Reflection

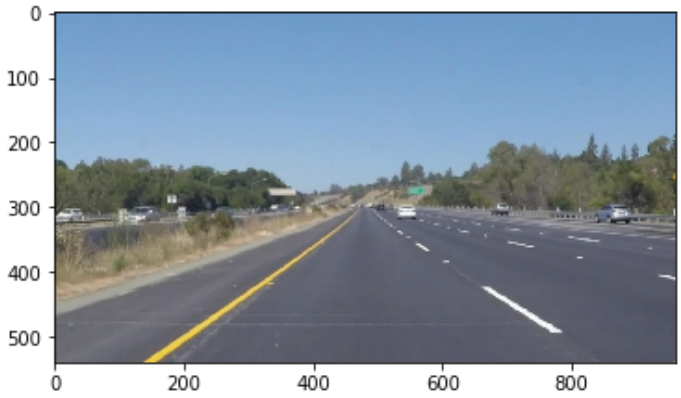
For the better test images and test videos

My improved pipeline(a series of image processing) consists of 6 steps:

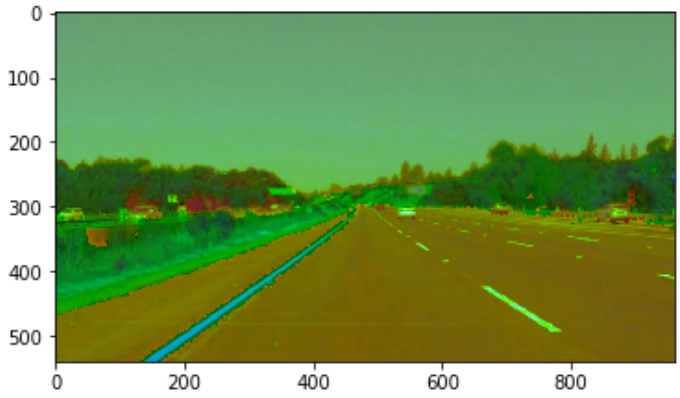
1) In order to detect the edges, converted the image(RGB color) to grayscale color;
2) In order to the image smoother, applied Gaussian Blur to grayscale image, the parameter of kernel_size is 5;
3) In order to find straight lines, applied the Canny algorithm to the 2), the parameter of low_threshold is 110, the parameter of high_threshold is 150;
4) Defined a four sided polygon to mask the left and right lane lines, vertices is [(0,imshape[0]), (450, 290), (490, 290), (imshape[1],imshape[0])]
5) Used the Hough line transform to detect lane lines in the edge images, the parameters:
   rho = 2, theta = np.pi/180, min_line_len = 30,
   in particular, threshold = 110, the higher the value of threshold, the less the number of lane lines detected by Hough algorithm.
   max_line_gap = 150, due to there are dotted lines in the road, so adjust high the value of max_line_gap.
6) In order to draw a single and solid line over left and right lane lines, I modified the draw_lines() function with draw_lines_improve() function,
   Firstly, detected the left and right lane lines by slope((y2-y1)/(x2-x1)), and store the points;
   Secondly, got the slope value and intercept value by the polyfit() function(least squares polynomial) with points, then draw the lines.

For Optional Challenge

The main challenge of video is **shadow**, which caused the above algorithm cannot accurately detect the yellow and white lane lines, so how to solve it, it is the key point.
The solution: converted the image(RGB color) to HLS(hue, saturation, lightness) color model. In the HLS model, the yellow is the blue, the white is green, then detect the yellow and white lines.



RGB model



HLS model

potential shortcomings with my current pipeline

One potential shortcoming is that it cannot get the coordinate value from the left line in quite exceptional circumstances, we can see it by the output of right_lines_x and right_lines_y, for example:

```
left_lines_x =  [258, 514, 276, 684, 298, 684, 274, 613, 264, 492, 283, 682]
left_lines_y =  [683, 511, 687, 402, 654, 403, 687, 450, 678, 524, 685, 395]
right_lines_x =  [726, 860, 1034, 1092]
right_lines_y =  [457, 541, 641, 674]
left_lines_x =  [258, 496, 276, 618, 393, 570, 278, 367, 269, 627]
left_lines_y =  [683, 523, 687, 448, 591, 476, 687, 623, 687, 446]
  2%|▌          | 4/251 [00:00<00:13, 18.60it/s]
right_lines_x =  []
right_lines_y =  []
left_lines_x =  [279, 625, 368, 591, 285, 584, 260, 418, 269, 628]
left_lines_y =  [687, 445, 608, 463, 685, 468, 684, 574, 687, 445]
  2%|▌          | 6/251 [00:00<00:13, 18.58it/s]
```

but the status does not happen in the right line, I do not know how it happen.

Suggest possible improvements to the pipeline

- Image from an infrared camera.
- Adding an outlier reduction approach like RANSAC on the hough lines.
- Using curve fitting to plot the curve instead of straight lines