数字图像处理

第七次作业

姓名：何佳

班级：自动化 64

学号：2160700002

提交日期：2019.4.26

摘要：本次作业主要内容为：采用 Sobel、canny 等算子进行边缘检测，并对边缘检测后的图像进行 hough 变换。最后对比了边缘检测各类算子的特点，并分析了 hough 变换中关键参数变化对变换结果的影响。以上内容实现方式主要采用 Matlab 自带函数。

一、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；

1.问题分析

1.1 边缘检测

边缘检测是图像处理和计算机视觉，尤其是特征提取中的基本问题。边缘像素是图像中灰度突变的那些像素，而边缘是连接的边缘像素的集合。边缘检测的基本思想是：首先利用边缘增强算子，突出图像中的局部边缘，然后定义像素的“边缘强度”，通过设置阈值的方法提取边缘点集。由于噪声和模糊的存在，检测到的边界可能会变宽或在某点处发生间断。因此，边界检测包括两个基本内容：①用边缘算子提取出反映灰度变化的边缘点集；②在边缘点集合中剔除某些边界点或填补边界间断点，并将这些边缘连接成完整的线。

为了检测局部像素变化，可采用微分来检测，包括一阶微分和二阶微分。一阶微分边缘检测算子又称梯度边缘算子，即图像梯度在边缘处取得极大值，包括 Roberts Cross 算子、Prewitt 算子、Sobel 算子、Kirsch 算子、罗盘算子等。二阶微分边缘检测算子是利用图像的二阶微分在边缘处出现零值这一特性进行边缘检测的，因此该算法又称零点算子和拉普拉斯算子，包括 Canny 算子，Laplacian 算子等。

1.2 Roberts 算子

Roberts 算子采用 2*2 模板大小，它采用对角线方向相邻两像素之差近似梯度幅值检测边缘。其形式最为简单，但对于用关于中心点对称的模板来计算边缘不是很有用。

-1	0
0	1

0	-1
1	0

Roberts 算子

1.3 Prewitt 算子

Prewitt 算子是一种一阶微分边缘检测算子，利用像素点上下、左右邻点的灰度差，在边缘处达到极值检测边缘，去掉部分伪边缘，对噪声具有平滑作用。其原理是在图像空间利用两个方向模板与图像进行邻域卷积来完成的。

Prewitt 算子：

-1	-1	-1
0	0	0
1	1	1

水平

-1	0	1
-1	0	1
-1	0	1

垂直

0	1	1
-1	0	1
-1	-1	0

-1	-1	0
-1	0	1
0	1	1

对角线

1.4 Sobel 算子

Sobel 算子是像素图像边缘检测中最重要的算子之一，在机器学习、数字媒体、计算机

视觉等信息科技领域起着举足轻重的作用。在技术上，它是一个离散的一阶差分算子，在图像的任何一点使用此算子，将会产生该点对应的梯度矢量或是其法矢量。以下是分别检测水平和垂直边缘的 Sobel 算子。

Sobel 算子：

-1	-2	-1
0	0	0
1	2	1

水平

-1	0	1
-2	0	2
-1	0	1

垂直

0	1	2
-1	0	1
-2	-1	0

-2	-1	0
-1	0	1
0	1	2

对角线

1.5 Log 算子

高斯拉普拉斯算子(laplacian of Gaussian, LoG) 将 Laplace 算子与高斯低通滤波相结合，其基本步骤是：首先对图像做高斯滤波，然后再求其拉普拉斯（Laplacian）二阶导数，即图像与 Laplacian of the Gaussian function 进行滤波运算，最后通过检测滤波结果的零交叉（Zero crossings）可以获得图像或物体的边缘。

$$G_\sigma(\mathbf{x},\mathbf{y}) = \frac{1}{2\pi\sigma}\exp(-\frac{x^2+y^2}{2\sigma^2})$$

$$LOG(f)(x,y) = \Delta(G_\sigma * f) = \Delta G_\sigma * f$$

LOG 算子如下：

$$\nabla^2 G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2} = \frac{-2\sigma^2 + x^2 + y^2}{2\pi\sigma^6} e^{-(x^2+y^2)/2\sigma^2}$$

根据 sigma 的不同以及 3*sigma 原则可以建立不同的模板，sigma 越大，图像越模糊滤除噪声效果越好，sigma 越小，效果相反。常用模板如下：

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 1 & 2 & -16 & 2 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

1.6 Canny 算子

Canny 算子的算法虽然较为复杂，但它是迄今为止讨论过的边缘检测器中最为优秀的。Canny 方法基于三个基本目标：①低错误率。所有边缘都应被找到，并且应该没有伪响应。也就是检测到的边缘必须尽可能是真实的边缘。②边缘点应被很好地定位。已定位边缘必须尽可能接近真实边缘，也就是由检测器标记为边缘的点和真实边缘的中心之间的距离应该最小。③单一的边缘点响应。对于真实的边缘点，检测器仅应返回个点，也就是真实边缘周围的局部最大数应该是最小的。这意味着在仅存个单边缘点的位置，检测器不应指出多个边缘

像素。

Canny 边缘检测算法是由下列基本步骤组成的：

①用一个高斯滤波器平滑输入图像。二维高斯函数如下：

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

②计算梯度幅值图像和角度图像。

$$\mathrm{G} = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \mathrm{atan2}(G_y, G_x)$$

③对梯度幅值图像应用非最大抑制。非最大抑制通俗意义上是指寻找像素点局部最大值，其方法是沿着梯度方向，比较它前面和后面的梯度值。

④用双阈值处理和连接分析来检测并连接边缘。双阈值处理的含义是：如果边缘像素点梯度值大于高阈值，则被认为是强边缘点。如果边缘梯度值小于高阈值，大于低阈值，则标记为弱边缘点。对于与强边缘连通的弱边缘进行保留，小于低阈值的点则被抑制掉，从而连接边缘。

2.编程思路

在 Matlab 图像处理工具箱中，提供了 edge 函数利用以上算子来检测灰度图像的边缘。对于彩色图像，首先要采用 rgb2gray(Img)函数将其转为灰度图像，再进行 edge 函数操作。

Roberts 算子: image = edge(in_image,'sobel',threshold);

Prewitt 算子: image = edge(in_image,'prewitt',threshold,direction);

Sobel 算子： image =edge(in_image,'sobel',threshold,direction);

