Nile University

License Plate Recognition

Aya Elneanaei Fouda	202002609
Nourhan Mohamed Yossef	211000628
Nada Radwan	211001128

Digital Image Processing Project

Supervised By:

Dr. Mostafa El-Atter

1.Introduction

Although the first automated system for recognizing car license plates (LPR) was introduced in the 1980s, numerous commercial systems emerged during the 1990s. Despite the abundance of LPR systems available on the market, research and development in this area persist, leading to the creation of new advanced methods for plate localization, character segmentation, and recognition.

Car license plates are now employed for various purposes, such as traffic control, border control, access control, parking time and payment calculation, searching for stolen cars or unpaid fees, and the need for accurate identification under different lighting conditions, the presence of random or structured noise in the plate, and nationality-specific features such as the plate's size and character order. To detect car plates more efficiently and eliminate human errors that may occur during manual procedures, we have developed an image processing algorithm that detects car plate characters with minimal error probability. Our implementation employs several techniques, including segmentation, adaptive thresholding, grayscale conversion, negative transformation, morphological operations, and contrast enhancement. By combining these image processing mechanisms, we have created a dependable tool for plate detection.

2.Methodology

1. Dataset

The training and validation are done using a letters dataset that is used for ALPR (Automated License Plate Recognition) character training. The dataset is divided into training and validation sets. Each contains 36 classes: 10 numbers (0-9) and 26 letters (A-Z). The training set has 36 classes, each containing 24 images with a total of 864 images. As for the validation set, 36 classes, each containing 6 images, with a total of 216 images. The images in each class are of the same letter but with different shapes and slight changes to cover the biggest variables when it comes to tested models. The testing dataset "Car License Plates" was found on Kaggle with the source dataset uploaded on Make ML. It contains 433 car images with their plates from different angles and distances, both front and back plates, each labeled with bounding box annotations. The annotation is in Pascal VOC (visual Object Classes) format, used for object detection datasets.

2. Libraries

The libraries that were used throughout the code were:

• **OpenCV**: Open Computer Vision is an open-source library that contains numerous computer vision algorithms that are used in video analysis, image processing, object detections and other applications that could be further complex.

• **NumPy**: provides the necessary mathematical built-in functions for numerical computations for various applications and on multidimensional arrays.
• **Matplotlib**: A wide library that provides visualization tools for data in python. A module of matplotlib is a pyplot, that uses graphical functions similar to MATLAB.

• **Pytesseract:** Python-Tesseract is an Optical Character Recognition (OCR) library; that contains the right tools and algorithms to electronically convert input manually-written texts or images.

• **PIL**: Pillow is a Python Imaging Library (PIL) that adds tools and algorithms to manipulate, edit and process images. The "image" module contains image-editing functions. As for the training, the libraries user were:

• **Sklearn:** a library that provides unsupervised and supervised learning algorithms, which is also built upon already-existing well-known libraries such as Numpy and Matplotlib.

• **TensorFlow**: used to build, train and process datasets and models for Machine Learning (ML) and more.

• **Keras**: one of the most used deep learning frameworks. It is an API for neural networks and is used as an interface for TensorFlow.

3. Processing
The idea is to use the libraries chosen to create a model that trains, validates and tests on the datasets used.
The training and validation are done on the letters dataset only to create a module that identifies the classes.
The main criteria the model is built on is the F1 score, which could be identified as the harmonic mean of
precision and recall:

$$F1 = 2 \times (\text{Precision} \times \text{Recall}) / \text{Precision} + \text{Recall} \qquad [1]$$

After training the model, the testing with the plates dataset is done through the following steps:
• The input image is read then converted using CV2 to grayscale.

• The contrast of the image is increased. Instead of directly using a contrast filter, morphological top/white hat (see eq. 2) and bottom/black hat (see eq. 3) operations are used. The functions can be explained as follows:

$$Tw(f) = f - f \circ b \qquad [2]$$
$$Tb\ (f) = f \bullet b - f \qquad [3]$$

where f is the original image and b is the structuring element used. Recall that ∘ represents the opening operation and the • the closing operations.

