Lab 1 report

Yiming Guo

105338468

My github page:

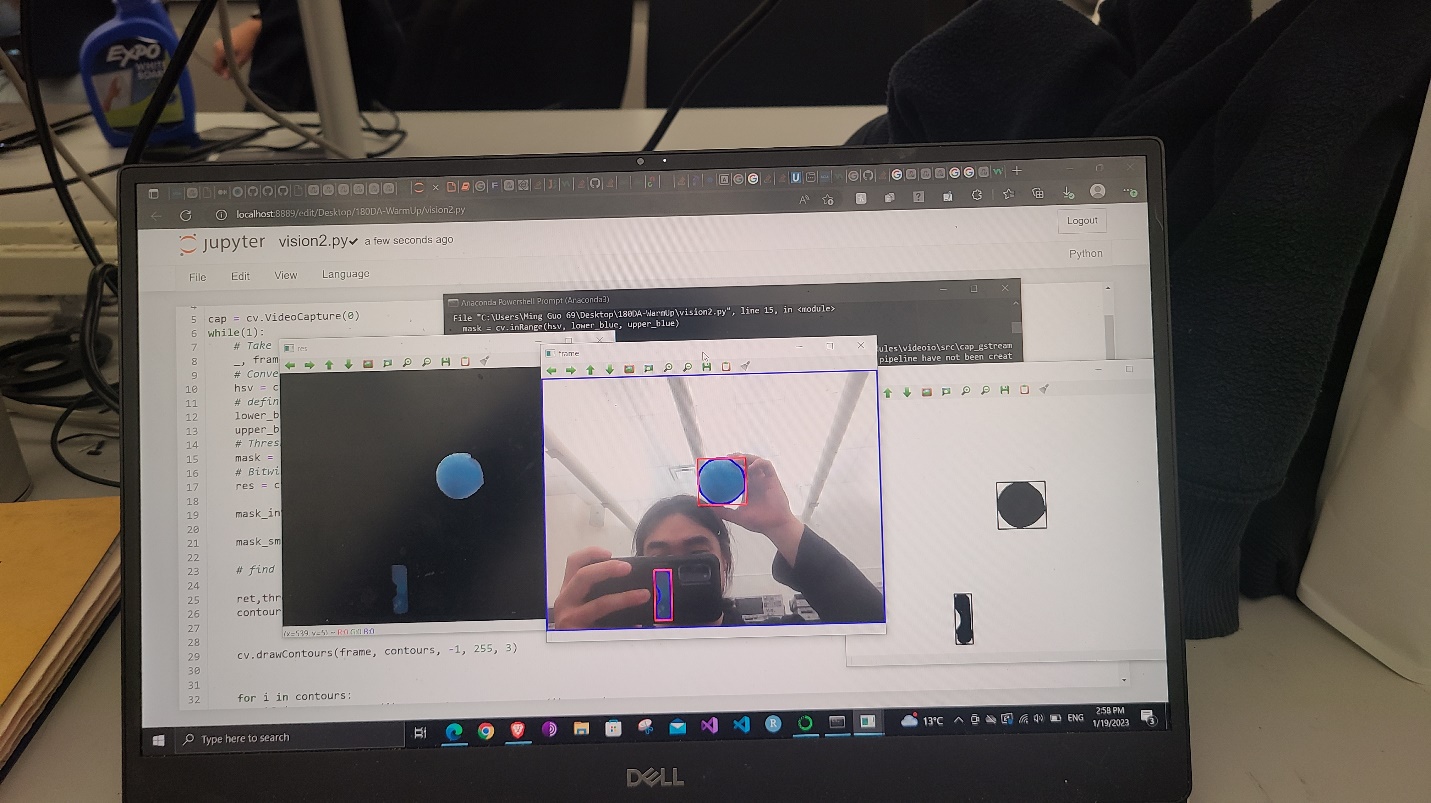
[Ben-Dove-666/180DA-WarmUp (github.com)](https://github.com/Ben-Dove-666/180DA-WarmUp)

Note: vision2.py is the file for task4. (contour tracking and color detection)

Task4/1

I think HSV is better than RGB in color detection since It takes into account the effect of hue and saturation, which is often influenced by the background light conditions. Since RGB does not account for this as well, it often leads to detection of undesired components of an image.

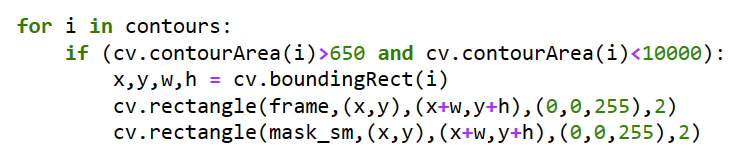
The HSV values I used was for detection of blue color

I am not sure what the largest threshold range was but the above threshold range was sufficient for detection of blue objects.

