# Homework 11: Identifying Circles

## Purpose:

To be able to identify circles using Hough transform.

## Tasks:

- **Problem:**

  Modify the code provided to identify circles in an image. Apply this to the "L06 sunflower.png Download L06 sunflower.png" or any other image of your choice. If you choose another image, please make sure you correctly cite it. For an implementation in MATLAB, check https://www.mathworks.com/help/images/ref/imfindcircles.htmlLinks to an external site.

## Answer →

To find circles within an image, *imfindcircles* function from MATLAB can be used. This function converts TrueColor images to grayscale using the function *rgb2gray* before processing them. This function also converts Binary (logical) and integer type images like the ones created out of edge function (like Canny edge-detection) into the data type single using the *im2single* function before processing. To improve the accuracy of the result for binary images, *imfindcircles* also applies Gaussian smoothing using *imfilter* as a preprocessing step.

Reference – https://www.mathworks.com/help/images/ref/imfindcircles.html#bvit6hx

We have processed the image L06 sunflower.png through the imfindcircles using multiple image processing combinations to estimate which option provides the best determination of circles in the image.

Combinations used:

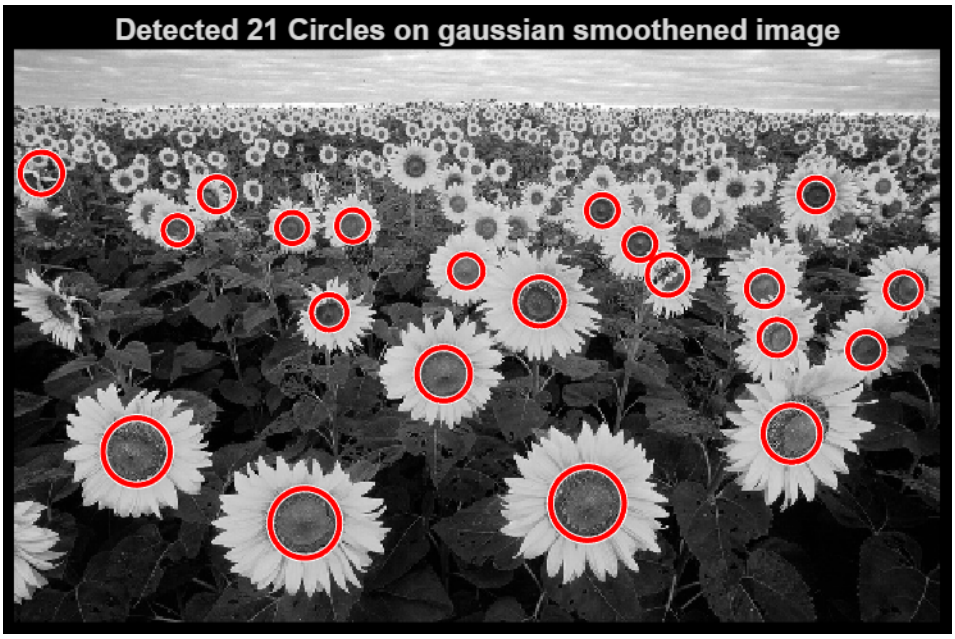
1. Detecting circles on original RGB colored image
2. Detecting circles on grayscale image
3. Detecting circles on Gaussian smoothened image (removing noise)
4. Detecting circles on Canny edge-detected image
5. Detecting circles on Gaussian smoothened image run through Canny edge- detector

The results are as follows:

1. Detecting circles on original RGB colored image

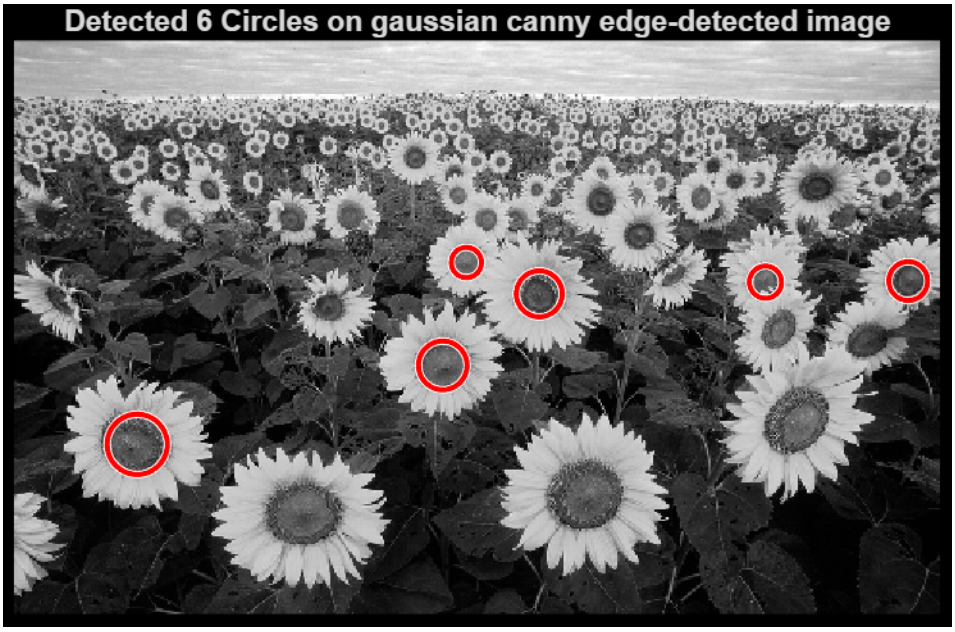

2. Detecting circles on grayscale image

3. Detecting circles on gaussian smoothened image




4. Detecting circles on canny edge-detected image

5. Detecting circles on gaussian canny edge-detected image




Observations:

1. Circle detection on original and grayscale images doesn't give us any results even though the imfindcircles function documentation mentions it carries out grayscale conversion before detecting circles. The possibility of poor results could be due to the presence of noise in the image.
2. Circle detection on gaussian smoothened image gives us the best results (21 circles). This shows that once the noise is suppressed in an image, the circle detection through imfindcircles function works great. There are some false positives within the result, but the recall rate is high.
3. Circle detection on canny edge detected image does give us lesser number of circles (6) as compared to gaussian smoothened image but there is a false positive within the results as well.
4. Circle detection on gaussian smoothened canny edge detected image does gives us better results in terms of recall rate even though the number of circles detected is the same (6) as the non-gaussian smoothened one.
5. Overall, we can say that while the imfindcircles documentation mentions that they run the image through gaussian filter as a processing step, running the image through Gaussian filter as a standalone function rather than relying on imfindcircles doing it, is a better approach.

MATLAB code:

```
%% Circular Hough Transform
clear; clc; close all;
I = imread('L06 sunflower.png'); % reading original image
I_gray = im2gray(I); % converting to grayscale image for edge detection
I_gauss = imgaussfilt(I_gray, 2); % using gaussian filter to remove noise
I_canny = edge(I_gray, 'canny'); % edge detection on grayscale image
I_gauss_canny = edge(I_gauss, 'canny'); % edge detection on smoothened image
% Display circles using original Image as input
hough_CHT(I, 'original', I);
% Display circles using grayscale Image as input
hough_CHT(I_gray, 'grayscale', I_gray);
% Display circles using guassian filtered Image as input
hough_CHT(I_gauss, 'gaussian smoothened', I_gray);
% Display circles using Canny edge-detected Image as input
hough_CHT(I_canny, 'canny edge-detected', I_gray);
% Display circles using Canny edge-detected Gaussian filtered Image as input
hough_CHT(I_gauss_canny, 'gaussian canny edge-detected', I_gray);
function hough_CHT(img, type, disp_img)
	figure;
	imshow(disp_img); hold on;
	% finding circles and their centers using imfindcircles
	[centers, radii] = imfindcircles(img, [15 50], 'ObjectPolarity', 'dark',
	'Sensitivity', 0.85);
	viscircles(centers, radii, 'EdgeColor', 'r');
	num = length(centers);
	title(sprintf(['Detected ', num2str(num),' Circles on ', type, ' image']));
	saveas(gcf, [type,'.png']); hold off;
end
```