MATLAB高级编程与工程应用

图像处理

学号：2020010768

班级：无05

姓名：付宇辉

时间：2022年8月4日

**目录**

[一、实验名称与目的 2](#_Toc110524880)

[二、实验内容 2](#_Toc110524881)

[（一）基础知识 2](#_Toc110524882)

[（二）图像压缩编码 4](#_Toc110524883)

[（三）信息隐藏 14](#_Toc110524884)

[（四）人脸检测 17](#_Toc110524885)

[三、文件清单 21](#_Toc110524886)

[四、心得体会 23](#_Toc110524887)

[五、参考内容 24](#_Toc110524888)

# 一、实验名称与目的

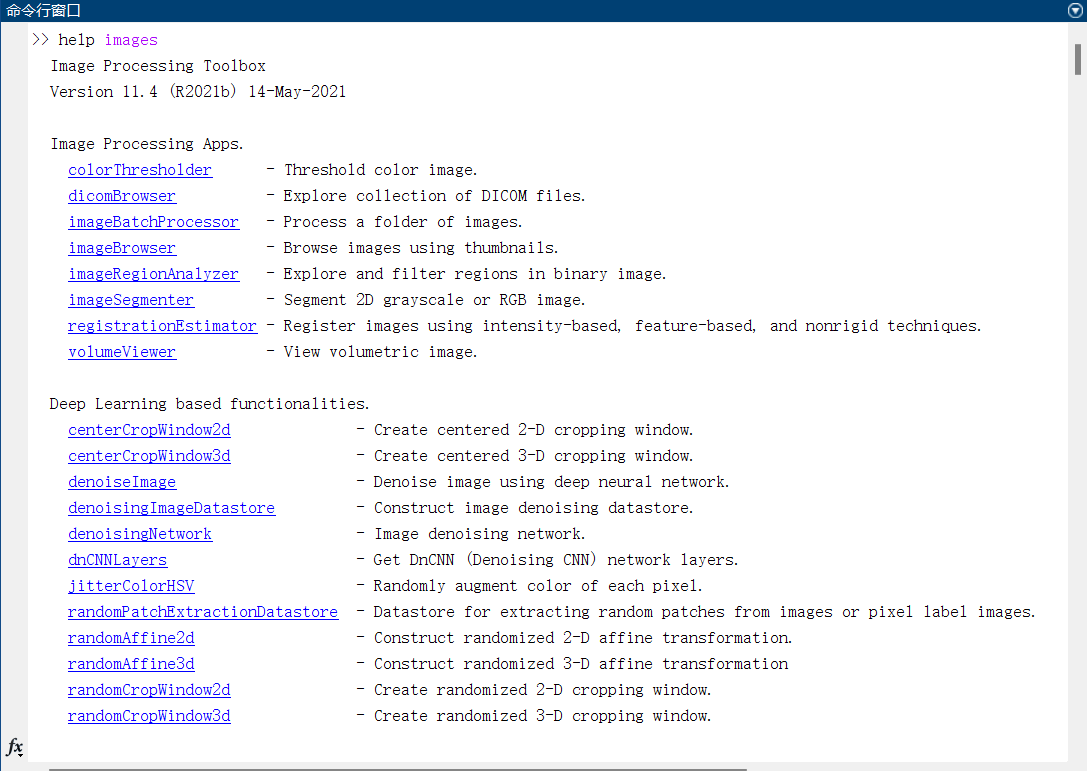
名称：图像处理

目的：了解计算机存储和处理图像的基础知识； 练习MATLAB编程中矩阵运算等相关技巧，同时增进对二维信号及其频谱的物理意义的理解；了解在变换域进行信息隐藏的方法。

# 二、实验内容

## （一）基础知识

1.【题目描述】 MATLAB提供了图像处理工具箱，在命令窗口输入help images可查看该工具箱内的所有函数。请阅读并大致了解这些函数的基本功能。



2. 【题目描述】利用MATLAB提供的image file I/O函数分别完成以下处理：

（a）以测试图像的中心为圆心，图像的长和宽中的较小值的一半为半径画一个红颜色的圆；

（b）将测试图像涂成国际象棋状的“黑白格”的样子，其中“黑”即黑色，“白”则意味着保留原图。

用一种看图软件浏览上述两个图，看是否达到了目标。

【主要思想】可以应用逻辑索引将图片需要修改数据的地方进行处理，从而进行对原图像的处理。

【核心代码】

% (1)

img\_circle = hall\_color;

radius = min(row/2, col/2);

dist = distance(row\_idx , col\_idx, (row+1)/2, (col+1)/2);

circle = (dist < radius^2) & (dist > 0.97 \* radius^2);

img\_circle(circle) = 255;

circle = ~circle;

img\_circle(:,:,2:3) = img\_circle(:,:,2:3).\*uint8(circle);

subplot(1,2,1)

imshow(img\_circle);

% (2)

img\_chessboard = hall\_color;

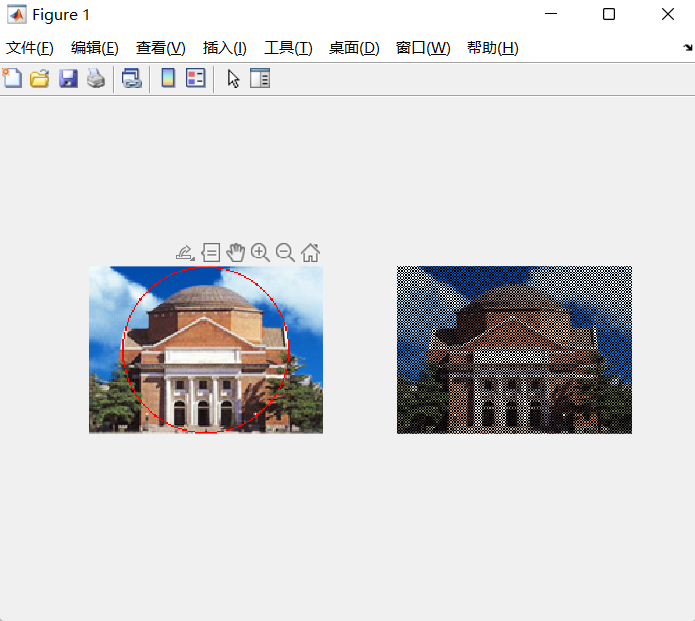
mask = mod(row\_idx + col\_idx, 2); %可以快速实现提取网格信息。

img\_chessboard = img\_chessboard.\*uint8(mask);

subplot(1,2,2)

imshow(img\_chessboard);

【运行结果】



## （二）图像压缩编码

1. 【题目描述】图像的预处理是将每个像素灰度值减去128，这个步骤是否可以在变换域进行？请在测试图像中截取一块验证你的结论。

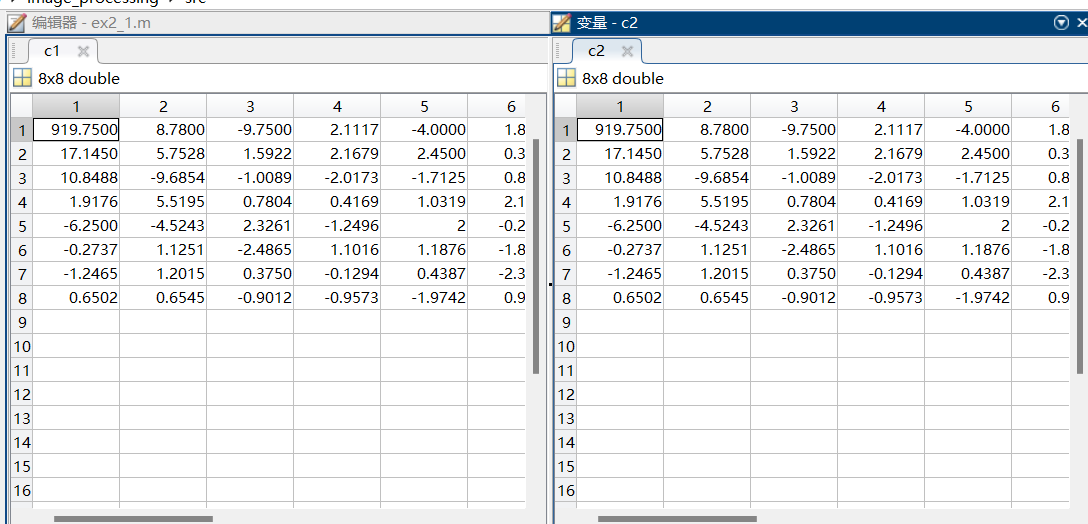
【主要思想】由于DCT具有线性性质，这个步骤可以在变换域进行。预处理将每个像素灰度值减去128，再做DCT，也可以将原图像直接进行DCT之后再与全为128的矩阵DCT相减，结果应该是一致的。

【核心代码】

c1 = dct2(hall\_gray(1:8, 1:8) - 128);

c2 = dct2(hall\_gray(1:8, 1:8)) - dct2(zeros(8, 8) + 128);

【运行结果】将c1，c2展示之后的结果如下图



【结果分析】两个变换域矩阵一致（误差很小），所以这个步骤是可以再在变换域进行的。

2. 【题目描述】请编程实现二维DCT，并和MATLAB自带的函数库dct2比较是否一致。

【主要思想】根据大作业说明书的描述和公式，不难写出代码

【核心代码】

function C = my\_dct2(P)

N = size(P,1);

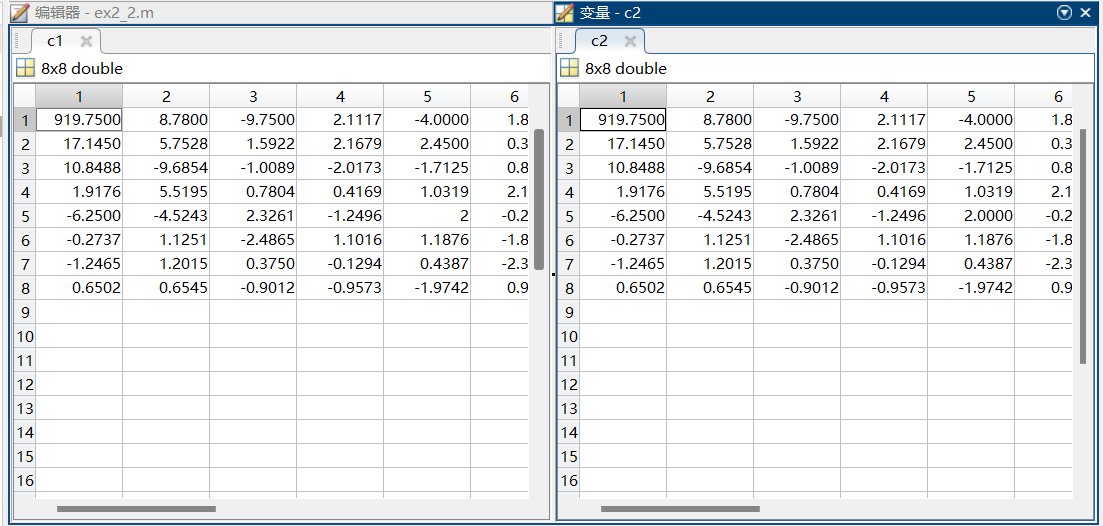
D = (zeros([N - 1, N]) + [1 : 1 : N-1]') .\* [1 : 2 : 2\*N-1];

