

# 数字图像处理

## 第八周课堂练习

李竹

杭州电子科技大学

电子信息学院



# 讨论

## 1. 关于调参的思考。

合理的参数设置，应该是基于对需要解决的问题的一些已知条件。如需要提取的线段的长度范围，需要定位的工件的尺寸、大小、形状等。



# 讨论

## 1.算法

图像降噪，直方图增强，二值化，频率分析，图像形态学，几何信息提取，特征提取，等各种数学方法。

尽可能多的输出结果。



## 2.策略

筛选出实际需要的结果。

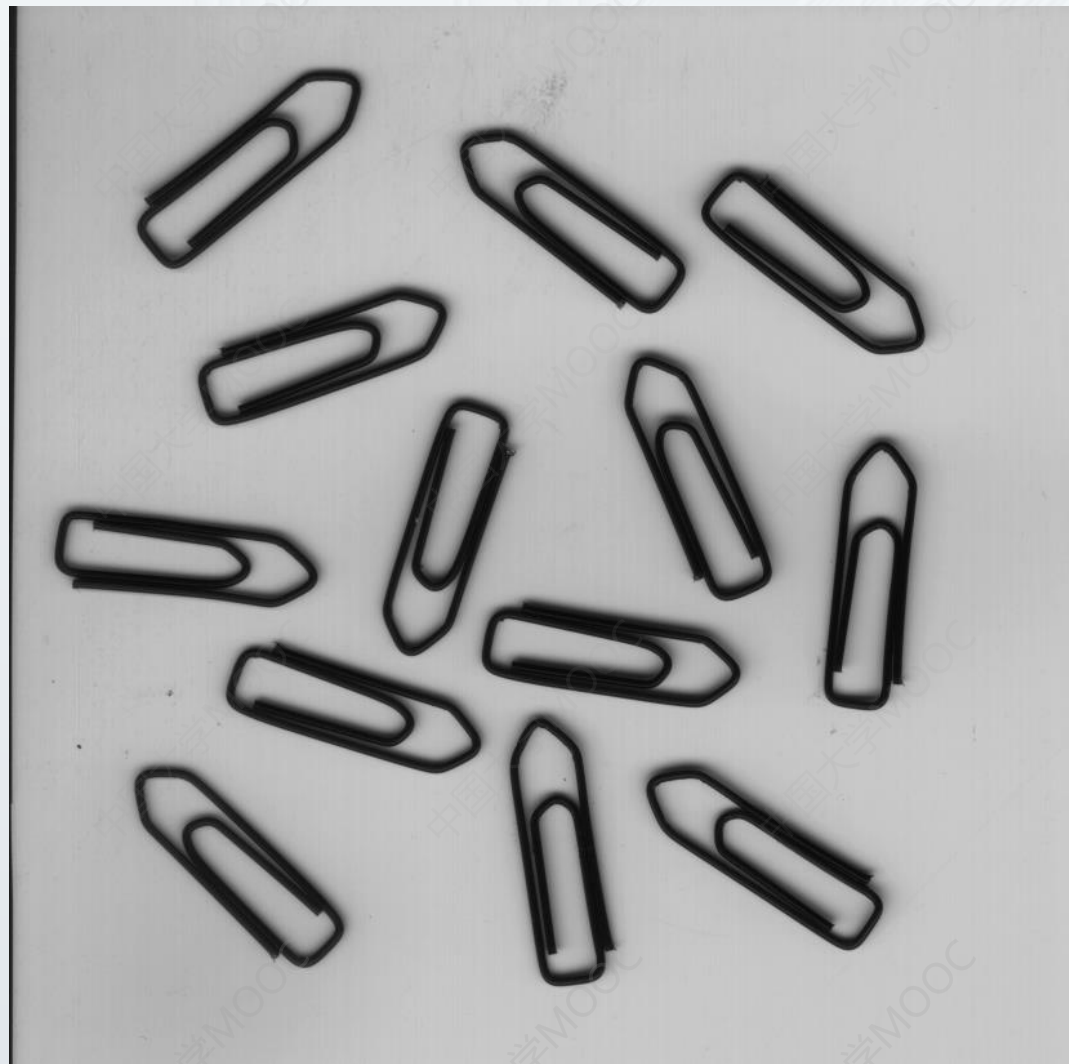
需要的信息和干扰信息的本质差距。



# 算法与策略

通过简单的二值化，连通域计数，会得到14个物体的结果。

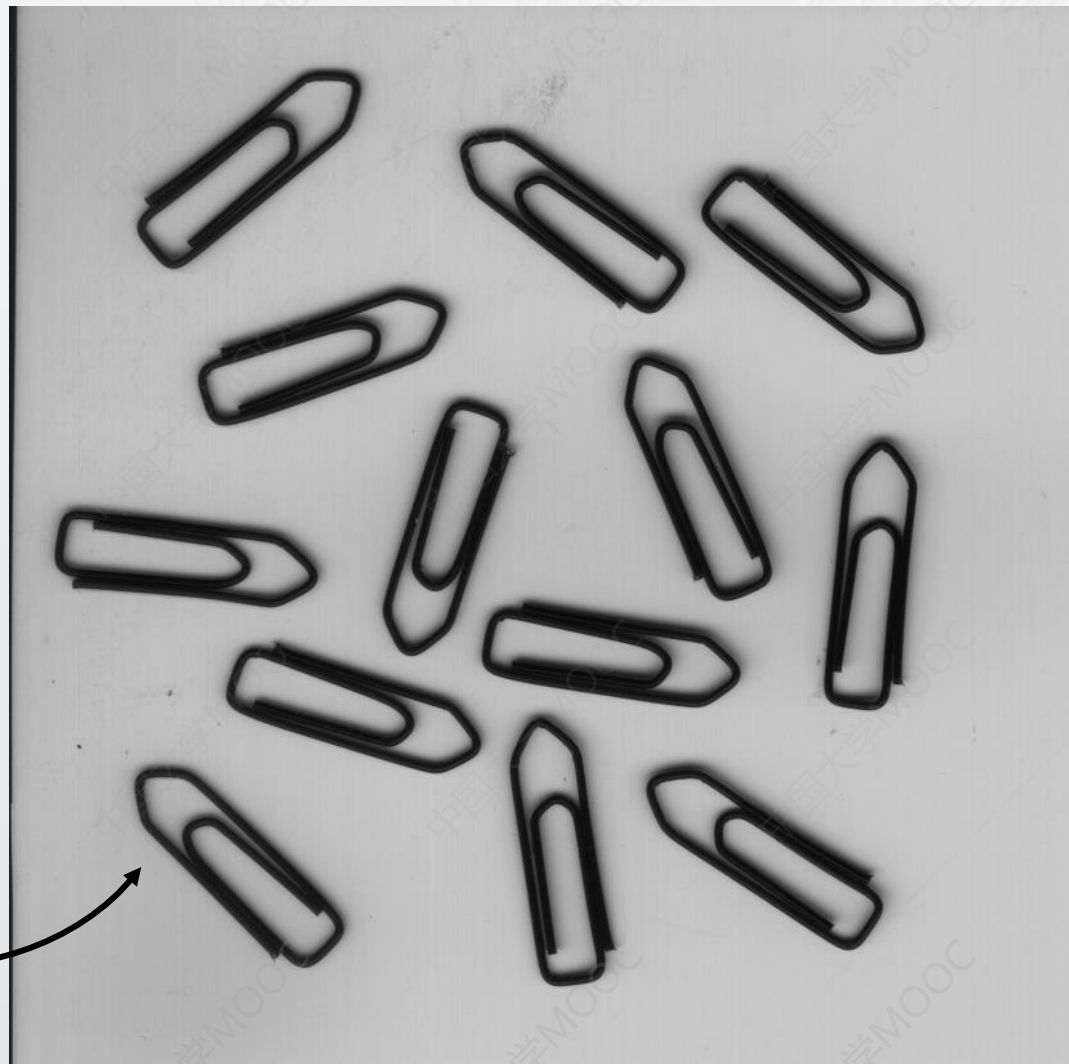
原因在于边缘有一个连通域对计数产生干扰。



# 算法与策略

通过简单的二值化，连通域计数，会得到14个物体的结果。

要将干扰物从目标物中排除，关键在于找出两者的区别，通过区别去筛选。

