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

1) **Code:** The Jupyter notebook "P1.ipynb" is included in the zip folder attached with the submission. The various steps of the pipeline are described in the notebook.

2) **Reflection:**

1. **Description of the pipeline:** The pipeline consist of 5 steps.
   - I) First, the images were converted to grayscale. A kernel size of 5 is used and the Image is then blurred using Gaussian blur.
   - II) Canny edge detection is then used to detect lines in the images. I have selected optimal thresholds for edge detection as follows
     a) First computed the **mean** of the gray scale image.
     b) Chosen two values (lower and upper thresholds) based on the mean value of the gray scale image.
   - III) Defined the four sided polygon mask to concentrate on the near part of the road while detecting the lane lines.
   - IV) Hough transform algorithm is applied on the masked image to find parallel lines in the image. The various parameters are set and all the lines are obtained.
   - V) Finally the right and the left lane lines are identified and drawn using the draw_line() function which works as follows.
     a) It keeps a tract of the frame number and the slope of the line drawn in the last 4 frames.
     b) The slope of all the parallel line segments are calculated and depending on their orientation are classified into left lane and right lane.
     c) An average position of all the line segments in each lane is taken and the lines with unacceptable slopes are ignored.
     d) If none of the slopes are acceptable in a frame, data from the previous frame is used to predict the slope of the lanes in that frame.
     e) The slopes and intercepts for a frame are averaged over the previous 4 frames to get rid of noises.
     f) After obtaining the slopes the lines are extrapolated and drawn on the image.
     g) Weighted_img is used to induce transparency in the image.

2. **Potential shortcomings:** The various possible shortcomings are as follows.
    I) The Kernel size for blur and the thresholds for the canny edge detection can be optimized further. There are chances we might not detect lanes if lane lines are not clearly visible in the photo, either they have been washed off or due to bad lighting and weather conditions.
    II) The dimensions of the Mask are hardcoded.
    III) The values set for the parameters used in the Hough transformation are fixed and can be further optimized.
    IV) The parallel lines are classified and based on hard coded slopes. And the lines are fit (1^{st} order pollyfit), and the road can curve significantly.
    V) There can be problems at the exit were diverging lines can be filtered.
    VI) I have used data from only 4 previous frames to filter the noises in the line segment detected, which might not smooth the lines.

3. **Possible improvements:**
    I) A possible improvement can be using median instead of mean in the canny edge detection. Hyper parameters can be optimized to make sure we don't miss edges in various rough conditions
    II) The Mask can be used in a smarter way instead of being hard coded.
    III) Hough transformation parameters can be further optimized.
    IV) More intuitive may can be used to classify the line segments and we can even fit a higher degree curve instead of a line.
    V) Higher order filtering can be done depending on the processing recourses available. But there are chances we can filter useful information.