D = sqrt(2 / N) \* [zeros(1, N) + sqrt(1/2); cos(D \* pi / (2\*N))];

C = D \* double(P) \* D';

end

【运行结果】同样利用hall\_gray的左上角8\*8块进行验证，将c1，c2展示之后的结果如下图



【结果分析】可以看到自己编程实现的DCT与MATLAB自带库的dct2结果一致。

3. 【题目描述】如果将DCT系数矩阵中右侧四列的系数全部置零，逆变换后的图像会发生什么变化？选取一块图验证你的结论。如果左侧的四列置零呢？

【主要思想】DCT的系数矩阵中，左上角是直流低频分量，左下角是纵向变化的高频分量，右上角是横向变化的高频分量，右下角是横向和纵向变换的高频分量。所以将右侧四列置零，对图像整体影响不大，只是在横向的色彩变化比较模糊；但是将左侧四列置零，图像影响很大，直流和低频分量全部消失，同时图像纵向色彩变化会降低，但是横向变化上可能保留明显的变化，出现纵向纹理图像。

【核心代码】

clear\_right\_handle = @(block\_struct) clear\_right(block\_struct.data);

clear\_left\_handle = @(block\_struct) clear\_left(block\_struct.data);

c1 = blockproc(double(hall\_gray)-128, [8 8], clear\_right\_handle);

c2 = blockproc(double(hall\_gray)-128, [8 8], clear\_left\_handle);

clear\_right\_image = uint8(blockproc(c1, [8 8], @(block\_struct) idct2(block\_struct.data) + 128));

clear\_left\_image = uint8(blockproc(c2, [8 8], @(block\_struct) idct2(block\_struct.data) + 128));

【运行结果】将原始图片和进行左右列擦除后的图片show出来之后观察，图片如下图：



【结果分析】可以看到和分析结果一致。

4. 【题目描述】若对DCT系数分别做转置、旋转90度和旋转180度操作（rot90），逆变换后恢复的图像有何变化？选取一块图验证你的结论。

【主要思想】做转置会使横纵向纹理较强和较弱的区块发生对调；对于旋转90度，会使低频分量的值变为纵向变化高频分量的值，会出现较强的横向纹理；矩阵旋转180度，会使低频分量的值变为横纵高频分量的值，出现大部分黑白变化的情况。

【核心代码】

c1 = dct2(sample)';

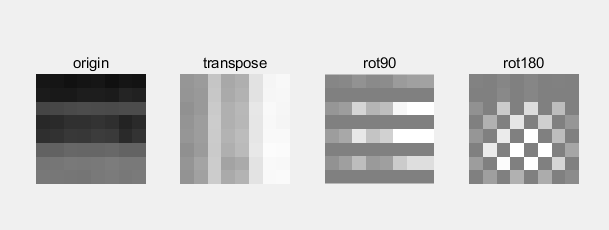
transpose\_image = uint8(idct2(c1)) + 128;

subplot(1,4,2);

imshow(transpose\_image);

title("transpose");

【运行结果】选取一个8\*8的块，运行结果如下图所示



【结果分析】可以看到和分析结果一致。

5. 【题目描述】如果认为差分编码是一个系统，请绘出这个系统的频率响应，说明他是一个\_\_（低通、高通、带通、带阻）滤波器。DC系数先进行差分编码再进行熵编码，说明DC系数的\_\_频率分量更多。

【主要思想】这个系统的差分方程为，利用matlab分析离散系统的工具，即可画出频率响应并进行相应的判断。

答案为：高通，高频

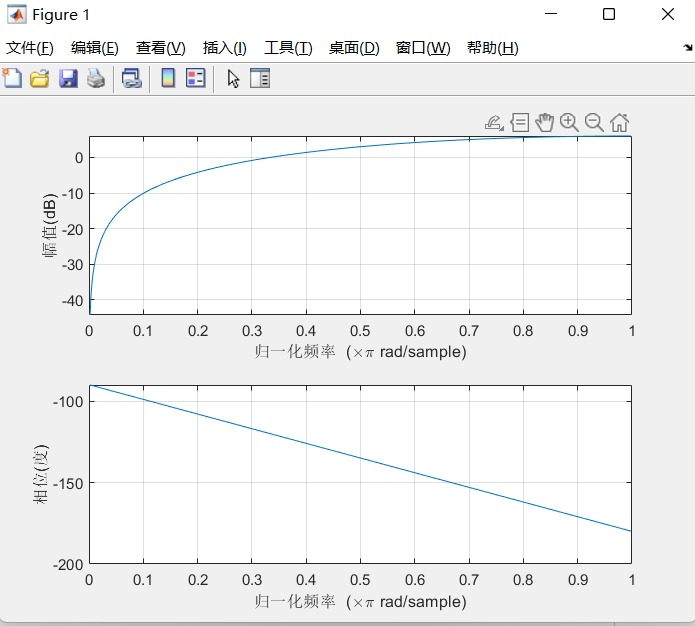
【核心代码】

a = 1;

b = [-1 1];

freqz(b, a);

