**MISY3875 Introduction to Computer Vision – Fall 2024**

Project-Shape Detection

Name-Surname: Mehmet Kepür

ID: 21MISY1007

Copy and paste your screen images here:

A screenshot of a video

Description automatically generated

A person with a blue face and a white circle

Description automatically generatedA screenshot of a video

Description automatically generatedA screenshot of a video

Description automatically generated

Add the codes of your project here:

import cv2

import numpy as np

def nothing(x):

    pass

def get\_color\_name(h, s, v):

    if s < 50 and v > 200:

        return "White"

    elif v < 50:

        return "Black"

    elif (h < 15 or h > 165) and s > 50:

        return "Red"

    elif 15 <= h < 35 and s > 50:

        return "Yellow"

    elif 35 <= h < 85 and s > 50:

        return "Green"

    elif 85 <= h < 125 and s > 50:

        return "Blue"

    elif 125 <= h < 165 and s > 50:

        return "Purple"

    else:

        return "Unknown"

cap = cv2.VideoCapture(0)

cv2.namedWindow("Settings")

cv2.createTrackbar("Lower-Hue", "Settings", 0, 180, nothing)

cv2.createTrackbar("Lower-Saturation", "Settings", 0, 255, nothing)

cv2.createTrackbar("Lower-Value", "Settings", 0, 255, nothing)

cv2.createTrackbar("Upper-Hue", "Settings", 0, 180, nothing)

cv2.createTrackbar("Upper-Saturation", "Settings", 0, 255, nothing)

cv2.createTrackbar("Upper-Value", "Settings", 0, 255, nothing)

font = cv2.FONT\_HERSHEY\_SIMPLEX

PIXELS\_TO\_CM = 0.0264

while True:

    ret, frame = cap.read()

    frame = cv2.flip(frame, 1)

    hsv = cv2.cvtColor(frame, cv2.COLOR\_BGR2HSV)

    lh = cv2.getTrackbarPos("Lower-Hue", "Settings")

    ls = cv2.getTrackbarPos("Lower-Saturation", "Settings")

    lv = cv2.getTrackbarPos("Lower-Value", "Settings")

    uh = cv2.getTrackbarPos("Upper-Hue", "Settings")

    us = cv2.getTrackbarPos("Upper-Saturation", "Settings")

    uv = cv2.getTrackbarPos("Upper-Value", "Settings")

    lower\_color = np.array([lh, ls, lv])

    upper\_color = np.array([uh, us, uv])

    mask = cv2.inRange(hsv, lower\_color, upper\_color)

    kernel = np.ones((5, 5), np.uint8)

    mask = cv2.erode(mask, kernel)

    contours, \_ = cv2.findContours(mask, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)

    for cnt in contours:

        area = cv2.contourArea(cnt)

        perimeter = cv2.arcLength(cnt, True)

        epsilon = 0.02 \* perimeter

        approx = cv2.approxPolyDP(cnt, epsilon, True)

        x = approx.ravel()[0]

        y = approx.ravel()[1]

        if area > 400:

            cv2.drawContours(frame, [approx], 0, (0, 0, 0), 5)

            area\_cm = area \* (PIXELS\_TO\_CM \*\* 2)

            perimeter\_cm = perimeter \* PIXELS\_TO\_CM

            shape = "Unknown"

            if len(approx) == 3:

                shape = "Triangle"

            elif len(approx) == 4:

                shape = "Rectangle"

            elif len(approx) == 5:

                shape = "Pentagon"

            elif len(approx) == 6:

                shape = "Hexagon"

            elif len(approx) > 6:

                shape = "Circle"

            mask\_shape = np.zeros\_like(mask)

            cv2.drawContours(mask\_shape, [cnt], -1, 255, -1)

            mean\_val = cv2.mean(hsv, mask=mask\_shape)

            color\_name = get\_color\_name(mean\_val[0], mean\_val[1], mean\_val[2])

            cv2.putText(frame, f"{shape} ({color\_name})", (x, y), font, 0.6, (255, 255, 255), 2)

            cv2.putText(frame, f"Area: {area\_cm:.2f} cm^2", (x, y + 30), font, 0.6, (255, 255, 255), 2)

            cv2.putText(frame, f"Perimeter: {perimeter\_cm:.2f} cm", (x, y + 60), font, 0.6, (255, 255, 255), 2)

    cv2.putText(frame, "21MISY1007", (10, 30), font, 0.8, (255, 255, 255), 2)

    cv2.imshow("Frame", frame)

    cv2.imshow("Mask", mask)

    if cv2.waitKey(3) & 0xFF == ord('q'):

        break

cap.release()

cv2.destroyAllWindows()