Finding Lane Lines on the Road

Project Write-up by Scindia Dhanasekaran

Reflection:

Reading image files:

I read the images of type `'numpy.ndarray'` with dimensions : `(540, 960, 3)` as below:

```
<matplotlib.image.AxesImage at 0x7fd3b7e8a240>
```

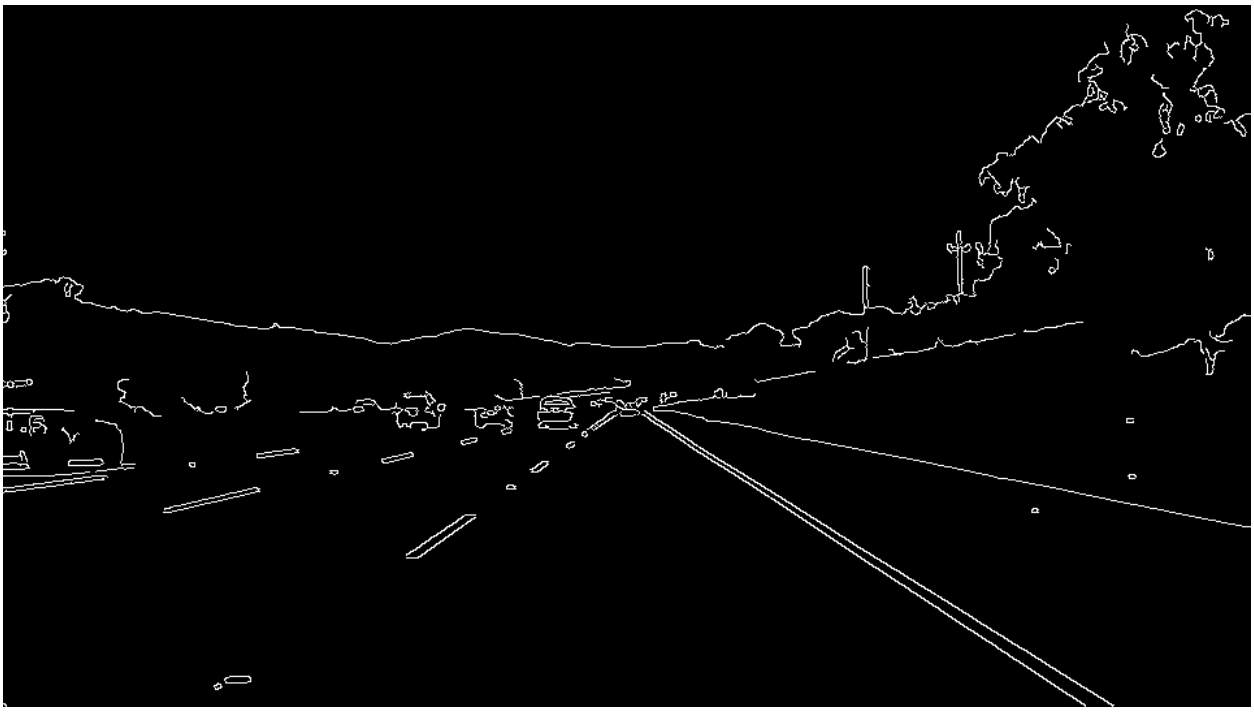
My pipeline consisted of following 5 steps:

Step 1: Transformed images to one color channel using Grayscale by using cv2.cvtColor(img, cv2.COLOR_RGB2GRAY):

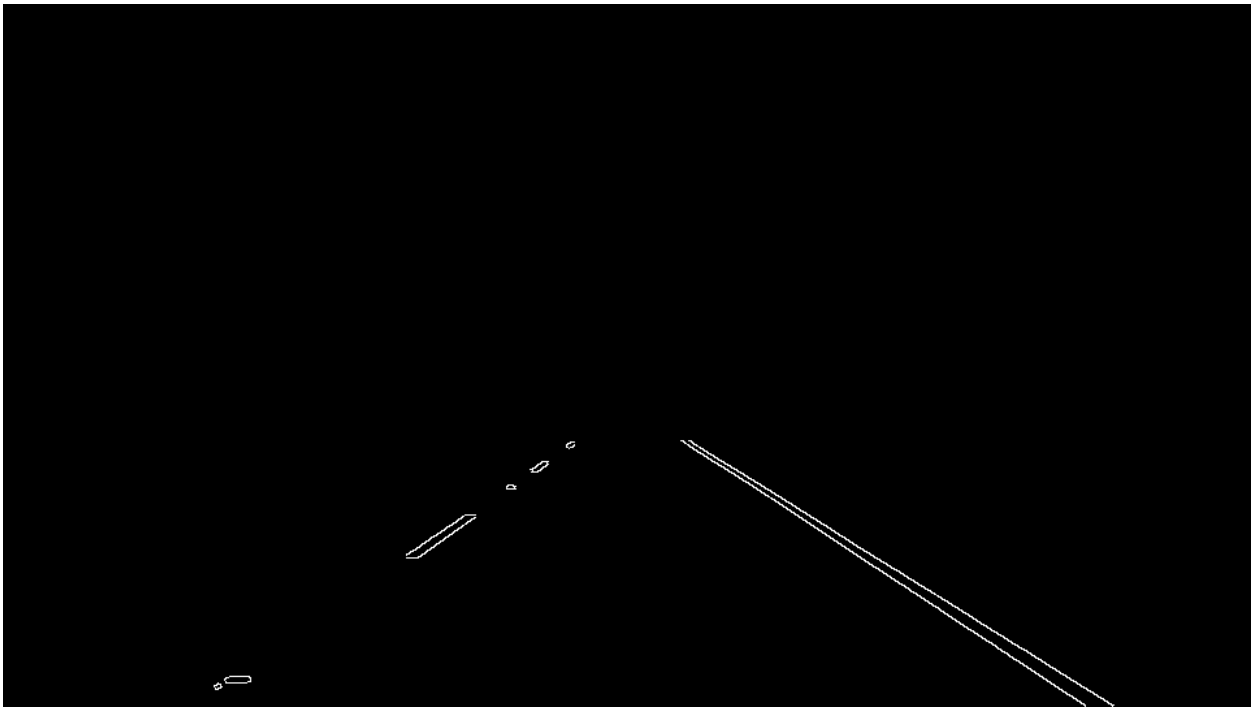
Step 2: Convert images to blurred using Gaussian Noise kernel(cv2.GaussianBlur(img, (kernel_size, kernel_size), 0))

Step 3: Applied canny operator to detect edges by using cv2.Canny(img, low_threshold, high_threshold)

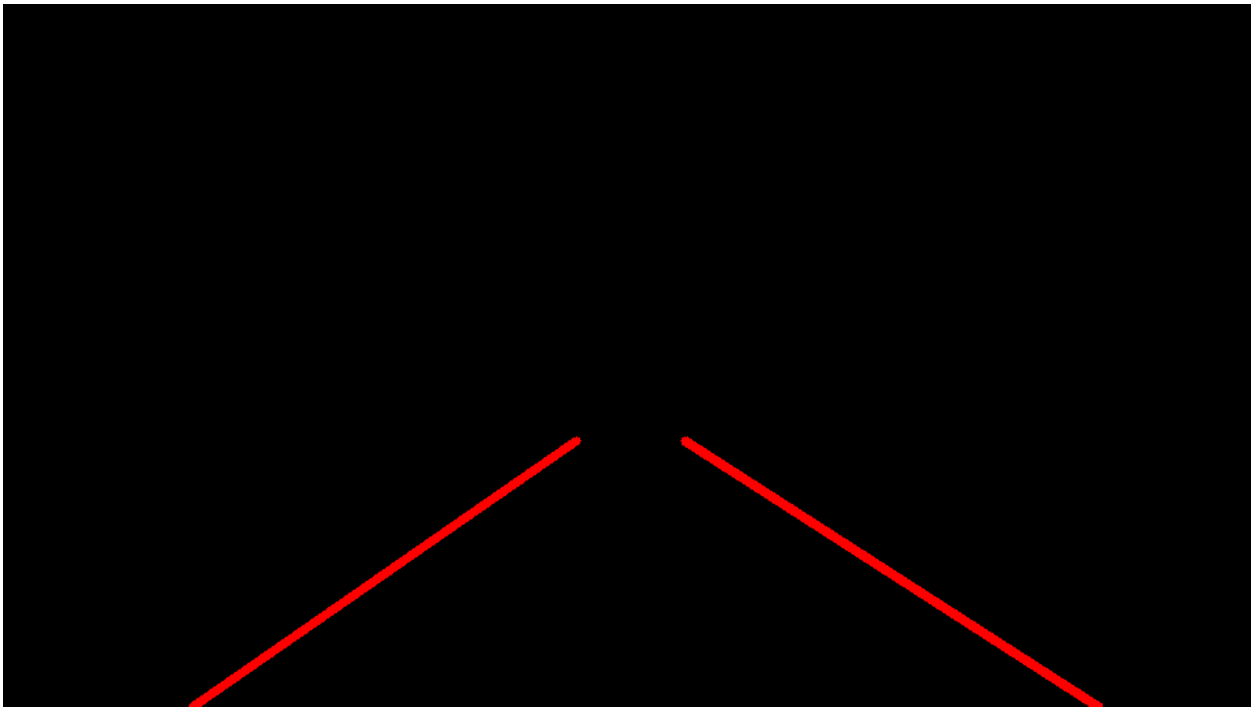
In this step I used 50 as low threshold and 150 as high threshold



Step 4: Masked image to keep the region of the image defined by the polygon formed from `vertices`. The rest of the image is set to black. By using



Step 5: Applied hough line transform:



Step 6: Detected Lane Line:

In this final step I set left set of lines to a single left line and right set of lines to single right line. This was achieved using draw_lines(). I had to discarded lines closer to slope 0, then took the mean of X and Y coordinates. This gives single line. Below is the output of draw_lines()

Limitations with my pipeline:

1. This pipeline when applied to challenge video might need additional processing to canny algorithm to reduce the noise.

2. Also on the same video my assumptions that lane ends at the bottom is not true.

Possible enhancements to my pipeline

1. Need to address real world nosie of animals, humans, snow, shadow etc..

2. Factor in bends and turns