matlab 图像识别学习

郭豪鑫*
同济大学桥梁馆 802

版本：*1.0*
最后更新：*March 27, 2019*

此示例显示如何使用 imfindcircles 自动检测图像中的圆形或圆形对象。它还使用 viscircles 来显示检测到的圆圈。

1 自动检测图像中的圆形或圆形对象

1.1 加载图像

该示例使用带有各种颜色的圆形塑料片的图像。如图1所示。

图 1: 原图像

```
rgb = imread('DetectCirclesExample_01.png');%读入图片
imshow(rgb)%显示图片
```

除了有足够的圆形可以检测外，这个图像中对圆形的检测时还有一些有趣的事情：

1. 存在不同颜色的芯片，其与背景具有不同的对比度。一方面，蓝色和红色在这个背景上有强烈的对比。另一方面，一些黄色芯片与背景对比较差;

*matlab 官方网站

2. 可以发现一些芯片是相互叠加的，还有一些芯片是靠近在一起并且几乎相互接触的。重叠对象边界和对象的遮挡通常是对象检测的挑战。

1.2 确定搜索圈的半径范围

imfindcircles 函数需要一个半径范围来搜索圆圈。找到合适的半径范围的快速方法是使用交互式工具 imdistline 来获得各种对象半径的近似估计。如图2所示。

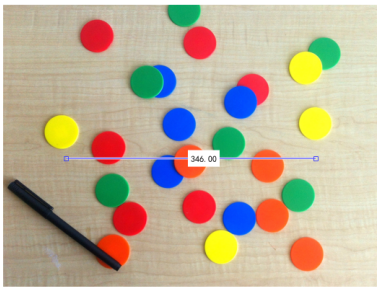

图 2: 图像

```
d = imdistline;%显示交互式工具，测量半径
```

imdistline 创建了一个可拖动的工具，可以移动以适应芯片，并且可以读取数字以获得其半径的近似估计。大多数芯片的半径范围为 20 像素。使用稍大的 20-25 像素的半径范围以确保可以检测到足够多的芯片。注意在进行下一步操作之前需要删除 imdistline 工具。

```
delete(d)%删除交互式工具
```

1.3 初步尝试查找圆形

在此图像上调用 imfindcircles，搜索半径为 [20 25] 像素。在此之前，最好查看物体是否比背景更亮或更暗。要查看图像的暗亮程度，可以调取此图像的灰度版本。

背景非常明亮，大多数芯片比背景更暗。但是，默认情况下，imfindcircles 会找到比背景更亮的圆形对象。因此，在 imfindcircles 中将参数'ObjectPolarity' 设置为'dark' 以搜索比背景暗的图像。

```
gray_image = rgb2gray(rgb);%图像的灰度版本
imshow(gray_image)%显示图像
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark');%查找圆形
```

得到对的结果如下：

```
centers =[]
radii =[]
```

请注意，输出中心位置和半径为空，这意味着没有找到圆圈。这种情况经常发生，因为 imfindcircles 是一个圆形探测器，与大多数探测器类似，imfindcircles 有一个内部探测阈值，决定了它的灵敏度。简单来说，这意味着检测器对某个（圆形）检测的置信度必须大于某个水平才能被认为是有效检测。

imfindcircles 有一个参数'Sensitivity'，可用于控制此内部阈值，从而控制算法的灵敏度。较高的“灵敏度”值会将检测阈值设置得较低，从而导致检测到更多圆圈。

1.4 提高检测灵敏度

回到圆形图像，有可能在默认的灵敏度水平下，所有圆都低于内部阈值，这就是没有检测到圆的原因。默认情况下，“灵敏度”（0 到 1 之间的数字）设置为 0.85。将“敏感度”提高到 0.9。

```
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.9)
```

这次 imfindcircles 发现了一些圆圈 - 准确地说是 5 个圆圈，如图3。中心包含圆心的位置，半径包含这些圆的估计半径。

1.5 在图像上绘制圆圈

函数 viscircles 可用于在图像上绘制圆圈。来自 imfindcircles 的输出变量中心和半径可以直接传递给 viscircles。

```
imshow(rgb)
h = viscircles(centers,radii);
```

