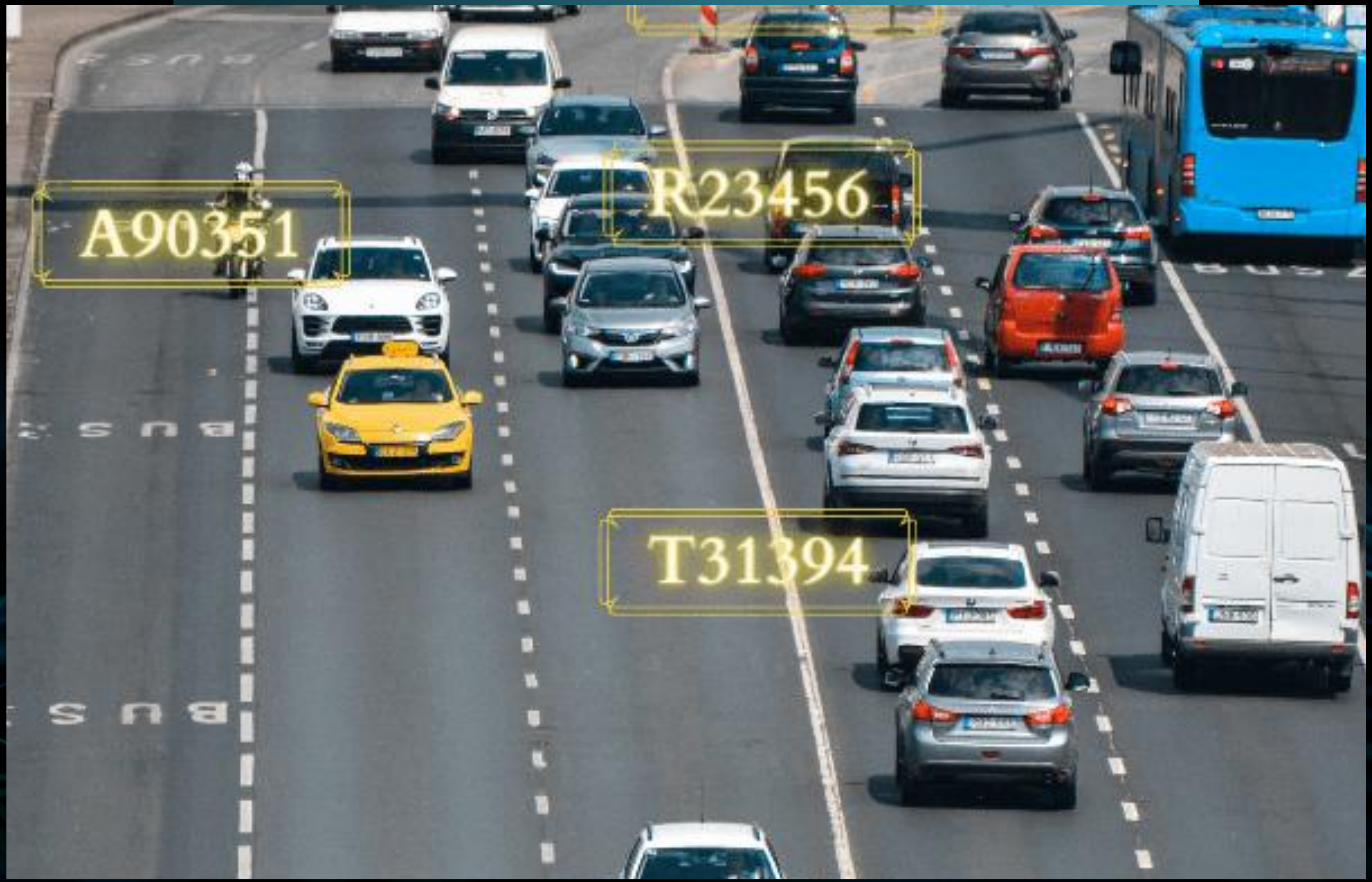
Automated License Plate Recognition

Detecting and recognizing car license plate

Licence Plate Number Recognition
for AI-powered smart parking automation
ABC 1234

Motivation and Problem statement

In modern smart cities, efficient vehicle identification and parking management are essential for enhancing security, reducing human effort, and improving operational efficiency, Traditional methods such as manual checking of license plates or physically monitoring parking lots are time-consuming, error-prone, and not scalable, With the rapid growth of image processing, computer vision and machine learning, automated License Plate Recognition (LPR) systems offer a faster and more reliable solution.