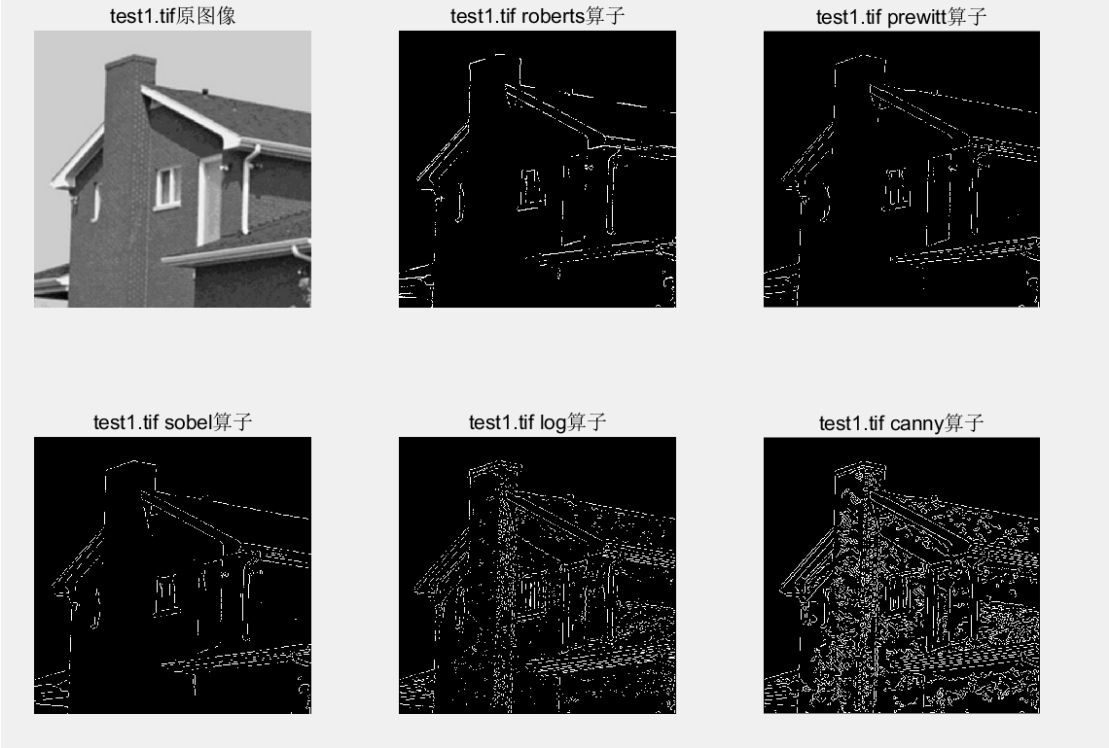
Log 算子： image =edge(in_image,'log',threshold,direction);

Canny 算子： image = edge(in_image,'canny',threshold);

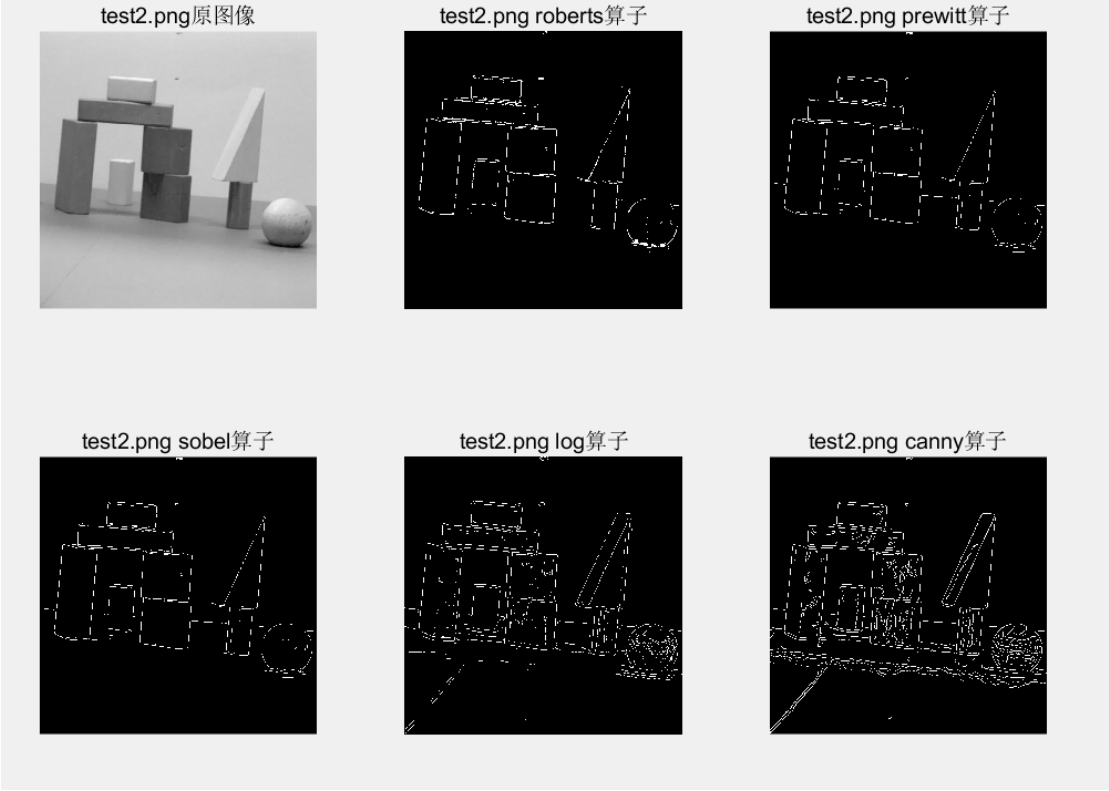
其中，in_image 是灰度图像，threshold 是阈值，direction 是方向。