图 3: 图像

圆心被正确定位，其相应的半径似乎与实际芯片很好地匹配。但仍有不少圆形被遗漏。尝试将“灵敏度”提高到 0.92。

```
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.92);
length(centers)
```

因此，增加“灵敏度”会让我们得到更多的圆形。再次在图像上绘制这些圆形，如图4。

```
delete(h)  % Delete previously drawn circles
h = viscircles(centers,radii);
```

图 4: 图像

1.6 使用第二种方法（两阶段）寻找圆圈

这个结果看起来更好。imfindcircles 有两种不同的方法来寻找圆圈。到目前为止，称为相位编码方法的默认方法用于检测圆。还有另一种方法，通常称为两阶段方法，可以在 imfindcircles 中使用。使用两阶段方法并显示结果。

```
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.92,'Method','twostage');\par
delete(h)
h = viscircles(centers,radii);
```

两阶段方法检测更多圆，灵敏度为 0.92。通常，这两种方法是互补的，因为它们具有不同的强度。与两阶段方法相比，相位编码方法通常更快并且对噪声更加鲁棒。但它也可能需要更高的“灵敏度”水平来获得与两阶段方法相同数量的检测。例如，如果将“灵敏度”级别提高到更高，相位编码方法也会找到相同的芯片，比如 0.95，结果如图5所示。

```
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.95);
delete(h)
viscircles(centers,radii);
```

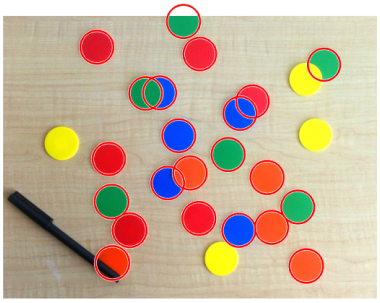

图 5: 图像

请注意，imfindcircles 中的两种方法都能准确地找到部分可见（遮挡）芯片的中心和半径。

1.7 为什么有些圈子仍然被错过？

看看最后的结果，很奇怪 imfindcircles 没有在图像中找到黄色芯片。黄色芯片与背景没有强烈的对比。事实上，它们似乎与背景具有非常相似的强度。是否有可能黄色芯片并不像假设的那样真正“背景”更暗？要确认，请再次显示此图像的灰度版本。

```
imshow(gray_image)
```

1.8 在图像中找到“明亮”的圆圈

与背景相比，黄色芯片的强度几乎相同，甚至更亮。因此，要检测黄色芯片，请将“ObjectPolarity”更改为“bright”。

```
[centersBright,radiiBright] = imfindcircles(rgb,[20 25], ...
'ObjectPolarity','bright','Sensitivity',0.92);
```

1.9 绘制不同颜色的“明亮”圆圈

通过更改 viscircles 中的“Color”参数，以不同的颜色绘制明亮的圆圈。

请注意，发现了三个缺失的黄色芯片，但仍然缺少一个黄色芯片，如图6所示。这些黄色芯片很难找到，因为它们在这种背景下并不像其他芯片那样突出。

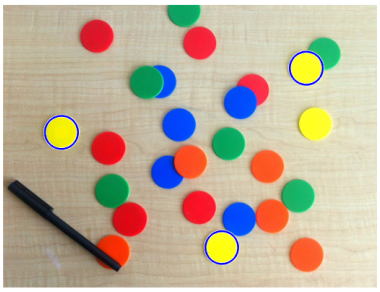

图 6: 图像

1.10 降低'EdgeThreshold' 的值

imfindcircles 中还有另一个参数可能在这里很有用，即'EdgeThreshold'。要查找圆圈，imfindcircles 仅使用图像中的边缘像素。这些边缘像素基本上是具有高梯度值的像素。

"EdgeThreshold" 参数控制像素的梯度值在被视为边缘像素并包含在计算中之前必须有多高。此参数的高值（接近 1）将仅允许包含强边（更高的梯度值），而低值（更接近 0）更宽松，甚至包括更弱的边（更低的梯度值）计算。在缺少黄色芯片的情况下，由于对比度低，所以预期一些边界像素（在芯片的圆周上）具有低梯度值。因此，降低 "EdgeThreshold" 参数以确保黄色芯片的大部分边缘像素都包含在计算中，如图7所示。

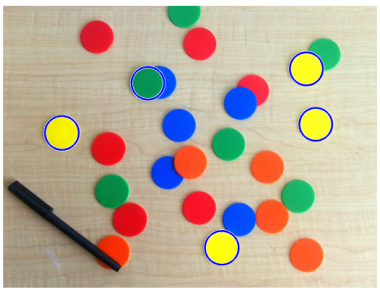

图 7: 图像

```
[centersBright,radiiBright,metricBright] = imfindcircles(rgb,[20 25], ...
'ObjectPolarity','bright','Sensitivity',0.92,'EdgeThreshold',0.1);
delete(hBright)
hBright = viscircles(centersBright, radiiBright,'Color','b');
```

1.11 一起绘制 "黑暗" 和 "明亮" 圆圈

```
h = viscircles(centers,radii);
```

现在，imfindcircles 找到所有黄色的，绿色的。用蓝色绘制这些圆形，以及之前发现的其他圆形（“ObjectPolarity”设置为“dark”），如图8。

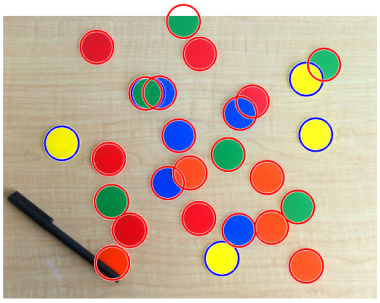

图 8: 图像

应该注意的是，将参数更改为在检测中更容易可能会发现更多的圆圈，但它也增加了检测假圆的可能性。可以找到的真实圆圈数（检测率）和与它们一起找到的假圆圈数量之间存在权衡（误报率）

2 代码

```
%% Step 1: Load Image
rgb = imread('DetectCirclesExample_01.png');
imshow(rgb)
%% Step 2: Determine Radius Range for Searching Circles
d = imdistline;
delete(d)
%%  Step 3: Initial Attempt to Find Circles
gray_image = rgb2gray(rgb);
imshow(gray_image)
[centers,radii] = imfindcircles(rgb,[35 43],'ObjectPolarity','dark');
%% Step 4: Increase Detection Sensitivity
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.9);
Step 5: Draw the Circles on the Image
imshow(rgb)
h = viscircles(centers,radii);
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.92);
length(centers)
delete(h)  % Delete previously drawn circles
```

```
h = viscircles(centers,radii);
%% Step 6: Use the Second Method (Two-stage) for Finding Circles
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.92,'Method','twostage');
delete(h)
h = viscircles(centers,radii);
[centers,radii] = imfindcircles(rgb,[20 25],'ObjectPolarity','dark', ...
'Sensitivity',0.95);
delete(h)
viscircles(centers,radii);
%% Step 7: Why are Some Circles Still Getting Missed?
imshow(gray_image)
%% Step 8: Find 'Bright' Circles in the Image
[centersBright,radiiBright] = imfindcircles(rgb,[20 25], ...
'ObjectPolarity','bright','Sensitivity',0.92);
%% Step 9: Draw 'Bright' Circles with Different Color
imshow(rgb)
hBright = viscircles(centersBright, radiiBright,'Color','b');
%% Step 10: Lower the Value of 'EdgeThreshold'
[centersBright,radiiBright,metricBright] = imfindcircles(rgb,[20 25], ...
'ObjectPolarity','bright','Sensitivity',0.92,'EdgeThreshold',0.1);
delete(hBright)
hBright = viscircles(centersBright, radiiBright,'Color','b');
%% Step 11: Draw 'Dark' and 'Bright' Circles Together
h = viscircles(centers,radii);
```