Above screenshot is proof that vision2.py can track a blue object on screen.

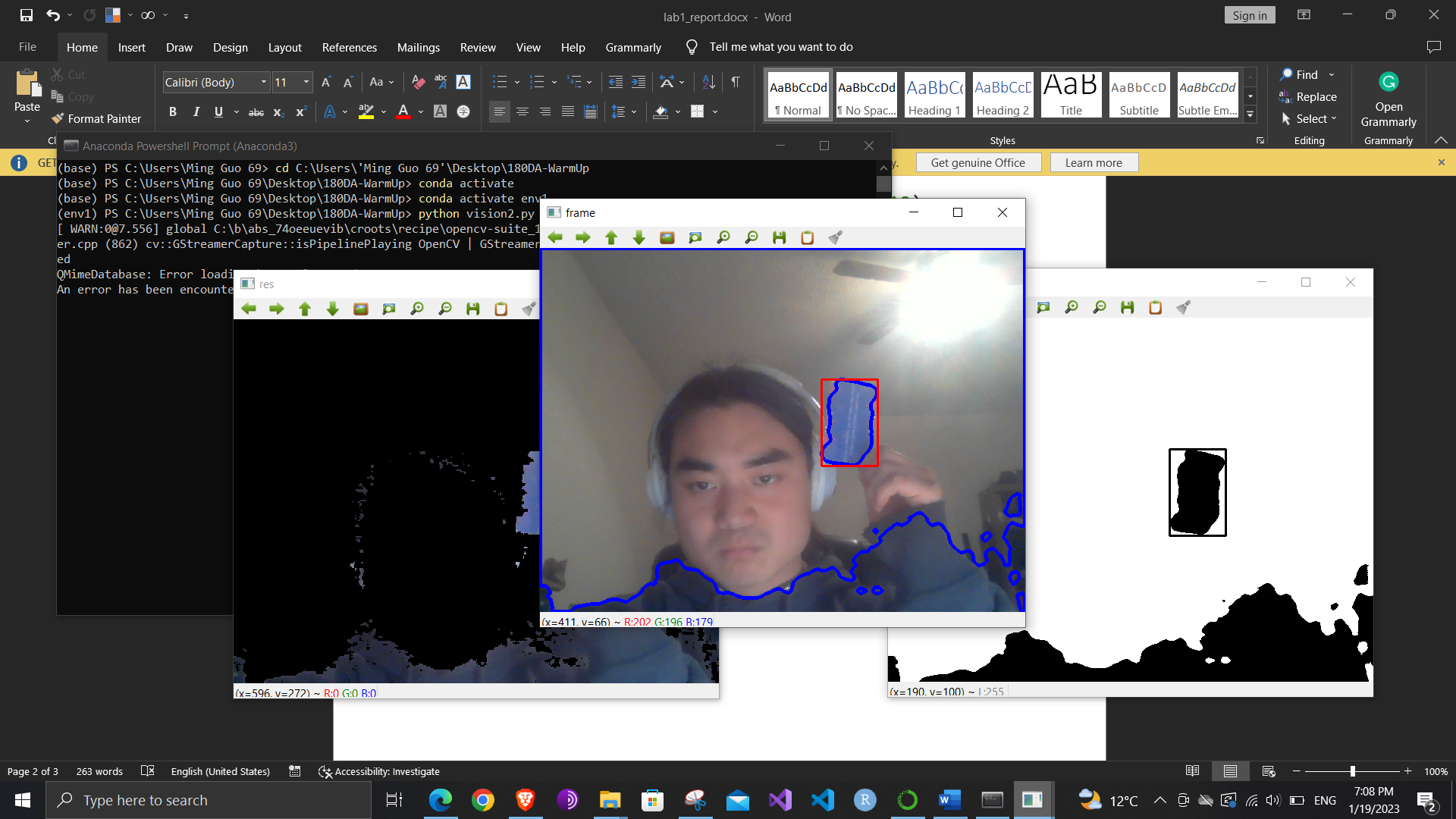
The baseline used for the program is [OpenCV: Changing Colorspaces](https://docs.opencv.org/4.x/df/d9d/tutorial_py_colorspaces.html) and the color detection code is sourced from [Finding Dominant Colour on an Image | by ayşe bilge gündüz | Code Like A Girl](https://code.likeagirl.io/finding-dominant-colour-on-an-image-b4e075f98097)