【运行结果】运行结果如下图所示



【结果分析】可以看到和分析结果一致，答案为：高通，高频

6. 【题目描述】DC预测误差的取值和Category值有何关系？如何利用预测误差计算出其Category？

【主要思想】假设Category的值为c，DC预测误差的取值为e，则有：

7. 【题目描述】你知道哪些实现Zig-Zag扫描的方法？请利用MATLAB的强大功能设计一种最佳方法。

【主要思想】利用高级的矩阵索引方式可以实现Zig-Zag扫描

【核心代码】

function y = zig\_zag(x)

order = [1, 2, 6, 7, 15, 16, 28, 29;

3, 5, 8, 14, 17, 27, 30, 43;

4, 9, 13, 18, 26, 31, 42, 44;

10, 12, 19, 25, 32, 41, 45, 54;

11, 20, 24, 33, 40, 46, 53, 55;

21, 23, 34, 39, 47, 52, 56, 61;

22, 35, 38, 48, 51, 57, 60, 62;

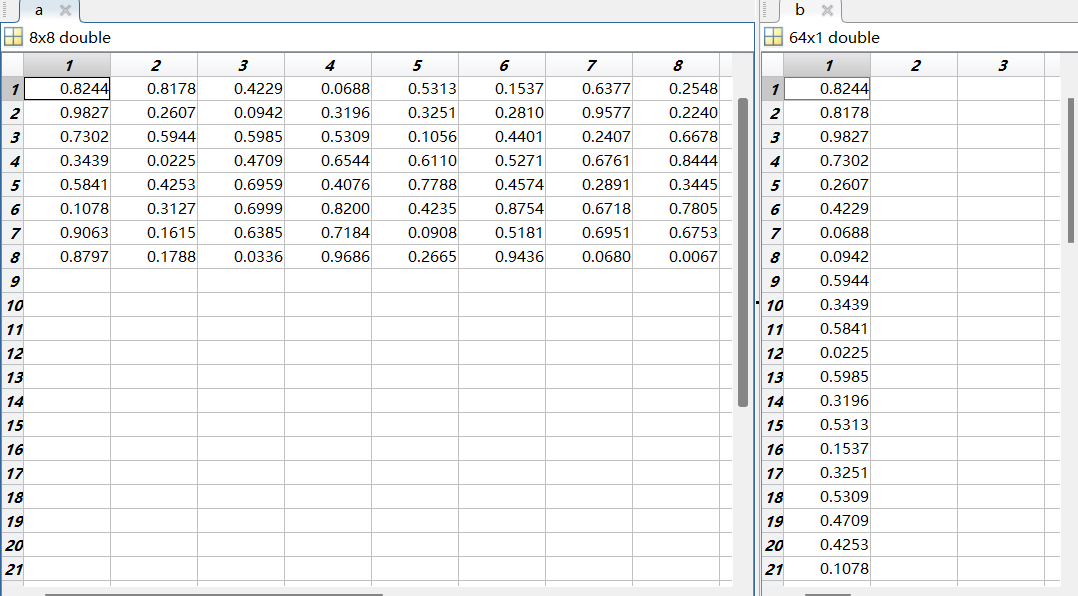
36, 37, 49, 50, 58, 59, 63, 64];

idx(order) = 1:64;

y = x(idx)';

end

【运行结果】运行结果如下图所示



【结果分析】成功实现了Zig-Zag扫描。

8. 【题目描述】对测试图像分块、DCT和量化，将量化后的系数写成矩阵的形式，其中每一列为一个块的DCT系数Zig-Zag扫描后形成的列矢量，第一行为各个块的DC系数。

【主要思想】按照作业说明书的顺序，逐步对测试图像分块、DCT和量化即可。

【核心代码】

c = blockproc(double(hall\_gray)-128, [8 8], ...

@(block\_struct) zig\_zag(round(dct2(block\_struct.data)./QTAB)));

[row, col] = size(c);

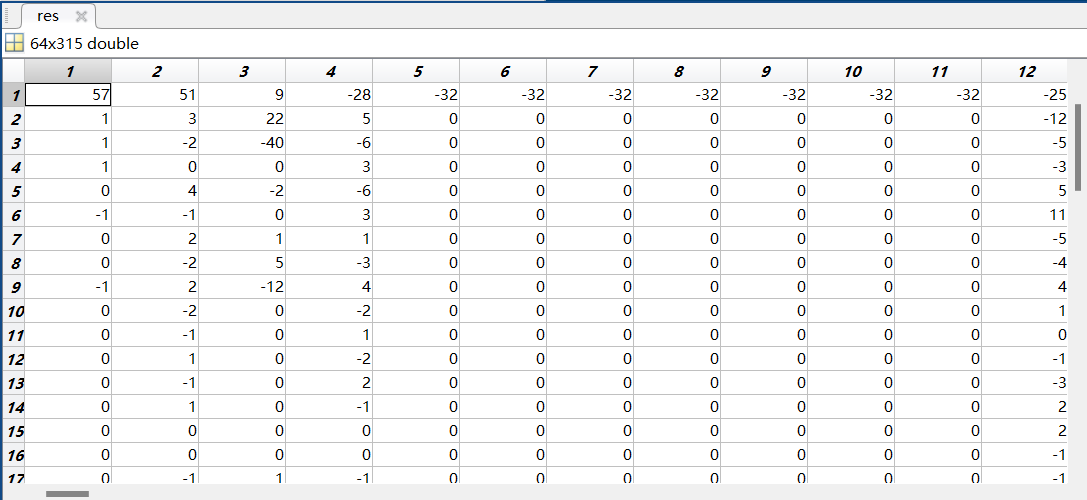
res = zeros([64, row/64\*col]);