3.处理结果

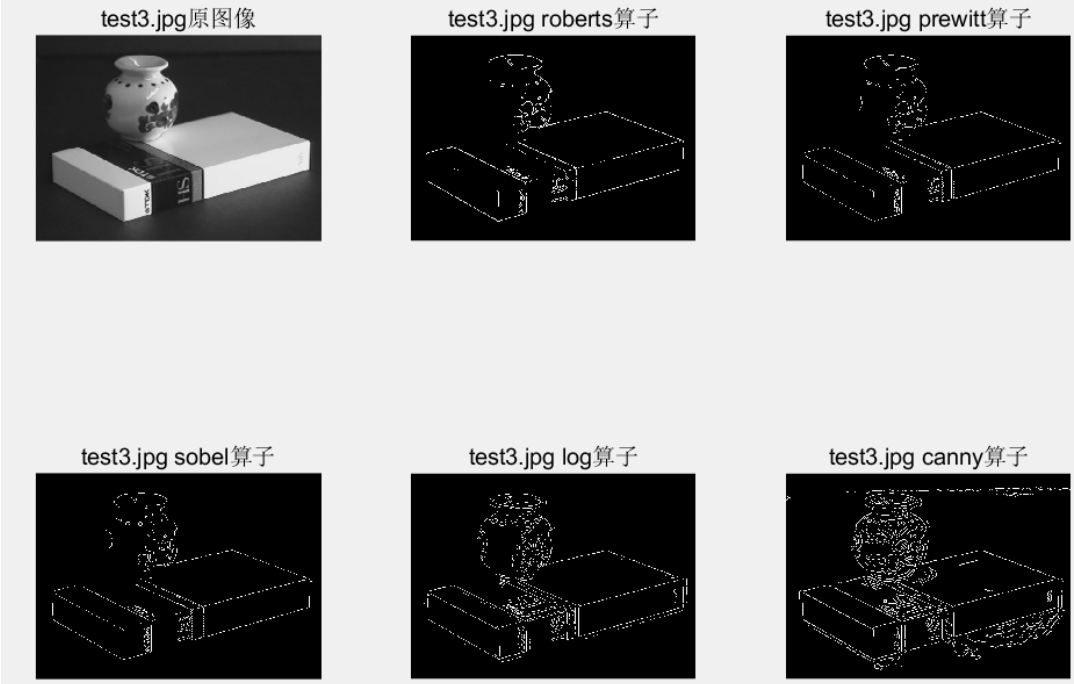
3.1 对“test1.tif”进行边缘检测



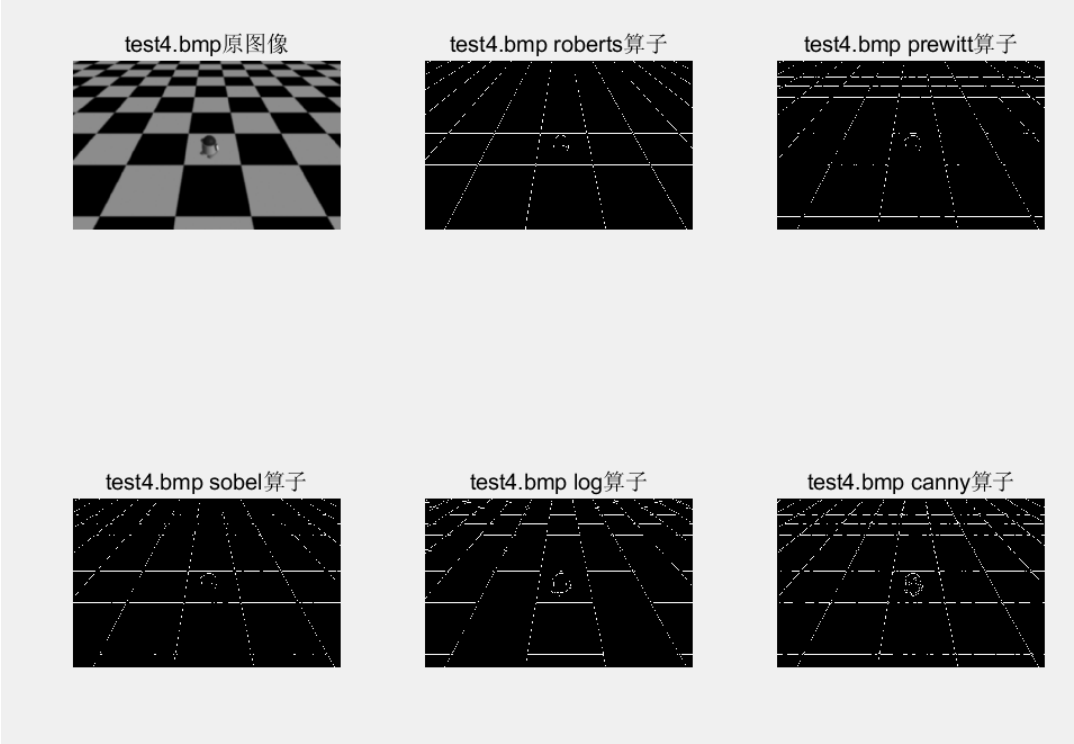
3.2 对“test2.png”进行边缘检测



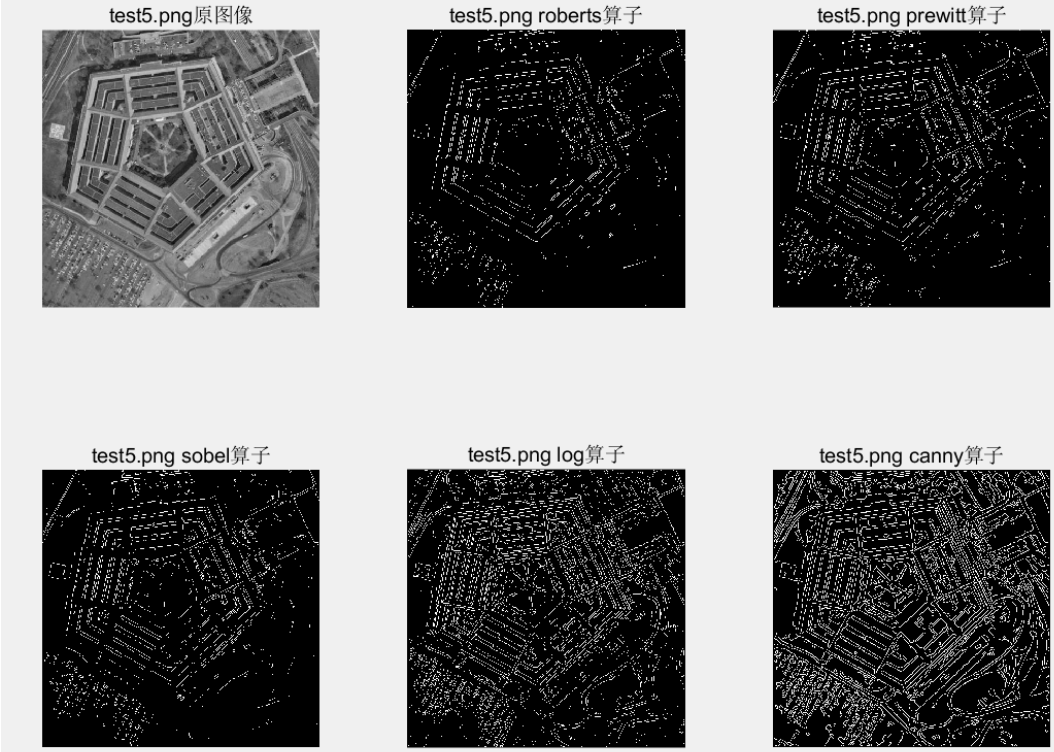
3.3 对“test3.jpg”进行边缘检测



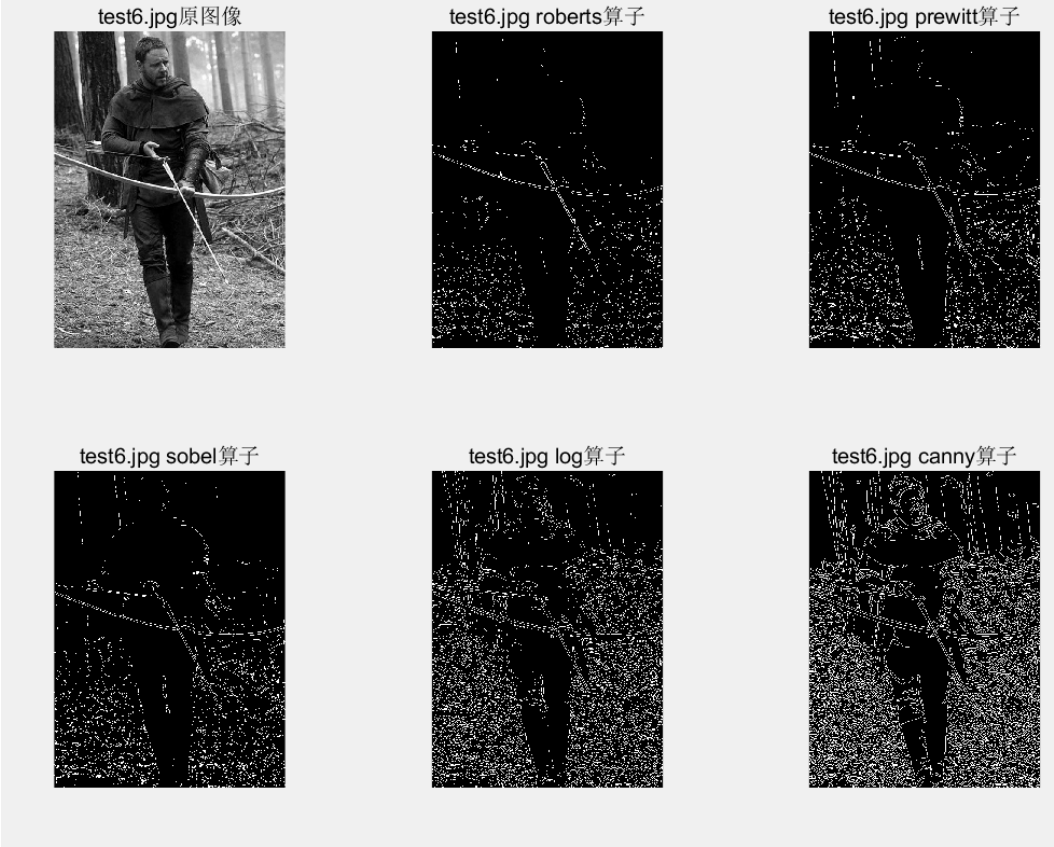
3.4 对“test4.bmp”进行边缘检测



3.5 对“test5.png”进行边缘检测



3.6 对“test6.jpg”进行边缘检测



4.结果分析

Roberts 算子实现简单、运算速度快，但易受噪声影响。主要原因：其一是实际边缘灰度与理想边缘灰度存在差异，有可能检测出多个边缘，边缘的定位不是很准确；其二是算子尺度较小，对图像细节分析能力较差。

Sobel 和 Prewitt：总体来看边缘检测效果很相近，对灰度渐变和噪声较多的图像处理效果较好，但对边缘定位不是很准确，边缘较粗。但由于 Sobel 算子是在 Prewitt 算子的基础上在中心系数上使用一个权值 2，相比较 Prewitt 算子，Sobel 模板能够较好的抑制（平滑）噪声。

Log 在检测边缘方面的结果要优于 roberts 和 sobel 算子，可以检测出更细节的部分，边缘比较完整，且抗噪能力较好，尤其是 gaussian 噪声。

Canny 算子增加了非最大值抑制和双阈值两项改进：利用非最大值抑制不仅可以有效地抑制多响应边缘，而且还可以提高边缘的定位精度，有效减少边缘的漏检率；而使用双阈值分别检测强边缘和弱边缘，并且当弱边缘和强边缘相连时，才将弱边缘包含在输出图像中。因此 Canny 边缘检测结果不容易受噪声干扰，能够检测到真正的弱边缘。

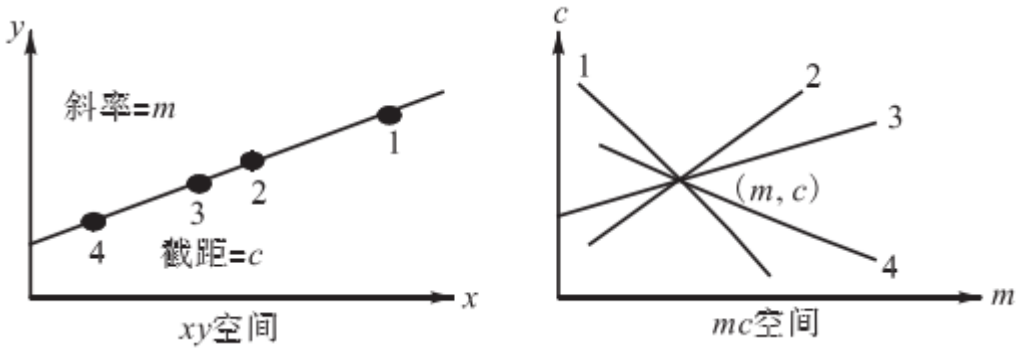
二、在边缘检测的基础上，用 hough 变换检测图中直线；

1.问题分析

1.1 霍夫变换

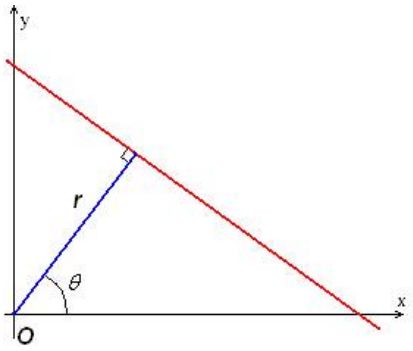
霍夫变换是图像处理中从图像中识别几何形状的基本方法之一。霍夫变换的基本原理在于利用点与线的对偶性，将原始图像空间的给定的曲线通过曲线表达形式变为参数空间的一个点。这样就把原始图像中给定曲线的检测问题转化为寻找参数空间中的峰值问题，也即把检测整体特性转化为检测局部特性。比如直线、椭圆、圆、弧线等。

