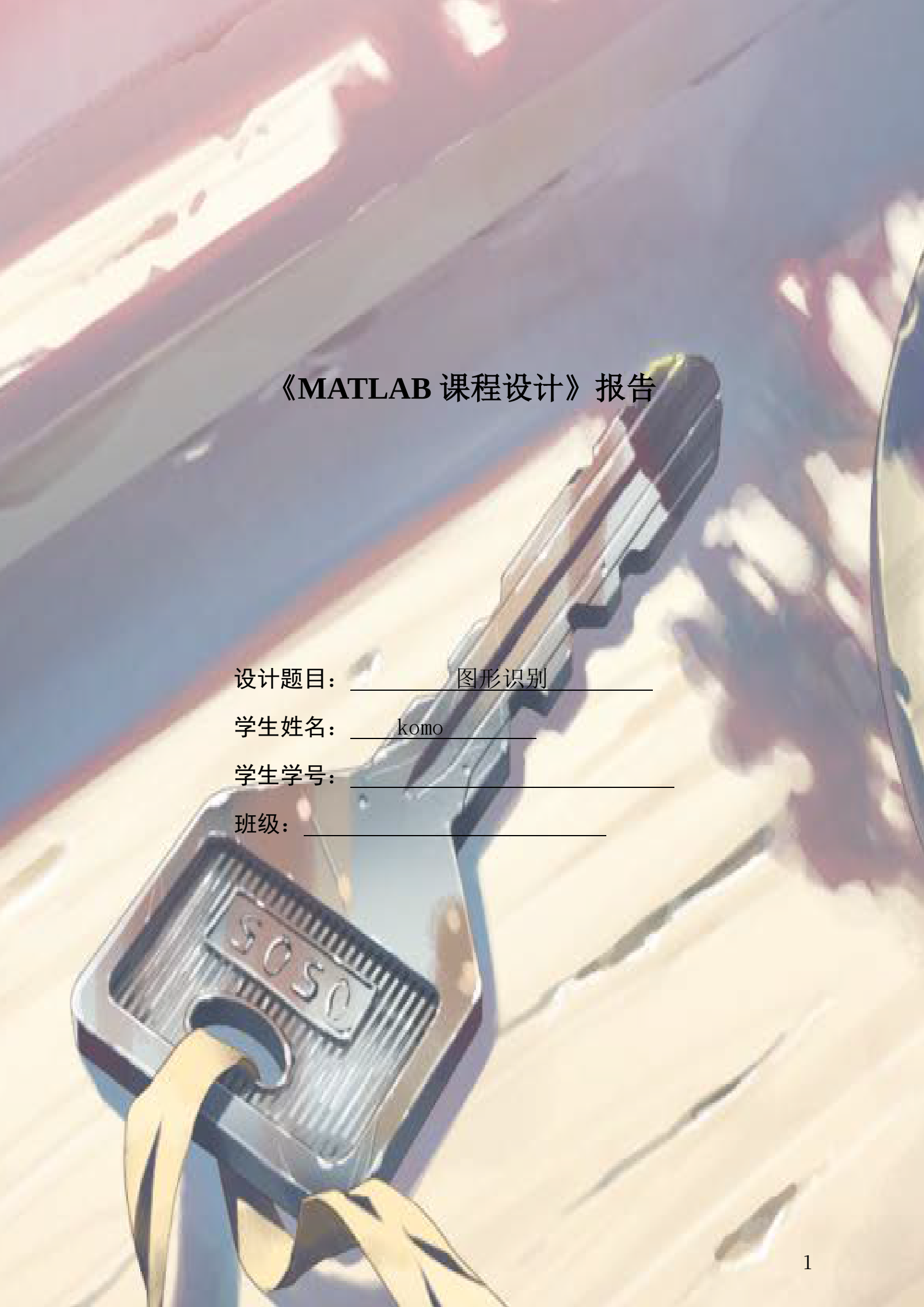


《MATLAB 课程设计》报告

设计题目：图形识别

学生姓名：komo

学生学号：

班级：

目录

一、设计目的及意义

该设计将解决使用 matlab 在同一张图片上面来识别各种规则图形，并且能较微的调节图像，使得他能够更好的进行处理。该设计一方面能够作为物象识别的基础部分，用来提取相关图形进行处理。也可作为一些图像处理的扩展程序，使得其能一键识别图像中的相关规则元素，加快图像开发速度。

