Finding Lane Lines on the Road

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

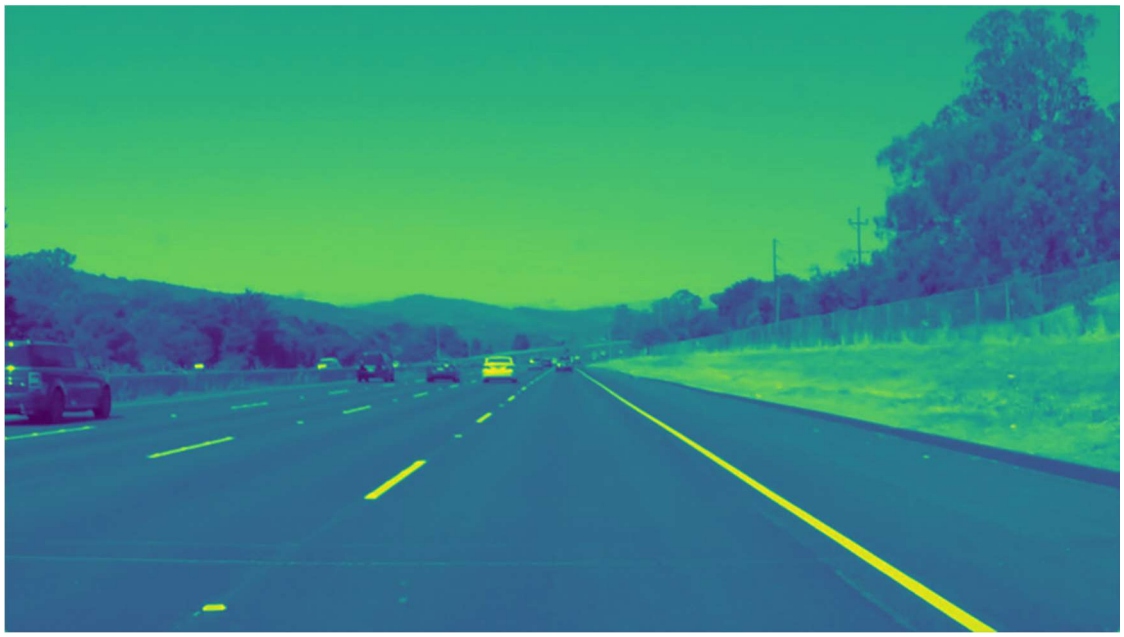
Reflection

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

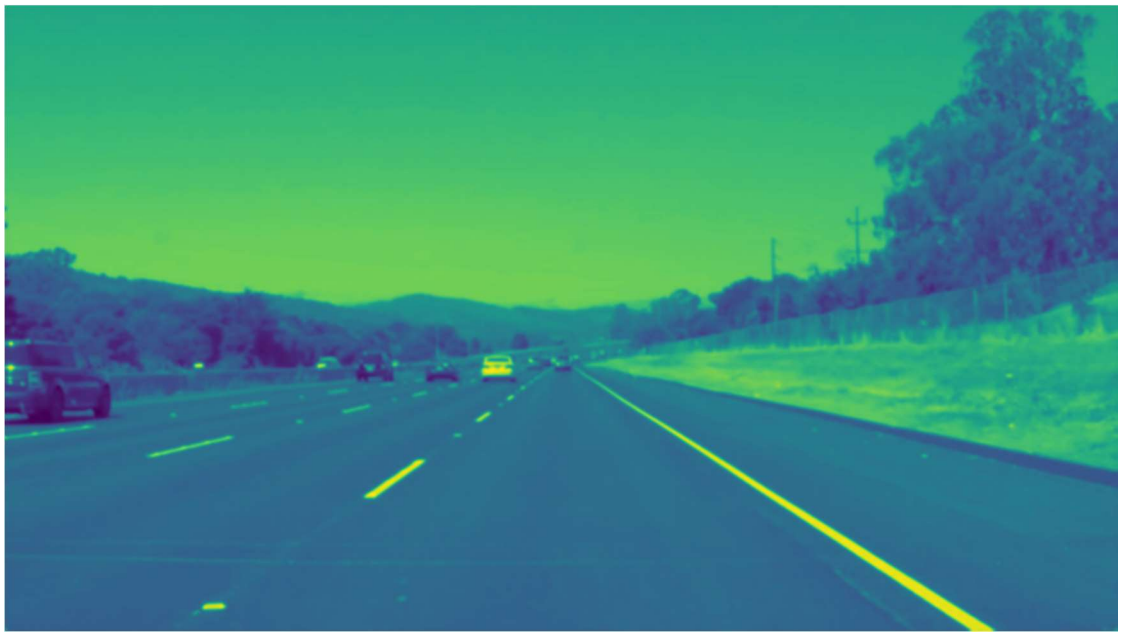
My pipeline consisted of the following steps:

1. Start with the original image

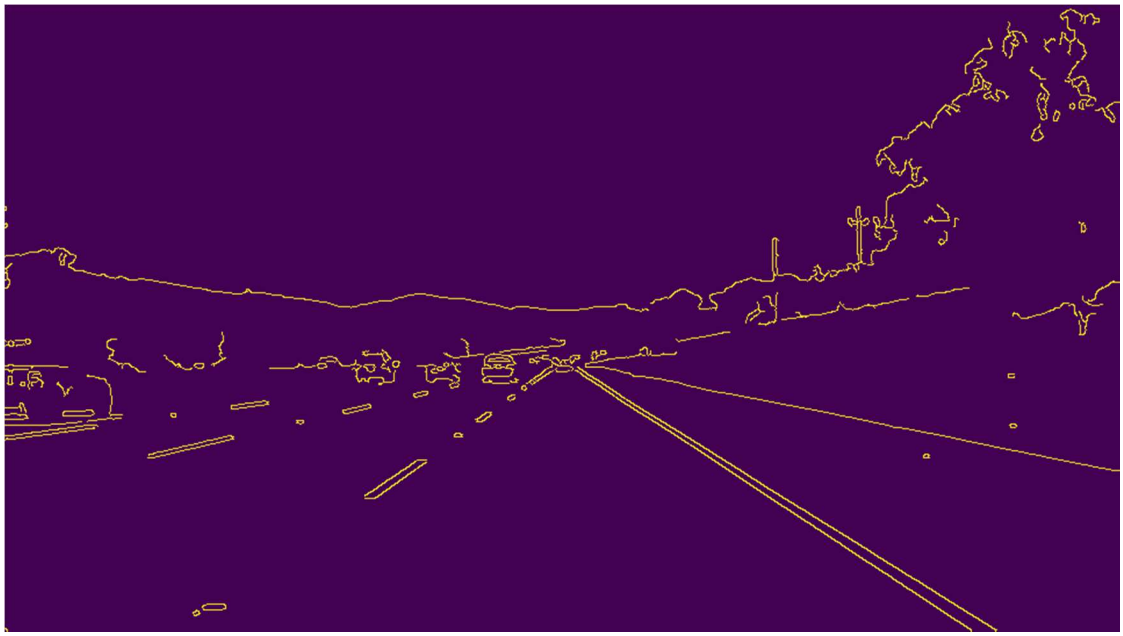
2. Convert the image to greyscale



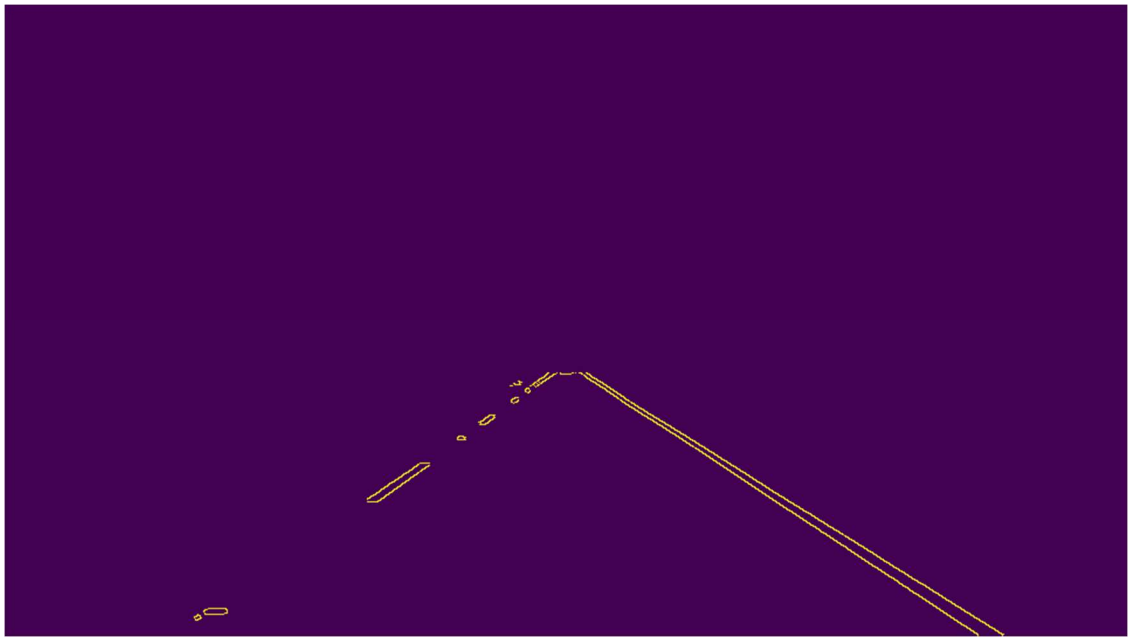
3. Apply Guassian smoothing (Gaussian Blur) to make the edges smoother and easier to detect.



4. Perform Canny edge detection to detect features within the image



5. Define the a Region of Interest within the edge image and apply a mask to the edges image:



6. Perform a Hough transform to detect lines from the edges image, extrapolate partial lane lines to cover the full lane length and draw straight lines to indicate lane boundaries within the image.

Multiple lines can be detected that make up a single lane boundary and some lane lines are only partially recognized by the draw_lines() function. In order to annotate each lane boundary as a single lane line, I modified the draw_lines() function to calculate an average line for the left and right side and extrapolate the line to cover the full lane length.

I also added a line sorting function to sort the lane lines by their slope – positive slope for the left lane and negative slope for the right lane.

2. Identify potential shortcomings with your current pipeline

The Optional Challenge video exposed several shortcomings of my program.

1) The region of interest boundary in my project was hard-coded meaning that the boundaries of my Region of Interest were calculated by using a fixed distance (number of pixel rows and columns) from the upper right corner. So the program will only work on images and videos that have the same resolution as the test images and videos.
2) The pipeline is written to work with straight lines and does not handle curved lanes lines as well, because the extrapolated line goes out straight ahead.
3) The pipeline does not account for rises and dips in the road where the region of interest would vary due to the changing camera perspectives.

4) The pipeline does not handle varying colors in the road well. When the camera passed over a light-colored section of road, the camera was not able to detect the yellow lane line in that section of road.

3. Suggest possible improvements to your pipeline

Possible improvements to my pipeline

1) Determine the region of interest relative to the bottom left corner rather than the top left (closer to the perspective of the car) and calculate Region of Interest vertices as a percentage of the total pixel rows and columns in order to make the pipeline work on images/videos of different sizes.
2) Detect curved lane lines and extrapolate left and right lines using curved overlays
3) Detect the perspective of view of the camera dynamically increase/decrease the height of the region of interest to account for changes in the camera perspective due to rises and dips in the road.