霍夫变换本质上是坐标变换，如下图，左半部分表示直线的 xy 空间，直线方程为 y=mx+c y=mx+cy=mx+c，其中斜率为 m，截距为 c。右半部分表示将直线从 xy 空间变换到 mc 空间，取直线在 xy 空间上的四点 1,2,3,4，在 mc 空间就是不同斜率和截距的四条直线。



那么，在 mc 空间中四条直线的交点处的 m 值和 c 值就对应 xy 空间中直线的 m 和 c。同理，对图片中的直线来说，将直线中的每个点都变换到变换后空间，找出多条直线相交的点就可以找出变换前空间的直线。

由于以上直线的截距式方程不允许斜率为无穷大，因此在实际应用中是将 xy 空间变换到极坐标系，方程为 ρ=xcos(θ)+ysin(θ) \rho=xcos(\theta)+ysin(\theta)ρ=xcos(θ)+ysin(θ)，如下图：



其中 r，即 ρ \rhoρ 为原点到直线的垂直距离，θ \thetaθ 为与 x 轴的夹角。从上述极坐标方程可以看出，将该直线变换到 ρθ \rho\thetaρθ 坐标系后将是一系列不同初相、幅度，但是周期均为 2π 2\pi2π 的正弦曲线，所有正弦曲线交点处的 ρ 和 θ \rho 和\thetaρ 和 θ 将代表 xy 空间中的这条直线。

2.编程思路

图像处理工具箱提供了三个与霍夫变换有关的函数。函数 hough 实现了前面讨论的概念，函数 houghpeaks 寻找霍夫变换的峰值(累加单元的高计数)，函数 houghlines 以来自其他两个函数的结果为基础在原始图像中提取线段。

2.1.hough：实现霍夫变换

基本语法：[H,theta,rho] = hough(BW, ParameterName, ParameterValue)

输入参数：

BW：一幅二值图像；

ParameterName：'RhoResolution'或'Theta'

RhoResolution：指定在累计数组中（检测极值）的检测间隔，默认为 1

Theta：指定检测的角度范围（不超过-90~90 度），例如-90:0.5:89.5，默认-90:1:89

输出参数：

H：霍夫变换矩阵；

theta：x 轴和 rho 向量之间的角度；

rho：原点到直线的距离。

2.2.houghpeaks：霍夫变换峰值检测

线检测和连接用的霍夫变换的第一步是用高的计数寻找累加单元(工具箱文本把高的计数单元作为峰值)。因为存在霍夫变换参数空间中的量化和典型图像的边缘并不是很完美的直线这样的事实，霍夫变换的峰值倾向于相比霍夫变换单元更多。函数 houghpeaks 用任意默认语法来寻找指定的峰值数：

基本语法：

peaks = houghpeaks(H，numpeaks)

peaks = houghpeaks(___,Name,Value,...)

