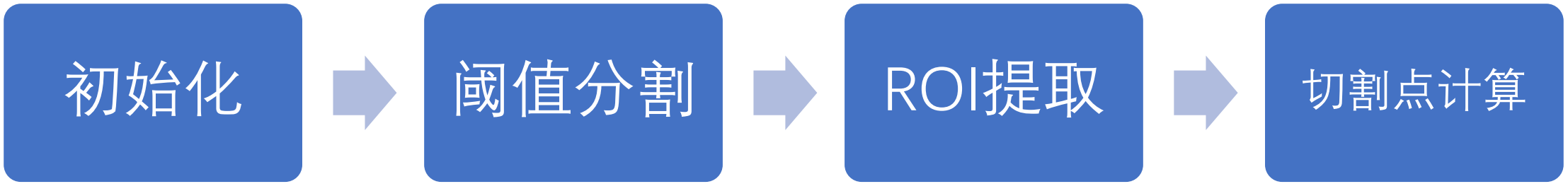
机器人收获果实时切割点的提取

——潘盛琪

实验原理

- 在草莓、黄瓜、西瓜等果梗细长的果蔬自动收获中，机器人通常切刀等方式切断果梗，实现果实与植株本体的分离。在分离前需要在尽量靠近果实与果梗连接处选择**切割点**。
- 采用分块定位法来计算矩形兴趣区内的切割点坐标

程序框图



困难

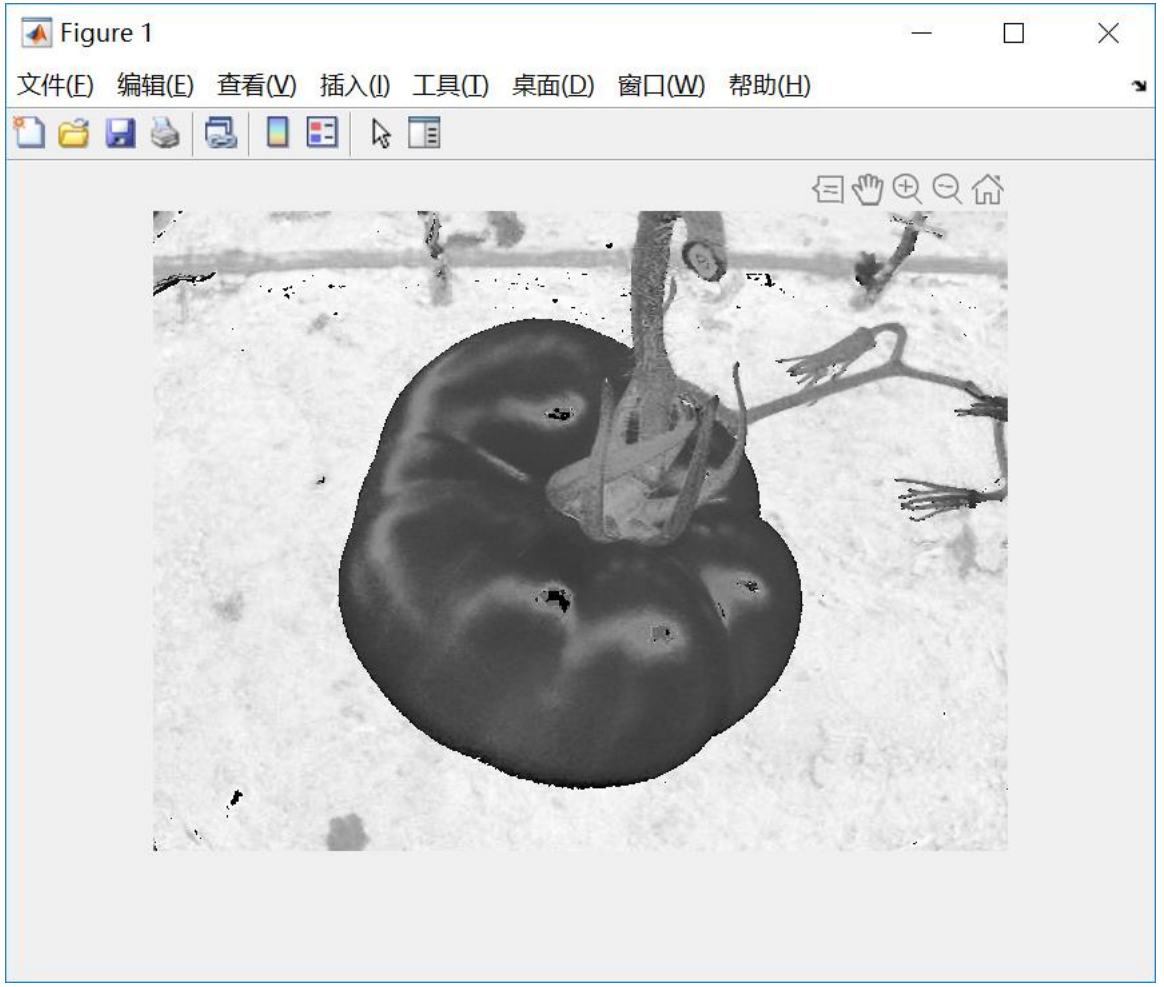
- 输入图片背景杂乱，不利于阈值分割
- 输入图片转化为灰度图后，灰度变化不明显

