12/25

from twitchslam i came to **lane marker detection**.

so i downloaded a driving video, to load this with python and opencv

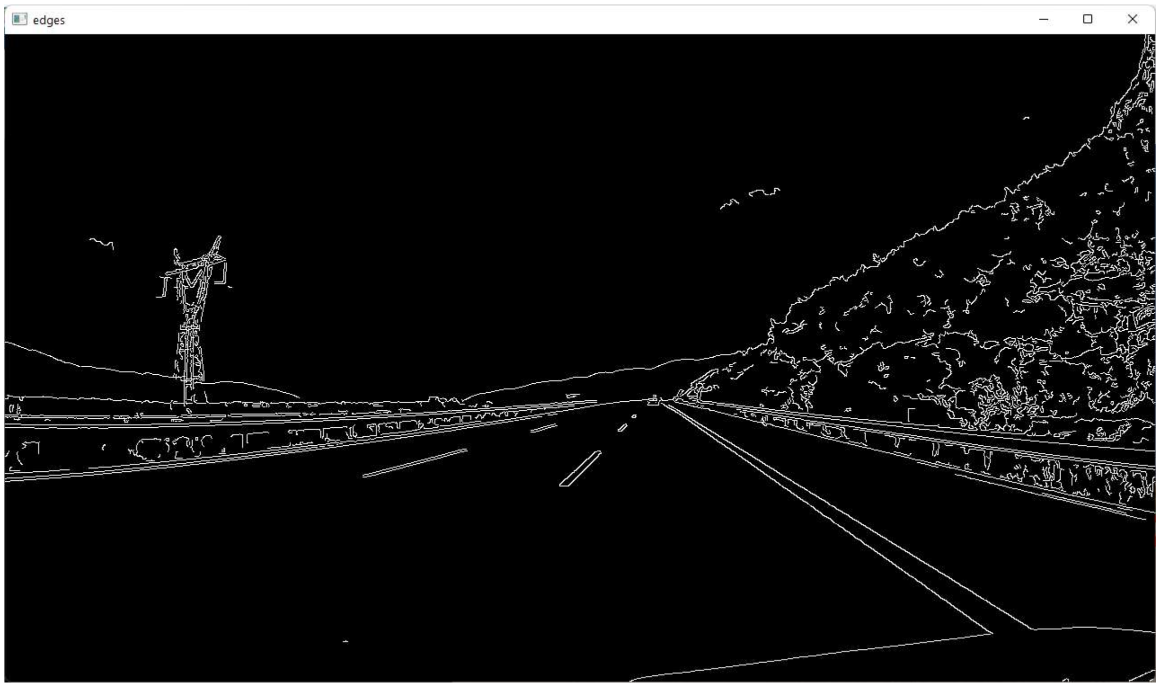
article 1

https://towardsdatascience.com/tutorial-build-a-lane-detector-679fd8953132

this is a pretty simple and straight forward pipeline

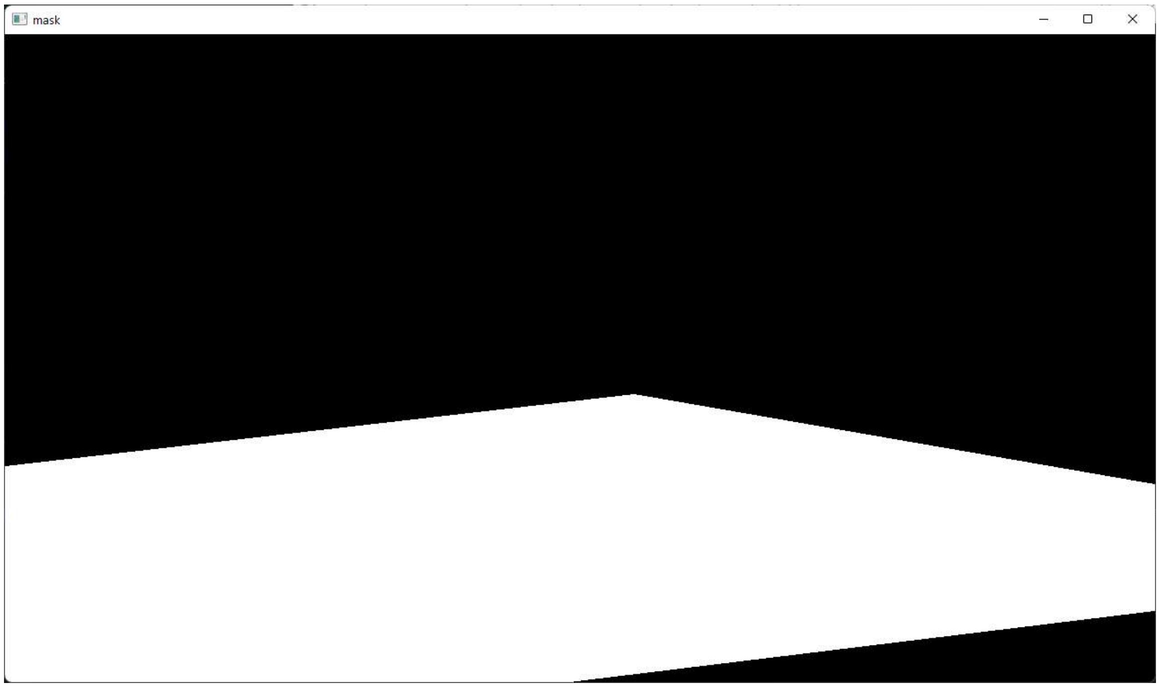
(0) gray scale and blur