输入参数：

H：霍夫变换矩阵；

numpeaks：所检测峰值数目的最大值，默认为 1；

Name,Value：成对的参数，用于对峰值检测做出约束；

Threshold：阈值，所检测峰值的最小值。默认是 0.5*max(H(:));

输出参数：

peaks：所检测到的峰值的坐标矩阵；

2.3.houghlines：霍夫变换线检测

一旦一组候选的峰值在霍夫变换中被识别出来，如果存在与这些峰值相关的有意义的线段，剩下的就是决定线的起始点和终点。函数 houghlines 用默认的语法执行这个任务：

基本语法：

lines = houghlines(BW,theta,rho,peaks)

lines = houghlines(___,Name,Value,...)

输入参数：

BW：二值图像；

theta：霍夫变换 hough 输出的角度；

rho：霍夫变换 hough 输出的 rho；

peaks：霍夫变换峰值检测输出的峰值；

输出参数：

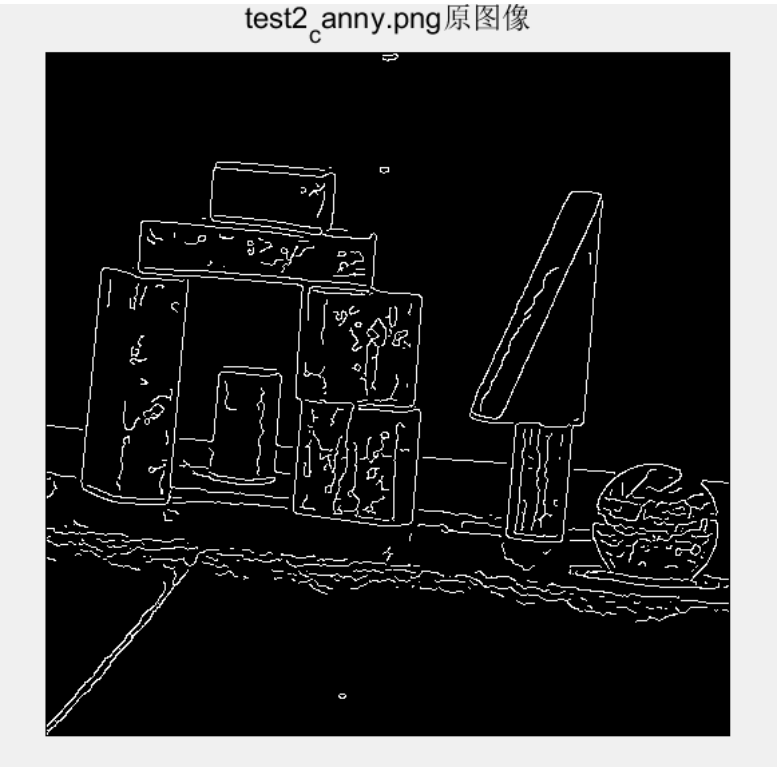
lines：是一个结构数组，包括 point1,point2,theta,rho，长度为所检测到的线的条数；

point1，2：所检测到线的起止坐标；

theta，rho：与前两个函数类似。

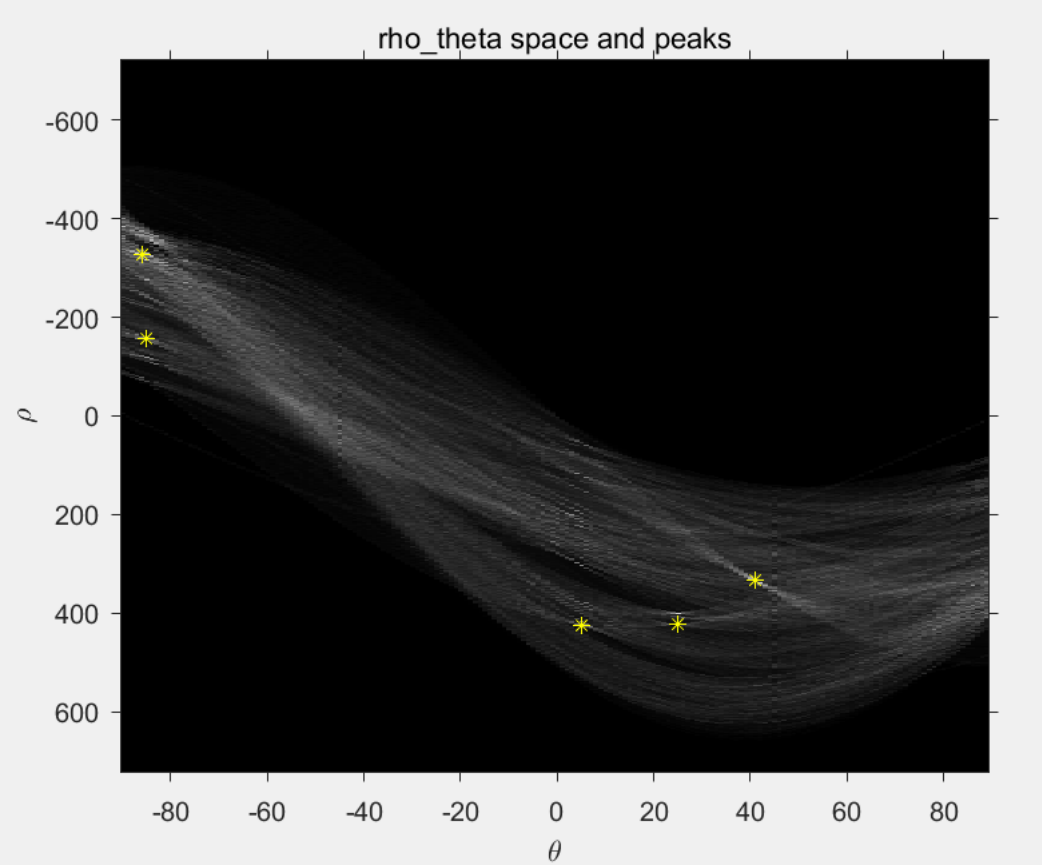
3.处理结果

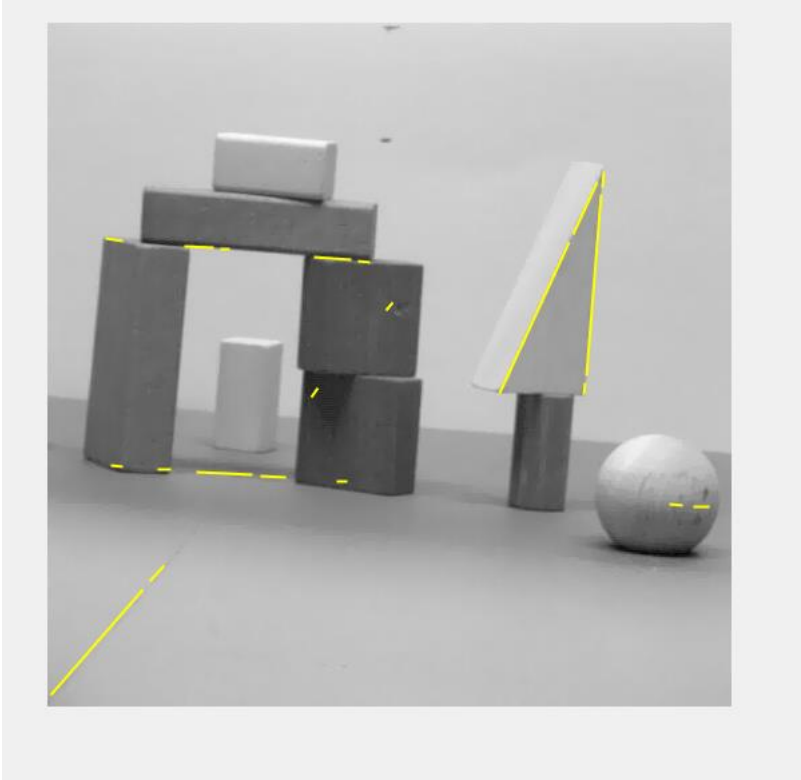
以下是“test2_canny.png”的原图像，即 test2.png 经过 canny 边缘检测后的结果：



