

《数字图像处理》实验报告

实验名称 : 实验 11 图像压缩编码

实验日期 : 2022.11.11

姓 名 : 傅康

学 号 : 084520126

班 级 : 医信 20

成 绩 :

信息技术学院

南京中医药大学

实验目的：

1. 了解图像编码的原理；
2. 掌握常用图像编码方法。

实验内容和要求

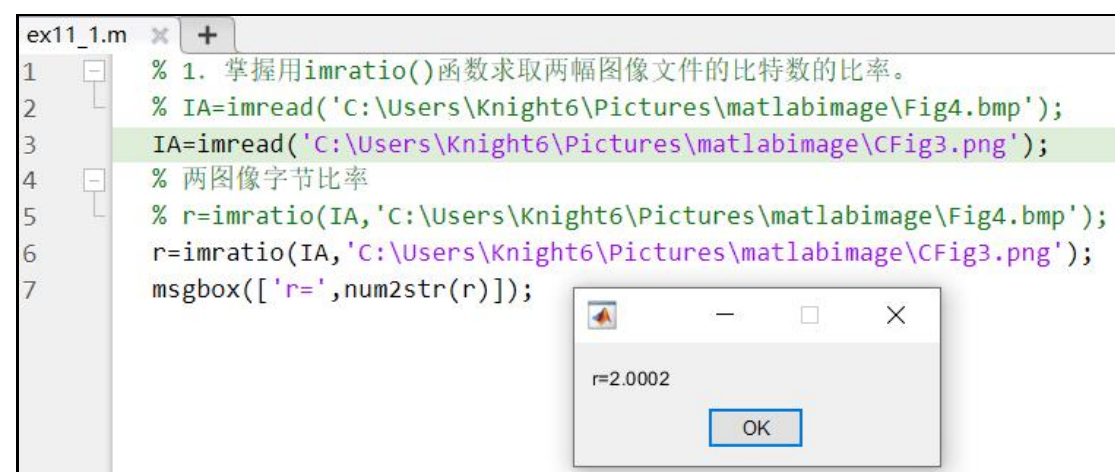
建立一个名为“xxxxxx 实验 11”的解决方案（xxxxxx 为自己的学号）

1. 掌握用 `imratio()` 函数求取两幅图像文件的比特数的比率。
2. 掌握用 `compare()` 函数比较压缩前后两幅的均方根误差；
3. 自主编程实现信源熵的计算；公式：
$$H(z) = -\sum_{j=1}^J p(a_j) \log P(a_j)$$
4. 用 `huff2mat()`, `mat2huff()` 函数对图像进行霍夫曼编码和解码，并用 `imration()` 和 `compare()` 对图像压缩前后的相关数据进行比较；
5. 记录和整理实验报告

运行结果（写清题号）

描述实验的基本步骤，用数据和图片给出各个步骤中取得的实验结果和源代码，并进行必要的讨论，必须包括原始图像及其计算/处理后的图像。

1. 掌握用 `imratio()` 函数求取两幅图像文件的比特数的比率。



```
ex11_1.m  x +
1  % 1. 掌握用imratio()函数求取两幅图像文件的比特数的比率。
2  % IA=imread('C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp');
3  IA=imread('C:\Users\Knight6\Pictures\matlabimage\CFig3.png');
4  % 两图像字节比率
5  % r=imratio(IA,'C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp');
6  r=imratio(IA,'C:\Users\Knight6\Pictures\matlabimage\CFig3.png');
7  msgbox(['r=',num2str(r)]);
```

The screenshot shows a MATLAB script in a file named 'ex11_1.m'. The script reads two images, 'Fig4.bmp' and 'CFig3.png', and calculates the ratio of their byte counts using the `imratio` function. A message box is displayed with the result 'r=2.0002'.