二、设计任务及分工

完成功能：最终完成的程序能够识别图片中的规则图形，并且能够标记其质心。

分工：

程序设计—谭佳滨、陈沛唯
PPT 制作—陈沛唯
报告填写—谭佳滨

三、设计过程

1・读取图像
2・因为彩图处理过于麻烦，而且相关功能受限，因此需要先转换为灰度

图

3 • 因为后面需要对图像进行边缘检测，因此在检测前需要使图像去噪，防止出现局部的高频分量。

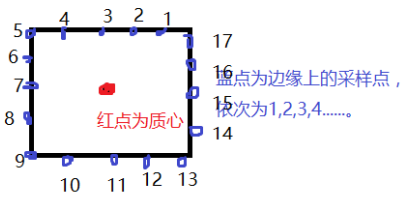
4 • 边缘检测

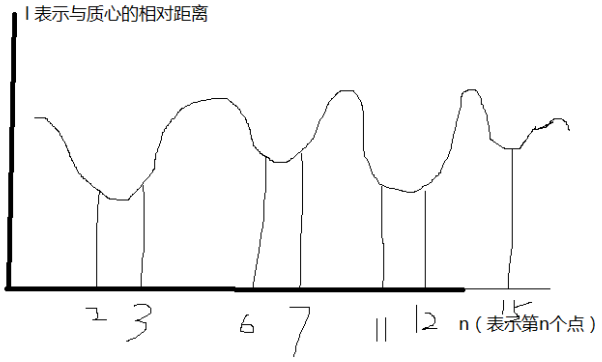
5 • 获取各个图形边缘

6 • 算质心

7 • 根据边缘个点到质心的距离，列出函数

8 • 根据函数的极值判断图形。(此处阐释较为麻烦，请参考如下示意图)。





可发现，对于矩形，其应该有四个极小值与4个极大值，根据这些特性，可分辨出图形的类别。

改进代码思想：

因为问题的出现是 bwlabel 函数的取值无规律造成的，因此，只需要将其取得的点按顺时针或者逆时针排序后再次进行后序操作，即可获得与我们预期一样的效果。这边需要注意的是，因为图片可能会出现特殊的读取情况，导致我的 edgetrack 函数读取不到所有点，此时需要一个函数_isFinished 来判断是否还需要读取。

