Finding Lane Lines on the Road

- Swaroop K.

Reflection

Following are the steps I followed to detect lanes:

1. Make a gray scale copy of the given image.
2. Smoothen the gray scale copy using **gaussian_blur()** with **kernel size 3**.
3. Detect the edges using the **canny()** function with thresholds (100, 300).
4. Cut out the area outside a trapezoid with vertices: (50, image.y), (image.x/2.4, image.y/1.6), (image.x/1.7, image.y/1.6), (image.x-50, image.y).
5. Find lines among the edges using the **hough_lines**() with following parametes:
   rho = 3, theta = np.pi/180, threshold = 1, min_line_len = 2, max_line_gap = 3

Continuous lane markings were detected as follows:

1. On the image obtained from hough_lines() function, for each 'x' find all values of 'y' that has a non-zero value.
2. Average the 'y' values thus found for each 'x'. This gives a list of points along the lane markings.
3. Interpolate these points using **np.polyFit()** to obtain the (m,b) parameters for the left and right lanes.
4. Draw lines using these parameters on the image with desired color and line width.

Shortcomings

- Manual cropping is required to eliminate possible edge cases.
- Manual tuning of parameters is needed.
- Fails of the lane markings are not sufficiently distinct from the tarmac.
- Doesn't distinguish color, solid vs punctuated lines.

Improvements

- Utilize lane colors for lane detections.
- When lane markings merge with tarmac color, estimate lane marking based on good detection from recent frames.