图像分割与特征提取练习题：

1. 读取图像 ab.jpg，提取灰度直方图特征，选取合适阈值进行二值化，获得二值图像并保存。

代码：

```
Thre=128;                          %手动设置阈值
I=imread('C:\Users\hp\Desktop\ab.jpg');     %载入 rgb 图像
I=rgb2gray(I);                     %转换为灰度图像
figure;imhist(I);            %绘制灰度直方图
title('灰度直方图')
a=find(I<Thre);
b=find(I>=Thre);
e1=mean(I(a));
e2=mean(I(b));
std1 = std(im2double(I(a)) * 255, 0);
std2 = std(im2double(I(b)) * 255, 0);

x=0:1:300;
y1=normpdf(x,e1,std1);
figure,
plot(x,y1)
hold on
%y2=gaussmf(x,[std2 e2]);
y2=normpdf(x,e2,std2);
plot(x,y2)
xlabel('gaussmf, P1=[e1 v1],P2=[e2 v2]')
legend('灰度小于 128 的灰度值正态分布','灰度大于 128 的灰度值正态分布')
title('两个高斯分布')

I1=im2bw(I,128/256);
figure;
subplot(221),imshow(I1);
xlabel('固定阈值分割方法(128)');
I2=im2bw(I,140/256);
subplot(222),imshow(I2);
xlabel('固定阈值分割方法(140)');
I3=im2bw(I,90/256);
subplot(223),imshow(I3);
xlabel('固定阈值分割方法(90)');
I4=im2bw(I,180/256);
subplot(224),imshow(I4);
xlabel('固定阈值分割方法(180)');
```