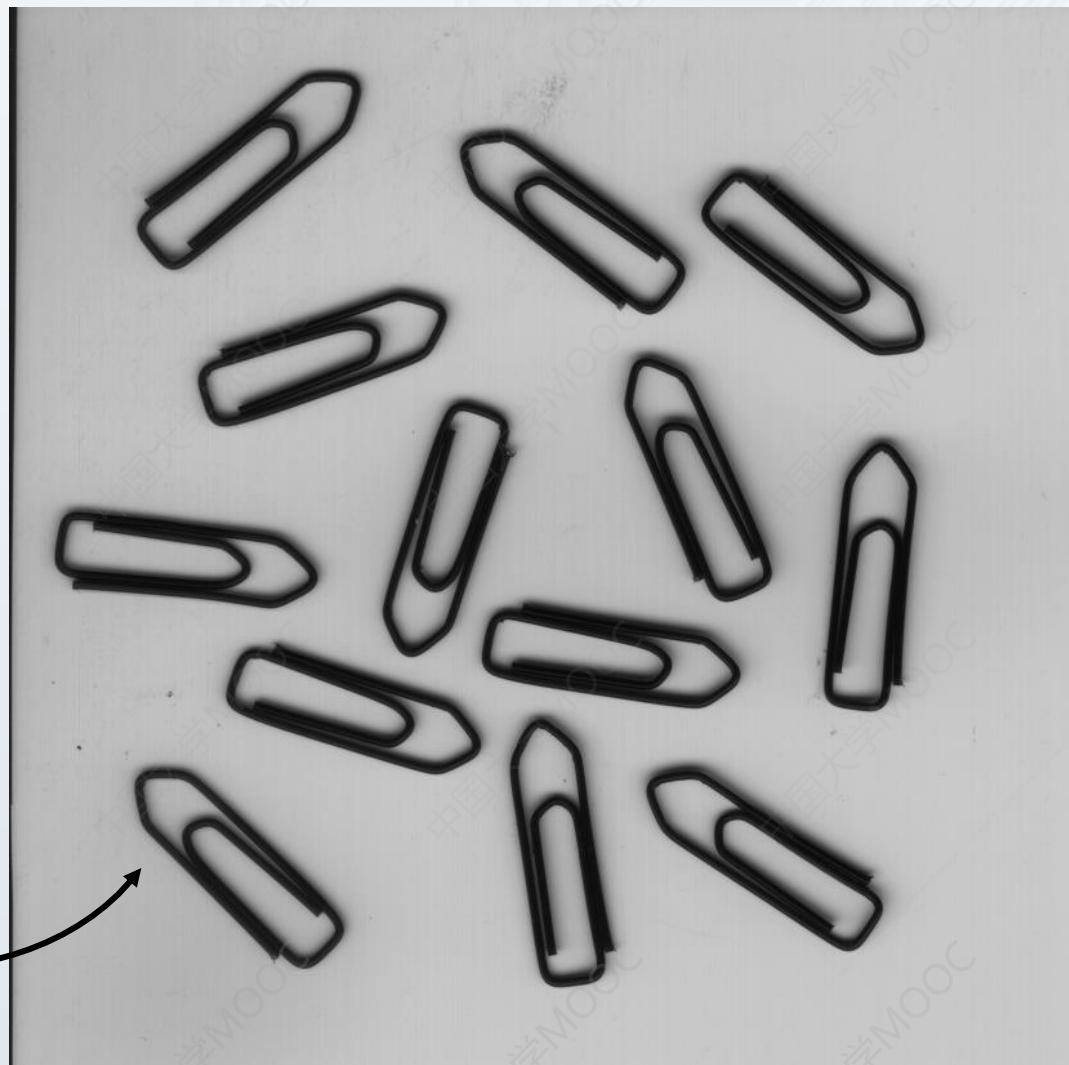


# 算法与策略

干扰物与目标物的高度和宽度  
存在较大区别。

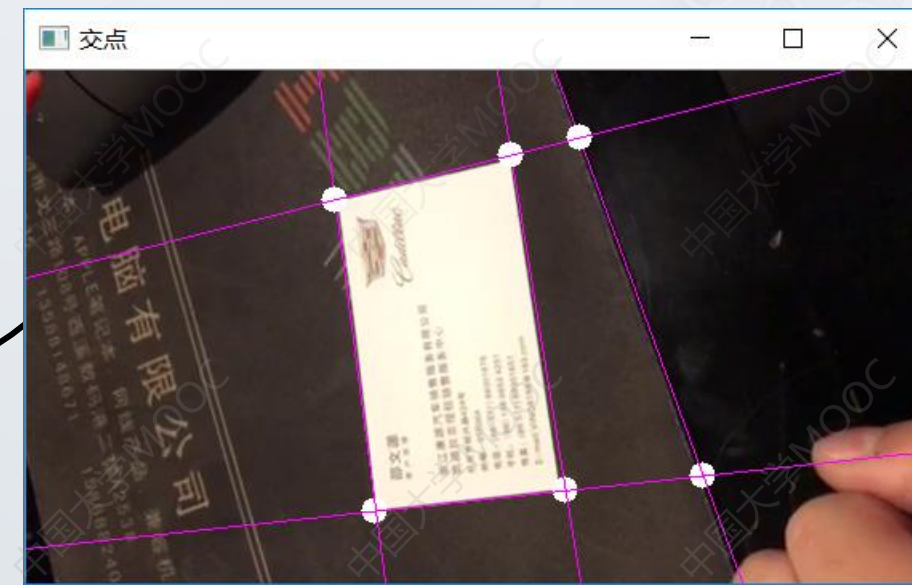
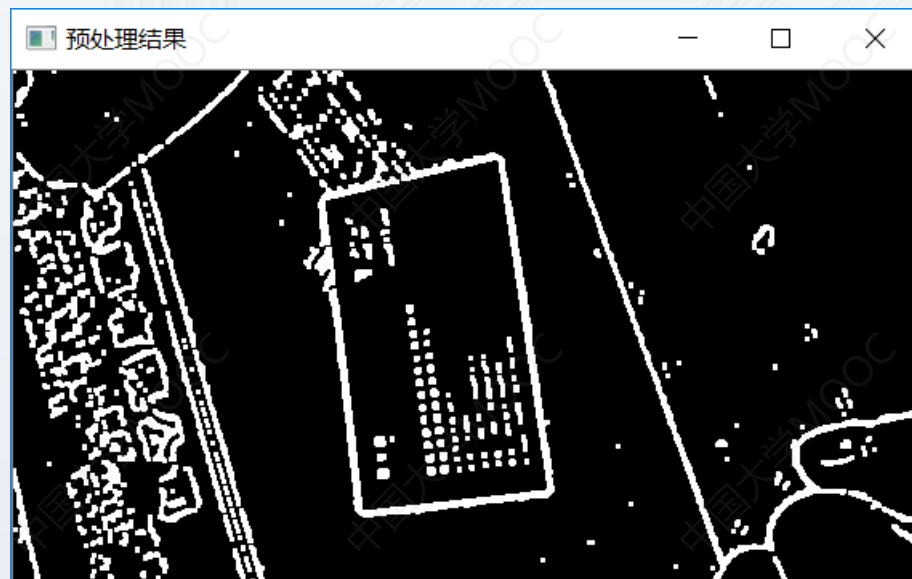
其他：

面积，周长，矩形度，圆形  
度。。。。。





# 算法与策略







# findContours

```
//通过findContours函数寻找连通域
vector<vector<Point>> contours;
vector<Vec4i> hierarchy;
findContours(bnyMat, contours, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_NONE);
```

参数4:

CV\_RETR\_EXTERNAL: 只检测最外围轮廓, 包含在外围轮廓内的内围轮廓被忽略;