(1) **canny**. sobel didn't work so well, i.e. created a lot of noise

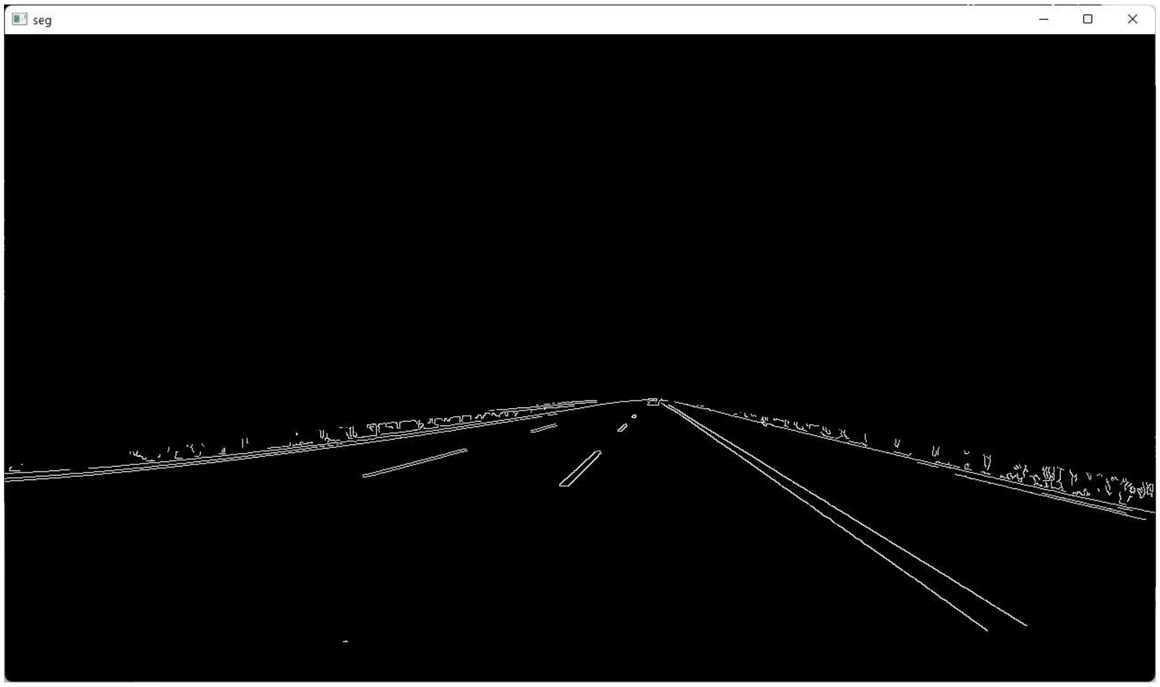


(2) mask non road area (heuristics), also avoid the hood

this is the mask

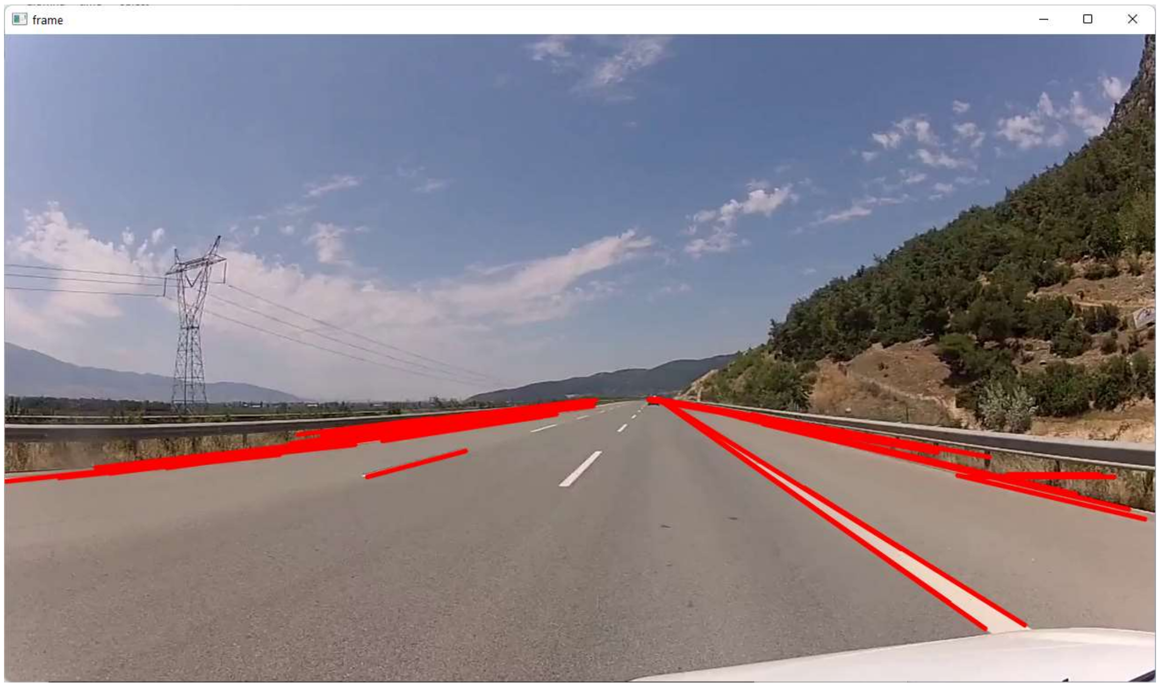


and this is the result after AND operation of canny and mask

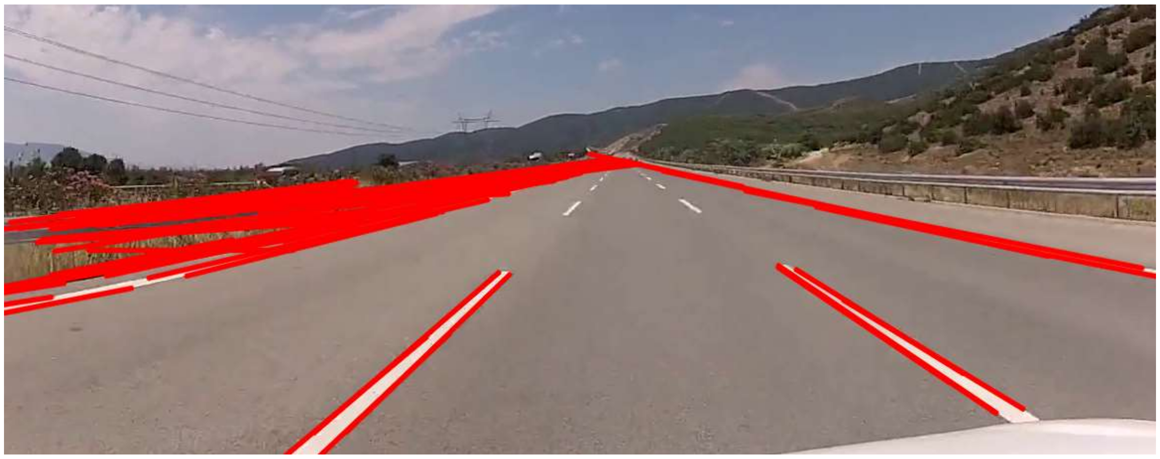


(3) Hough transform - can only get straight lines, so no road curves detectable

and drawing the resulting lines into the frame:



at this point i have just too many hough detections. the article does some sorting out of lines, so i have to look into this. [...] well, this sorts markers into left and right and then averages both. however we have more than 1 lane here, so this approach will not work anyway. [...] giving up on this