输出结果：

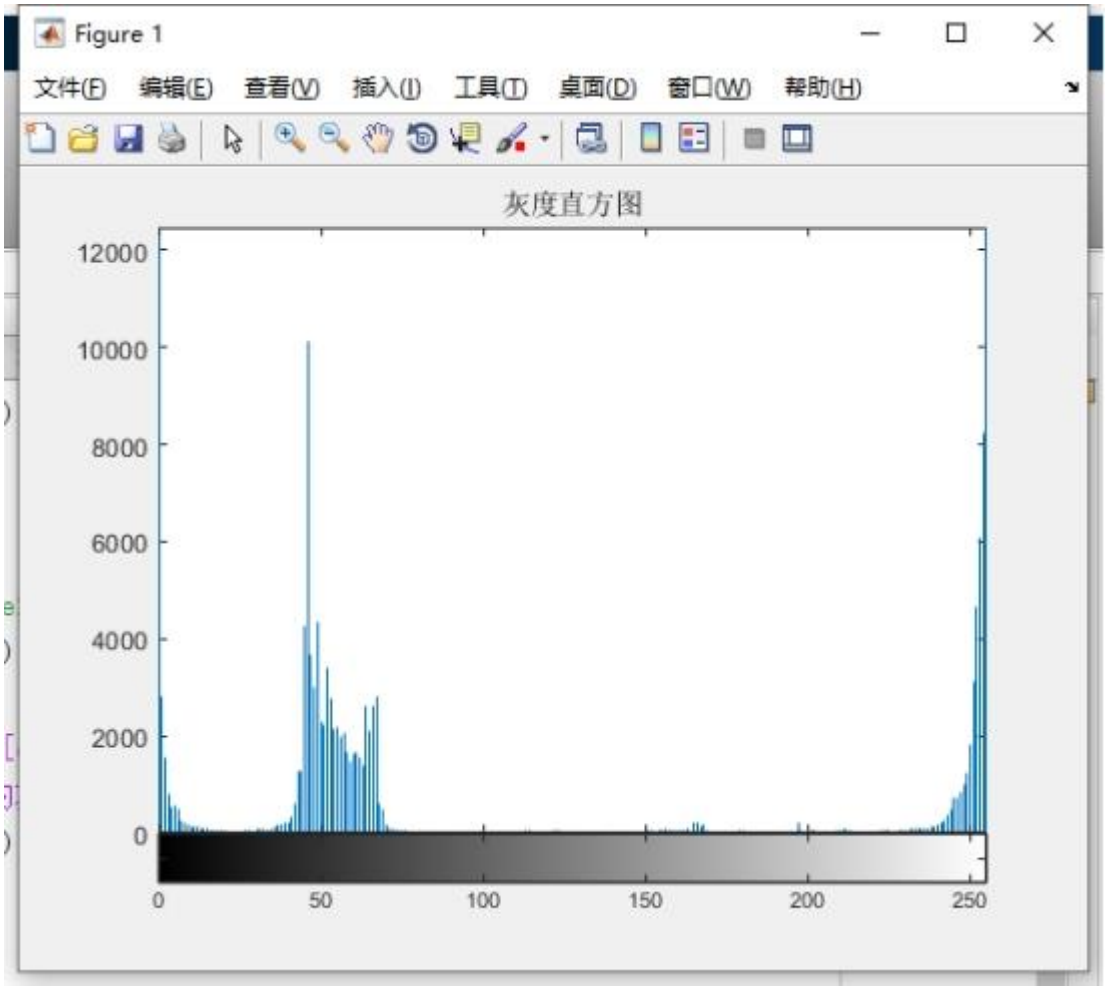


Figure 1
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
灰度直方图
12000
10000
8000
6000
4000
2000
0
0
50
100
150
200
250

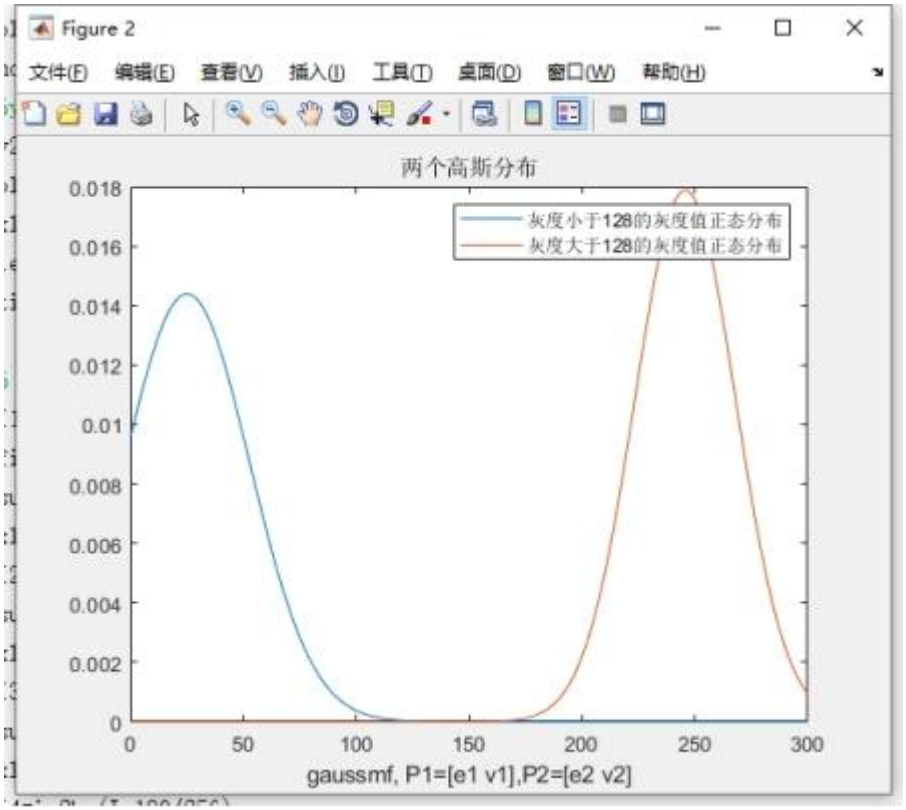
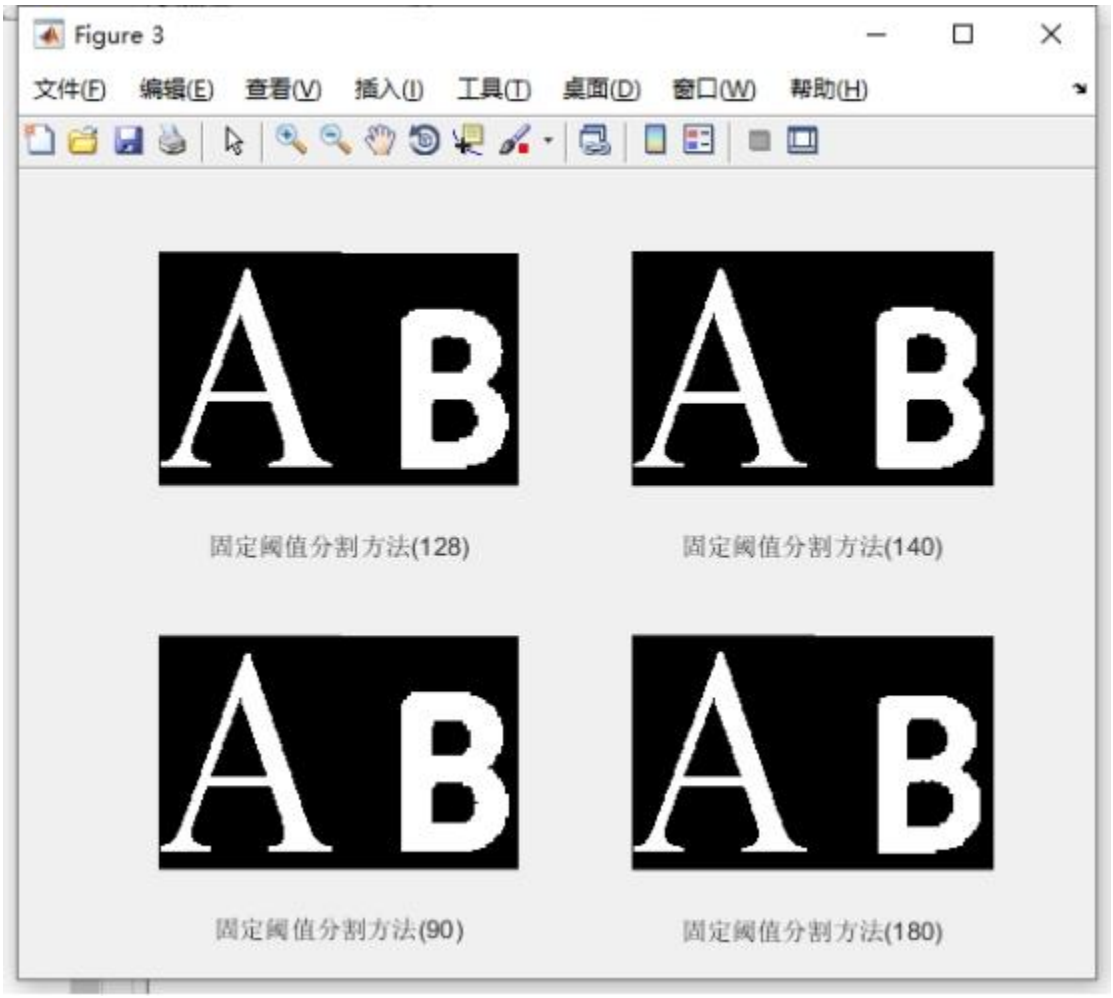


Figure 2
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
两个高斯分布
灰度小于128的灰度值正态分布
灰度大于128的灰度值正态分布
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
0
50
100
150
200
250
300
gaussmf, P1=[e1 v1],P2=[e2 v2]



分析结果：

使用 matlab 中 DIP 工具箱函数 im2bw,可以利用阈值变换法把灰度图像转换成二值图像。

（1）Figure1 展示的是灰度直方图，可以观察到，在 80~220 左右是分布比较少的，如果想选择一个阈值将两边彻底分开，就需要在这个范围内选取。

（2）正常默认二值化阈值为 128，按照大于 128 和小于 128 分为俩个组，按两组的均值和标准差绘制两个高斯分布曲线，能很好地呈现出左右两个波峰。如果两条有重叠部分，代表了图像中像素被误分的概率，原为目标物体的被分为背景，或者原为背景被分为目标物体，从而导致图像分割效果不佳。观察 figure2 可以发现：基本没有重叠部分，阈值 128 可以实现物体和背景的二值化分割。

（3）通过观察直方图确定可选阈值的范围，在这个范围内手动选择多个阈值观察效果，分别绘制了阈值选择 90，128，140，180 时候的二值化效果图像。观察图像发现还是 128 分割的效果好一些，对字符 B 的边缘部分分的更准确一些。

