Title : To create a program for shape(edge) detection system in python

Introduction :

Edge detection is a technique of image processing used to identify points in a digital image with discontinuities, simply to say, sharp changes in the image brightness. These points where the image brightness varies sharply are called the edges (or boundaries) of the image.

Edge detection is the main tool in pattern recognition, image segmentation and scene analysis. It is a type of filter which is applied to extract the edge

Data Used :

Librarys:

Here we use turtle, numpy,cv2 and matplotlib;

Turtle :

Turtle is a pre-installed Python library that enables users to create pictures and shapes by providing them with a virtual canvas. The onscreen pen that you use for drawing is called the turtle and this is what gives the library its name.

Numpy :

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package.

Cv2 :

cv2 is the module import name for opencv-python, "Unofficial pre-built CPU-only OpenCV packages for Python" Basically this library is use for image processing

Matplotlib :

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

Methods :

cvtColor() - cv2.cvtColor() method is used to convert an image from one color space to another. There are more than 150 color-space conversion methods available in OpenCV.

Canny()- we will see the **Canny Edge** filter in OpenCV. Canny() Function in OpenCV is used to detect the edges in an image

Hugelinep() - The Hough Transform is a method that is used in image processing to detect any shape, if that shape can be represented in mathematical form. It can detect the shape even if it is broken or distorted a little bit. We will see how Hough transform works for line detection using the HoughLine transform method. To apply the Houghline method, first an edge detection of the specific image is desirable. For the edge detection technique go through the article

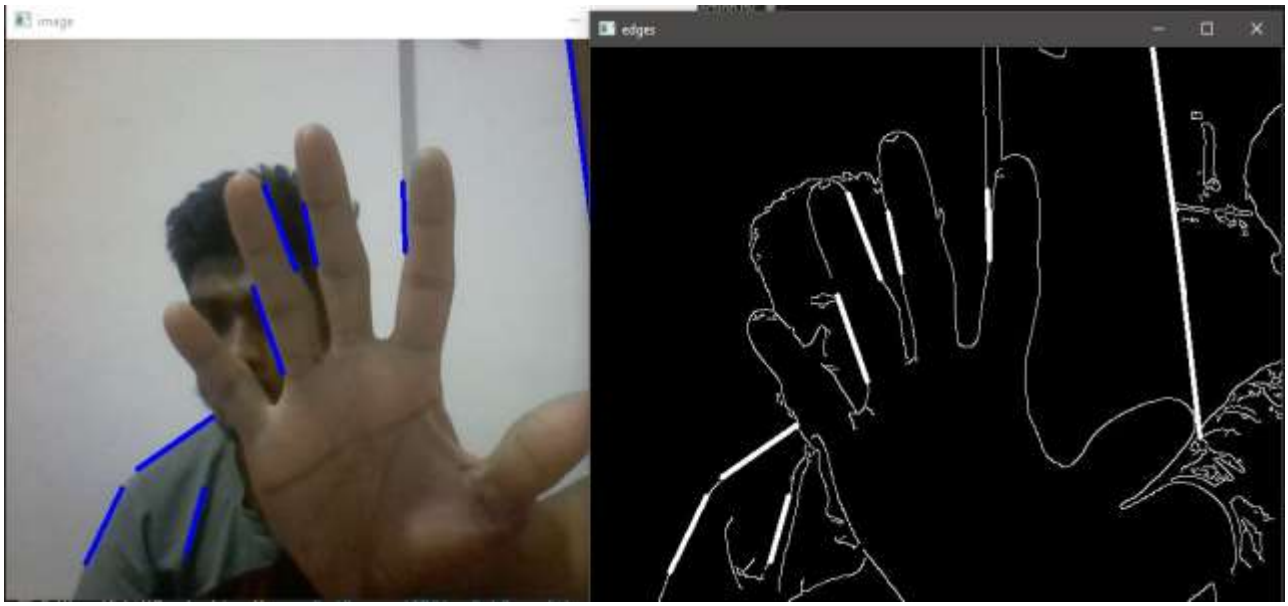
Code :

```
from turtle import clear # enables users to create pictures and shapes by providing them with a virtual canvas

import numpy as np # use numerical operations

import matplotlib.pyplot as plt # use for data manipulation
```

```
import cv2 # use for image processsing & video processing


cap = cv2.VideoCapture(0) # video capturing


while True:
    _, image = cap.read()
    # covert to greyscale
    grayscale = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # detect edges
    edges = cv2.Canny(grayscale, 30, 100)
    #detect lines in the image using hough lines technique
    lines = cv2.HoughLinesP(edges, 1, np.pi/180, 60, np.array([]), 50, 5)
    # drwaining lines
    for line in lines:
        for x1, y1, x2, y2 in line:
            cv2.line(image, (x1, y1), (x2, y2), (255, 0, 0), 3)
            cv2.line(edges, (x1, y1), (x2, y2), (255, 0, 0), 3)
    # show images
    cv2.imshow("image", image)
    cv2.imshow("edges", edges)
    if cv2.waitKey(250) == ord("q"):
        break


cap.release()
```

```
cv2.destroyAllWindows()
```

Output :



Conclusion and Future Scope :

This paper surveys the various state of art of Edge detection techniques in digital image processing. All techniques and algorithm have their own advantages and disadvantages. The analysis of various edge detection techniques are done on the basis of certain parameters. Fast processing response is the main requirement in many image processing applications. The operations performed by image processing algorithms can be computationally costly due to

their manipulating large amount of data. To make a program execute in real time, data needs to be processed in parallel and often a great deal of optimization needs to be utilized. In future work parallelizing an edge detection algorithms, provides better performance results for image-processing applications.

Applications :

Perhaps the most widespread application of edge detection is for object recognition in computer vision. Whenever objects are present in an image, there will be an edge between the boundary of the object and the background behind the object. These algorithms identify thresholds where there is a large gradient between neighbouring pixels, indicating a significant change in the intensity of the image. Even today, simple yet powerful algorithms such as Canny edge detection are widely used as a pre-processing step for many computer vision systems.

- AI API
- AI Robots
- AI Object detection
- Facial Recognition

Reference's :

1. Github
2. Python.org
3. Geeks for geek