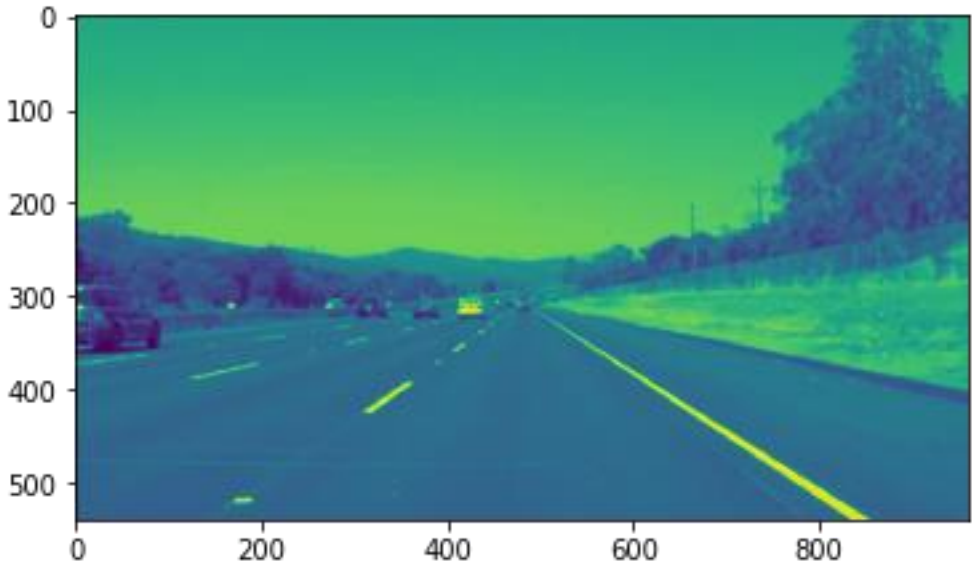
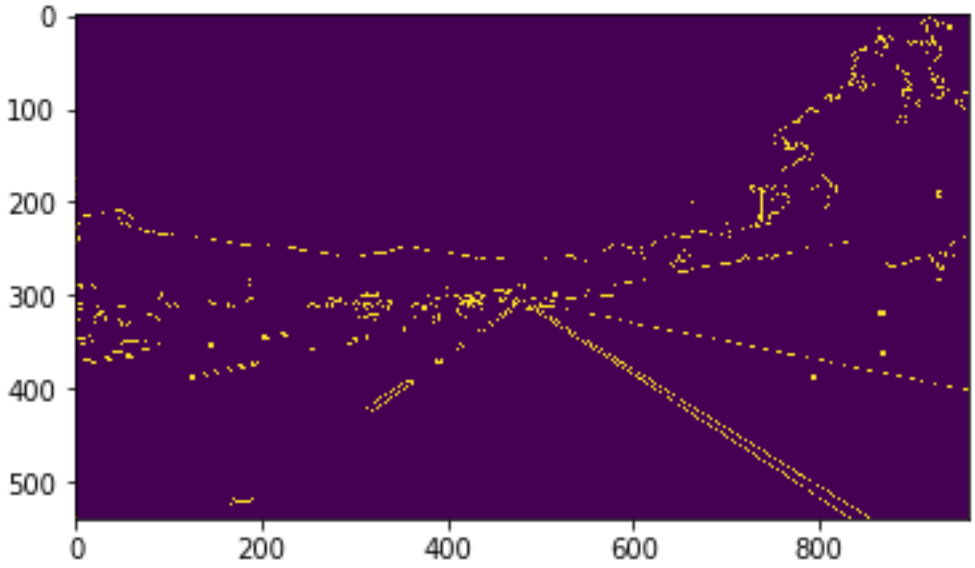
1. Describe your pipeline. As part of the description, explain how you modified the draw_lines() function.

