图像处理作业三实验报告

姓名：果宏帅 班级：软工一班 学号：3017218058

一. **对 LoG 的数学形式进行数学推导。**

LoG即直接对高斯模型求二阶导数

高斯卷积函数定义为：

https://img-blog.csdn.net/20131109135729171

而原始图像f(x,y)与高斯卷积定义为：

https://img-blog.csdn.net/20131109135916953

因为：

https://img-blog.csdn.net/20131109140039875

所以LoG可以通过先对高斯函数进行偏导操作，然后进行卷积求解。公式表示为：

https://img-blog.csdn.net/20131109140354671

和

https://img-blog.csdn.net/20131109140414250

因此，我们可以将LOG核函数定义为：

https://img-blog.csdn.net/20131109141029625

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。

I.

1.最小二乘法直线拟合

代码：function []=leastSquare()

% 数据点集的大小

point\_size = 100;

% 产生高斯分布数据

x=normrnd(0,0.1,1,point\_size);

% 原曲线为y=x

y=x;

% outlier数量

outlier\_size=20;

% 产生outlier

outlier\_x=rand(1,outlier\_size)\*(max(x)-min(x))+min(x);

outlier\_y=rand(1,outlier\_size)\*(max(x)-min(x))+min(x);

% 合并数据

x=[x outlier\_x];

y=[y outlier\_y];

scatter(x,y);hold on;

% 最小二乘法的系数设置

    a = x\*x';

    b = sum(x);

    c = x\*y';

    d = sum(y);

% 求解斜率k

k = (length(x).\*c-b\*d)./(length(x).\*a-b\*b)

% 求解截距t

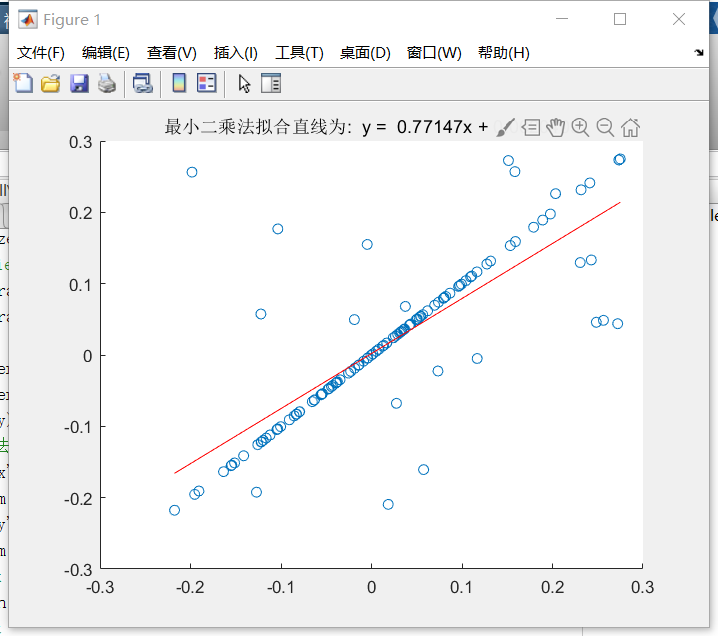
t = (a.\*d-c.\*b)/(a\*length(x)-b.\*b)

y2=x\*k+t;

plot(x,y2,'r');

title(['最小二乘法拟合直线为:  y =  ',num2str(k),'x + ',num2str(t)]);

实验结果：



2.RANSAC法直线拟合

代码：function []=RANSAC()

% 数据点集的大小

point\_size = 100;

% 产生高斯分布数据

x=normrnd(0,0.1,1,point\_size);

% 原曲线为y=x

y=x;

% outlier数量

outlier\_size=20;

% 产生outlier

outlier\_x=rand(1,outlier\_size)\*(max(x)-min(x))+min(x);

outlier\_y=rand(1,outlier\_size)\*(max(x)-min(x))+min(x);

% 合并数据

x=[x outlier\_x];

y=[y outlier\_y];

scatter(x,y);

hold on;

data = [x' y']';

number = point\_size+outlier\_size;

k=0;

b=0;

% 最佳匹配的参数

sigma=1;

% 符合拟合模型的数据的个数

pretotal=0;

%迭代100次

for i=1:100

% 随机选择两个点

    idx = randperm(number,2);

    sample = data(:,idx)

% 拟合直线方程 y=kx+b

    x = sample(1, :)

    y = sample(2, :);

% 直线斜率

    k=(y(1)-y(2))/(x(1)-x(2));

    b = y(1) - k\*x(1);

    line = [k -1 b];

% 求每个数据到拟合直线的距离

    mask=abs(line\*[data; ones(1,size(data,2))]);

