CMPS327
COURSE PROJECT REPORT

Name of Project:
License Plate Recognizer

Presented By:

Omar Shalabi **ID:202100157**
Mohammad Sidani **ID:202100611**
Jad Ayoubi **ID:202100088**

Submitted to:
Dr Lama Affara

Contents

Project Name & Members:

Project Name: License Plate Recognizer

Project Members:
-Mohammad Sidani
-Omar Shalabi
-Jad Ayoubi

General Overview:

"License Plate Recognizer" is a python programmed application with a Graphical User Interface (GUI) created from scratch. The user is asked to enter an image of a car. This image is then contoured and recognized. After the recognition process, each character will be displayed in an entry box. The main idea of this program is to design and develop effective image processing techniques and algorithms to localize the license plate in the captured image, to separate the characters from that number plate, and to identify each character of the segment by using the open computer vision library (OpenCV) and EasyOCR Library. Many applications can be implemented by using this system, such as a security surveillance system, highway speed

detection, violation of traffic light, discovery of stolen cars, and automatic fee collection systems.

Discussion:

After tireless efforts of searching and googling, we came to a final decision concerning the idea of our project. Our train of thoughts began to surface in a Zoom meeting between the 3 members present. We all agreed on creating a license plate recognizer with a GUI, listed all methods and functions to be used in the project then divided the load among the three of us.

Inspiration:

With all possible ideas we could have settled on, we believed that such a project could hold lots of image processing techniques and ideas that we could implement. From basic blurring to advanced and complex text recognition, this project cycled through most of the concepts and knowledge we acquired during this course. The seamless experience of the user was also a main concern. To counter any possible unpleasant experience, we decided that adding a graphical user interface would improve our project and make it easier for anyone using it. Finally, the numerous applications that could be based on a license plate

recognizer keep a large margin of creativity and ideas to be used later on in a more advanced beneficial product.

About The Application:

The user is welcomed with a beautiful GUI and is prompted to select an image of his choice. Upon the selection of an image, the user clicks the “recognize” button and within a few seconds, the numbers and letters of the license plate will be displayed in a sequence of boxes located below the original image. This brief description explains the process on a high level, where the detailed description will be presented in the following pages.

Division Of Work:

Every part of the project was always completed upon the approval of all project members where we communicated through Zoom and WhatsApp. Mohammad and Omar were concerned with designing the GUI and learning then necessary concepts and syntax to complete this task. Jad handled most of the plate recognition code where he followed a pipeline of image processing techniques used to recognize the plate at the end. Then, Omar and Mohammad helped with the debugging of the plate recognition code after understanding the full implementation.

Finally, Omar mapped each function to its corresponding GUI widget with the help of group members.

Graphical User Interface (GUI):

As mentioned earlier, A clean and efficient user interface was our target. We chose Tkinter Library for building the UI for a couple of reasons, like its simplicity and the huge number of online resources that provided a great start for us to learn the implementation from scratch. We started by setting the application window to 1000x600 and making it unresizable to give a unified look. The window icon and name were also changed to fit the context of the project. Using the pillow library, we imported an image to make it the main background image. This process was also done using Canvas. Canvas were used throughout the entire UI to hold all other widgets such as text, buttons, images…. Then a colorful cover image of Lebanese car plates was placed in the center which would be replaced with the image that the user selects. On the right side, two buttons are placed. “Recognize” button is initially disabled until an image is selected by the user, the clicking of this button calls the Recognize method which we will walkthrough in the following page. “Exit” button serves as a terminate button after displaying a window to ask the user if he really wants to quit. This was achieved using MessageBox package. If for any reason the program

could not terminate, an error message would be displayed. Moving on to the image selection, a Selection button below the image in the center is used to open the user's file system upon clicking. The select method is called where only JPG and PNG image formats are accepted. The mentioned method was made possible using the "filedialog" package. Finally, the path of the selected image is displayed in an entry box above the selected image. (The first screen of the GUI is shown below)

Recognition:

Libraries Used:

We used several libraries and packages to get the best results possible:

EasyOCR: EasyOCR is a Python package that allows computer vision developers to effortlessly perform Optical Recognition. EasyOCR is by far the most straightforward way to apply Optical Character Recognition: The EasyOCR package can be installed with a single pip command.

CV2: cv2 is also known as OpenCV, is a great tool for **image processing and performing computer vision tasks**. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.