for i = 1:row/64

res(:, (i-1)\*col+1:i\*col) = c((i-1)\*64+1:i\*64, 1:col);

end

【运行结果】结果存储在变量res中。



9. 【题目描述】请实现本章介绍的JPEG编码（不包括写JFIF文件），输出为DC系数的码流、AC系数的码流、图像高度和图像宽度，将这四个变量写入jpegcodes.mat文件。

【主要思想】将上述步骤串起来，再加入huffman编码可以获得相应的码流。

【核心代码】

cD = res(1, :)';

cD = [cD(1); -diff(cD)];

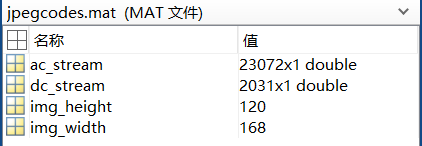
category = ceil(log2(abs(cD) + 1));

dc\_stream = cell2mat(arrayfun(@(i) ...

[DCTAB(category(i)+1, 2:DCTAB(category(i)+1, 1)+1), my\_dec2bin(cD(i))]', ...

(1:length(cD))', 'UniformOutput', false));

【运行结果】



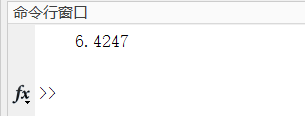
10. 【题目描述】计算压缩比（输入文件长度/输出码流长度），注意转换为相同进制。

【主要思想】输入文件长度即为图像的像素块数量（单位是字节）乘8，输出码流长度为DC和AC流的长度。

【核心代码】

CR = (img\_width\*img\_height\*8) / (length(dc\_stream)+length(ac\_stream));

【运行结果】



【结果分析】压缩比为6.4247，还是很高的。

11. 【题目描述】请实现本章介绍的JPEG解码，输入是你生成的jpegcodes.mat文件。分别用客观（PSNR）和主观方式评价编解码效果如何。

【主要思想】解码的过程与编码过程正好相反，步骤为DC解码、AC解码、重组为系数矩阵、逆DCT变换。DC编码时，有三种情况，编码00，幅度0序列00；编码是三位时，幅度序列长度为十进制编码减一；编码多于三位时，幅度序列长度为第一个数加二。所以可以看0出现的位置判断编码。而对于 AC 编码，Huffman 码表较为复杂，故采用ismember和find结合的方式，查找序 列的前 i 位是否存在于表中。对于解码得到的 64×blocks 的系数矩阵，先使用reshape，转换为 64×横向块数×纵向块数 的张量。再使用permute交换行列，以满足 Matlab 列优先的特性。最后使用reshape和转置 变形为 64纵向块数×横向块数 的矩阵。

【核心代码】

while(cnt <= blocks)

if(all(dc\_stream(1:2) == [0 0]))

dc\_stream(1:2) = [];

else

zero\_ind = find(dc\_stream==0, 1);

if(zero\_ind < 4)

huffman\_code = dc\_stream(1:3);

dc\_stream(1:3) = [];

zero\_ind = bin2dec(strjoin(string(huffman\_code), '')) - 1;

else

dc\_stream(1:zero\_ind) = [];

zero\_ind = zero\_ind + 2;

end

mag = bin\_vec2dec(dc\_stream(1:zero\_ind));

DC\_coeff(cnt) = mag;

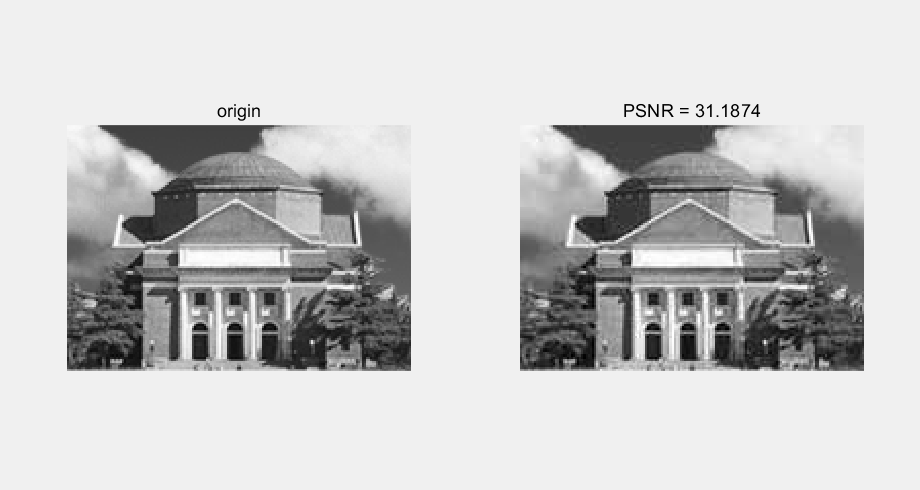
dc\_stream(1:zero\_ind) = [];

end

cnt = cnt + 1;

end

【运行结果】



【结果分析】从客观上，PSNR的值为31左右，足够大，说明效果还可以，并不算失真太严重。主观观察两幅图片，发现效果确实还可以。

12. 【题目描述】将量化步长减小为原来的一半，重做解编码。同标准化步长的情况比较压缩比和图像质量。

【主要思想】将QTAB除以2，剩下只需要重复操作即可。

【核心代码】

QTAB = QTAB / 2;

