# Licence Plate Recognition System

Ossian Adrian & Pacurar Cristian

# Licence Plate Recognition System

## System use case

- Car parking ticket system for malls
- Traffic monitoring
- Speed ticket system
- Dirty plate recognition *

# Licence Plate Recognition System

## The idea behind the scenes

- Histogram equalization
- Binarization of the image
- Image closing
- Bounding box
- Character labeling
- Rectangle drawing

# The Algorithm

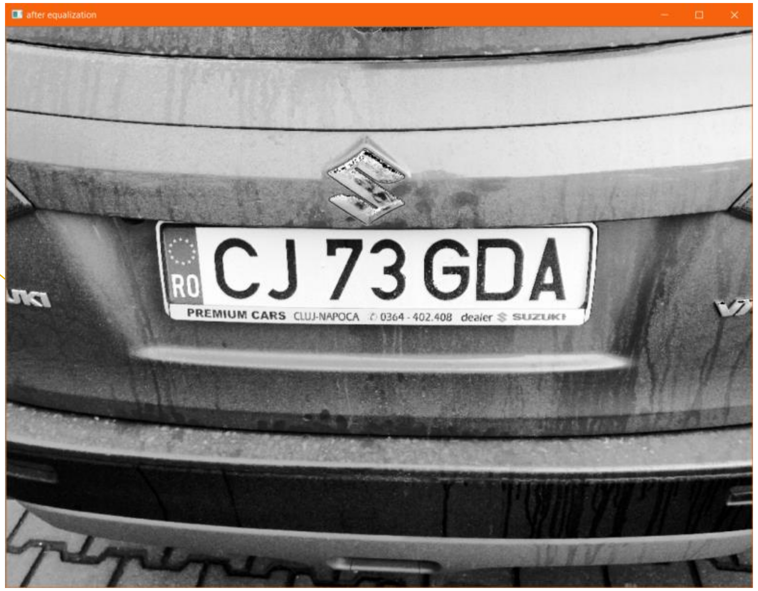
## Initialization

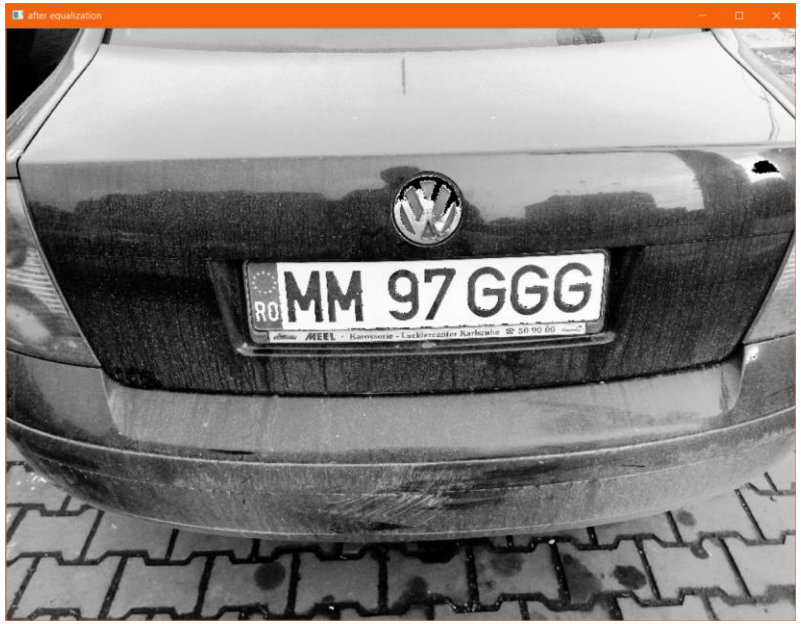
- Firstly, after we read the image, we resize it to get faster processing time, because most of the sample images are very large and we want to to be able to see all the changes in a relative short time. Of course, we keep the aspect ratio of the image so it doesn’t get distorted.
- After the resize, we copy a gray scale version of it. We will work with this one from now on.
- After all the additional initializations, we are ready to apply all the algorithms that will help us to get to our goal, namely, to be able to recognize the characters from a licence plate in an image.

# The Algorithm

## Histogram equalization

- Our first proper operation on the image is histogram equalization. Histogram equalization is a method in image processing of contrast adjustment using the image's histogram. Histogram equalization is best method for image enhancement. It provides better quality of images without loss of any information. It helped us to enhance the quality of the grayscale image.

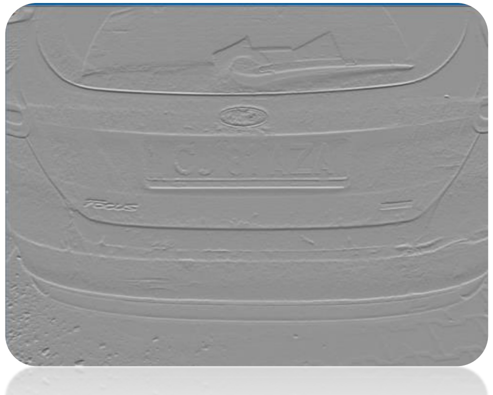




# The Algorithm

## Convolution

- Here we tried to apply different convolutions on the images: Gaussian, Line Core and Column Core. We tried to enhance image quality by using those convolutions, but we found out that not every one of them are usefull for our problem.
- In image processing, a kernel, convolution matrix, or mask is a small matrix. It is used for blurring, sharpening, embossing, edge detection, and more. This is accomplished by doing a convolution between a kernel and an image.
- Convolution is the process of adding each element of the image to its local neighbors, weighted by the kernel. This is related to a form of mathematical convolution.



