图像处理作业三

软工一班
陈仟雅
3017218053
2019.12.05

题目一

对 LoG 的数学形式进行数学推导。

分析

图像中的边缘，角点，拐点都可以称之为图像的特征点，Laplacian 算子利用求图像的二阶微分极值点来获得图像的边缘，二阶微分定义如下：

$$\Delta = \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

但是由于 laplacian 算子对噪声敏感，因此我们首先利用二维高斯核函数对图像进行模糊去噪，然后求二阶微分极值点来获得图像中边缘位置，得：

$$I'(x,y) = g(x,y) * I(x,y) = \iint_D g（x-u, y-v）I(u,v)dudv$$

$$\Delta|I'(x,y)| = \Delta|g(x,y) * I(x,y)| = \frac{\partial^2 \iint_D g（x-u, y-v）I(u,v)dudv}{\partial x^2} + \frac{\partial^2 \iint_D g（x-u, y-v）I(u,v)dudv}{\partial y^2}$$

$$\Delta|I'(x,y)| = \iint_D \frac{\partial^2 g（x-u, y-v）}{\partial x^2} I(u,v)dudv + \iint_D \frac{\partial^2 g（x-u, y-v）}{\partial y^2} I(u,v)dudv$$

$$\Delta|I'(x,y)| = \frac{\partial^2 g（x,y）}{\partial x^2} * I(x,y) + \frac{\partial^2 g（x,y）}{\partial y^2} * I(x,y)$$

根据卷积结合律得：

$$\Delta|I'(x,y)| = (\frac{\partial^2 g（x,y）}{\partial x^2} + \frac{\partial^2 g（x,y）}{\partial y^2}) * I(x,y) \qquad (1)$$

由（1）式可知，对图像进行高斯核函数卷积再求二阶微分等价于对高斯核函数先求二阶微分核后再用该核与图像进行卷积。我们定义 LoG 算子为二阶微分后的高斯核函数

$$LoG = \Delta|g(x,y)| = \frac{\partial^2 g\ (x,y)}{\partial x^2} + \frac{\partial^2 g\ (x,y)}{\partial y^2} \qquad (2)$$

$$= \frac{\partial^2(\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}})}{\partial x^2} + \frac{\partial^2(\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}})}{\partial y^2}$$

$$= (\frac{x^2}{\sigma^4} - \frac{1}{\sigma^2})\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} + (\frac{y^2}{\sigma^4} - \frac{1}{\sigma^2})\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= (\frac{x^2+y^2}{\sigma^4} - \frac{2}{\sigma^2})\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (3)$$

我们知道归一化二维高斯核

$$g\ (x,y)\ = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

该函数对σ进行求导得

$$\frac{\partial g\ (x,y)}{\partial \sigma} = \frac{\partial(\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}})}{\partial \sigma}$$

$$= \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \frac{x^2+y^2}{\sigma^3} + \frac{-2}{2\pi\sigma^3} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$= \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} (\frac{x^2+y^2}{\sigma^3} - \frac{2}{\sigma}) \qquad (4)$$

