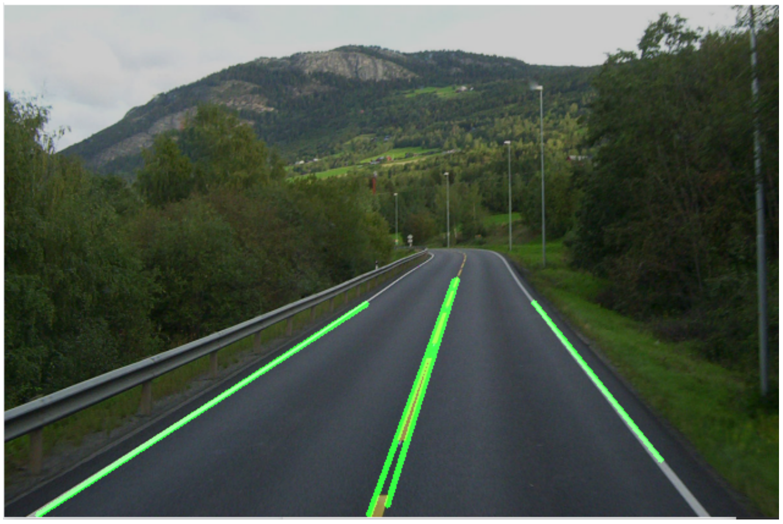
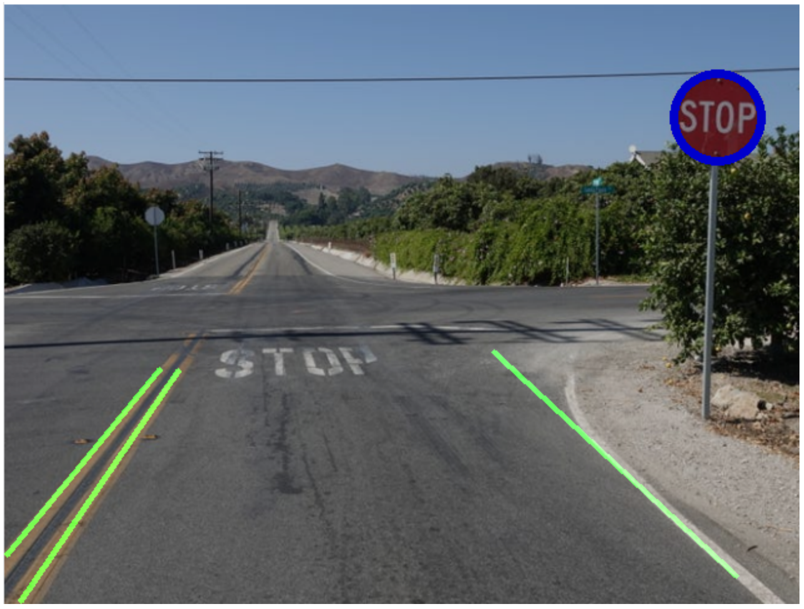
Ansheng Li
AXL170004

Project 1

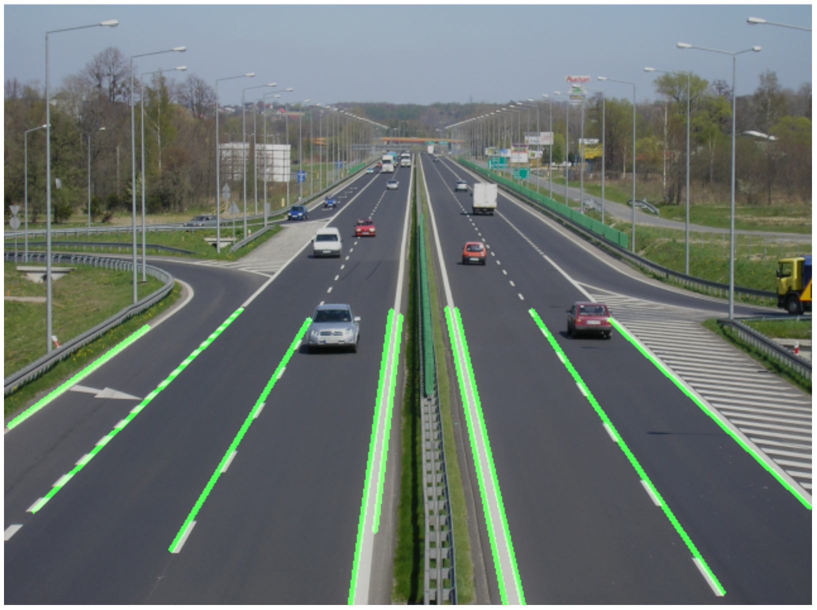
The pipeline for my approach was resizing the image, converting to gray, applying a mask for white, yellow, and red, blurring the image, applying a binary threshold, auto-generating edges using a canny edge detector, cropping lower half the image, and performing Hough transform. Because the process is generalized, there are some images that are processed better than others. Some challenges that still remain include merging of lines into a single line and picking up certain lines that get lost in the preprocessing pipeline. Some images produce more noise than others, and I used a median Canny edge detector which resulted in the reduction of noise. The reduction of noise is much greater than the default values but less than a tight range of (225, 250) of Canny edge detector. However, applying a threshold before hand produces no significant changes to the amount of edges produced.