CV\_RETR\_LIST: 检测所有的轮廓, 包括内围、外围轮廓, 但是检测到的轮廓不建立等级关系, 彼此之间独立, 没有等级关系, 这就意味着这个检索模式下不存在父轮廓或内嵌轮廓, 所以hierarchy向量内所有元素的第3、第4个分量都会被置为-1, 具体下文会讲到;

CV\_RETR\_CCOMP: 检测所有的轮廓, 但所有轮廓只建立两个等级关系, 外围为顶层, 若外围内的内围轮廓还包含了其他的轮廓信息, 则内围内的所有轮廓均归属于顶层;

CV\_RETR\_TREE: 检测所有轮廓, 所有轮廓建立一个等级树结构。外层轮廓包含内层轮廓, 内层轮廓还可以继续包含内嵌轮廓。

参数5:

CV\_CHAIN\_APPROX\_NONE: 保存物体边界上所有连续的轮廓点到contours向量内;

CV\_CHAIN\_APPROX\_SIMPLE: 仅保存轮廓的拐点信息, 把所有轮廓拐点处的点保存入contours向量内, 拐点与拐点之间直线段上的信息点不予保留;

CV\_CHAIN\_APPROX\_TC89\_L1: 使用teh-Chin1 chain 近似算法;

CV\_CHAIN\_APPROX\_TC89\_KCOS: 使用teh-Chin1 chain 近似算法。

# findContours

```
//通过findContours函数寻找连通域
vector<vector<Point>> contours;
vector<Vec4i> hierarchy;
findContours(bnyMat, contours, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_NONE);
```

参数4:

CV\_RETR\_EXTERNAL: 只检测最外围轮廓, 包含在外围轮廓内的内围轮廓被忽略;

CV\_RETR\_LIST: 检测所有的轮廓, 包括内围、外围轮廓, 但是检测到的轮廓不建立等级关系, 彼此之间独立, 没有等级关系, 这就意味着这个检索模式下不存在父轮廓或内嵌轮廓, 所以hierarchy向量内所有元素的第3、第4个分量都会被置为-1, 具体下文会讲到;

CV\_RETR\_CCOMP: 检测所有的轮廓, 但所有轮廓只建立两个等级关系, 外围为顶层, 若外围内的内围轮廓还包含了其他的轮廓信息, 则内围内的所有轮廓均归属于顶层;

CV\_RETR\_TREE: 检测所有轮廓, 所有轮廓建立一个等级树结构。外层轮廓包含内层轮廓, 内层轮廓还可以继续包含内嵌轮廓。

参数5:

CV\_CHAIN\_APPROX\_NONE: 保存物体边界上所有连续的轮廓点到contours向量内;

CV\_CHAIN\_APPROX\_SIMPLE: 仅保存轮廓的拐点信息, 把所有轮廓拐点处的点保存入contours向量内, 拐点与拐点之间直线段上的信息点不予保留;

CV\_CHAIN\_APPROX\_TC89\_L1: 使用teh-Chin1 chain 近似算法;

CV\_CHAIN\_APPROX\_TC89\_KCOS: 使用teh-Chin1 chain 近似算法。



# findContours

## 获得轮廓的最小外接四边形

```
//获得最小外界四边形
RotatedRect rbox = minAreaRect(contours[i]);

rbox.
```

- angle
- boundingRect
- boundingRect2f
- center
- points
- size

public : float cv::RotatedRect::angle  
! returns the rotation angle. When the angle is 0, 90, 180, 270 etc., the  
File: types.hpp

