COL780 Assignment 1
Lane Detection Algorithm

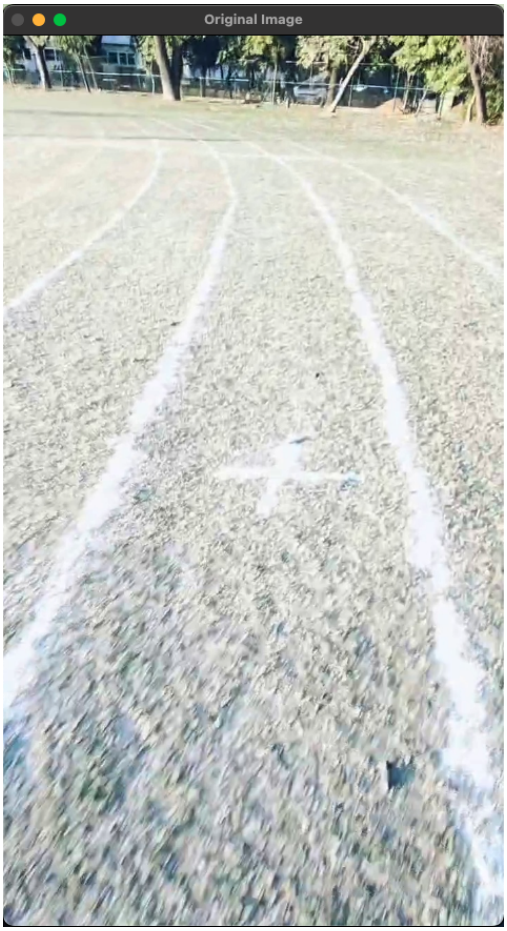
Yash Bansal - 2022CS51133

Contents

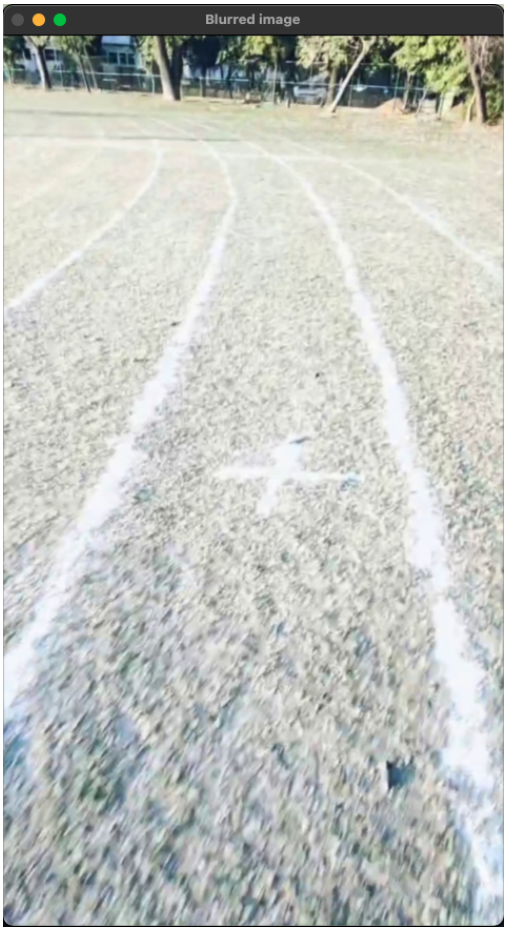
1 White Lanes in Grassy Images

The following steps are used to detect white lanes in the images with grassy grounds.

1. **Gaussian Blur:-** A Gaussian blur with kernel size 9 and sigma = 1 is used to reduce noises in the image. This highly reduces noise due to grasses and leaves while preserving the white lanes. The kernel size is decided after multiple experiments.

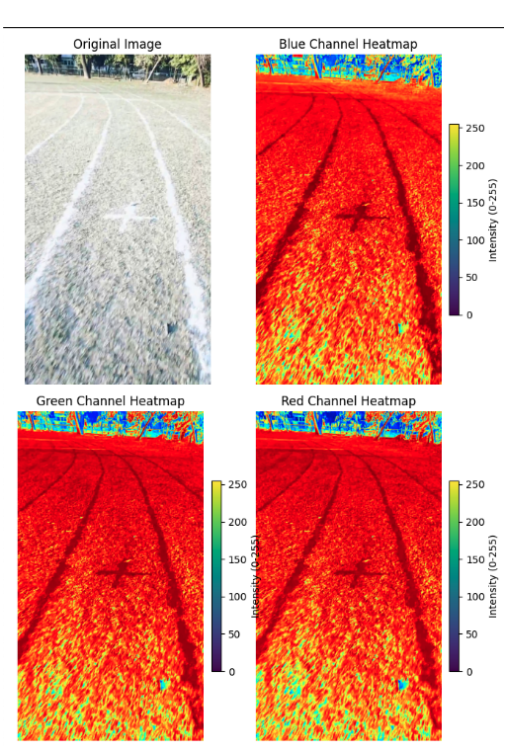


(a) Original Image

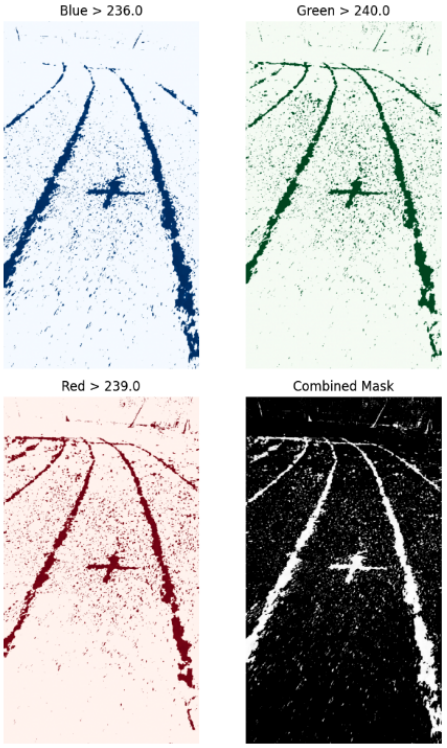


(b) Blurred Image

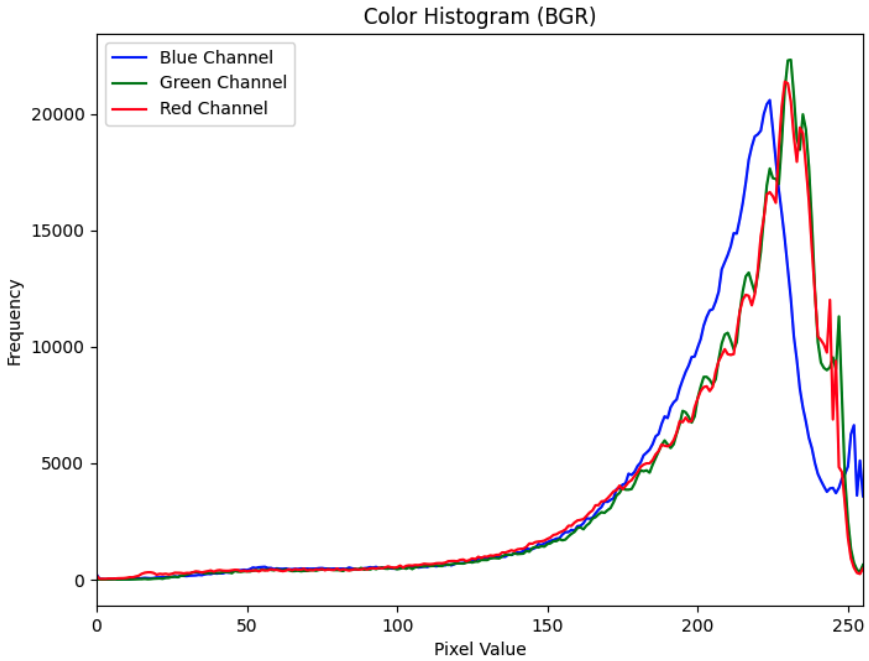
2. **Enhancing White regions:-** Then I did a heatmap analysis of the image for each channel. From this, it can be seen that wherever there is a white lane in the image, the red, blue, and green components are also very high. However, the thresholds for each channel can be very different, depending upon the frequency distribution of the particular channel in the image. Thus, an adaptive thresholding method is used as described below.



(a) Heatmap for each channel



(b) Adaptive Thresholding Mask



(a) Each colour histogram:- Different distributions, thus different thresholds required

3. **Adaptive thresholding:-** For adaptive thresholding of each channel, the top 90 percentile pixels in each channel are taken, and other pixel values are reduced to zero. Then, the image is converted to a grayscale image.

4. **Median Noise Reduction:-** Now, after enhancing the white regions, there is still some noise between the lanes due to leaves/grasses in between. To remove this noise, a median noise reduction kernel of size 9 is applied. Kernel size 9 is chosen because most noise pixels occurs in batches of 20-30 pixels, thus kernel size 9 can very effectively reduce this noise, while preserving the lane pixels effectively.

(a) Enhanced White Image

(b) Reducing Noise after enhancing

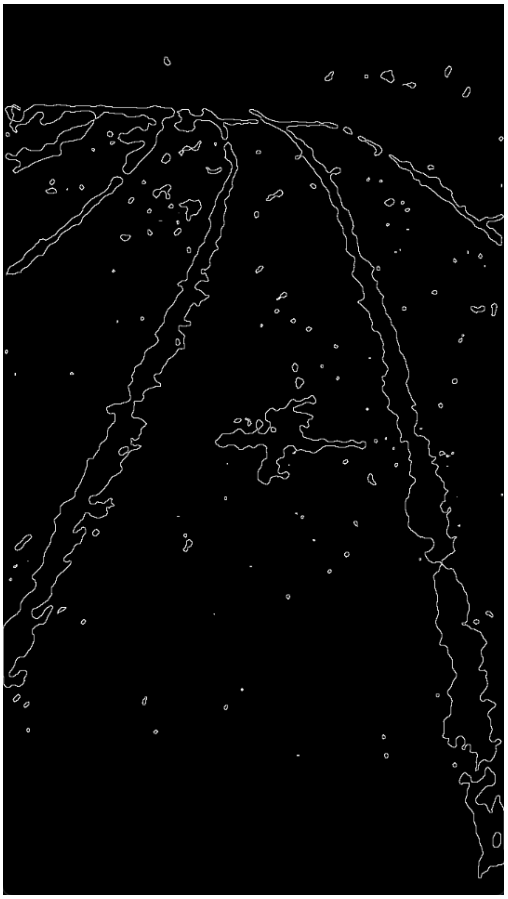
5. **Converting to binary image:-** Now, the image is converted to a binary image, as it helps the edge detection filter not focus on small noisy changes but on larger changes.

6. **Edge Detection:-** Now we pass the image through an edge detection kernel

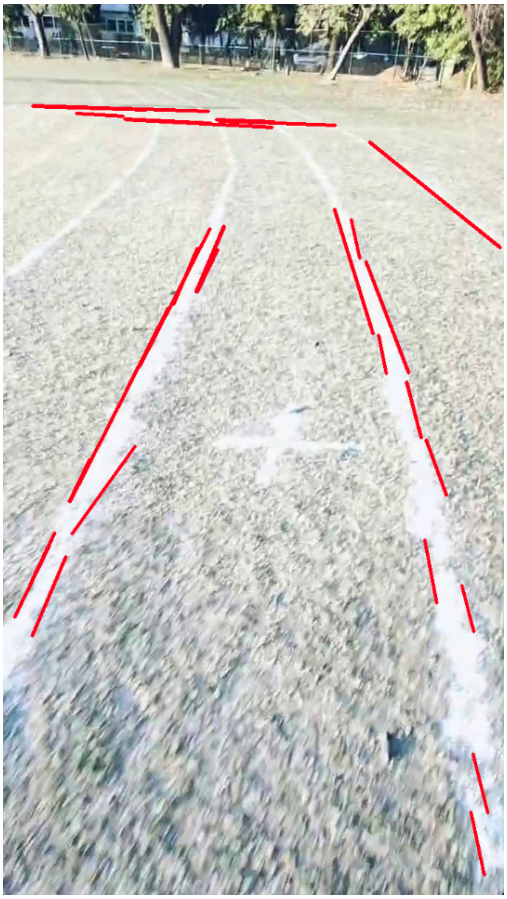
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

to detect the edges in the obtained binary image.

7. **Line Detection:-** Now, Hough transform in its polar form is used to detect lines in the edge-detected image. For this, the votes threshold is kept at 50, the maximum line gap is 20, and the minimum line length is 50. The function randomly chooses a point from the set of remaining points and tries to fit some line from that point, and if the maximum number of votes of that line crosses the votes threshold and other criteria are satisfied, then the line is chosen. This process continues till the number of detected lines reaches 30 or we are out of available points. This detects approximately 30 lines of each image. After this, the lines are improvised.



(a) Edge Detection in preprocessed image



(b) Initial detected lines

8. **Filter by slope:-** The detected lines having an absolute value of slope ¡ 0.3, which amounts to almost 16 degrees slope, are removed, as these lines are just noises due to shadows and trees. No lanes in the images have slopes in this range.

9. **Merging Similar Lines:-** Now, for all pairs of lines, if the lines have slope difference in range ¡ 0.3, and any of the perpendicular distances from endpoints to another line is less than 40, then these two lines are merged. This is because it is highly likely that these lines belong to the same lane. This process is done iteratively until there is no change in the number of lines after merging or the number of lines reaches less than 5. This process is again repeated, now with a slope threshold of 0.4 and a distance threshold of 50.

10. **Filter by Length:-** Now, the length of all the lines is calculated, and the longest 7 lines are kept. These are the final detected lines in the image.

11. **Task 2:- Line Fit score:-** For each pair of detected lines, we calculate the intersection points between each pair of lines, compute the centroid of all the intersection points, and then take the sum of distances of each intersection points from the centroid. It gives good results when the lanes are straight lines.

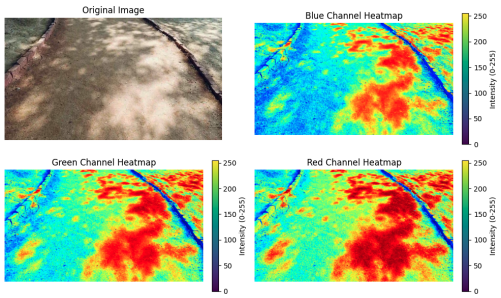
(a) Final lines detected

2 Red Brick lanes

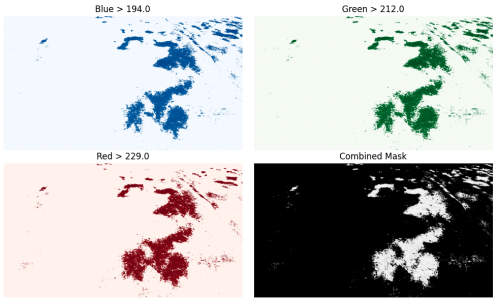
The following steps are used to detect red brick lanes in red ground images.

1. **Heatmap:-** Like white images, I tried to create a heatmap for red images to see some pattern, but any such pattern cannot be seen from the heatmap nor from colour histograms.

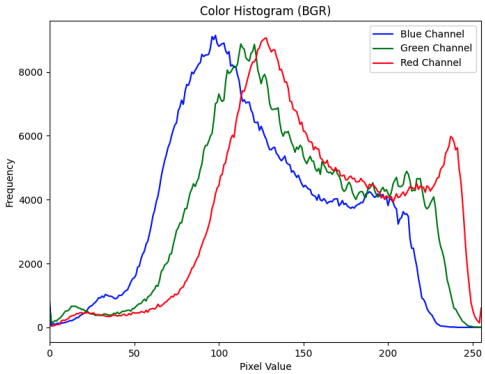
(a) original Image



(b) Channel heatmaps



(a) Masked channels



(b) Color histogram

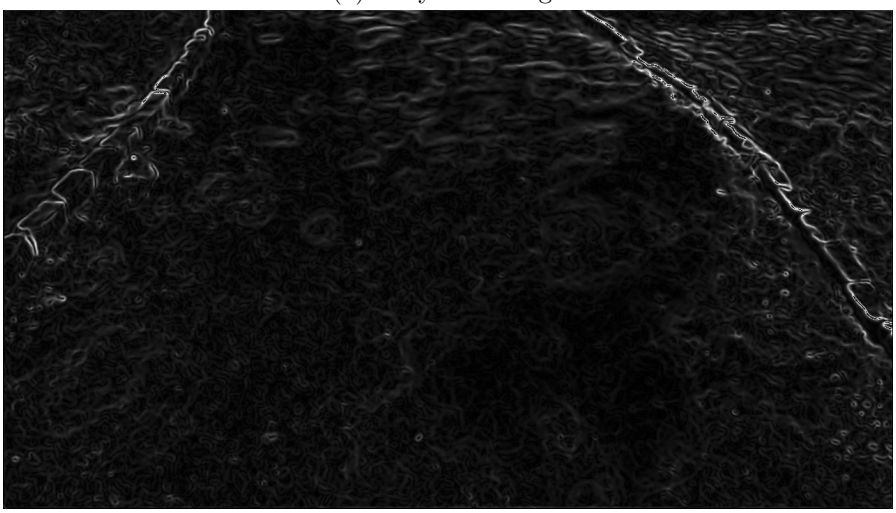
2. **Noise reduction:-** A median filter of kernel size 5 and then a mean filter of kernel size 3 are applied to the image to reduce the noise.

3. **Adaptive Blurring:-** To preserve edges while blurring but also reduce noise in an effective way, an adaptive blurring procedure is used. For this, a blur intensity is calculated based on the standard deviation of the pixel from its neighbourhood pixels of kernel size 5. If the standard deviation is low, then this is likely not an edge. Thus, there will be a high blurring factor to kernel size 9 Gaussian. If the standard deviation is high, then low blurring of Gaussian kernel 3 is applied. This method effectively preserves edges while blurring. This blurring is done on all the 3 channels independently, to preserve edges along all the 3 colors.

(a) After noise reduction

(b) After blurring

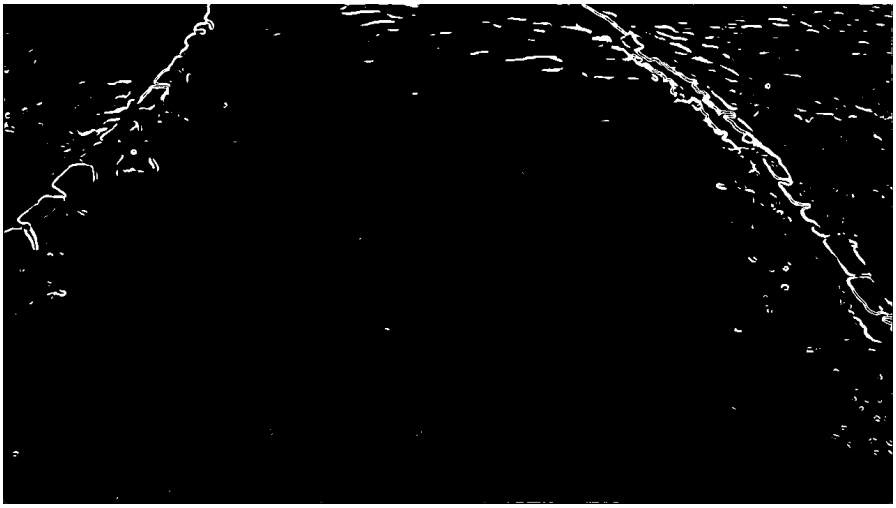
(a) Grayscale image



(b) Initial Sobel edges

4. **Sobel edge detection:-** Now the image is converted to grayscale, and Sobel edge

detection is applied to the image. The threshold value for edges is 75.



(a) Sobel edges after applying threshold

(b) Initially detected lines

5. **Line Detection:-** Now, similar line detection, as of the white lanes one, is applied to detect the lines in the image.

(a) Final Lines detected

3 Image Segregation Procedure

By carefully analyzing the images and their histograms, it is seen that for brick images, there is a high red ratio compared to green. At the same time, it is the opposite for grassy images (logical too). Thus, the images are segregated accordingly.