```python
import cv2
import numpy as np

def avoidObs(frame, px, w, h):
    ### Zero the percent of frame makeup ###
    percent_makeup = 0


    ### Computer vision boundaries ###
    # Looking for red and blue objects (obstacles) within this region of BGR values #
    bluebound = ([80, 40, 10], [150, 80, 30])
    redbound = ([0, 0, 50], [60, 80, 150])
    searchbound = ([],[])

    ### Print pixel BGR value ###
    print(px)

    ### Determine if the object is red, blue, or neither ###

    ### Check the B value ###
    if px[0] > bluebound[0][0] and px[0] < bluebound[1][0]:

        ### Check the G value ###
        if px[1] > bluebound[0][1] and px[0] < bluebound[1][1]:

            ### Check the R value ###
            if px[2] > bluebound[0][2] and px[0] < bluebound[1][2]:
                searchbound = bluebound
                print('Found something blue!')
            else:
                searchbound = ([0,0,0],[0,0,0])
        else:
            searchbound = ([0,0,0],[0,0,0])

    ### Check the B value ###
    elif px[0] > redbound[0][0] and px[0] < redbound[1][0]:

        ### Check the G value ###
        if px[1] > redbound[0][1] and px[0] < redbound[1][1]:

            ### Check the R value ###
            if px[2] > redbound[0][2] and px[0] < redbound[1][2]:
                searchbound = redbound
                print('Found something red!')
            else:
                searchbound = ([0,0,0],[0,0,0])
        else:
            searchbound = ([0,0,0],[0,0,0])
    else:
        searchbound = ([0,0,0],[0,0,0])



    ### Set the boundaries to the color boundaries from above ###
    boundaries = [
        searchbound
    ]

    ### For each of the bounds, iterate ###
    for (lower, upper) in boundaries:

        ### Create NumPy arrays from the boundaries ###
        lower = np.array(lower, dtype = "uint8")
        upper = np.array(upper, dtype = "uint8")

        ### Mask the colors within the boundaries ###
        mask = cv2.inRange(frame, lower, upper)

        ### Bitwise_and the frame using the mask  ###
        output = cv2.bitwise_and(frame, frame, mask = mask)


    ### Threshold the masked image, make the blue/red -> white, everything else black ###
    _, thresh = cv2.threshold(mask, 0, 255, 0)

    ### Find the contours (blobs) of the threshold image ###
    _, contours, _ = cv2.findContours(thresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)


    ### If there are contours: ###
    if len(contours) != 0:

        ### Find the biggest contour ###
        c = max(contours, key=cv2.contourArea)

        ### Find the x,y of the center, the width/height of the contour
        x,y,w,h = cv2.boundingRect(c)

        ### If the contour is too small, disregard it ###
        if w < 50 or h < 50:
            w = 0
            h = 0

    ### If the width/height exist: ###
    if h and w:

        ### Calculate the percent makeup of the frame ###
        percent_makeup = (h*w)/(240*320)*100

    ### Return the value ###
    return percent_makeup

```