Finding Lane Lines on the Road

The goals / steps of this project are the following:

- Make a pipeline that finds lane lines on the road
- Reflect on your work in a written report

Reflection

1. Describe your pipeline. As part of the description, explain how you modified the draw_lines() function.

The pipeline basically follows the given steps from the introduction tutorial.

1. Takes the raw image and transform it to HSV using OpenCV to prepare for colour thresholding. An interesting problem encountered here was that OpenCV had both RGB and BGR. Using BGR would cause the hue values to change, but once the values are adjusted for this, the algorithm would produce the same results as when using RGB.
2. Colour threshold based on hue and saturation found using histograms plus some manual correction to narrow the ranges through trial and error.
3. Convert the resulting image to greyscale and smooth it to prepare for canny edge detection.
4. Perform Canny edge detection with parameters that are found based on trial and error.
5. Filter off non-lane marking regions by providing the four vertices of a trapezoid. Again, the vertices are determined based on visual approximation from output images. The four vertices' x-values are labeled X1, X2, X3, and X4 from left to right.
6. Run Hough line detection on the filtered image. Here, the default Hough line helper had to be modified to also return the list of lines so they can be converted to points for RANSAC.
7. Separate the points from the previous step into right and left lane markings using the algorithm described further below.
8. Run RANSAC on the list of points by picking a minimum of 2 points at a time to train using a maximum of 500 training cycles.
9. The left lane marking is finally drawn between (X1, RANSAC(X1)) and (X2, RANSAC(X2)) where RANSAC(X) = the y-value on the line of best fit. The right lane marking is drawn between x-values of X3 and X4. These x-values are those of the trapezoid's as defined in step 5.

1.1. Separate Points into Left and Right

1. Get the x-value for the half way point in the region of interest.
2. Get the (x, y) coordinates of the start and end of the line as provided by previous Hough line detection.
3. If the line is vertical (x1 == x2), then compare the x value with the half way point to determine if it is to the left or right.
4. Calculate the slope of the line using the standard slope calculation.
5. If the slope if positive, it's sloping from top left to bottom right, which is likely the right lane.
6. If the slope if negative, it's sloping from bottom left to top right, which is likely the left lane.
7. If the line is horizontal, then do the same as the vertical lines, but only compare the starting point with the half way point (since the x-values actually change for horizontal lines).

2. Identify potential shortcomings with your current pipeline

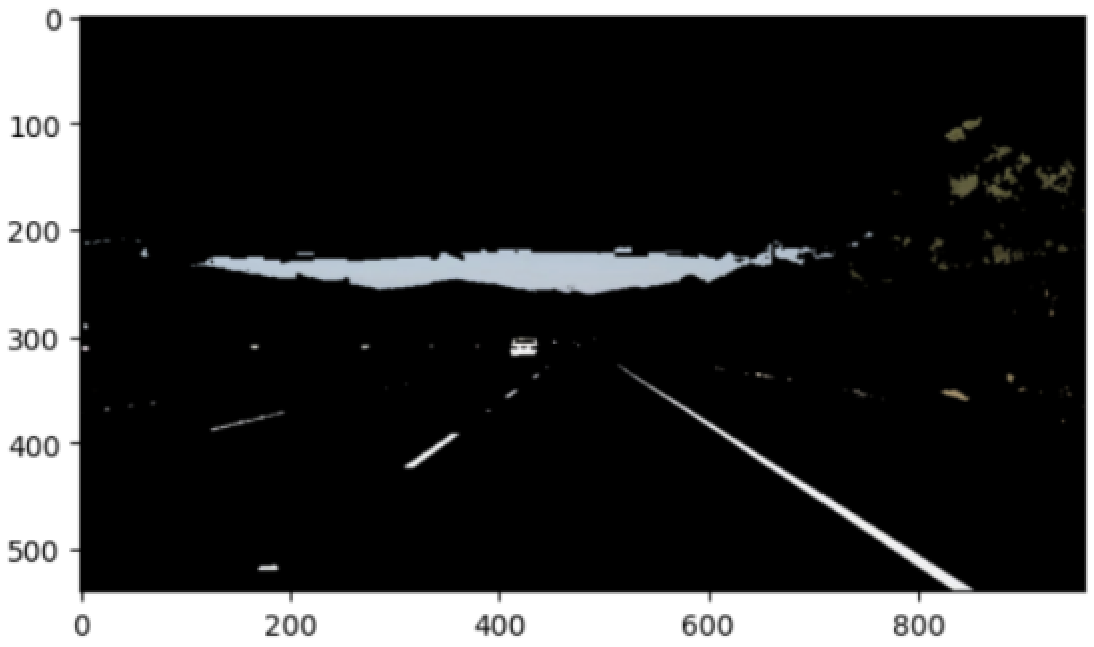
1. The colour masking is predefined to search for a range of qualifying colours. If cars, signs or foliage are a similar colour (and some are in the examples), they will also be left after the masking.