• A gaussian blur filter is then applied, then afterwards gaussian adaptive thresholding is done to convert the gray image to a binary one with desirable contrast.

• Boundary extraction is done by finding and defining rectangular contours for each object, with different sizes based on the object's original size.

• Elimination of contour boxes by size is done through a loop to check every box in the image if it meets the defined chosen size.

• Later, more boxes are eliminated by box arrangements. Plate objects' contours are supposedly consecutive in the image and thus are arranged closely to each other. Hence a specific distance is defined in a loop, where each object is compared with the surrounding contours and eliminated if the distance chosen is exceeded.

• After the elimination, the contour boxes are added to the original plate image.
• Rotation with specific angles is done in case the original image and plate is taken from an angle,to guarantee character clarity.

• Gaussian blurring and thresholding are reapplied to clarify the extracted plate characters. The negative of the image is taken as an extra step just for elaboration and to affirm the extracted characters.

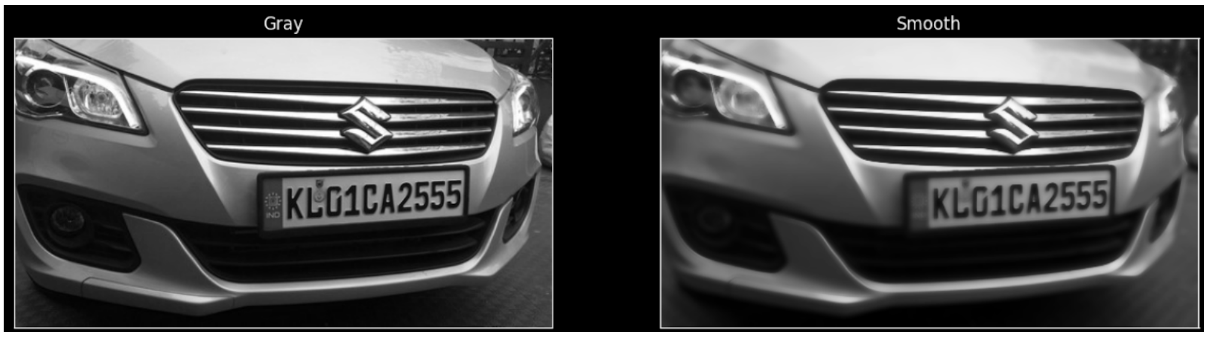
• Prediction is done by comparing each extracted character with the ones in the model and mapping the actual segmented vs the predicted letter.

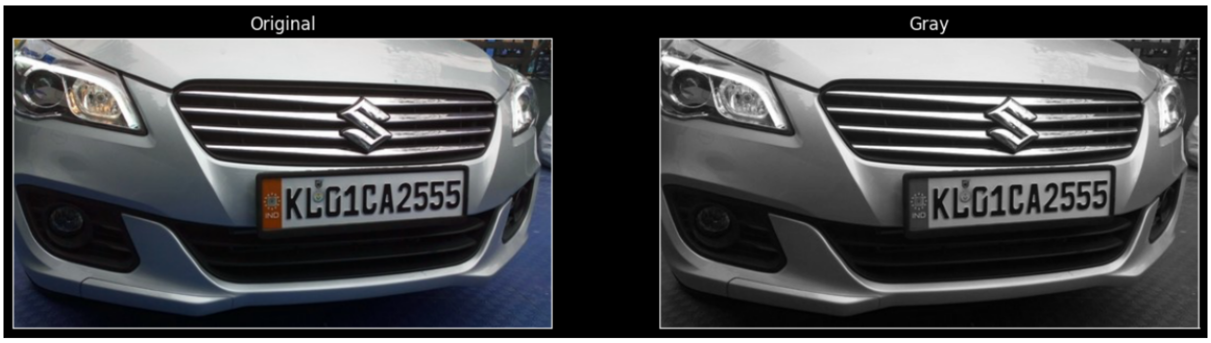
• The final prediction is converted to a text output to the user.