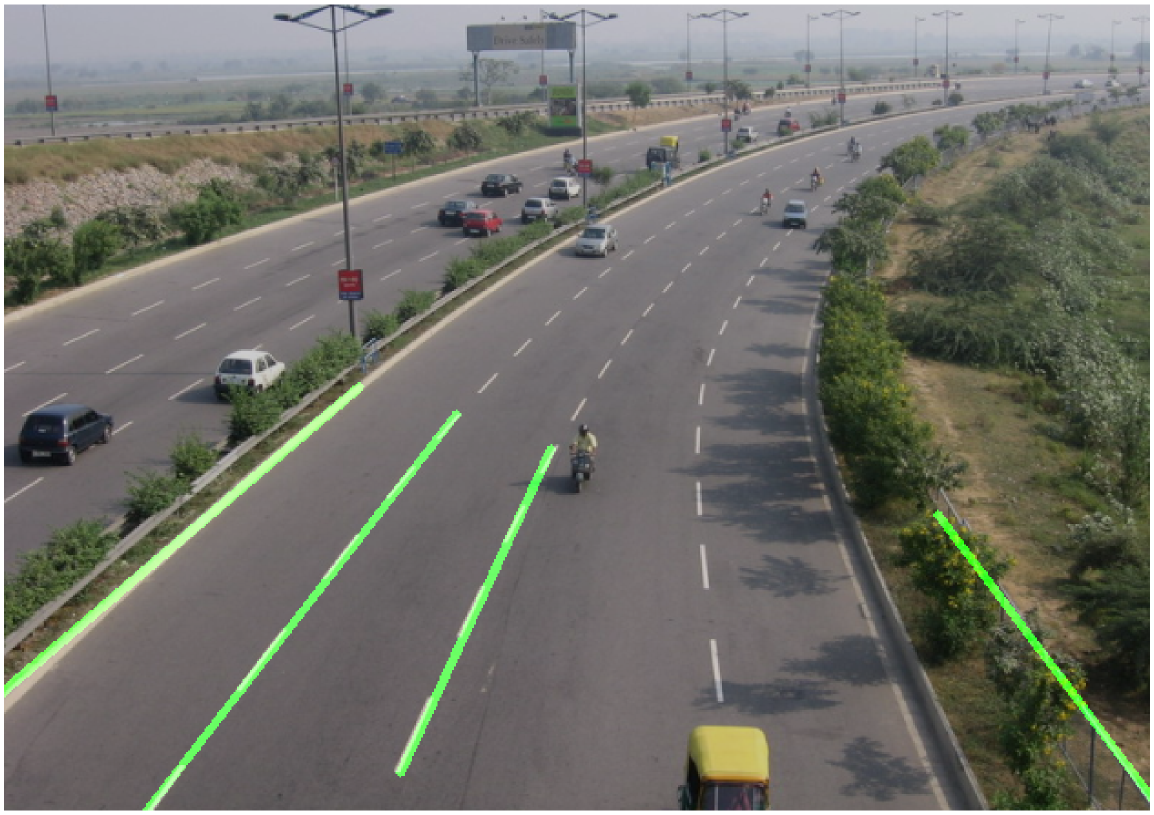
At first, the default values of the probabilistic Hough transform produced too many line segments. I increased the number of votes to 70 so that a line is only formed within a more dense area. The minimum line length is increased to 100 and max line gap is increased to 150 which would help create a longer line and combine segmented lines. There were many remaining lines generated by Hough transform and I reduced most of them by merging the ones with similar slope giving some degree of deviation. The ones that did not merge were the thicker lines that would produce double lines. In addition, curved roads are problematic in my detection since my parameters are looking for fairly long and straight lines. By turning down the threshold, min length, and gap size would most likely detect such curvature better.



The merging of the double lines does not work well for images that have a high slope since the closer it is to the vertical midpoint of the image, the slope increases at a much higher rate. In addition to the lanes, I was able to implement the detection of stop signs. I process the image again in a similar fashion but using a red mask only. The image would be better for detecting the octagonal shape of a stop sign. By finding the contour and detecting the number of sides, I was able to draw a circle with the given coordinates.



Another problem I encountered is the coloring of the image, specifically the color range of white. Since white is detected fairly easily, most of the noise I produce is from gray and white spots in the image. If the road is white-gray in color then many lines will be produced. I tried to mask out most of the white-gray spots but that removes smaller dotted line segments of the road. Therefore, my pipeline would no longer detect that lane.



Additionally, shadows disrupt the detection of lanes and long white areas are also considered as a lane line. The fence on the right produces a line as the image after processing shows a long set of points that resembles a lane line. While the inner lane on the right was not detected as the shadows from the tree removed the white mask from picking it up.

In conclusion, detection is based on the quality and angle of the image. While my values and threshold works for most images, there will be additional lines or missing lines due to the quality of noise detection. In this classical approach, it is difficult to perfectly process all types of images as different values and different parameters produce different results.