原图像

迭代阈值法图像

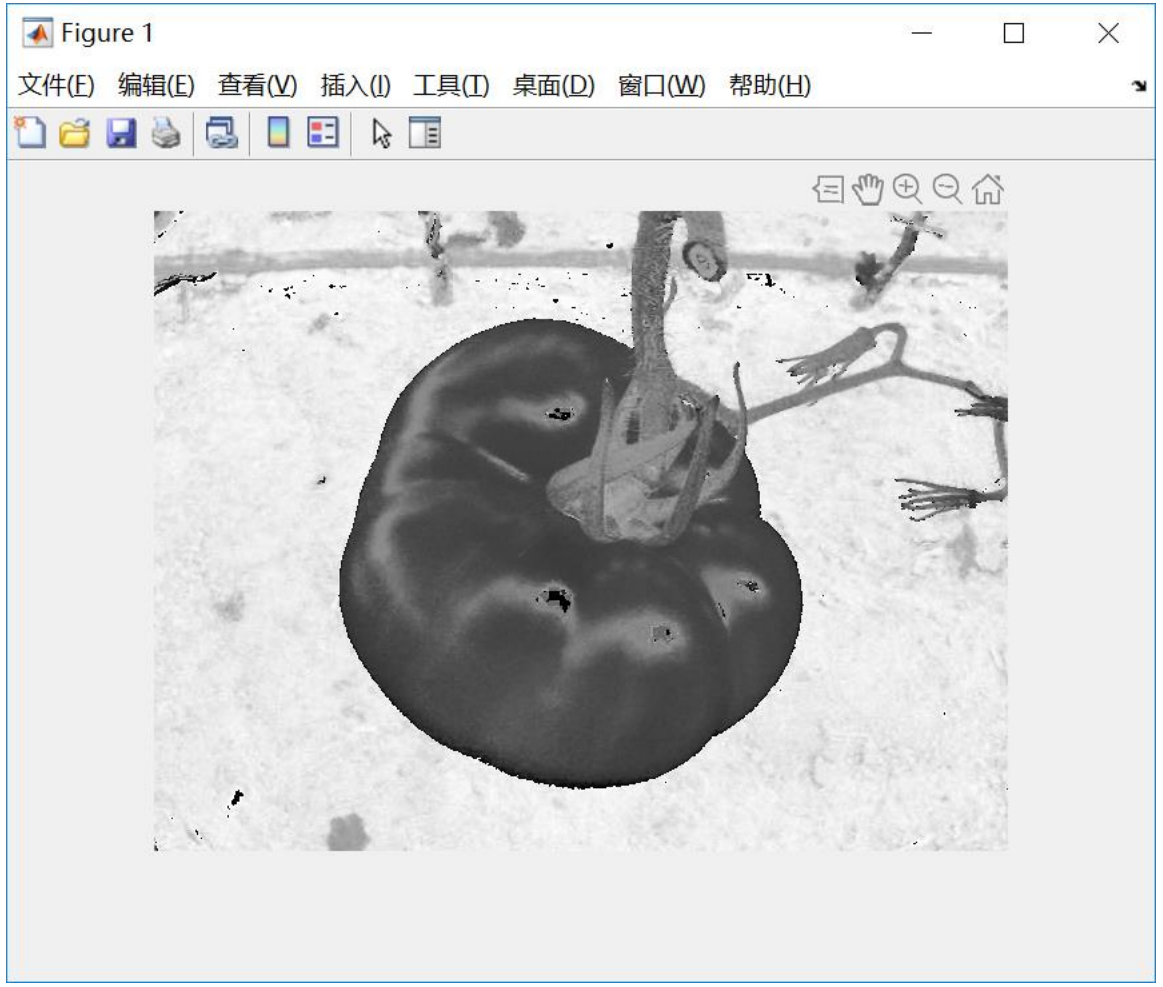
解决方案

- 进行腐蚀膨胀
- 转到HSV色彩空间



初始化

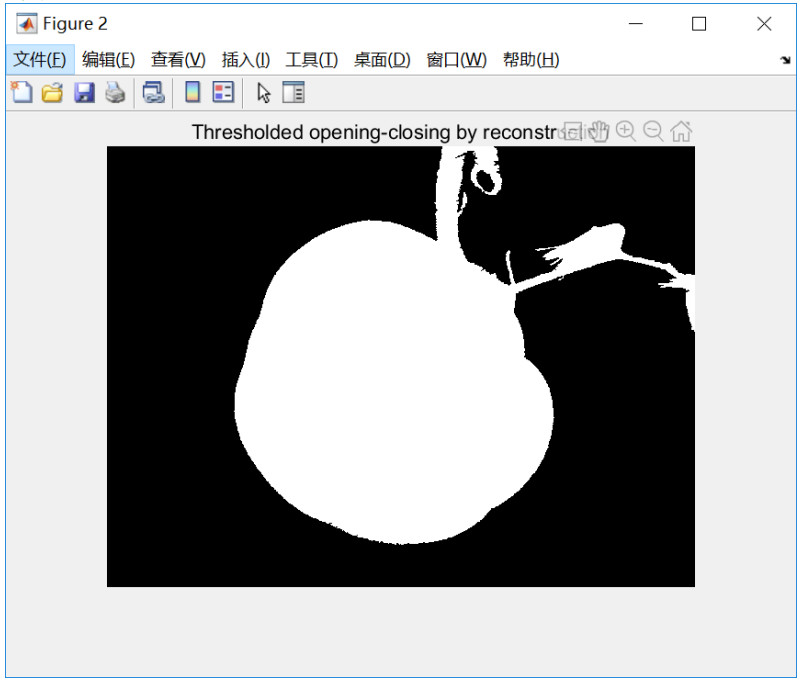
```
rgbimg = imread('test.jpg');
imshow(rgbimg);
r = rgbimg(:, :, 1);
g = rgbimg(:, :, 2);
b = rgbimg(:, :, 3);

grayimg = rgb2gray(rgbimg);
[row, col] = size(grayimg);%行数和列数

hsvimg = rgb2hsv(rgbimg);
H = hsvimg(:, :, 1);
S = hsvimg(:, :, 2);
V = hsvimg(:, :, 3);
```



