Mini-Project Title: Circular Object Detection in Images

Aim: To develop an image processing algorithm that can accurately detect and locate circular objects within a given image.

Apparatus Required:

- A computer with MATLAB or Octave installed
- An image containing circular objects

Theory:

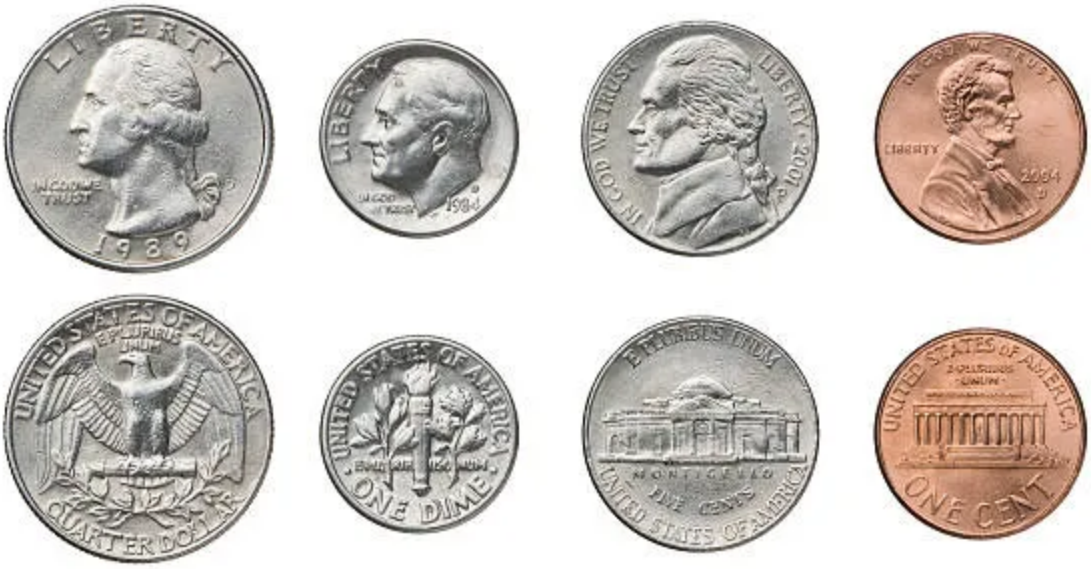
1. **Image Reading and Conversion:**
    - **imread:** Reads the image from the specified file path.
    - **rgb2gray:** Converts the image from RGB color space to grayscale.
2. **Adaptive Thresholding:**
    - **adaptthresh:** Applies adaptive thresholding to enhance contrast and segment the image into foreground and background regions.
    - **imbinarize:** Converts the grayscale image into a binary image based on the threshold.
3. **Morphological Operations:**
    - **strel:** Creates a structuring element (a disk in this case) for morphological operations.
    - **imopen:** Removes small objects and noise from the binary image.
    - **imclose:** Fills small holes and gaps in the objects.
4. **Hough Circle Transform:**
    - **imfindcircles:** Detects circles in the binary image using the Hough Circle Transform.
    - **centers:** Stores the x and y coordinates of the detected circle centers.
    - **radii:** Stores the radii of the detected circles.
5. **Visualization:**
    - **imshow:** Displays the original image.
    - **viscircles:** Overlays the detected circles on the image.
    - **plot:** Marks the center of each circle with a red dot.

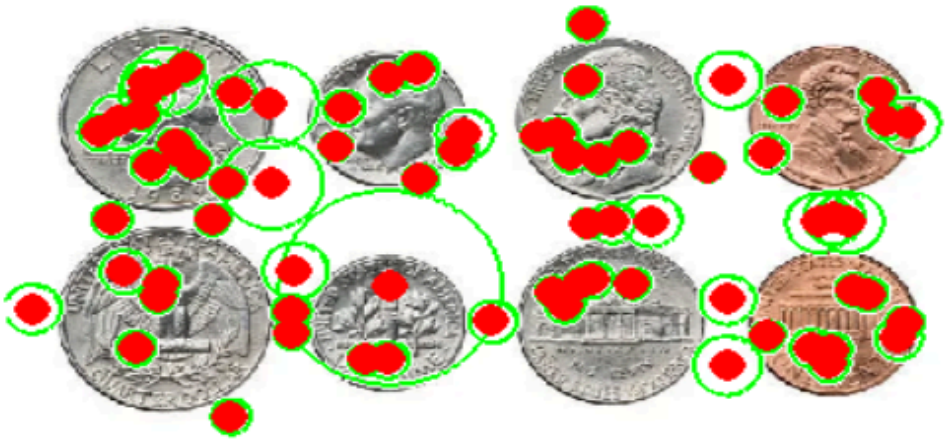
Program:-

```
% Read the image

image = imread('uuuuu.png');  % Replace with your image path

% Convert the image to grayscale

grayImage = rgb2gray(image);

% Apply Gaussian Blur to reduce noise and improve detection

blurredImage = imgaussfilt(grayImage, 2);

% Detect circles using the Hough Circle Transform

[centers, radii] = imfindcircles(blurredImage, [10 100], 'ObjectPolarity',
'bright', 'Sensitivity', 0.9);

% Display the original image

imshow(image);

hold on;

% Plot the detected circles

viscircles(centers, radii, 'EdgeColor', 'g', 'LineWidth', 2);

% Mark the center of the circles with a red dot

plot(centers(:,1), centers(:,2), 'ro', 'MarkerFaceColor', 'r');

hold off;

% Optionally, save the output image with circles drawn

saveas(gcf, 'output_image.jpg');
```

Applications: This technique can be applied to various real-world scenarios, including:

- **Industrial Inspection:** Detecting defects in circular objects.
- **Medical Image Analysis:** Analyzing circular structures in medical images (e.g., cells, tumors).
- **Autonomous Vehicle Navigation:** Detecting traffic signs and other circular objects.
- **Object Tracking:** Tracking circular objects in videos.


LIBERTY
1989
LIBERTY
1984
IN GOD WE TRUST
LIBERTY
2001
IN GOD WE TRUST
LIBERTY
2004
UNITED STATES OF AMERICA
E PLURIBUS UNUM
QUARTER DOLLAR
UNITED STATES OF AMERICA
ONE DIME
E PLURIBUS UNUM
MONTICELLO
FIVE CENTS
UNITED STATES OF AMERICA
UNITED STATES OF AMERICA
ONE CENT



Results: Upon executing the code, the following output will be generated:

- The original image will be displayed.
- Detected circles will be outlined in green.
- The center of each circle will be marked with a red dot.