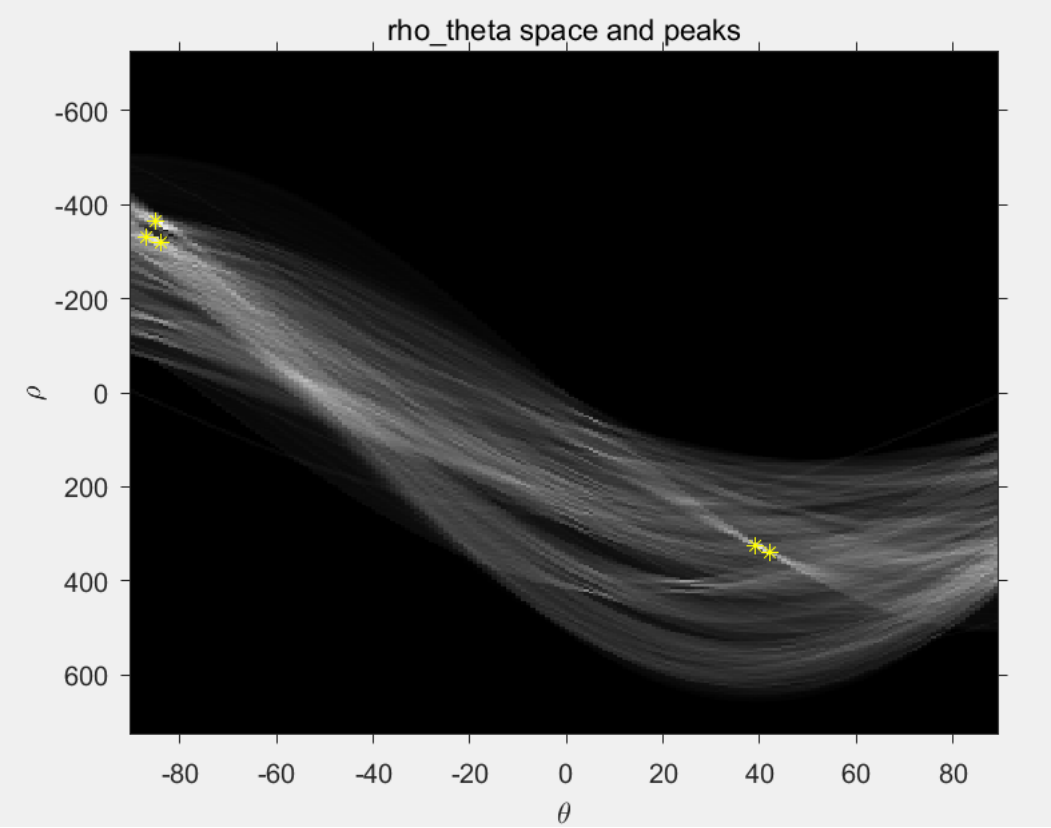
下面针对霍夫变换函数：[H,theta,rho] = hough(BW, ParameterName, ParameterValue)中的'RhoResolution'和'Theta'参数进行调节，对以上“test2_canny.png”进行霍夫变换参数不同的处理。

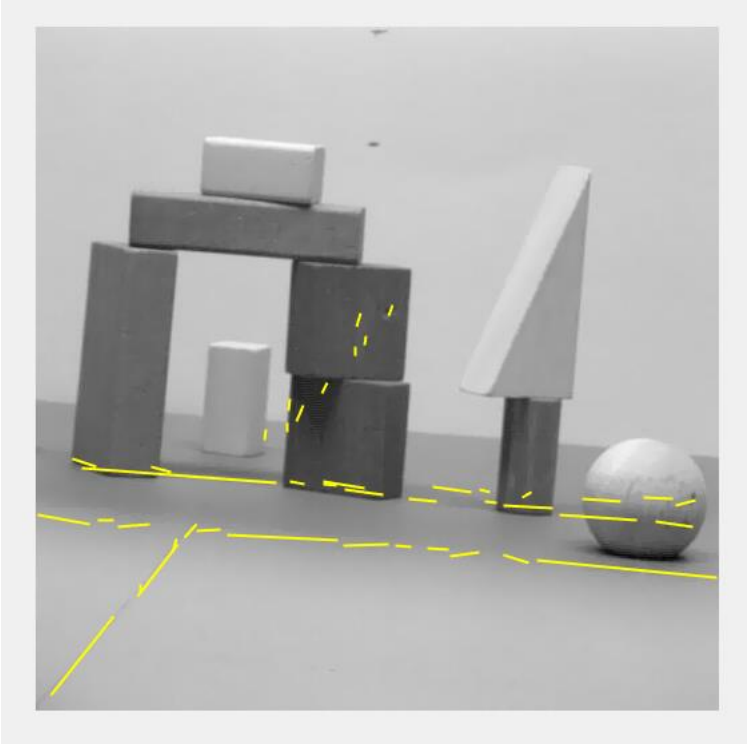
3.1 'RhoResolution'为 1，'Theta'为-90:1:89（函数默认值）



