EDGE DETECTION

EE18B042 and EE18B039
Department of Electrical Engineering
Indian Institute of Technology, Tirupati

Abstract **- Our aim in this project is to detect edges in the given image as edge detection is used in various fields such as image segmentation and data extraction in areas such as image processing, computer vision and machine vision. Generally, objects in any image has more intensity than the background on the edges. this makes it easier for a computer to identify or track an object once its edge is detected. These many uses of edge detection motivated us for this project and gave us urge to implement it in MATLAB. We have analysed various methods for edge detection and found out that Canny edge detection method is the efficient one to detect edges with more accuracy. We have implemented canny method in MATLAB without using any inbuilt functions. This project will help us in our further study about image processing and computer vision.**

Keywords – canny, edge, edge detection, non-maximum suppression, hysteresis thresholding, image processing.

Introduction

Edge detection is an essential step in image analysis. It is used to detect the boundaries/edges in the image. It works by detecting discontinuities in image brightness. The points at which image brightness changes sharply are organized into a set of curved line segments called edges. Classical methods of edge detection engage convolving the image through an operator, which is constructed to be perceptive to large gradients in the image although returning values of zero in uniform regions. A lot of edge detection techniques are available such as Sobel, Canny, Prewitt, Roberts etc. Different operators that can be used in edge detection can be optimized to find vertical, horizontal and diagonal edges.

Method

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. This method is applied to grayscale image. Grayscale image is one in which the only colours are shades of grey. Grayscale intensity is stored as an 8-bit integer giving 256 possible shades of grey from black (0) to white (255).

The algorithm runs in separate steps:

1. Converting image to grayscale
2. Smoothing (Gaussian blur)
3. Applying Sobel filters to detect edges
4. Non-maximum suppression
5. Double thresholding
6. Edge hysteresis

Our approach for the different steps and implementation of the same in MATLAB:

Step #1 Converting image in grayscale:

If image is not in grayscale, then our first step is to convert it to grayscale and this can be done by using following equation in MATLAB:

$$I = 0.299*II(:,:,1) + 0.587*II(:,:,2) + 0.114*II(:,:,3)$$

This equation adds different weights of all three-color channels red, green and blue.

Step #2 Smoothing:

We used the Gaussian blur kernel of size 3*3 and convolved this kernel with grayscale image(2d).

The kernel that we have used is

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

This kernel helped us to smoothen the image and remove noise from it.

Step #3 Sobel operator:

It is a basic algorithm used to calculate gradient of intensity for a grayscale image. It emphasizes edges of image. It uses kernels (3*3 matrix), which is convolved with the whole image to get the gradient magnitude for each pixel. And after applying both these filters we obtain two images one containing horizontal edges and other containing vertical edges. We take magnitude of both these images using: **MAGNITUDE = sqrt (H.*H + V.*V);**

-1	-2	-1
0	0	0
1	2	1

This is Sobel operator to find gradient component in y-axis

-1	0	1
-2	0	2
-1	0	1

This is Sobel operator to find gradient component in x-axis

Step #4 Non-maximum suppression:

Edges obtained after applying Sobel operator are thick so to make them thin, we use this non-maximum suppression. This will mark the local maxima only as the edges. For this first we have to find the gradient of edges, and then we compare the value of pixels both sides to the given pixel in the direction as per the value of gradient and if given pixel intensity is greater than both the pixels then it is considered as edge otherwise not i.e. local maxima. For gradient we use the formula: **gradient = $\tan^{-1}$ (V/H)**

Step #5 Double thresholding:

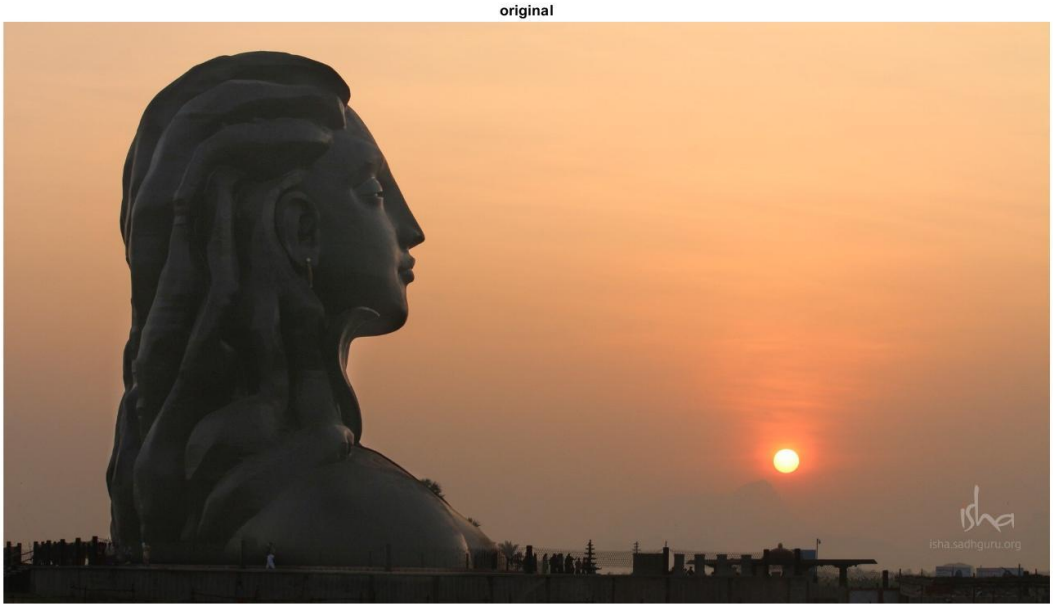
In this we apply two threshold levels (upper threshold and lower threshold). Those edges which are having value less than lower threshold value will be ignored and those whose value lies between the two threshold values will be considered as weak edges and the edges whose value is greater than the upper threshold value will be considered as strong edges.

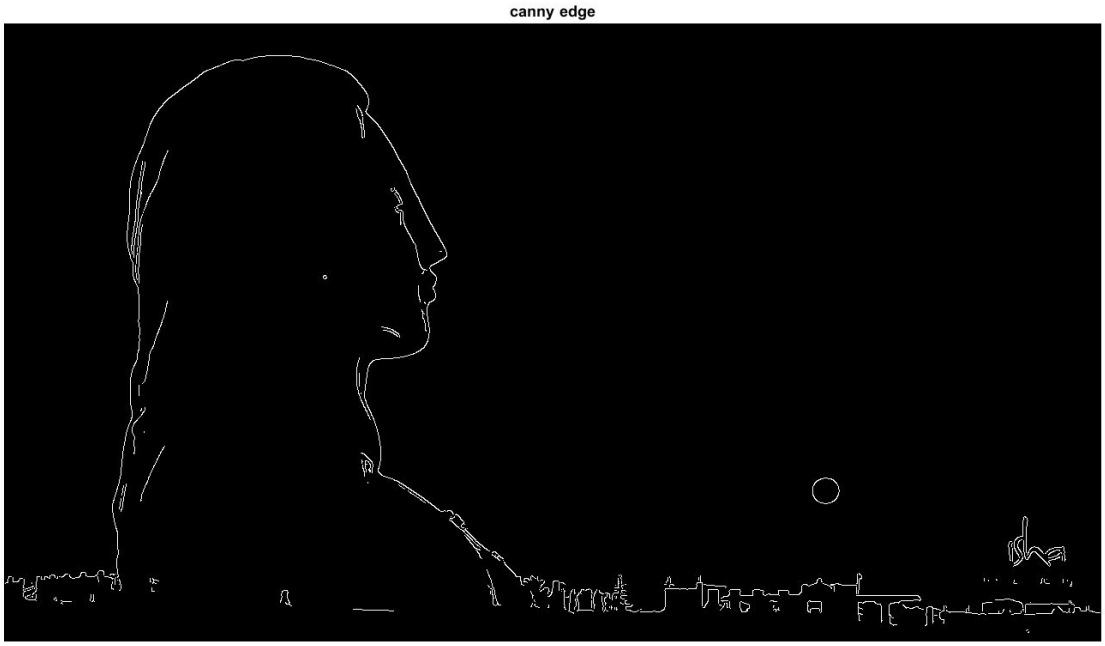
So, after applying double threshold we neglect the edges having value less than lower threshold value and keep only those edges whose value is greater than the lower threshold value.

Step #6 Edge hysteresis:

After applying double threshold there will be some weak edges which are not connected to strong edges so we have to remove all those weak edges which are not connected to strong edges and keep only connected ones. To track the edge connection, connected component analysis is applied by looking at the weak edge and its 8-connected neighbourhood pixels. If there is at least one strong edge pixel is connected to weak edge then we have to preserve that weak edge otherwise we ignore it.

Experimental results





Conclusion

To analyse an image, it's important for machine to know about its structure. This can be done by various edge detection algorithms. One such algorithm is canny edge detection which we have implemented in this project, it is one of the most efficient algorithms for edge detection. We have observed the results of each and every step and how it removes the noise and find the strong edges of the image step by step. Edge detection found it's uses in various areas such as object detection, robotic vision, satellite image processing, fingerprint scanner etc. In self driving cars edge detection is used to detect road limits, roundabouts, road separators, nearby vehicles etc. it is also used in finding pathological objects in medical images such as tumours. We learnt many things in this project which will be helpful to us in our further study of image processing and computer vision.