Computer vision course

Lab 4 - Edge and line detection

Pietrobon Andrea

27 maggio 2023

1 | Task 1

The first task was not complicated. I managed to implement the 2 sliders easily. and apart from a small confusion problem with the variables, which caused the second slider to change the same values as the first. Fixed this bug everything worked fine. The Figure 1.2 shows the image with the 2 sliders.

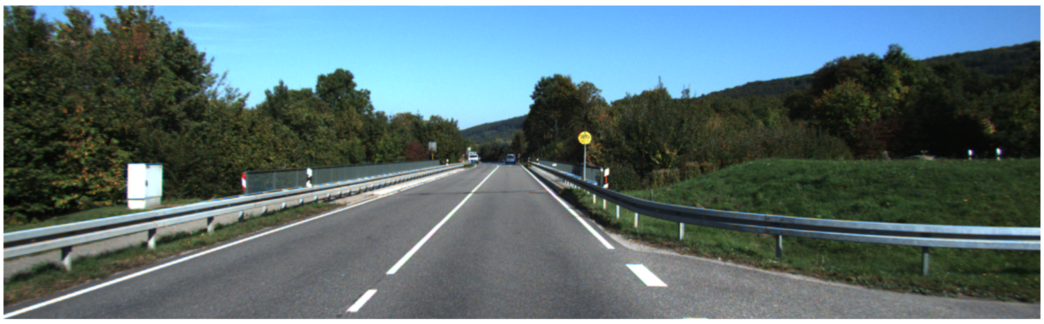

Figura 1.1 Original Image.

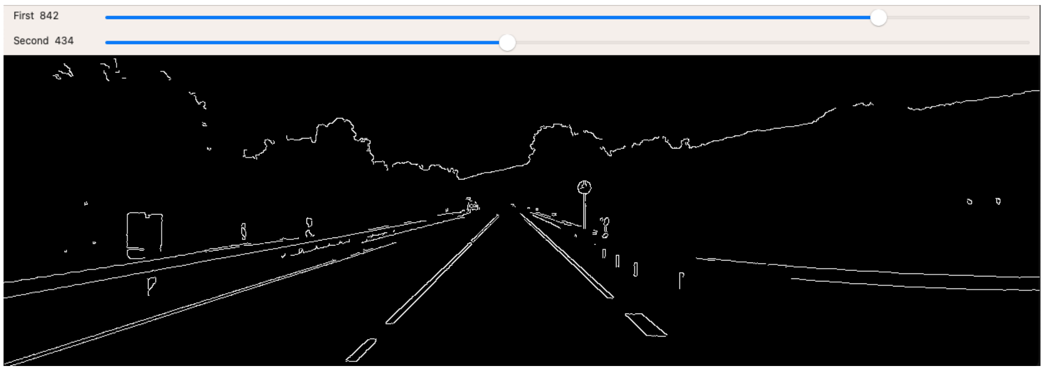

Figura 1.2 Edge detection with canny and the sliders.

2 | Task 2

In this task, the first step of image cannying was not complicated. The second step (i.e. removing unnecessary edges) was more complicated. Without using specific functions to find the lines like cv::HoughLines() and then calculate the slope I didn't know how to do it. Because here I used multiple for loops to clean up the image and finally also to fill the blank lines. I wasted a lot of time looking for "more elegant" solutions than the current one but unfortunately I was unable to do better. Which has very bad time complexity. In Figure 2.1 you can see the image after trimming the canny of excess lines and then the final image in Figure 2.2.

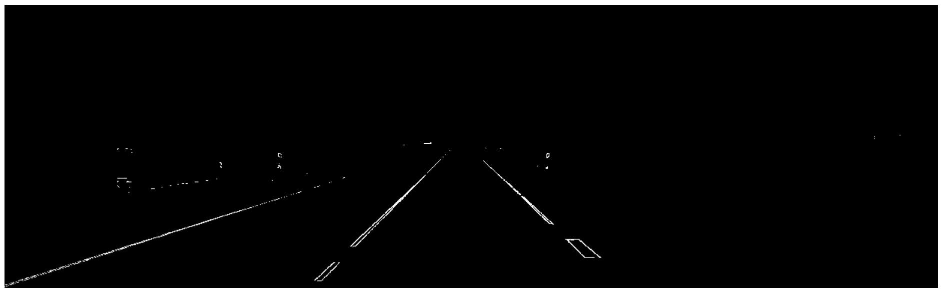

Figura 2.1 Edge detection with canny.

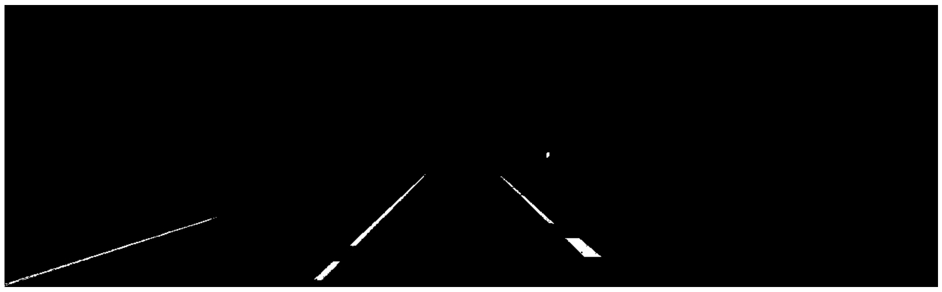

Figura 2.2 Final result.

3 | Task 3

After various attempts, including the use of a mask to take only the affected area of the image. I was able to keep only the necessary straight lines thanks to their inclination (Figure 3.1). This allowed me to delimit the edges of the affected area although not with little difficulty. In fact, the lines started from the outside of the image and this made it difficult for me to find the angles that delimited the figure. I used a vector with all the points of the two lines to find the vertices of interest and then thanks to them I traced the image on the main figure (as you can see in the Figure 3.2).

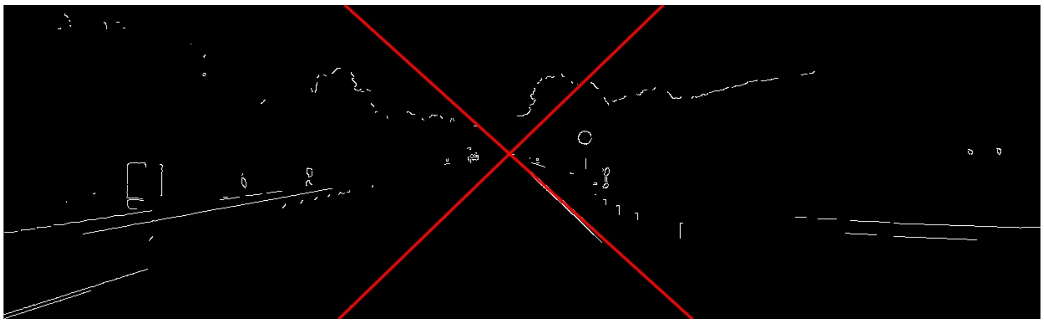

Figura 3.1 Image with only the main lines detected with HugeLines.

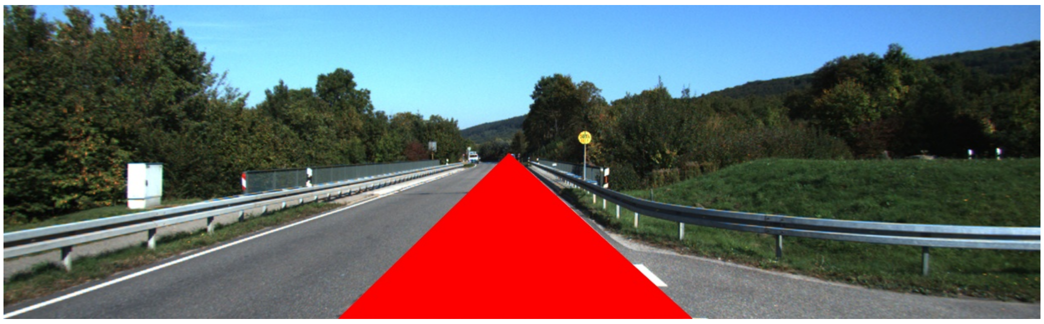

Figura 3.2 Final image.

4 | Task 4

The last task was faster than the previous 2 to run although I had some problems using cv::HoughCircles() on the bordered image found by canny. In fact it found circles above it but completely randomly (although it had a low threshold) and when I tried to color them directly in the final image with canny it didn't color them, leaving the image the same as it was immediately after the canny . I fixed it by simply using the grayscale image and directly printing the circles on the color image. In Figure 4.1 you can see the final result.

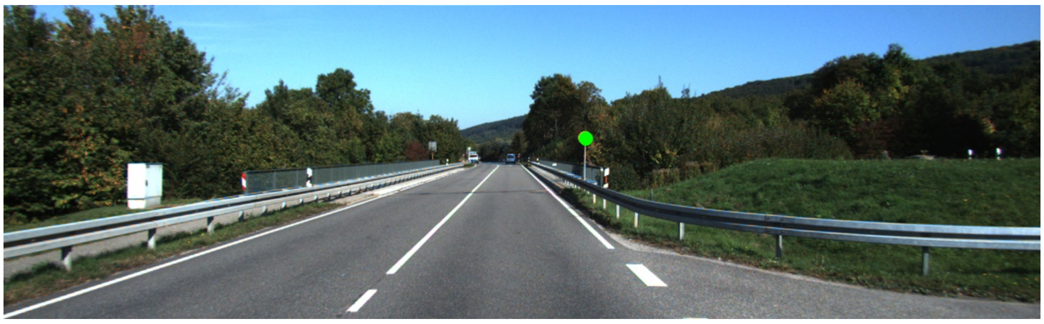

Figura 4.1 Final image.