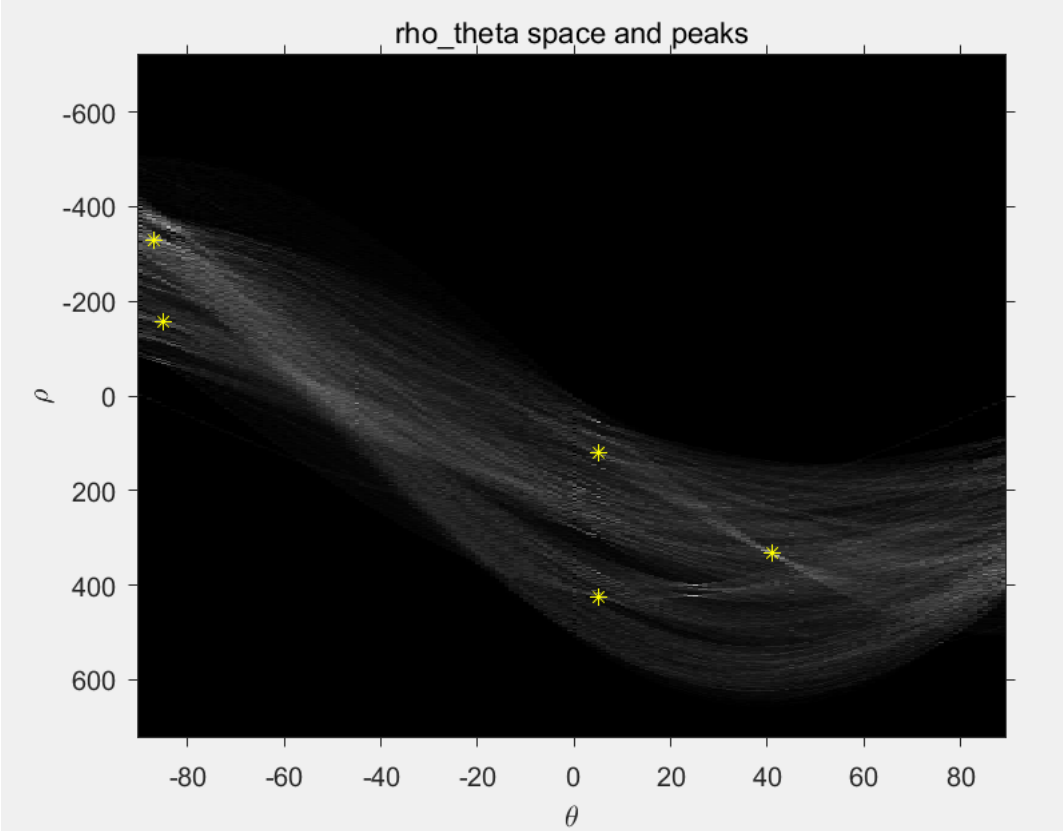


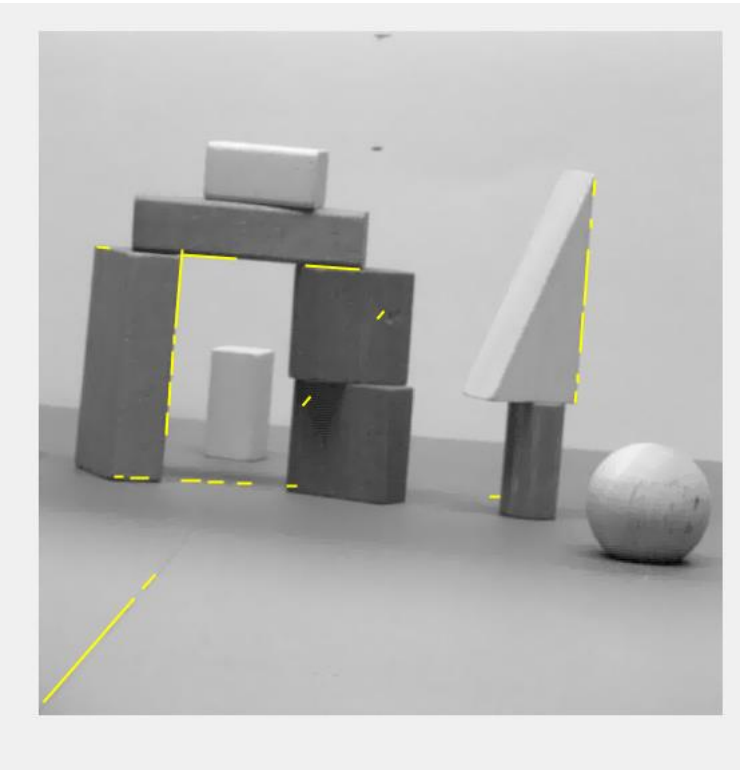
3.2 'RhoResolution'为 5， 'Theta'为-90:1:89