取HSV中H的结果

阈值分割

```
% 分别对前景和背景进行标记：使用形态学重建技术对前景对象进行标记，
首先使用开操作，开操作之后可以去掉一些很小的目标。
se = strel('disk', 40);%圆形结构元素
% 腐蚀
Io = imopen(I, se);%形态学开操作
Ie = imerode(I, se);%对图像进行腐蚀
Iobr = imreconstruct(Ie, I);%形态学重建
% 膨胀
Ioc = imclose(Io, se);%形态学关操作
Iobrd = imdilate(Iobr, se);%对图像进行膨胀
Iobrcbr = imreconstruct(imcomplement(Iobrd),
imcomplement(Iobr));%形态学重建
Iobrcbr = imcomplement(Iobrcbr);%图像求反
bw = imbinarize(Iobrcbr, graythresh(Iobrcbr));%转化为二值图像
bw = imcomplement(bw);%图像求反
imshow(bw) %显示二值图像
title('Thresholded opening-closing by reconstruction')
```



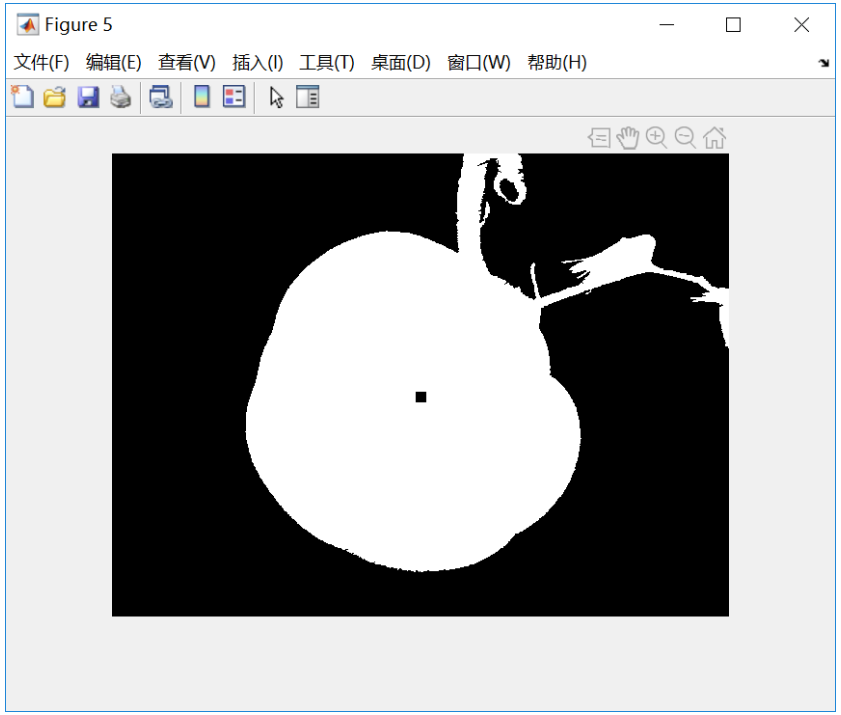
阈值分割的结果

ROI提取——求重心

```
XProject = sum(bw);%二值图在x轴的投影
YProject = sum(bw, 2);%二值图在y轴的投影
XProjectSum = XProject;
YProjectSum = YProject;
for i = 2 : col
    XProjectSum(i) = XProjectSum(i) + XProjectSum(i-1);
end
for i = 2 : row
    YProjectSum(i) = YProjectSum(i) + YProjectSum(i-1);
end
% 分别求x轴y轴重心
XHalf = sum(XProject)/2;%求x轴一半
YHalf = sum(YProject)/2;%求y轴一半
```

ROI提取——求重心

```
for i = 2 : col
    if XHalf > XProjectSum(i-1) && XHalf < XProjectSum(i)
        break;
    end
end
XCenter = i;%x轴中心
for i = 2 : row
    if YHalf > YProjectSum(i-1) && YHalf < YProjectSum(i)
        break;
    end
end
YCenter = i;%y轴中心
bwCenter = bw;
bwCenter(YCenter - 5 : YCenter + 5, XCenter - 5 :
XCenter + 5) = 0;
figure;
imshow(bwCenter);
% 求重心
```