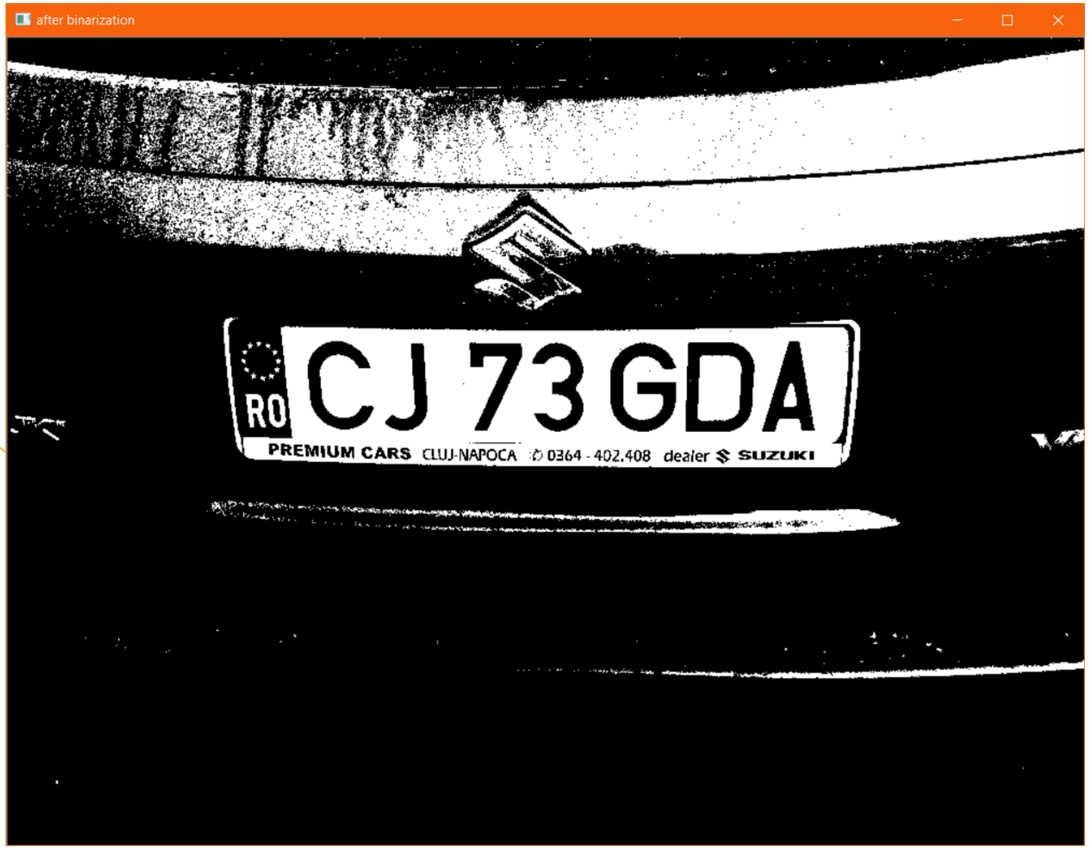
# The Algorithm

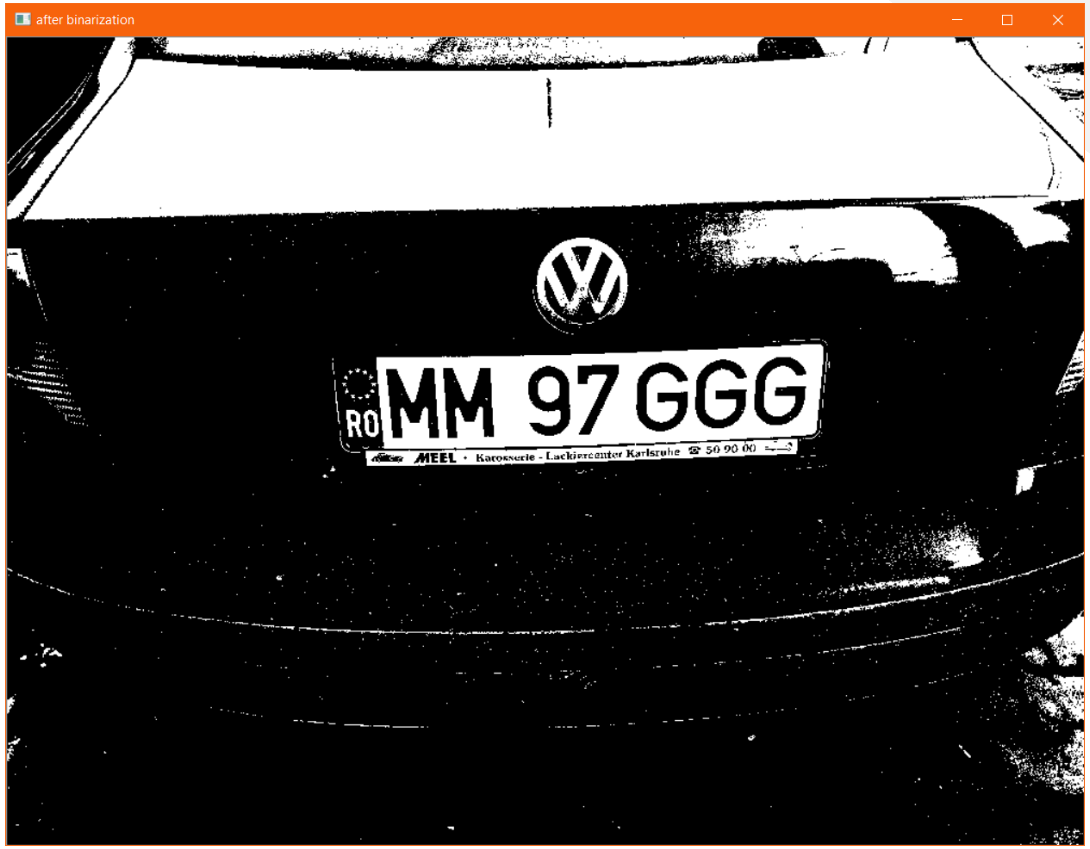
## Automatic Binarization

- In digital image processing, thresholding (binarization) is the simplest method of segmenting images. From a grayscale image, thresholding can be used to create binary images.
- Automatic thresholding (binarization) is a great way to extract useful information encoded into pixels while minimizing background noise. This is accomplished by utilizing a feedback loop to optimize the threshold value before converting the original grayscale image to binary. The idea is to separate the image into two parts: the background and foreground.
- Automatic binzarization will work best when a good background to foreground contrast ratio exists. Meaning the picture must be taken in good lighting conditions with minimal glare.
- **We applied automatic binarization using only with the values in the licence plate area. This results in a clean image of the characters for the next operations.**

# The Algorithm

## Automatic Binarization





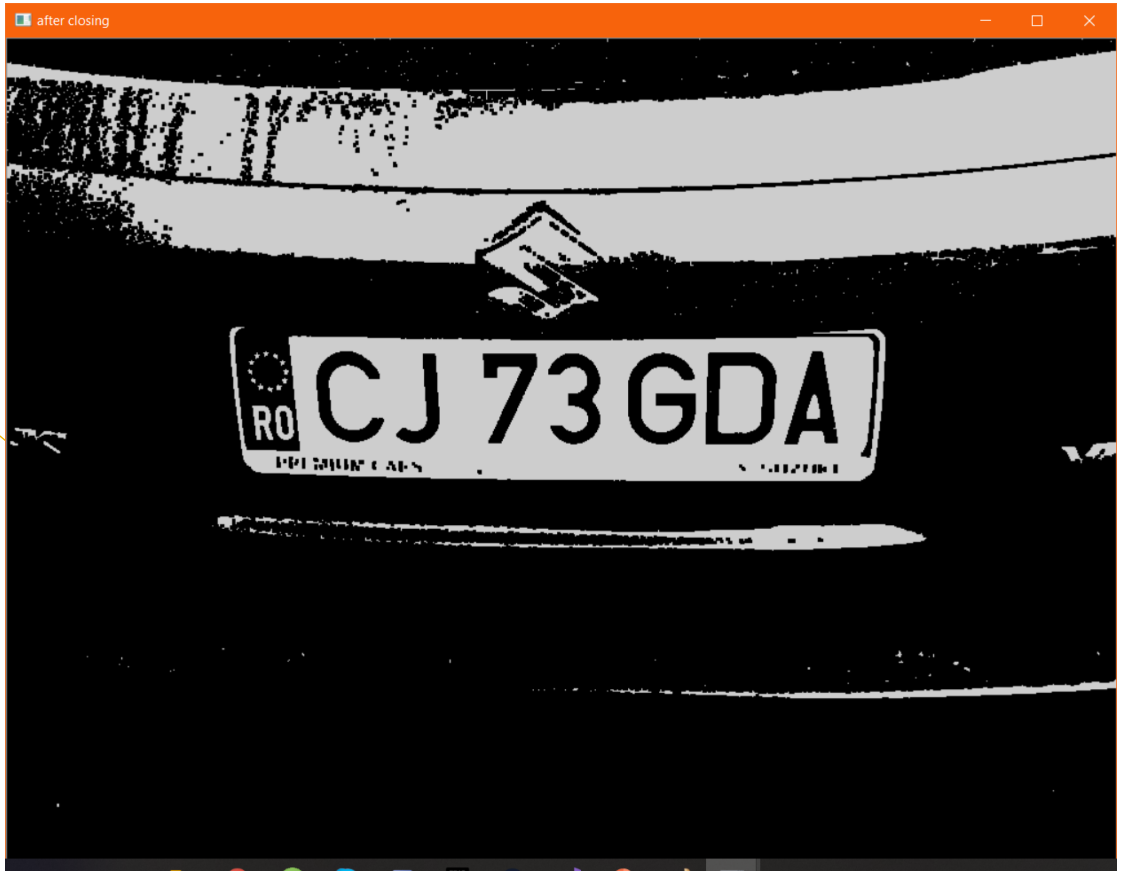
# The Algorithm

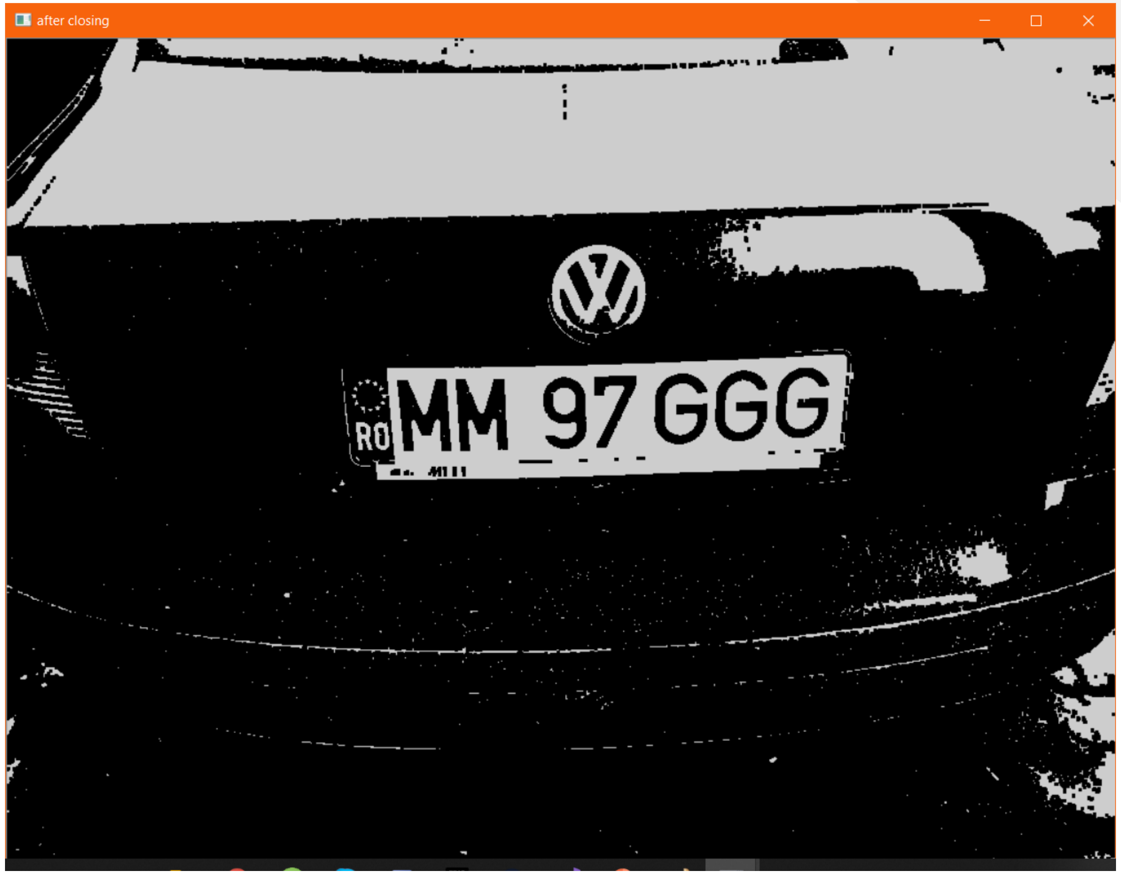
## Image Closing

- Closing, like its dual operator opening, it can be derived from the fundamental operations of erosion and dilation. Like those operators it is normally applied to binary images, although there are graylevel versions.
- Closing is opening performed in reverse. It is defined simply as a dilation followed by an erosion using the same structuring element for both operations.
- The basic effect of the erosion on a binary image is to erode away the boundaries of regions of foreground pixels. Thus areas of foreground pixels shrink in size, and holes within those areas become larger.
- The dilation is the opposite operation of erosion.
- We use closing to eliminate small random groups of pixels form the image, so we can enhance the view of the licence plate and the characters.
- **The main use for this step was to get rid of any small unwanted group of black pixels so we can apply labeling in an efficient way**

