In [1]:

```
import cv2
import numpy as np

cap = cv2.VideoCapture(0)

while(cap.isOpened()):
    _,frame = cap.read()
    b_f = np.zeros(frame.shape)
    #frame  = cv2.medianBlur(frame,5)
    f_c = frame.copy()
    roi = frame[100:300,100:300]
    gray = cv2.cvtColor(roi,cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray,(7,7),0)
    #hist = cv2.equalizeHist(gray)
    edge = cv2.Canny(gray,25,255)
    contour,_ = cv2.findContours(edge.copy(),cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)

    #cnt = max(contour, key= lambda x: cv2.contourArea(x))

    #print(len(contour))
    cv2.rectangle(frame,(100,100),(350,350),(255,0,0),3)
    cv2.imshow("frame",frame)
    cv2.drawContours(b_f,contour,-1,(0,255,0),3)
    cv2.imshow("b_f",b_f)
    cv2.imshow("edge",edge)
    #cv2.imshow("roi",roi)
    if cv2.waitKey(1) & 0XFF == ord("q"):
        break
cap.release()
cv2.destroyAllWindows()
```

In [ ]:

```
import cv2
import numpy as np

cap = cv2.VideoCapture(0)


while(cap.isOpened()):
    try:

        _,frame = cap.read()
        b_f = np.zeros(frame.shape)
        #frame  = cv2.medianBlur(frame,5)
        f_c = frame.copy()
        roi = frame[100:300,100:300]
        gray = cv2.cvtColor(roi,cv2.COLOR_BGR2GRAY)
        gray = cv2.GaussianBlur(gray,(7,7),0)
        #hist = cv2.equalizeHist(gray)
        edge = cv2.Canny(gray,25,255)
        contour,_ = cv2.findContours(edge.copy(),cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
        print(len(contour))
        cnt = max(contour, key= lambda x: cv2.contourArea(x))

        epsilon = 0.0005*cv2.arcLength(cnt,True)
        approx = cv2.approxPolyDP(cnt,epsilon,True)
        hull = cv2.convexHull(cnt)

        areahull = cv2.contourArea(hull)
        areacnt = cv2.contourArea(cnt)

        arearatio = ((areahull - areacnt)/areacnt)*100
        hull = cv2.convexHull(approx,returnPoints=False)
        defects = cv2.convexityDefects(approx,hull)
        print(defects)
        #print(len(contour))
        cv2.rectangle(frame,(100,100),(350,350),(255,0,0),3)
        cv2.imshow("frame",frame)
        cv2.drawContours(b_f,contour,-1,(0,255,0),3)
        cv2.imshow("b_f",b_f)
        cv2.imshow("edge",edge)
    except:
        pass
    #cv2.imshow("roi",roi)
    if cv2.waitKey(1) & 0XFF == ord("q"):
        break
cap.release()
cv2.destroyAllWindows()
```

In [2]:

```
import cv2
import numpy as np
import math
cap = cv2.VideoCapture(0)


while(cap.isOpened()):
    try:

        _,frame = cap.read()
        b_f = np.zeros(frame.shape)
        #frame  = cv2.medianBlur(frame,5)
        f_c = frame.copy()
        roi = frame[100:300,100:300]
        gray = cv2.cvtColor(roi,cv2.COLOR_BGR2GRAY)
        gray = cv2.GaussianBlur(gray,(7,7),0)
        #hist = cv2.equalizeHist(gray)
        edge = cv2.Canny(gray,25,255)
        contour,_ = cv2.findContours(edge.copy(),cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
        print(len(contour))
        cnt = max(contour, key= lambda x: cv2.contourArea(x))

        epsilon = 0.0005*cv2.arcLength(cnt,True)
        approx = cv2.approxPolyDP(cnt,epsilon,True)
        hull = cv2.convexHull(cnt)

        areahull = cv2.contourArea(hull)
        areacnt = cv2.contourArea(cnt)

        arearatio = ((areahull - areacnt)/areacnt)*100
        hull = cv2.convexHull(approx,returnPoints=False)
        defects = cv2.convexityDefects(approx,hull)
        print(defects)
        l=0

    #code for finding no. of defects due to fingers
        for i in range(defects.shape[0]):
            s,e,f,d = defects[i,0]
            start = tuple(approx[s][0])
            end = tuple(approx[e][0])
            far = tuple(approx[f][0])
            pt= (100,180)


            # find length of all sides of triangle
            a = math.sqrt((end[0] - start[0])**2 + (end[1] - start[1])**2)
            b = math.sqrt((far[0] - start[0])**2 + (far[1] - start[1])**2)
            c = math.sqrt((end[0] - far[0])**2 + (end[1] - far[1])**2)
            s = (a+b+c)/2
            ar = math.sqrt(s*(s-a)*(s-b)*(s-c))

            #distance between point and convex hull
            d=(2*ar)/a

            # apply cosine rule here
            angle = math.acos((b**2 + c**2 - a**2)/(2*b*c)) * 57


            # ignore angles > 90 and ignore points very close to convex hull(they generally
```

```
            if angle <= 90 and d>30:
                l += 1
                cv2.circle(b_f, far, 3, [255,0,0], -1)

            #draw lines around hand
            cv2.line(b_f,start, end, [0,255,0], 2)


        l+=1

        #print corresponding gestures which are in their ranges
        font = cv2.FONT_HERSHEY_SIMPLEX
        if l==1:
            if areacnt<2000:
                cv2.putText(frame,'Put hand in the box',(0,50), font, 2, (0,0,255), 3, cv2.
            else:
                if arearatio<12:
                    cv2.putText(frame,'0',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
                elif arearatio<17.5:
                    cv2.putText(frame,'Best of luck',(0,50), font, 2, (0,0,255), 3, cv2.LIN

                else:
                    cv2.putText(frame,'1',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

        elif l==2:
            cv2.putText(frame,'2',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

        elif l==3:
            if arearatio<27:
                cv2.putText(frame,'3',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
            else:
                cv2.putText(frame,'ok',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

        elif l==4:
            cv2.putText(frame,'4',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

        elif l==5:
            cv2.putText(frame,'5',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

        elif l==6:
            cv2.putText(frame,'reposition',(0,50), font, 2, (0,0,255), 3, cv2.LINE_AA)

        else :
            cv2.putText(frame,'reposition',(10,50), font, 2, (0,0,255), 3, cv2.LINE_AA)
        cv2.rectangle(frame,(100,100),(350,350),(255,0,0),3)
        cv2.imshow("frame",frame)
        cv2.drawContours(b_f,contour,-1,(0,255,0),3)
        #cv2.circle(bf,leftmost,8,(255,0,0),-1)
        cv2.imshow("b_f",b_f)
        cv2.imshow("edge",edge)
    except:
        pass
    #cv2.imshow("roi",roi)
    if cv2.waitKey(1) & 0XFF == ord("q"):
        break
cap.release()
cv2.destroyAllWindows()
```

```
3
[[[   0    6    3  243]]

 [[   6   26    9  419]]

 [[  27   31   30  179]]

 [[  31   41   32  114]]

 [[  42   81   58 2287]]]
3
[[[   1    5    2  162]]

 [[   5    7    6  114]]

 [[   7   25    8  469]]

 [[  25   27   26  142]]
```

In [ ]: