**哈尔滨工业大学（威海）**

《数字图像处理》

实验报告

**院系： 计算机科学与技术**

**班级： 2104102**

**学号： 2021211041**

**姓名： 蒲海博**

|  |  |  |
| --- | --- | --- |
| 实验1 | 实验2 | 总分 |
|  |  |  |

**实验一**

**实验一题目1的运行代码、使用参数、实验结果截图、实验结论：**

**题目1：基本MATLAB图像处理程序练习**

**1、基本MATLAB 图像处理函数的使用**

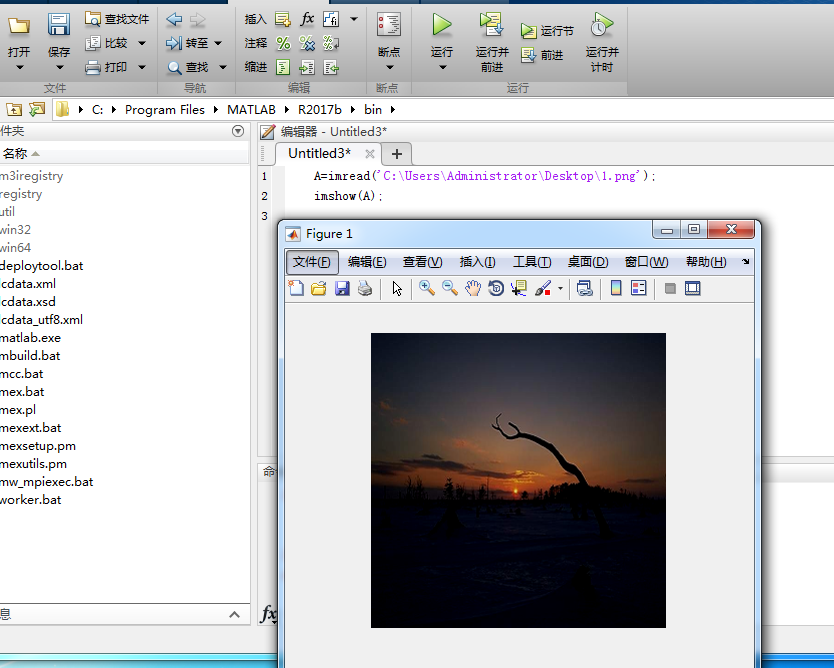
1. imread 函数用来实现图像文件的读取。

代码：

A=imread('C:\Users\Administrator\Desktop\1.png');

imshow(A);

截图：



结论：

imread函数读取了该文件夹中的图像

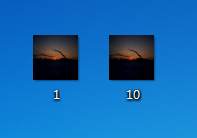
imshow函数显示图像

（2）imwrite 函数用来实现图像文件的写入。

代码：

imwrite(A,' C:\Users\ Administrator\Desktop\1.png');

截图：



结论：imwrite函数：向文件中写入该图像

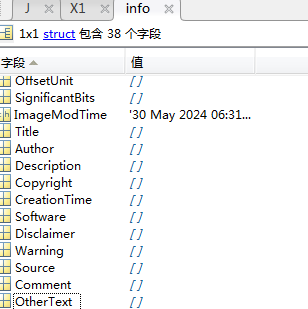
（3）imfinfo 函数用来查询图像文件信息。

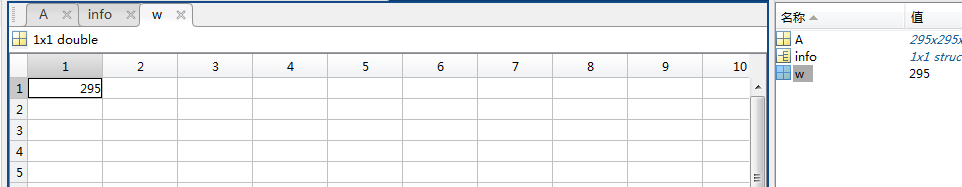
代码：

info=imfinfo('C:\Users\Administrator\Desktop\1.png');%单步调试看info中成员

w=info.Width;%等其他信息

截图：





结论：info函数显示该图片信息参数

Width：图像宽度，像素为单位（图中宽为295）

（4）subimage 函数实现在一个图形窗口中显示多幅图像。

代码：

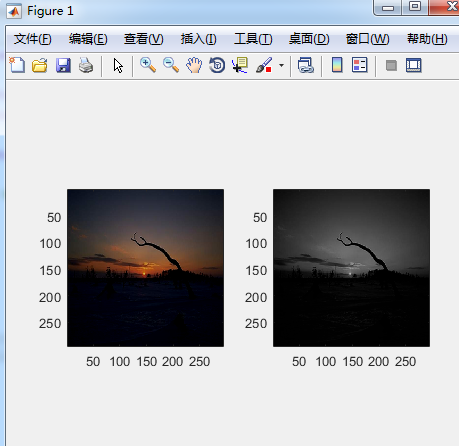
RGB=imread(' C:\Users\Administrator\Desktop\1.png ');

I=rgb2gray(RGB);

subplot(1,2,1),subimage(RGB) %subimage 实现在一个图形窗口中显示多幅图像

subplot(1,2,2),subimage(I)

截图：



结论：rgb2gray 将rgb图像转换成灰度图像

subimage 实现在一个图形窗口中显示多幅图像

Subplot（x,y,z）表示x行y列分块，并在第z块进行操作

**2、图像的空间域操作**

（1）imresize 函数实现图像的缩放。

代码：

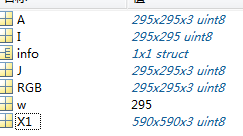
J=imread('C:\Users\Administrator\Desktop\1.png');

X1=imresize(J,2); %对图像进行缩放

figure,imshow(J)

截图：

1. 单步调试，查看J和X1的大小，学习参数2的含义

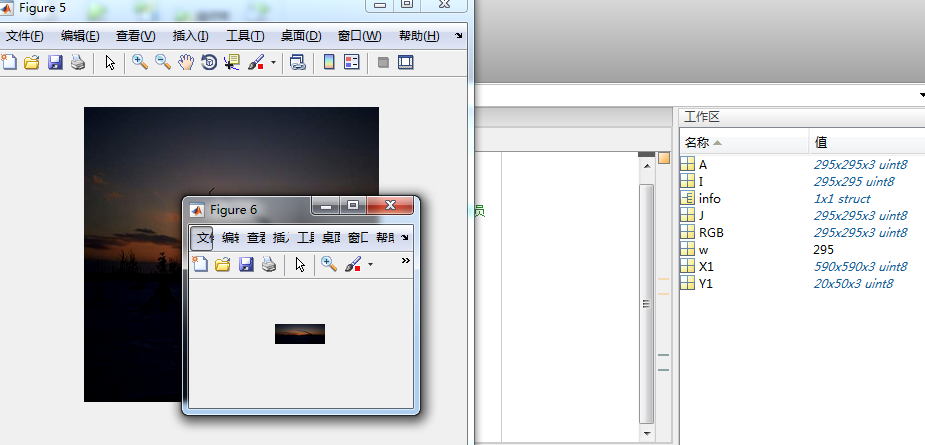


结论：

imresize(j,x) 将j图像长宽放大x倍

2）使用help命令，查看imresize函数功能，将图像J转换为20\*50大小图像并显示。





结论：imresize(j,[x,y]) 将J图像设置为x×y大小

（2）imrotate 函数实现图像的旋转。

代码：

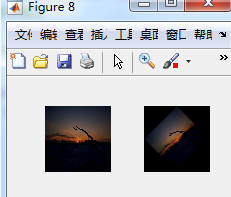
I=imread('C:\Users\Administrator\Desktop\1.png');

J=imrotate(I,45,'bilinear'); %对图像进行旋转

subplot(1,2,1),imshow(I);

subplot(1,2,2),imshow(J);

截图：



结论：B=imrotate(A,angle,method)，利用指定的插值方法method进行图像A的旋转

method是文本字符串，需要引号引出，默认值用大括号{}括起来。

method包括：

{‘nearest’}最临近插值-默认值

'bilinear’双线性的

'bicubic’双三次的

（3）imcrop 函数实现图像的剪切。

代码：

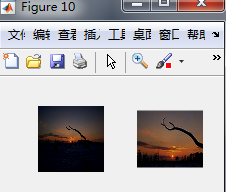
I=imread('C:\Users\Administrator\Desktop\1.png');

I2=imcrop(I,[75 68 130 112]); %对图像进行剪切

subplot(1,2,1),imshow(I);

subplot(1,2,2),imshow(I2);

截图：



结论：imcrop函数 ：前一个因素表示要裁剪的图像，后一个因素表示裁剪区域

（4）roicolor 函数是对RGB 图像和灰度图像实现按灰度或亮度值选择区域进行处理。

代码：

a=imread('C:\Users\Administrator\Desktop\1.png');

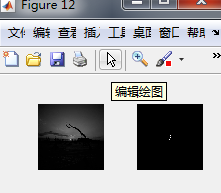
I=rgb2gray(a);

BW=roicolor(I,128,225); %按灰度值选择的区域

subplot(1,2,1),imshow(I);

subplot(1,2,2),imshow(BW)

截图：



结论：

BW = roicolor(A,low,high)色图范围为[low high]，返回这些像素选择区域。BW 为大小和 A 相同的数组，若 A 中元素值在[low high]范围内，则返回值 1，否则返回 0.

**3、图像代数运算**

（1）imadd 函数实现两幅图像的相加或者给一幅图像加上一个常数。给图像每个像素都增加亮度的程序如下：

代码：

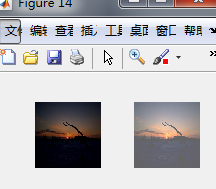
I=imread('C:\Users\Administrator\Desktop\1.png');

J=imadd(I,100); %给图像增加亮度

subplot(1,2,1),imshow(I)

subplot(1,2,2),imshow(J)

截图：



结论：

加法运算imadd函数：imadd(X,Y)将数组 X 中的每个元素与数组 Y 中的对应元素相加，并在输出数组 Z 的对应元素中返回和;若Y为常数，则给图像增加亮度。

（2）imsubtract 函数实现将一幅图像从另一幅图像中减去，或者从一幅图像中减去一个常数。

代码：

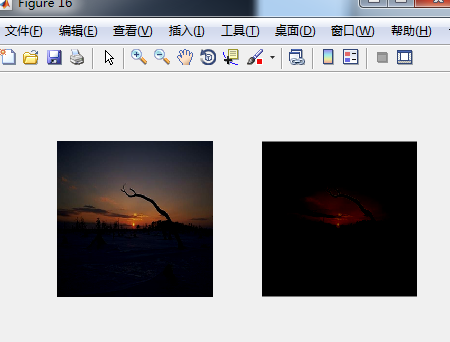
I=imread('C:\Users\Administrator\Desktop\1.png');

J=imsubtract(I,100); %给图像减少亮度

subplot(1,2,1),imshow(I)

subplot(1,2,2),imshow(J)

截图：



结论：

imsubtract函数：imsubtract(X,Y)将数组 X 中的每个元素与数组 Y 中的对应元素相减，并在输出数组 Z 的对应元素中返回差;若Y为常数，则给图像减弱亮度

（3）immultiply 实现两幅图像的相乘或者一幅图像的亮度缩放。

代码：

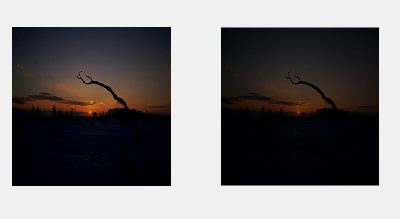
I=imread('C:\Users\Administrator\Desktop\1.png');

J=immultiply(I,0.5); %对图像进行亮度缩放

subplot(1,2,1),imshow(I)

subplot(1,2,2),imshow(J)

代码：



结论：

Z=immultiply(X,Y)两幅图像进行乘法运算可以实现掩模操作，即屏蔽掉图像的某些部分。一幅图像乘以一个常数通常称为缩放。缩放通常将产生比简单添加像偏移量自然得多的明暗效果。如果使用的缩放因数大于1，那么将增强图像的亮度，如果因数小于1，则会使图像变暗。 在MATLAAB中，使用immultiply函数实现两幅图形的乘法。该函数将两幅图像相应的像素值进行元素对元素的乘法，并将乘法的运算结果作为输出图像相应的像素值。

（4）imdivide 函数实现两幅图像的除法或一幅图像的亮度缩放。

代码：

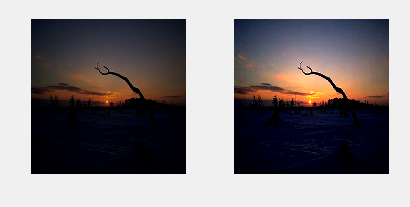
I=imread('C:\Users\Administrator\Desktop\1.png');

J=imdivide(I,0.5); %图像的亮度缩放

subplot(1,2,1),imshow(I)

subplot(1,2,2),imshow(J)

截图：



结论：

Z=imdivide(X,Y)图像除法运算又称为比率变换，给出的是相应像素值的变化比率，而不是每个像素的绝对差异，可用于校正成像设备的非线性影响。 在MATLAB中使用imdivide函数进行两幅图像的除法，该函数对两幅输入图像的所有相应像素执行元素对元素的除法操作，并将得到的结果作为输出图像的相应像素值。

**4、图像增强、变换和分割**

（1）imhist 函数产生图像的直方图。

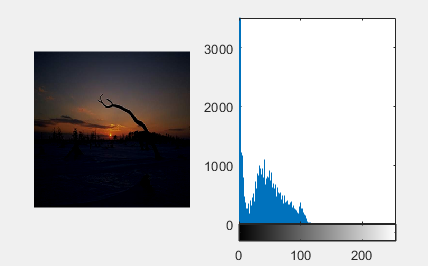
代码：A=imread('C:\Users\Administrator\Desktop\1.png'); %读入图像

B=rgb2gray(A); %把RGB 图像转化成灰度图像

imshow(B); %显示灰度图像

imhist(B); %显示灰度图像的直方图

截图：



结论：

imhist()函数显示灰度图像的直方图

（2）histeq 函数用于对图像的直方图均衡化。

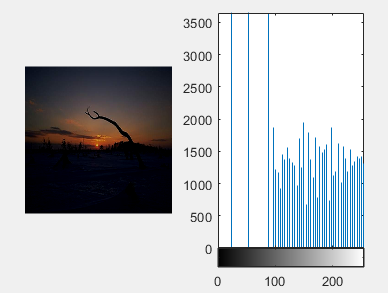
代码：

C=histeq(B); %对图像B 进行均衡化

imshow(C); %显示图像

imhist(C)； %得到均衡化后的灰度直方图

截图：

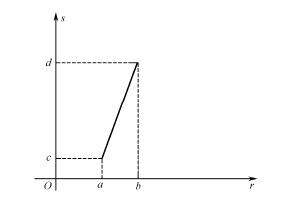


结论：

histeq()函数将图像进行均衡化

（3）线性与非线性灰度映射。

线性映射函数s=T(r)：



非线性映射函数s=T(r)：

*s* = c log(1 + *r*)

灰度倒置s=T(r)：

s =255-r

二值化：确定一阈值*r*1，0<*r*1<255。*r*>*r*1 的灰度值置白，*r*<*r*1 的灰度值置黑。

代码：

I=imread(' C:\Users\Administrator\Desktop\1.png ');

figure;

subplot(1,3,1);

imshow(I);

title('原图');

J=imadjust(I,[0.3;0.6],[0.1;0.9]);

%设置输入/输出变换的灰度级范围，*a*=0.3, *b*=0.6, *c*=0.1, *d*=0.9。

subplot(1,3,2);

imshow(J);

title('线性映射');

I1=double(I);

I2=I1/255;

C=2;% 设置非线性映射函数的参数c=2。

K=C\*log(1+I2); %求图像的对数变换

subplot(1,3,3);

imshow(K);

title('非线性映射');

M=255-I; %将此图像取反

figure;

subplot(1,3,1);

imshow(M);

title('灰度倒置');

N1=im2bw(I,0.4); %将此图像二值化，阈值为0.4

N2=im2bw(I,0.7); %将此图像二值化，阈值为0.7

subplot(1,3,2);

imshow(N1);

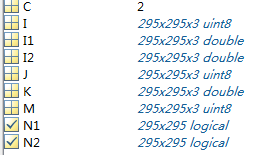
title('二值化阈值0.4');

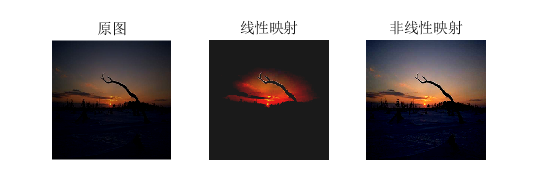
subplot(1,3,3);

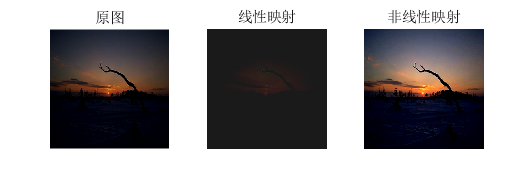
imshow(N2);

title('二值化阈值0.7');

截图：

1）分析调试上面代码，查看其中各步骤变量类型与值的变化，分析程序功能。

2）分别查看*d-c*大于和小于*b-a*的情况，观察输出图像的对比度增大减小情况。

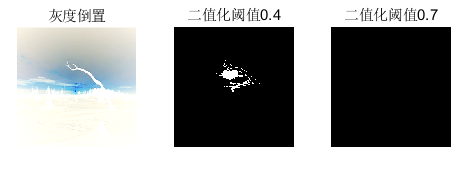
D=0.2 

3）分析查看对数非线性映射使图像的高、低灰度级的变化情况。

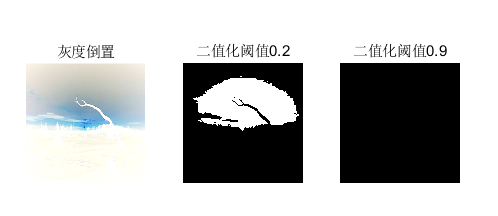
C=1



4）设置不同参数进行实验，思考线性映射与对数非线性映射的图像处理效果的区别。

4）查看灰度倒置的图像效果。

5）选择不同的阈值，观察阈值对图像二值化的影响。



结论：

imadjust(I,[0.3;0.6],[0.1;0.9])实现像素亮度的线性映射

C\*log(1+I2)1可以实现像素亮度的非线性映射

255-I，操作可以实现像素的灰度倒置

im2bw(I,阈值)可以实现将图片依阈值进行二值化

（4）filter2 函数实现均值滤波。

代码：

a=imread(' C:\Users\Administrator\Desktop\1.png');

P1=imnoise(M,'gaussian',0.02);

P2=imnoise(M,'salt & pepper',0.09);

I=rgb2gray(a);

imshow(I);

K1=filter2(fspecial('average',3),I)/255; %3×3 的均值滤波

K2=filter2(fspecial('average',5),I)/255; %5×5 的均值滤波

K3=filter2(fspecial('average',7),I)/255; %7×7 的均值滤波

figure,imshow(K1);

figure,imshow(K2);

figure,imshow(K3);

1. medfilt2 函数实现中值滤波。

代码：

a=imread(' C:\Users\Administrator\Desktop\1.png');

P1=imnoise(M,'gaussian',0.02);

P2=imnoise(M,'salt & pepper',0.09);

I=rgb2gray(a);

imshow(I);

K1=medfilt2(I,[3,3]); %3×3 中值滤波

K2=medfilt2(I,[5,5]); %5×5 中值滤波

K3=medfilt2(I,[7,7]); %7×7 中值滤波

figure,imshow(K1);

figure,imshow(K2);

figure,imshow(K3);

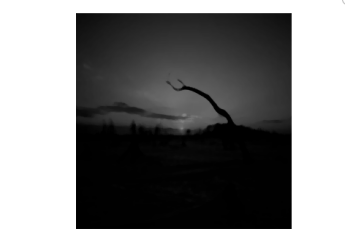
截图：

1. 增大滤波器模板，观察图像效果变化，比较均值滤波和中值滤波的效果差别。

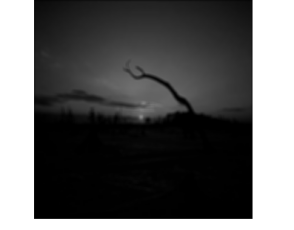
均值：

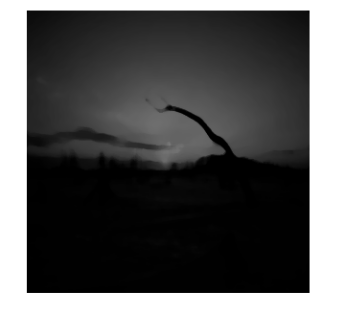
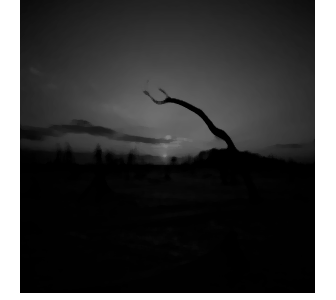
   

中值：



2）加入高斯噪声（P1=imnoise(M,'gaussian',0.02);）和椒盐噪声（P2=imnoise(M,'salt & pepper',0.09);）观察不同模板尺寸下，均值滤波和中值滤波对两种噪声的滤波效果差别。



结论：

medfilt2(picture,[x,x])实现x,x维的中值滤波

P1=imnoise(M,'gaussian',0.02)可加入高斯噪声;方差越大，图片越模糊。

P2=imnoise(M,'salt & pepper',0.09)可加入椒盐噪声。

（6）fft2 函数和ifft2 函数分别是计算二维的傅里叶变换和反变换。

代码：

f=zeros(100,100);

f(20:70,40:60)=1;

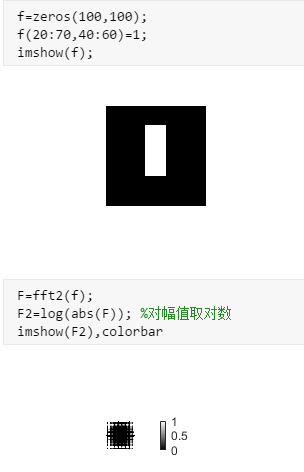
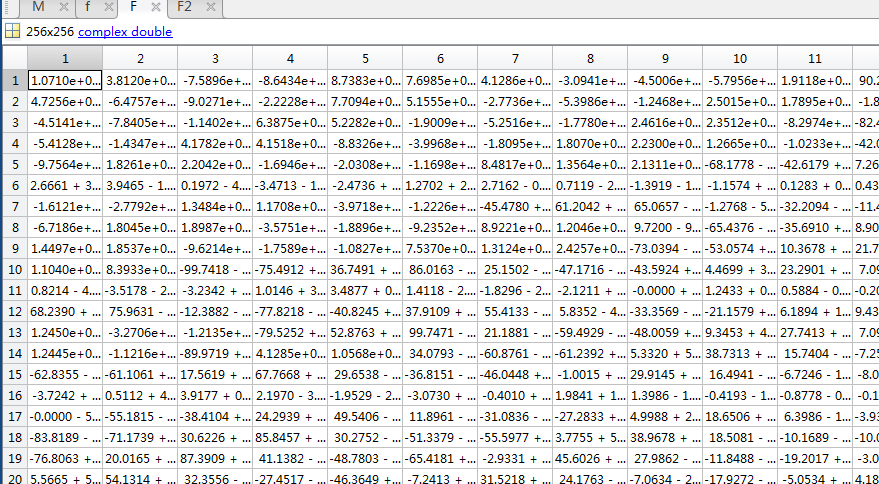
imshow(f);

F=fft2(f);

F2=log(abs(F)); %对幅值取对数

imshow(F2),colorbar

截图：

结论：

fft2 函数和ifft2 函数分别是计算二维的傅里叶变换和反变换。

（7）fftshift 函数实现了补零操作和改变图像显示象限。

代码：

f=zeros(100,100);

f(20:70,40:60)=1;

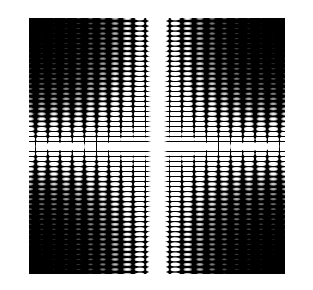
imshow(f);

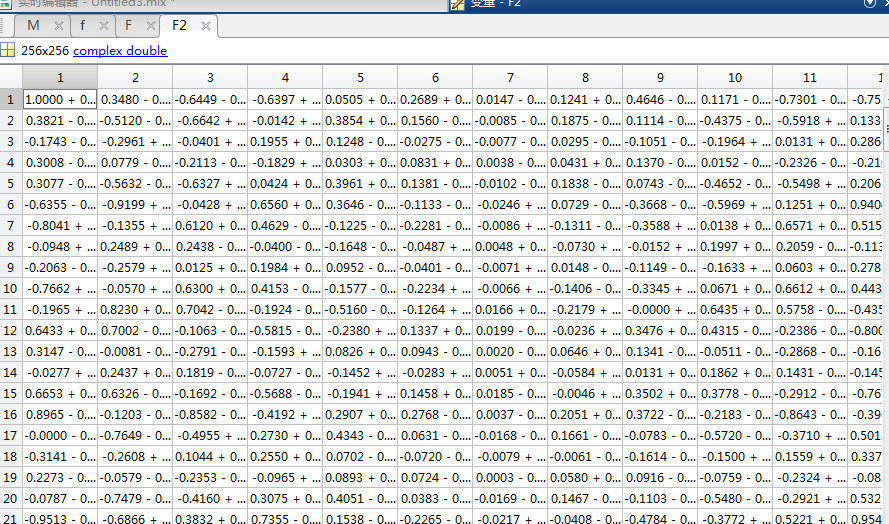
F=fft2(f,256,256);

F2=fftshift(F); %实现补零操作

imshow(log(abs(F2)));

截图：



结论：

fftshift针对频域，将FFT的DC分量移到频谱中心，重新排列fft，fft1和……ftn的输出结果。

（8）edge 函数用于提取图像的边缘。

代码：

I=imread('cameraman.tif');

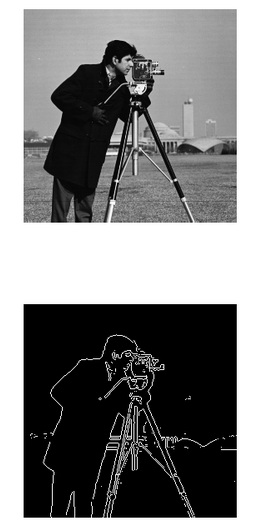
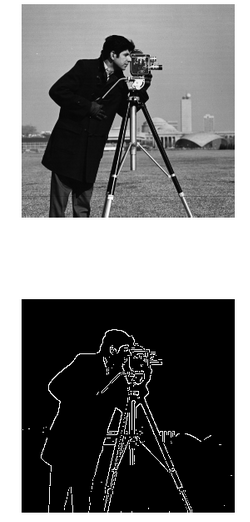
%I=rgb2gray(I);

BW=edge(I,'canny/rewitt/obel/oberts/og');

imshow(I),figure,imshow(BW,[]);

截图：

**prewitt** **Canny** 

**Sobel****roberts****log** 

**题目2**

1. **图像采样**
2. 运行并分析以上代码，学习matlab中for语句的使用方法，并逐行解释上述代码。
3. 尝试对图像的16倍减采样，在同一窗口中查看其原图、4倍、16倍减采样效果。

代码：

a = imread(' C:\Users\Administrator\Desktop\2.png');

b = rgb2gray(a);

[wid,hei]=size(b);

%4 倍减采样

quartimg = zeros(wid/2+1,hei/2+1);

i1 = 1;

j1 = 1;

for i=1:4wid

for j=1:4hei

quartimg(i1,j1) = b(i,j);

j1 = j1 + 1;

end

i1 = i1 + 1;

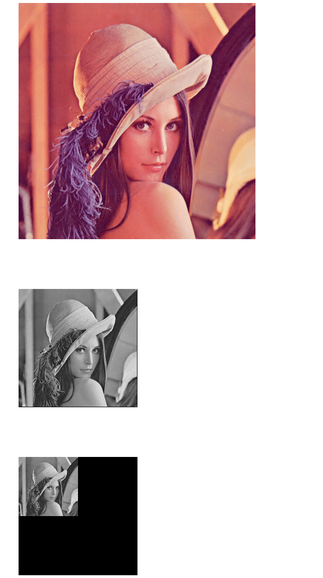
j1 = 1;

end

figure

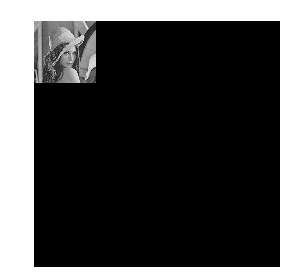
imshow(uint8(quartimg))

截图：



1. 思考将一幅图如果进行4 倍、16 倍和64 倍增采样会出现什么情况？

（有可能出现错误：wid/2+1,hei/2+1不是整数，这是需要将原图像resize成偶数长宽）

结论：

1、Matlab 中的for语句可以从s以步长b遍历到e

2、不同倍数的采样，以4倍采样为例，即隔一个点采样一次，这样就可以使得图像的大小减小为原图的1 /4。

3、若对一幅图进行4 倍、16 倍和64 倍增采样，会使得图像的尺寸缩为原来的1/2,1/4,1/8。

**2、直方图规定化处理（选做）**

代码：

I=imread('C:\Users\Xin\Desktop\f1.jpg');

clear all

close all

%

I=double(imread('cameraman.tif'));

figure,imshow(I,[])

N=32;

Hist\_image=hist(I(:),N);

Hist\_image=Hist\_image/sum(Hist\_image);

Hist\_image\_cumulation=cumsum(Hist\_image);

figure,stem([0:N-1],Hist\_image);

%1.

Index=0:N-1;

%

Hist{1}=exp(-(Index-4).^2/8);

Hist{1}=Hist{1}/sum(Hist{1});

Hist\_cumulation{1}=cumsum(Hist{1});

figure,stem([0:N-1],Hist{1})

%

Hist{2}=abs(2\*N-1-2\*Index);

Hist{2}=Hist{2}/sum(Hist{2});

Hist\_cumulation{2}=cumsum(Hist{2});

figure,stem([0:N-1],Hist{2})

%2.

for m=1:2 %

Image=I;

%2.1

for k=1:N

Temp=abs(Hist\_image\_cumulation(k)-Hist\_cumulation{m});

[Temp1,Project{m}(k)]=min(Temp);

end

%

for k=1:N

Temp=find(Project{m}==k);

if isempty(Temp)

Hist\_result{m}(k)=0;

else

Hist\_result{m}(k)=sum(Hist\_image(Temp));

end

end

figure,stem([0:N-1],Hist\_result{m});

%

Step=256/N;

for k=1:N

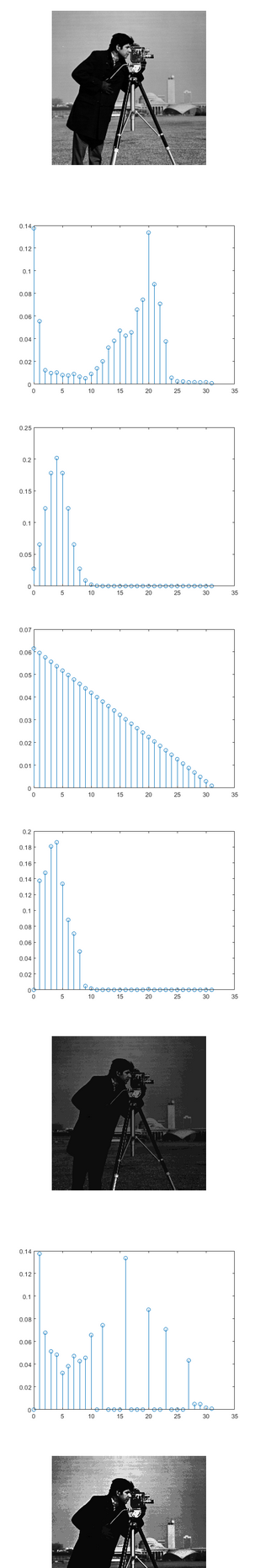
Index=find(I>=Step\*(k-1)&I<Step\*k);

Image(Index)=Project{m}(k);

end

figure,imshow(Image,[])

end

2）分析图像处

**实验二题目1的运行代码、使用参数、实验结果截图、实验结论：**

**题目1、车牌分割及图像傅里叶变换性质验证**



代码：

I = imread(' C:\Users\Administrator\Desktop\carplate.jpg');

figure,imshow(I);

%二值化

N1=im2bw(I,0.4);

figure,imshow(N1)

flag = 0;

[m n] = size(N1);

base = zeros(256,256);

new\_pic = base;

work = 0

while n>1

left = 1;wide = 0;

falg = 0;

while sum(N1(:,wide + 1)) > 0

wide = wide +1;

flag = 1;

end

if flag == 1 & wide > 10

falg = 0;

temp = imcrop(N1,[1 1 wide m]);

temp1 = imresize(temp,[100 150]);

path = strcat(num2str(work),'0.png');

imwrite(temp1,path);

for xx = 78:177

for yy = 52:201

new\_pic(xx,yy) = base(xx,yy) + temp1(xx-77,yy-51);

end

end

figure,imshow(new\_pic);

path = strcat(num2str(work),'.png');

imwrite(new\_pic,path);

work = work +1;

end

temp = imcrop(N1,[wide+2 0 n m]);

N1 = temp;

[m n] = size(N1);

end

figure;

for i = 0:7

path = strcat(num2str(i),'.png');

f = imread(path);

F=fft2(f); %傅里叶变换

F1=log(abs(F)+1); %取模并进行缩放

subplot(2,4,i+1),imshow(F1,[]);title('傅里叶变换频谱图');

end

II = imread('4.png');

subplot(2,5,1),imshow(II);

F=fft2(II); %傅里叶变换

F1=log(abs(F)+1); %取模并进行缩放

subplot(2,5,6),imshow(F1,[]);title('傅里叶变换频谱图');

path\_1 = '40.png';

I = imread('40.png');

base = zeros(256,256);

new\_pic = base;

for xx = 128:227

for yy = 52:201

new\_pic(xx,yy) = base(xx,yy) + I(xx-127,yy-51);

end

end

subplot(2,5,2),imshow(new\_pic);

F=fft2(new\_pic); %傅里叶变换

F1=log(abs(F)+1); %取模并进行缩放

subplot(2,5,7),imshow(F1,[]);title('向下平移');

J=imrotate(new\_pic,90,'bilinear');

subplot(2,5,3),imshow(J);