isFinished 函数的思想是读取以当前点位置为中心+-15 个单位的矩阵。然后取得最近的点，然后在进行读点。

四、结论及效果图

①在去除边框过程中，因为我们采用的是阈值法，因而无法正确控制边缘去除大小，此时可以通过图像腐蚀使得边缘最大化（对非边缘部分影响非常小）。

②在质心过程中，因为我们是使用bwlabel来确定图形的，因此需要将其转置后去坐标，才能获得正确的坐标。

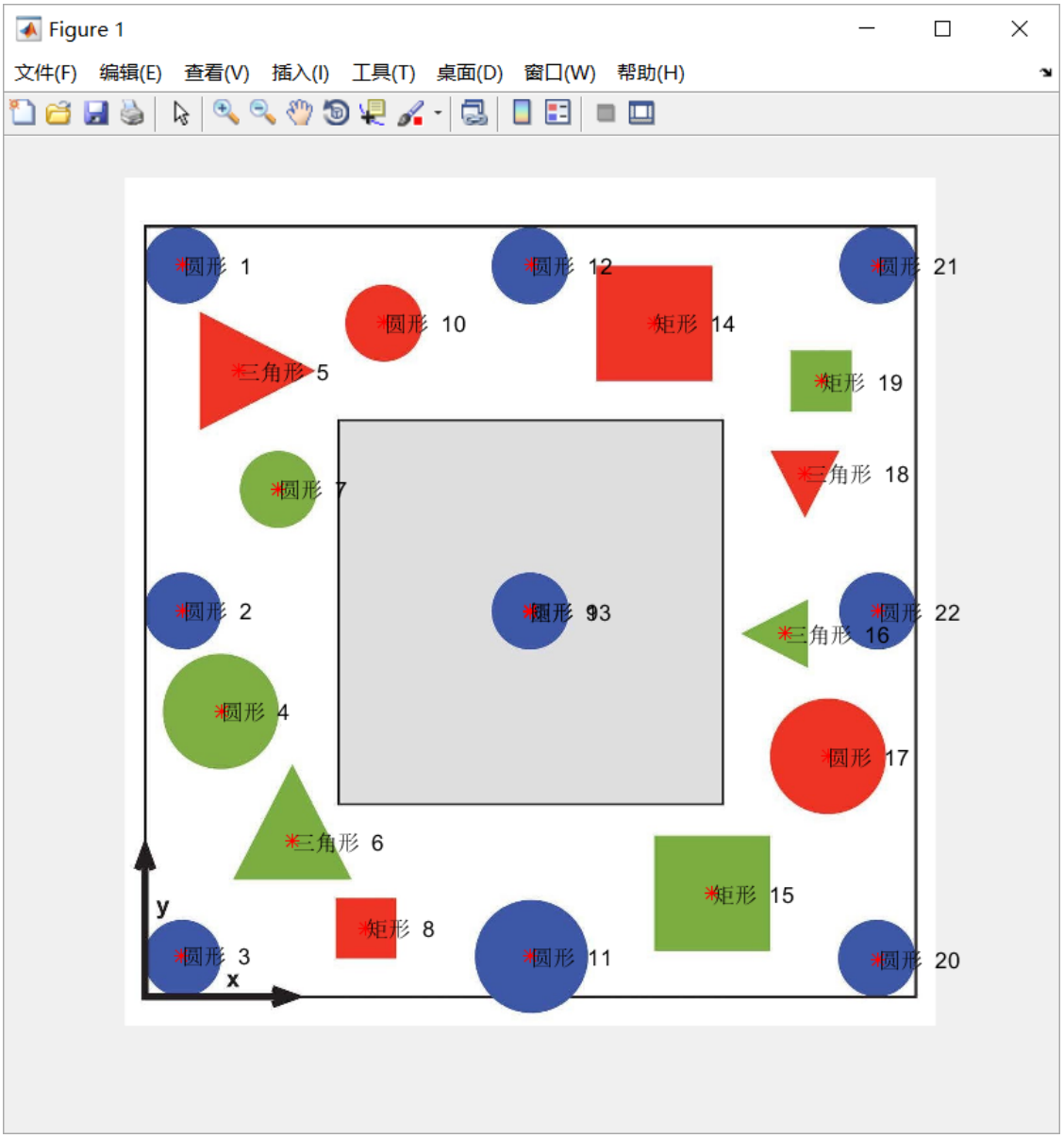
③拟合函数过程中，要注意到bwlabel的坐标计数位置的特殊性，因此取最大长度的极点时，只有三个极点。