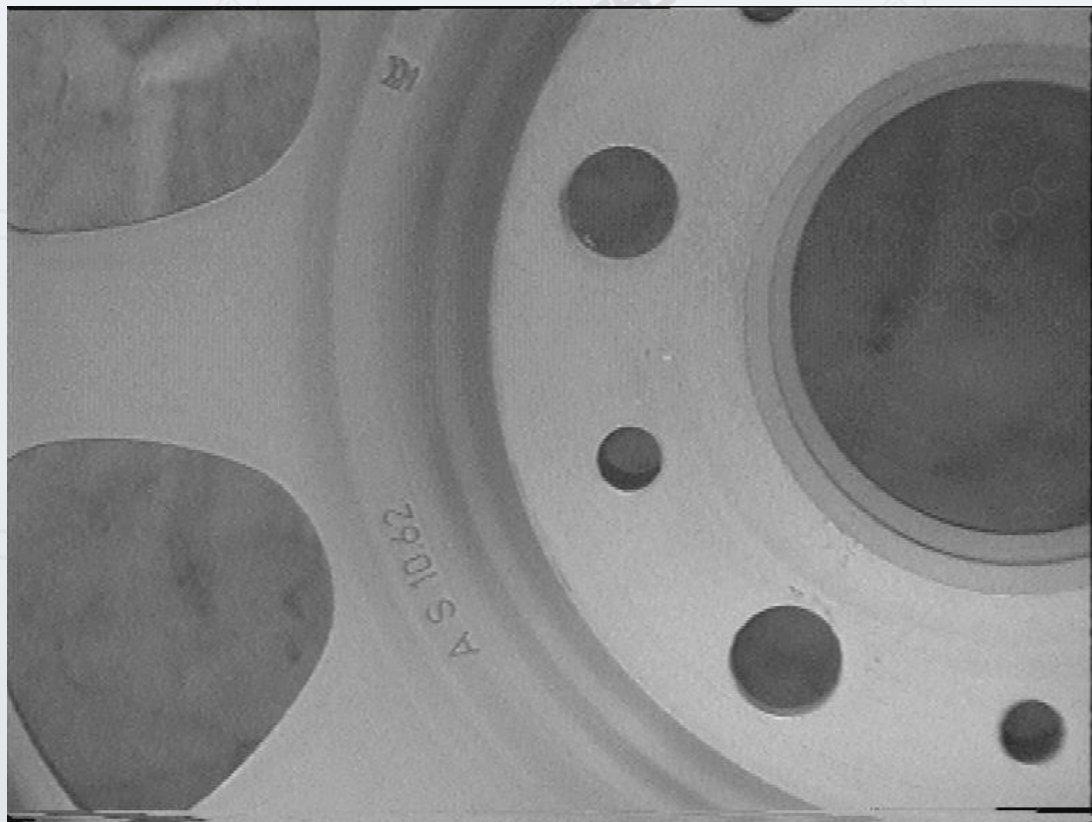
## 绘制轮廓及最小外接四边形

```
drawContours(disMat, contours, i, Scalar(0,255,255), 1, 8);
cv::Point2f vtx[4];
rbox.points(vtx);
for (int i = 0; i < 4; ++i) {
    cv::line(disMat, vtx[i], vtx[i<3 ? i + 1 : 0], cv::Scalar(0, 0, 255), 2, CV_AA);
}
```

