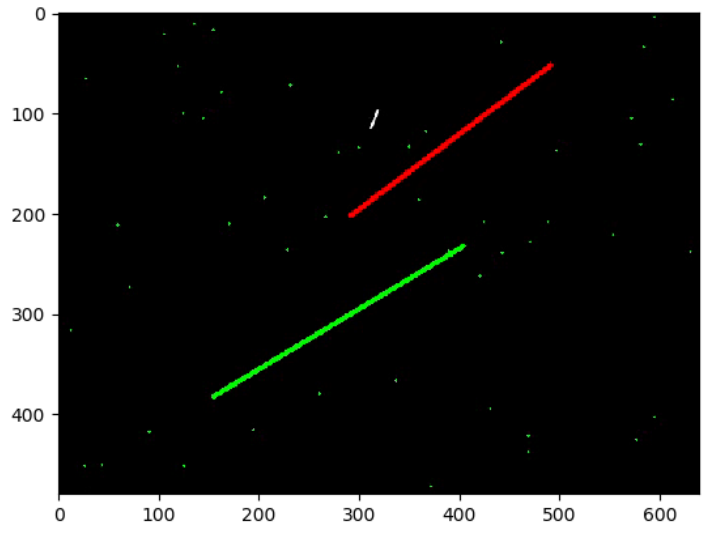
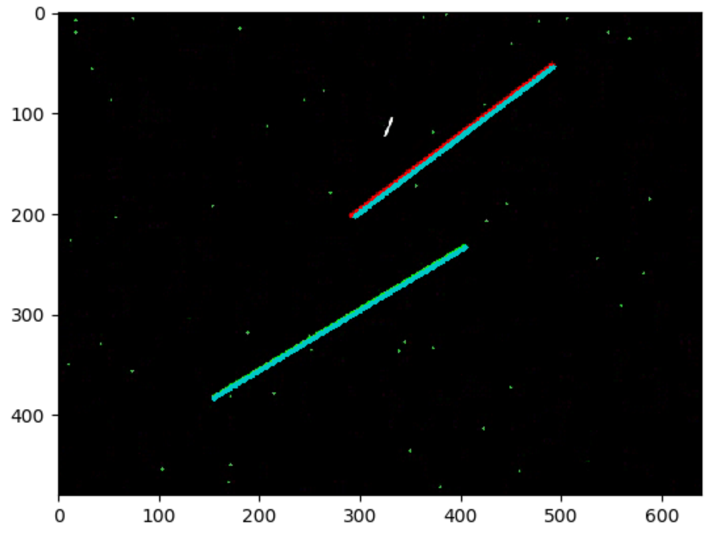
summing moving handwritten digits

line detection

- dynamic line detection, finding coordinates of its starting and ending point
- using open-cv api to prepare image for edge detection (Canny) and finding coordinates using hough transformation (HoughLinesP function)
- improving accuracy by maximizing line length based of first 20 frames



original input

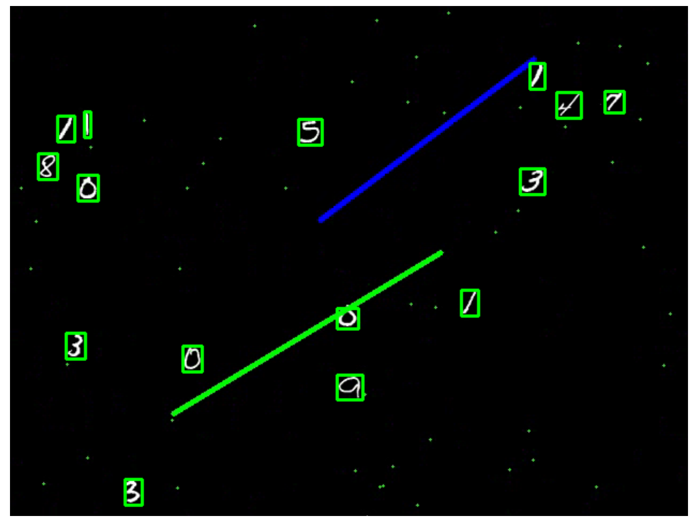


detected output

blue line:	[294 204] [492 55]
green line:	[154 384] [405 234]

digit region detection

- using open-cv api to separate white numbers from black background and to remove noise (green dots)
- calling findCountours on digit-only binary image, drawing bounding rectangles by respecting following parameters: area > 90, 18 < height < 50, 1 < width < 50
- making array coordinates array along with its dimensions



bounded rectangles

detecting region distance from lines

- detecting distance of the bottom-right corner from both of the lines for every frame
- calculations based on shortest distance from point to line using points normal on the line
- specifying bootom distance threshold for crossing the line
- sending crossed region to cnn for prediction
- setting timeout for next 7 frames to avoid multiple crossings for the same region

convolutional neural network training

- using keras tensorflow cnn implementation to form the model
- using mnist dataset to fit the cnn
- taking advantage of google colab notebook for cnn train
- normalizing training set
- 100 epohs, 1 input layer, 3 hidden layers, 10 outputs (possibilities)

prediction & summing

- normalizing region data to fit the model input
- prediction by cnn
- adding / subtracting predicted result

Soft Computing 2019 project, Milan Lazić