# Canny edge detection

MD Facihul Azam
Department of Information Technology

## Abstract:
In this project the widely used edge detection algorithm canny edge detector is implemented and the experiment was carried out by using different image both color and gray scale with different parameter values. Also a graphical user interface is created where user can select the parameter such as higher threshold, lower threshold for hysteresis edge linking and sigma Gaussian derivative.

## Introduction:
Edge characterizes the boundaries of an object. Edges of an image are the strong contrast that is a


jump from one pixel to next pixel. Edge detecting is to reduce the low frequency information while preserving the important structural properties. Canny Edge detection algorithm is widely used in image processing to detect edge. Edge is a useful feature in computer vision to recognize the shape of the object and also is useful to understand the structure of the object in an image. The general criteria for edge detection includes following statements:
"Detection of edge with low error rate, which means detection should catch as many edges as possible. The edge point detected from the operator should accurately localize on the centre of the edge. A given edge in the image should only be marked once, and where possible and image noise should not create false edges"[1]. Based on these criteria canny algorithm first smooths the image with Gaussian filter to remove noise. It then finds the image gradient to highlight the region with high intensity. The algorithm then suppress the pixel that is not maximum using nom maximum suppression. After non maxima suppression the algorithm set double thresholding and edge linking by hysteresis to determine the edges.

## Methodology and Results:
The project work consist of a graphical user interface (figure1) where user can load picture after that three parameter can be set to detect edge such as higher threshold, lower threshold and sigma. User then needs to click the edge button to execute the canny algorithm. By clicking the clear axes button user can refresh the axes. The system is consist of four matlab functions:
**check_image.m:** It checks the input image whether it is gray or color image. If it is gray image function returns the image. If the image is color image function converts the image into gray scale using *rgb2gray* matlab routine and returns the gray image.
**gaussian_dervative.m:** returns the first derivative Gaussian filter in x and y direction. The kernel size is 10x10.
**edge.m:** executes canny algorithm by calling above functions.
**Canny_edge_detector.m:** For graphical user interface.
Canny edge detection is carried out by four consecutive steps. These steps are described bellow with output images:

step1. Smoothing the image with Gaussian function using matlab routine *conv2 and gaussian_derivative call.*

$$f_x = \frac{\partial}{\partial x}(f * G) = f * \frac{\partial}{\partial x} G = f * G_x$$

$$f_y = \frac{\partial}{\partial y}(f * G) = f * \frac{\partial}{\partial y} G = f * G_y$$

$G(x, y)$ is the Gaussian function

$G_x(x, y)$ is the derivate of $G(x, y)$ with respect to $x$: $G_x(x, y) = \frac{-x}{\sigma^2} G(x, y)$

$G_y(x, y)$ is the derivate of $G(x, y)$ with respect to $y$: $G_y(x, y) = \frac{-y}{\sigma^2} G(x, y)$

step2. Once it finds the Gaussian derivative of X and Y direction namely Gx and Gy, it then

calculates gradient magnitude

$$magn(i,j) = \sqrt{f_x{}^2 + f_y{}^2}$$

figure 2 shows step 1 and step 2 output image.

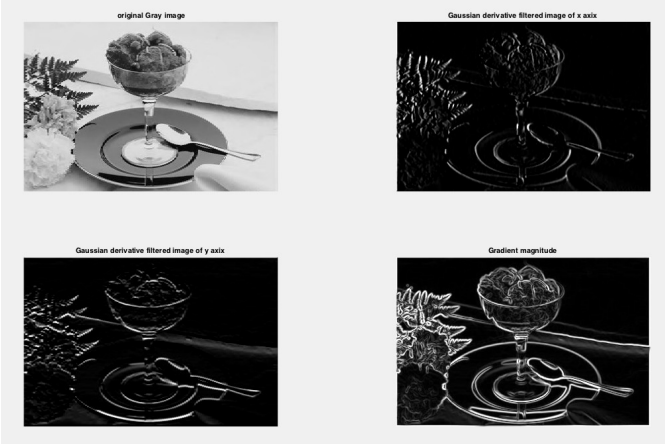


figure2

step3: Finds the local maxima of the gradient magnitude. All values along the direction of the gradient that are not peak values of a ridge are suppressed.

For each pixel (x,y) do:

if $magn(i,j) < magn(i_1,j_1)$ or $magn(i,j) < magn(i_2,j_2)$
then $I_N(i,j) = 0$
else $I_N(i,j) = magn(i,j)$

step4: non-maxima suppression still holds some noise. To remove noise and preserve potential edge two threshold image are created I1(i,j) and I2(i,j).