3.Results:

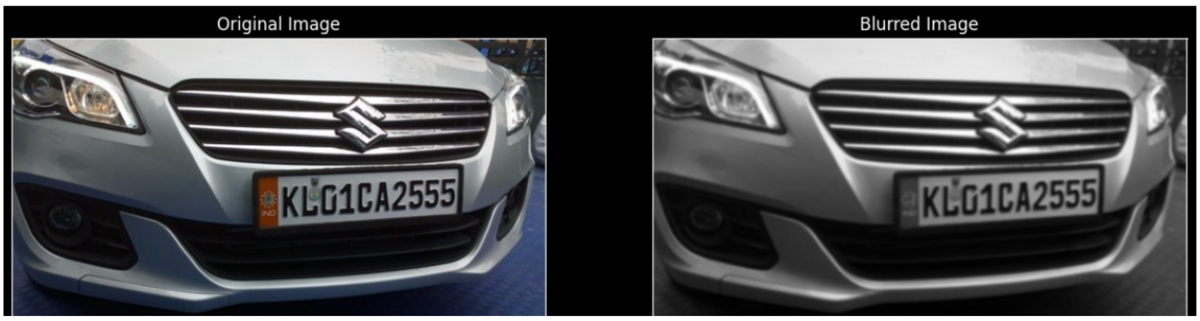
1.converts the original color image to grayscale

2. appling a bilateral filter to the grayscale image, resulting in a smoother version of the image. The smoothed image is then displayed alongside the original grayscale image with the titles "Gray" and "Smooth", respectively

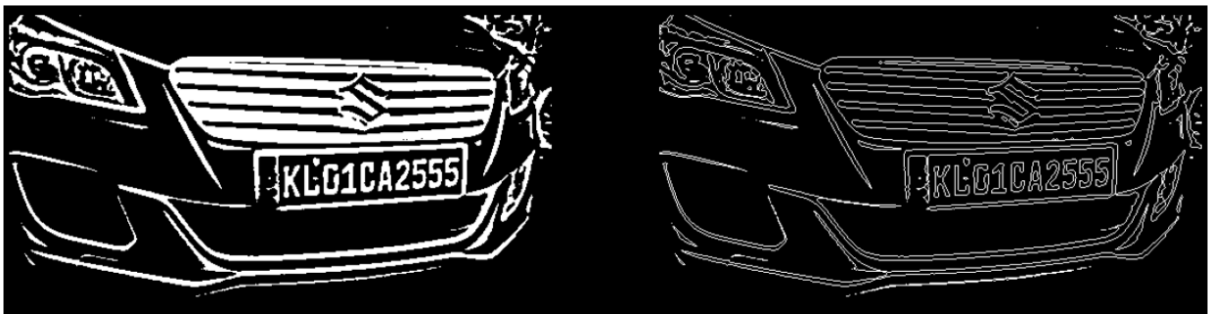




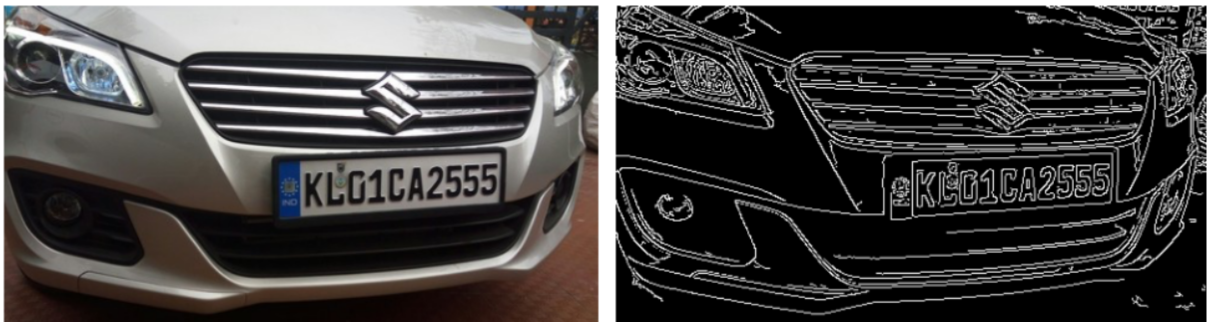
3.Our image processing technique using Gaussian blur enhances the sleek lines of this car while reducing unwanted noise.



