OPENCV-PROJECT
FIFA Team Detection Using OpenCV

The code is limited to detecting Brazil and Portugal (2018) FIFA home game jerseys.

After importing the required libraries, a function was defined in order to differentiate between Brazil and Portugal players.

```
def team(n):
    if n == 0:
        name = "Portugal"
        lower = (0, 90, 70)
        upper = (10, 255, 255)
        return (name, lower, upper)
    if n == 1:
        name = "Brazil"
        lower = (20, 100, 100)
        upper = (30, 255, 255)
        return (name, lower, upper)
```

If the argument is 0, the function returns the information for Portugal and the argument of 1 returns detail for Brazil. These HSV values were obtained by trial and error.

> I realized that I should have used Trackbars in order to obtain the exact values for more accuracy, so I am trying to do that now. The following is the code I used, it was obtained from the tutorial video that was provided earlier to learn OpenCV. However, the values were not changing in real-time when I moved the trackbars. I am in the process of figuring out the issues with this.

```
cv2.namedWindow("Trackbars")
cv2.resizeWindow("Trackbars", 640, 240)
cv2.createTrackbar("Hue Min", "Trackbars", 0, 179, empty)
cv2.createTrackbar("Hue Max", "Trackbars", 179, 179, empty)
cv2.createTrackbar("Sat Min", "Trackbars", 0, 255, empty)
cv2.createTrackbar("Sat Max", "Trackbars", 255, 255, empty)
cv2.createTrackbar("Val Min", "Trackbars", 0, 255, empty)
cv2.createTrackbar("Val Max", "Trackbars", 255, 255, empty)

while True:
    img = cv2.imread(path)
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    h_min = cv2.getTrackbarPos("Hue Min", "TrackBars")
    h_max = cv2.getTrackbarPos("Hue Max", "TrackBars")
    s_min = cv2.getTrackbarPos("Sat Min", "TrackBars")
    s_max = cv2.getTrackbarPos("Sat Max", "TrackBars")
    v_min = cv2.getTrackbarPos("Val Min", "TrackBars")
    v_max = cv2.getTrackbarPos("Val Max", "TrackBars")
    print(h_min, h_max, s_min, s_max, v_min, v_max)
    lower = np.array([h_min, s_min, v_min])
    upper = np.array([h_max, s_max, v_max])
    mask = cv2.inRange(hsv, lower, upper)
```

To identify the colors, the first step is to convert the image to HSV, in the following manner.

```
img=cv2.imread(path)
hsv=cv2.cvtColor(img,cv2.COLOR_BGR2HSV)
```

Then a for loop was used to obtain the colors for the different teams which was later used to create the masks. Since the function has three return values, three variables were created to store the return values.

These values were used to create the masks and to find the contours, using the findContours method. The biggest area is needed to draw the bounding rectangle so the sorted method is used to sort the areas in ascending order. As for the biggest area, the reverse=True is used to sort the areas in the descending order. Then the biggest area is used to obtain the coordinates which are later used to draw the rectangle using the rectangle method.

```
for i in range(2):
    name,lower,upper=team(i)
    mask=cv2.inRange(hsv, lower, upper)

contour,her=cv2.findContours(mask.copy(),cv2.RETR_EXTERNAL,cv2.CHAIN_APPROX_N
ONE)
    big=sorted(contour,key=cv2.contourArea,reverse=True)[0]
    rect=cv2.boundingRect(big)
    x,y,w,h=rect
    cv2.rectangle(img,(x,y),(x+w,y+h),(255,23,0),2)
    cv2.putText(img,name, (x,y),cv2.FONT_HERSHEY_SIMPLEX,2,(255,23,0))
```

The putText method finally labels the rectangles based on the name value in obtained from the team function.

The imshow method displays the image to the user in the end.

```
cv2.imshow("Image",img)
#cv2.imshow("Mask",mask)
cv2.waitKey(0)
```