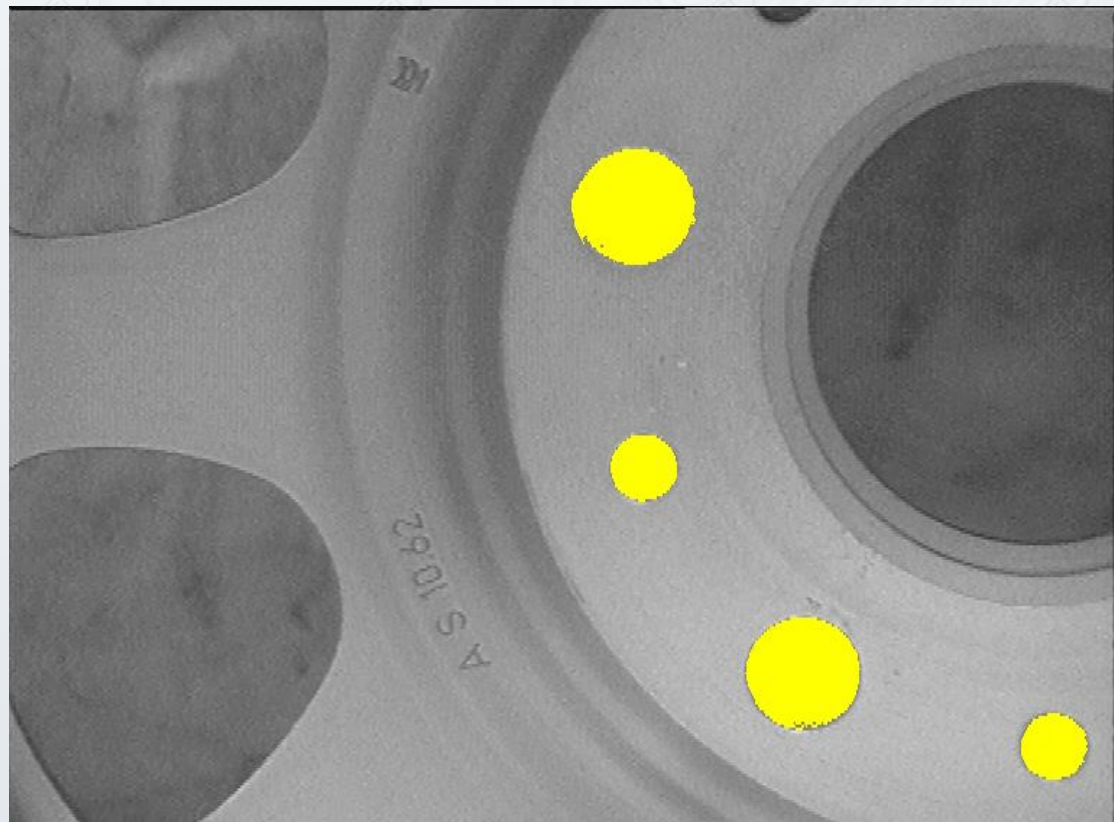


# 练习1

原图

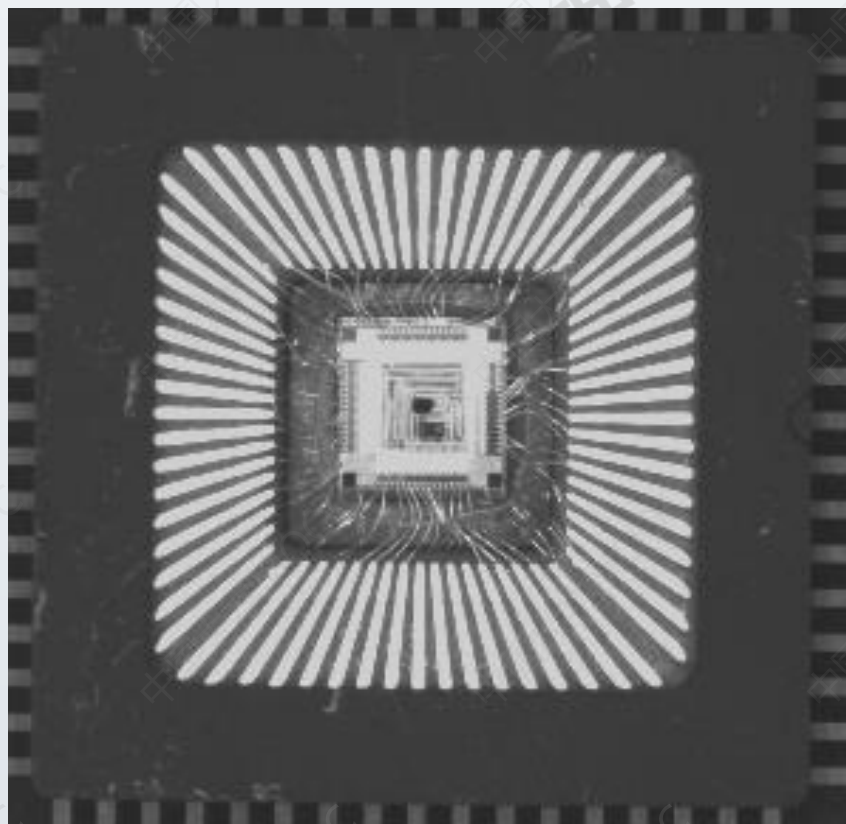


检测结果