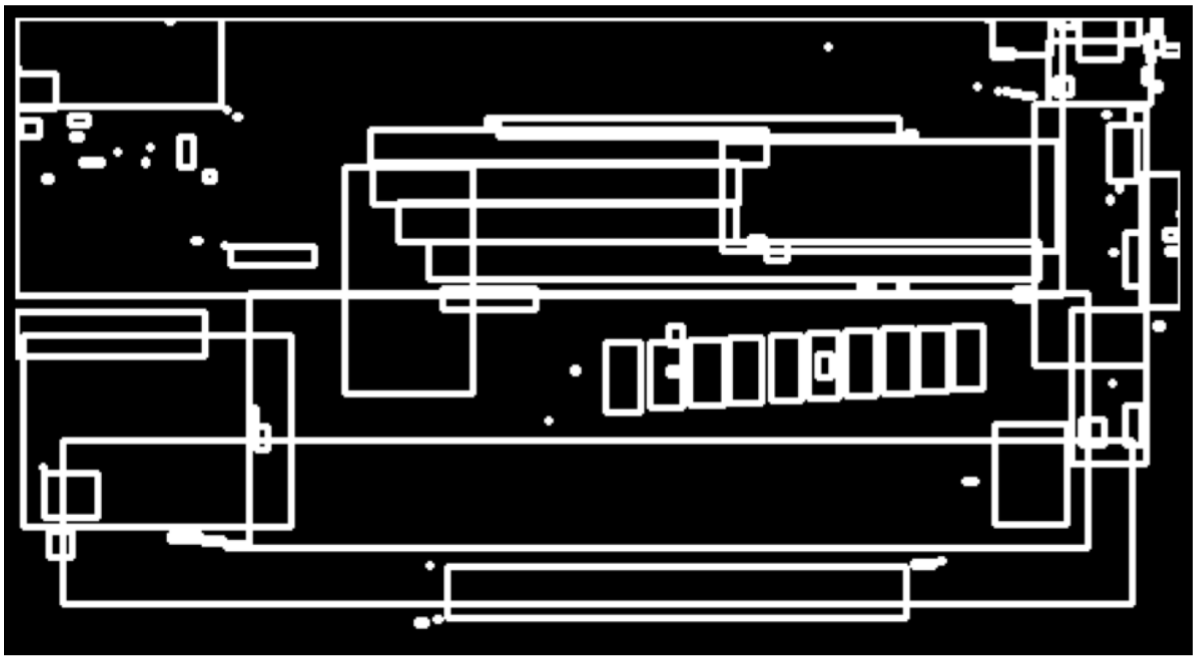
4.Our image processing technique uses thresholding and contour detection to identify and highlight the car's edges and shapes. By displaying the thresholded image and the image with contours side by side



