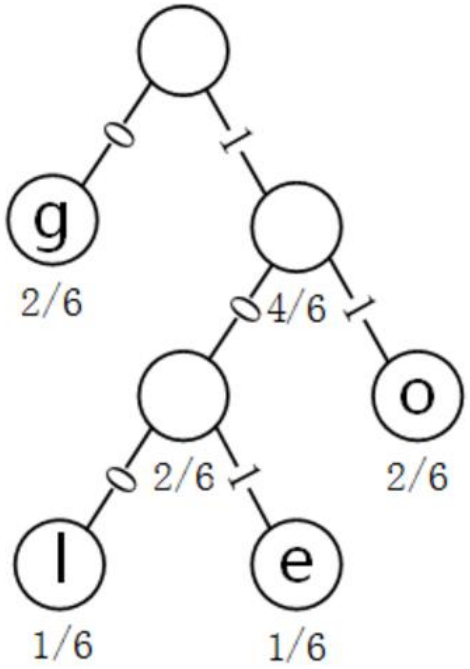
- Huffman编码（最典型的熵编码）
  - David A. Huffman在1952年提出



Huffman编码可能不唯一

Huffman编码存在不唯一的问题。举个例子，对于字符串`google`进行Huffman编码可以是如下：






```
g : 0
o : 1 0
l : 1 1 0
e : 1 1 1
```

```
g : 0
o : 1 1
l : 1 0 0
e : 1 0 1
```

利用Matlab自带函数对字符串进行Huffman编码

```
3  str = 'You have to believe in yourself. That is the secret of success.';
4
5  %根据字符串str得到符号集symbols，并计算各集合元素的出现概率数组p
6  len = length(str);
7  unique_str = unique(str);
8  unique_len = length(unique_str);
9
10 symbols = cell(1, unique_len);
11 p = zeros(1, unique_len);
12
13 for i = 1:unique_len
14     symbols{1,i} = unique_str(i);
15     p(i) = numel(find(str==unique_str(i))) / len;
16 end
17
18 %根据符号集symbols和概率数组p计算Huffman编码词典
19 [dict, avglen] = huffmandict(symbols, p);
```

利用Matlab自带函数对字符串进行Huffman编码

```
` ` : 0 0 0
`.` : 1 0 0 1 0
`T` : 1 0 0 1 1 1
`Y` : 1 0 0 1 1 0
`a` : 0 1 1 0 1
`b` : 0 1 0 0 0 0 1
`c` : 1 1 0 1
`e` : 0 0 1
`f` : 0 1 1 0 0
`h` : 1 1 0 0
`i` : 1 1 1 1
`l` : 0 1 1 1 1
`n` : 0 1 0 0 0 0 0
`o` : 1 0 0 0
`r` : 0 1 1 1 0
`s` : 1 0 1
`t` : 0 1 0 1
`u` : 1 1 1 0
`v` : 0 1 0 0 1
`y` : 0 1 0 0 0 1
平均码长 : 3.968254
信源熵 : 3.927877
编码效率 : 0.254590
编码前字符串总长度 : 63
编码后字符串二进制总长度 : 250
编码后字符串字节总长度(250/8) : 32
编码结果 : 100110 1000 1110 000 1100 01101 01001 001 000 0101 1000 000 0100001 001 01111 1111 001 0100
```

利用Matlab自带函数对字符串进行Huffman编码

https://blog.csdn.net/solomonlangrui/article/details/52930245

3.2 Differential Coding of Images

Given an original image I (x, y), defining a difference image d (x, y)

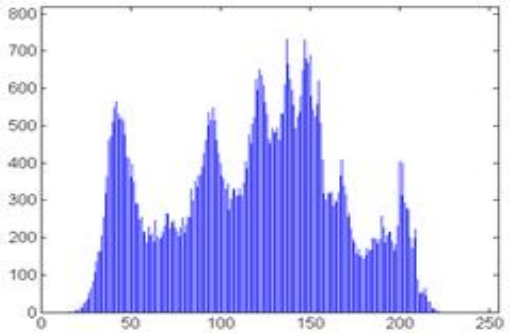
- Using a simple difference operator
  - d(x,y)=I(x,y)-I(x-1,y)
- Discrete 2D Laplacian operator
  - d(x,y)=4I(x,y)-I(x,y-1)-I(x,y+1)-I(x+1,y)-I(x-1,y)

- Due to *spatial redundancy* existed in normal images I, the difference image d will have a narrower histogram and hence a smaller entropy
  - VLC -- shorter bit-length for the difference image
  - Compression works better on a difference image

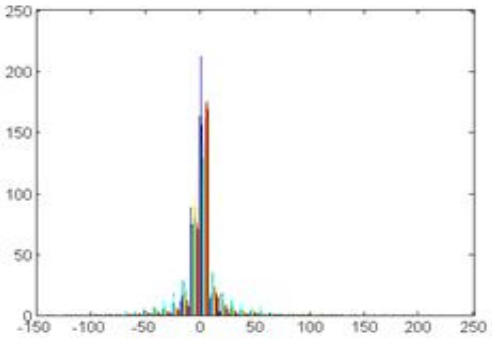
3.2 Differential Coding of Images

(a)

(b)



(c)



(d)

3.3 Lossless JPEG

- **Lossless JPEG**: A special case of the JPEG image compression.

