Lab 4 - Edge and Line detection
Computer Vision

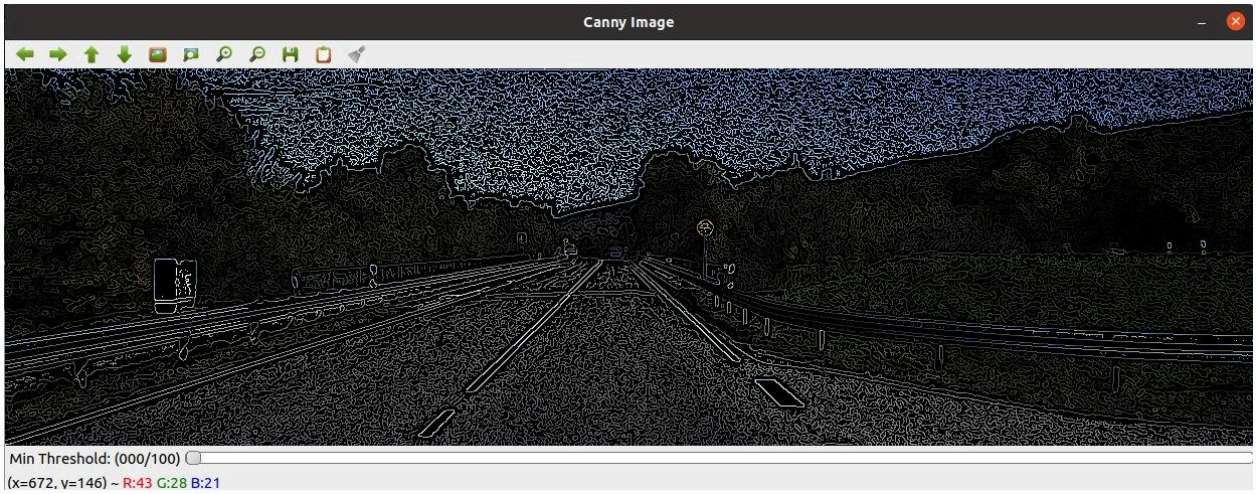
Christy Jo Manthara
2080644
MSc Computer Engineering

Task 1
Write a program that loads the image provided (street_scene.png), shows it and evaluates the Canny image. To verify the effect on the final result, add one or more trackbar(s) 1 to control the parameters of the Canny edge detector. Move the trackbars and check how changing each parameter has an influence on the resulting image. Please note: the Canny image shall be refreshed every time a trackbar is modified.

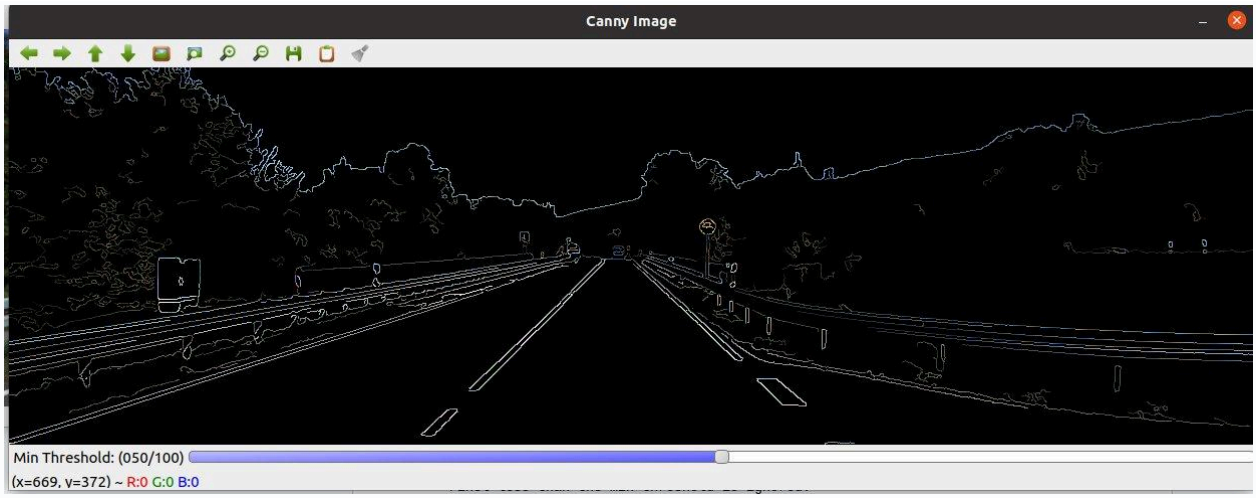
The required image is obtained with the imread().
Then this image is converted into a grayscale image.
The converted greyscale image is then made smooth using a gaussian filter with kernel k = 3.
This smoothened image is used as the input to create the canny edges.
The trackbar takes the min threshold and the max threshold.