Targeting the car

01

First thing we do that we read and visualize the target car

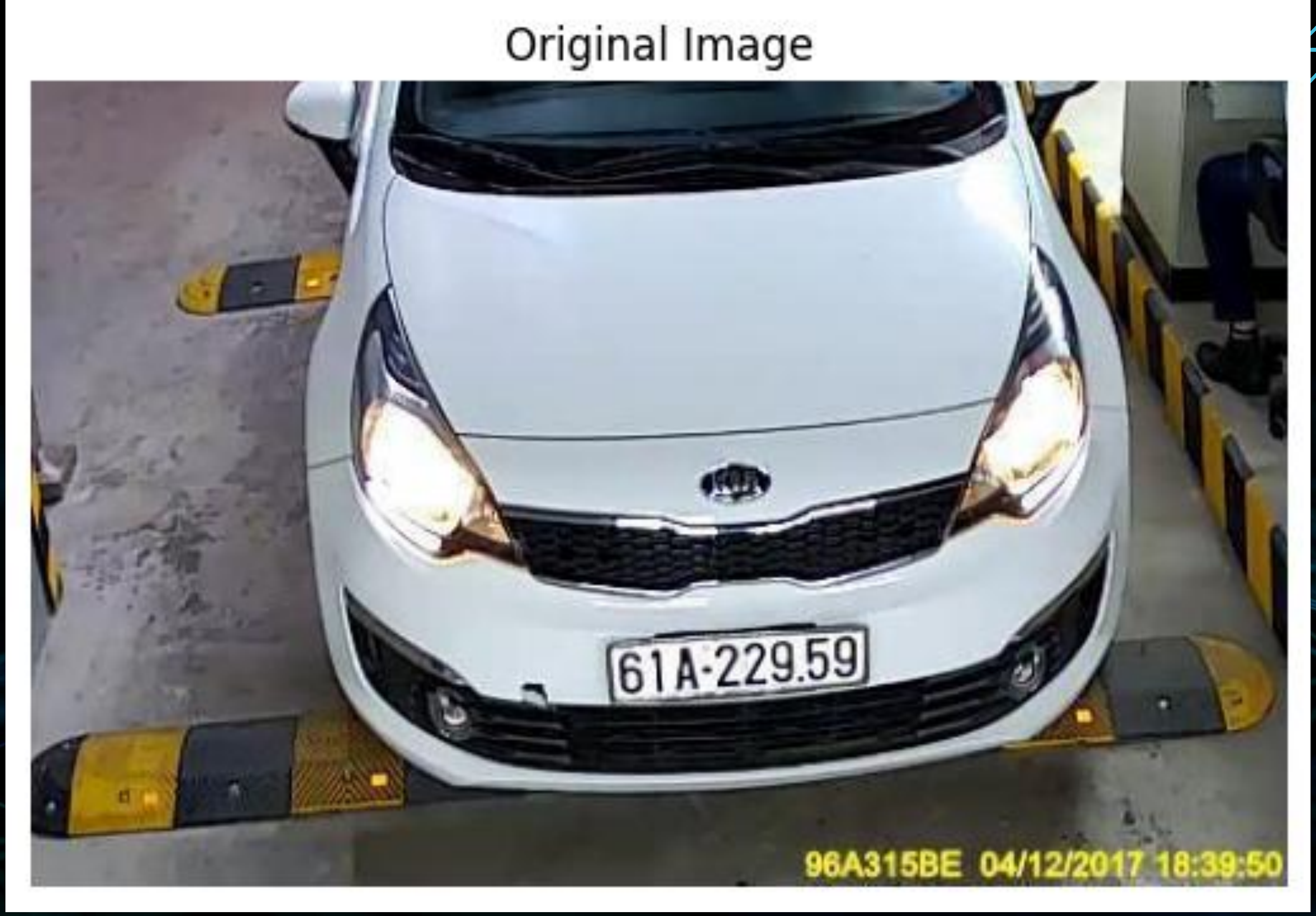
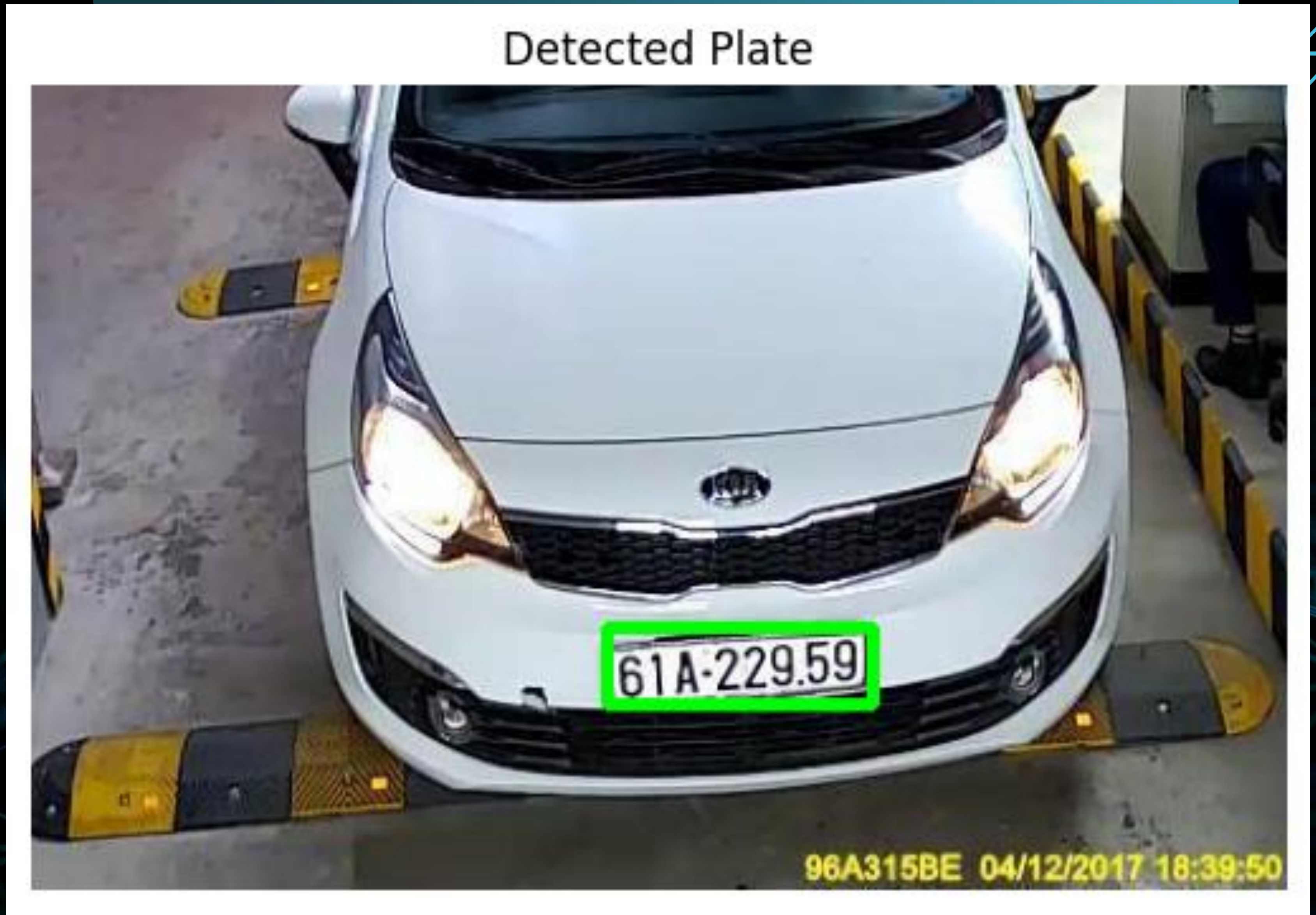


Plate detection

02

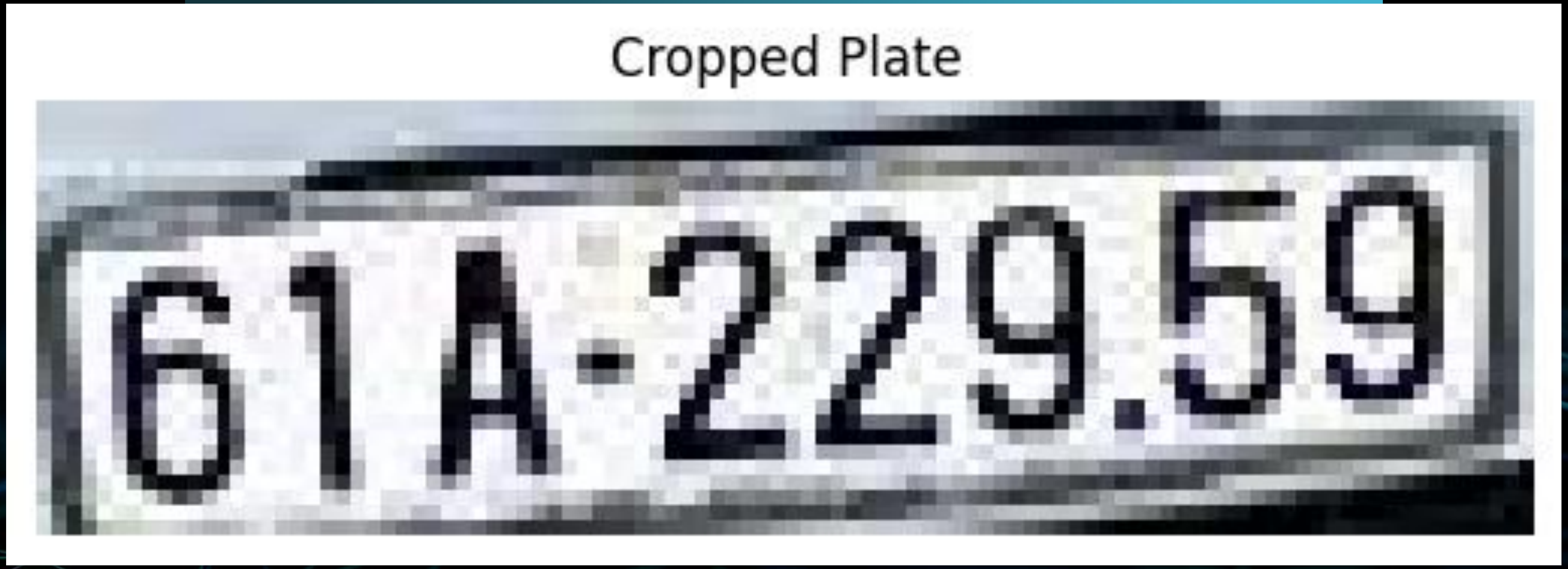
This is the stage where we used our plate detector model that we have trained before, this easily can give us the coordination of the detected plate so we can crop the plate later and make some image processing tasks on it.



Cropping the plate

03

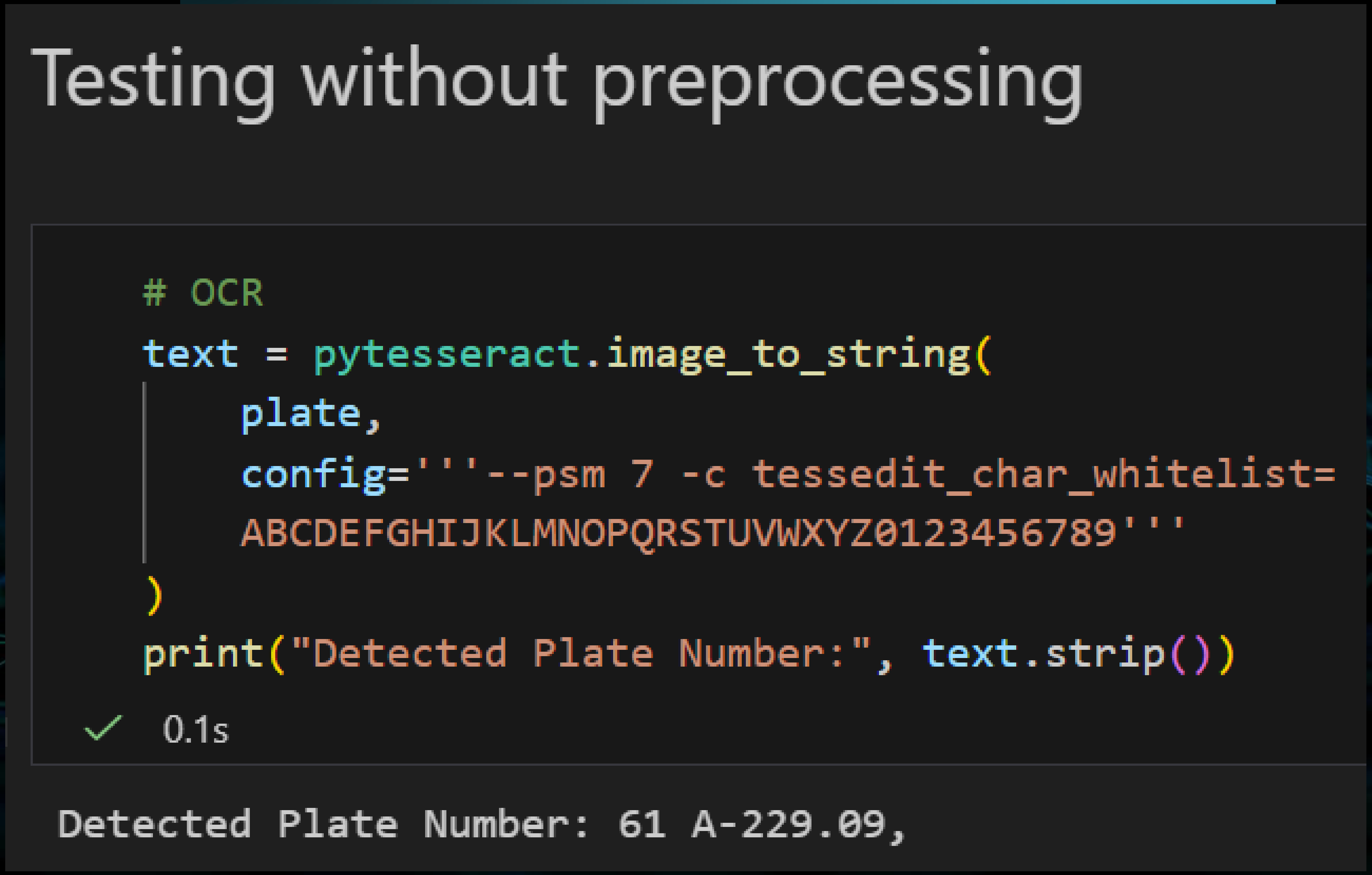
After our model detected the plate and its coordination has been known, it comes the phase where we crop this detected area so we can process and read what is written in it.



