# 实验报告1

## 算法流程

霍夫圆变换是一种用于图像处理中检测圆形物体的技术，以下是算法的流程。

1. **图像压缩（非必要）**

首先，原始图像通过`compress_image`函数被等比缩放到指定的宽度（在这个例子中为400像素），保持不同图片在实验中的一致性。

2. **应用高斯模糊**

接着，使用`apply_gaussian_blur`函数对图像应用高斯模糊，这有助于减少图像的噪声和不必要的细节，使边缘检测更加准确。高斯模糊通过平滑图像来实现，其程度由核的大小和标准差（σ）控制。

3. **计算梯度**

`compute_gradients`函数被用于计算高斯模糊后图像的梯度幅值和方向。这一步是边缘检测的关键，梯度幅值表示边缘的强度，而梯度方向表示边缘的方向。

4. **非极大值抑制**

使用`non_maximum_suppression`函数对梯度幅值进行非极大值抑制处理。这个步骤的目的是仅保留局部最大梯度幅值的边缘，从而获得更细致、精确的边缘。

5. **双阈值处理**

`double_threshold`函数应用双阈值方法进一步细化边缘。通过设置高阈值和低阈值，该步骤区分强边缘和弱边缘，帮助去除噪声并最终确定真正的边缘。

6. **霍夫圆变换**

然后，使用`hough_circle_transform`函数在确定的边缘图像上执行霍夫圆变换，检测图像中的圆形。该函数搜索所有可能的圆心和半径，通过累加器空间识别出现次数最多的圆心和半径，作为检测到的圆。

7. **绘制检测到的圆（展示用）**

最后，`draw_detected_circles`函数将检测到的圆绘制到原始图像上。

## 函数功能说明

**1.**`compress_image(image, base_width=400)`

**功能**：该函数用于等比缩放图像至指定的宽度，保持图像原始的宽高比不变。

**参数**：

- `image`：Pillow图像对象，即将被缩放的原始图像。
- `base_width`：整数，目标宽度。默认值为100像素。

**返回**：返回一个新的Pillow图像对象，它是原始图像按照指定宽度等比缩放后的结果。

**2.** `apply_gaussian_blur(image, kernel_size=9, sigma=2.0)`

**功能**：对图像应用高斯模糊，以平滑图像并减少噪点。

**参数**：

- `image`：Pillow图像对象，原始图像。
- `kernel_size`：整数，高斯核的大小。默认值为9。
- `sigma`：浮点数，高斯核的标准差。默认值为2.0。

**返回**：返回一个新的Pillow图像对象，它是原始图像经过高斯模糊处理后的结果。该处理使用指定的核大小和σ值。

**3.** `compute_gradients(image)`

**功能**：计算图像的梯度幅值和方向。

**参数**：

- `image`：灰度图像的NumPy数组。

**返回**：

- `gradient_magnitude`：一个NumPy数组，包含了图像每一点的梯度幅值。
- `gradient_direction`：一个NumPy数组，包含了图像每一点的梯度方向。

**4.** `non_maximum_suppression(gradient_magnitude, gradient_direction)`

**功能**：应用非极大值抑制，以细化图像的边缘。

**参数**：

- `gradient_magnitude`：NumPy数组，包含图像的梯度幅值。
- `gradient_direction`：NumPy数组，包含图像的梯度方向。

**返回**：返回一个NumPy数组，它是应用了非极大值抑制算法后的图像。此图像仅保留了局部最大梯度幅值的边缘点，其余边缘点被抑制。

**5.** `double_threshold(nms_image, low_threshold, high_threshold)`

**功能**：进一步细化由非极大值抑制(`non_maximum_suppression`)步骤产生的边缘，通过设定两个阈值（一个低阈值和一个高阈值）来确定哪些边缘是真正的边缘。这个方法有助于减少错误边缘的检测，提高边缘检测的准确性。

**参数**：

- `nms_image`：经过非极大值抑制处理的图像，它是一个灰度图像，其中的像素值代表梯度幅值。
- `low_threshold`：低阈值，用于边缘检测。低于此阈值的边缘将被忽略，不认为是边缘。

- `high_threshold`：高阈值，用于确定强边缘。高于此阈值的边缘被认为是真正的边缘。

**6.** `hough_circle_transform(edges, radius_range, radius_step=1, angle_step=1, threshold=50, quantity=100, merge_distance=30)`