F=fft2(J); %傅里叶变换

F1=log(abs(F)+1); %取模并进行缩放

subplot(2,5,8),imshow(F1,[]);title('向下平移且旋转');

fangda = imresize(I,1.5);

fangda\_new\_pic = zeros(256,256);

for xx =53:202

for yy = 17:236

fangda\_new\_pic(xx,yy) = base(xx,yy) + fangda(xx-52,yy-16);

end

end

subplot(2,5,4),imshow(fangda\_new\_pic);

F=fft2(fangda\_new\_pic); %傅里叶变换

F1=log(abs(F)+1); %取模并进行缩放

subplot(2,5,9),imshow(F1,[]);title('放大1.5倍');

r\_180\_II = imrotate(II,90,'bilinear');

subplot(2,5,5),imshow(r\_180\_II);

F=fft2(fangda\_new\_pic); %傅里叶变换

F1=log(abs(F)+1); %取模并进行缩放

subplot(2,5,10),imshow(F1,[]);title('取反');

使用参数：

二值化阈值为0.4

Imcrop(N1,[wide+2 0 n m ])

F1=log(abs(F)+1)

imrotate(new\_pic,90,'bilinear')

log(abs(F)+1)

imrotate(II,90,'bilinear');

截图：

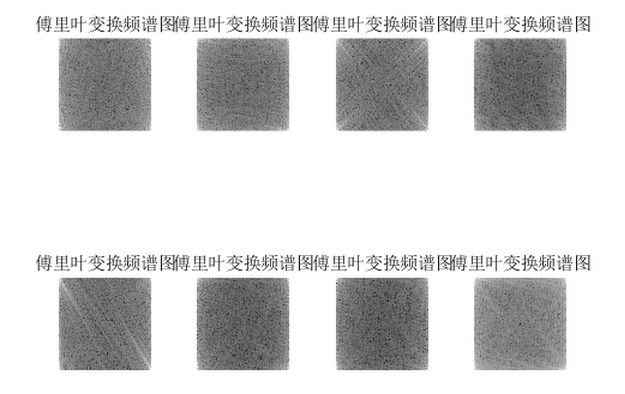
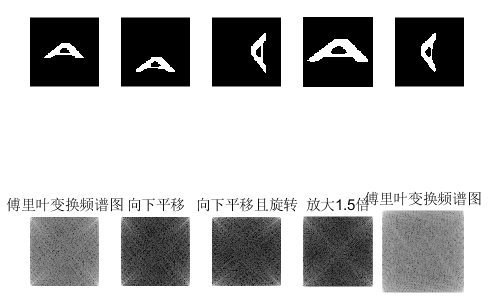
1）针对以上车牌图像，计算其图像直方图，选择合适的阈值进行二值化，而后基于二值图像编程分割出每个字符，将其缩放为统一标准大小100\*150，而后放入256\*256的黑色背景图像的中间位置。

2）查看生成的各个字符图像（256\*256）的傅里叶频谱图。

3）选择其中一个字符图像（256\*256），将其中的100\*150字符平移一定位置，再次计算其傅里叶频谱；对平移后的字符图像（256\*256）进行90度旋转，再次计算其频谱图；将图像中的字符放大或缩小一定量（仍放在256\*256黑色背景中），再次计算其频谱图；将图像取反，再次计算其频谱图。在同一窗口中显示5张原图与5张频谱图，查看傅里叶变换的平移、旋转、翻转特性，总结频谱图高低频成分与图像空间梯度的关系。



结论：

切割图像时，可以通过遍历某一列的全部像素的值，如果有点的值为255则认为这一列是一个字符的边界，同理找出其他的边界，然后就分割出了全部的字符。

然后通过两个for语句，将切割下来的图片放入256\*256的背景图上。低频信息反映了图像图像灰度随空间的缓慢变化，而高频信息反映了图像灰度随空间的快速变化，即相邻区域的灰度值变化剧烈。

**实验二题目2的运行代码、使用参数、实验结果截图、实验结论：**

dct2 函数实现离散余弦变换。与之对应，idct2 函数实现图像的二维逆离散余弦变换。

代码：

I = imread('cameraman.tif');

figure(1)

imshow(I)

%I = rgb2gray(RGB);

J = dct2(I);

figure(2)

imshow(log(abs(J)),[])

figure(3);

J(abs(J) < 10) = 0;

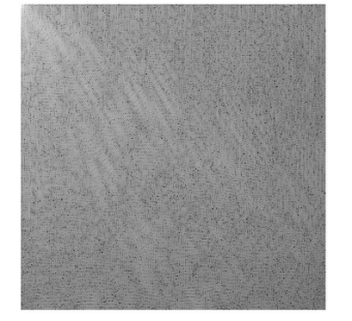
K = idct2(J)/255;

imshow(K)

截图：

1）逐行分析以上离散余弦变换程序，查看各步骤变量类型与值的变化，分析程序功能。

2）总结余弦变换系数的能量集中特性。分别选取不同的高、低频丢弃范围，观测恢复图像的质量变化。

  j<10

j<40 j>60 

结论：

图形大多数信号的的能量都集中在离散余弦变换后的低频部分。一般来说，选取的丢弃范围的频率越高，图像越模糊，质量越差。