image_proc02

October 31, 2021

1 Knowing how to detect edges

This is a follow up and advancement of the first tutorial on image processing.

Disclaimer: You can always refer to **the first tutorial** anytime.

But if you have understood all that was described in the first, **congrats on your first research!**

[7]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
% matplotlib inline
```

```
UsageError: Line magic function `%` not found.
```

1.0.1 Edge Detection

Boundaries of any object is called an edge. In image processing you can extract the edges of object by very ease. It is basically a high pass filter that completely removes anything that is not an edge.

[2]:
```
img = cv2.imread('samples/D (198).jpg')
im = cv2.resize(img, (500,500))
gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)

edges = cv2.Canny(gray, 550, 150)

cv2.imshow('im', im)
cv2.imshow('gray', gray)
cv2.imshow('edges', edges)

cv2.waitKey(0)
```

[2]:
```
113
```

1.0.2 Try it out with a video

[8]:
```
cap = cv2.VideoCapture(0)

while(cap.isOpened()):
    _, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 100)

    cv2.imshow('frame', frame)
    cv2.imshow('edges', edges)

    if cv2.waitKey(1) & 0xff == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

1.0.3 Object Tracking

As we have covered all the basics of image processing which includes masking, finding edges, knowing the shape of images and frames of video; we can now proceed towards **making an object tracker from scratch!**

This will include the concept of: - threshold - contours - image moments - finding the area of images - finding the coordinates of contours - drawing the boundary around the object using those coordinates

Contours: Any closed area is a contour. As seen from the edge detector concept, every closed area you see there is a contour. #### Image Moments: The weighted average of all the pixel intensities in a image is called moment. We can find out the center of the weighted average hence, can calculate the centroid of an image.

[11]:
```
cap = cv2.VideoCapture(0)
min_area = 100*100

while(cap.isOpened):
    _,frame = cap.read()
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    upper = np.array([162,58,62])
    lower = np.array([255,255,255])

    mask = cv2.inRange(hsv, upper, lower)
    res = cv2.bitwise_and(frame, frame, mask=mask)

    # detection of contours
    res_gray = cv2.cvtColor(res, cv2.COLOR_BGR2GRAY)
```

```
    _, threshold = cv2.threshold(res_gray, 0, 255, cv2.THRESH_BINARY+cv2.
↪THRESH_OTSU)
    contours, _ = cv2.findContours(threshold, cv2.RETR_EXTERNAL, cv2.
↪CHAIN_APPROX_NONE)

    for contour in contours:
        area = cv2.contourArea(contour)
        if area>min_area:
            # calculating the centroid
            moment = cv2.moments(contour)
            cx = int(moment['m10']/moment['m00'])
            cy = int(moment['m01']/moment['m00'])

            # make a rectangle bounding the contour
            [x, y, w, h] = cv2.boundingRect(contour)

            # draw a rectangle surrounding the contour image
            cv2.rectangle(frame, (x, y), (w+x, h+y), (0,255,0), 2)

            # put the centroid text
            cv2.putText(frame, str(cx)+','+str(cy), (cx,cy), 2, 1, (255,0,0),␣
↪1, 0)
        #endif
    #endfor
    cv2.imshow('frame', frame)
    if cv2.waitKey(1) & 0xff == ord('q'):
        break
    #endif
#endwhile

cap.release()
cv2.destroyAllWindows()
```

1.0.4 Let's do the fancy stuff in the next

Open the third tutorial where we will implement simple trignometry rules and plot them

[ ]: