数字图像处理

第七次作业

姓名：张勤东

班级：自动化 63

学号：2160504080

日期：2019 年 5 月 1 日

摘要：本报告主要解决了数字图像处理中关于图像边缘检测和图像中直线检测的一些问题。首先，分别使用 Sobel 算子、Canny 算子以及 Roberts 算子对测试图像进行边缘检测，并分析了各种算子的优缺点。接着，在使用三种算子进行边缘检测的基础上，使用 hough 变换检测图中的直线，对比了不同的边缘检测算法对直线检测的影响。调整 hough 变换的参数，得到不同的直线检测结果，并对不同的结果进行分析。

作业内容：直线检测

具体要求：

1. 首先对测试图像（文件名为：test1~test6）进行边缘检测，可采用书上介绍的 Sobel 等模板或者 canny 算子方法；

2. 在边缘检测的基础上，用 hough 变换检测图中直线；

3. 比较不同边缘检测算法（2 种以上）、不同 hough 变换参数对直线检测的影响；

4. 可以采用 Matlab、OpenCV 等自带函数。

解决思路：

1.边缘检测原理

在一幅灰度图像中，图像的边缘是不同内容的边界，区分了不同的内容。图像的边缘表现为图像中灰度值突变的部分。通过对灰度值突变的像素点进行检测，并根据检测出的边缘点的集合，将图片转化为一幅二值图像，从而完成图像的边缘检测。

进行边缘检测的一般步骤为：首先，对图像进行平滑处理，降低图像的噪声。然后，基于图像灰度变化的一阶或二阶导数值的大小对图像中的边缘点进行检测。最后，根据阈值大小，从候选边缘点中选出符合要求的点组成边缘，并得到一幅由边缘组成的二值图像。

(1). Sobel 算子

Sobel 算子是基于图像梯度信息进行边缘检测的算子，它能较好地抑制噪声。根据以下公式可以分别得到 x 方向和 y 方向的偏导数的值。

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

根据 x 方向和 y 方向的偏导数的值，用绝对值来近似梯度的大小。公式如下：

$$M(x,y) \approx |g_x| + |g_y|$$

依次计算出图像中每个像素位置的梯度值，根据设置的阈值大小，将梯度值符合要求的像素点作为边缘像素点，将该位置对应的像素赋为 1，其他不符合要求的像素赋为 0，从而得到一幅由边缘组成的二值图像。

(2). Canny 算子

Canny 算子基于图像梯度信息进行边缘检测的算子，该算子具有更加优秀的

边缘检测效果。

令$f(x,y)$表示输入图像，$G(x,y)$表示高斯函数：

$$G(x,y) = \mathrm{e}^{-\frac{(x^2+y^2)}{2\delta^2}}$$

将$f(x,y)$和$G(x,y)$相卷积，得到平滑后的图像$f_s(x,y)$，公式如下：

$$f_s(x,y) = G(x,y) * f(x,y)$$

计算图像的梯度幅值和梯度方向：

$$\mathrm{M(x,y)} = \sqrt{\mathrm{g_x}^2 + \mathrm{g_y}^2}$$

$$\alpha(\mathrm{x,y}) = \arctan[\frac{\mathrm{g_y}}{\mathrm{g_x}}]$$

令$\mathrm{d_1}$，$\mathrm{d_2}$，$\mathrm{d_3}$和$\mathrm{d_4}$表示四个基本边缘方向：水平、垂直、正 45 度、负 45 度。寻找与$\alpha(\mathrm{x,y})$最接近的方向d_k。若$\mathrm{M(x,y)}$小于沿d_k的两个邻居之一，则令$\mathrm{g}_N(x,y) = 0$；否则，令$\mathrm{g}_N(x,y) = \mathrm{M(x,y)}$，完成非最大抑制。

接着，为了减少伪边缘点并保留有效边缘点，采用双阈值的处理方式。设定一个高阈值和一个低阈值。将高于高阈值的点直接作为边缘点，接着遍历所有高于高阈值的点，将与它们满足 8 连通关系并且高于低阈值的点也作为边缘点。最终，得到一幅由边缘组成的二值图像。

(3). Roberts 算子

Roberts 算子是基于图像梯度信息进行边缘检测的算子，该算子对对角线方向的边缘更敏感。根据以下公式可以分别得到 x 方向和 y 方向的偏导数的值。

$$\mathrm{g}_x = \frac{\partial f}{\partial x} = z_4 - z_1$$

$$\mathrm{g}_y = \frac{\partial f}{\partial y} = z_3 - z_2$$

根据 x 方向和 y 方向的偏导数的值，用绝对值来近似梯度的大小。公式如下：

$$\mathrm{M(x,y)} \approx |\mathrm{g}_x| + |\mathrm{g}_y|$$

依次计算出图像中每个像素位置的梯度值，根据设置的阈值大小，将梯度值符合要求的像素点作为边缘像素点，将该位置对应的像素赋为 1，其他不符合要求的像素赋为 0，从而得到一幅由边缘组成的二值图像。

2.hough 变换进行直线检测原理

当得到一幅由边缘组成的二值图像后，通过 hough 变换可以计算出这些边缘点构成的直线，从而达到对图像进行直线检测的目的。

经过一个点(x,y)的所有直线，可以用以下直线的法线方程表示：

$$\mathrm{x} \cdot \cos\theta + y \cdot \sin\theta = \rho$$

指定$\rho\theta$平面的细分，将$\rho\theta$参数空间划分为累加单元。将二值图像中边缘点(x_k, y_k)作为参数，遍历θ轴上的每个允许的细分值，根据以下方程计算出对应的ρ值。

$$\rho = x_k \cdot \cos\theta + y_k \cdot \sin\theta$$

将ρ值进行四舍五入到ρ轴上允许的细分值。以边缘点(x_k, y_k)作为参数得到一条$\rho\theta$空间曲线，将该曲线经过的累加单元中的值加一。根据累加单元中的值，可以选出相应的直线。

接着，检验在同一条直线上的点的连续性。若属于同一条直线的像素点间的距离比阈值小，则将它们用直线连接起来。最终，完成直线检测的任务。

3.matlab 实现思路

在 matlab 中，使用 imread()函数读入图像。对于 bmp 格式图片，使用 ind2gray()函数，将索引图像和调色板读为一幅灰度图像。

对各幅测试图像调用 edge()函数，将参数分别设置为' Sobel'、' Canny'和' Roberts'，对测试图像进行边缘检测。用 imshow()函数将进行边缘检测后的二值图像显示出来。

为了对比不同边缘检测算法对直线检测的影响，在已使用三种边缘检测算法进行边缘检测的图像上分别进行 hough 变换。调用 hough()函数，得到$\rho\theta$参数空间的累加单元矩阵，其中细分值取默认值 1。调用 houghpeaks()函数对累加单元的数量进行统计，取出累加单元数最多的 10 个点，并且设定阈值为累加单元数量最大值的 50%，若累加单元数量小于该值，则舍去该点。根据各累加单元在$\rho\theta$参数空间的位置，调用 houghlines()函数，得到直线方程以及直线的起始点和终止点，并且设置阈值参数对同一条直线上的像素进行连续性检验。其中，'FillGap'参数设置为 15，表示当两个像素点之间的距离小于 15 时，则将它们用直线连接起来；设置'MinLength'参数为 30，表示若一段直线的总长度小于 30，则将其舍去。接着，根据得到的直线方程以及起始点和终止点的坐标，在原始图像上将检测出的直线标记出来，并将其显示出来。在此基础上，可以对比不同边缘检测算法对直线检测的影响。

为了对比不同的 hough 变换参数对直线检测的影响，在采用 Sobel 算子进行边缘检测得到的二值图像上进行不同的参数的 hough 变换。第一组 hough 变换作为参考，参数值取之前使用的参数值。第二组改变细分值的大小，细分值取 0.5。第三组改变累加单元数量的阈值，将该阈值提高到累加单元数量最大值的 80%，若累加单元数量小于该阈值，则舍去该点。第四组改变判断连续性的阈值，'FillGap'参数改为 30，表示当两个像素点之间的距离小于 30 时，则将它们用直线连接起来；设置'MinLength'参数改为 60，表示若一段直线的总长度小于 60，则将其舍去。将四组由不同 hough 变换参数得到的直线检测结果显示出来，可以对比不同的 hough 变换参数对直线检测的影响。

