

基于 MATLAB 的图像处理的课程设计(2009-01-16 17:02:21)

标签:matlab 图像处理 课程设计 文化

基于 MATLAB 的图像处理的课程设计

一、课程设计的目的:

综合运用 MATLAB 工具箱实现图像处理的 GUI 程序设计。

二、课程设计的基本要求

- 1) 熟悉和掌握 MATLAB 程序设计方法
- 2) 掌握 MATLAB GUI 程序设计
- 3) 熟悉 MATLAB 图像处理工具箱
- 4) 学会运用 MATLAB 工具箱对图像进行处理和分析

三、课程设计的内容

要求利用 MATLAB GUI 设计实现图像处理的图形用户界面, 利用 MATLAB 图像处理工具箱实现以下的图像处理功能:

双击打开 MATLAB 7.0→File→New→GUI→单击, 调整 axes1 大小→单击 OK, 调整按钮大小和颜色, 修改名称→再建 axes2→单击 OK, 调整按钮大小和颜色, 修改名称→保存
→View→M-file Edit→写程序

1) 图像的读取和保存。

在 function open_Callback(hObject, eventdata, handles) 后面输入如下程序

```
[name,path]=uigetfile('*.*.','');
```

```
file=[path,name];
```

```
axes(handles.axes1);
```

```
x=imread(file); %读取图像
```

```
handles.img=x;

guidata(hObject, handles);

imshow(x);           %显示图像

title(' 打开');
```

在 function save_Callback(hObject, eventdata, handles)后面输入如下程序

```
[name,path]=uigetfile('*.','');

file=[path,name];

axes(handles.axes1);

x=imread(file);

handles.img=x;

guidata(hObject, handles);

imshow(x);

imwrite(x,'new.jpg');    %保存图像

title(' 保存');
```

2) 设计图形用户界面, 让用户能够对图像进行任意的亮度和对比度变化调整, 显示和对比变换前后的图像。

在 function liangdu_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);

x=(handles.img);

y=imadjust(x,[0.15 0.9], [0 1]);    %增亮图像
```

```
imshow(y);  
  
title('亮度')
```

在 function huidu_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);  
  
x=rgb2gray(handles.img);           %RGB 图像转换为灰度图像  
  
imshow(x);  
  
title('灰度')
```

3) 设计图形用户界面, 让用户能够用鼠标选取图像感兴趣区域, 显示和保存该选择区域。

在 function cut_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);  
  
x=imcrop(handles.img);             %截图  
  
imshow(x);  
  
imwrite(x, 'cut.jpg');              %保存图像  
  
title('截图');
```

4) 编写程序通过最近邻插值和双线性插值等算法将用户所选取的图像区域进行放大和缩小整数倍的操作, 并保存, 比较几种插值的效果。

在 function nearfangda_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);  
  
x=imresize(handles.img, 2, 'nearest');           %最近邻插值法放大
```

```
imshow(x);

imwrite(x, 'nearfangda. jpg')

title('最近邻插值法放大');
```

在 function nearsuoxiao_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);

x=imresize(handles.img, 0.5, 'nearest');           %最近邻插值法缩小

imshow(x);

imwrite(x, 'nearsuoxiao. jpg')

title('最近邻插值法缩小');
```

在 function doublefangda_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);

x=imresize(handles.img, 5, 'bilinear');           %双线性插值法放大图像

imshow(x);

imwrite(x, 'doublefangda. jpg');

title('双线性插值法放大');
```

在 function doublesuoxiao_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);

x=imresize(handles.img, 0.5, 'bilinear');           %双线性插值法缩小图像
```

```

imshow(x);

imwrite(x, 'doublesuoxiao.jpg');

title('双线性插值法缩小');

```

5) 图像直方图统计和直方图均衡, 要求显示直方图统计, 比较直方图均衡后的效果。

在 function zhifangtu_Callback(hObject, eventdata, handles) 后面输入如下程序

```

set(handles.axes2, 'HandleVisibility', 'ON');

axes(handles.axes2);

x=imhist(handles.img); %直方图统计

x1=x(1:10:256);

horz=1:10:256;

bar(horz, x1);

%axis([0 255 0 150000]);

set(handles.axes2, 'xtick', 0:50:255);

%set(handles.axes2, 'ytick', 0:2000:15000);

set(handles.axes2, 'HandleVisibility', 'OFF');

```

在 function junheng_Callback(hObject, eventdata, handles) 后面输入如下程序

```

set(handles.axes2, 'HandleVisibility', 'ON');

axes(handles.axes2);

h=histeq(handles.img); %直方图均衡

imshow(h);

```

```
%set(handles.axes2,'ytick',0:2000:15000);
```

```
set(handles.axes2,'HandleVisibility','OFF');
```

原创力文档

max.book118.com

6)能对图像加入各种噪声,并通过几种滤波算法实现去噪并显示结果。比较去噪效果。

在 function gaussian_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);
```

```
x=(handles.img);
```

```
y=imnoise(x,'gaussian',0,0.05); %加高斯噪声
```

```
imshow(y);
```

```
imwrite(y,'gaussian.jpg');
```

```
title('加高斯噪声')
```

在 function salt_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);
```

```
x=(handles.img);
```

```
y=imnoise(x,'salt & pepper',0.04); %加椒盐噪声
```

```
imshow(y);
```

```
imwrite(y,'salt.jpg');
```

```
title('加椒盐噪声')
```

在 function medfilt_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);
```

```

x=(handles.img);

y=imnoise(x,'salt & pepper',0.04);           %加椒盐噪声

z=medfilt2(y,[5 5],'symmetric');              %中值滤波

imshow(z);

imwrite(z,'medfilt.jpg');

title('中值滤波');

```

在 function wiener_Callback(hObject, eventdata, handles) 后面输入如下程序

```

axes(handles.axes2);

x=(handles.img);

y=imnoise(x,'gaussian',0,0.05);               %加高斯噪声

z=wiener2(y,[5 5]);                           %自适应滤波

imshow(z);

imwrite(z,'wiener.jpg');

title('自适应滤波')

```

7) 频谱处理, 能够分析图像频谱, 显示频谱图。

在 function pinputu_Callback(hObject, eventdata, handles) 后面输入如下程序

```

axes(handles.axes2);

x=(handles.img);

J2=fft2(x);                                   %傅立叶变换

K2=fftshift(J2);                             %转换数据矩阵

```

```
imshow(log(abs(K2)), []); %显示频谱图
```

```
title(' 频谱图');
```

8) 设计巴特沃斯低通滤波对图像进行低通滤波处理, 显示结果。

在 function butterdi_Callback(hObject, eventdata, handles) 后面输入如下程序

```
axes(handles.axes2);
```

```
x=(handles.img);
```

```
y1=imnoise(x, 'salt & pepper'); % 叠加椒盐噪声
```

```
f=double(y1); % 数据类型转换, MATLAB 不支持图像的无符号整型的计算
```

```
g=fft2(f); % 傅立叶变换
```

```
g=fftshift(g); % 转换数据矩阵
```

```
[M,N]=size(g);
```

```
nn=2; % 二阶巴特沃斯(Butterworth)低通滤波器
```

```
d0=50; %截止频率为 50
```

```
m=fix(M/2); n=fix(N/2);
```

```
for i=1:M
```

```
    for j=1:N
```

```
        d=sqrt((i-m)^2+(j-n)^2);
```

```
        h=1/(1+0.414*(d/d0)^(2*nn)); % 计算低通滤波器传递函数
```

```
        result(i,j)=h*g(i,j);
```

```
    end
```

```

end

result=ifftshift(result);

y2=ifft2(result);

y3=uint8(real(y2));

imshow(y3); % 显示滤波处理后的图像

imwrite(y3,'butterdi.jpg');

title('巴特沃斯低通滤波')

```

9) 设计高斯高通滤波器, 显示结果。

```

axes(handles.axes2);

x=(handles.img);

y1=imnoise(x,'gaussian'); %加高斯噪声

f=double(y1); % 数据类型转换

k=fft2(f); % 傅立叶变换

g=fftshift(k); % 转换数据矩阵

[M,N]=size(g);

nn=2;

d0=25; %截止频率为 25

m=fix(M/2); n=fix(N/2);

for i=1:M

    for j=1:N

        d=sqrt((i-m)^2+(j-n)^2); % 计算高通滤波器传递函数

```

```

        if d<=d0

            h=0;

        else h=1;

        end

        result(i,j)=h*g(i,j);

    end

end

result=ifftshift(result);

y2=ifft2(result);

y3=uint8(real(y2));

imshow(y3);                % 显示滤波处理后的图像

imwrite(y3,'gaussiangao.jpg');

title('高斯高通滤波')

```

四、课程设计的体会

通过这次基于 MATLAB 的图像处理的课程设计,熟悉和掌握了 MATLAB 程序设计方法、MATLAB GUI 程序设计、MATLAB 图像处理工具箱,了解了图形用户界面的制作的设计原则和一般步骤:简单性、一致性、习惯性、响应要迅速、连续等原则,学会了运用 MATLAB 工具箱对图像进行处理和分析。放大缩小图像时使用了两种方法,最近邻插值法比双线性插值法运行时间快;就效果而言,使用最邻近插值法确实出现了锯齿,但并不明显,而双线性插值则几乎没有什么改变。在做到加高斯噪声和椒盐噪声时,程序运行较慢,我认为这可能有两方面的原因:一是程序的问题,二是计算机的反应速度。通过不同的加噪方法,得到了不同的加噪效果;不同的滤波方法,得到的滤波效果图也不同。中值滤波较自适应滤波运行速度快。下面是通过不同的方法放大、缩小、加噪、滤波后保存的图像:(其中 macs、bacteria 为原图)



五、附录

1、参考书目：

- 1) 《MATLAB 语言及其在电子信息工程中的应用》 王洪元主编 清华大学出版社
- 2) 《MATLAB 在数字图像处理中的应用》 机械工业出版社

2、源代码

采用灰度变换的方法增强图像的对比度：

```
I=imread('rice.tif');
```

```
imshow(I);
```

```
figure, imhist(I);
```

```
J=imadjust(I, [0.15 0.9], [0 1]);
```

```
figure, imshow(J);
```

```
figure, imhist(J);
```

模拟图像受高斯白噪声和椒盐噪声的影响:

```
I=imread('eight.tif');
```

```
imshow(I) ;
```

```
J1=imnoise(I, 'gaussian', 0, 0.02);           % 叠加均值为 0, 方差为 0.02 的高斯噪声
```

```
figure, imshow (J1);
```

```
J2=imnoise(I, 'salt & pepper', 0.04);        % 叠加密度为 0.04 的椒盐噪声。
```

```
figure, imshow(J2);
```