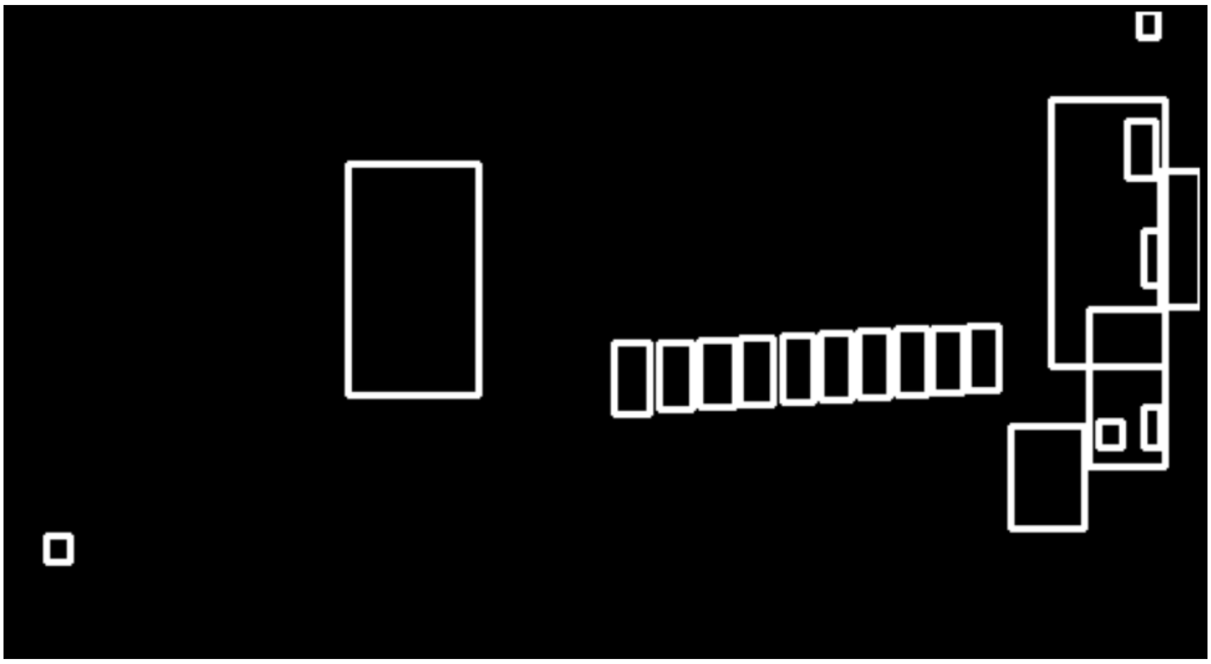
5.using Canny edge detection to highlight the car's edges and contours. By applying this technique, we can effectively extract the car's shape from the original image, resulting in a more striking and visually appealing representation.



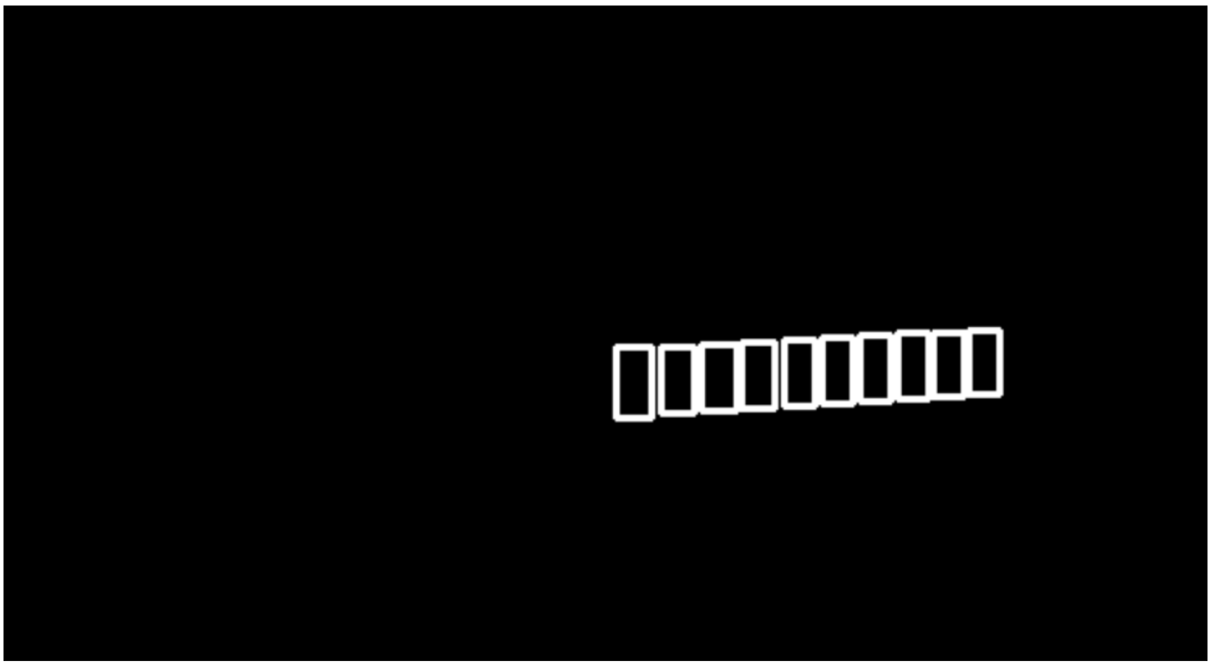
6. using contour detection to create precise bounding boxes around the car's different parts, highlighting its shape and structure