Testing without Preprocessing

04

If we try to read the image and extract the content of the plate before doing image processing the result will be like you see here some numbers were recognized wrongly.



Testing without preprocessing

```
# OCR
text = pytesseract.image_to_string(
    plate,
    config='''--psm 7 -c tessedit_char_whitelist=
    ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'''
)
print("Detected Plate Number:", text.strip())
```

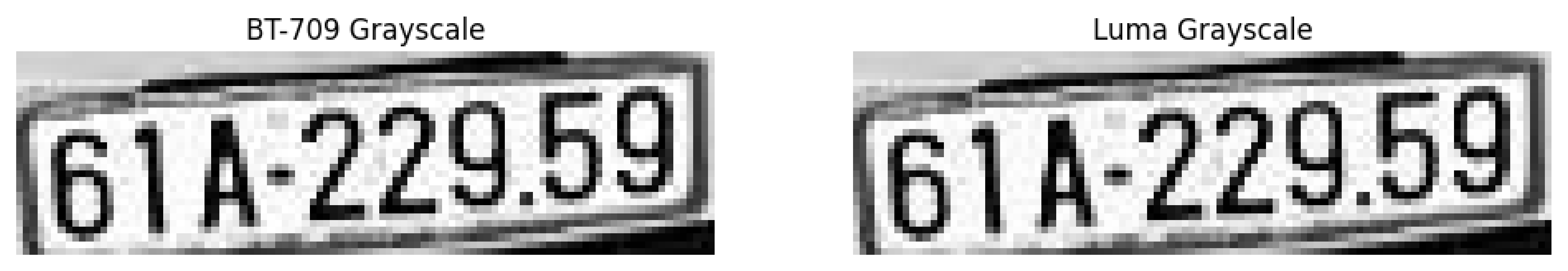
✓ 0.1s

```
Detected Plate Number: 61 A-229.09,
```

BT-709 and Luma Grayscale

05

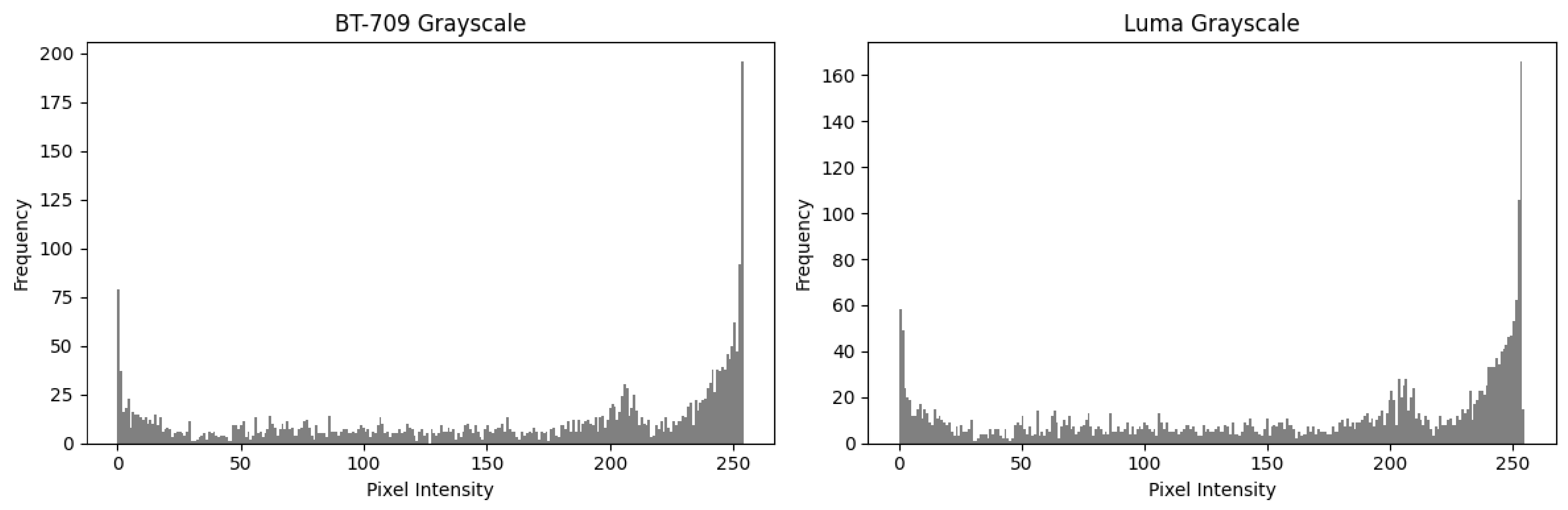
In this stage we will try to make some image enhancements, we will use BT-709 and Luma Grayscale to convert the color to gray.