2．问题 1 得到的二值图像进行区域分割，分别获得目标 a、目标 b 的二值图并保存。

代码：

```
src = imread('C:\Users\hp\Desktop\ab.jpg'); %读取文件数据
figure(1);
%ostu 二值化
level=graythresh(src);
```

```
bw=im2bw(src,level);
subplot(2,2,1),imshow(bw),title('二值图像')
%开运算消噪
se = strel('disk',2);
openbw=imopen(bw,se);
subplot(2,2,2),imshow(openbw),title('开运算后的效果图')
%获取连通区域并显示
[L,num] = bwlabel(openbw,8);
RGB = label2rgb(L);
subplot(2,2,3),imshow(RGB),title('用 rgb 颜色标记不同区域')
stats = regionprops(openbw, 'basic');
centroids = cat(1, stats.Centroid);
subplot(2,2,4),imshow(openbw),title('框出目标');
for i=1:size(stats)

rectangle('Position',[stats(i).BoundingBox],'LineWidth',2,'LineStyle','-','EdgeColor','r');
        stats(i).BoundingBox
end
        stats(2).BoundingBox(1)
        abs(stats(2).BoundingBox(3)-stats(2).BoundingBox(1))


objectA=imcrop(openbw,[stats(1).BoundingBox(1),stats(1).BoundingBox(2),abs(stats(1)
.BoundingBox(3)),abs(stats(1).BoundingBox(4))]);
        figure(2);
        imshow(objectA),title('objectA');
        imwrite(objectA,'objectA.png');
        objectB                                                                    =
imcrop(openbw,[stats(2).BoundingBox(1),stats(2).BoundingBox(2),abs(stats(2).Boundi
ngBox(3)),abs(stats(2).BoundingBox(4))]);
        figure(3);
        imshow(objectB),title('objectB');
        imwrite(objectB,'objectB.png');
```

输出结果：

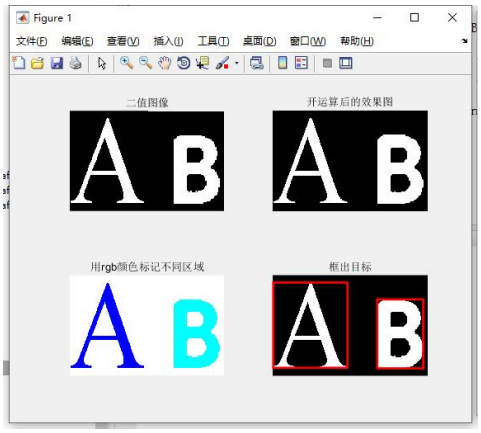


Figure 1
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
二值图像
开运算后的效果图
用rgb颜色标记不同区域
框出目标