The pixel is accepted as an edge if its higher than the max threshold.
Pixel less than the min threshold is ignored.
A pixel higher than the min threshold but less than the max threshold is selected only if it is connected to a pixel with threshold more than max threshold.

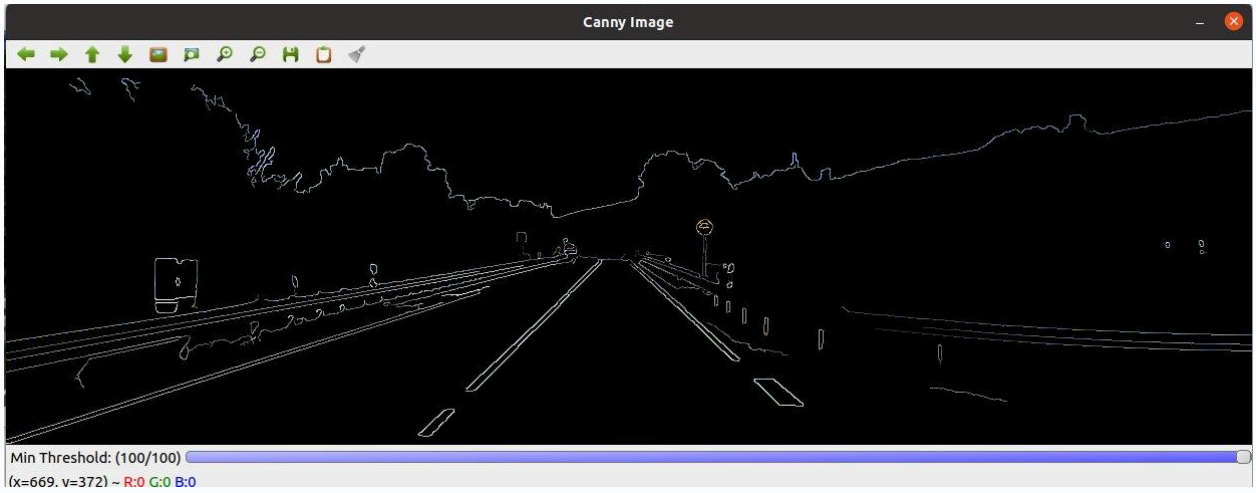
Furthermore the output we get will change based on the blurring obtained from the Gaussian.



Canny 0



Canny 50



Canny 100

Task 2
You now need to detect the white markings on the road. How could you tackle this problem
without using the Hough transform? Some suggestions:

- **consider edge orientation;**
- **consider colors close to edge points.**

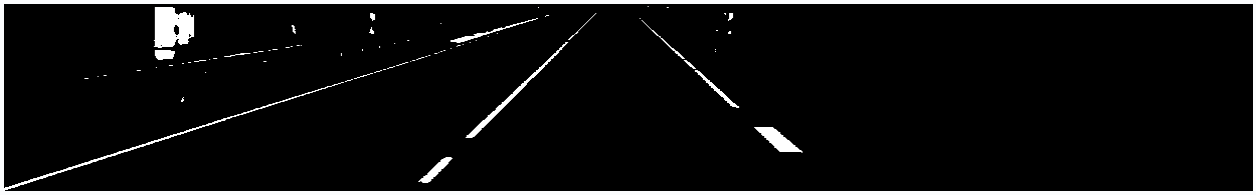
I used some other idea i had in mind. It was to use the threshold() with appropriate thresholds.
I got the following output for

```
threshold(grayImage, grayImage, 240, 255, cv::THRESH_BINARY);
```

The image also selects the object o the side of the road.

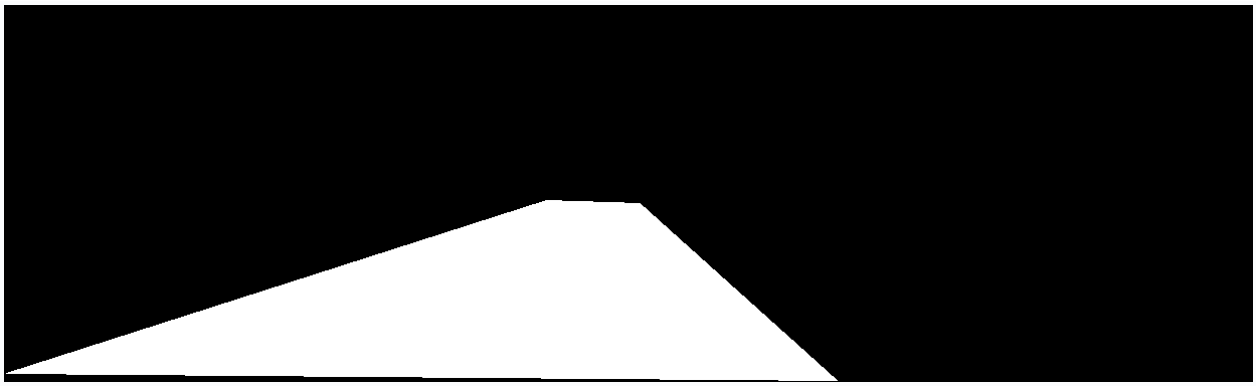
Now i crop out the top half of the image and apply the threshold to only the bottom half:

The cropped threshold is:



Now i will create a roi with the coordinates of the road and run the threshold in that to get the lines on the road.

On filling the polynomial we get



The final output on applying bitwise_and operator with the mask created is:-

Task3
Now detect white markings using the Hough transform. Check online sources and apply it
using the cv::HoughLines() function. Suggestion: consider the two strongest lines detected,
and select their orientation. Color in red the area between the lines - example below.

I converted the image into grayscale and applied threshold on it to convert it to binary.
Then use median blur with kernel size 5 . // gaussian blur was giving a lot of noise while plotting the lines.

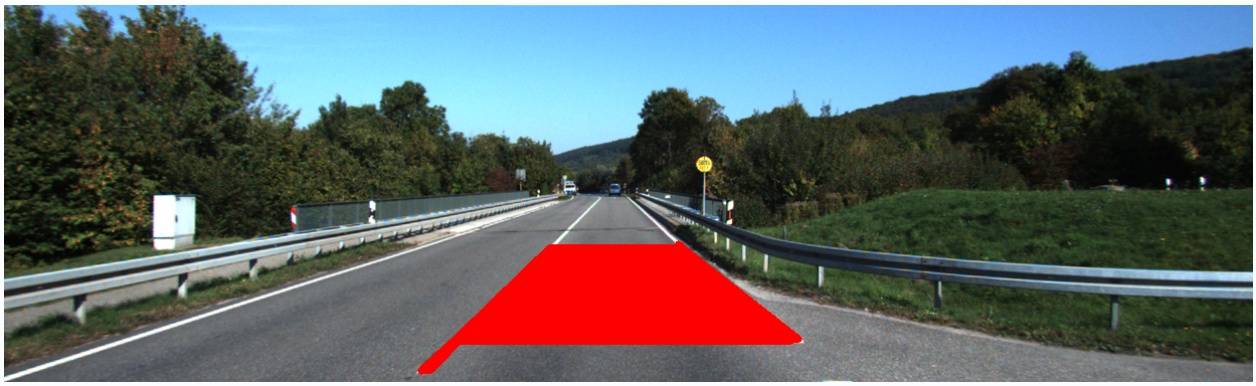
Then i used the hough transform with HoughLinesP()

```
HoughLinesP(grayImage, lines, 10, CV_PI/180, 50, 100, 70 );
```

I used the probabilistic hough transform as the standard hough transform was not giving me any standard output.

Blurred grey image

ploted lines in the image using the HoughLinesP function and identified the strongest lines.



Painting over the final lines to create the colored area.

Edit:

I have also done the same using the Standard houghline detector

Output of the canny on a grayscale which has been further blurred.

The lines drawn and plotted by the HoughLines()

The main issue i observed here was to effectively plot the values returned from the HoughLines().
The values returned were r and theta.

Which had to reconstructed to a line using the equation
For a line though the point (x1,y1) would be

r = x1* Cos(theta) + y1* Sin(theta)

Then you choose points on this line at two distances.
I wanted to detect only lines at approx 45degrees and approx 135 degrees from the origin. Hence i used an if -else condition to obtain this.

After coloring my output looks like

The file that gives this solution is Lab4task3_2.cpp

Task 4
Detect the road sign using the Hough circular transform - function cv::HoughCircles()

The circle was identified by HoughCircles.

This was further drawn on the color image using the FILLED attribute of the circle().
An unfilled circle is drawn on the grayscale image.