④因为bwlabel函数的特殊性，因此对于图像的要求比较严格。比如三角形需是等腰，且防止方向有要求。

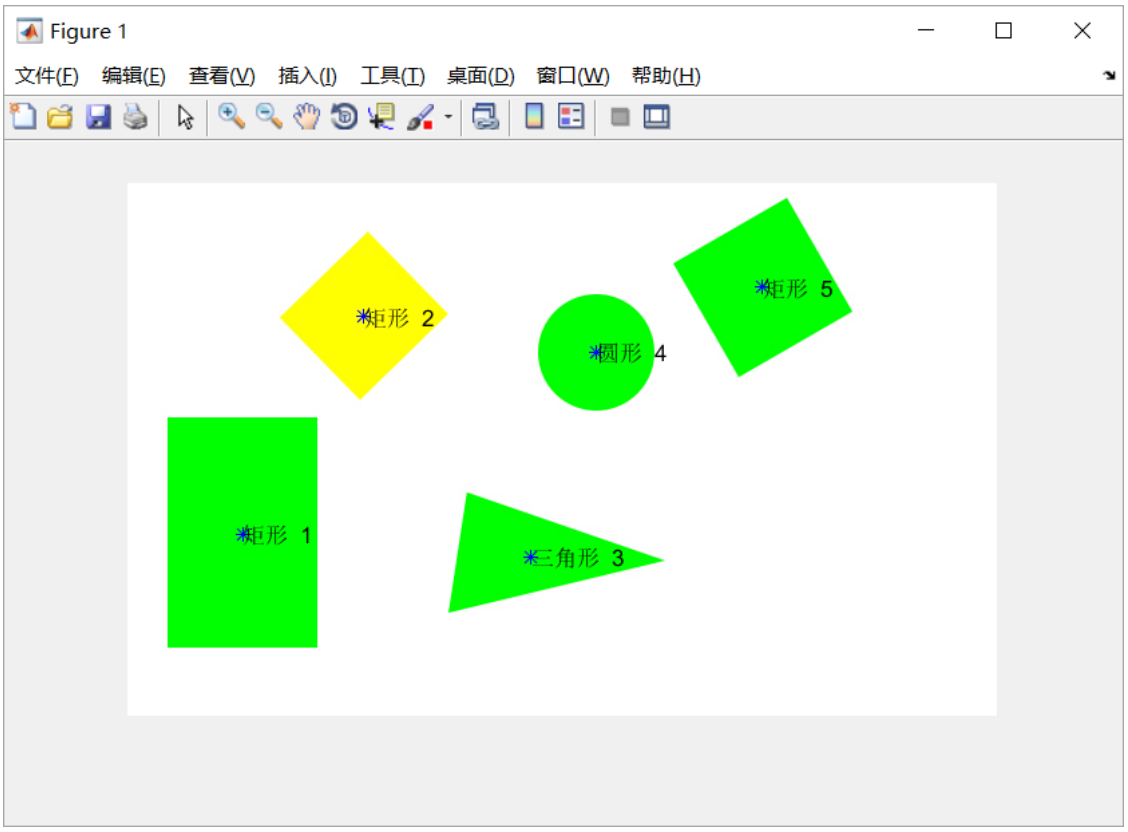
⑤鉴于程序受限原因为bwlabel取点原因，因此，我可以使用边缘追踪将取点依

次获得。改进代码见代码清单的改进部分。

未优化代码效果图



优化后代码效果图：



五、设计体会

对于图像处理，主要还是要理解，对于上面的程序，在 bwlabel 这个函数上，因为在起先，我们将它的取点位置当做是呈环形依次取点的，因此，最后发现其存在非常多的与我们预期完全不同的地方。比如极值以及取点后的再现图状态。对于这个问题，我在代码中添加了多处图像状态追踪代码。比如：

```
% %     for k=1:25
% %
s=['plot(r(',num2str(k*10),'),c(',num2str(k*10),''
b*'')'];
% %         eval(s);
% %         text(r(k*10),c(k*10),sprintf('%d',k))
% %     end
%
```

用来观察 bwlabel 的取点方式。最终将函数作用弄明白。

最重要的是，我觉得 matlab 在图像处理方面并不是非常友好，我每次调试都需要在庞大的图像数据中一个一个找过去。

六、程序清单即效果图

```
f4=imread('1.jpg');      %读取图像
 imshow(f4);
```

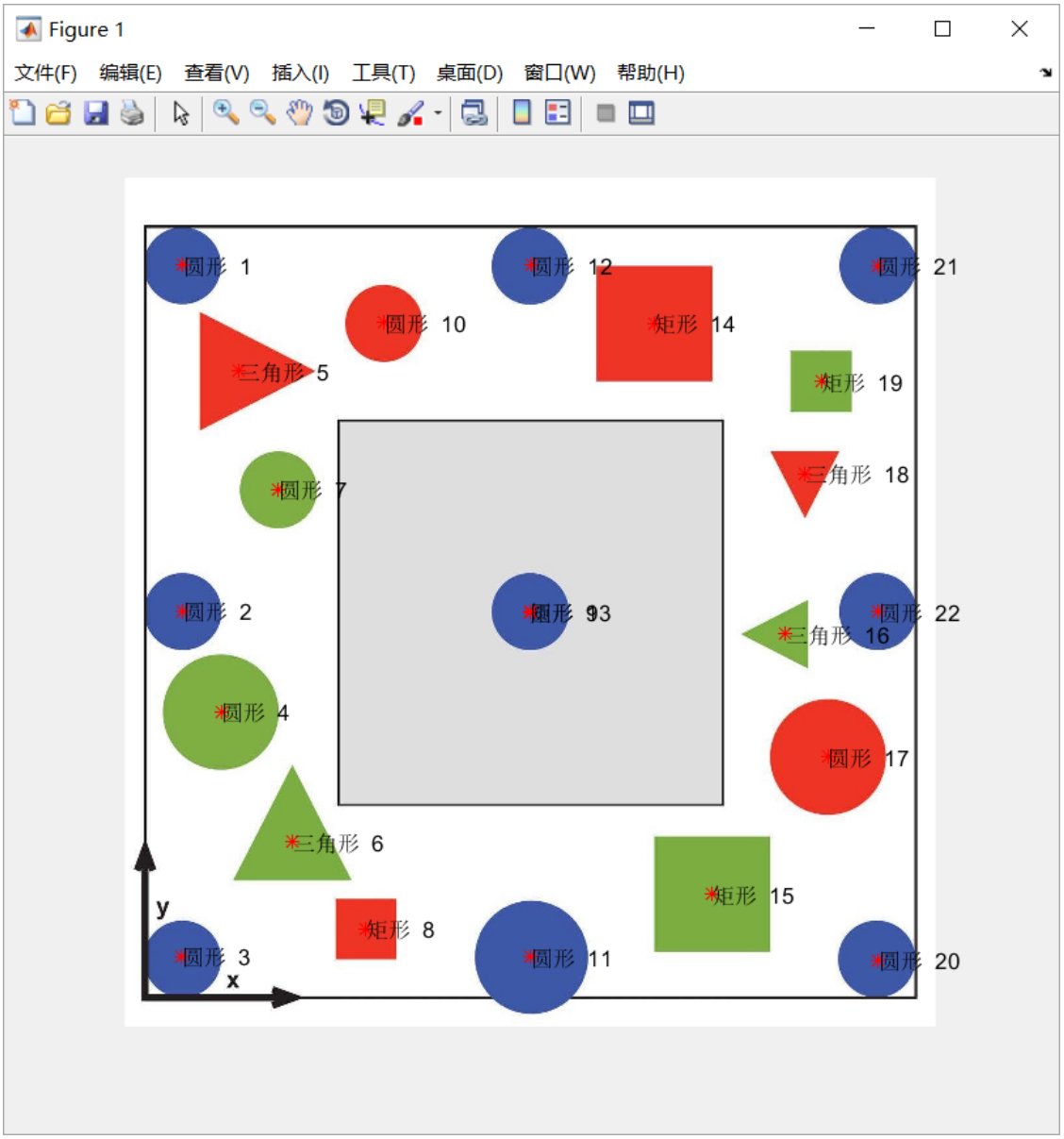
```
a=rgb2gray(f4);              %将彩色图像转换成灰度图像
a_size = size(a);
b = ones(a_size);

for i =1:a_size(1)%去掉边框
    for j = 1:a_size(2)
        if a(i,j)>=0 && a(i,j)<=50

            b(i,j)=0;
        end
    end
end

 B =[1 1 1 1;1 1 1 1;1 1 1 1;1 1 1 1];  %扩大边缘宽度，这是一个需要注意的地方
  b = imerode(b,B);
for i =1:a_size(1)
    for j = 1:a_size(2)
        if  b(i,j)==0
            a(i,j)=255;
        end
    end
end

bw=edge(a,'prewitt');%边缘检测    边缘检测结束后发现还是有一些鼓励的小点，不多它们没有形成闭合的曲线
[L,num] = bwlabel(bw);%取得联通块

%这里已经给每个区域标好号了,使用 bwlabel 的话会把鼓励的不成闭合曲线的点也算进去
%一些独立点的像素数量是比较少的,所以可以通过检测每一块区域的像素点大小
```