Showing Histograms before

06

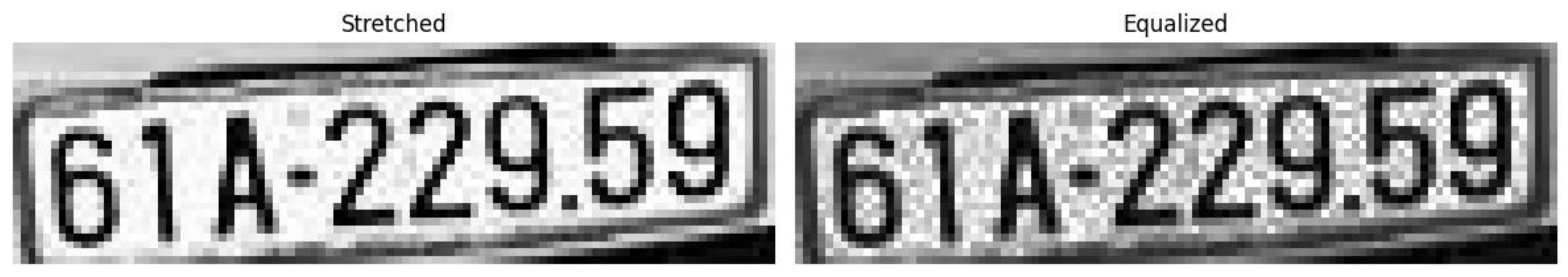
Showing the pixel intensity of the two images before doing Histogram stretching or equalization



HISTOGRAM STRETCHING AND EQUALIZATION

07

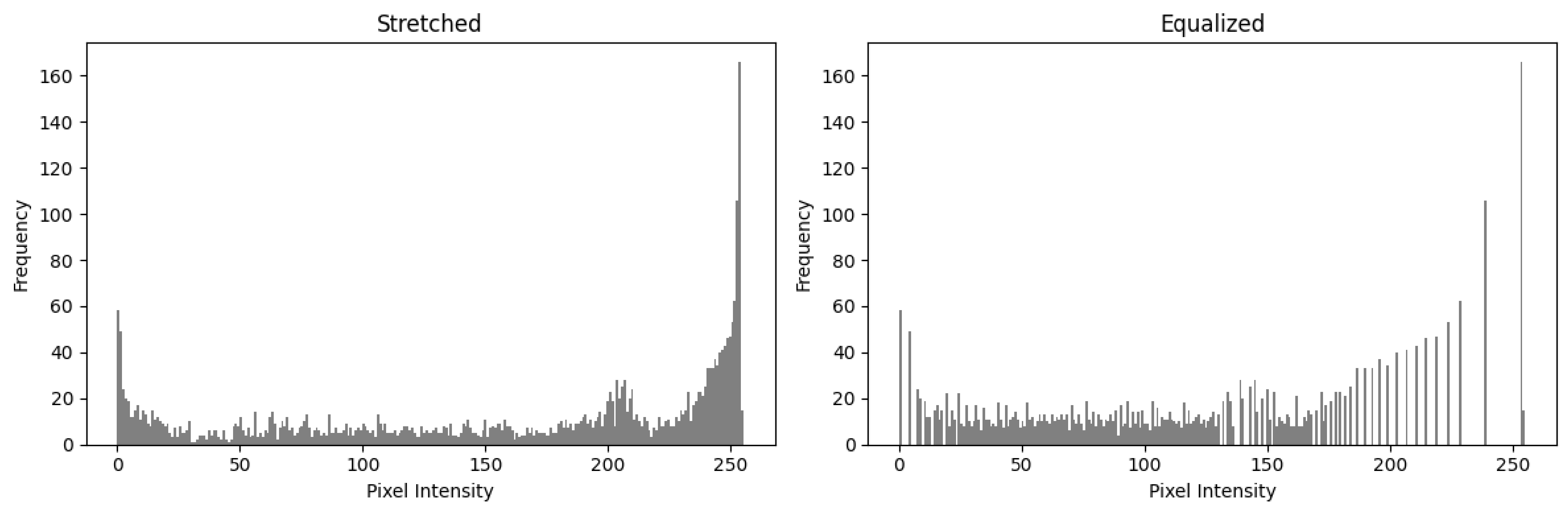
Here we tried to make some enhancements on the image and playing with its contrast by doing histogram stretching and equalization



Showing Histograms After

08

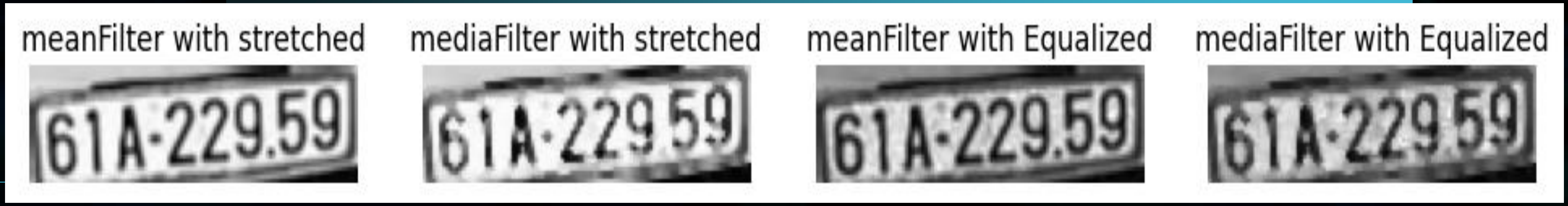
After we've done the stretching and equalization our histogram looks like this, we noticed some changes in the equalized one.



Mean and Median filtering

09

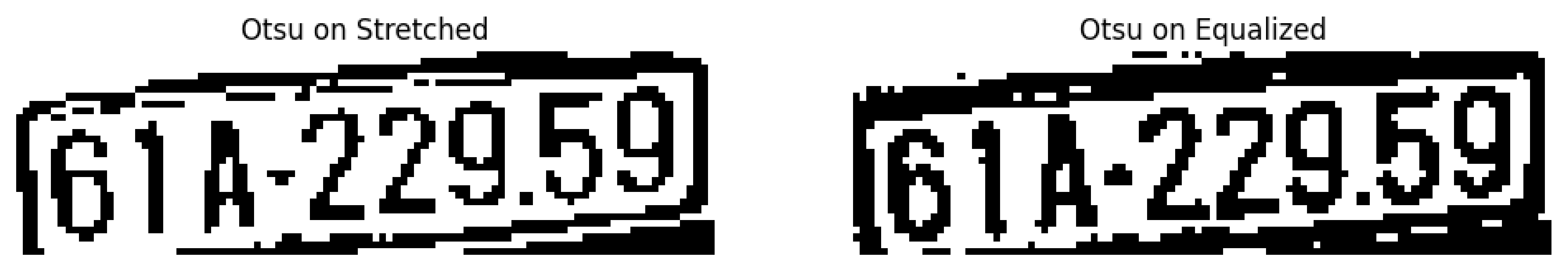
Here we tried to apply mean and median to both stretched and equalized one to reduce the noise.


meanFilter with stretched
61A-229.59
mediaFilter with stretched
61A-229.59
meanFilter with Equalized
61A-229.59
mediaFilter with Equalized
61A-229.59

OTSU THRESHOLDING

10

After we've adjusted the contrast it's time to convert the image to binary so we can easily read the plate, here we used OTSU Threshold.