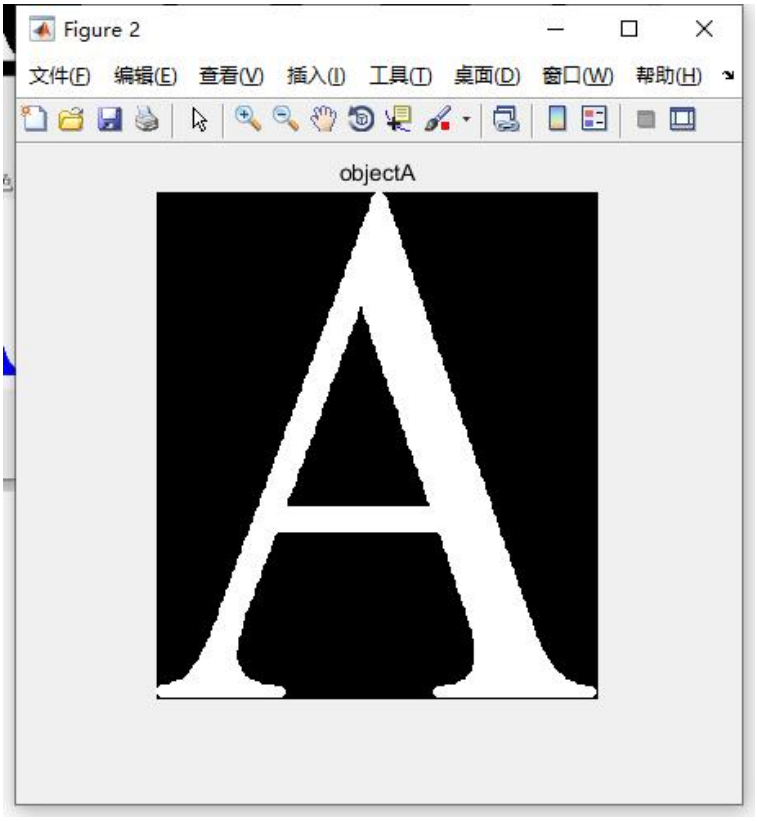
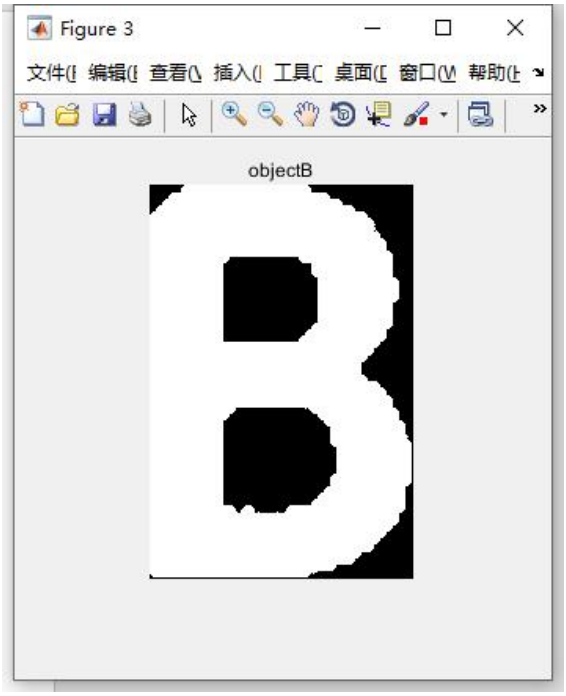


Figure 2
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
objectA



3. 对目标 a、目标 b 二值图分别计算欧拉数、面积、周长、圆形度、形状复杂度特征

代码：

```
objectA=imread('objectA.png');
objectB=imread('objectB.png');

eul1=bweuler(~objectA,8);S1=bwarea(objectA);
img1=bwperim(objectA,8);%求二值图中的边缘点
[m,n]=size(img1);
P=0;%周长初始化
for i=1:m
    for j=1:n
        if(img1(i,j)>0)
            P=P+1;
        end
    end
end
L1=P;
C1=4*pi*S1/power(L1,2);
e1=power(L1,2)/S1;
```

```
eul2=bweuler(~objectB,8);S2=bwarea(objectB);
img2=bwperim(objectB,8);%求二值图中的边缘点
[m,n]=size(img2);
P=0;%周长初始化
for i=1:m
    for j=1:n
        if(img2(i,j)>0)
            P=P+1;
        end
    end
end
L2=P ;
C2=4*pi*S2/power(L2,2);
e2=power(L2,2)/S2;
fprintf('目标 A 的欧拉数为 %8.1f\n',eul1),
fprintf('面积为 %8.2f\n',S1)
fprintf('周长为 %8.2f\n',L1)
fprintf('圆形度为 %8.5f\n',C1)
fprintf('形状复杂度为 %8.5f\n',e1)
fprintf('目标 B 的欧拉数为 %8.1f\n',eul2),
fprintf('面积为 %8.2f\n',S2)
fprintf('周长为 %8.2f\n',L2)
fprintf('圆形度为 %8.5f\n',C2)
fprintf('形状复杂度为 %8.5f\n',e2)
```

输出结果截图：

```
>> shijue3
目标A的欧拉数为        3.0
面积为 18686.75
周长为   1886.00
圆形度为   0.06602
形状复杂度为 190.34856
目标B的欧拉数为        4.0
面积为 29958.88
周长为   1411.00
圆形度为   0.18910
形状复杂度为 66.45513
```