# The Algorithm

## Image Closing





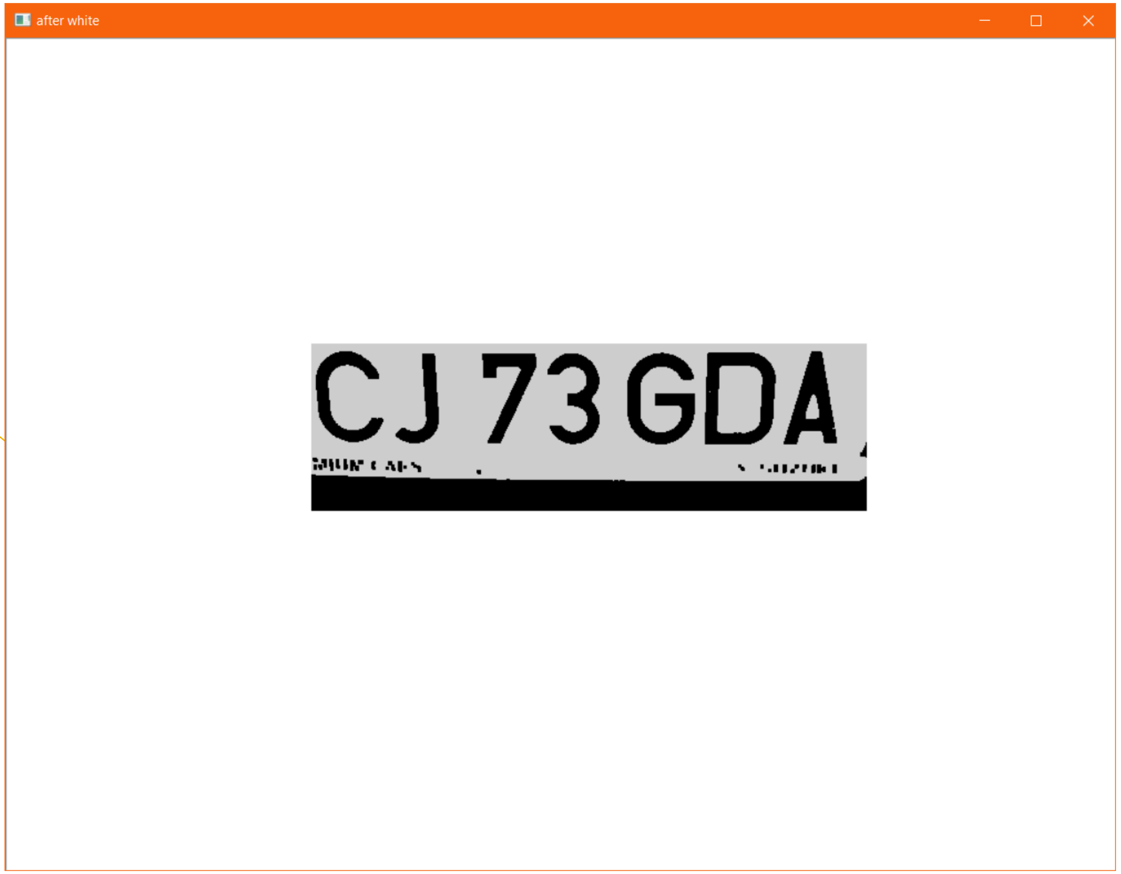
# The Algorithm

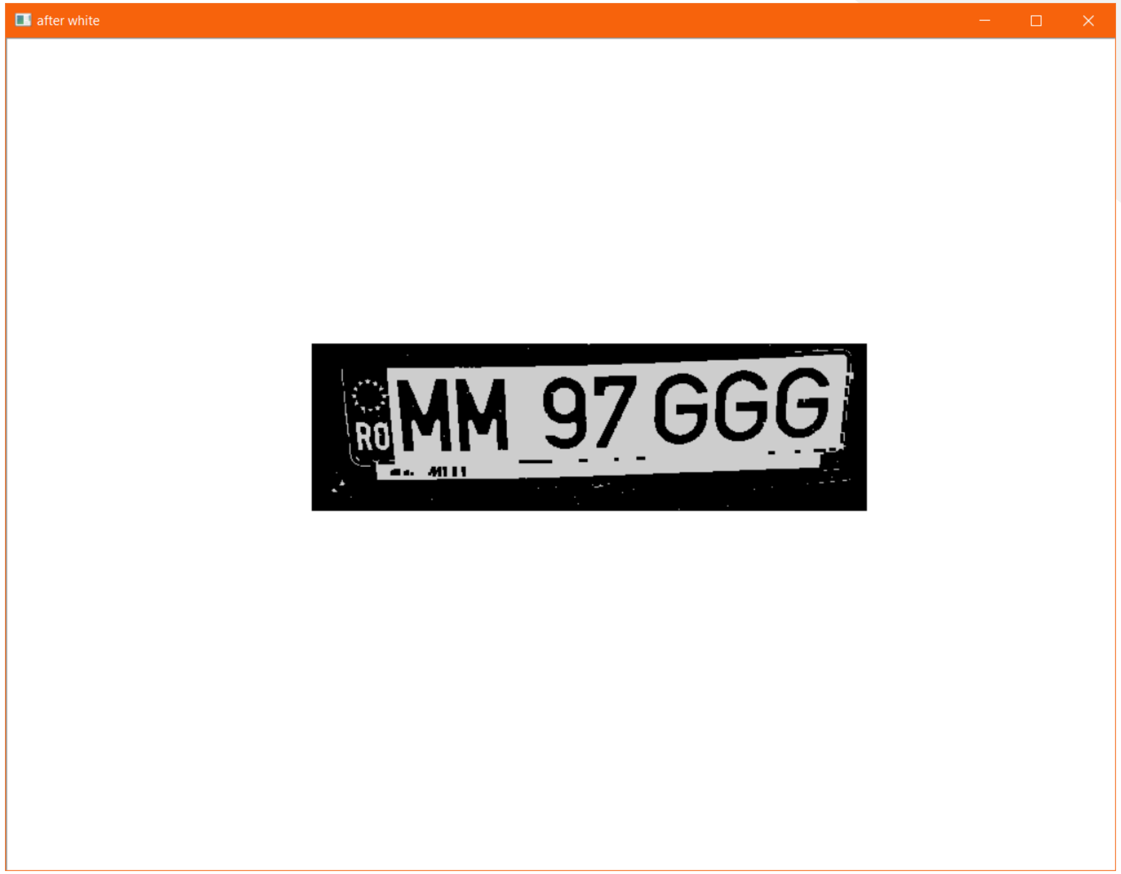
## Region of interest

- After we applied all the algorithms that could help us to improve the quality of the plate and at the same time remove unncesessary information from the image, we wanted to focus only on the portion of the image where the plate is located.
- We assumed that all the licence plates are located on the center of the image, so we took a bounding box that contains only that area. The rest of the pixels from the images, are converted to white pixels and the result is roughly only the plate.
- For the region of interest we approximated the center of the image and we took a custom length and width to contain each plate and at the same time to eliminate all extra pixels, for a better character recognition.