来决定是不是要删除此像素块

```
for i= 1:num    %去噪
        [r,c]=find(L==i);
        size_L = size([r,c]);
        if size_L(1,1)<30
            L(r,c)=0;
        end
end
L = logical(L);%单一化，方便处理

se = strel('disk',4);   %平滑图形，补足圆形
L = imclose(L,se);    %闭处理，先膨胀，后腐蚀
[L,num1] = bwlabel(L);%平滑后的联通块
L = rot90(L,3);
L = fliplr(L);%转置
pixel = cell([num1,1]);%产生空矩阵
centre = zeros(num1,2);%质心
size_L = size(L);%行列向量
for i=1:num1
    [r,c]=find(L==i);%寻找联通块
%     points=edgetrack(L);
%     figure,imshow(L(points));
    pixel{i} = [r,c];

    hold on
    mean_pixel = mean(pixel{i});%质心坐标
    centre(i,:) = mean_pixel;   %记录质心
    plot(mean_pixel(1,1),mean_pixel(1,2),'r*');%画出质心
    size_r = size(r);
    distance = zeros(size_r);
    for j = 1:1:size_r(1)
```

```
            distance(j)      =      sqrt((r(j)-mean_pixel(1))^2      +
(c(j)-mean_pixel(2))^2); %算出各个图形边缘距离质心的长度
    end
    p=polyfit((1:size_r(1))',distance,7);
    x = (1:size_r(1))';
    y = p(1)*x.^7 + p(2)*x.^6 + p(3)*x.^5 + p(4)*x.^4 + p(5)*x.^3 +
p(6)*x.^2 + p(7)*x.^1 + p(8);  %产生拟合函数

    min_distance = min(distance);
    max_distance = max(distance);
    min_y        =  min(y);
    max_y        =  max(y);
    num_peaks    =  size(findpeaks(-y));

%      求极小值
%根据极小值判断形状
    if (max_distance - min_distance)<= 15 && (max_y - min_y) <= 15%当边
缘与质心距离相等时，为圆
        text(mean_pixel(1,1),mean_pixel(1,2),sprintf('圆形  %d',i))
    elseif num_peaks(1) == 2%有 2 个波峰，为三角形
        text(mean_pixel(1,1),mean_pixel(1,2),sprintf('三角形  %d',i))
    else%三个波峰为矩形
        %也可以写成 num_peaks(1)==3 时，为矩形
        text(mean_pixel(1,1),mean_pixel(1,2),sprintf('矩形  %d',i))
    end
end
```