% 计算数据距离直线小于一定阈值的数据的个数

    total=sum(mask<sigma);

% 找到符合拟合直线数据最多的拟合直线

    if total>pretotal

        pretotal=total;

        bestline=line;

    end

end

% 最佳拟合的数据

mask=abs(bestline\*[data; ones(1,size(data,2))])<sigma;

k=1;

for i=1:length(mask)

    if mask(i)

        inliers(1,k) = data(1,i);

        k=k+1;

    end

end

% 绘制最佳匹配曲线

k = -bestline(1)/bestline(2);

b = -bestline(3)/bestline(2);

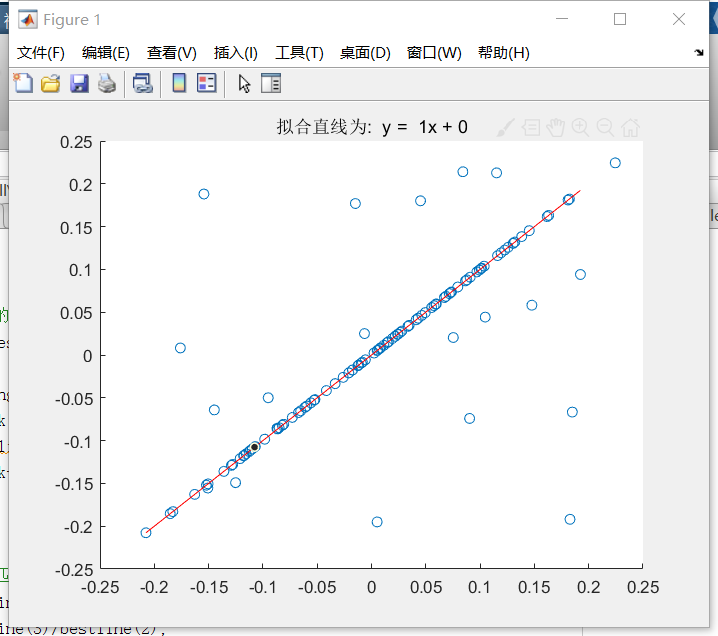
x = min(inliers(1,:)):0.1:max(inliers(1,:));

y = k\*x + b;

plot(x,y,'r');

title(['拟合直线为:  y =  ',num2str(k),'x + ',num2str(b)]);

实验结果：



3.霍夫变换法拟合直线

代码：function []=houghTrans()

% 数据点集的大小

point\_size = 100;

% 产生高斯分布数据

x=normrnd(0,0.1,1,point\_size);

% 原曲线为y=x

y=x;

% outlier数量

outlier\_size=20;

% 产生outlier

outlier\_x=rand(1,outlier\_size)\*(max(x)-min(x))+min(x);

outlier\_y=rand(1,outlier\_size)\*(max(x)-min(x))+min(x);

% 合并数据

x=[x outlier\_x];

y=[y outlier\_y];

scatter(x,y);hold on;

data=[x;y];

number=point\_size+outlier\_size;

% 霍夫空间

n\_max=5;

h=zeros(315,2\*n\_max);

theta\_i=1;

% 拟合阈值

sigma=80;

for theta=0:0.01:3.14

    p=[-sin(theta),cos(theta)];

    d=p\*data;

    for i=1:number

   %由于霍夫空间中d比较大，对d值进行了缩放

    h(theta\_i,round(d(i)/10+n\_max))=h(theta\_i,round(d(i)/10+n\_max))+1;

    end

    theta\_i=theta\_i+1;

end

[tx,p]=find(h>sigma);

% 符合直线条数

lines=size(tx);

% 将还原回距离R

r=(p-n\_max)\*10;

% 将theta还原

tx=0.01\*tx;

x=min(data(:)):0.05:max(data(:));

% 画出拟合曲线

    for i=1:40:lines

        y = tan(tx(i))\*x+r(i)/cos(tx(i));

        plot(x,y,'r');

    end

end

实验结果：

![C:\Users\dell\AppData\Roaming\Tencent\Users\377213836\QQ\WinTemp\RichOle\~FT7M%OI37G}Q_{O]({3Q77.png](data:image/png;base64,)

II.对测试图像test.jpg使用拉普拉斯算子提取边缘点，之后拟合

1.最小二乘法拟合：

代码：function [] = leastSquareImg()

img=imread('test.jpg ');

%  拉普拉斯算子

filter=[0,1,0;1,-4,1;0,1,0];

% 算子大小

fsize=3;

flength = (fsize-1)/2;

% 图像灰度转换

bwImg = double (rgb2gray(img));

[imgH,imgW]=size(bwImg);

p=1;

imshow(bwImg);

% 处理图像，结果保存在gNewImg

for i=1+flength:imgH-flength

    for j=1+flength:imgW-flength

            temp = bwImg(i-flength:i+flength,j-flength:j+flength);

            newImg(i,j)=sum(sum(temp.\*filter));

%             记录边缘点坐标

            if newImg(i,j) ~= 0

                x(p)=i;

                y(p)=j;

                p=p+1;

            end

    end

end

imshow(newImg);

% scatter(x,y);

hold on;

for i=20:20:400

    x1=x(:,i-19:i);

    y1=y(:,i-19:i);

    % 最小二乘法的系数设置

    a = x1\*x1';

    b = sum(x1);

    c = x1\*y1';

    d = sum(y1);

    % 求解斜率k

    k = (length(x1).\*c-b\*d)./(length(x1).\*a-b\*b);

    % 求解截距t

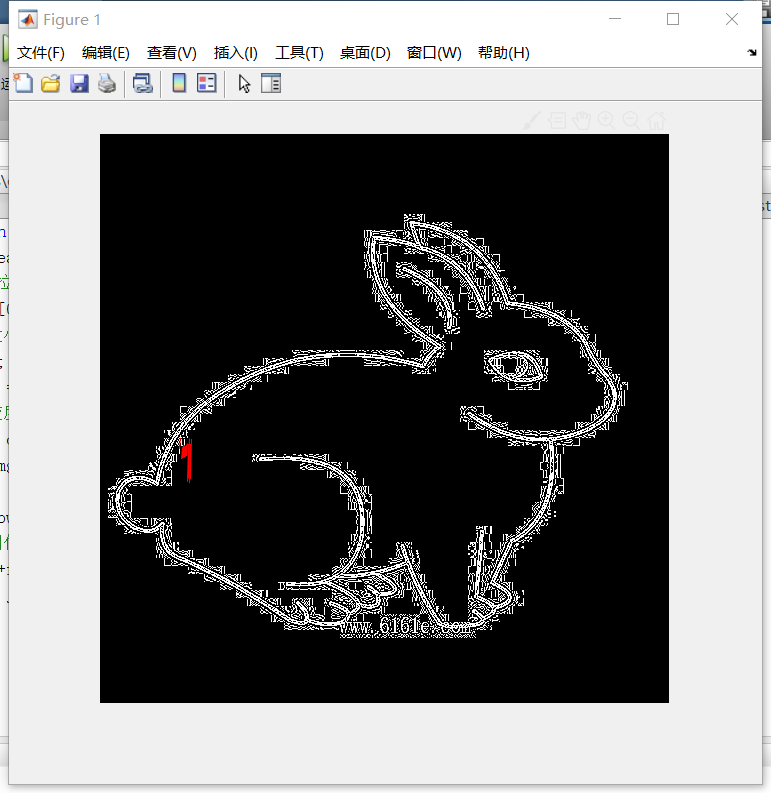
    t = (a.\*d-c.\*b)/(a\*length(x1)-b.\*b);

    y1=x1\*k+t;

    plot(x1,y1,'r');

end

结果：



2.RANSC拟合

代码: function []=RANSACImg()

img=imread('test.jpg');

%  拉普拉斯算子

filter=[0,1,0;1,-4,1;0,1,0];

% 算子大小

fsize=3;

flength = (fsize-1)/2;

% 图像灰度转换

bwImg = double (rgb2gray(img));

[imgH,imgW]=size(bwImg);

p=1;

imshow(bwImg);

% 处理图像，结果保存在gNewImg

for i=1+flength:imgH-flength

    for j=1+flength:imgW-flength

            temp = bwImg(i-flength:i+flength,j-flength:j+flength);

            newImg(i,j)=sum(sum(temp.\*filter));

%             记录边缘点坐标

            if newImg(i,j) ~= 0

                x(p)=i;

                y(p)=j;

                p=p+1;

            end

    end

end

imshow(newImg);

% 对于边缘点拟合曲线

data = [x' y']';

% 显示数据点

% figure;

% scatter(data(1,:),data(2,:));

hold on;

number = size(data,2);

k=0;

b=0;

% 最佳匹配的参数

sigma=1;

for i=1:100

% 随机选择两个点

    idx = randperm(number,2);

    sample = data(:,idx)

% 拟合直线方程 y=kx+b

    x = sample(1, :)

    y = sample(2, :);

% 直线斜率

    k=(y(1)-y(2))/(x(1)-x(2));

    b = y(1) - k\*x(1);

    line = [k -1 b];

% 求每个数据到拟合直线的距离

    mask=abs(line\*[data; ones(1,size(data,2))]);

% 计算数据距离直线小于一定阈值的数据的个数

    total=sum(mask<sigma);

% 找到符合拟合直线数据最多的拟合直线

    if total>25

        pretotal=total;

        bestline=line;

        % 最佳拟合的数据

        mask=abs(bestline\*[data; ones(1,size(data,2))])<sigma;

        k=1;

        for i=1:length(mask)

            if mask(i)

                inliers(1,k) = data(1,i);

                k=k+1;

            end

        end

        % 绘制最佳匹配曲线

        k = -bestline(1)/bestline(2);

        b = -bestline(3)/bestline(2);

        x = min(inliers(1,:)):0.1:max(inliers(1,:));

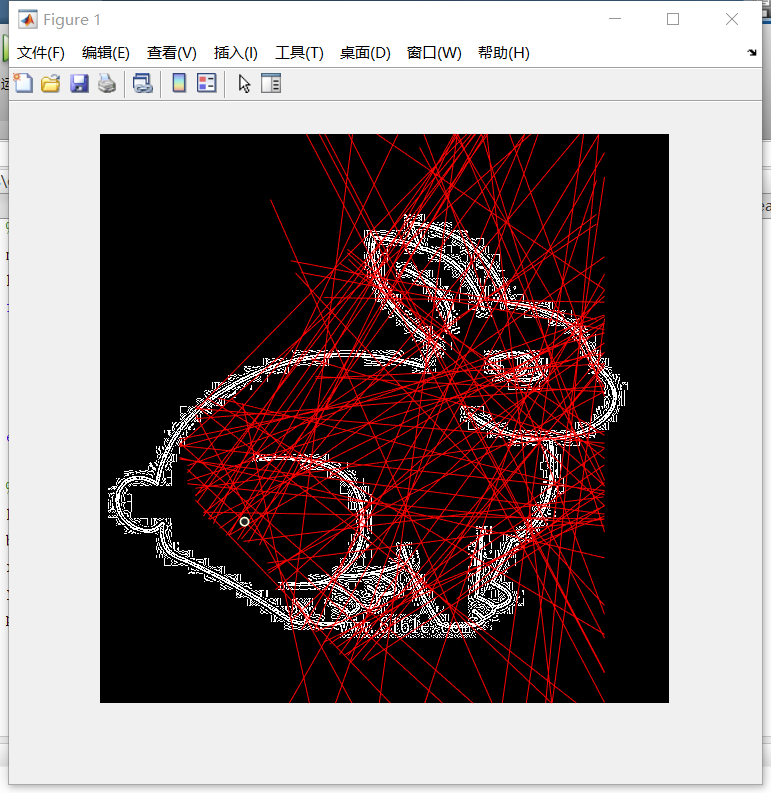
        y = k\*x + b;

        plot(x,y,'r');

    end

end

结果：



3.霍夫变换拟合

代码：function []=houImg()

img=imread('test.jpg');

%  拉普拉斯算子

filter=[0,1,0;1,-4,1;0,1,0];

% 算子大小

fsize=3;

flength = (fsize-1)/2;

% 图像灰度转换

bwImg = double (rgb2gray(img));

[imgH,imgW]=size(bwImg);

id=1;

imshow(bwImg);

% 处理图像，结果保存在gNewImg

for i=1+flength:imgH-flength

    for j=1+flength:imgW-flength

            temp = bwImg(i-flength:i+flength,j-flength:j+flength);

            newImg(i,j)=sum(sum(temp.\*filter));

%             记录边缘点坐标

            if newImg(i,j) ~= 0

                x(id)=i;

                y(id)=j;

                id=id+1;

            end

    end

end

imshow(newImg);

% figure;

% scatter(x,y);

hold on;

data=[x;y];

number=size(data,2);

% 霍夫空间

n\_max=65;

h=zeros(315,2\*n\_max);

ti=1;

% 拟合阈值

ma=80;

for theta=0:0.01:3.14

    p=[-sin(theta),cos(theta)];

    d=p\*data;

    for i=1:number

   %由于霍夫空间中d比较大，对d值进行了缩放

    h(ti,round(d(i)/10+n\_max))=h(ti,round(d(i)/10+n\_max))+1;

    end

    ti=ti+1;

end

[tx,p]=find(h>ma);

% 符合直线条数

lines=size(tx);

% 将还原回距离R

r=(p-n\_max)\*10;

% 将theta还原

tx=0.01\*tx;

x=min(data(:)):20:max(data(:));

% 画出拟合曲线

    for i=1:80:lines

        y = tan(tx(i))\*x+r(i)/cos(tx(i));

        if max(y)<160 && min(y)>0

          plot(x,y,'r');

        end

    end

end

结果：