处理结果：

不同算子得到的边缘检测图像

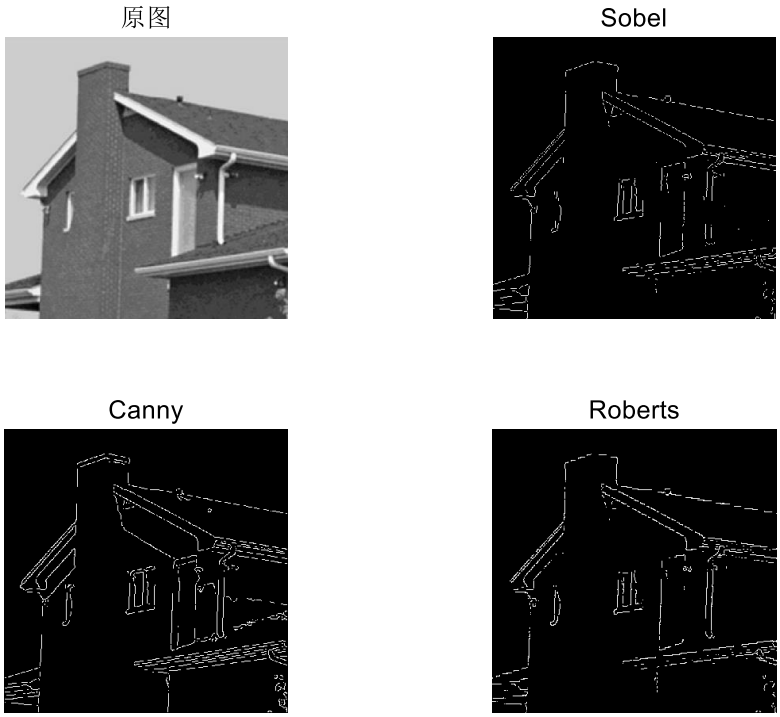


图 Test1

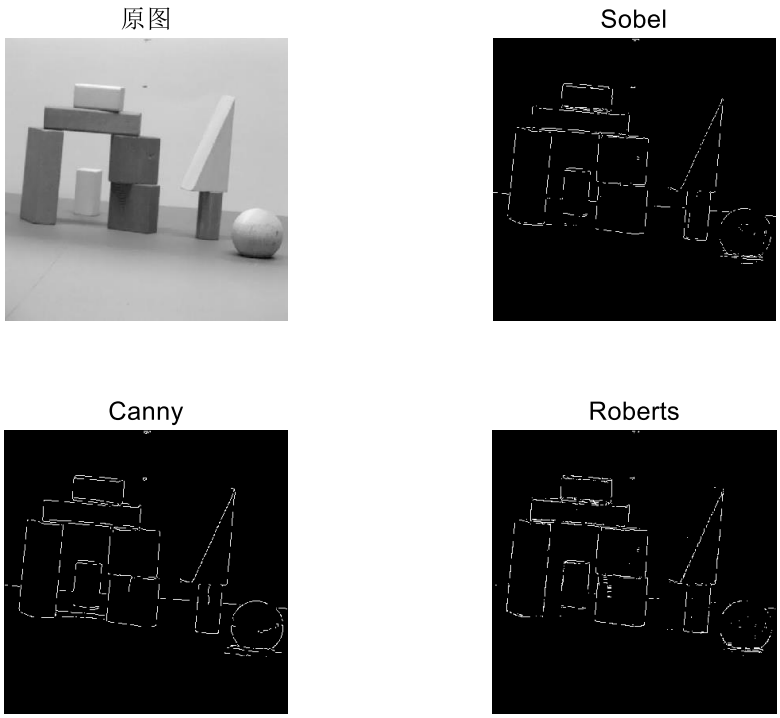
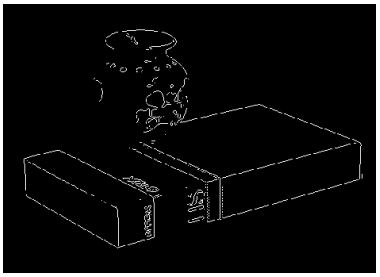


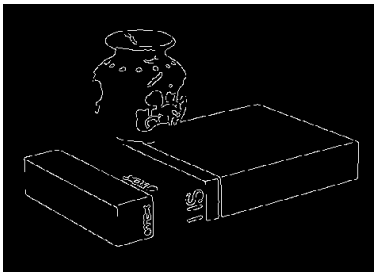
图 Test2

原图

Sobel



Canny



Roberts

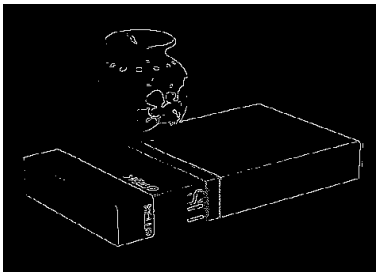
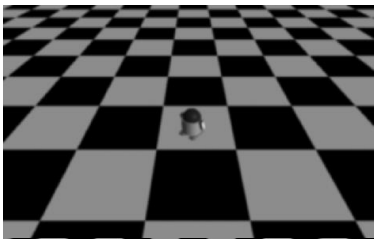
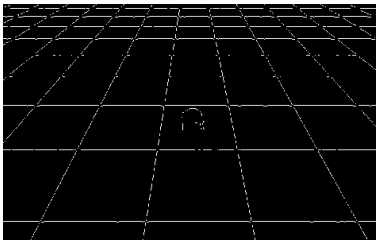