效果展示：



七、代码改进（让代码取点为顺时针）

因为代码过多，并且跟上面代码过较多相似之处，这边仅贴关键函数代码。具体代码我已经传到本人 github。

```
function points=edgetrack(edgeIm,num)
[i, j] = find(edgeIm==num);
[row,col]=size(edgeIm);
%轮廓点数
numPoints = size(i, 1);
curNum = 0;
```

```
%初始搜索点坐标
currentR = i(1, 1);
currentC = j(1, 1);

%初始化轮廓点坐标
% points = zeros(numPoints, 2);

%开始搜索
curNum = curNum + 1;
points(curNum,:) = [currentR, currentC];
edgeIm(currentR, currentC) = 0;
while curNum ~= numPoints

 if(curNum==335)
                a=1;
                end
          if currentC-1>0&&edgeIm(currentR, currentC-1)== num
                curNum = curNum + 1;
                currentC = currentC - 1;
                points(curNum,:) = [currentR, currentC];
                edgeIm(currentR, currentC) = 0;

                elseif          currentC-1>0&&currentR-1>0&&edgeIm(currentR-1,
currentC-1)== num
                     curNum = curNum + 1;
                      currentR = currentR - 1;
                      currentC = currentC - 1;
                      points(curNum,:) = [currentR, currentC];
                      edgeIm(currentR, currentC) = 0;

                elseif currentR-1>0&&edgeIm(currentR-1, currentC)== num
```

```
                curNum = curNum + 1;
                currentR = currentR - 1;
                points(curNum,:) = [currentR, currentC];
                edgeIm(currentR, currentC) = 0;

            elseif currentR-1>0&&currentC+1<=col&&edgeIm(currentR-1,
currentC+1)== num
                curNum = curNum + 1;
                currentR = currentR - 1;
                currentC = currentC + 1;
                points(curNum,:) = [currentR, currentC];
                edgeIm(currentR, currentC) = 0;
            elseif currentC+1<=col&&edgeIm(currentR, currentC+1)== num
                curNum = curNum + 1;
                currentC = currentC + 1;
                points(curNum,:) = [currentR, currentC];
                edgeIm(currentR, currentC) = 0;

            elseif currentC+1<=col&&currentR+1<=row&&edgeIm(currentR+1,
currentC+1)== num
                curNum = curNum + 1;
                currentR = currentR + 1;
                currentC = currentC + 1;
                points(curNum,:) = [currentR, currentC];
                edgeIm(currentR, currentC) = 0;

            elseif currentR+1<=row&&edgeIm(currentR+1, currentC)== num
                curNum = curNum + 1;
                currentR = currentR + 1;
                points(curNum,:) = [currentR, currentC];
                edgeIm(currentR, currentC) = 0;
```

```
                elseif      currentR+1<=row&&currentC-1>0&&edgeIm(currentR+1,
currentC-1)== num
                    curNum = curNum + 1;
                   currentR = currentR + 1;
                   currentC = currentC - 1;
                   points(curNum,:) = [currentR, currentC];
                    edgeIm(currentR, currentC) = 0;
          elseif numPoints-curNum>14

              Lt=currentR-15;
              Lb=currentR+15;
              Cl=currentC-15;
              Cr=currentC+15;
              if currentR<15
                   Lt=0;
              elseif row-currentR<15
                   Lb=row;

              end
              if currentC<15
                   Cl=0;

              elseif col-currentC<15
                   Cr=col;

              end
              A=edgeIm([Lt:Lb],[Cl:Cr]);
              [Ar,Ac]=find(A==num);
              k=size(Ar);
              if k(1,1)==0||k(1,2)==0
```

```
                break;
            end
            A=(Ar-16).^2+(Ac-16).^2;
            [data,Arow]=min(A);
            currentR=currentR+Ar(Arow)-16;
            currentC=currentC+Ac(Arow)-16;
            curNum=curNum+1;
            points(curNum,:) = [currentR, currentC];
            edgeIm(currentR, currentC) = 0;
        else break;
        end
end
end
```

效果图