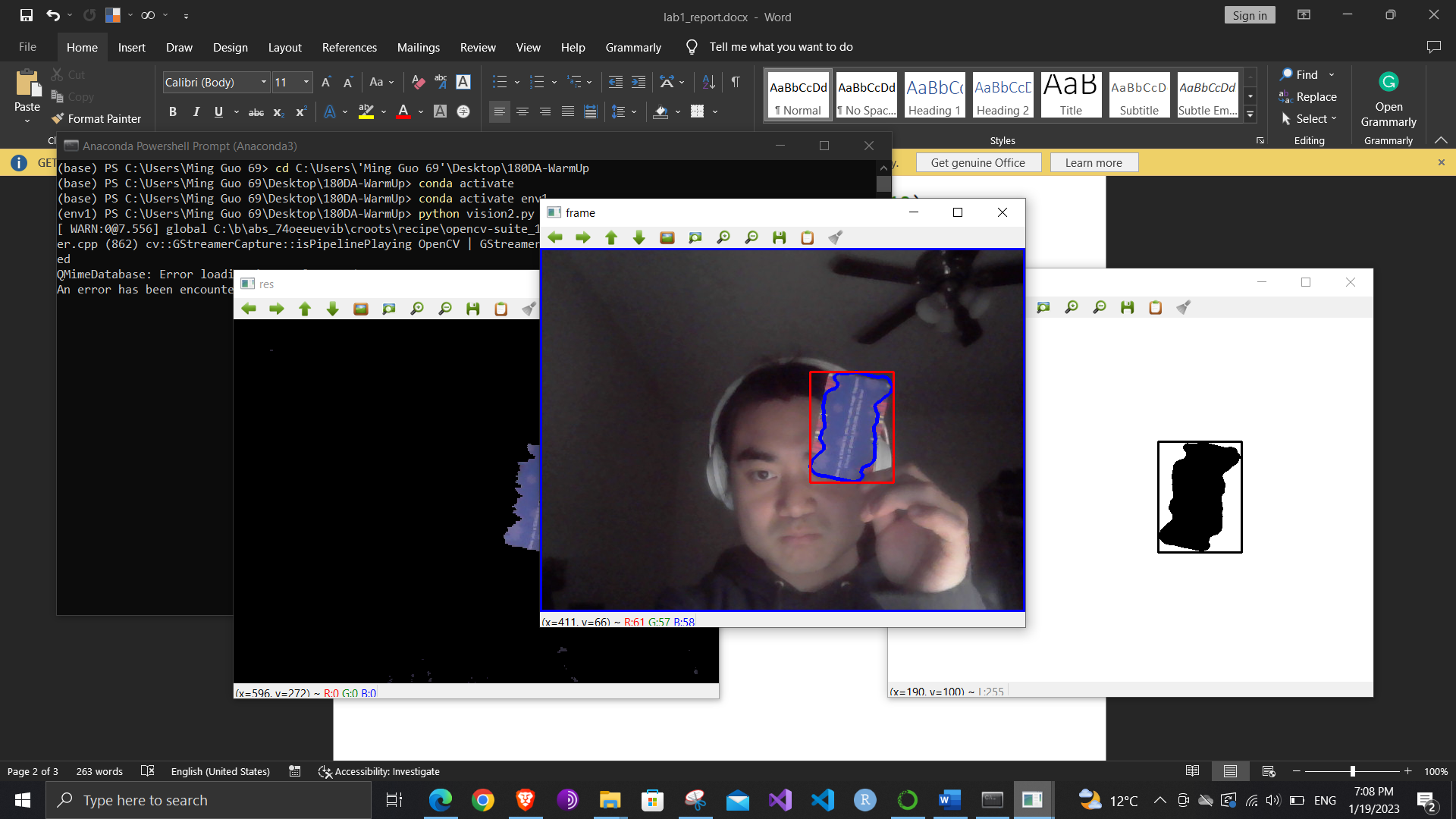
I made a few improvements for it to work:

1. The cv.findContours function tracks contours around a black object above a white background. However, the mask provided in the code is reversed. Hence I reversed the image to be able to track the contour around desired blue object
2. The camera on my laptop is very grainy, leading to the program tracing the contours of many smaller color patches caused by the noise, instead of the entire object. I solved this problem by smoothing out the mask with a median filter
3. For some reason the program tracks the entire frame of the camera image as a contour. Hence I set a condition on the contour to holefully select the correct contour around the desired object by limiting the area inside the contour.