图 Test3

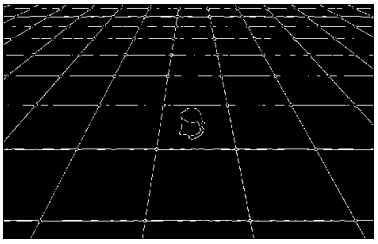
原图



Sobel



Canny



Roberts

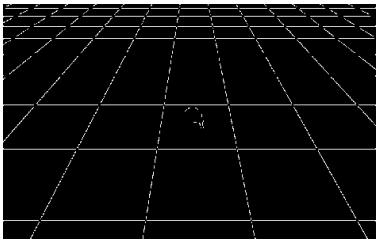


图 Test4

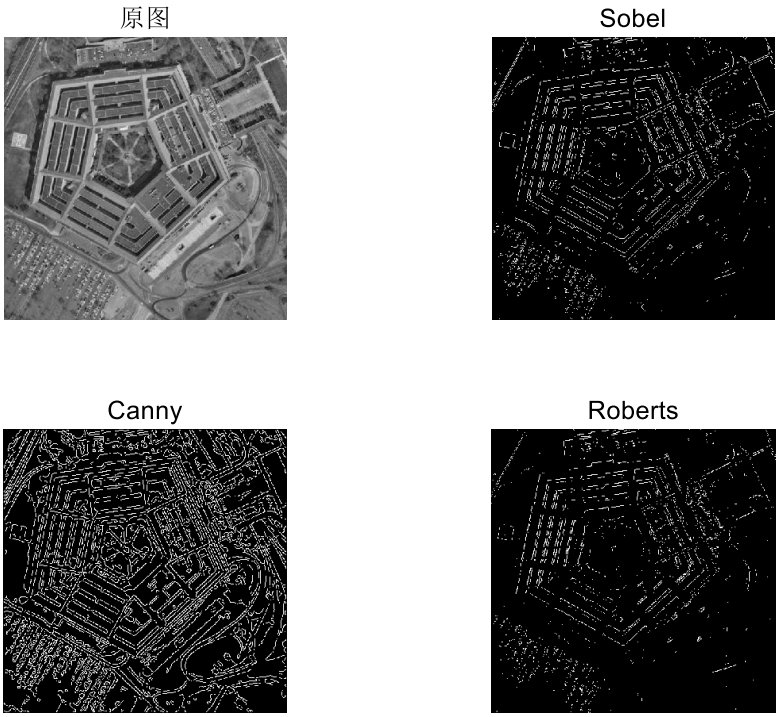


图 Test5

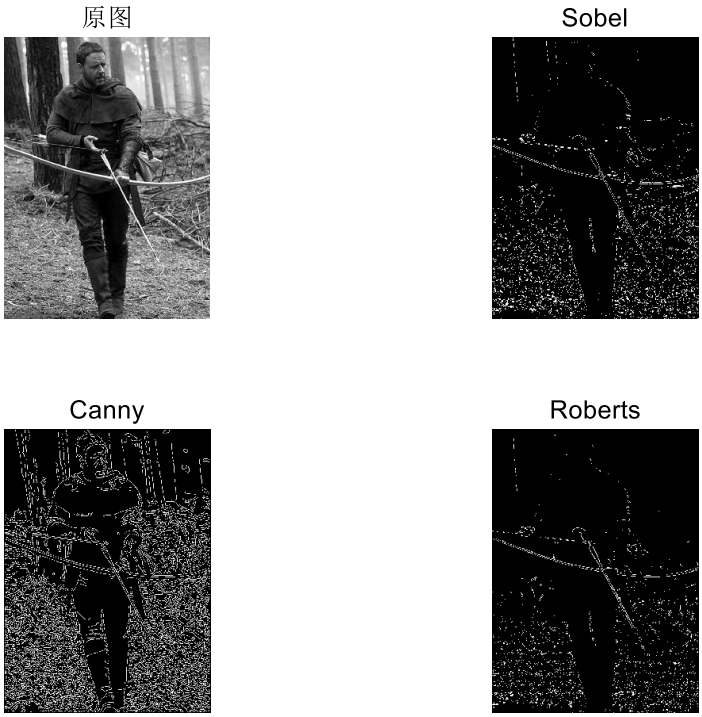


图 Test6

在不同边缘检测算法下得到的直线检测图像

Sobel Canny Roberts

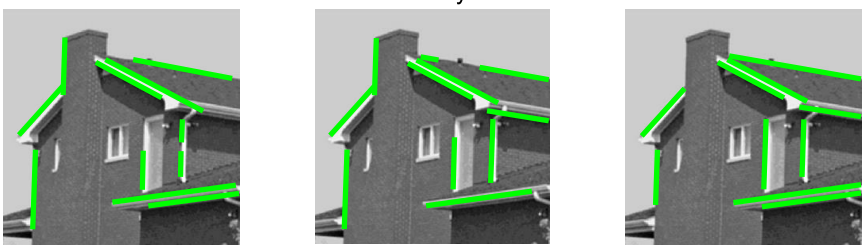


图 Test1

Sobel Canny Roberts

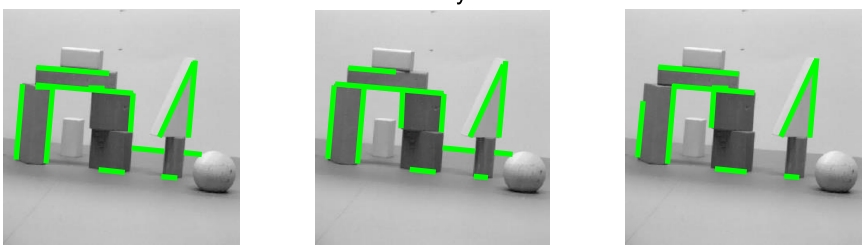


图 Test2