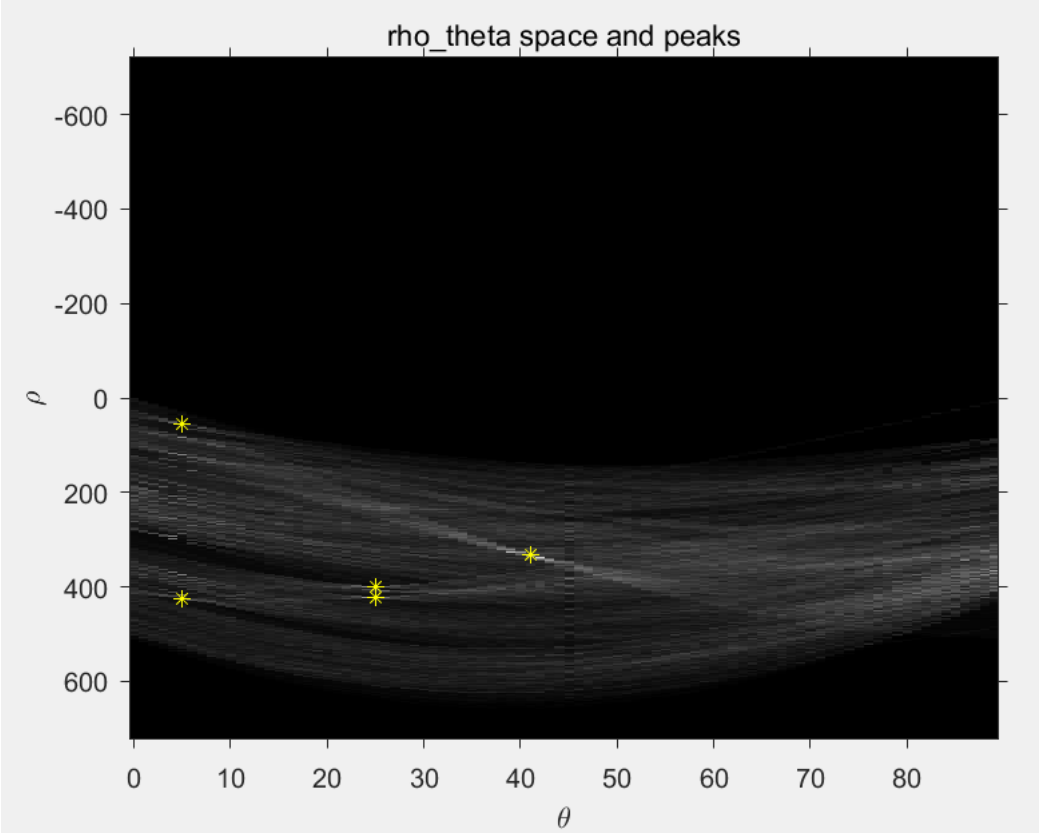


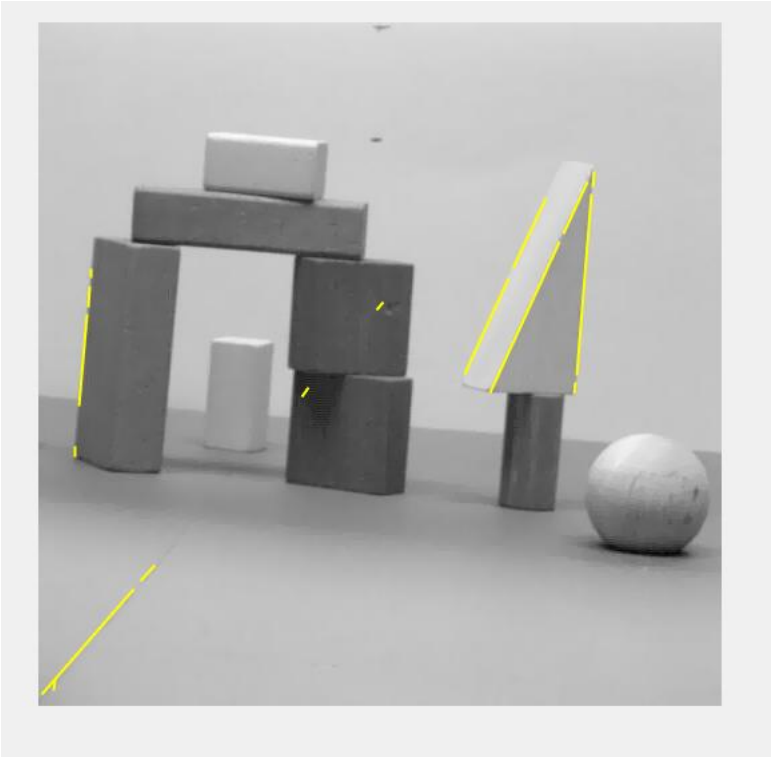
3.3 'RhoResolution'为 0.8， 'Theta'为-90:1:89



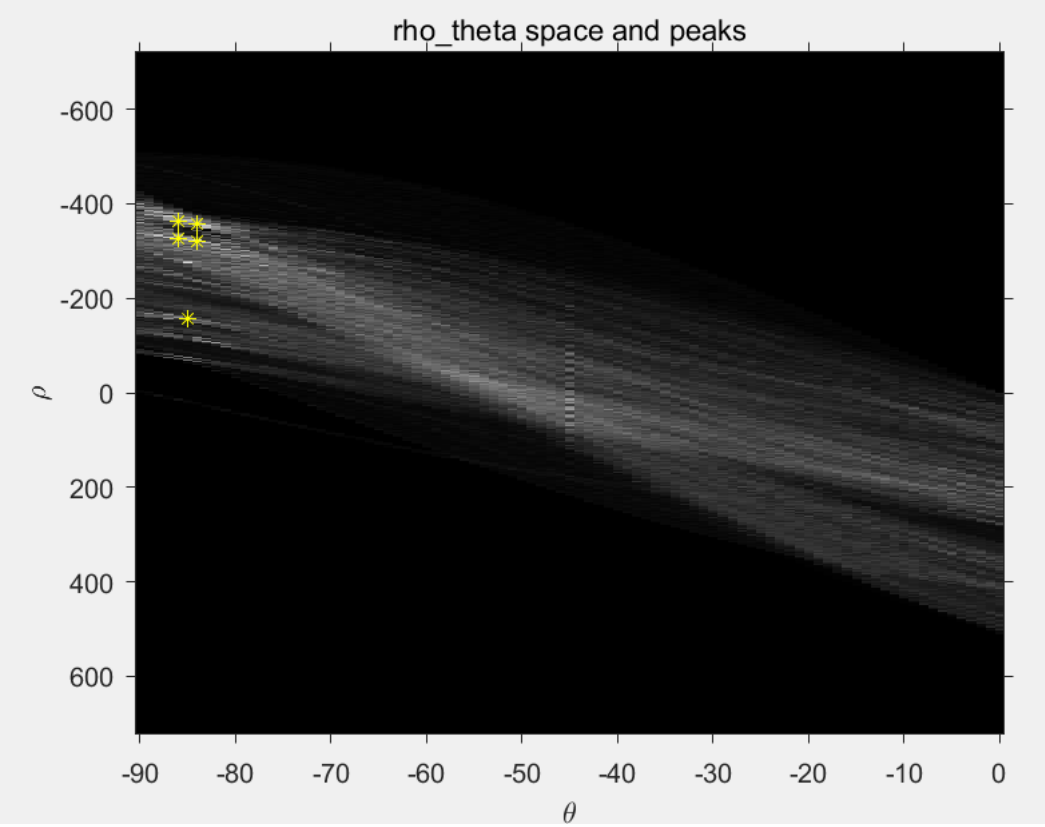


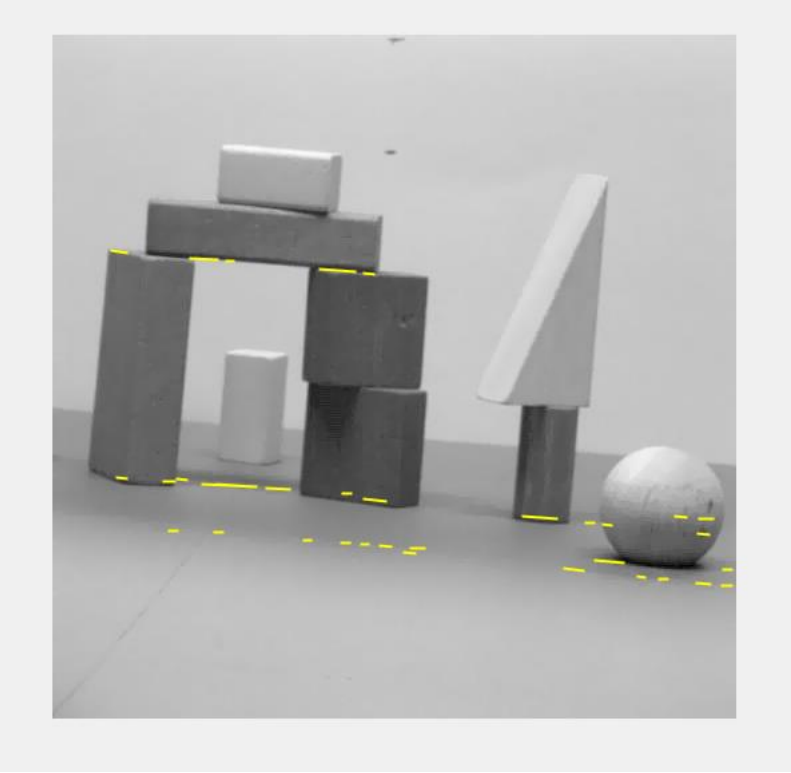
3.4 'RhoResolution'为 1，'Theta'为 0:1:89





3.5 'RhoResolution'为 1，'Theta'为-90:1:0





4.结果分析

Matlab 中 hough 函数包含两个关键参数，一个是'RhoResolution'，另一个是'Theta'。'RhoResolution'指定了在累计数组中(检测极值)的检测间隔，因此在保持'Theta'为(-90,-1,89)的情况下，采用不同的间隔值（以上分别选用 1,5,0.8 进行对比），所选出的极值点不同，因

此检测到的边缘有较大差异。'Theta'指定了检测的角度范围，当保持'RhoResolution'为 1 的情况下，采用不同的角度范围（以上分别采用-90:-1:89、0:1:90、-90:1:0 进行对比），当采用-90:-1:89 时检测到的边缘最全面，当采用 0:1:90 时仅检测到原图中倾斜向上（斜率为正）的边缘，当采用-90:1:0 时仅检测到原图中倾斜向下（斜率为负）的边缘。

参考文献

https://blog.csdn.net/qq_35206244/article/details/80174487
https://blog.csdn.net/linxid/article/details/78545678
https://blog.csdn.net/lkj345/article/details/50699981