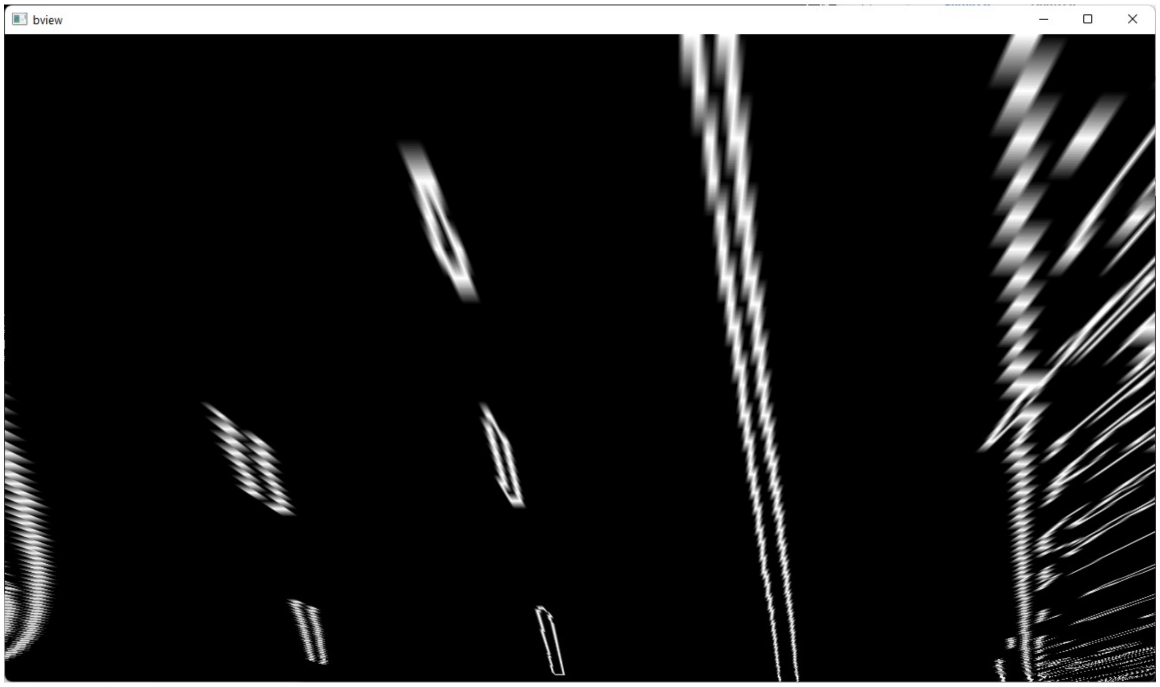
detection performance still is rather poor, i.e. a lot of markers are not detected.. (maybe not bad, if markers in vicinity are found).

also, the number of lanes and width and ego pos within lane is missing, so i have to look into a more sophisticated approach

article 2

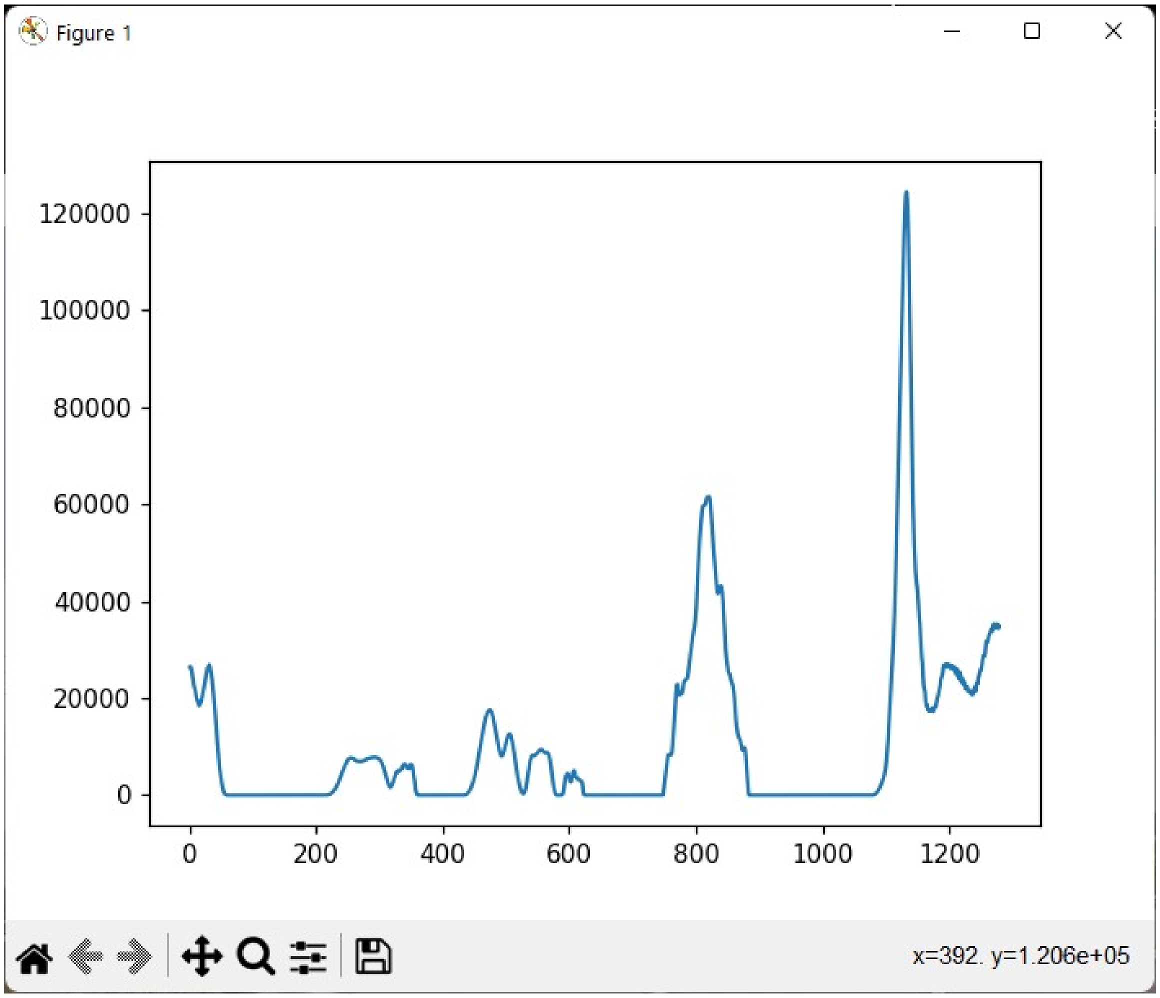
https://medium.com/@SunEdition/lane-detection-and-turn-prediction-algorithm-for-autonomous-vehicles-6423f77dc841

now, this introduces **perspective warping**, to get the birdseye view of the lane. works ok, with some finetuning of the trapezoidal coordinates:



projection solves problem of floating center?

histogram of the bview clearly shows 5 markers (with a 0-segment inbetween) **// this method is also employed by Andreas** Geiger https://www.youtube.com/watch?v=DPRFZODRf9E on a row-based method



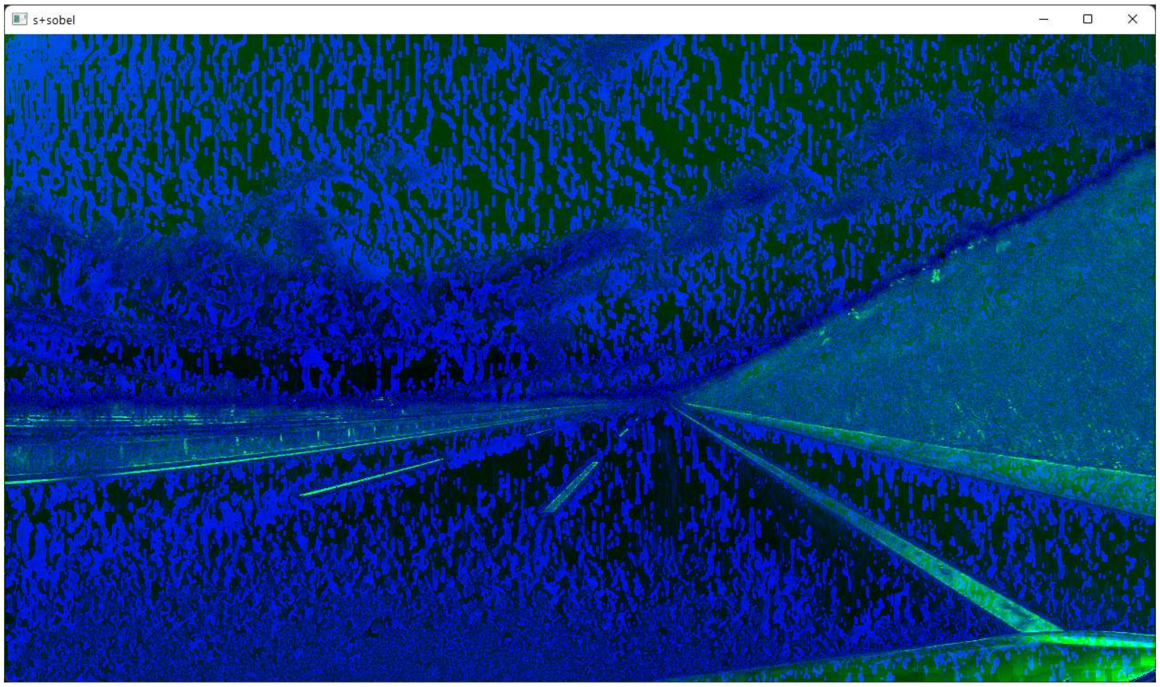
well, the rest is done very similar to paper 1, so it works with 1 lane only.

paper (3)

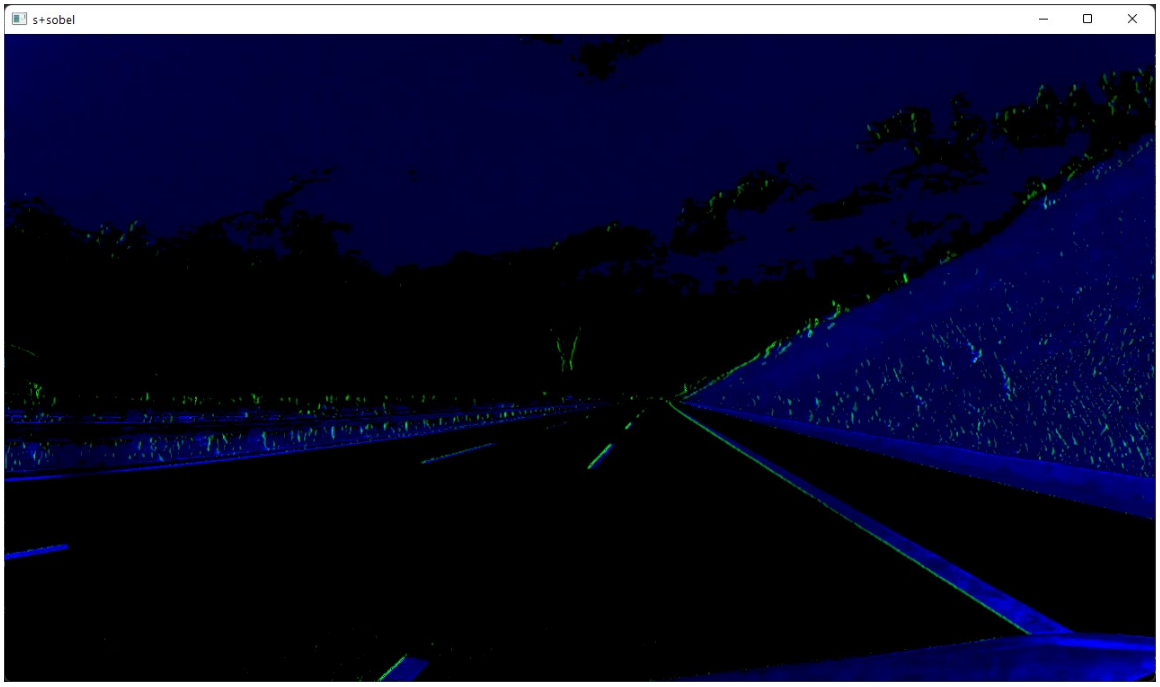
novelty: use B-spline curve fitting instead of Hough, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6679325/pdf/sensors-19-03166.pdf

camera calibration! -> image distortion removal (can't be done on a stock video)

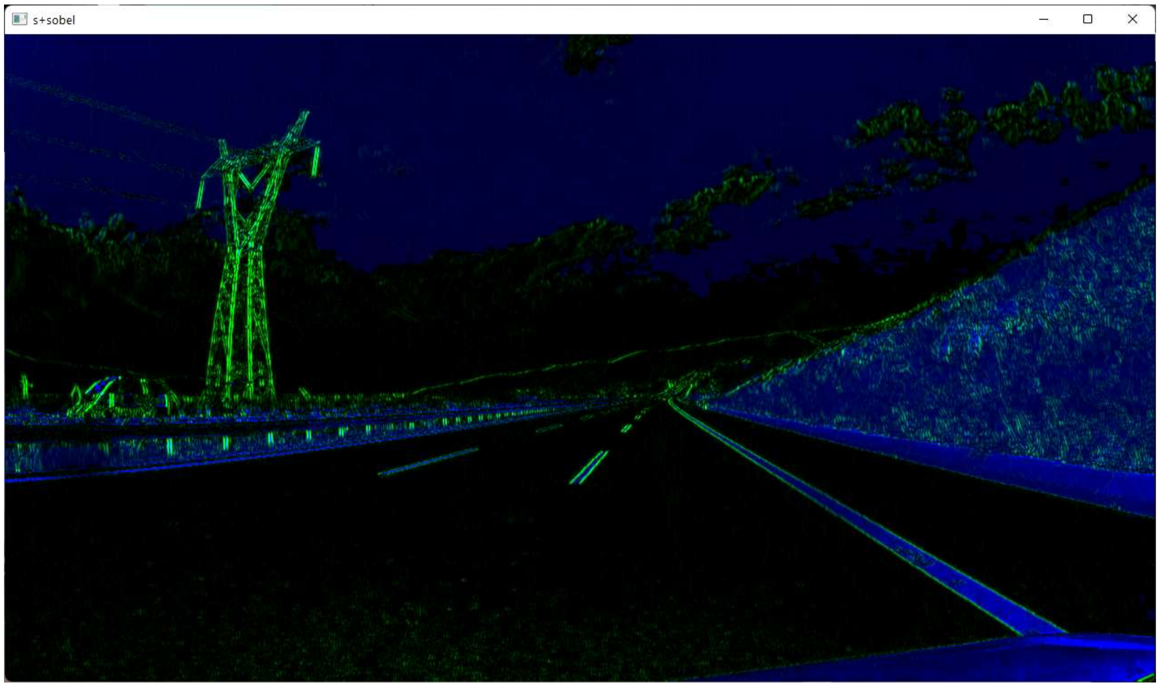
superposition threshold instead of canny, result looks awful



much better after setting the sobel-x kernel to 3x3 instead of 5x5 and thresholding the HSL, s-component to 50, also thresholding sobel to 50

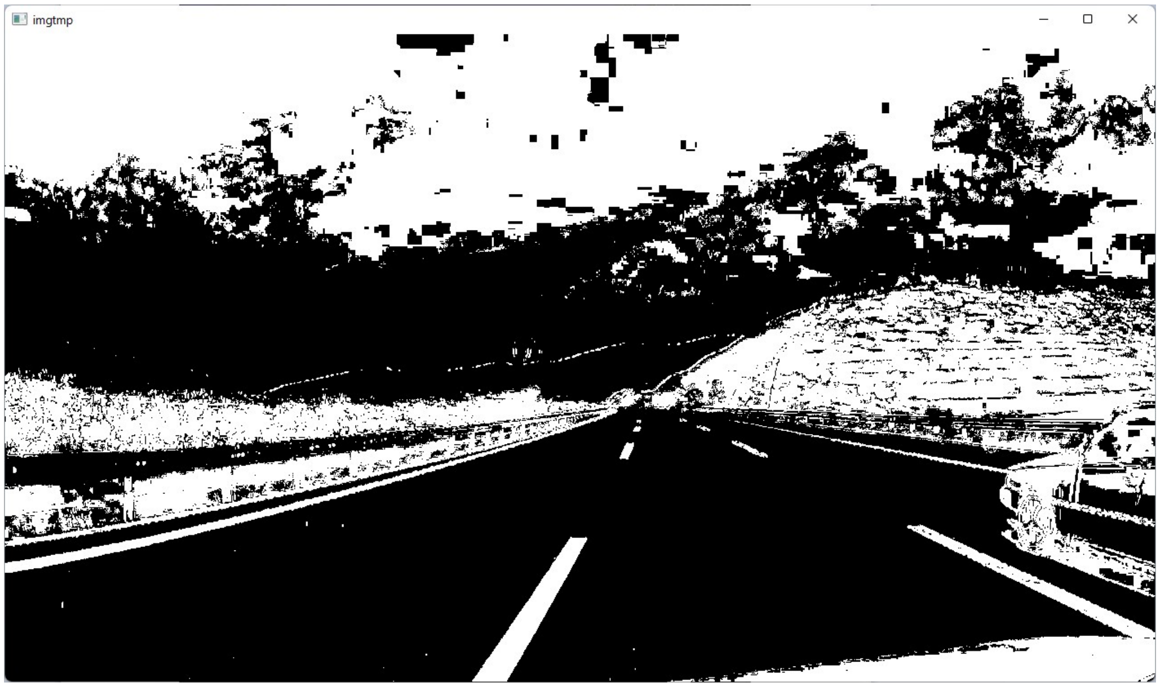


finally with a "symmetric sobel" both left and right side of the markers are emphasized:

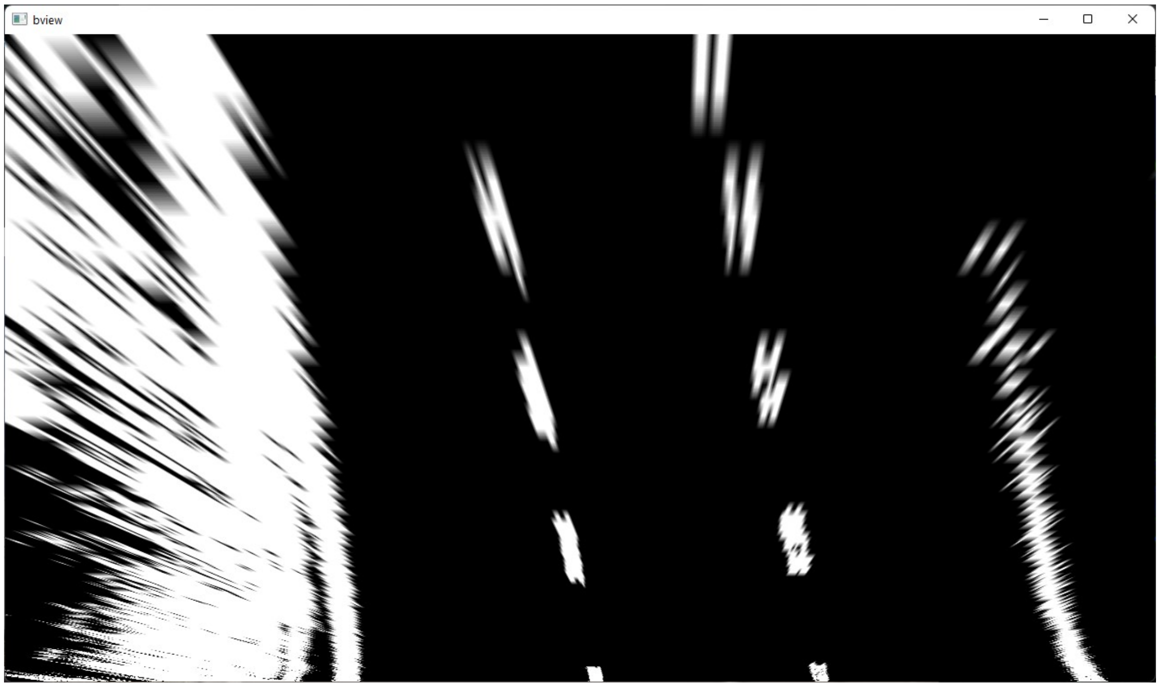


works all really nice, only problem is cv2.bitwise_and and the different image formats, which are not compatible

"binarization", threshold = 55, all > 55 =255, all <= 55 = 0

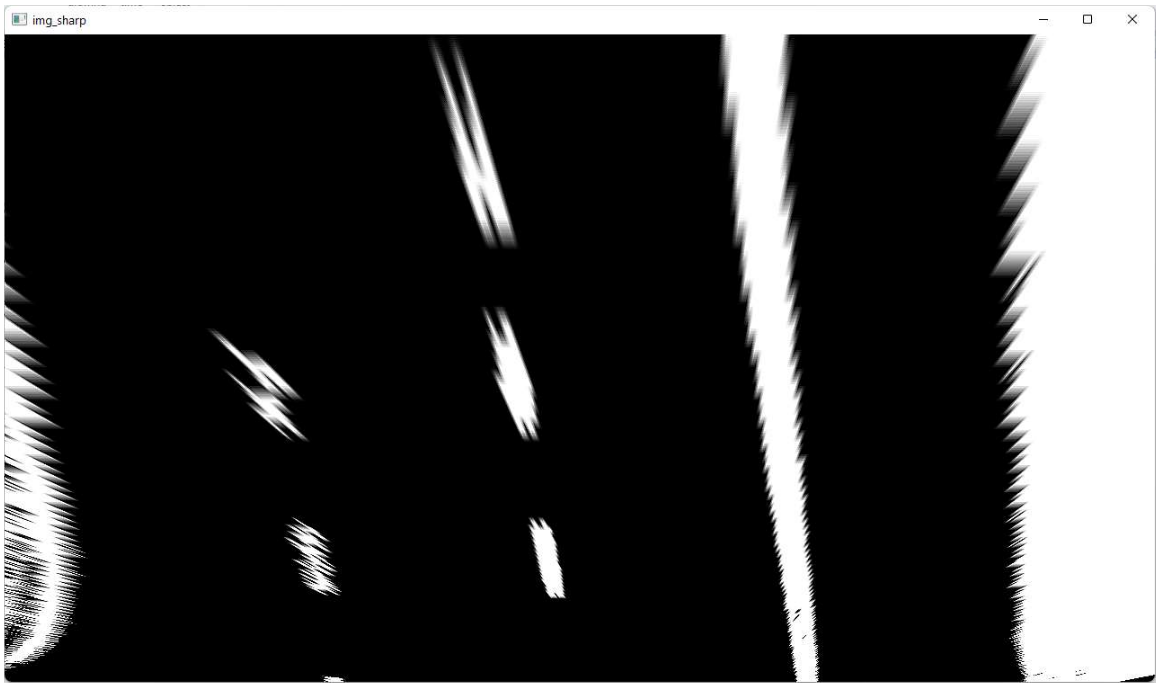


inverse perspective transformation:



at this point i ask myself, if the time-consuming processing of paper (3) better is than the canny approach in article 2. to verify we need (a) frame output to directly compare, (b) a metric to evaluate the lane detection result.

```
after sharpening with    [0,-1, 0]
                         [-1,5,-1]
                         [0,-1, 0]
```


img_sharp