James Vaughn – Lane Lines – May 29, 2017

Jvaughn0822@gmail.com

Finding Lane Lines on the Road

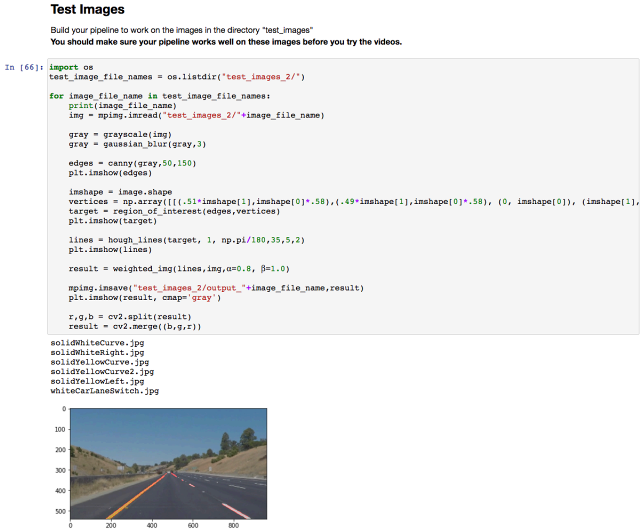
The goals / steps of this project are the following:
* Make a pipeline that finds lane lines on the road
* Reflect on your work in a written report

Reflection

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

Pipeline consisted of 8 steps.

1) I read each image
2) Converted RGB to grayscale using grayscale function; single channel
3) Applied Gaussian blur to reduce contrast noise
4) Applied canny to find dotted-edges
5) Applied a mask to isolate the road lanes; set vertices at bottom left/right and slightly above mid-line
6) Applied Hough lines to draw lines (e.g. connected dotted-edges)
7) Adjusted line weight
8) Reversed RGB to BGR to show correct image color
9) Saved images with new name → "outcome_"+file



Test Images

Build your pipeline to work on the images in the directory "test_images"

You should make sure your pipeline works well on these images before you try the videos.

In [66]:
```
import os
test_image_file_names = os.listdir("test_images_2/")

for image_file_name in test_image_file_names:
    print(image_file_name)
    img = mpimg.imread("test_images_2/"+image_file_name)

    gray = grayscale(img)
    gray = gaussian_blur(gray,3)

    edges = canny(gray,50,150)
    plt.imshow(edges)

    imshape = image.shape
    vertices = np.array([[(.51*imshape[1],imshape[0]*.58),(.49*imshape[1],imshape[0]*.58), (0, imshape[0]), (imshape[1],
    target = region_of_interest(edges,vertices)
    plt.imshow(target)

    lines = hough_lines(target, 1, np.pi/180,35,5,2)
    plt.imshow(lines)

    result = weighted_img(lines,img,α=0.8, β=1.0)

    mpimg.imsave("test_images_2/output_"+image_file_name,result)
    plt.imshow(result, cmap='gray')

    r,g,b = cv2.split(result)
    result = cv2.merge((b,g,r))
```

```
solidWhiteCurve.jpg
solidWhiteRight.jpg
solidYellowCurve.jpg
solidYellowCurve2.jpg
solidYellowLeft.jpg
whiteCarLaneSwitch.jpg
```

In order to draw a single line on the left and right lanes, I modified the draw_lines() function by …

1) #iterate through each line points (x1,y1,x2,y2)

2) # calculate slopes and centers
   #slope = (y2-y1)/(x2-x1)
   #center = [(x2+x1)/2,(y2+y1)/2]

3) #separate right and left by measuring slope

4) #if slope is between 0 and 0.6 then put the slope and center value in the left list

5) #if slope is between 0 and -.5 then put the slop and center values in the right lists

6) #average over all right/left center, slope values to get single center and slope

7) #find new (x1,y1,x2,y2) from the center and slope values and fitting to bottom of image and the specified height

2. Identify potential shortcomings with your current pipeline

- Images must be the same size, can be problematic
- Images must be of sufficient resolution
- Road lines may shift as car travels uphill, around curves
- Not all roads have line markers

3. Suggest possible improvements to your pipeline

- Add a dynamic area of interest
- Use object detection to interpolate road position