UDACITY : SELF DRIVING CAR NANO DEGREE - FEB 2017 BATCH

Lane Line Detection - Project 1

Prepared for: Udacity nano degree - Self Driving Cars

Prepared by: Sulabh Matele

February 23, 2017

Objective

Main object of this project are:

* Basic understanding about the lane detection concepts.
* Make a pipeline that finds lane lines on the road
* If the lane marks are broken then still making them as straight find line.
* Using this pipeline and applying to video so the lane line can be located in video as well.

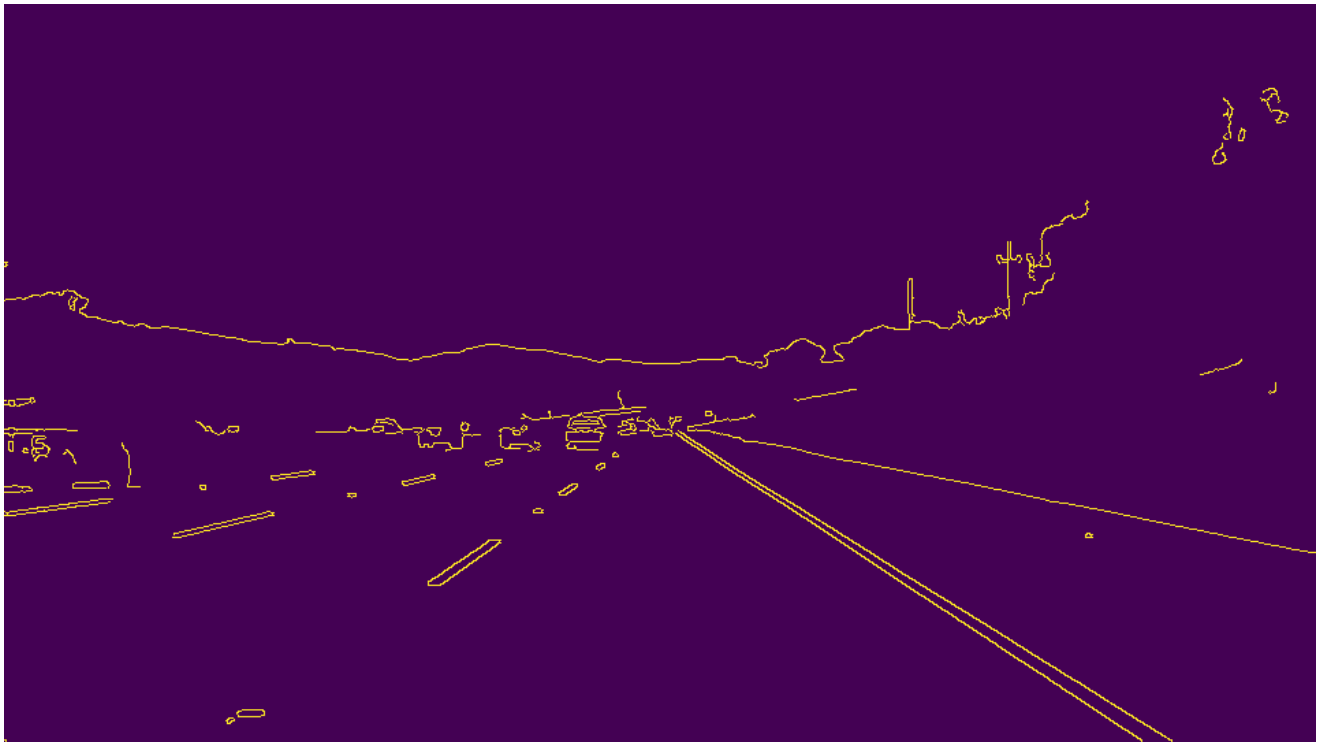
Project Work Outline

Different stages of image processing first making it gray, applying the Gaussian blur transform, detecting edges on the image by canny edge detection algorithm, applying region of interest to find interested region for lanes and then finally drawing lines on image by Hough lines.

Stage by stage some sample images:

Gray image:

Canny edge image looks like:



And to create the image as follows with lane line marked.

Short Description on different stages and challenges:

Each stage in the description represents the desired pipeline node:

Stage 1 :

This stage involves making the image to be processed, to Gray scale and this was an important step since lines could be of any color, and in gray scale we get the bunch of white, gray and black points which makes it easier to find the or detect lane lines of any color.

Stage 2:

This stage performs the Gaussian blur using open CV api. At this stage we try to suppress noise in the gray image.

Stage 3:

At this stage we use open CV Canny api to apply the Canny transform internally. We pass the low and high threshold for pixels to be filtered and we get the pure black image pixel boundaries traced out.

Stage 4:

At this stage we use the region of interest by defining a polygon boundaries, and then draw only the required Canny edges on an empty same size file, then just mask the resulted file with the original image to draw the detected lines.

Stage 5:

At this stage we apply Hough transform to find out the all points in the stage 4 result image, that could make a line with the given boundary which are passed as parameter to Hough open CV api. These parameter controls the accuracy by applying max length, min length threshold (max intersect points) etc.

Then the remaining work is to draw lines on the Hough points.

Challenges Involved:

After stage 5, we get the out put image as below, where we get the detected lines as segments.

For drawing line we need top and bottom coordinates of left and right lanes.

We know that slop = Change in height / Change in run

(We know Negative slop is Left and Positive slop is Right)

We apply the above formula to all the points which were the output of Hough transform and try to find slop or all the points, we also discard the points which are of very steep slop. And also locate center by:

Center = (x1 + x2) / (y1 + y2)

So now using line equation:

(x1 - x2) = m (y1 - y2)

We average all the points to get single top and bottom left and single top and bottom right.

And draw line accordingly, and apply the same steps when it runs on video as well, since video is collection of images.

Short comings:

One short coming could be that, currently it works only for the defined polygons, where I feel on the real road we might need something dynamic which can adjust the polygon by sensing road.

Another short coming here is, that currently it does not work well on curves.

Ofcourse we have not covered n number of other use cases which might happen with the real car, like snow, roads without lane lines etc.

Improvements:

Possible improvement could be to make this pipeline more accurate and working with curves, night, low light etc.