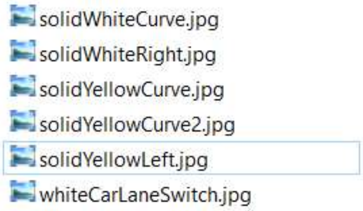
Finding Lane Lines On Road

In this Project our main goal is detect and show lane lines on road.

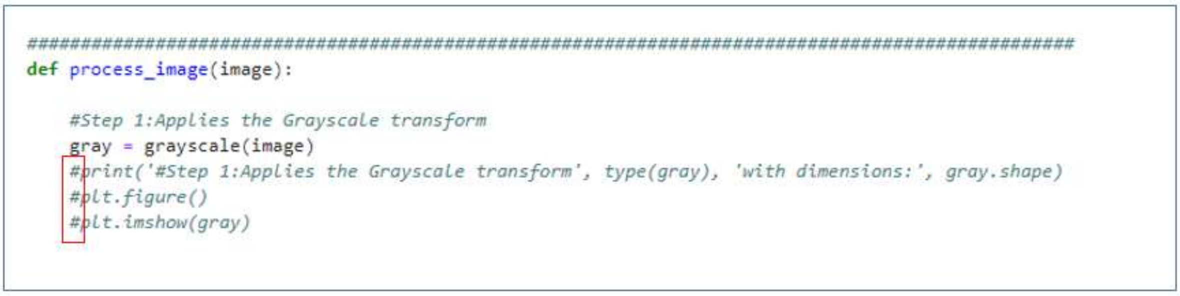
The template images:



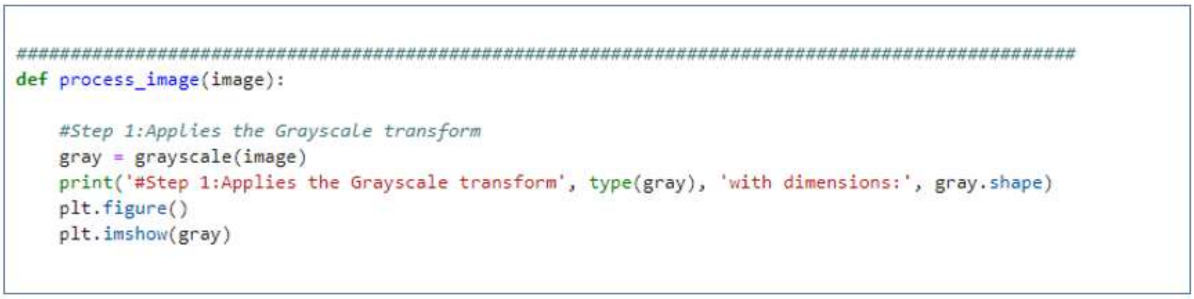
```
solidWhiteCurve.jpg
solidWhiteRight.jpg
solidYellowCurve.jpg
solidYellowCurve2.jpg
solidYellowLeft.jpg
whiteCarLaneSwitch.jpg
```

I have 6 different steps to find the left and Right trafic road lan eline.Each steps result can be seen with deactivate the # parameters from imshow() functions.

Example:



```
####################################################################################################
def process_image(image):

    #Step 1:Applies the Grayscale transform
    gray = grayscale(image)
    #print('#Step 1:Applies the Grayscale transform', type(gray), 'with dimensions:', gray.shape)
    #plt.figure()
    #plt.imshow(gray)
```