观察（3）式与（4）式，我们发现其在结果上只相差一个常数σ即

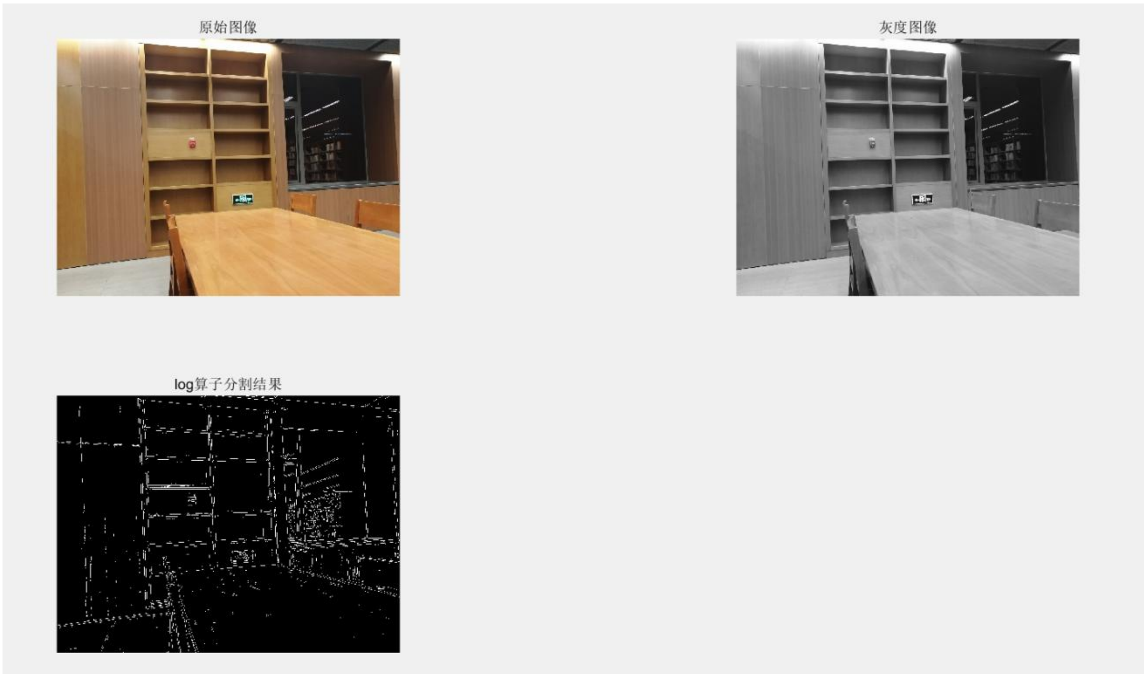
$$\sigma LoG = \frac{\partial g}{\partial \sigma} \qquad (5)$$

代码段

```
I=imread('a.jpg');            %读取当前路径下的图片
subplot(2,2,1);
imshow(I);
title('原始图像');
I1=rgb2gray(I);
subplot(2,2,2);
imshow(I1);
title('灰度图像');
I2=edge(I1,'log');
subplot(2,2,3);
imshow(I2);
```

```
title('log算子分割结果');
```

测试结果



题目二

实现最小二乘法、RANSAC 法、 霍夫变换法

- 对于直线方程 y=ax+b，生成一系列纵坐标符合高斯分布的点，再人工加入一系列的 outlier, 使用上述三种方法拟合一条直线。
- 找到一幅实际图像(较简单的)，使用一阶导数或二阶导数找出边缘点，使用上述三种方法，找到其中的直线。

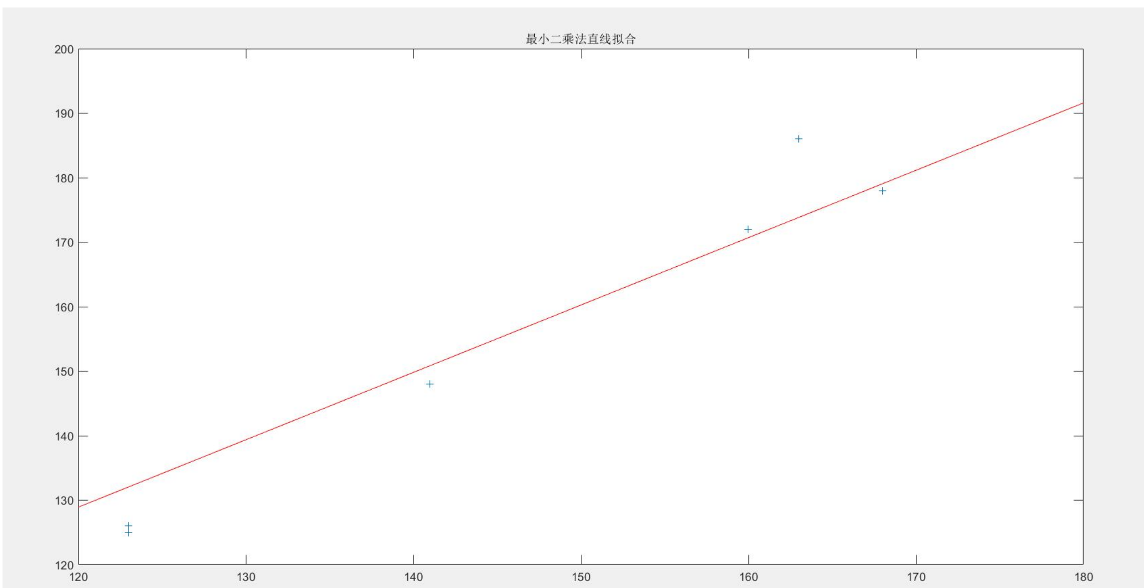
分析

1、

（1）最小二乘法直线拟合

```
%原始数据
X=[163 123 160 123 141 168];
Y=[186 126 172 125 148 178];
n=5;                    %一共5个变量

x2=sum(X.^2);           % 求Σ(xi^2)
x1=sum(X);              % 求Σ(xi)
```