# The Algorithm

## Region of interest





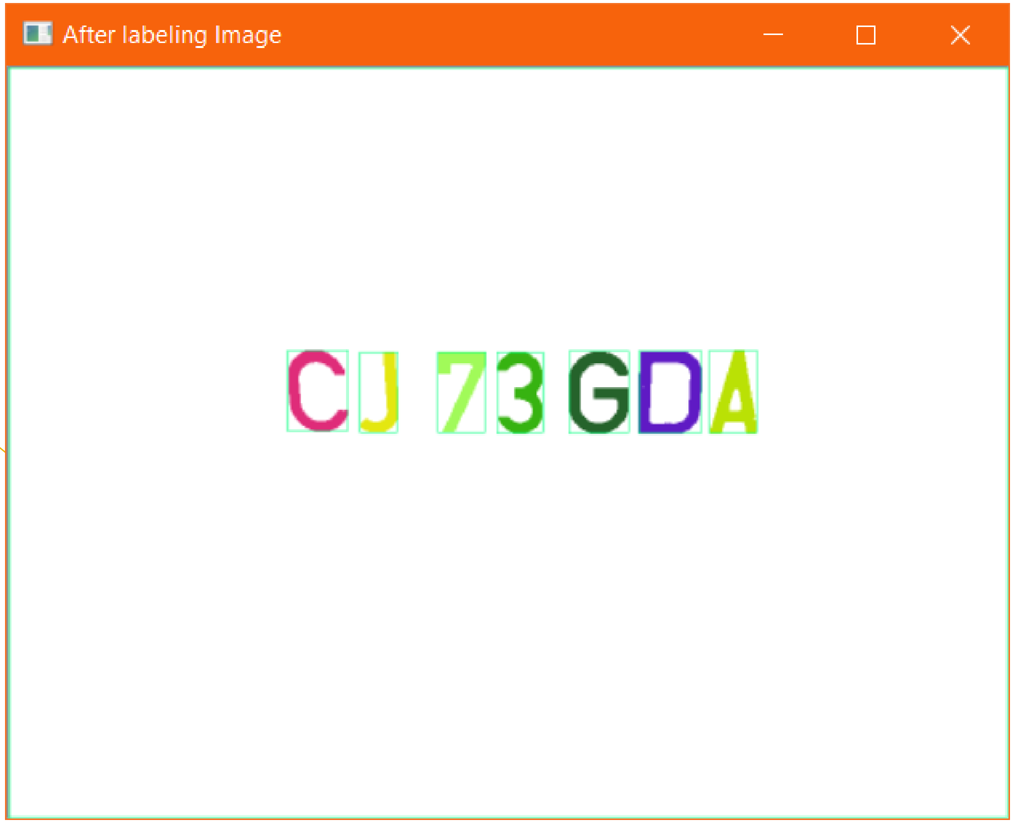
# The Algorithm

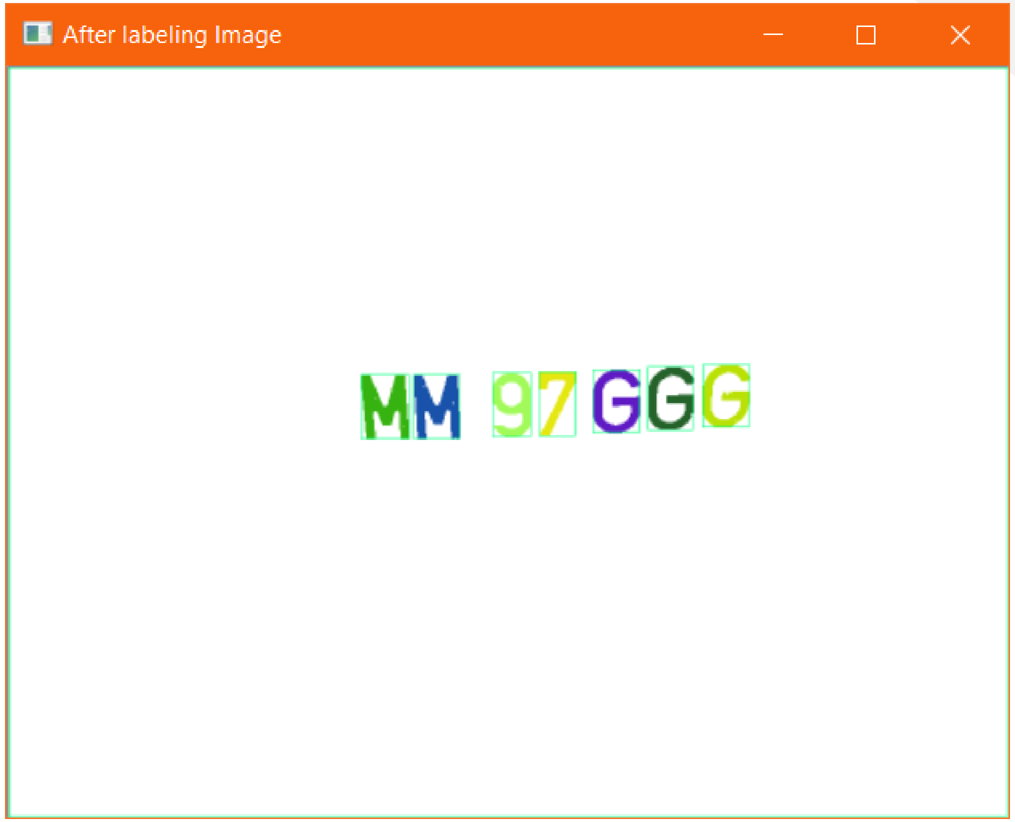
## Image Labeling & Rectangle drawing

- Now that we have the plate, we can go further and try to recognize the characters by labeling each one on the plate. We use breadth first traversal algorithm which is a straightforward method for labeling that relies on graph traversal of the image. This algorithm searches unlabeled objects, it gives it a new label and propagates the label to its neighbors.
- Because our bounding box isn't perfect and it contains other random pixels, in this step **we eliminate** the objects **that are too small or too large**, compared to a character on the plate.
- After each character is labeled, we can draw small rectangles around them, so we can see the algorithm worked correctly.