1. Produce two thresholded images $I_1(i,j)$ and $I_2(i,j)$.

(note: since $I_2(i,j)$ was formed with a high threshold, it will contain fewer false edges but there might be gaps in the contours)

2. Link the edges in $I_2(i,j)$ into contours

2.1 Look in $I_1(i,j)$ when a gap is found.

2.2 By examining the 8 neighbors in $I_1(i,j)$, gather edge points from $I_1(i,j)$ until the gap has been bridged to an edge in $I_2(i,j)$.

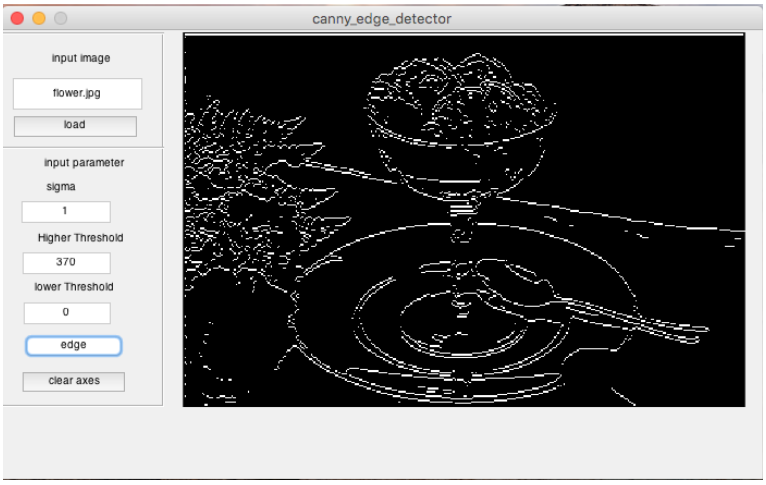


figure3

Different value of sigma creates different edge detection. Below shows four different edge detection for different sigma value.

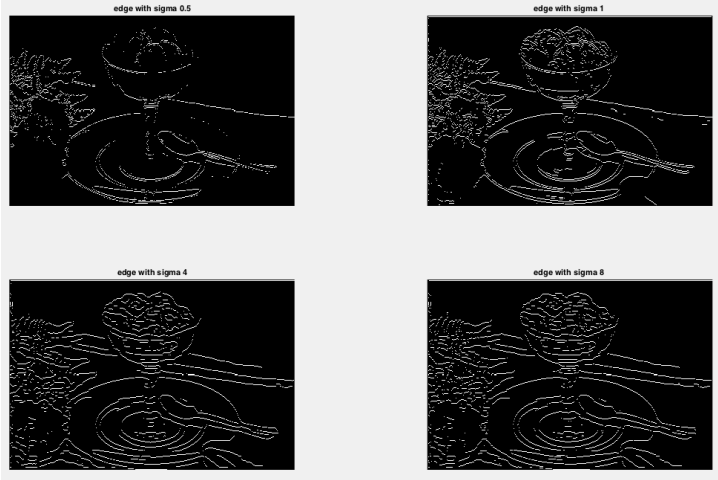


figure4

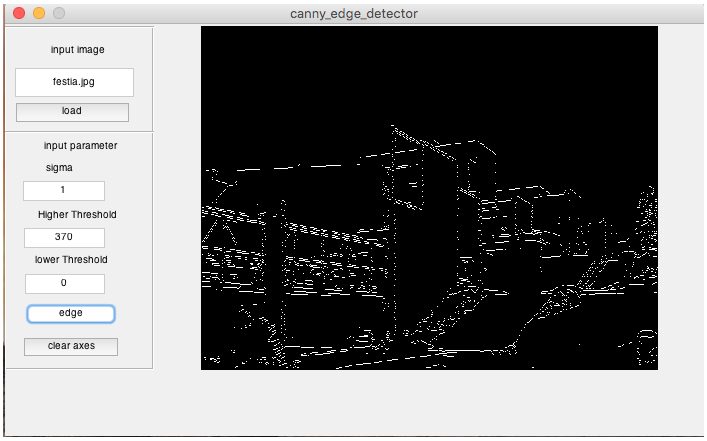


figure 5

## Conclusion:

From the experiment it shows that the parameter setting is subject dependent. The optimum threshold and sigma creates potential edge detection. On the above picture from figure4, lower value of sigma cannot detect some edge links, on the other hand higher sigma detects more edges with some false detection. In this case figure3, lower threshold 0 and higher threshold 370 with sigma 1 gives better result. Some other picture was tested on this system and it is found that lena.jpg gray scale picture also detects good result with the same parameter but festia.jpg color picture detects only side walls and windows, stare case detects as background picture, figure 5. So it proves that parameter setting depends on the picture which is detected.

## Reference:

1. https://en.wikipedia.org/wiki/Canny_edge_detector
2. Matlab routine used un this project work:
rgb2gray
interp2
conv2