2. The region of interest masking is also predefined, which can include artifacts from items other than lane markings. This also can be an issue when the car is turning as the trapezoid will need to be skewed or shortened based on the angle of turn.

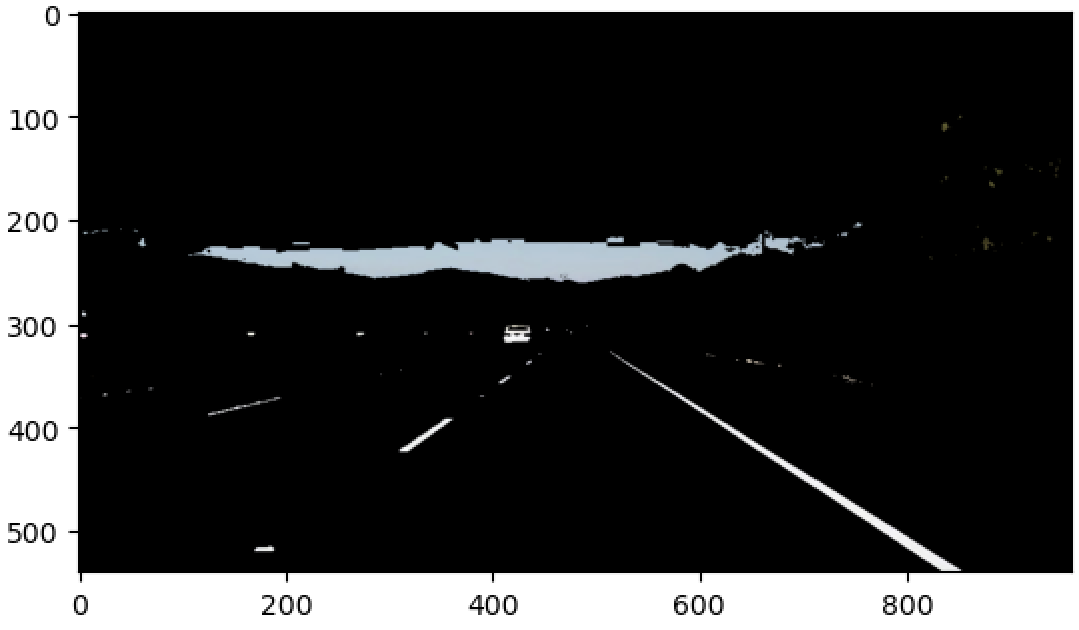
When the above issues are combined, we can foresee when a white or yellow car is really close to the ego vehicle, it can leave a huge blob in between lane markings. In this scenario, there will be lines outlining the area of the car, which will skew the RANSAC algorithm, causing the final lanes to be drawn skewed to the centre of the lane.

3. The left-right line separation algorithm doesn't always categorize the lines correctly. Noticeably, some lines from either lane can have the "wrong" slope because of perspective. This is especially true for the short horizontal lines on the ends of each marking.

2.1. Example Change Due to Colour Masking Parameters



The above image uses a larger saturation range for yellow. Compared to the image below which uses a larger saturation range, there are noticeably more trees and grass on the right side.



The same problem can happen for whites if parts of the road are of a lighter colour, such as in sections where the road is under repair.

3. Suggest possible improvements to your pipeline

If left and right lines are separated only using the centre x-value of the region of interest, it will not have the problem described in point 3 in the previous section. However, it will then be a lot more sensitive to the hand-picked trapezoid, which can bring about issues when doing tight turns.

The masking related issues can be prevented by adding more advanced algorithms designed to identify larger structures such as cars and signs. For example, after Hough line detection, we first separate the lines for cars, signs, and lane markings. We can then only examine lines from lane markings in the RANSAC algorithm.