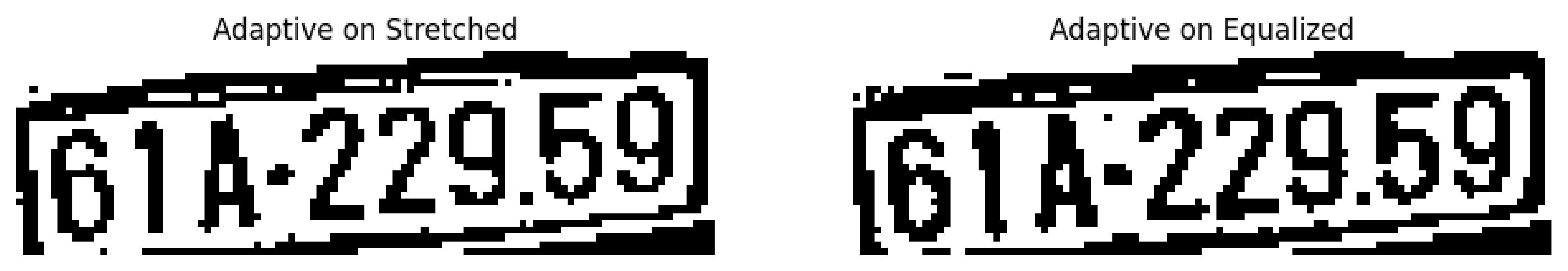


Otsu on Stretched

Otsu on Equalized

ADAPTIVE THRESHOLDING

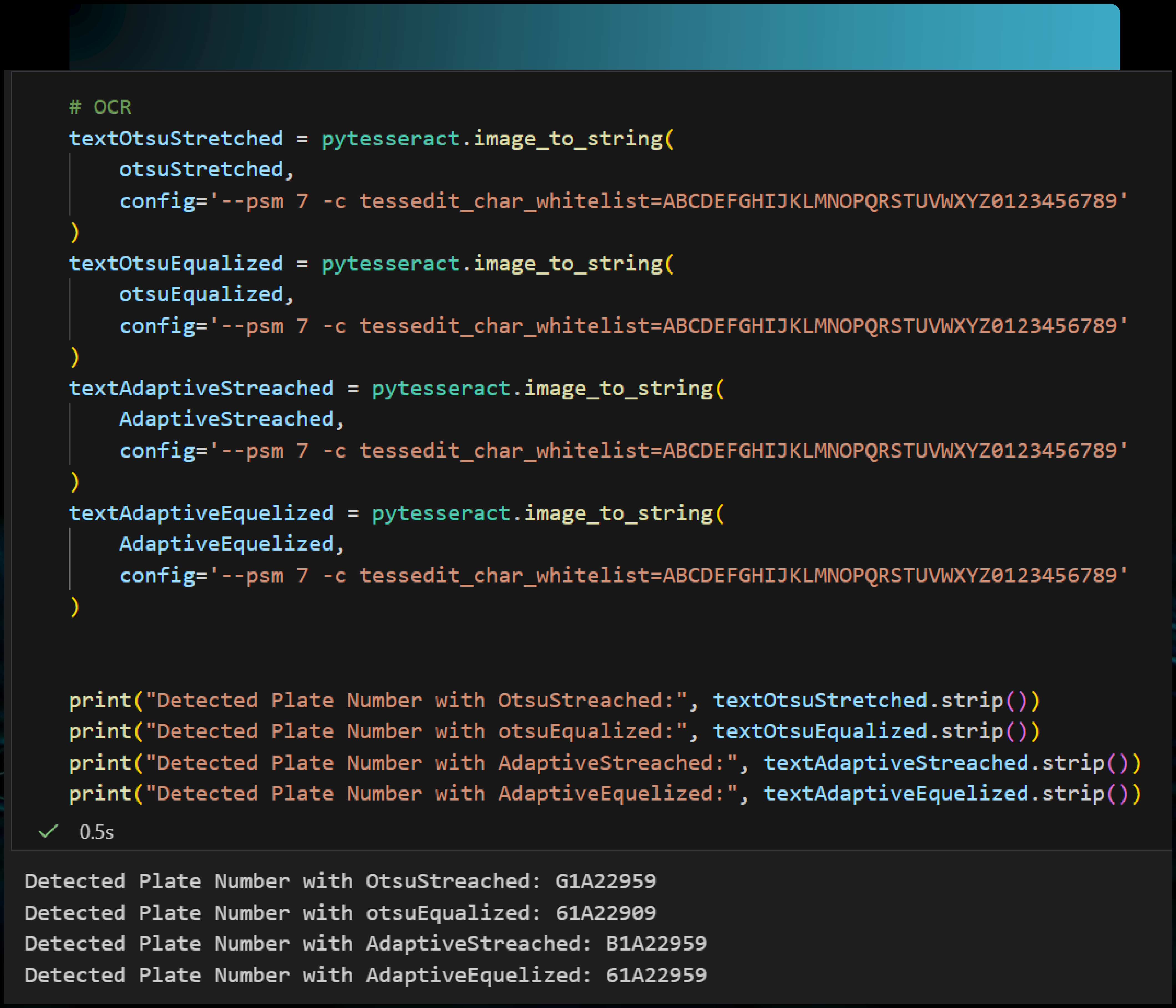
Here we tried to use another powerful Threshold which is Adaptive to convert the image to binary, so our plate now is ready to be red.



Final Test

12

After all the processes we have been through we tested all the four approaches and we see that only one has extracted the plate number correctly, which is Adaptive Equalized approach as it shows us the correct plate number without been mistaken even in one character.



```
# OCR
textOtsuStretched = pytesseract.image_to_string(
    otsuStretched,
    config='--psm 7 -c tessedit_char_whitelist=ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
)
textOtsuEqualized = pytesseract.image_to_string(
    otsuEqualized,
    config='--psm 7 -c tessedit_char_whitelist=ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
)
textAdaptiveStreached = pytesseract.image_to_string(
    AdaptiveStreached,
    config='--psm 7 -c tessedit_char_whitelist=ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
)
textAdaptiveEquelized = pytesseract.image_to_string(
    AdaptiveEquelized,
    config='--psm 7 -c tessedit_char_whitelist=ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
)


print("Detected Plate Number with OtsuStreached:", textOtsuStretched.strip())
print("Detected Plate Number with otsuEqualized:", textOtsuEqualized.strip())
print("Detected Plate Number with AdaptiveStreached:", textAdaptiveStreached.strip())
print("Detected Plate Number with AdaptiveEquelized:", textAdaptiveEquelized.strip())
```

✓ 0.5s

```
Detected Plate Number with OtsuStreached: G1A22959
Detected Plate Number with otsuEqualized: 61A22909
Detected Plate Number with AdaptiveStreached: B1A22959
Detected Plate Number with AdaptiveEquelized: 61A22959
```

Prof:	EFTAL ŞEHIRLI
StdName:	ABDULRAHMAN ALHALQI
StdNo:	2110205596

LINK

LinkedIn LINK