1-Edge Detection

Bilateral filter: OpenCV has a function called bilateralFilter(). We used this filter to smooth the image and reduce the noise by passing the grey image we had already displayed above it

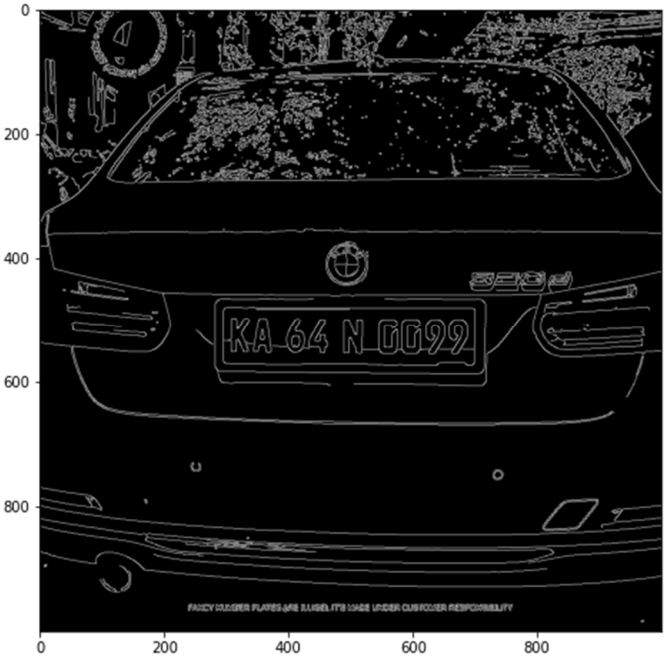
After that we used:

Canny edge detection: The Canny edge filter is an edge detection operator. This allows us to detect edges. It started to become clearer to us where exactly the number plate actually is.

Canny filter progresses through the following pipeline:

1. Suppress Noise
2. Compute gradient magnitude and direction
3. Apply Non-Maximum Suppression
4. Use hysteresis and connectivity analysis to detect edges

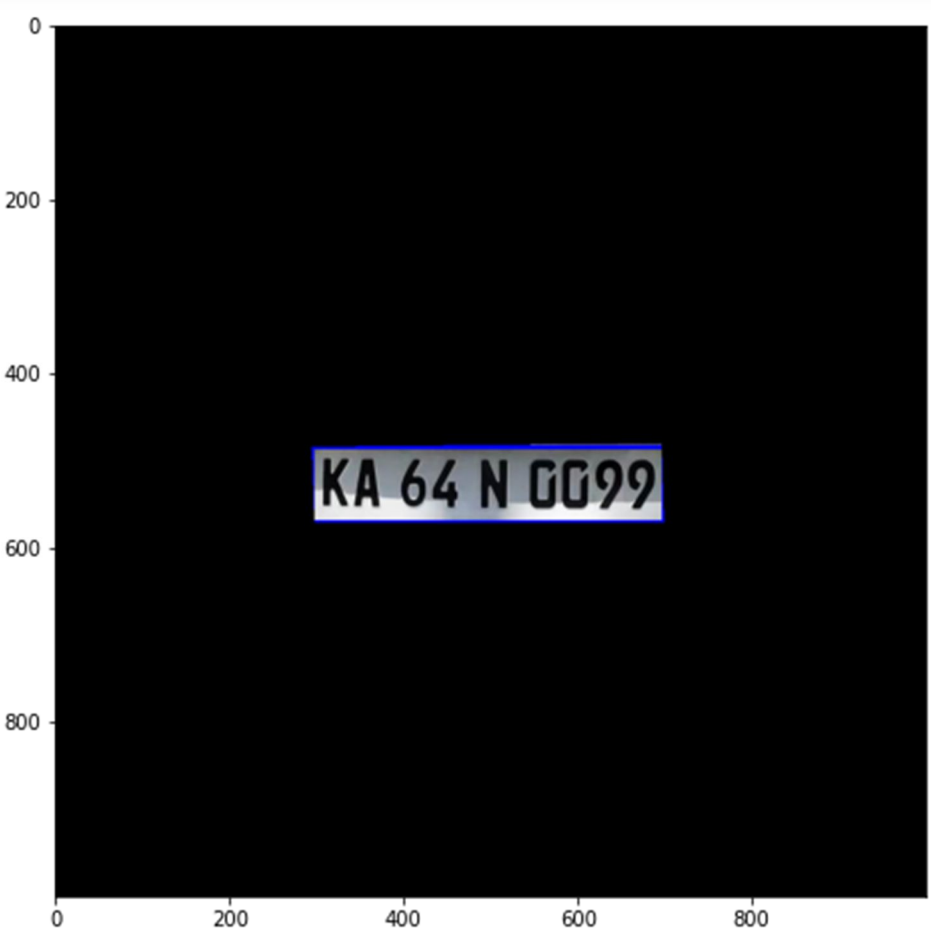
The Result:



2-Contouring:

Contours is a Python list of all the contours in the image. In our case it detects where the number plate lines are and detects polygons in between these lines. We are looking for is a contour which has 4 points so ideally, we want to be able to see a rectangle because that is most likely to be the shape of our number plate. We achieved that by using cv2.findContours which basically goes through our image and try to find the shape (contour).

After location of our number plate, we applied some masking and isolated the car plate by applying our contour search as shown below:



After that we decided to rather than having a huge blank in the image, we isolated the car plate because it makes it a lot easier when we pass it to the EasyOCR reader. As you can see below, we cropped our image so we get an image that purely represents the car plate.