**功能**：该函数使用霍夫圆变换方法，在给定的边缘图像中检测圆形。它首先创建一个三维累积器空间，然后对于每个边缘像素和每个可能的圆半径以及圆心位置，通过投票机制来识别可能的圆形。

**参数**：

- `edges`：边缘检测后的图像矩阵，边缘应标记为255，非边缘为0或其他较低值。
- `radius_range`：一个元组，指定搜索圆形的最小和最大半径范围（R_min, R_max）。
- `radius_step`：半径的搜索步长，默认为1。
- `angle_step`：计算圆心时的角度步长（以度为单位），默认为1度。
- `threshold`：累加器的阈值，只有当圆的累加器值大于或等于此值时，才被认为是有效的圆，默认为50。
- `quantity`：最大返回圆形数量，默认为100。
- `merge_distance`：合并圆心接近的圆时使用的距离阈值，默认为30。

**返回**：返回一个圆形列表。每个圆形由一个元组表示，格式为`(y, x, r)`，其中`y`和`x`是圆心的行列坐标，`r`是圆的半径。这些圆形是根据函数参数和边缘图像通过霍夫圆变换检测到的。

**7.** `draw_detected_circles(image, circles, display_centers=True)`

**功能**：在图像上绘制检测到的圆及其圆心。该函数对于可视化霍夫圆变换的结果特别有用，可以直观地展示检测到的圆和它们的位置。

**参数**：

- `image`：Pillow图像对象，用作绘制圆形的背景图像。
- `circles`：检测到的圆的列表，每个元素为一个元组`(x, y, radius)`，其中`x`和`y`表示圆心的坐标，`radius`表示圆的半径。
- `display_centers`：一个布尔值，指定是否在图像上显示圆心。默认为True。

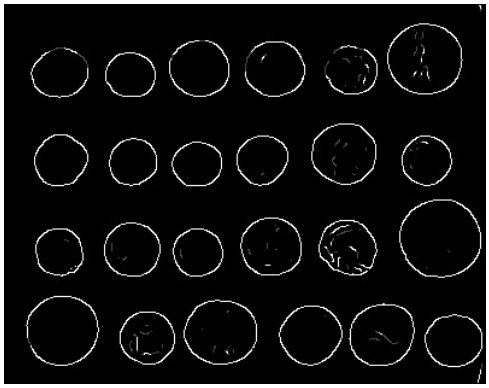
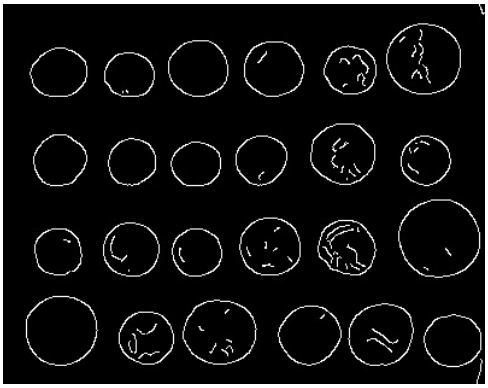
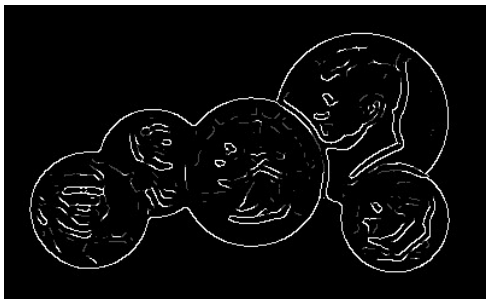
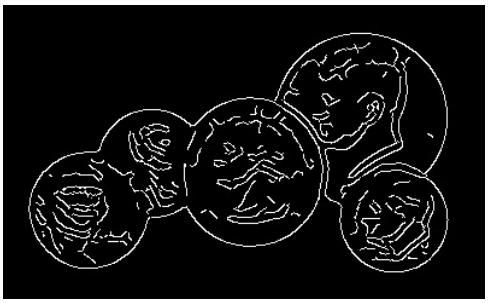
**返回**：返回一个新的Pillow图像对象，它在原始图像的基础上绘制了检测到的圆及其圆心（如果`display_centers`为True）。