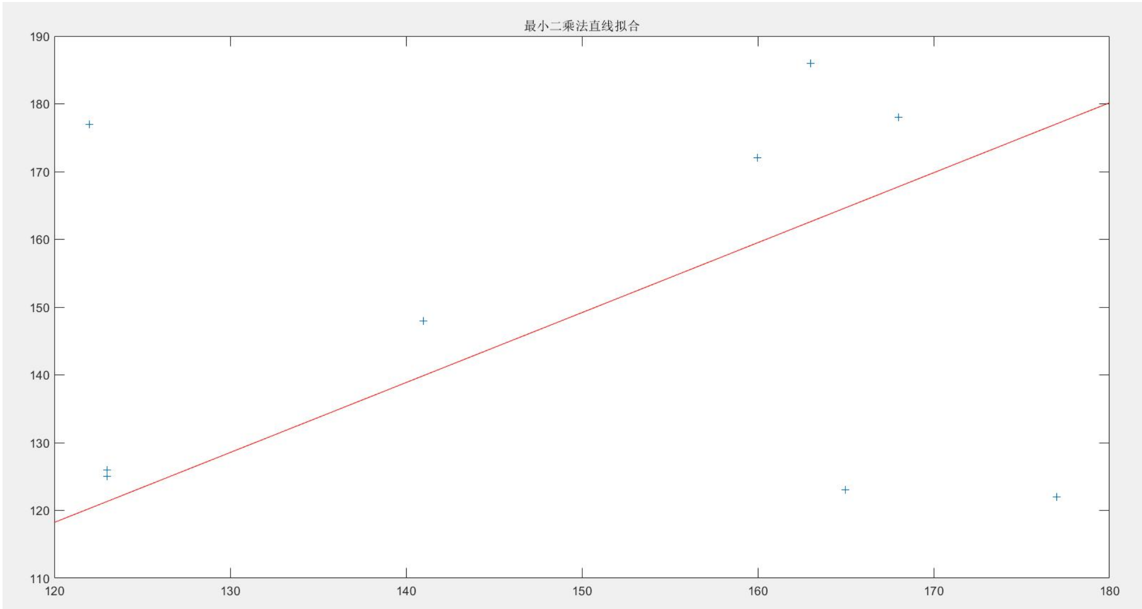
```
x1y1=sum(X.*Y);      % 求Σ(xi*yi)
y1=sum(Y);           % 求Σ(yi)

a=(n*x1y1-x1*y1)/(n*x2-x1*x1);       %解出直线斜率 b=(y1-a*x1)/n
b=(y1-a*x1)/n;                       %解出直线截距
%作图
% 先把原始数据点用蓝色十字描出来
figure
plot(X,Y,'+');
hold on
% 用红色绘制拟合出的直线
px=linspace(120,165,45);%这里直线区间根据自己实际需求改写
py=a*px+b;
plot(px,py,'r');title('最小二乘法直线拟合');
```



人工加入一系列的 outlier：

修改如下

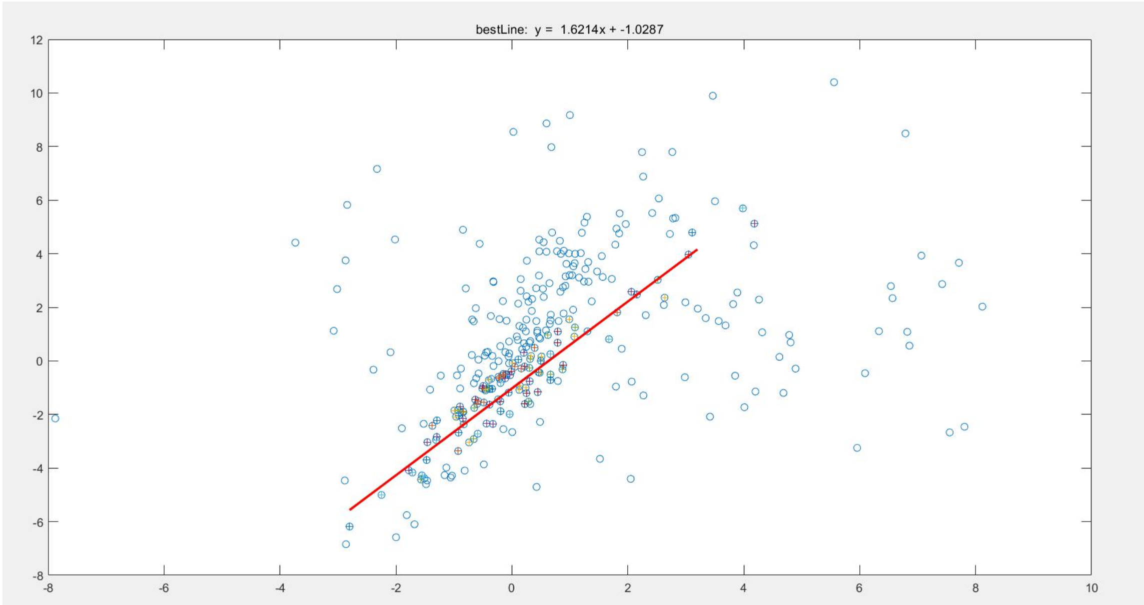
```
%原始数据
X=[163 123 160 123 141 168 177 122 165];
Y=[186 126 172 125 148 178 122 177 123];
```



（2）**RANSAC** 直线拟合

```
%%%二维直线拟合
%%%生成随机数据
mu=[0 0];  %均值
S=[1 2.5;2.5 8];  %协方差
data1=mvnrnd(mu,S,200);   %产生 200 个高斯分布数据
%外点
mu=[2 2];
S=[8 0;0 8];
data2=mvnrnd(mu,S,100);     %产生 100 个噪声数据
%合并数据
data=[data1’,data2’];
iter = 100;

 %%% 绘制数据点
 figure;plot(data(1,:),data(2,:),’o’);hold on; % 显示数据点
 number = size(data,2); % 总点数
 bestParameter1=0; bestParameter2=0; % 最佳匹配的参数
 sigma = 1;
 pretotal=0;     %符合拟合模型的数据的个数

 for i=1:iter
 %%% 随机选择两个点
     idx = randperm(number,2);
     sample = data(:,idx);
```

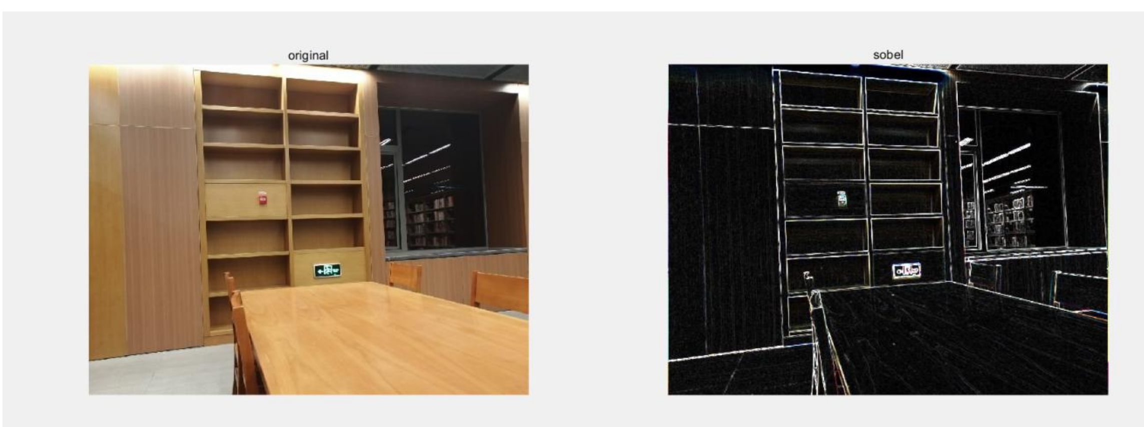
```
    %%%拟合直线方程 y=ax+b
    line = zeros(1,3);
    x = sample(:, 1);
    y = sample(:, 2);

    k=(y(1)-y(2))/(x(1)-x(2));      %直线斜率
    b = y(1) - k*x(1);
    line = [k -1 b];

    mask=abs(line*[data; ones(1,size(data,2))]);    %求每个数据到拟合直线的距离
    total=sum(mask<sigma);              %计算数据距离直线小于一定阈值的数据的个数

    if total>pretotal            %找到符合拟合直线数据最多的拟合直线
        pretotal=total;
        bestline=line;          %找到最好的拟合直线
   end
 end
 %显示符合最佳拟合的数据
mask=abs(bestline*[data; ones(1,size(data,2))])<sigma;
hold on;
k=1;
for i=1:length(mask)
    if mask(i)
        inliers(1,k) = data(1,i);
        k=k+1;
        plot(data(1,i),data(2,i),'+');
    end
end

%%% 绘制最佳匹配曲线
bestParameter1 = -bestline(1)/bestline(2);
bestParameter2 = -bestline(3)/bestline(2);
xAxis = min(inliers(1,:)):max(inliers(1,:));
yAxis = bestParameter1*xAxis + bestParameter2;
plot(xAxis,yAxis,'r-','LineWidth',2);
title(['bestLine:  y =  ',num2str(bestParameter1),'x + ',num2str(bestParameter2)]);
```