[dc\_stream, ac\_stream, img\_height, img\_width] = JPEG\_encode(hall\_gray, QTAB, DCTAB, ACTAB);

img = JPEG\_decode(dc\_stream', ac\_stream', img\_height, img\_width, QTAB, ACTAB);

【运行结果】



【结果分析】量化步长变为原来的一般使得，压缩比下降为4.41，PSNR变为34.2。实际观感上，图像噪声有所改善。

13. 【题目描述】看电视时偶尔能看到美丽的雪花图像（见snow.mat），请对其解编码。和测试图像的压缩比和图像质量进行比较，并解释比较结果。

【主要思想】只需要将snow替换掉hall\_gray即可

【核心代码】

[dc\_stream, ac\_stream, img\_height, img\_width] = JPEG\_encode(snow, QTAB, DCTAB, ACTAB);

img = JPEG\_decode(dc\_stream', ac\_stream', img\_height, img\_width, QTAB, ACTAB);

CR = (img\_width\*img\_height\*8) / (length(dc\_stream)+length(ac\_stream));

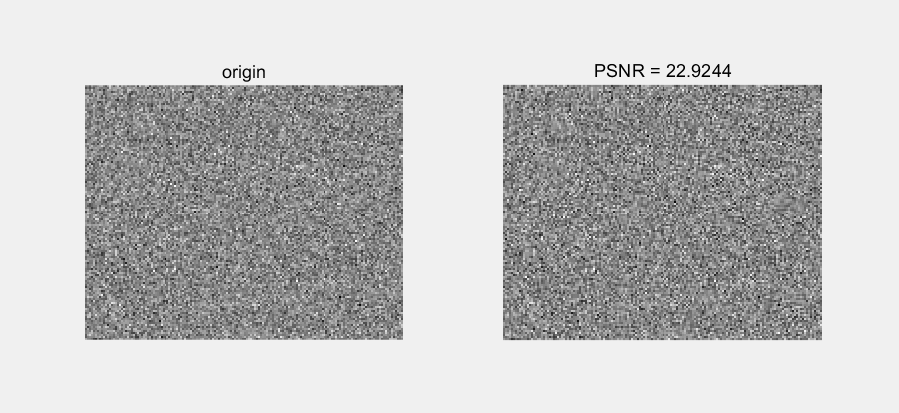
disp(CR);

MSE = sum((double(img) - double(snow)).^2, 'all') / (img\_height \* img\_width);

PSNR = 10 \* log10(255 \* 255 / MSE);

disp(PSNR);

【运行结果】





【结果分析】相比于原来的图片，换成雪花图片之后，压缩比下降了，PSNR也下降了。原因可能雪花图接近随机矩阵，量化表不适合对其进行处理，故最终压缩比相对较低。高频分量被量化导致高频分量存在失真，而雪花图高频分量较多，因此图片的失真比较严重，PSNR 较低。

## （三）信息隐藏

1. 【题目描述】实现本章介绍的空域隐藏方法和提取方法。验证其抗JPEG编码能力。

【主要思想】随机生成相应的加密信息，然后通过多次测试，进行相应的编码和解码，计算正确率。

【核心代码】

for i = 1:test\_time

seq = randi([0, 1], 1, randi(max\_length));

secret = [seq, zeros(1, max\_length - length(seq))];

encode\_data = reshape(bitand(img, zeros(size(img))), 1, []) + secret;

encode\_data = reshape(encode\_data, size(img));

[dc\_stream, ac\_stream, img\_height, img\_width] = JPEG\_encode(encode\_data, QTAB, DCTAB, ACTAB);

decode\_data = JPEG\_decode(dc\_stream', ac\_stream', img\_height, img\_width, QTAB, ACTAB);

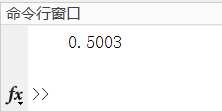
secret\_decode = reshape(decode\_data, 1, []);

secret\_decode = mod(secret\_decode(1:length(seq)), 2);

correct = correct + sum(secret\_decode == seq) / length(seq);

end

【运行结果】



【结果分析】看到正确率再50%左右，并没有将隐藏的信息完美的保留下来。空域隐藏法抗编解码能力弱。

2. 【题目描述】依次实现本章介绍的三种变换域信息隐藏方法和提取方法，分析嵌密方法的隐蔽性以及嵌密后JPEG图像的质量变化和压缩比变化。

【主要思想】

1. 用信息位逐一替换掉每个量化后的DCT系数的最低位，在进行熵编码。
2. 不是每一个系数都嵌入信息。
3. 先将待隐藏信息用1，-1的序列表示，再逐一将信息位追加在每个块zig-zag顺序的最后一个非0DCT系数之后；如果原本该图像块的最后一个系数就不为0，那就用信息位替换该位。

【核心代码】

code1 = randi([0, 1], size(c\_res, 1), size(c\_res, 2));

code2 = randi([0, 1], 1, size(c\_res, 2));

c\_hide1 = double(bitset(int64(round(c\_res)), 1, code1));

c\_hide2 = c\_res;