Task2/2

below are the results for detecting a blue card in my own room, with ceiling lights on and off

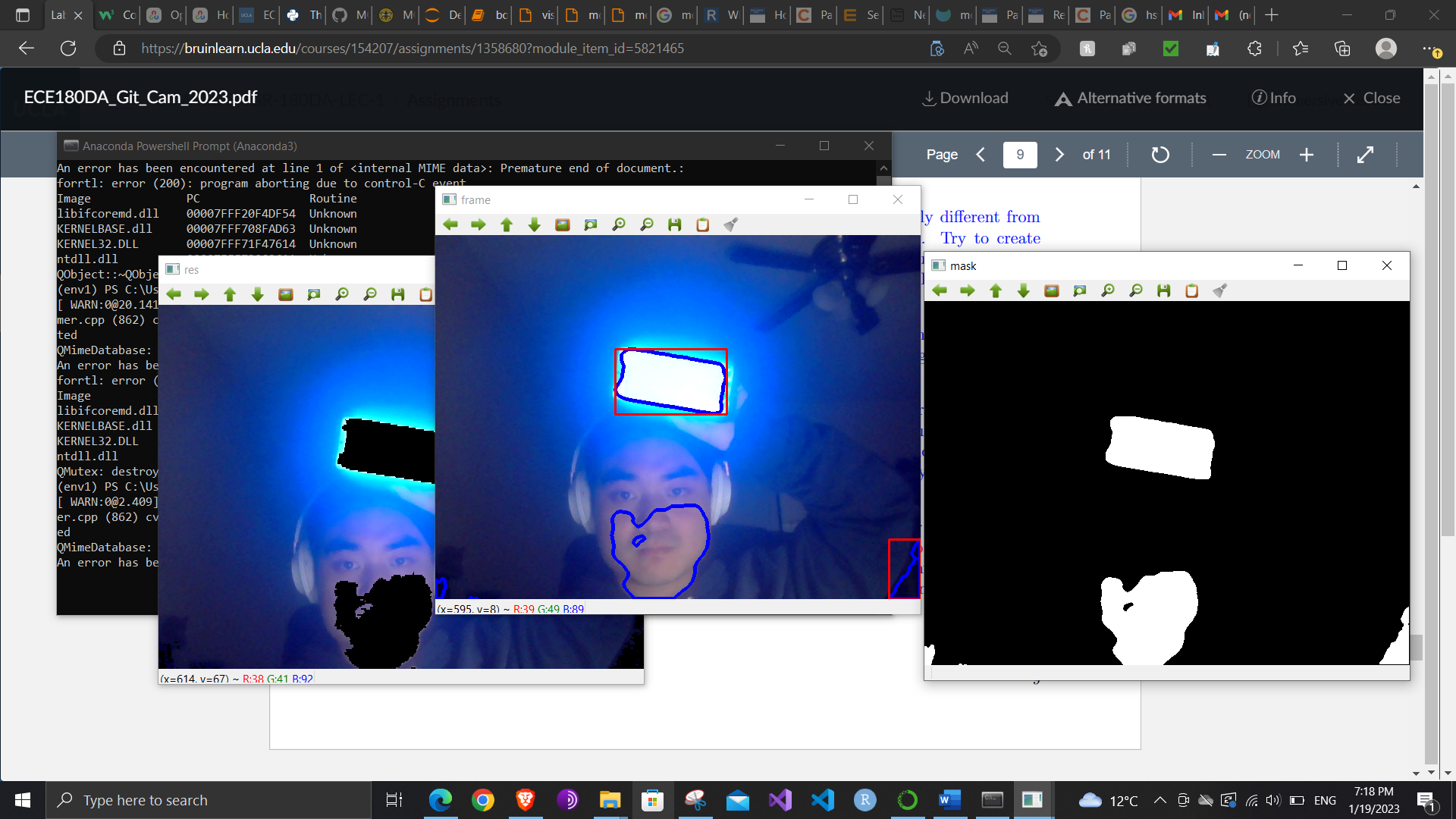