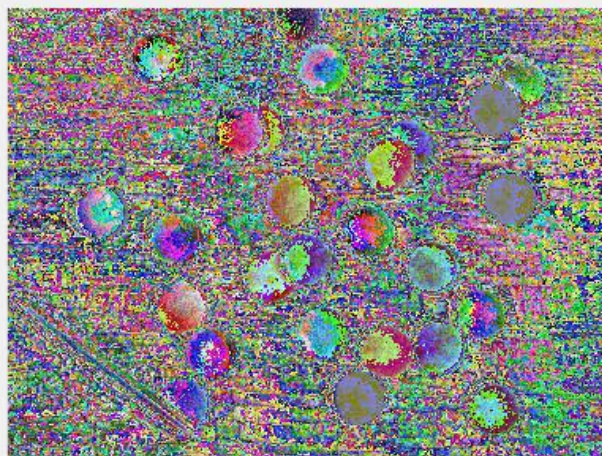
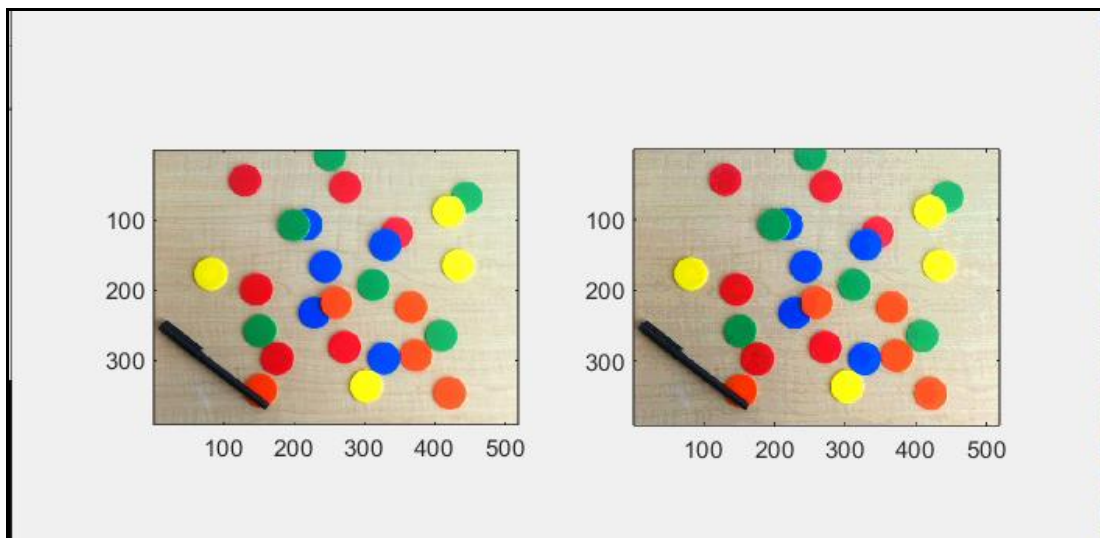
Png 对图像进行了无损压缩，原图字节/压缩后的字节数>1。

注意的一点是，一定要了解不同格式图像是否压缩以及压缩方式，比如 BMP 位图格式就不是压缩图片，其特点是包含的图像信息较丰富，几乎不进行压缩，由此导致了占用磁盘空间过大的缺点，所以产生了很多冗余数据，这也是为啥我最开始用 bmp 图片的比特数比率小于 1 的原因。

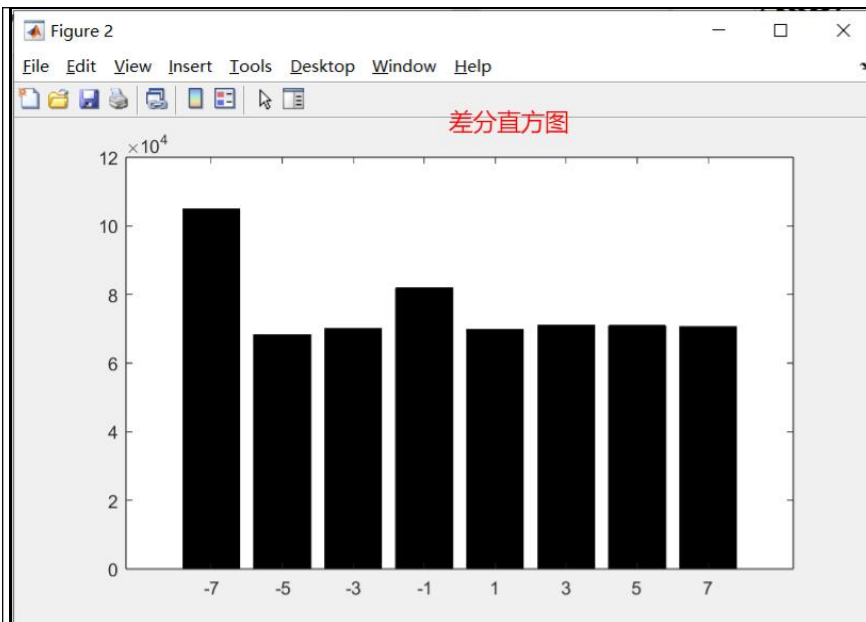
```
ex11_1.m x +
1 % 1. 掌握用imratio()函数求取两幅图像文件的比特数的比率。
2 IA=imread('C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp');
3 % IA=imread('C:\Users\Knight6\Pictures\matlabimage\CFig3.png');
4 % 两图像字节比率
5 r=imratio(IA,'C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp');
6 % r=imratio(IA,'C:\Users\Knight6\Pictures\matlabimage\CFig3.png');
7 msgbox(['r=',num2str(r)]);
```



2. 掌握用 compare() 函数比较压缩前后两幅的均方根误差;



缩放后的差异图像



RMSE=4.856

OK

3. 自主编程实现信源熵的计算；公式：

$$H(z) = -\sum_{j=1}^J p(a_j) \log P(a_j)$$

```

ex11_3.m
1  % 1.一元熵,自主编程实现信源熵的计算
2  Im=imread('C:\Users\Knight6\Pictures\matlabimage\Fig4.bmp');
3  [M,N]=size(Im);
4  temp=zeros(1,256);
5  %对图像的灰度值在[0,255]上做统计
6  for m=1:M
7      for n=1:N
8          if Im(m,n)==0
9              i=1;
10             else
11                 i=Im(m,n)+1;
12             end
13             temp(i)=temp(i)+1;
14         end
15     end
16     temp=temp./(M*N); %每个灰度值的出现概率p(i)
17     %由熵的定义做计算
18     result=0;
19     for i=1:length(temp)
20         if temp(i)==0
21             continue;
22         else
23             result=result-temp(i)*log2(temp(i));%2为底
24         end
25     end
26     msgbox(num2str(result));
  
```

7.4455

OK

4. 用 `huff2mat()`, `mat2huff()` 函数对图像进行霍夫曼编码和解码，并用 `imratio()` 和 `compare()` 对图像压缩前后的相关数据进行比较；

出错：需要 C 的编译器

```
Error in huff2mat (line 61)
x = unravel(y.code', link, m * n);    % Decode using C 'unravel'

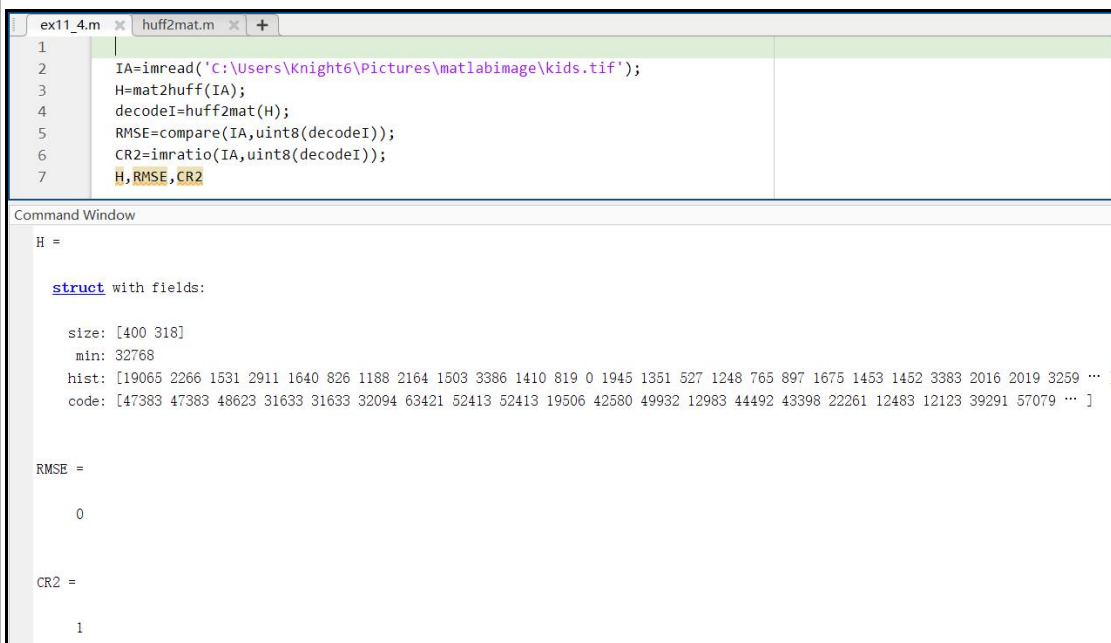
Error in ex11_4 (line 3)
decodeI=huff2mat(H);
```

[JPEG 解码中如何编译 unravel 文件 - MATLAB 中文论坛 - Powered by Discuz! \(ilovematlab.cn\)](#)

[Matlab mex -setup 找不到编译器：为 MATLAB 安装 MinGW64 Compiler 编译器 - 哔哩哔哩 \(bilibili.com\)](#)

[MATLAB 调用 C MEX 文件总是报错 Attempt to execute SCRIPT unravel as a fun... - MATLAB 中文论坛 \(ilovematlab.cn\)](#)

安装好编译器后运行如下：



```
ex11_4.m x huff2mat.m x +
1
2 IA=imread('C:\Users\Knight6\Pictures\matlabimage\kids.tif');
3 H=mat2huff(IA);
4 decodeI=huff2mat(H);
5 RMSE=compare(IA,uint8(decodeI));
6 CR2=imratio(IA,uint8(decodeI));
7 H,RMSE,CR2

Command Window

H =

struct with fields:

    size: [400 318]
    min: 32768
    hist: [19065 2266 1531 2911 1640 826 1188 2164 1503 3386 1410 819 0 1945 1351 527 1248 765 897 1675 1453 1452 3383 2016 2019 3259 ...]
    code: [47383 47383 48623 31633 31633 32094 63421 52413 52413 19506 42580 49932 12983 44492 43398 22261 12483 12123 39291 57079 ...]

RMSE =

0

CR2 =

1
```

结构体 `H` 中存储了霍夫曼编码后的相关信息，霍夫曼编码属于无损压缩格式，通过 `imratio()` 验证 `CR2 = 1`，`compare()` 验证 `RMSE = 0`。

实验的体会与思考题

1. 对比霍夫曼编码和解码的图像属性，谈谈图像压缩的应用？

Huffman 编码是一种无损编码，编码前解码后的图像属性完全相同，其依据信源符号出现的概率来构造其码字，对出现概率大的字符使用较短的码字，对出现概率低的字符则使用较长的码字，从而达到压缩数据的目的，进而缩减其存储空间，减少传输过程中网络宽带的浪费。

这也是图像压缩应用的核心思想，图像压缩是为了节省存储空间，增加传输速度，理想标准是信息丢失的最少，压缩比例的最大。