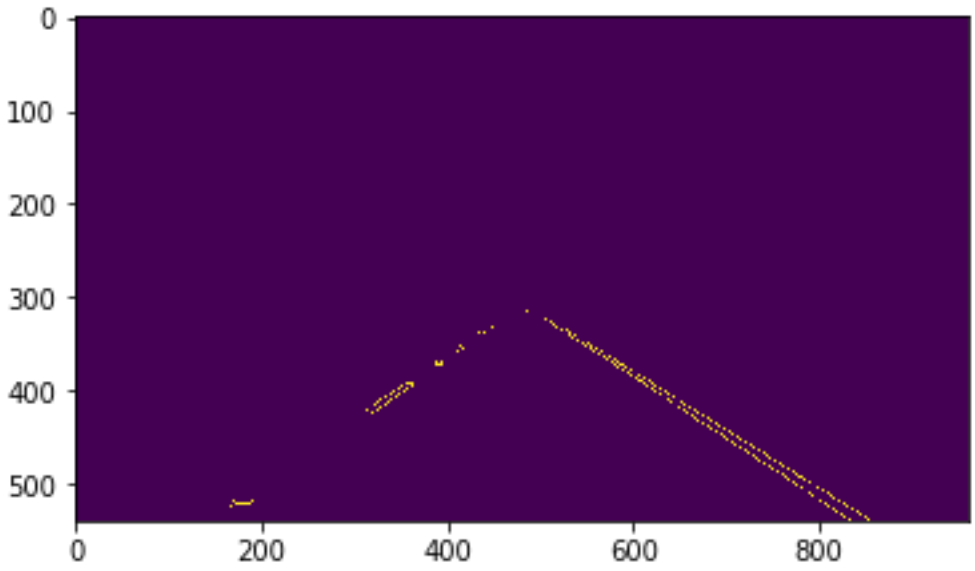
My pipeline consisted of 5 steps. First, I converted the images to grayscale to reduce the noises dimensions brought by RGB.



Then I applied Gaussian smoothing on the grayscale image to blur the borders between lanes and road in order to reduce unnecessary details and noises.



Next, I drew a four-sided polygon to mask the area where the lines are most likely falling under. This step enables the pipeline to focus on the area with lanes, and narrows down my interest of study.



After that, I applied hough transformation on the masked image to detect lines. As an output of this step, we could possibly observe discontinued lines on the left or right sides, if the lanes are doted. In the end, I draw the detected lines on top of the initial image to label out the detected lanes.

In order to draw a single line on the left and right lanes, I modified the draw_lines() function by adding an averaging and extrapolating mechanism. I first calculated the slope and intercept of the detected lines. The lines with positive slopes are the right lanes, while the ones with negative slopes are the left lanes. For each of the left and right group, I calculated the weighted average lane slope and intercept, with the weights being the length of the lines. The concluded slope and intercept for each group are the weighted averages.

For more complicated images, for example, the ones with the front of the car, shadows, or painted road signs included in the image, there could be unwanted lines detected in the masked area. In order to exclude those lines, I only included lines with certain slopes which are more probable to match the lanes.

When applying the pipeline to a video, the output can sometimes be flickering. In order to smooth out the output lanes between each output channel, I applied a moving average to the slope and intercept by giving certain weightings to the parameters in the previous channel.

2. Identify potential shortcomings with your current pipeline

One potential shortcoming would be what would happen when the car is making a sharp turn and the lanes are supposed to be very curved. Our pipeline can only produce straight lines as detected lanes.

Another shortcoming could be when there are multiple lanes, road curbs, shadows, painted road signs or even other cars in the image, the pipeline will encounter a hard time detecting which ones are the targeted lanes, and the result could jump around due to the noises.

Furthermore, the location of our masked area is image dependent. If the camera is placed at a different angle, the vertices of the mask could require further adjustment.

3. Suggest possible improvements to your pipeline

A possible improvement would be to add an automated detector for the vertices of the mask based on the shapes of the lines detected.

Another potential improvement could be to take into consideration the typical width of lanes, and find out the target lanes based on the relative positions of all detected lines.

Additionally, we can fit polynomials instead of straight lines to capture curved lanes in case of sharp turns.