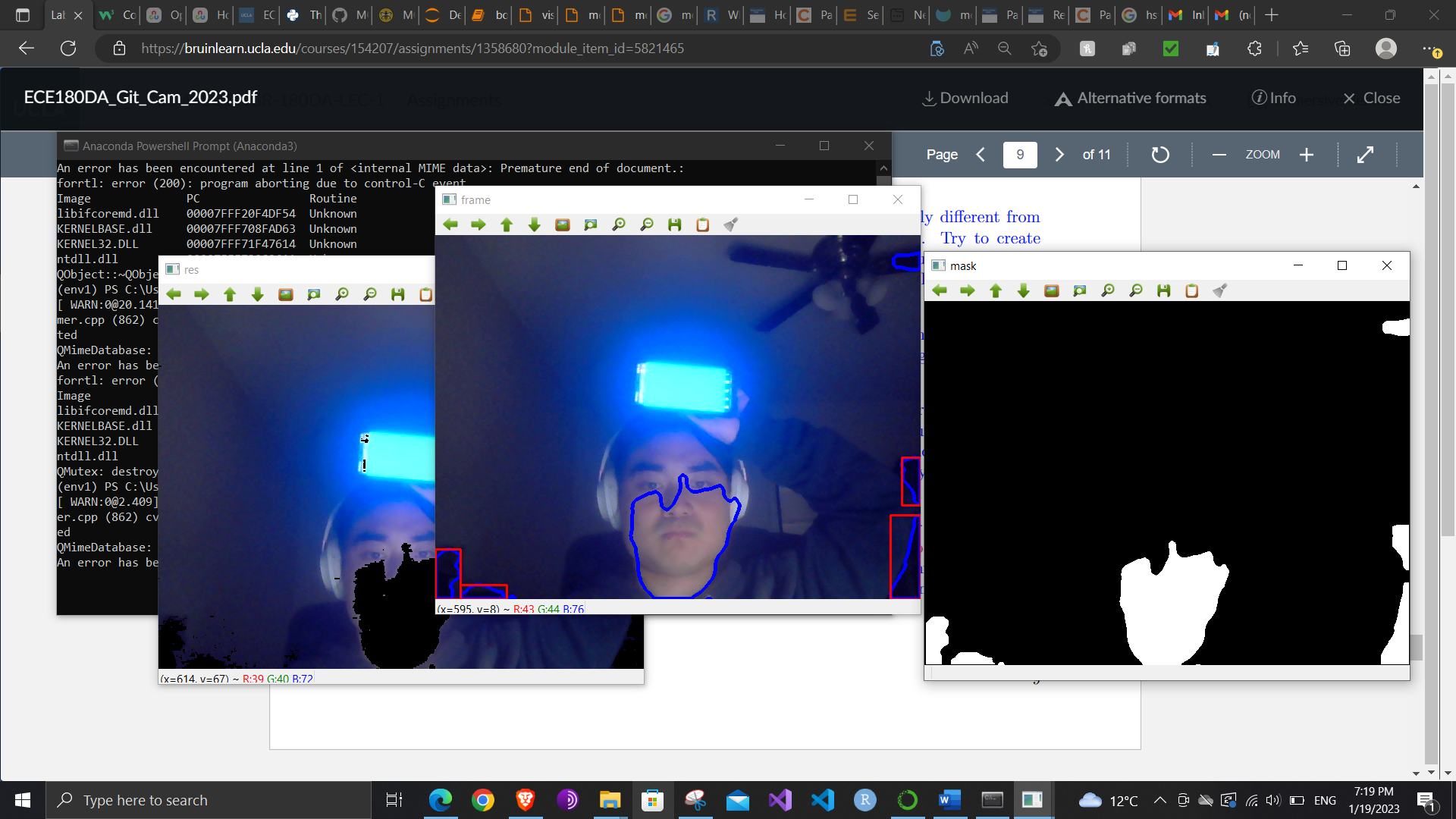


I didn’t notice a huge change in the detection ability of the program with regard to the presence of background lighting