3-Implemenation Of EasyOCR

After contouring we used EasyOCR to read what is written on the car plate using EasyOCR.read and pass the language we are going to use in addition to EasyOCR.readtext method. This outputs an array that includes the plate's location and what are written on the car plate.

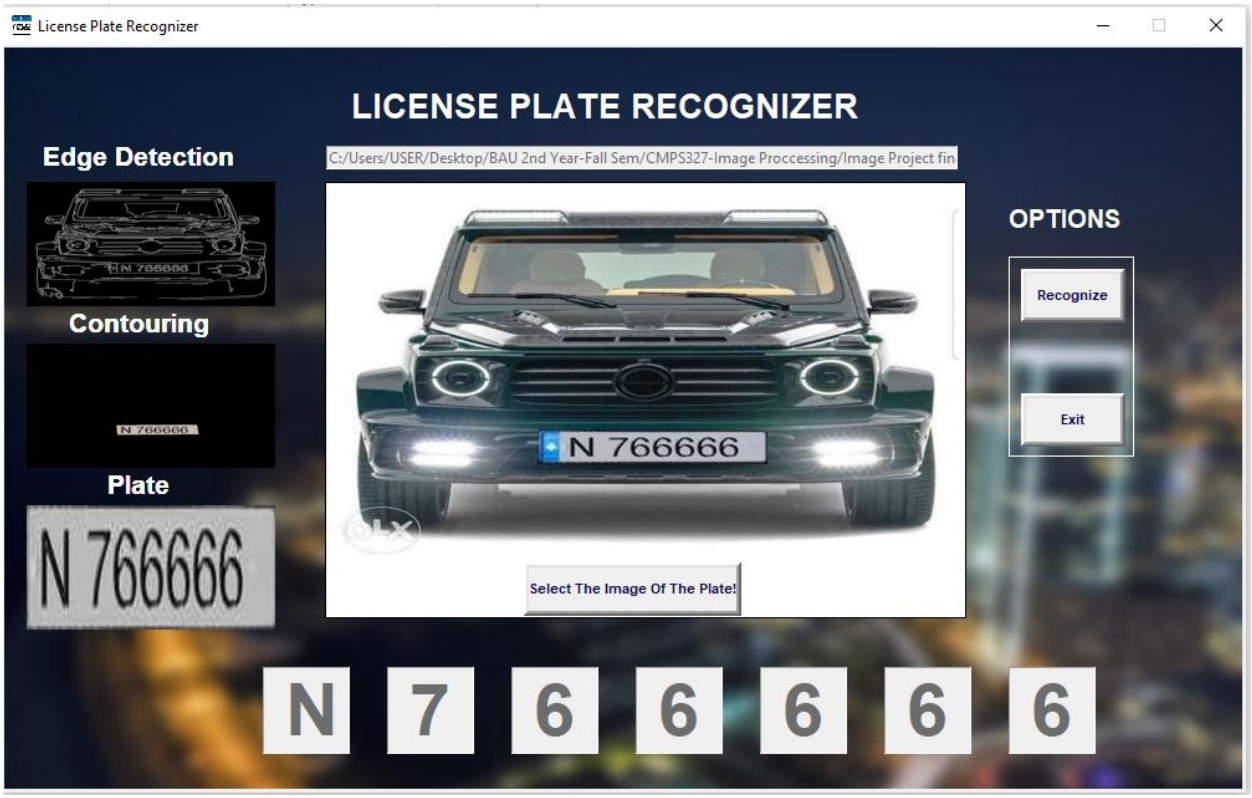
4-Recognition And Display

After reading the plate number, we identified the result we got above as text, then set font size, color, and scale to the text. Also, we added a rectangle shape around the plate after setting its thickness, color, and line type. The selected font is used to put the number of the plate under the car plate on the original picture alongside with the rectangle and finally we visualize that using plt.imshow (using imutils library) and convert it from BGR to RGB to get clearer image as shown below

Final Product:

We achieved our goal of making a License Plate Recognition program, which allows the user to enter an image saved in his device then recognize the plate number in the selected image, using Python and Tkinter library for GUI. This project visualizes the interaction between a human and a computer to experience this advanced technology of Image Processing . We can say that we are proud of what we have achieved and the final results are satisfying .

Final Screen:



Individual Benefits:

The secret behind every successful project is teamwork and cooperation between team members . We learned how to make a program with a Graphical User Interface, enhanced our understanding of programming concepts using Python, and enriched our knowledge in Image processing through visualizing many methods such as edge detection, localization, and contouring. Each member of the three of us contributed in his own way to further advance the production of this

project. That is what really matters in teamwork, where this chemistry between us is what made our efforts fruitful.

Resources:

-GeeksForGeeks (Canny edge and bilateral filter implementation)
-YouTube (Codemy Tkinter course)
-W3Schools(Python syntax and keywords)
-StackOverFlow (debugging)
-Youtube (Nicholas Renotte Channel)