# 实验结果

## 1.边缘检测（和cv2（右）对比）

分析两张图像可以看到，在参数一致的条件下，自定义的函数和cv2的canny边缘检测结果相似。

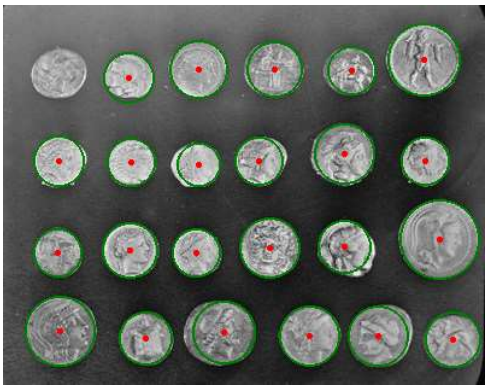
由于在自定义的函数中加入了双重阈值的处理，所以有些边的颜色比较淡，在后续圆形检测时候低阈值部分将提供更少的votes（0.5），这样子更加容易找到圆而不会被内部或者外部的弱边缘影响。

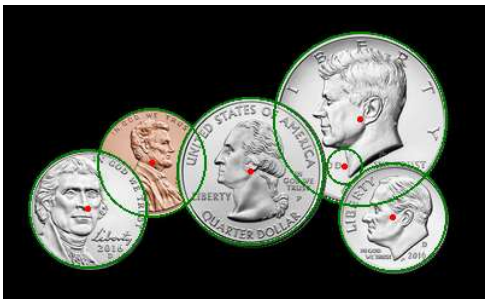
## 2.圆形检测

可以看到除了左上角的硬币都能够成功检测到。（和cv2的结果一致）

在实验中，经过几次参数的调整，左上角的圆也能检测出来，但是在进行其他图像的实验室，该参数的泛用性太差，所以这里还是展示此结果。

全部的圆都被检测出来了，但是可以看到多了几个内部的圆形



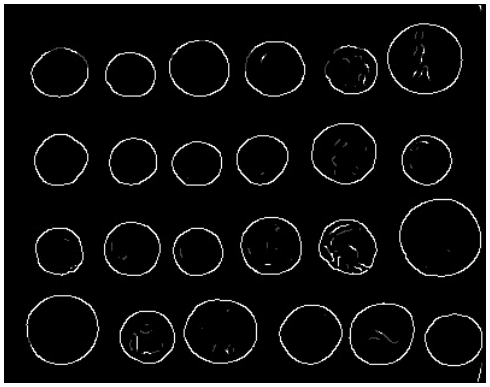
## 3.不同参数对于结果的影响

**高斯模糊：**

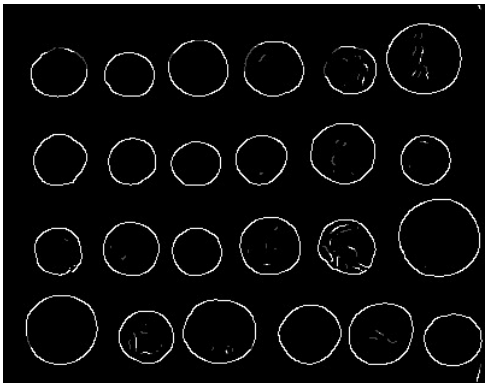
1. **核大小（`kernel_size`）**

**影响**：核大小决定了高斯滤波器的尺寸。较大的核会涵盖更多的邻域像素，导致更强的模糊效果。这有助于去除图像的噪声，但也可能导致边缘模糊，减少细节。但是实验中发现影响不大。

左图为9*9（sigma=2.0）



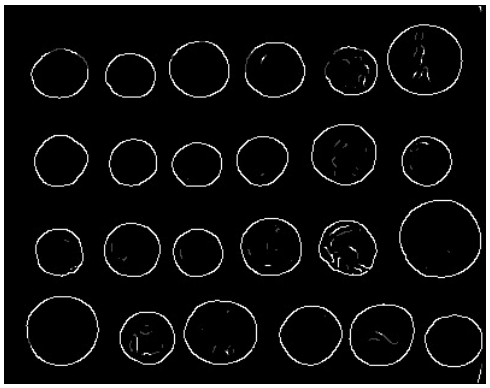
左图为45*45（sigma=2.0）



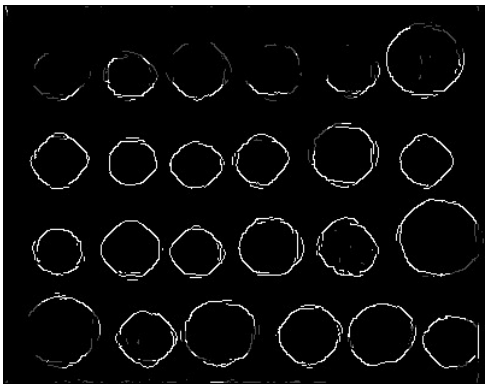
2. **标准差（`sigma`）**

**影响**：σ值决定了高斯函数的宽度，进而影响模糊的程度。较高的σ值意味着更宽的高斯函数，导致更强的模糊效果，但是也很容易让边缘难以分辨。

左图为2.0（kernel_size=9）



左图为6.0（kernel_size=9）

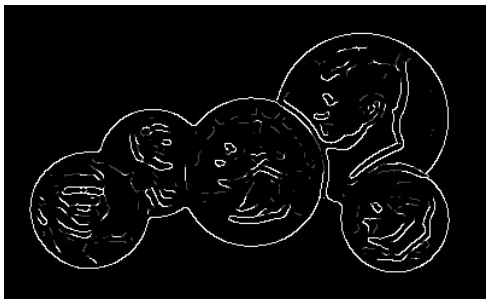


**边缘检测：**

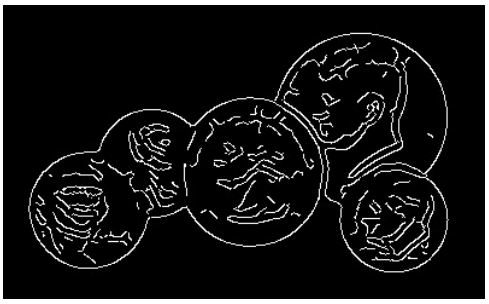
1.**边缘阈值（**`threshold`**）**

**影响**：决定了判断是否为边缘的阈值，更大的阈值对非边缘的排除更好，更小的阈值对边缘的采样更充分。

左图为90



左图为70

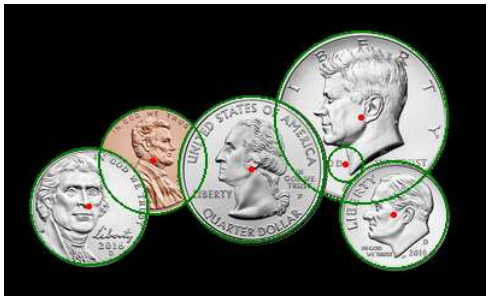


**圆形检测：**

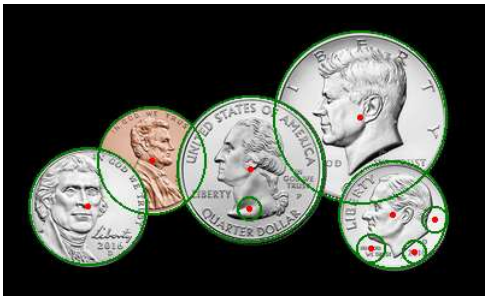
1. **半径范围（**`radius_range`**）**

**影响**：半径范围决定了算法搜索圆形的大小范围。如果半径范围设置得过窄，可能会错过一些圆形；如果设置得过宽，则会增加计算复杂度，且可能检测到错误的圆形。

左图为（20，100）



右图为（10，100）



2. **累加器阈值（**`threshold`**）**

**影响**：累加器阈值决定了一个位置被认定为圆心需要的最低票数。较高的阈值可以减少假阳性（即错误地检测到的圆形），但也可能错过一些真正的圆形。在实操中进一步在merge前先进行了圆形数量的限制（最多100个圆），导致这个阈值不是唯一决定的参数，对结果影响有限。

3. **合并距离（**`merge_distance`**）**

**影响**：合并距离决定了多少个检测到的圆形会被认为是相同的圆并合并。较小的距离可能导致多个相近的圆形被检测为不同的圆，而较大的距离可能将不同的圆合并为一个。由于有重叠的部分，所以建议此参数尽可能调低，避免把重叠的硬币合为一个。这里建议把合并距离设置为图中最短半径大小（需要自己实验得到最短半径）

**结论：**

**在对不同图片进行实验的过程中，发现参数必须根据数据的不同而进行调整，实际应用中，可能需要通过多次试验来调整这些阈值，以获得最佳的边缘检测效果。**