# 图像第三次作业

**姓名：万焱 年级：2017 班级：2**

基本理论：

高斯卷积函数定义： $G_\sigma(x,y)=\frac{1}{\sqrt{2\pi\sigma^2}}exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$

原始图像和高斯卷积的定义： $\triangle[G_\sigma(x,y)*f(x,y)]=[\triangle G_\sigma(x,y)]*f(x,y)=LoG*f(x,y)$

由于 $\frac{d}{dt}[h(t)*f(t)]=\frac{d}{dt}\int f(\tau)h(t-\tau)d\tau=\int f(\tau)\frac{d}{dt}h(t-\tau)d\tau=f(t)*\frac{d}{dt}h(t)$

所以LOG $\triangle G_\sigma(x,y)$ 通过对高斯函数求偏导数然后再进行卷积求解。

公式为： $\frac{\partial}{\partial x}G_\sigma(x,y)=\frac{\partial}{\partial x}e^{-(x^2+y^2)/2\sigma^2}=-\frac{x}{\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$

和 $\frac{\partial^2}{\partial^2 x}G_\sigma(x,y)=\frac{x^2}{\sigma^4}e^{-(x^2+y^2)/2\sigma^2}-\frac{1}{\sigma^2}e^{-(x^2+y^2)/2\sigma^2}=\frac{x^2-\sigma^2}{\sigma^4}e^{-(x^2+y^2)/2\sigma^2}$

因此LOG函数定义： $LoG\triangleq\triangle G_\sigma(x,y)=\frac{\partial^2}{\partial x^2}G_\sigma(x,y)+\frac{\partial^2}{\partial y^2}G_\sigma(x,y)=\frac{x^2+y^2-2\sigma^2}{\sigma^4}e^{-(x^2+y^2)/2\sigma^2}$

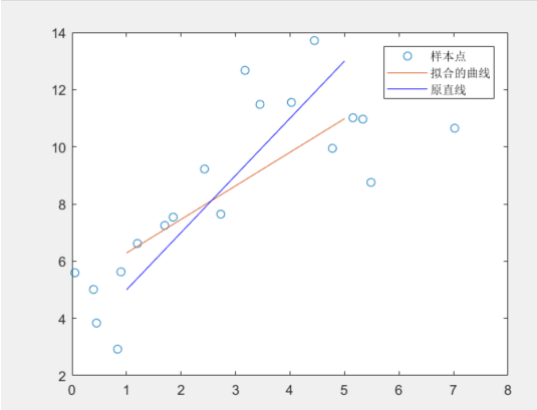
## 拟合直线 最小二乘法

代码：

```
function [] = user_define()

% 设将要拟合的直线为 y = 2x+3，该直线为理想情况
% 现对直线上 x>0 随机取点20个，分别对应x[20] y[20]
clear
clc
%y = 2*x +3;
a = [1,5];  % 直线上的点（1，5）
b = [5,13];     % 直线上的点（5，13）
% 初始化参数，以提升计算效率
% 否则会提示警告：“变量似乎会随着迭代次数改变而变化，请预分配内存，以提高运行速度”
% t = zero(1,20);
% p = zero(1,20);
t(1) = 0;
p(1) = 0;
% 生成的t(i)和p（i）为直线上的随机点
for i = 1:20
    t(i) = rand(1,1)*5;
    p(i) = 2*t(i)+3;
end

% 对采集到的离散的t(i),p(i)加入wng(1,size(t),1)的高斯白噪声，生成的序列作为含误差的测量序列
t = sort(t);
p = sort(p);
% 采用带绝对值的高斯白噪声
```

```
% t_g = t + abs(wgn(1,20,0));
% p_g = p + abs(wgn(1,20,0));
% 使用（-1，1）的随机数来给测试序列添加噪声
% t_g = t + (rand(1,20)*2-1);
% p_g = p + (rand(1,20)*2-1);
% 高斯白噪声
t_g = t + wgn(1,20,0);
p_g = p + wgn(1,20,0);
% 最小二乘法对 t_g 和 p_g 拟合
% 这里才开始最小二乘法
% 首先使用Matlab内置的最小二乘函数 polyfit 进行多项式拟合，具体自己查help
% 注：type函数可帮助查看源代码
line1 = polyfit(t_g,p_g,1);
t_g2 = 1:0.1:5;
p_g2 = polyval(line1,t_g2);
plot(t_g,p_g,'o',t_g2,p_g2);
legend('拟合的曲线');
hold on
ideal_line = 2*t_g2+3;
plot(t_g2,ideal_line,'b');
legend('样本点','拟合的曲线','原直线');
```