# The Algorithm

## Image Labeling & Rectangle drawing

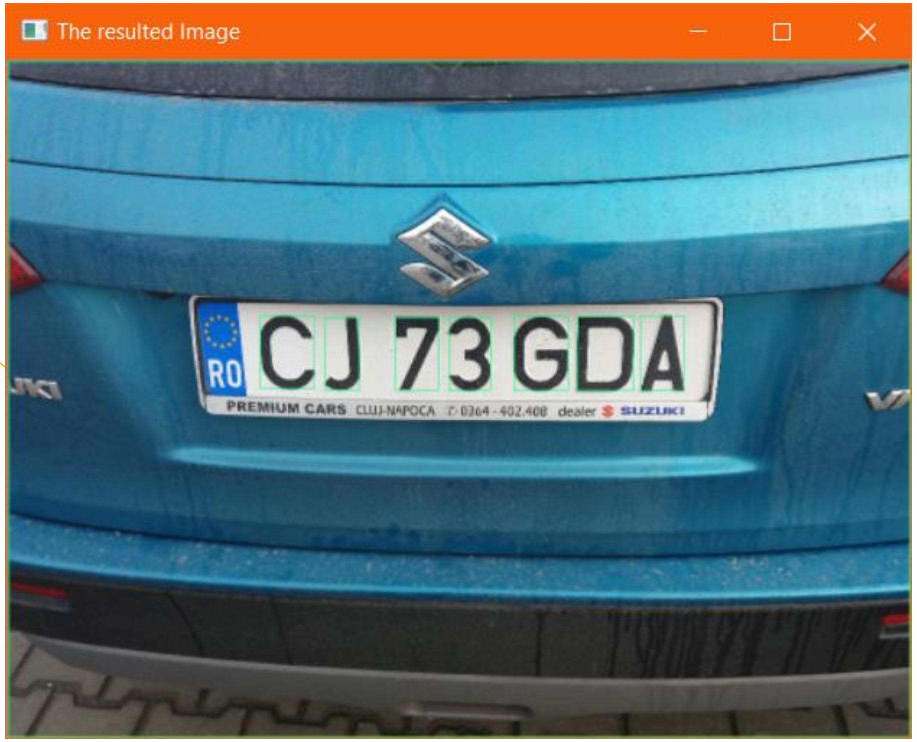


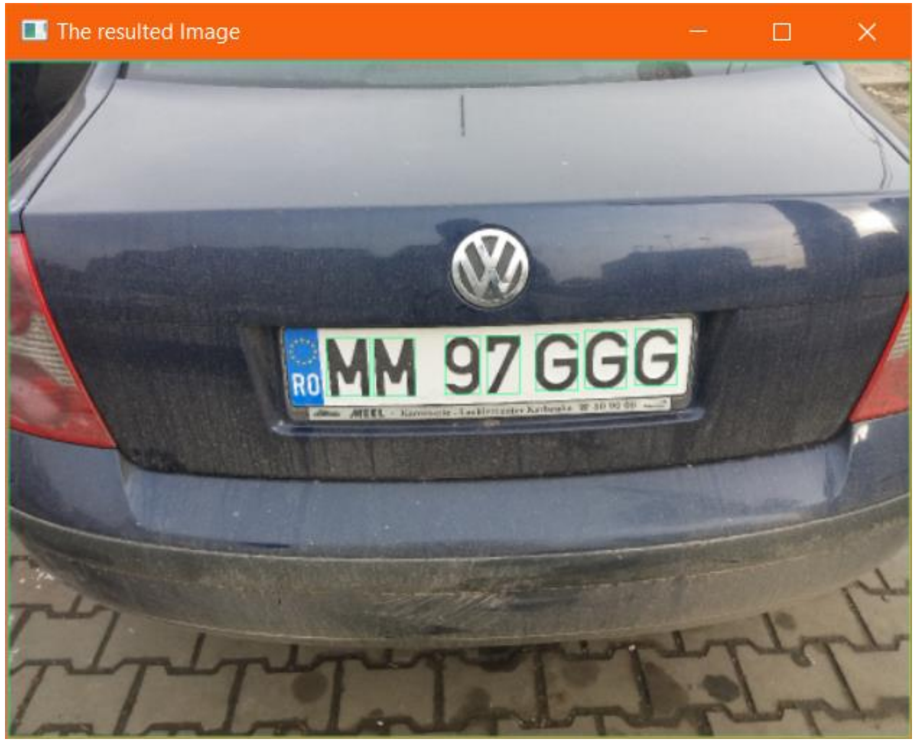


# The Algorithm

## Image Overlap

- After we obtained the rectangles, we can overlap the current image and the original image, so the rectangles are on the right location on the original image.





# Worse Scenarios

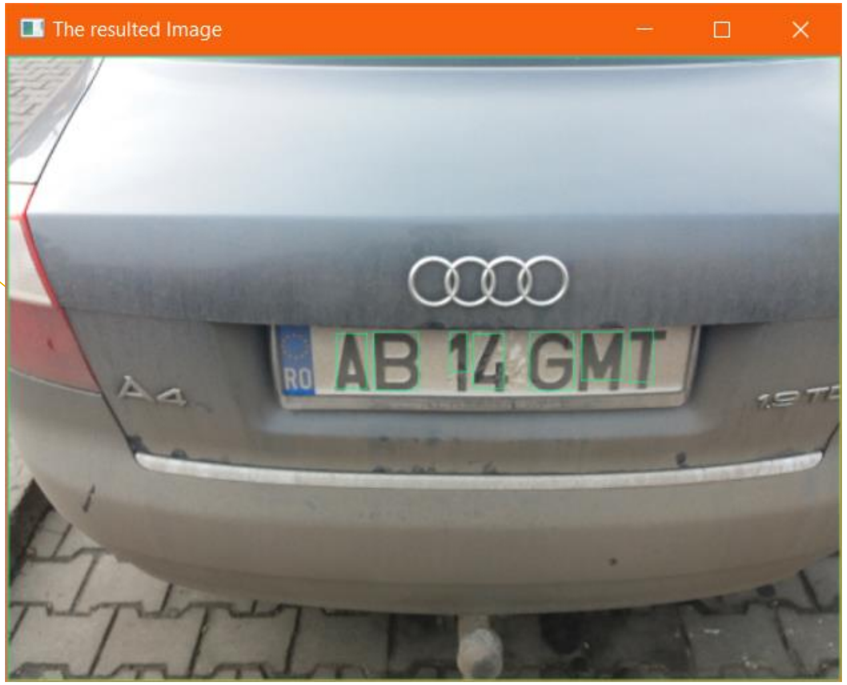
Results

- The problems with this approach start to rise when the licence plate is dirtier than usual. Our algorithm can handel dirtiness to a certain level, but after that, even human eyes can have a hard time to reconginez the letters on the plate.
- We tried different methods to enhance the image quality by convolution or histogram equalization but in the end we got the same results, although the intermediar results were better.
- We have two articular images that are worse but have different results: in the first one some letter are partially dirty and in the second one all the letters are dirty, and also the licence plate.