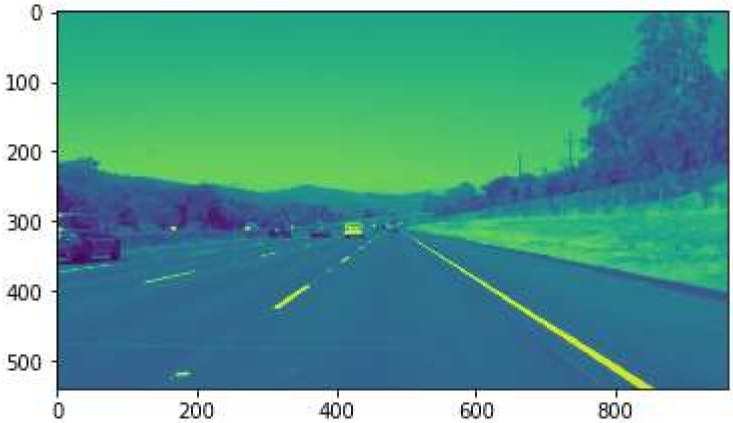


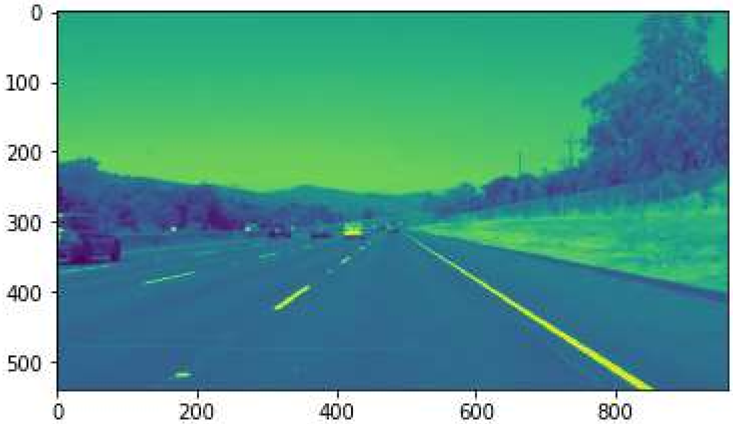
```
####################################################################################################
def process_image(image):

    #Step 1:Applies the Grayscale transform
    gray = grayscale(image)
    print('#Step 1:Applies the Grayscale transform', type(gray), 'with dimensions:', gray.shape)
    plt.figure()
    plt.imshow(gray)
```

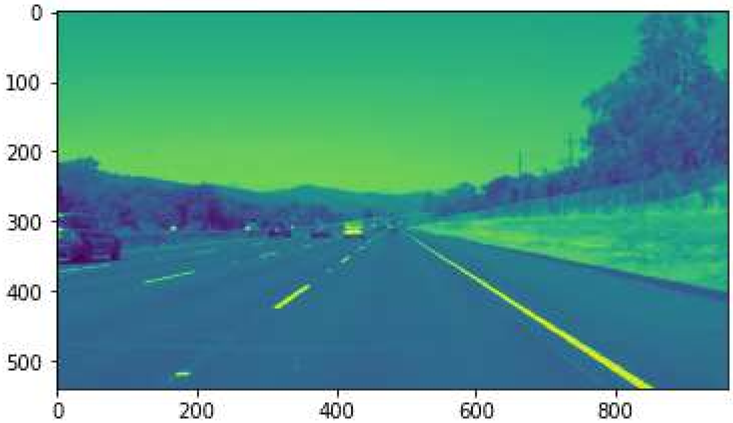
We have def process_image(image): function to process image from video input file.

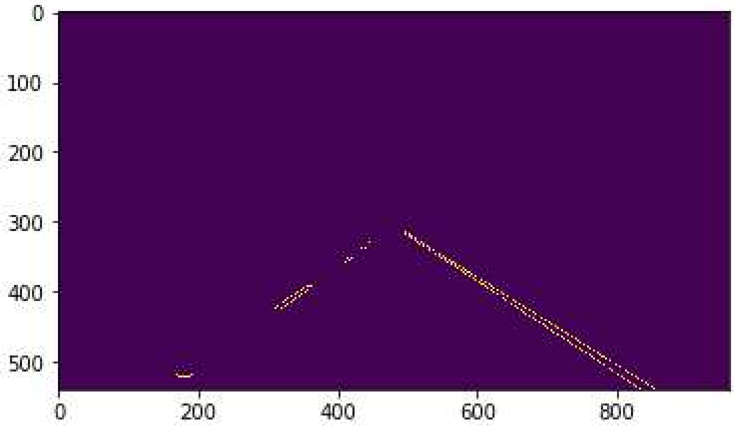
```
####################################################################################################
def process_image(image):
```

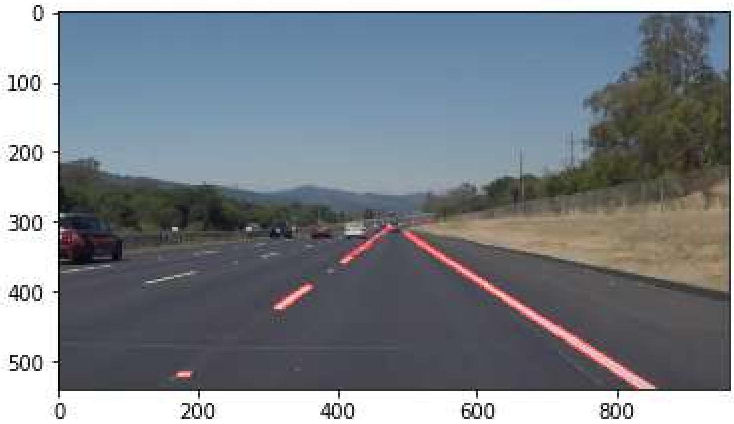
#Step 1:Applies the Grayscale transform <class 'numpy.ndarray'> with dimensions: (540, 960)
#Step 2:Applies the Gaussian Blur transform <class 'numpy.ndarray'> with dimensions: (540, 960)
#Step 3:Applies Canny transform <class 'numpy.ndarray'> with dimensions: (540, 960)
#Step 4:Mask image with triangular shape <class 'numpy.ndarray'> with dimensions: (540, 960)
#Step 5:Applies the Hough transform <class 'numpy.ndarray'> with dimensions: (540, 960, 3)
#Step 6:Creating a Single Left and Right Lane Line <class 'numpy.ndarray'> with dimensions: (540, 960, 3)

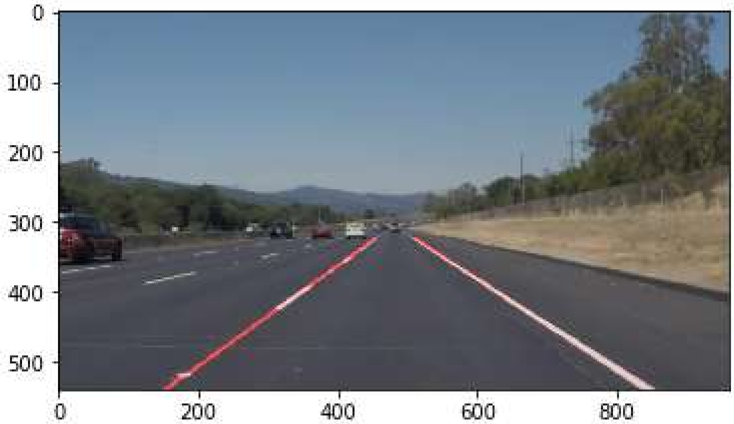

0
100
200
300
400
500
0
200
400
600
800


0
100
200
300
400
500
0
200
400
600
800

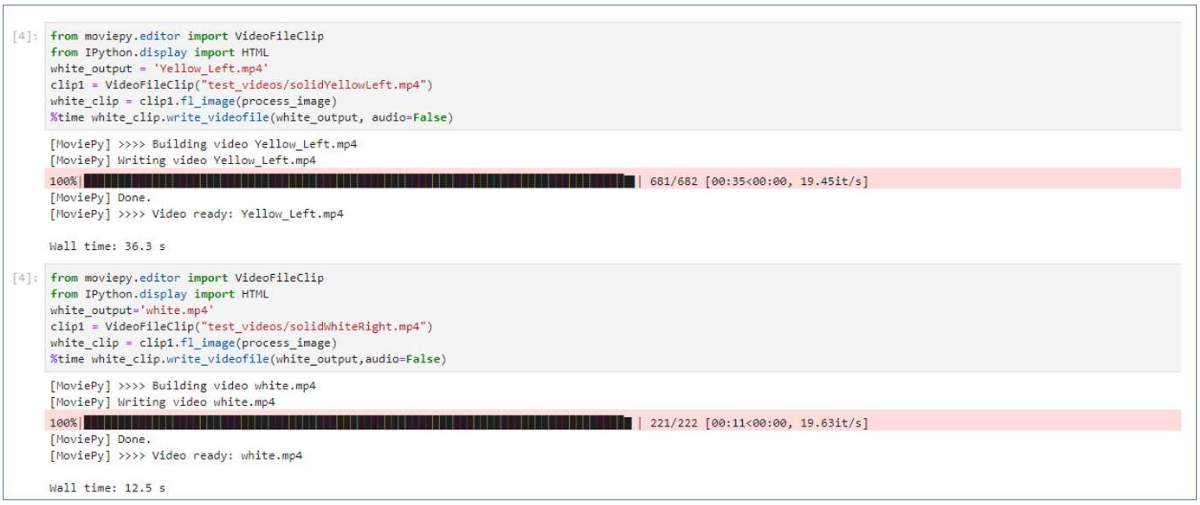

0
100
200
300
400
500
0
200
400
600
800


0
100
200
300
400
500
0
200
400
600
800


0
100
200
300
400
500
0
200
400
600
800

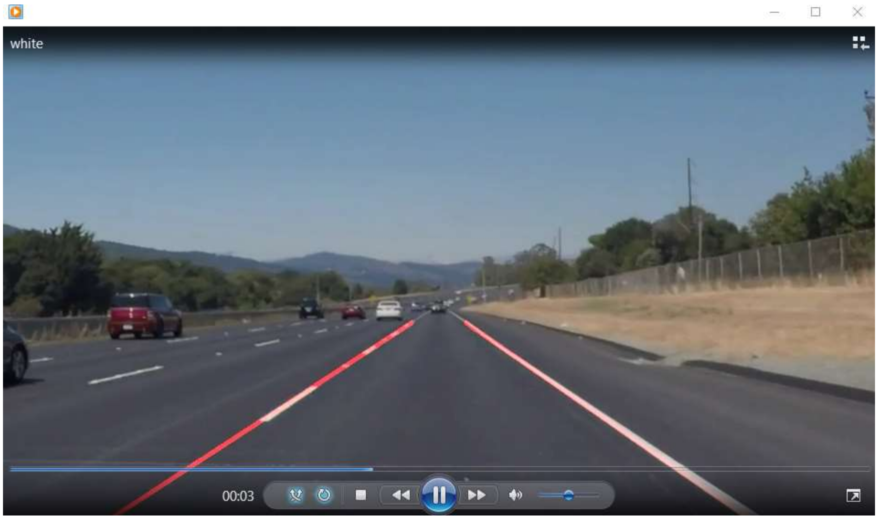

0
100
200
300
400
500
0
200
400
600
800

Video generation output:

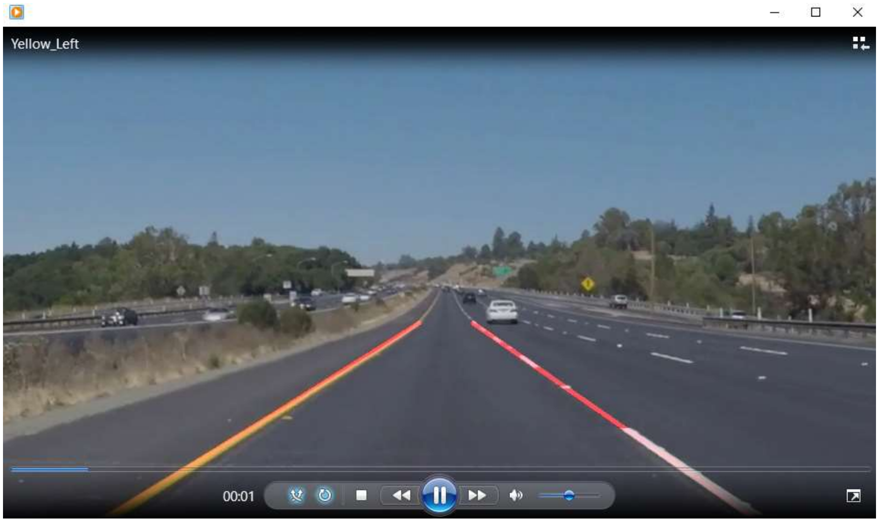


```
[4]: from moviepy.editor import VideoFileClip
     from IPython.display import HTML
     white_output = 'Yellow_Left.mp4'
     clip1 = VideoFileClip("test_videos/solidYellowLeft.mp4")
     white_clip = clip1.fl_image(process_image)
     %time white_clip.write_videofile(white_output, audio=False)

     [MoviePy] >>>> Building video Yellow_Left.mp4
     [MoviePy] Writing video Yellow_Left.mp4
     100%|████████████████████████████████████████████████████████████████| 681/682 [00:35<00:00, 19.45it/s]
     [MoviePy] Done.
     [MoviePy] >>>> Video ready: Yellow_Left.mp4

     Wall time: 36.3 s

[4]: from moviepy.editor import VideoFileClip
     from IPython.display import HTML
     white_output='white.mp4'
     clip1 = VideoFileClip("test_videos/solidWhiteRight.mp4")
     white_clip = clip1.fl_image(process_image)
     %time white_clip.write_videofile(white_output,audio=False)

     [MoviePy] >>>> Building video white.mp4
     [MoviePy] Writing video white.mp4
     100%|████████████████████████████████████████████████████████████████| 221/222 [00:11<00:00, 19.63it/s]
     [MoviePy] Done.
     [MoviePy] >>>> Video ready: white.mp4

     Wall time: 12.5 s
```

white.mp4



Yellow_Left.mp4



2. Identify potential shortcomings with your current pipeline

After detection all the lines in the triangular mask, I realized that, the lines have discontinous and the lines were broken. I think it is about canny filter parameters.On the other hand after Hough transform I can achive to find the lines.

3. Suggest possible improvements to your pipeline

A possible improvement would be to apply color filters to find the lines. And also high pass filters to see the details on the lines. On the other hand , I used triangle mask to see the in front of the Car,it can be solved by another type mask shape.

Murat Tunç

muratbekonet.mt@gmail.com