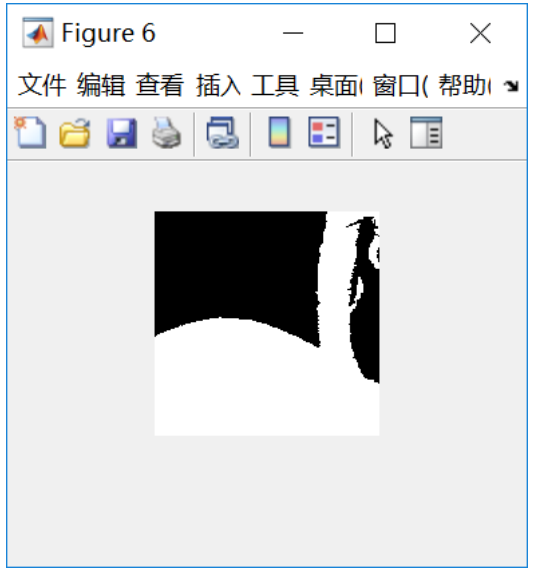
求重心结果

ROI提取——求半径

```
for i = XCenter : col
    if bw(YCenter, i) == 0
        break;
    end
end
RXR = i - XCenter;
for i = XCenter : -1 : 1
    if bw(YCenter, i) == 0
        break;
    end
end
RXL = XCenter - i;
R = (RXR + RXL) / 2;
```

ROI提取——ROI区域的确定

```
RecLDown = [floor(YCenter - R / 2), floor(XCenter - R / 2)];%矩形左下角的点为RecLDown
RecRDown = [floor(YCenter - R / 2), floor(XCenter + R / 2)];%矩形右下角的点为RecRDown
RecLUp = [floor(YCenter - R / 2 - R), floor(XCenter - R / 2)];%矩形左上角的点为RECLUp
RecRUp = [floor(YCenter - R / 2 - R), floor(XCenter + R / 2)];%矩形右上角的点为RecRUp
ROI = bw(RecLUp(1) : RecRDown(1), RecLDown(2) : RecRUp(2));
figure;
imshow(ROI);
```



切割点计算——对ROI的分块与投影

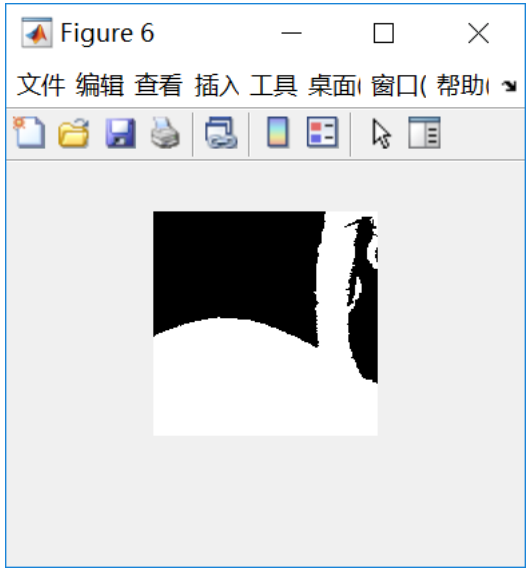
```
[ROIrow, ROIcol] = size(ROI);
ROIXProject = sum(ROI);%二值图在x轴的投影
ROIYProject = sum(ROI, 2);%二值图在y轴的投影
ROIXProjectSum = ROIXProject;
ROIYProjectSum = ROIYProject;
for i = 2 : ROIcol
    ROIXProjectSum(i) = ROIXProjectSum(i) +
ROIXProjectSum(i-1);
end
for i = 2 : ROIrow
    ROIYProjectSum(i) = ROIYProjectSum(i) +
ROIYProjectSum(i-1);
end
% 分块
XStep = floor(ROIcol / 10);%x轴每块的大小
YStep = floor(ROIrow / 10);%y轴每块的大小
```