拟合效果：



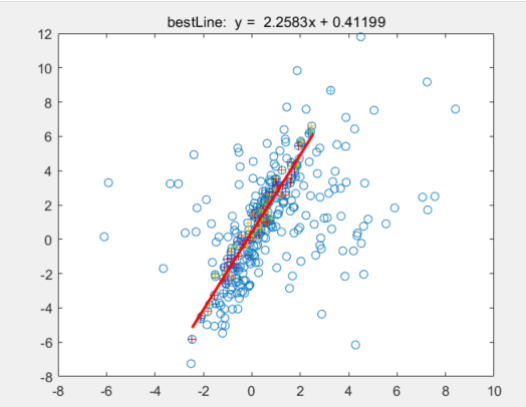

## RANSAC拟合直线

代码：

```
function[] = RANSAC()
clc;clear all;close all;

%%%二维直线拟合
%%%生成随机数据
%内点
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=kx+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)]);
```

拟合效果:



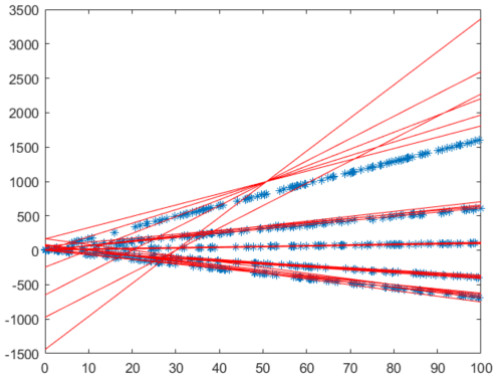

## 霍夫曼拟合直线

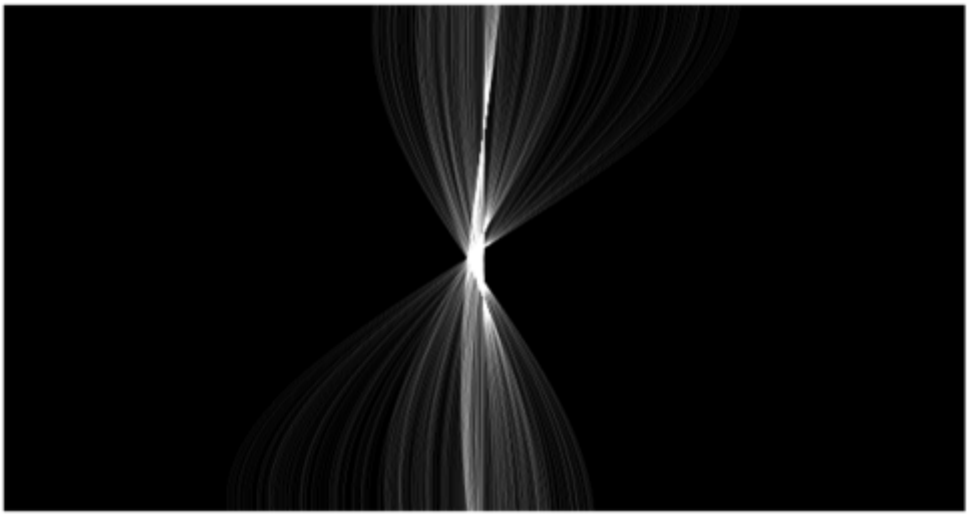
代码：

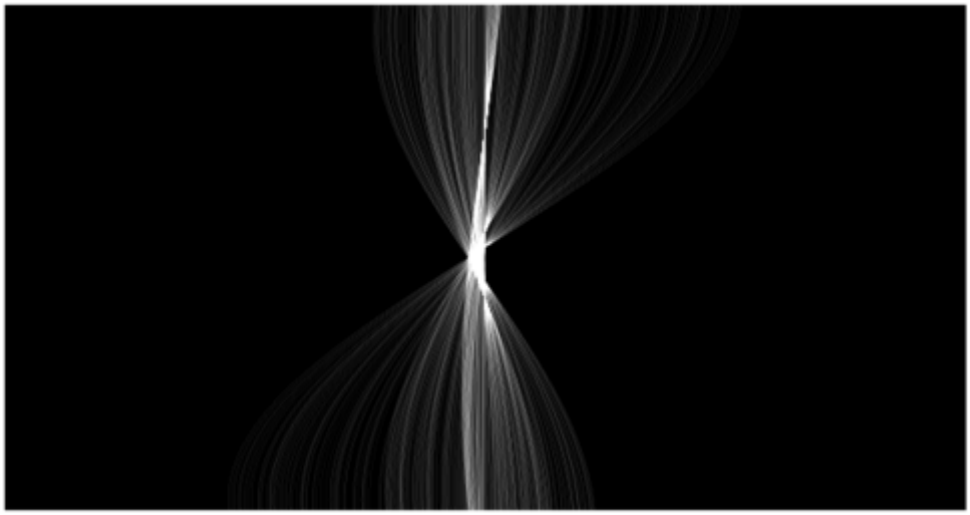
```
function [] = houghman()
clc
clear
%生成五条标准直线用于直线检测%
% x1=sort(100.*rand(1,100));
% y1=-4*x1+2;
% data1=[x1;y1];
% x2=sort(10+100.*rand(1,100));
% y2=x2+4;
% data2=[x2;y2];
% x3=sort(100.*rand(1,100));
% y3=14*x3+6;
% data3=[x3;y3];
% x4=sort(100.*rand(1,100));
% y4=-7*x4+8;
% data4=[x4;y4];
% x5=sort(100.*rand(1,100));
% y5=6*x5+10;
% data5=[x5;y5];
% data=[data1,data2,data3,data4,data5];%构建点集

%生成五条带噪声直线用于直线拟合%
%两个数据集二选一使用
x1=sort(100.*rand(1,100));
y1=x1+2+2.*rand(1,100);
data1=[x1;y1];
x2=sort(100.*rand(1,100));
y2=-4*x2+4+4.*randn(1,100);
data2=[x2;y2];
x3=sort(100.*rand(1,100));
y3=16*x3+6+6.*rand(1,100);
data3=[x3;y3];
x4=sort(100.*rand(1,100));
y4=-7*x4+8+8.*rand(1,100);
data4=[x4;y4];
x5=sort(100.*rand(1,100));
y5=6*x5+10+10.*randn(1,100);
data5=[x5;y5];
data=[data1,data2,data3,data4,data5];%构建点集
[m,n]=size(data);%统计点数
%构建霍夫空间
```

```
n_max=300;%霍夫空间的纵轴最大值
h=zeros(315,2*n_max);
theta_i=1;
sigma=70;%设置拟合阈值
i=0;
%直线公式推导
%y=sin(theta)/cos(theta)*x+b
%->p=b*cos(theta)=-sin(theta)*x+cos(theta)*y
for theta=0:0.01:3.14
    p=[-sin(theta),cos(theta)];
    d=p*data;
    for i=1:n
   %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
[theta_x,p]=find(h>sigma);%查找投票数大于sigma的位置
l_number=size(theta_x);%符合直线条数
r=(p-n_max)*10;%将还原回距离R
theta_x=0.01*theta_x;%将theta还原
figure('color','w');
plot(data(1,:),data(2,:),'*');
hold on
x_line=0:20:100;

for i=1:l_number
    if(abs(cos(theta_x(i)))<0.01)%斜率不存在的情况
        x=r(i);y=1:100;
        plot(x,y,'r');
    else
        y=tan(theta_x(i))*x_line+r(i)/cos(theta_x(i));%画出拟合曲线
        plot(x_line,y,'r');
    end
end
hold off
figure('color','w');
imshow(uint8(10*h));%展示霍夫空间结果
```

拟合效果:





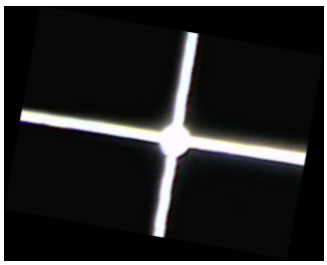
## 图像监测直线霍夫曼

代码:

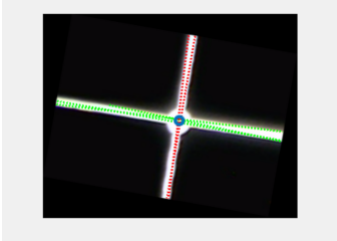
```
function[] = MATLAB_code()
clear;clc;close all
I1=imread('1.bmp');
I=rgb2gray(I1);
index=find(I<50);
I(index)=0;
BW=im2bw(I);
[H,T,R]=hough(BW);
imshow(H,[],'XData',T,'YData',R,'InitialMagnification','fit');
xlabel('\theta'), ylabel('\rho');
```