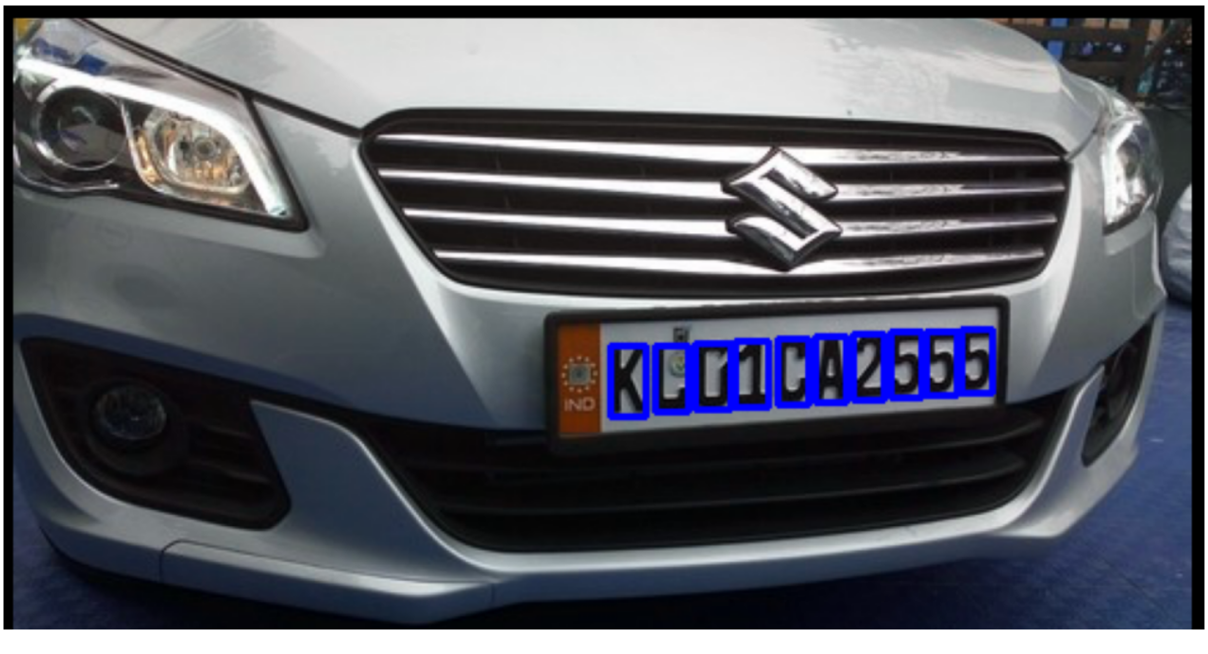
7. identifying and isolating the car's different parts using criteria based on size and aspect ratio



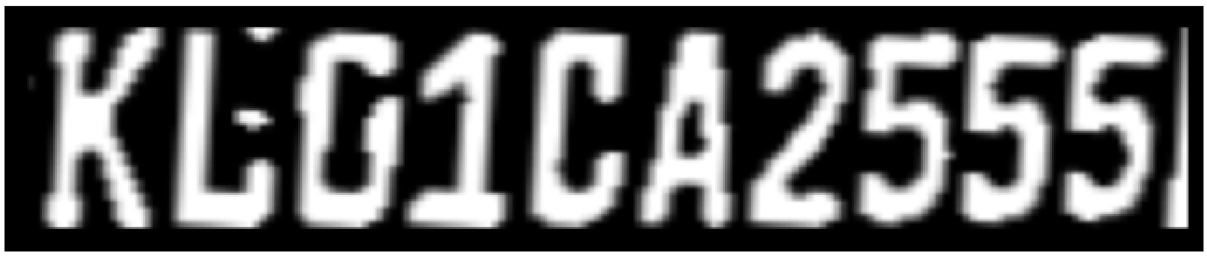
8.identifying and grouping contours that are likely to be part of the same character. By comparing the size, shape, and position of each contour.