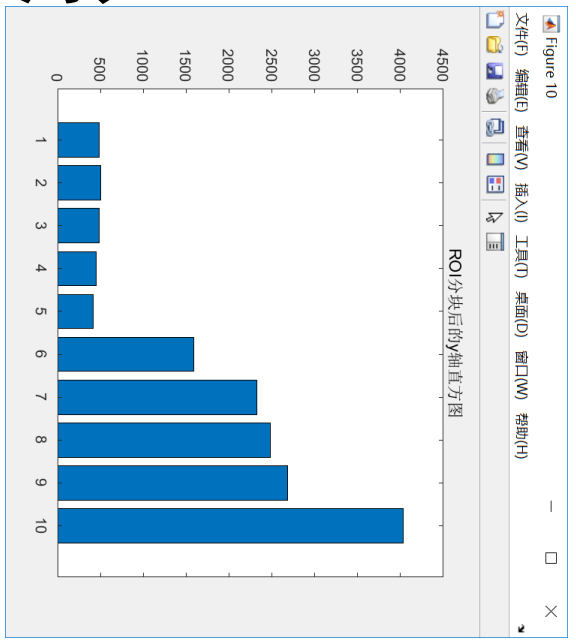
切割点计算——对ROI的分块与投影

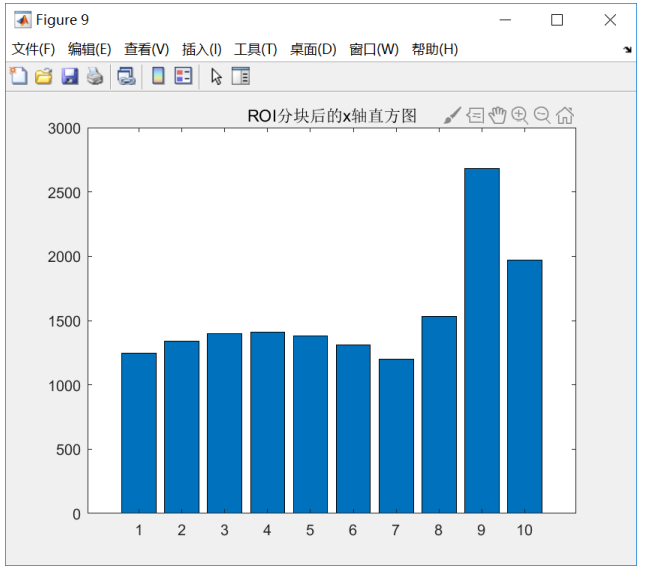
```
%初始化xy轴每一块的值
XBlock = zeros(1, 10);
YBlock = zeros(1, 10);
for i = 1 : 10
    if i == 1
        XBlock(1) = ROIXProjectSum(XStep * i);
        YBlock(1) = ROIYProjectSum(YStep * i);
        continue;
    end
    if i == 10
        XBlock(10) = ROIXProjectSum(ROIcol) - ROIXProjectSum(XStep * 9);
        YBlock(10) = ROIYProjectSum(ROIrow) - ROIYProjectSum(YStep * 9);
        break;
    end
    XBlock(i) = ROIXProjectSum(XStep * i) - ROIXProjectSum(XStep * (i - 1));%计算x轴每一块的值
    YBlock(i) = ROIYProjectSum(YStep * i) - ROIYProjectSum(YStep * (i - 1));%计算y轴每一块的值
end
```

切割点计算——对ROI的分块与投影

```
% 画图
ROIXAxis = 1 : 10;
ROIYAxis = 1 : 10;
figure;
bar(ROIXAxis, XBlock);
title('ROI分块后的x轴直方图');
figure;
bar(ROIYAxis, YBlock);
title('ROI分块后的y轴直方图');
```







切割点计算——切割点的最终确定

```
% 确定最终的切割点
[ROIXMax, ROIXPlace] = max(XBlock);%确定切割位置
的x轴坐标
i = 10;%初始化i
if YBlock(10) == ROIcol * (ROIrow - 9 * YStep)
    for i = 9 : -1 : 1
        if YBlock(i) ~= ROIcol * YStep
            break;
        end
    end
end
ROIYPlace = i - 1;
```