- **The Predictive method**

  1. **Forming a differential prediction**: A predictor combines the values of up to three neighboring pixels as the predicted value for the current pixel, indicated by 'X' in Fig. 7.10. The predictor can use any one of the seven schemes listed in Table 7.6.

  2. **Encoding**: The encoder compares the prediction with the actual pixel value at the position 'X' and encodes the difference using one of the lossless compression techniques we have discussed, e.g., the Huffman coding scheme.

3.3 Lossless JPEG

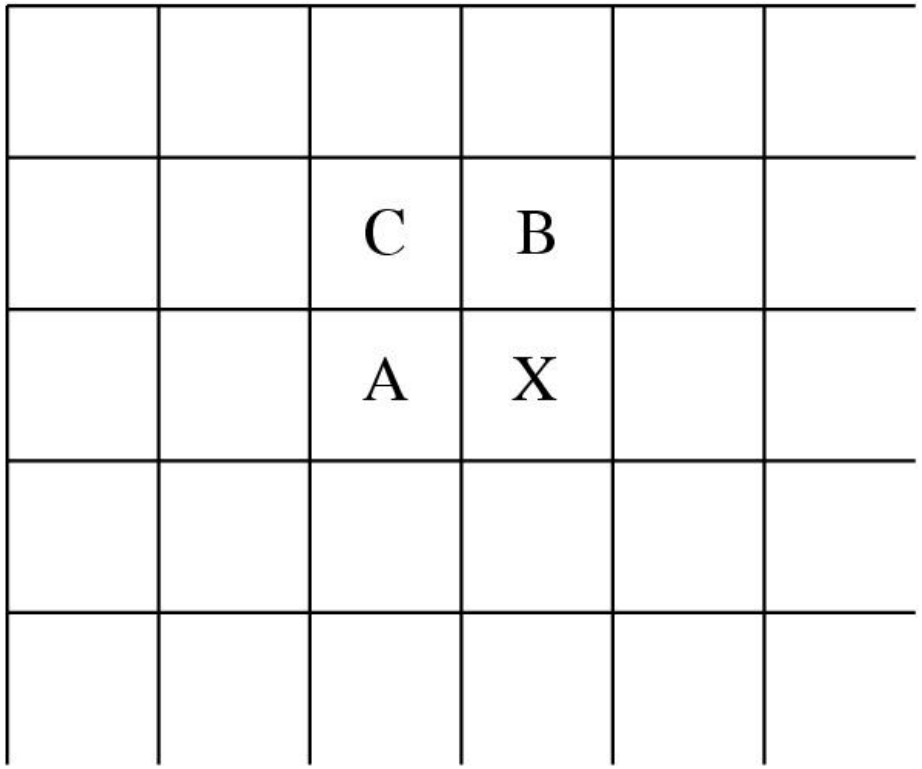


Fig. 7.10: Neighboring Pixels for Predictors in Lossless JPEG.

- **Note**: Any of A, B, or C has already been decoded before it is used in the predictor, on the decoder side of an encode-decode cycle.

3.3 Lossless JPEG

Table 7.6: Predictors for Lossless JPEG

Predictor	Prediction
P1	A
P2	B
P3	C
P4	A + B – C
P5	A + (B – C) / 2
P6	B + (A – C) / 2
P7	(A + B) / 2

Matlab实现基于Huffman编码的图像压缩

ENCO = huffmanenco(SIG, DICT) : 哈夫曼编码函数，SIG为输入编码信号，DICT为编码字典，由函数huffmandict（）生成；
DECO = huffmandeco(COMP, DICT) ： 哈夫曼解码函数，COMP为哈夫曼编码向量，即上面的ENCO；
DICT = huffmandict(SYM, PROB) ： 哈夫曼字典生成函数，SYM为信源符号向量，包含信息中所有符号，PROB为相应符号出现的概率；

```
clear;
clear all;
I = imread('F:\Myfile\Matlab\Test_picture\1_1.jpg');

[M,N] = size(I);
 I1 = I(:);
 P = zeros(1,256);
 %获取各符号的概率;
 for i = 0:255
     P(i+1) = length(find(I1 == i))/(M*N);
 end

 k = 0:255;
 dict = huffmandict(k,P); %生成字典
 enco = huffmanenco(I1,dict); %编码
 deco = huffmandeco(enco,dict); %解码
 Ide = col2im(deco,[M,N],[M,N],'distinct'); %把向量重新转换成图像块;

 subplot(1,2,1);imshow(I);title('original image');
 subplot(1,2,2);imshow(uint8(Ide));title('deco image');
```

课程实验4

- 编写代码实现如下功能：
  - 输入灰度/真彩色图像，利用差分编码思想对其进行重编码并存储为数据文件；
  - 装载编码后的数据文件进行解码得到原始图像文件并显示。