[~, min\_idx] = min(zig\_zag(QTAB));

c\_hide2(min\_idx, :) = double(bitset(int64(round(c\_res(min\_idx, :))), 1, code2));

code3 = code2;

code3\_transfrom = code3;

code3\_transfrom(code3 == 0) = -1;

c\_hide3 = zeros(size(c\_res));

for i = 1:size(c\_res, 2)

seq = c\_res(:, i);

if seq == 0

c\_hide3(:, i) = [code3\_transfrom(i); c\_res(2:end, i)];

else

not\_zero = flipud(seq ~= 0);

seq = flipud(seq);

[~, idx] = max(not\_zero);

if idx == 1

seq(1) = code3\_transfrom(i);

else

seq(idx-1) = code3\_transfrom(i);

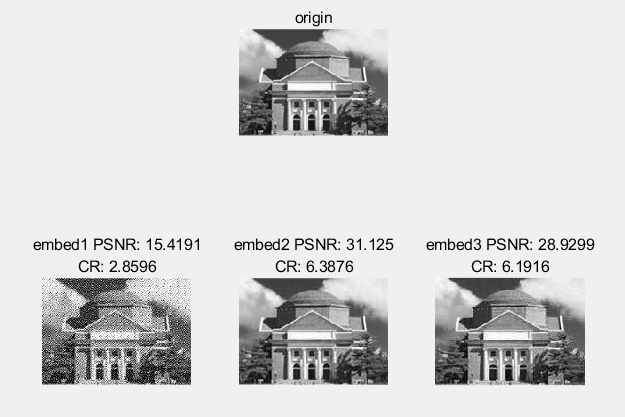
end

c\_hide3(:, i) = flipud(seq);

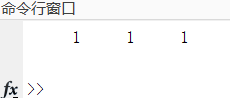
end

end

【运行结果】



正确率：



【结果分析】

方法1的质量和压缩比明显下降，并且看起来有明显的棋盘状图案，基本没有起到“隐藏”的意义。方法2和3添加信息较少，压缩比与原图像比时大致相同。

## （四）人脸检测

1. 【题目描述】所给资料Faces目录下包含从网图中截取的28张人脸，试以其作为样本训练人脸标准v

【主要思想】

1. 样本人脸大小不一，是否需要首先将图像调整为相同大小？

解答：不需要，因为我们统计的是样本中各种颜色所占的比例，而与图片大小无关。

1. 假设L分别取3，4，5，所得三个v之间有何关系？

解答：三个v的关系就是相当于“分辨率”大小的关系，L越大，相当于将一个点的色彩范围缩得越小。L=3，4，5分别对应着一个点的色彩范围位2^5，2^4，2^3。

【核心代码】

function output = get\_color\_ratio(img, L)

bins = 0:2^(3\*L);

img = floor(double(img) / 2^(8-L));

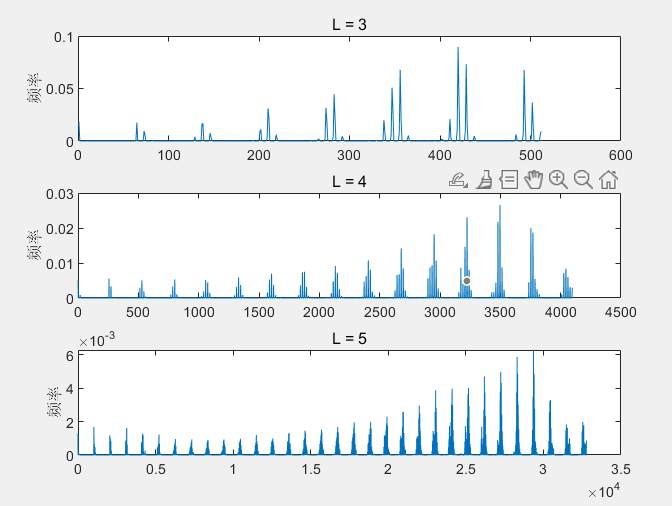
pixel = img(:, :, 1) \* 2^(2\*L) + img(:, :, 2) \* 2^L + img(:, :, 3);

pixel = reshape(pixel, 1, []);

output = histcounts(pixel, bins) \* 3 / numel(img);

end

【运行结果】



2. 【题目描述】设计一种从任意大小的图片中检测任意多张人脸的算法并编程实现（输出图像在判定为人脸的位置加上红色的方框）。随意选取一张多人图片（比如支部活动或者足球比赛），对程序进行测试。尝试L分别取不同的值，评价检测结果有何区别。

【主要思想】自己设置人脸的大小，然后利用框滑动的办法实现多张人脸的检测。

【核心代码】

for j = 0 : ww - 1

for i = 0 : hh - 1

block = img(i \* stride(1) + 1 : i \* stride(1) + block\_size(1), ...

j \* stride(2) + 1 : j \* stride(2) + block\_size(2), :);

blk\_hist = get\_color\_ratio(block, L);

distance = 1 - sum(sqrt(blk\_hist .\* color\_v));

output(sub2ind([hh, ww], i + 1, j + 1), :) = ...