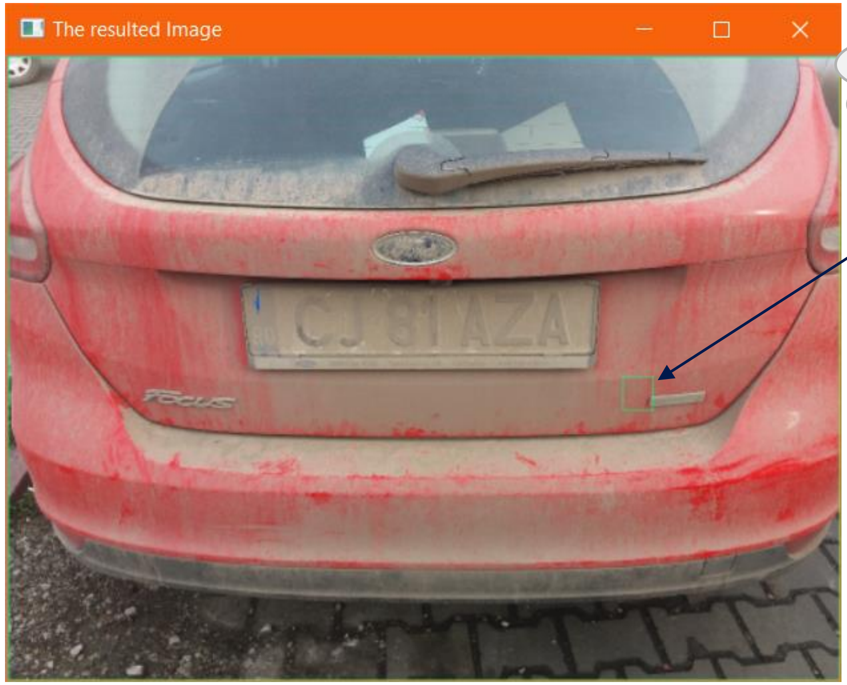
# Worse Scenarios

Results

## Partially recognized



## Completely unrecognized



The program thinks this is a character

# Worse Scenarios

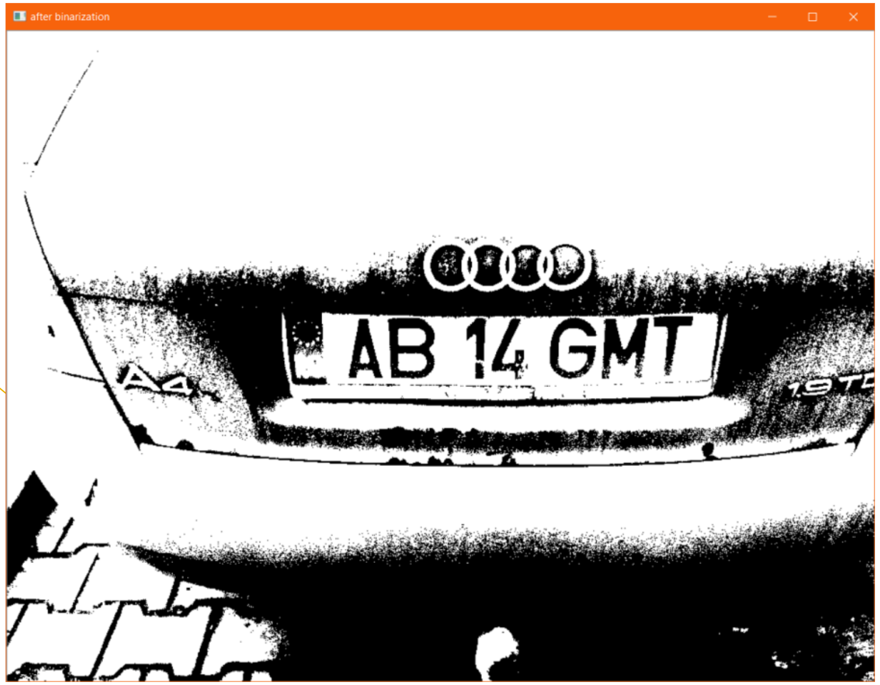
Results

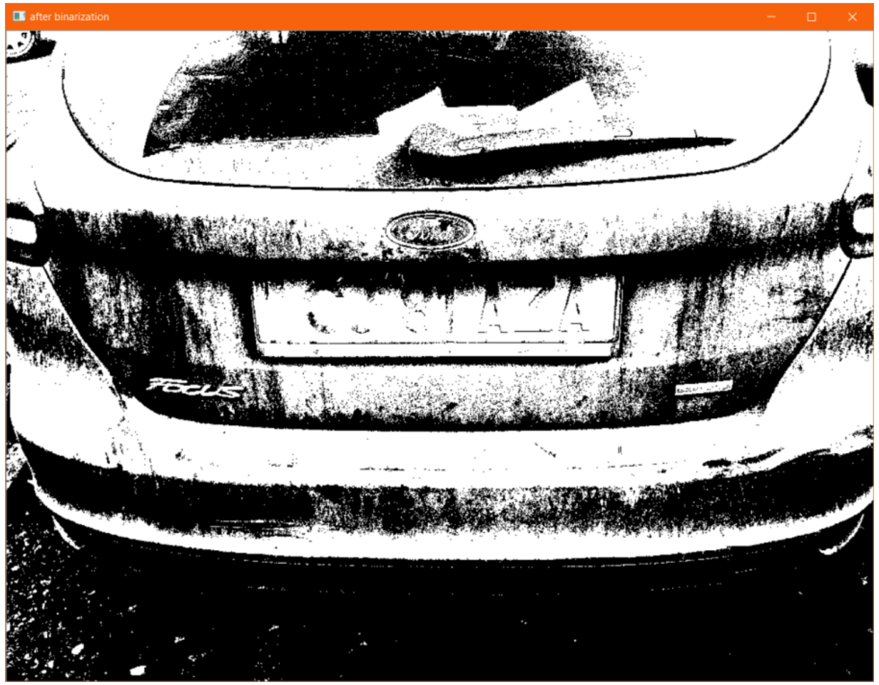
- Because the letters and the plate are approximately the same color, many information is lost after binarization so we can’t really distinguish and label each letters correctly. We tried with different thresholds for binarization but the results haven’t changed very much.
- Fortunately, we can turn this problem into a solution for another problem. Currently it is illegal to drive with dirty plates, so our algorithm could detect if a plate is too dirty and notify the authorities to take the appropriate measures.

# Worse Scenarios

Results

## After binarization





# Further Development

Ideas

- For further improvements of the algorithm, the commonly used solution is machine learning. This method is used wordwide to recognized all kinds of objects. Of course that will mean more code and a data set of images to train the algorithm, but it will have better results.
- As for better enhancement of the images, there could be more advanced tehniques that we could use, but we did our best to cover all the edge cases.
- Our solution could be used for different real life scenarios like: car parking ticket system for malls, traffic monitoring, speed ticket system and much more.

# Thank You.

Ossian Adrian & Pacurar Cristian

ossianadrian@yahoo.com &
pacurarcristian31@gmail.com

# Photos link for testing

https://drive.google.com/open?id=1E3NWAsIggLNgo1oUvcpPi4bvdRIEdELQ