Sobel Canny Roberts

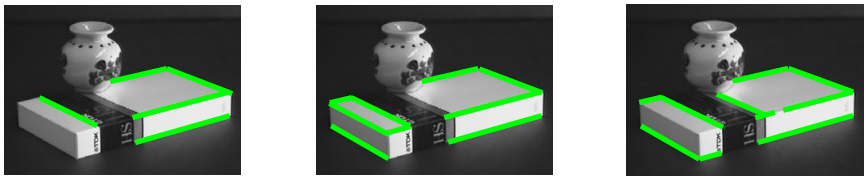


图 Test3

Sobel Canny Roberts

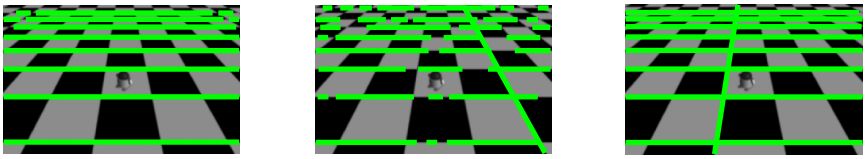


图 Test4

Sobel Canny Roberts

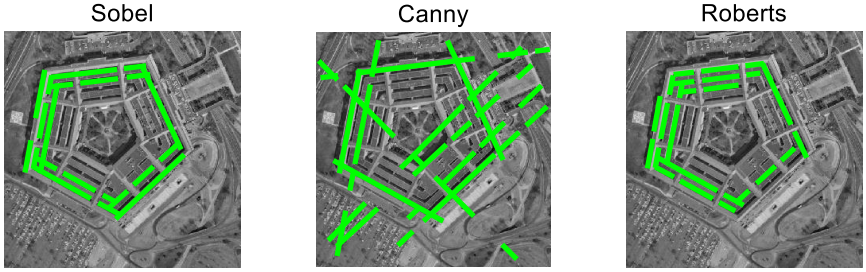


图 Test5

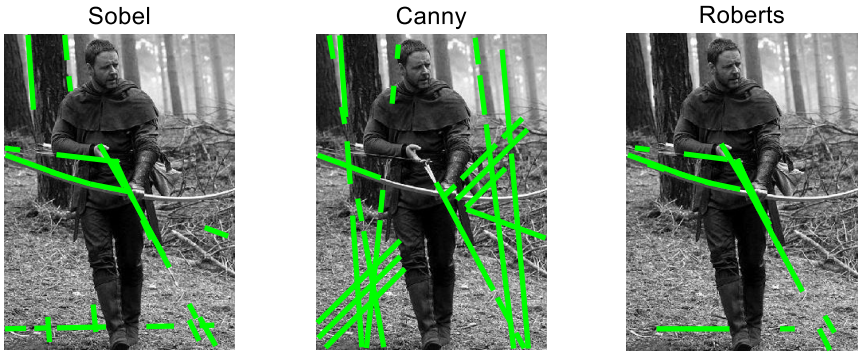


图 Test6

不同参数下直线检测图像

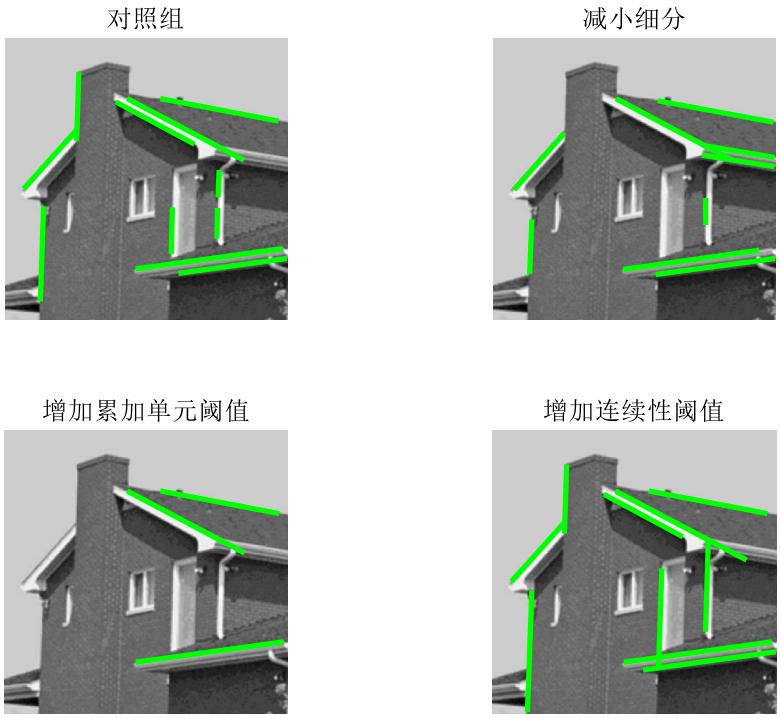
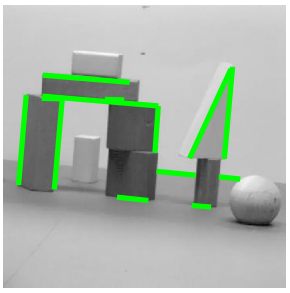
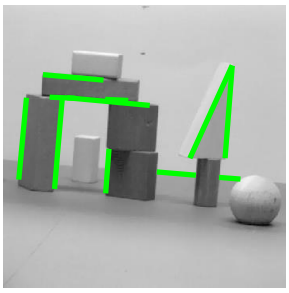


图 Test1