Task 4/3

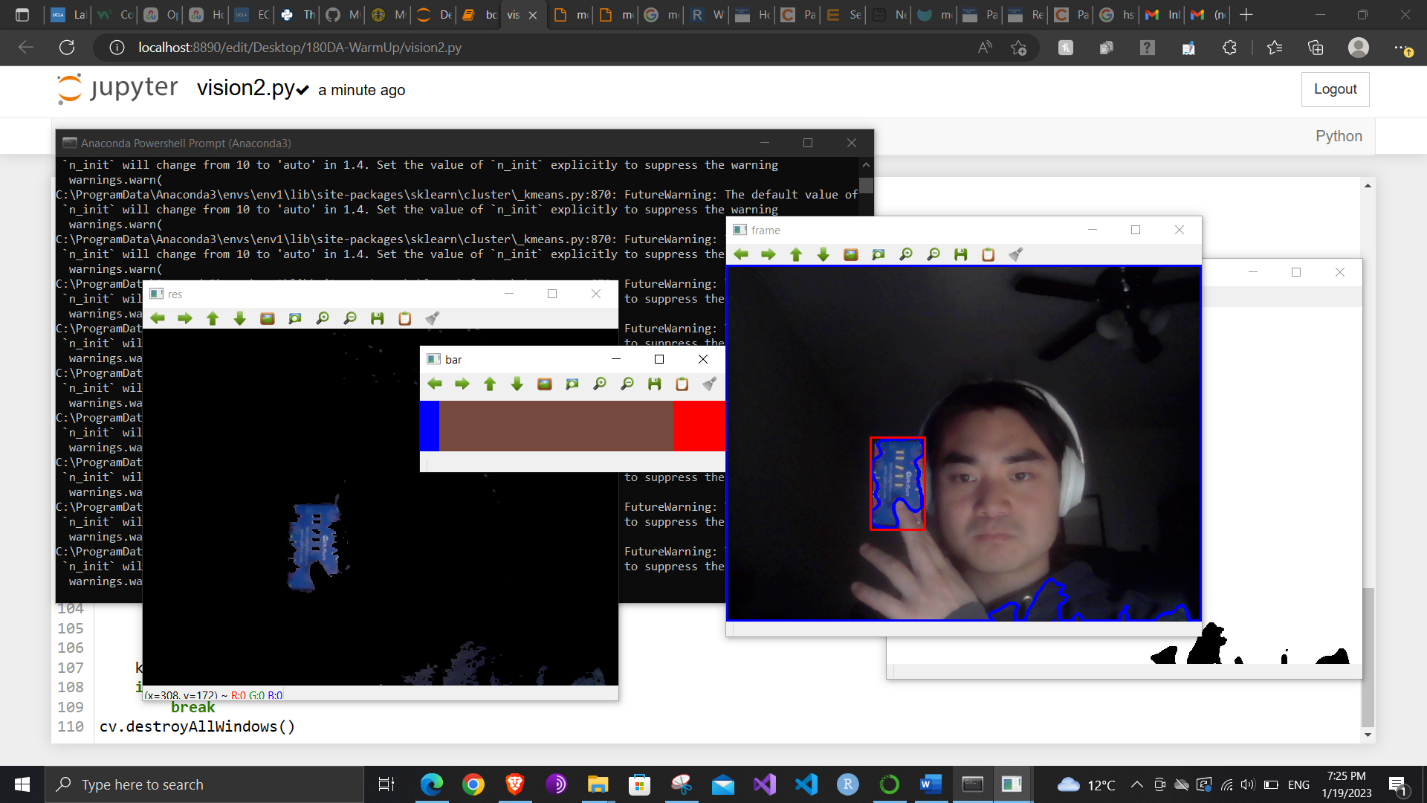
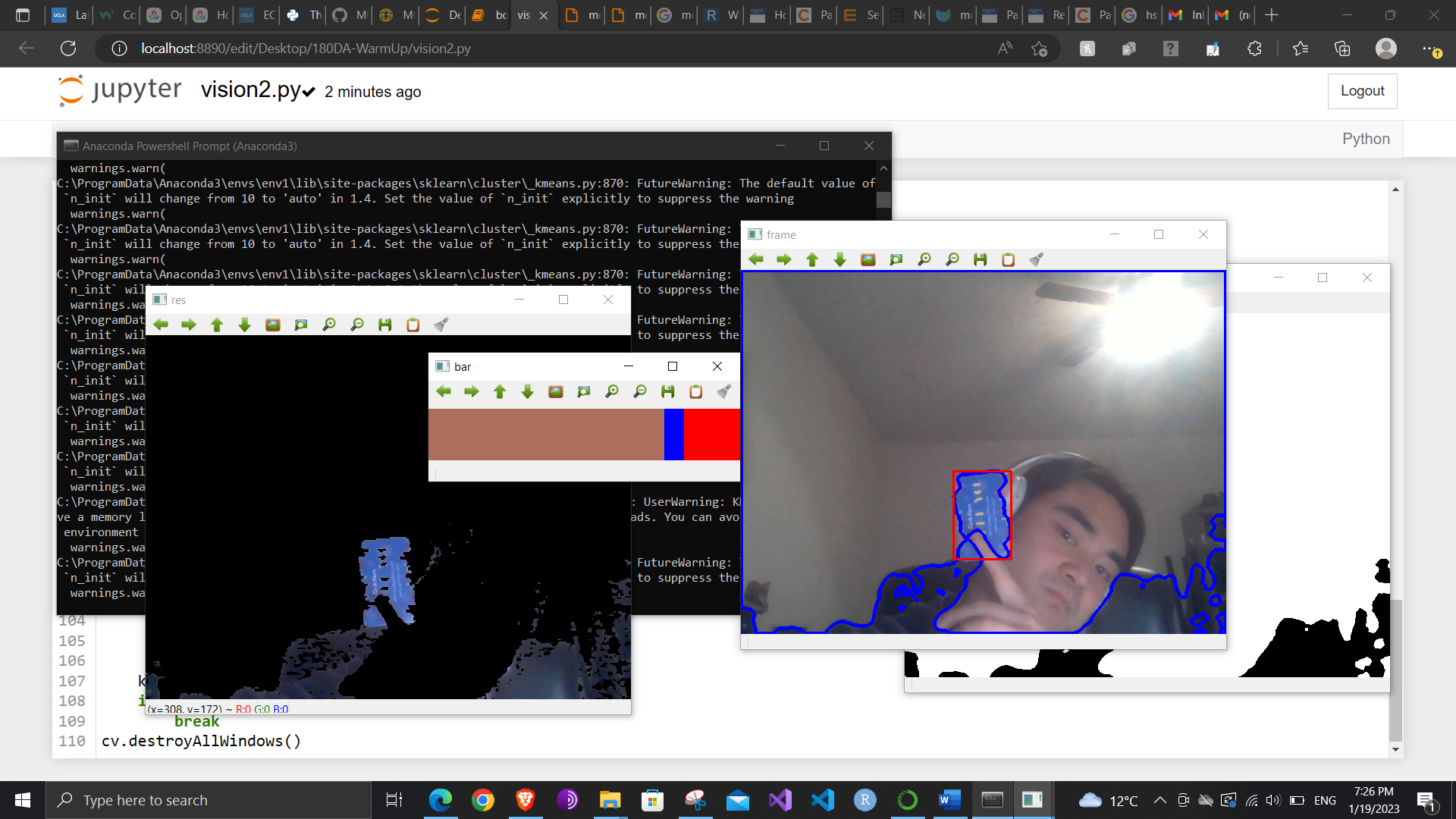
Below are the results when I expereimented with detecting a color on my phone. A color that can be detected when the screen brightness was turned to maximum could not be detected when the screen brightness is turned down to a normal level.