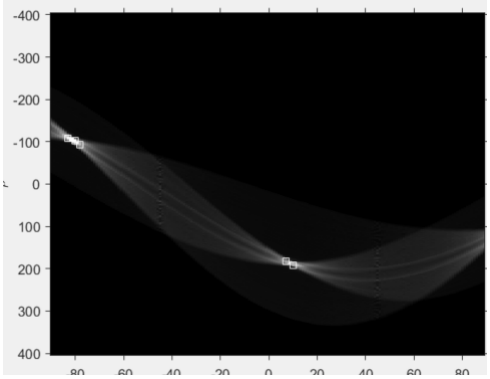
```
axis on
axis normal
hold on
P=houghpeaks(H,5);
x=T(P(:,2));
y=R(P(:,1));
plot(x,y,'s','color','white');
lines=houghlines(BW,T,R,P);
figure
imshow(I1)
hold on
k=[];
b=[];
for n=1:length(lines)
   xy=[lines(n).point1; lines(n).point2];
   k(n)=(xy(1,2)-xy(2,2))/(xy(1,1)-xy(2,1)+1e-8);
   b(n)=xy(1,2)-k(n)*xy(1,1);
end
k1=[];
k2=[];
b1=[];
b2=[];
alpha=rad2deg(atan(k));
distance=abs(alpha-alpha(1));
index=find(distance>120);
distance(index)=180-distance(index);

index1=find(distance<45);
index2=find(distance>=45);

k1=k(index1);
k2=k(index2);
b1=b(index1);
b2=b(index2);
for n=index1
   xy=[lines(n).point1; lines(n).point2];
   plot(xy(:,1),xy(:,2),':g','LineWidth',1.5);
end

for n=index2
   xy=[lines(n).point1; lines(n).point2];
   plot(xy(:,1),xy(:,2),':r','LineWidth',1.5);
end
x0=[];
y0=[];
count=1;
for i=1:length(k1)
    for j=1:length(k2)
        x0(count)=(b1(i)-b2(j))/(k2(j)-k1(i));
        y0(count)=k1(i)*x0(count)+b1(i);
        count=count+1;
    end
end
plot(mean(x0),mean(y0),'o','LineWidth',2.5)
```

监测原图像：



效果图



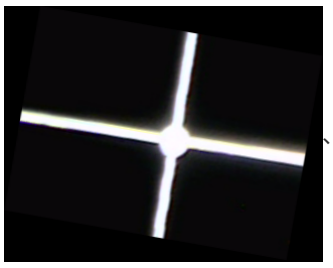



## 直线监测-RANSAC

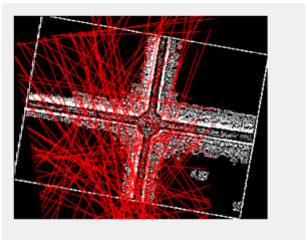
```
img=imread('1.bmp');
%  拉普拉斯算子
filter=[0,1,0;1,-4,1;0,1,0];
% 算子大小
fsize=3;
flength = (fsize-1)/2;
% 图像灰度转换
bwImg = double (rgb2gray(img));
[imgH,imgW]=size(bwImg);
p=1;
imshow(bwImg);
% 处理图像，结果保存在gNewImg
for i=1+flength:imgH-flength
    for j=1+flength:imgW-flength
            temp = bwImg(i-flength:i+flength,j-flength:j+flength);
            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
```

原图:



效果图：



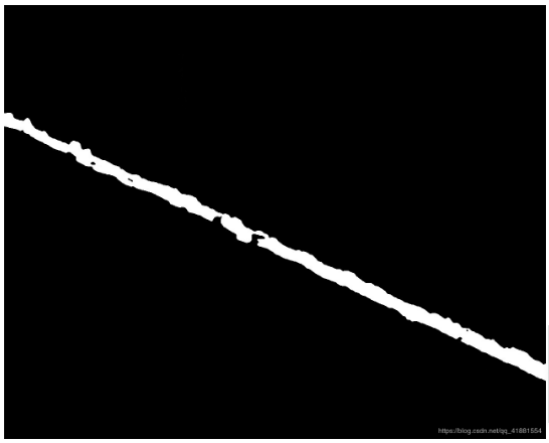
## 直线监测最小二分法

代码

```
im=imread('1.jpg');    %读取图片

%im=rgb2gray(im);    %如果是rgb图片则转为灰度图

[h,w]=size(im);          %获取图片高(h)、宽(w)

%扫描每一个像素,并记录白点（值为1）坐标及个数
n=0;
for y0=1:1:h
    for x0=1:1:w
        if(im(y0,x0)==255)
            n=n+1;
            y(n)=y0;
            x(n)=x0;
        end
    end
end

%最小二乘法拟合直线
A = 0.0;
B = 0.0;
C = 0.0;
D = 0.0;
for i=1:1:n
    A=A+x(i)*x(i);
    B=B+x(i);
    C=C+x(i)*y(i);
    D=D+y(i);
end

a = (C*n - B*D) / (A*n - B*B);
b = (A*D - C*B) / (A*n - B*B);

%灰度图转rgb彩色图片
imrgb=repmat(im,[1,1,3]);

%画线，线宽3，红色
for i=1:1:w
    y=a*i+b;
    y2=int32(y);          %把y转换成整数
    %把线上的通道1（红色）置为255
    imrgb(y2-1,i,1)=255;
    imrgb(y2,i,1)=255;
    imrgb(y2+1,i,1)=255;
```

```
    %把线上的其他通道（绿色和蓝色）置为0
    imrgb(y2-1,i,2)=0;
    imrgb(y2,i,2)=0;
    imrgb(y2+1,i,2)=0;
    imrgb(y2-1,i,3)=0;
    imrgb(y2,i,3)=0;
    imrgb(y2+1,i,3)=0;
end

%显示图片
imshow(imrgb);
```

原图：



效果图：