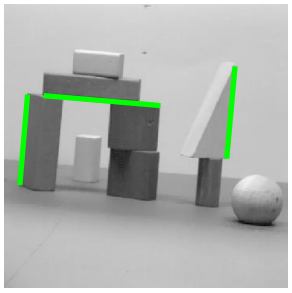
对照组



减小细分



增加累加单元阈值



增加连续性阈值

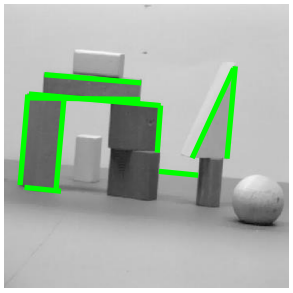
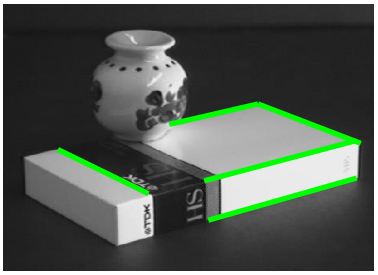
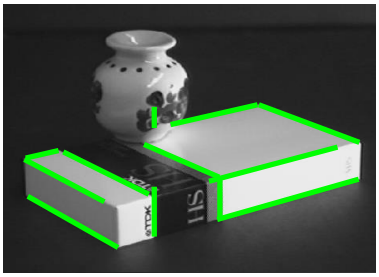


图 Test2

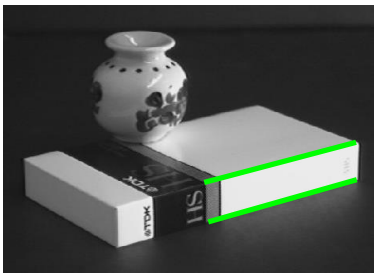
对照组



减小细分



增加累加单元阈值



增加连续性阈值

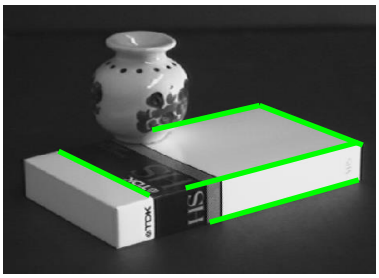
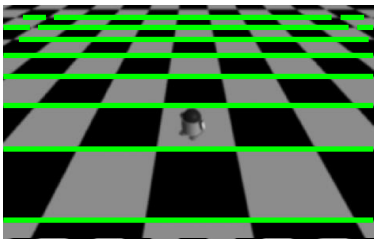
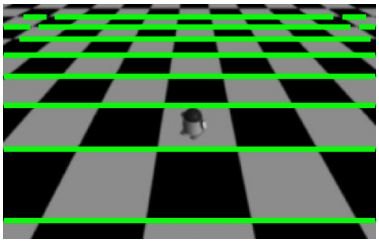