4.将目标 a 或目标 b 进行旋转 45 度、放大一倍、镜面对称，并分别计算四幅图片的七

个不变矩特征，并对实验结果进行对比和分析。

代码：

函数封装：image_change.m

```
function image_change(image)                    %image 为要求解的图像
A=double(image);                                 %将图像数据转换为 double 类
型
[m,n]=size(A);                                   %求矩阵 A 的大小
[x,y]=meshgrid(1:n,1:m);                         %生成网格采样点的数据，x,y 的行数
等于 m，列数等于 n
x=x(:);                                          %矩阵赋值
y=y(:);
A=A(:);
m00=sum(A);                                      %求矩阵 A 中每列的和，得到 m00
是行向量
if m00==0                                        %如果 m00=0，则赋值 m00=eps，
即 m00=0
    m00=eps;
end
m10=sum(x.*A);                                   %以下为 7 阶矩求解过程，参见 7 阶
矩的公式
m01=sum(y.*A);
xmean=m10/m00;
ymean=m01/m00;
cm00=m00;
cm02=(sum((y-ymean).^2.*A))/(m00^2);
cm03=(sum((y-ymean).^3.*A))/(m00^2.5);
cm11=(sum((x-xmean).*(y-ymean).*A))/(m00^2);
cm12=(sum((x-xmean).*(y-ymean).^2.*A))/(m00^2.5);
cm20=(sum((x-xmean).^2.*A))/(m00^2);
cm21=(sum((x-xmean).^2.*(y-ymean).*A))/(m00^2.5);
cm30=(sum((x-xmean).^3.*A))/(m00^2.5);
Mon(1)=cm20+cm02;                                %1 阶矩 Mon(1)
Mon(2)=(cm20-cm02)^2+4*cm11^2;                   %2 阶矩 Mon(2)
Mon(3)=(cm30-3*cm12)^2+(3*cm21-cm03)^2;          %3 阶矩 Mon(3)
Mon(4)=(cm30+cm12)^2+(cm21+cm03)^2;              %4 阶矩 Mon(4)
Mon(5)=(cm30-3*cm12)*(cm30+cm12)*((cm30+cm12)^2-3*(cm21+cm03)^2)+(3*(c
m30+cm12)^2-(cm21+cm03)^2);                                              %5
阶矩 Mon(5)
Mon(6)=(cm20-cm02)*((cm30+cm12)^2-(cm21+cm03)^2)+4*cm11*(cm30+cm12)*(
cm21+cm03); %6 阶矩 Mon(6)
Mon(7)=(3*cm21-cm03)*(cm30+cm12)*((cm30+cm12)^2-3*(cm21+cm03)^2)+(3*c
m12-cm30)*(cm21+cm03)*(3*(cm30+cm12)^2-(cm21+cm03)^2);                   %7
阶矩 Mon(7)
m_seven=abs(log(Mon));                           %采用 log 函数缩小不变矩的动态
```

范围值

函数调用：

```
img1 = imread('objectA.png');%原图
figure;
subplot(2,4,1);imshow(img1);title('原图')
image_change(img1);
I1=imrotate(img1,45,'bilinear');%旋转变化
subplot(2,4,2);imshow(I1);title('旋转');
image_change(I1);
I2=imresize(img1,2,'bilinear');%尺度变化
subplot(2,4,3);imshow(I2);title('尺度变化');
image_change(I2);
I3=flipdim(img1,2);%原图像的水平镜像
subplot(2,4,4);imshow(I3);title('镜像变化');
image_change(I3);
img2 = imread('objectB.png');%原图
subplot(2,4,5);imshow(img2);
image_change(img2);title('原图')
I5=imrotate(img2,45,'bilinear');%旋转变化
subplot(2,4,6);imshow(I5);title('旋转');
image_change(I5);
I6=imresize(img2,2,'bilinear');%尺度变化
subplot(2,4,7);imshow(I6);title('尺度变化');
image_change(I6);
I7=flipdim(img2,2);%原图像的水平镜像
subplot(2,4,8);imshow(I7);title('镜像变化');
image_change(I7);
```