转换到原坐标下并作图

```
XPlaceL = (ROIXPlace - 1) * XStep + RecLDown(2) - 1;
XPlaceR = ROIXPlace * XStep + RecLDown(2) - 1;
YPlaceU = (ROIYPlace - 1) * YStep + RecLUp(1) - 1;
YPlaceD = ROIYPlace * YStep + RecLUp(1) - 1;
FinalImg = bw;
FinalImg(YPlaceU : YPlaceD, XPlaceL : XPlaceR) = 0;
figure;
imshow(FinalImg);
figure;
Finalrgbimg = rgbimg;
Finalrgbimg(YPlaceU : YPlaceD, XPlaceL : XPlaceR, 1) = 256;
Finalrgbimg(YPlaceU : YPlaceD, XPlaceL : XPlaceR, 2 : 3) = 1;
imshow(Finalrgbimg);
```

测试效果

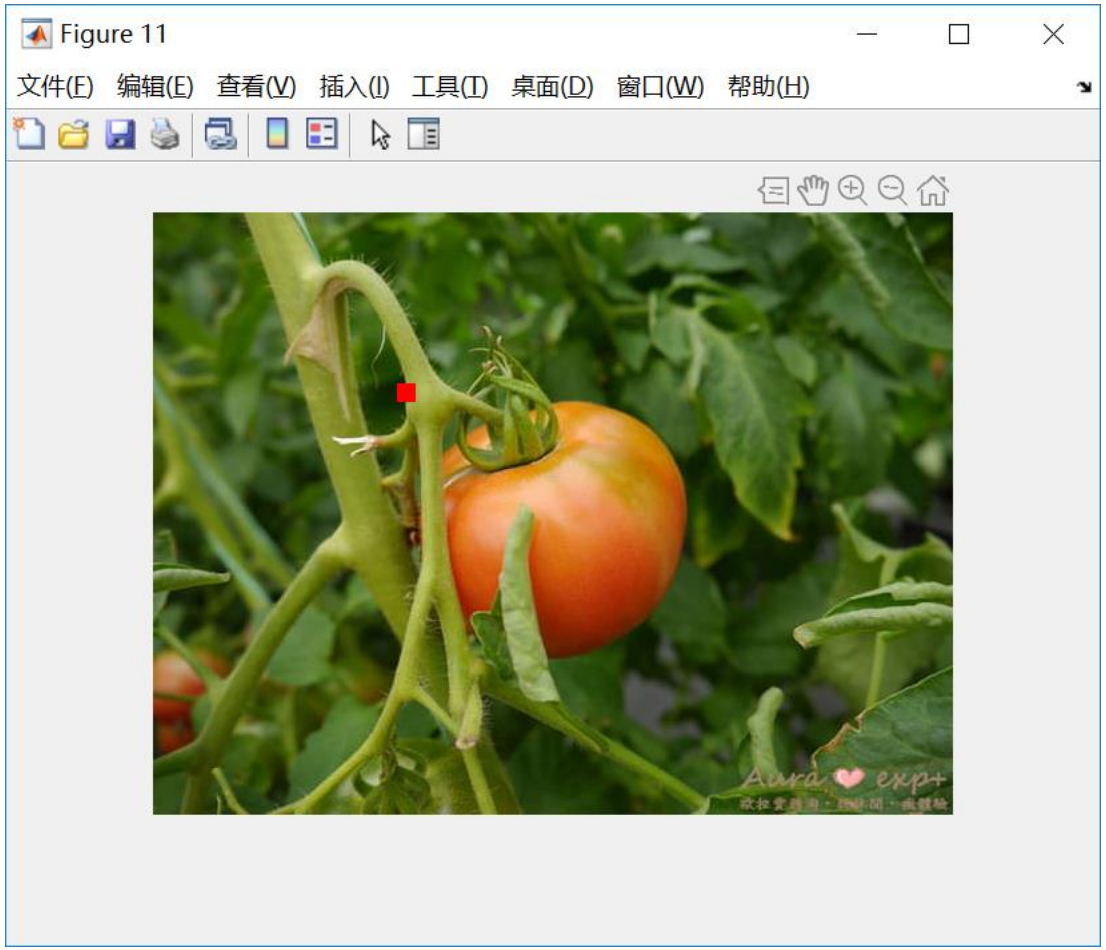


Figure 11
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)