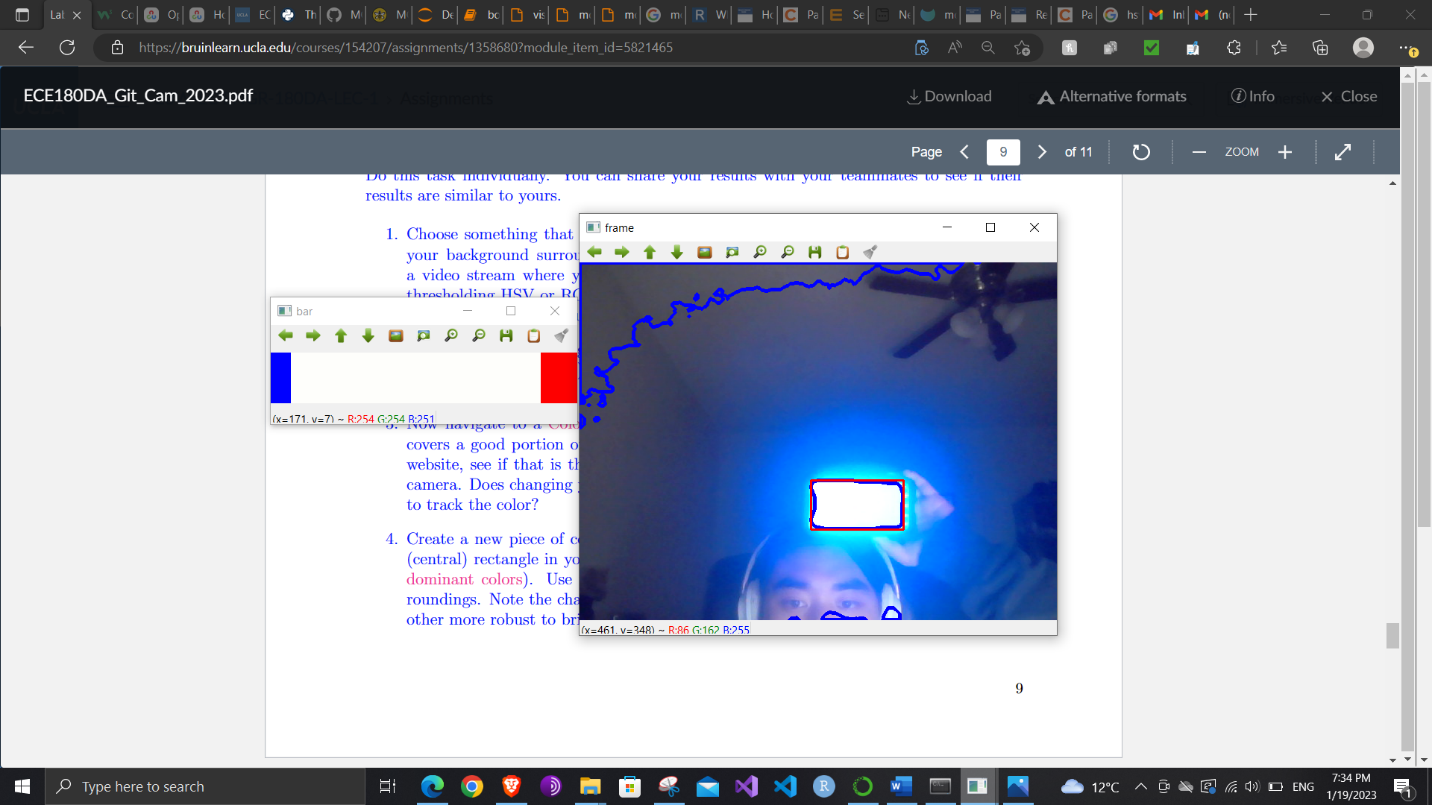
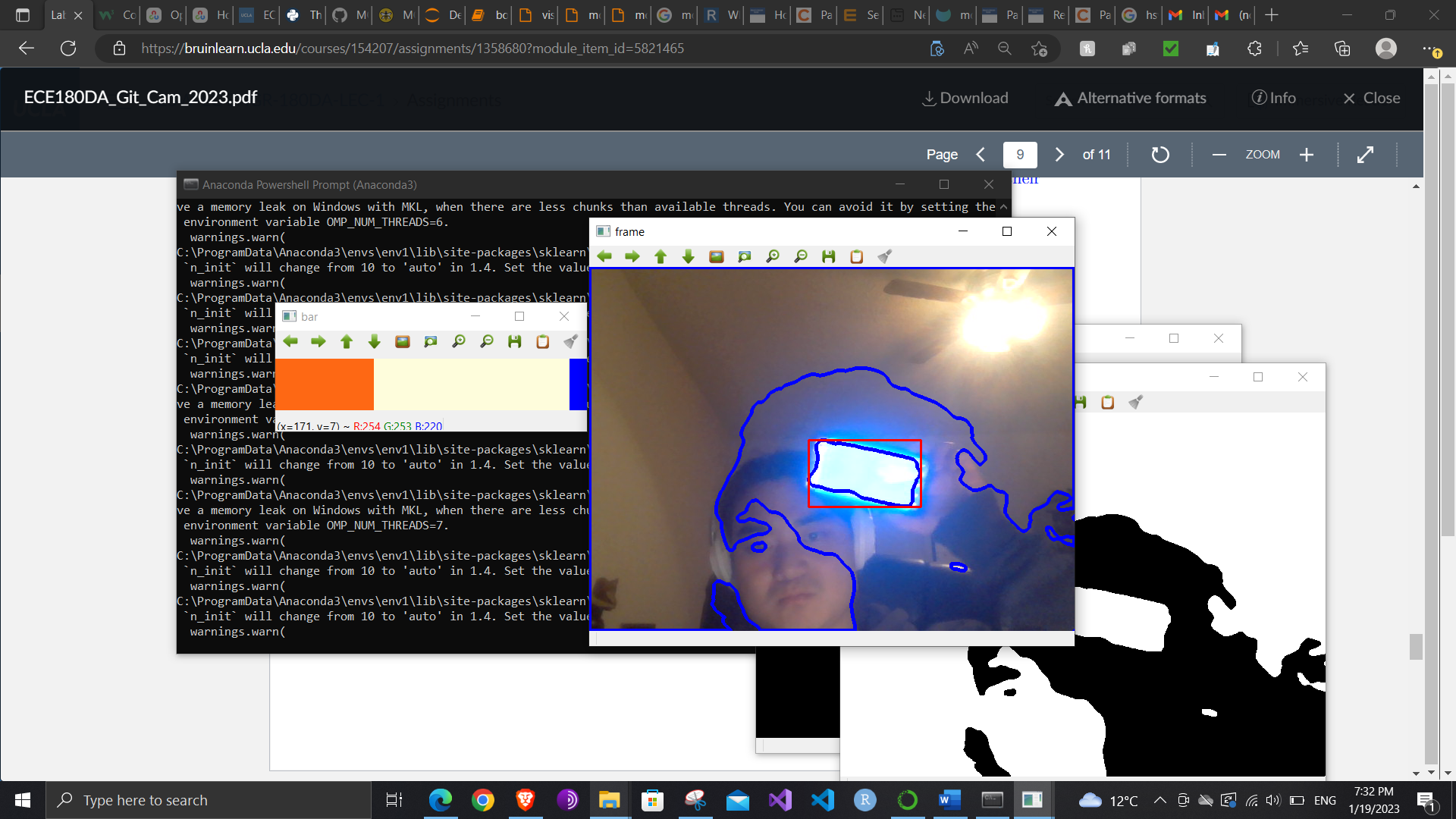


Task 4/4

Below are the color detection for the same blue card, with ceiling light on and off. The light on secenrio produced color composition that was lighter in cue.



Below are the same expereiemnt applied to the phone color screen. As you can see, under lighted conditions, the colors detected took on the orange hue of the background light, whereas under no backgrund light conditions, the colors detected are more pure white color emitted from the screen. The non-light emitting non phone object was more robust to change in light.



Conclusion

After the above expereiments, I have developed an object detection procedure for detection of a more or less single-colored object. I have discovered that the background brightness doesn’t necessarily affect the detection ability of my program, however, if the object I am tracking is light-emitting, then the brightness of the light from the object and backgrounf brightness does come into play. This is useful for the localization part for the project, where the camera can be used to detect the position of a person, based on perhaps the color of the person’s clothing, and calculate the position of the person to the camera by calculating the size of the detected person.