输出结果：

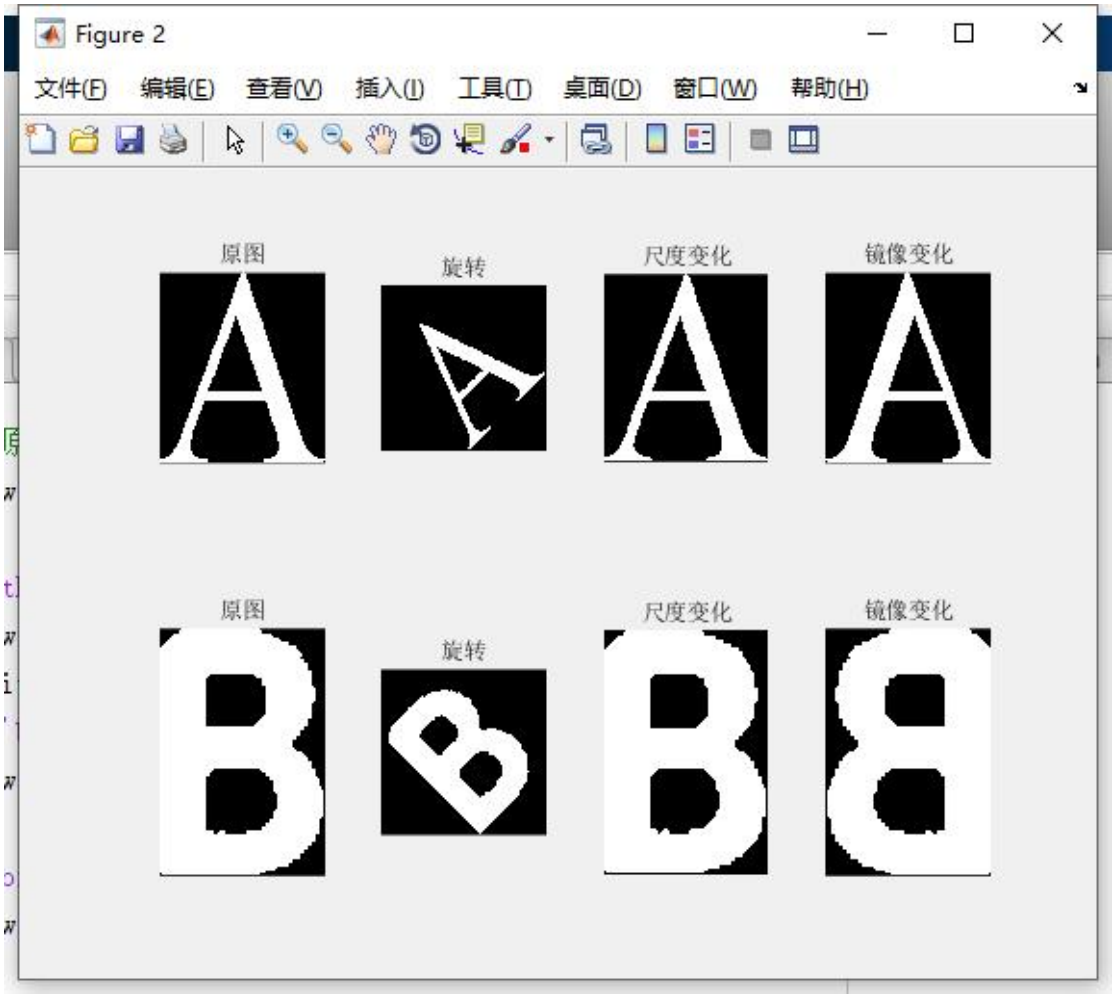


Figure 2
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
原图
旋转
尺度变化
镜像变化
原图
旋转
尺度变化
镜像变化

七个不变矩特征：

```
>> shijue4

m_seven =

    0.5940    3.2502    2.3333    5.8326    6.2914   11.6975   10.0271


m_seven =

    0.5905    3.2396    2.3281    5.8069    6.6361   12.9172    9.9715


m_seven =

    0.5939    3.2502    2.3333    5.8326    6.2914   11.6975   10.0271


m_seven =

    0.5940    3.2502    2.3333    5.8326    6.2914   11.6975   10.5077


m_seven =

    1.3587    4.5969    8.5990   12.1523   15.0410   15.1416   23.1234


m_seven =

    1.3583    4.5883    8.5770   12.0289   12.7114   15.1280   23.1302

m_seven =

    1.3587    4.5969    8.5990   12.1523   15.0410   15.1416   23.3358
```

分析结果：

1. 图像进行放大后，明显观察到细节放大像素点更加粗糙边缘齿轮状程度加深
2. 旋转是通过旋转矩阵实现，整体的边缘效果没有改变。