CodeJAM 2

Aaryen Mehta

Approach

Detection of Bots

→ Detection of bots is done by detecting **motion** of the bots.
→ In this approach two consecutive frames are taken and the change in movement is detected.
→ For this, some basic image processing techniques are applied.
→ Such as, **changing to grayscale, blurring, thresholding, dilation**.
→ Then the contours in the image are detected and they are sorted according to area.
→ And the two largest contours are the ones that are covering the motion of the two bots.
→ Then for the detection of the bot, centroid is marked and tracked.
→ For this **centroid tracking** is used.
→ Then a bounding rectangle for both contours is found and drawn.
→ And labels, **bot 1 and bot 2** is put on the centroid coordinate.
→ The centroid, is sadly, selected **manually**.
→ I tried applying **template matching, aruco codes and contour based approach** but no effective results were obtained.
→ So I had to go for manually assignment of two centroids to their respective bot numbers.

Algorithm

→ No specific algorithms used.

Results

→ Results can be found in this repo in a file named **output.avi**:
→ Frame rate of the solution is same as the original video, **24 fps**.
→ The source code can also be found in this repo under **source.py**.
→ Here is the source code for reference purposes:

```
from cv2 import cv2
import numpy as np
import math


def dist(x1,y1,x2,y2):
    '''Function to compute Euclidean Distance'''
    return np.linalg.norm(np.array([x2-x1,y2-y1]))


def mind(x1o,y1o,x1n,y1n,x2n,y2n):
    '''This function finds closest point among two new centroid points of two bots'''
    per = 70/100  #this is the error criteria
    if dist(x1o,y1o,x1n,y1n) > dist(x1o,y1o,x2n,y2n):
        if x1o*(1+per) > x2n and x1o*(1-per) < x2n and y1o*(1+per) > y2n and y1o*(1-per) < y2n:
            return (x2n,y2n)
        else :
            return (x1o,y1o)
    elif dist(x1o,y1o,x1n,y1n) <= dist(x1o,y1o,x2n,y2n):
        if x1o*(1+per) > x1n and x1o*(1-per) < x1n and y1o*(1+per) > y1n and y1o*(1-per) < y1n:
            return (x1n,y1n)
        else :
            return (x1o,y1o)


cap = cv2.VideoCapture('sentry3.mkv')

fourcc = cv2.VideoWriter_fourcc(*'XVID')

out = cv2.VideoWriter('output.avi',fourcc, 24.0, (1440,810))

ret, frame1 = cap.read()
ret, frame2 = cap.read()

count = 0

while cap.isOpened():

    try:
        diff = cv2.absdiff(frame1,frame2)
    except:
```

```
        break

    gray = cv2.cvtColor(diff,cv2.COLOR_BGR2GRAY)
    blur = cv2.GaussianBlur(gray,(5,5),0)
    _,thresh = cv2.threshold(blur,20,255,cv2.THRESH_BINARY)
    dilated = cv2.dilate(thresh,None,iterations=3)
    contours,heirarchy = cv2.findContours(dilated,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
    contours = sorted(contours,key=cv2.contourArea,reverse=True)[0:2]

    m2 = cv2.moments(contours[1])
    a,b,c,d = cv2.boundingRect(contours[1])
    x2_n = a+c/2
    y2_n = b+d/2

    m1 = cv2.moments(contours[0])
    a,b,c,d = cv2.boundingRect(contours[0])
    x1_n = a+c/2
    y1_n = b+d/2

    if count == 0 :
        x1_o,x2_o,y1_o,y2_o = x1_n,x2_n,y1_n,y2_n
        count += 1
    else :
        (x1_o,y1_o),(x2_o,y2_o) = mind(x1_o,y1_o,x1_n,y1_n,x2_n,y2_n),mind(x2_o,y2_o,x1_n,y1_n,x2_n,y2_n)
        #print('x1_o=',x1_o,'y1_o=',y1_o,'x2_o=',x2_o,'y2_o=',y2_o,'x1_n=',x1_n,'y1_n=',y1_n,'x2_n=',x2_n,'y2_n=',y2_n)
        if x1_o == x2_o and y1_o == y2_o:
            print()
    #cv2.circle(frame1, (x2_o, y2_o), 5, (0, 0, 255), -1)
    for cnt in (contours):
        (x,y,w,h) = cv2.boundingRect(cnt)

        #if cv2.contourArea(cnt) < 3000 :
        #    continue
        cv2.rectangle(frame1,(x,y),(x+w,y+h),(0,255,0),2)
        cv2.putText(frame1, 'bot 1', (int(x1_o), int(y1_o)+100), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0,255,0), 2)
```

```
77          cv2.putText(frame1, 'bot 2', (int(x2_o), int(
    y2_o)+100), cv2.FONT_HERSHEY_SIMPLEX, 0.9,
    (0,255,0), 2)
78
79      out.write(frame1)
80
81      cv2.imshow('feed',frame1)
82
83      frame1 = frame2
84      ret,frame2 = cap.read()
85
86      if cv2.waitKey(42) == 27:
87          break
88
89  cap.release()
90  out.release()
91  cv2.destroyAllWindows()
```

Listing 1: Source Code