2、

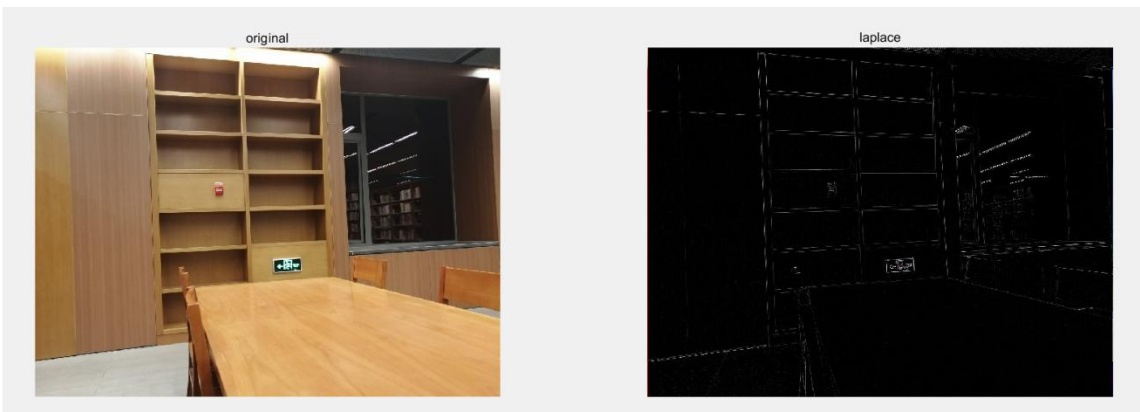
（1）使用一阶导数找出边缘点

```
[I,map]=imread('a.jpg ');
[H,W]=size(I);
M=double(I);
J=M;
for i=2:H-1
for j=2:W-1
J(i,j)=abs(M(i-1,j+1)-M(i-1,j-1)+2*M(i,j+1)-2*M(i,j-1)+M(i+1,j+1)-M(i+1,j-1))+abs(
M(i-1,j-1)-M(i+1,j-1)+2*M(i-1,j)-2*M(i+1,j)+M(i-1,j+1)-M(i+1,j+1));
end
end
subplot(1,2,1);imshow(I);title('original');
subplot(1,2,2);imshow(uint8(J));title('sobel');
```



使用二阶导数找出边缘点：

```
I=imread('a.jpg');
[H,W]=size(I);
M=double(I);
J=M;
for i=2:H-1
for j=2:W-1
J(i,j)=4*M(i,j)-[M(i+1,j)+M(i-1,j)+M(i,j+1)+M(i,j-1)];
end
end
subplot(1,2,1);imshow(I);title('original');
subplot(1,2,2);imshow(uint8(J));title('laplace');
```



（2）使用霍夫变换法找到图像中的直线：

```
image=imread('a.jpg ');
image=rgb2gray(image);
image=double(image);
figure(2);
subplot(1,2,1),imshow(image,[]);
title('Road Image');
thresh=[0.02,0.2];
sigma=2;
BW = edge(image,'canny',thresh,sigma);

%-----------------得到参数空间------------------
[H, theta, rho]= hough(BW,'RhoResolution', 0.5,'ThetaResolution',0.5);

peak=houghpeaks(H,5); %求极值点
lines=houghlines(BW,theta,rho,peak); %返回原图直线信息
subplot(1,2,2),imshow(BW,[]),title('Hough Transform Detect Result'),hold on
```

```
for k=1:length(lines)
    xy=[lines(k).point1;lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',4);
end
```