图 Test3

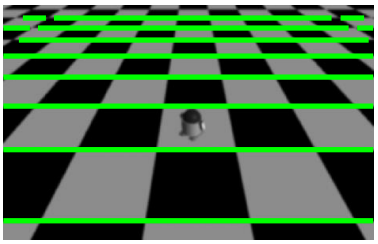
对照组



减小细分



增加累加单元阈值



增加连续性阈值

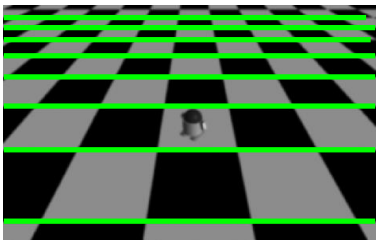
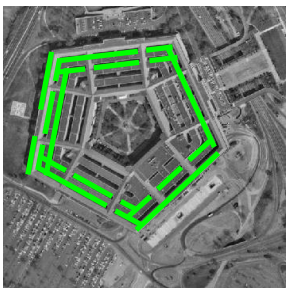
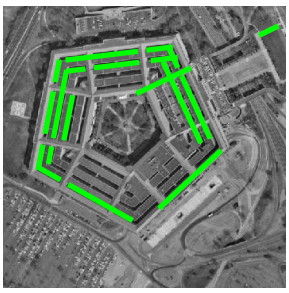


图 Test4

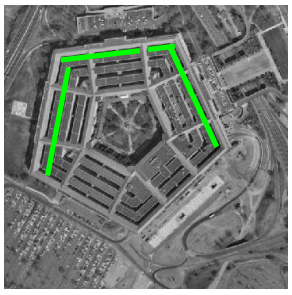
对照组



减小细分



增加累加单元阈值



增加连续性阈值

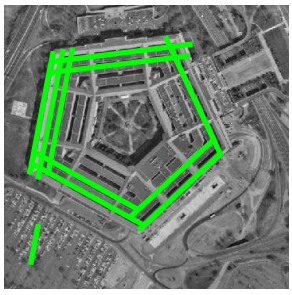


图 Test5

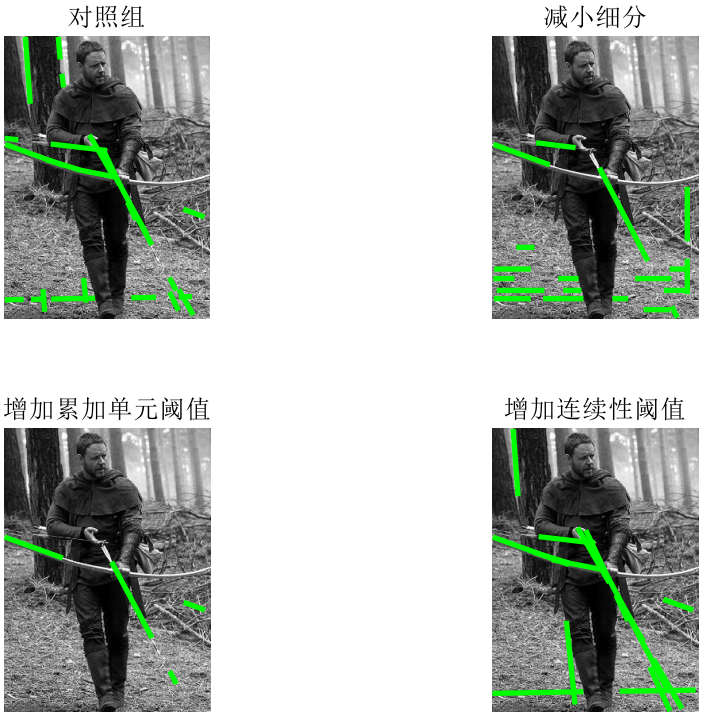


图 Test6

结果分析：

观察边缘检测部分的图像，对比使用 Sobel 算子、Canny 算子以及 Roberts 算子得到的二值图像，可以发现使用 Canny 算子进行边缘检测的效果最好，Sobel 算子的效果其次，Roberts 算子的效果较差。使用 Canny 算子进行边缘检测，对原始测试图像的边缘部分有很好的保留，图像主要的边缘信息都得到了保留。使用 Sobel 算子进行边缘检测，对原始测试图像的边缘部分检测效果较好，并且对水平和垂直的边缘更加敏感。使用 Roberts 算子进行边缘检测，对原始测试图像的边缘部分检测效果一般，但是该算子与 Sobel 算子相比，该算子对正负 45 度的边缘更加敏感。

对比在不同边缘检测算法下得到的直线检测图像，可以发现直线检测的好坏与使用的边缘检测算法有密切的关系。以 Canny 算子为基础得到的直线检测图像，总体上效果更好，但是当所检测图像的边缘信息非常多，以 Canny 算子为基础进行直线检测会检测出一些原始图像上不存在的直线。这主要是因为 Canny 边缘检测算法对边缘检测的效果很好，当所检测图像的边缘信息非常多时，Canny 算法检测出的边缘较多，使用 hough 变换进行直线检测时，将不属于同一边缘但在同一直线上的点都连接了起来，造成检测出一些原始图像上不存在的直线。以 Sobel 算子为基础得到的直线检测图像，综合效果较为不错，不会检测出伪直线，但是对一些边缘不明显的直线的检测效果不如以 Canny 算子为基础进行的直线检测。以 Roberts 算子为基础进行直线检测，效果较差，主要受到 Roberts 边缘检测算法效果较差的影响。