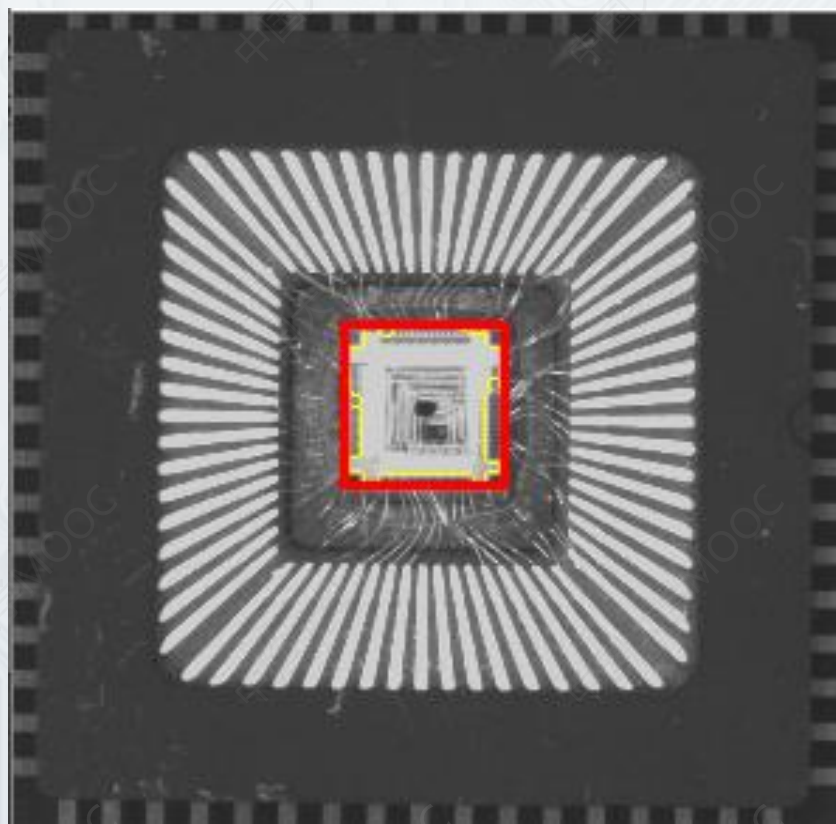


## 练习2

原图



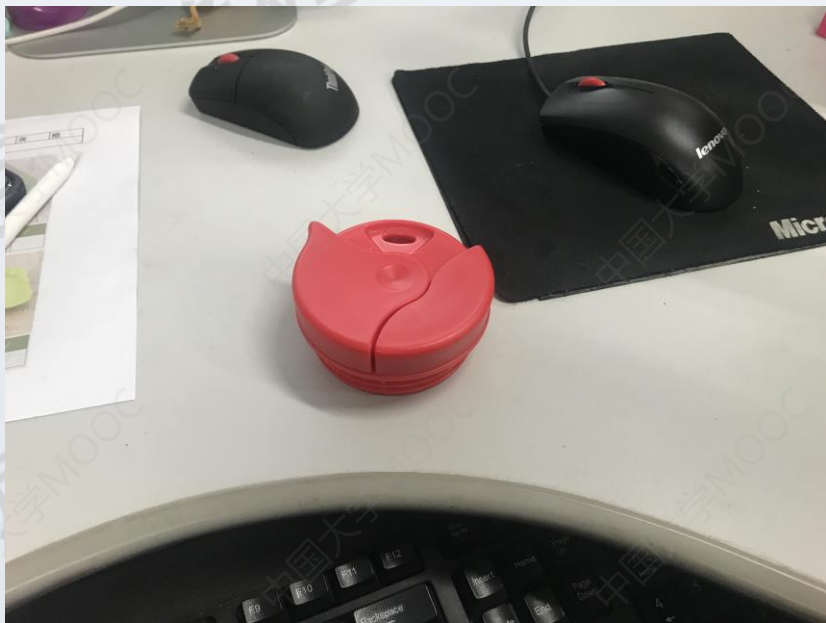
检测结果





# 练习3

原图



结果