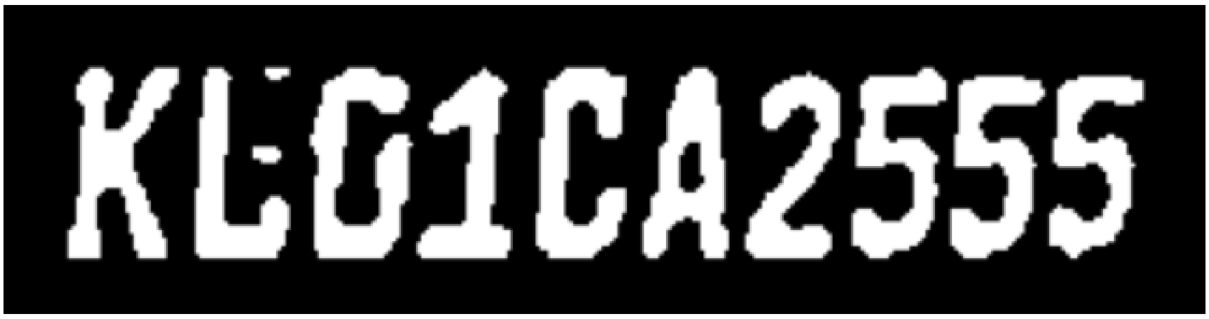
9.Our image processing technique groups together similar contours and draws rectangles around them to better highlight the car's features. The resulting image shows the rectangles around the matched contours overlaid on the original image, resulting in a more detailed and visually interesting representation of the car



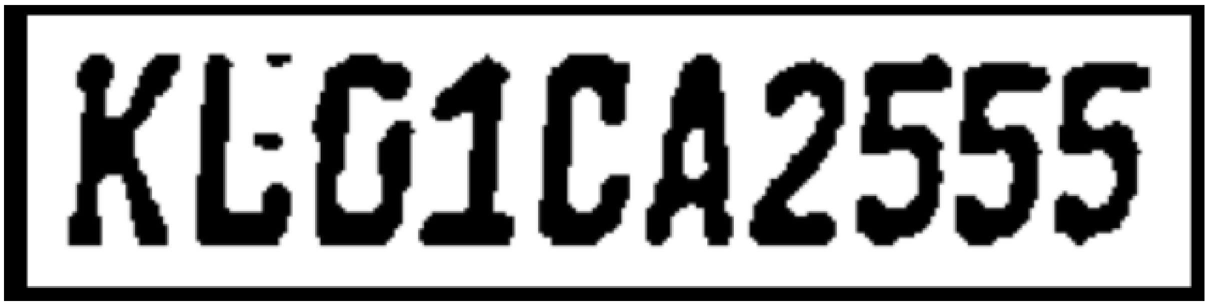
10.identifying the likely positions of license plates based on the size, shape, and position of the matched contours. By grouping together similar contours and calculating the center coordinates of the plate . The resulting image shows the rotated and cropped license plate, resulting in a more detailed and visually representation of the car's features.



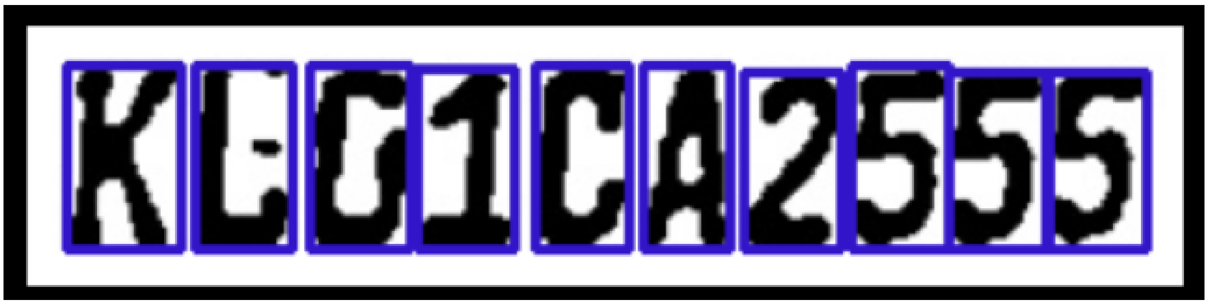
11.applies various filters to enhance the visibility of the characters on the license plates. By resizing, thresholding, and extracting the region of interest based on the minimum and maximum coordinates



