Final Project Write-Up
Authors: Kush Chopra, Bosit Anvarov, and Hanat Samatar
Date: 12/7/22

How to run the program:

1. Open the visual studio program
2. There are test images inside the folder
3. Pick any of them and write the image name (“example.jpg”)
4. Run the program
5. The result will be shown in the prompt, and will also be written as “output.jpg” inside the directory

For our project, we created a program for detecting road objects from an image source in order to intertwine our collective interest for technologies related to vehicles and traffic safety. (Traffic Light, Stop Sign, Pedestrian Sign, and Speed Limit Sign). We executed this idea with the thought that being able to detect and utilize its data to the driver's advantage would be a major convenience for the operator.

We were able to configure the road sign detection software to be able to detect signs and identify their purpose through image recognition. The detection and display of the road sign type was the hardest blocker to overcome solely due to the lack of data and training out there for american/local regulatory signs. To overcome this we trained our model using signs nearby through color coding. Initially we were only able to display red signs, but through further development we were able to implement more and more regulatory American signs with a variety of colors.

As a result of our program, we were able to display the road sign type. At the moment, we have been successful in detecting multiple traffic signs according to their HSV values; however, it does not fully specify and display the type of sign if there's multiple signs being processed for detection that are the same color (works if they are different colors).

Overall, we are satisfied by our results and clear understanding of Program's Two and Three to utilize as a main resource towards our project when referring to edge detection, pixel management, and overall handling of the program. In the future, we can broaden the program so that it is able to detect multiple signs in an image, the traffic lights (opencv YOLO), the number of pedestrians on the sidewalk, and the number of cars in a particular image/video using deep learning. We could also alter the program to detect cars changing lanes as a bonus to blind spot indicators. However these would all require more sources within machine learning.

Sources/References:

- **Program Two**
- **Program Three**
- **"C Program to Change RGB Color Model to HSV Color Model."** ***Tutorials Point*****, https://www.tutorialspoint.com/c-program-to-change-rgb-color-model-to-hsv-color-model.**
- **"Contour Approximation Method."** ***OpenCV*****, https://docs.opencv.org/3.4/d4/d73/tutorial_py_contours_begin.html.**
- **Hatim. "OpenCV - Contour Detection Using C++."** ***Another Techs*****, 7 Dec. 2022, https://anothertechs.com/programming/cpp/opencv/contour-detection/.**
- **"Mask Operations on Matrices."** ***OpenCV*****, https://docs.opencv.org/4.x/d7/d37/tutorial_mat_mask_operations.html.**
- **"Program to Change RGB Color Model to HSV Color Model."** ***GeeksforGeeks*****, 1 Aug. 2022, https://www.geeksforgeeks.org/program-change-rgb-color-model-hsv-color-model/.**
- **Yacine, Rouizi. "Edge and Contour Detection with Opencv and Python."** ***Edge and Contour Detection with OpenCV and Python | Don't Repeat Yourself*****, https://dontrepeatyourself.org/post/edge-and-contour-detection-with-opencv-and-python/.**