[i \* stride(1) + 1, j \* stride(2) + 1, distance];

end

end

【运行结果】



【结果分析】

L越大，颜色细节越多，需要适当调大阈值以得到相似的结果。

3. 【题目描述】图像进行处理后，分析结果

【主要思想】分别将图像旋转90度，宽度拉伸2倍，改变颜色分析

【核心代码】

img\_rotate = imrotate(img, 90);

img\_resize = imresize(img, [size(img, 1), size(img, 2) \* 2]);

img\_adjust = imadjust(img,[.2 .3 0; .6 .7 1],[]);

rect\_img\_rotate = face\_detect(img\_rotate, v\_3, 3, [40, 50], stride, 0.5, 100);

rect\_img\_resize = face\_detect(img\_resize, v\_3, 3, [50, 80], stride, 0.5, 100);

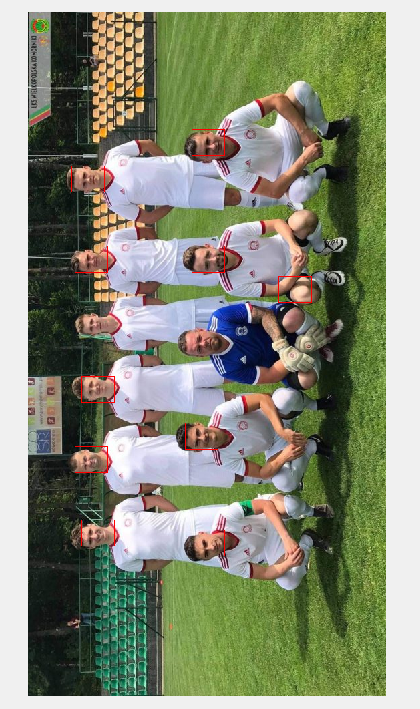
rect\_img\_adjust = face\_detect(img\_adjust, v\_3, 3, block\_size, stride, 0.5, 100);

img\_rotate = insertShape(img\_rotate, 'Rectangle', rect\_img\_rotate, 'Color', 'red');

img\_resize = insertShape(img\_resize, 'Rectangle', rect\_img\_resize, 'Color', 'red');

img\_adjust = insertShape(img\_adjust, 'Rectangle', rect\_img\_adjust, 'Color', 'red');

【运行结果】







【结果分析】

旋转没有影响检测结果；而拉伸可能会插值，使结果不同；颜色调整会带来较大的影响。这种检测方法稳定性十分不足。

4. 【题目描述】如果可以重新选择人脸样本训练标准，你觉得应该如何选取？

【主要思想】受限于这种方法对于颜色高度敏感，人脸样本的选取应该与要检测的数据相类似，比如人的肤色需要十分相近，以及照片的光线等需要和检测数据类似，才能使这种方法达到一个比较不错的效果。

# 三、文件清单

本次报告的目录结构如下：

├── doc

│   └── report.pdf

├── res

│   ├── Faces

│   │   ├── 1.bmp

│   │   ├── 10.bmp

│   │   ├── 11.bmp

│   │   ├── 12.bmp

│   │   ├── 13.bmp

│   │   ├── 14.bmp

│   │   ├── 15.bmp

│   │   ├── 16.bmp

│   │   ├── 17.bmp

│   │   ├── 18.bmp

│   │   ├── 19.bmp

│   │   ├── 2.bmp

│   │   ├── 20.bmp

│   │   ├── 21.bmp

│   │   ├── 22.bmp

│   │   ├── 23.bmp

│   │   ├── 24.bmp

│   │   ├── 25.bmp

│   │   ├── 26.bmp

│   │   ├── 27.bmp

│   │   ├── 28.bmp

│   │   ├── 29.bmp

│   │   ├── 3.bmp

│   │   ├── 30.bmp

│   │   ├── 31.bmp

│   │   ├── 32.bmp

│   │   ├── 33.bmp

│   │   ├── 4.bmp

│   │   ├── 5.bmp

│   │   ├── 6.bmp

│   │   ├── 7.bmp

│   │   ├── 8.bmp

│   │   └── 9.bmp

│   ├── JpegCoeff.mat

│   ├── faces.jpg

│   ├── hall.mat

│   └── snow.mat

├── src

│   ├── JPEG\_decode.m

│   ├── JPEG\_encode.m

│   ├── ex1\_2.m

│   ├── ex2\_1.m

│   ├── ex2\_10.m

│   ├── ex2\_11.m

│   ├── ex2\_12.m

│   ├── ex2\_13.m

│   ├── ex2\_2.m

│   ├── ex2\_3.m

│   ├── ex2\_4.m

│   ├── ex2\_5.m

│   ├── ex2\_7.m

│   ├── ex2\_8.m

│   ├── ex2\_9.m

│   ├── ex3\_1.m

│   ├── ex3\_2.m

│   ├── ex4\_1.m

│   ├── ex4\_2.m

│   ├── ex4\_3.m

│   ├── face\_detect.m

│   ├── face\_v.mat

│   ├── get\_color\_ratio.m

│   ├── jpegcodes.mat

│   ├── my\_dct2.m

│   └── zig\_zag.m

└── tree.txt

# 四、心得体会

这次实验让我体会到了信号与系统的有趣之处以及有用的地方。我是姚铮老师班上的学生，所以matlab课堂是我第一次接触谷老师，谷老师十分有趣，并且将内容活灵活现地展现在我面前。

这次图像处理作业，让我巩固了信号与系统的知识以及让我对图像处理有了初步的了解，总体来说还是非常有收获的。

但是实验过程中也有很多不足的地方：人脸识别的效果并没有做的很好；有些地方能用矩阵运算解决的地方并没有使用，而是用了较为缓慢的循环运算，等等。希望之后利用matlab可以避免这些问题。

在作业当中收获到很多，谢谢老师和助教的指导和帮助！

# 五、参考内容

* MATLAB官方help文档
* 《信号与系统——MATLAB综合实验》谷源涛 应启珩 郑君里