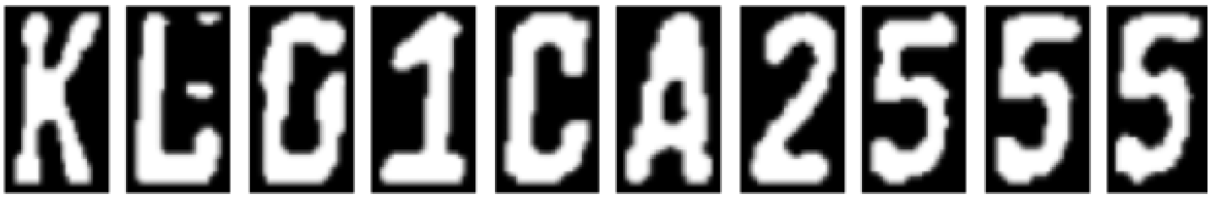
12.To further enhance the visibility of the license plate characters, we create a negative image by subtracting the thresholded plate image from 255.



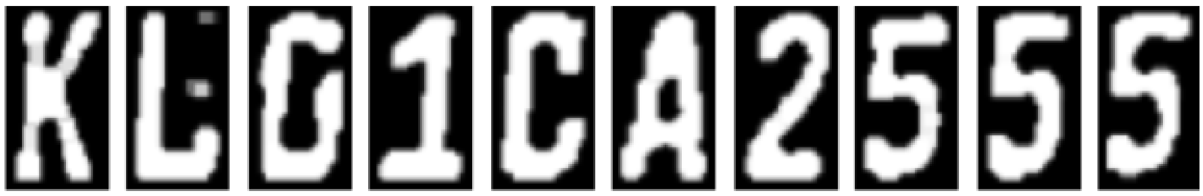
13. we segment the individual characters from the license plate image using a function called segment_characters(



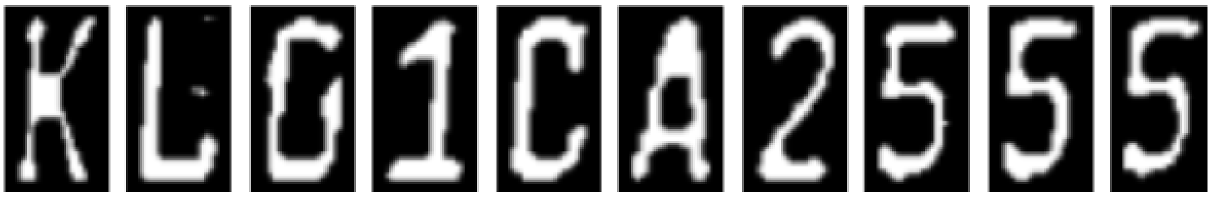
14. creating a subplot for each character in the char object and display the grayscale image of each character using the 'imshow' function



15.applying morphological operations to each character image in the char object to enhance the visibility of the characters. Specifically, we apply erosion followed by dilation to each character image, using a kernel of size 3x3 and one iteration each. The resulting dilated_img object likely contains the dilated character image. The resulting dilated images are displayed using the imshow function in separate subplots



16.We apply erosion to each character image in the char object to enhance the visibility of the characters. Specifically, we apply erosion to each character image using a kernel of size 3x3 and one iteration.



4.conclusion

To sum up, using image processing techniques in various fields can be an effective method. For instance, it can be useful in car parking facilities to identify license plates and record entry and exit times, or in security systems to detect license plates and monitor any criminal activity. Moreover, the margin of error associated with this approach is relatively low, making it a reliable option for security and computational purposes that rely on the outcomes of the image processing technique utilized.