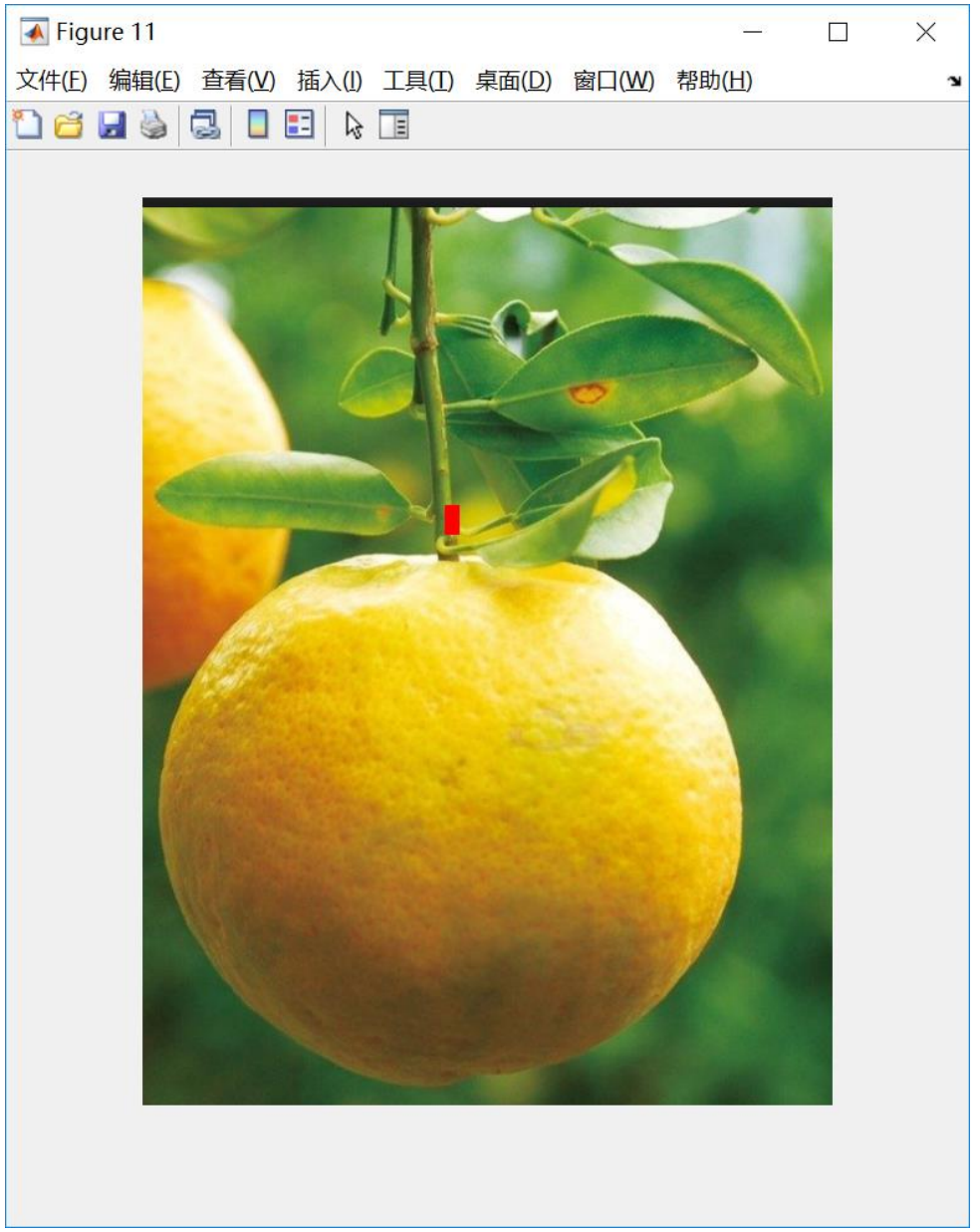


Figure 11
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)