为了对比不同的 hough 变换参数对直线检测的影响，最后一组直线检测图像均是在采用 Sobel 算子进行边缘检测得到的二值图像上进行 hough 变换得到的。观察减小细分后的直线检测图像，可以发现，减小细分后，提高了共线精度，检测出的直线精度更高，减少了将原本不属于同一直线的点画在同一直线上的情况。观察增加累加单元数量阈值的直线检测图像，可以发现，增加阈值后，检测出的直线数量减少，主要是因为增加阈值后更多的点所在直线才能被检测出来，提高了对被检测直线的要求。观察改变判断连续性阈值的直线检测图像，可以发现检测出的直线连续性更好，但有时检测出的直线会比实际原始图像中的边缘直线更长，这主要是因为增加'FillGap'参数和'MinLength'参数后，会将间隔更大的像素点连接起来，并且增加了对每一段直线最短长度的要求。总体比较不同 hough 变换参数得到的直线检测结果，可以发现，不同 hough 变换参数对直线检测的结果有很大影响，合理设置 hough 变换参数，可以得到更好的直线检测效果。

附录

```
pic=cell(6,4);
pic00=imread('test1.tif');
pic{1,1}=pic00(:,:,1);
pic{2,1}=imread('test2.png');
pic{3,1}=imread('test3.jpg');
pic{5,1}=imread('test5.png');
pic{6,1}=imread('test6.jpg');
[pic0,map0]=imread('test4.bmp');
pic{4,1}=ind2gray(pic0,map0);



for i=1:6
    pic{i,2}=double(edge(pic{i,1},'sobel'));
    pic{i,3}=double(edge(pic{i,1},'canny',0.2));
    pic{i,4}=double(edge(pic{i,1},'Roberts'));
end

for i=1:6
    figure(i)
    subplot(2,2,1)
    imshow(pic{i,1})
    title('原图')
    subplot(2,2,2)
    imshow(pic{i,2})
    title('Sobel')
    subplot(2,2,3)
    imshow(pic{i,3})
    title('Canny')
    subplot(2,2,4)
    imshow(pic{i,4})
    title('Roberts')
end

for ii=1:6
    figure
    for iii=2:4

[H,T,R] = hough(pic{ii,iii});
P  = houghpeaks(H,10,'threshold',ceil(0.5*max(H(:))));
```

```
lines = houghlines(pic{ii,iii},T,R,P,'FillGap',15,'MinLength',30);
subplot(1,3,iii-1)
imshow(pic{ii,1}), hold on
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
end

if iii==2
      title('Sobel')
elseif iii==3
      title('Canny')
else
      title('Roberts')
end

      end
end



for kk=1:6

figure
[H,T,R] = hough(pic{kk,2});
P  = houghpeaks(H,10,'threshold',ceil(0.5*max(H(:))));
lines = houghlines(pic{kk,2},T,R,P,'FillGap',15,'MinLength',30);
subplot(2,2,1)
imshow(pic{kk,1}), hold on
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
end
title('对照组')

[H,T,R] = hough(pic{kk,2},'RhoResolution',0.5,'Theta',-90:0.5:89);
P  = houghpeaks(H,10,'threshold',ceil(0.5*max(H(:))));
lines = houghlines(pic{kk,2},T,R,P,'FillGap',15,'MinLength',30);
subplot(2,2,2)
imshow(pic{kk,1}), hold on
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
end
```

```
title('减小细分')

[H,T,R] = hough(pic{kk,2});
P  = houghpeaks(H,10,'threshold',ceil(0.8*max(H(:))));
lines = houghlines(pic{kk,2},T,R,P,'FillGap',15,'MinLength',30);
subplot(2,2,3)
imshow(pic{kk,1}), hold on
for k = 1:length(lines)
   xy = [lines(k).point1; lines(k).point2];
   plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
end
title('增加累加单元阈值')

[H,T,R] = hough(pic{kk,2});
P  = houghpeaks(H,10,'threshold',ceil(0.5*max(H(:))));
lines = houghlines(pic{kk,2},T,R,P,'FillGap',30,'MinLength',60);
subplot(2,2,4)
imshow(pic{kk,1}), hold on
for k = 1:length(lines)
   xy = [lines(k).point1; lines(k).point2];
   plot(xy(:,1),xy(:,2),'LineWidth',2,'Color','green');
end
title('增加连续性阈值')

end
```