Designing Proposed Method

Proposed methodology: OpenCV for image preprocessing and PyTesseract for OCR

1. Image Loading and Preprocessing: Loading the input image, resizing it for standardization, and converting it to grayscale. A bilateral filter is applied to reduce noise, and edge detection is performed using the Canny algorithm.

2. Contours Detection: Contours in the edged image are detected using the OpenCV function cv2.findContours. These contours represent potential shapes within the image.

3. Visualizing Contours on Original Image: The original image is copied, and the detected contours are drawn on it to aid visualization and understanding of the regions of interest.

4. Displaying the Original Image Before Drawing Contours: Another copy of the original image is displayed before drawing any contours to provide a visual reference.

5. Sorting and Selecting Top Contours: Contours are sorted based on their areas in descending order, and the top 30 contours are selected for further processing.

6. Identifying License Plate Contour: A loop iterates over the selected contours, approximating each contour's shape. If a contour has four corners, it is considered a candidate for a license plate. The script extracts the region of interest (ROI) around the license plate.

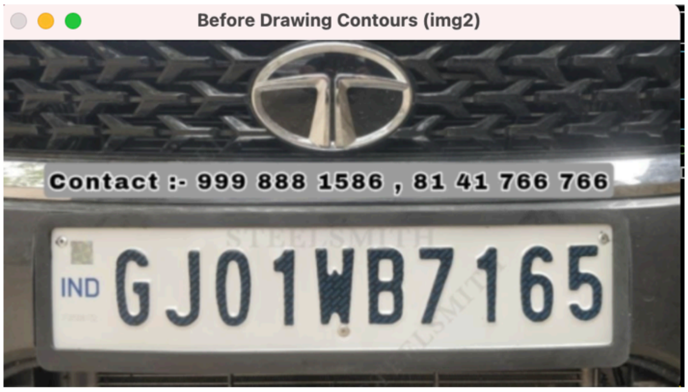
7. Visualizing Contours on Original Image After Drawing: The original image is again displayed, this time with the selected contours drawn on it, allowing visual confirmation of the identified license plate region.

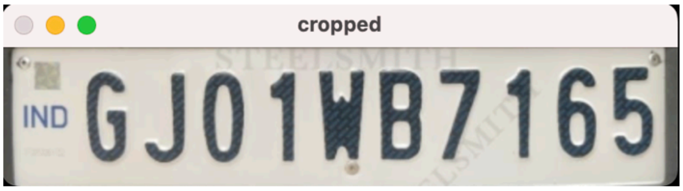
8. Displaying and OCR on the Cropped License Plate: The cropped license plate region is displayed, and OCR is performed using Tesseract to extract text, representing the license plate number.

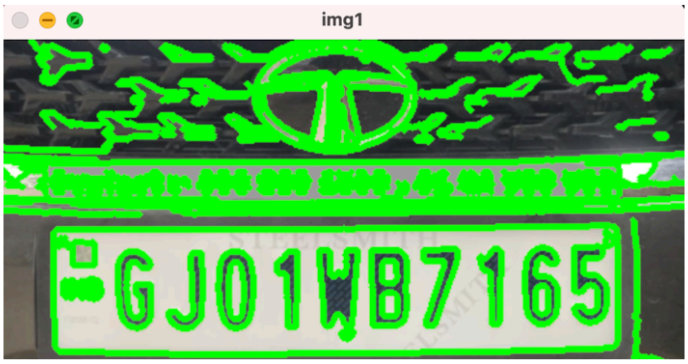
Code:

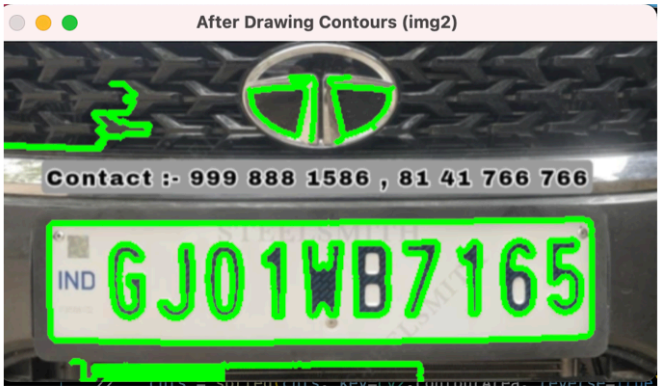
```
import cv2
import imutils
import numpy as np
import pytesseract
from PIL import Image

pytesseract.pytesseract.tesseract_cmd = '/usr/local/bin/tesseract'
img = cv2.imread('/Users/anshahhegde/Desktop/car63.png', cv2.IMREAD_COLOR)
img = imutils.resize(img, width=500)
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  # convert to grey scale
gray = cv2.bilateralFilter(gray, 11, 17, 17)  # Blur to reduce noise
edged = cv2.Canny(gray, 30, 200)  # Perform Edge detection
cnts, new = cv2.findContours(edged.copy(), cv2.RETR_LIST, 
cv2.CHAIN_APPROX_SIMPLE)
img1 = img.copy()
cv2.drawContours(img1, cnts, -1, (0, 255, 0), 3)
cv2.imshow("img1", img1)

# Display img2 before drawing contours
img2 = img.copy()
cv2.imshow("Before Drawing Contours (img2)", img2)

cnts = sorted(cnts, key=cv2.contourArea, reverse=True)[:30]
screenCnt = None  # will store the number plate contour

# Initialize idx before the loop
idx = 7

# loop over contours
for c in cnts:
    # approximate the contour
    peri = cv2.arcLength(c, True)
    approx = cv2.approxPolyDP(c, 0.018 * peri, True)
    if len(approx) == 4:  # chooses contours with 4 corners
        screenCnt = approx
        x, y, w, h = cv2.boundingRect(c)  # finds co-ordinates of the plate
        new_img = img[y:y + h, x:x + w]
        cv2.imwrite('./' + str(idx) + '.png', new_img)  # stores the new image
        idx += 1
        break

# Display img2 after drawing contours
cv2.drawContours(img2, cnts, -1, (0, 255, 0), 3)
cv2.imshow("After Drawing Contours (img2)", img2)

# ...
Cropped_loc = '/Users/anshahhegde/Desktop/ci63.png'  # Update with the correct 
path and filename of the cropped image
cv2.imshow("cropped", cv2.imread(Cropped_loc))
pytesseract.pytesseract.tesseract_cmd = r"/usr/local/bin/tesseract"  # exe file 
for using OCR

text = pytesseract.image_to_string(Cropped_loc, lang='eng')  # converts image 
characters to string
print("Number is:", text)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

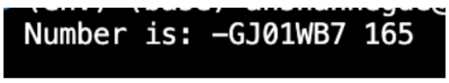
Results:


Before Drawing Contours (img2)
Contact :- 999 888 1586 , 81 41 766 766
IND GJ01WB7165
STEELSMITH

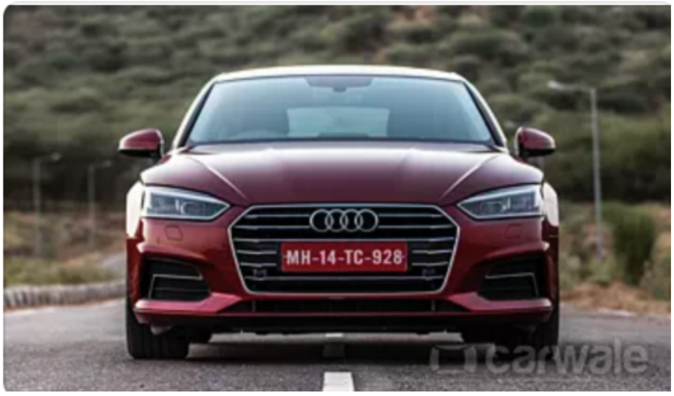


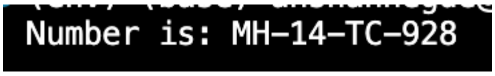



After Drawing Contours (img2)
Contact :- 999 888 1586 , 81 41 766 766
IND GJ01WB7165

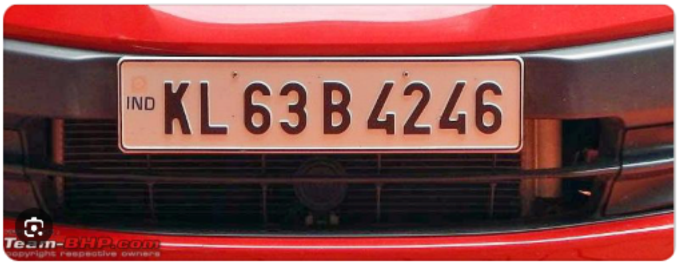


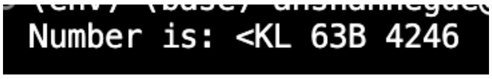
```
Number is: -GJ01WB7 165
```



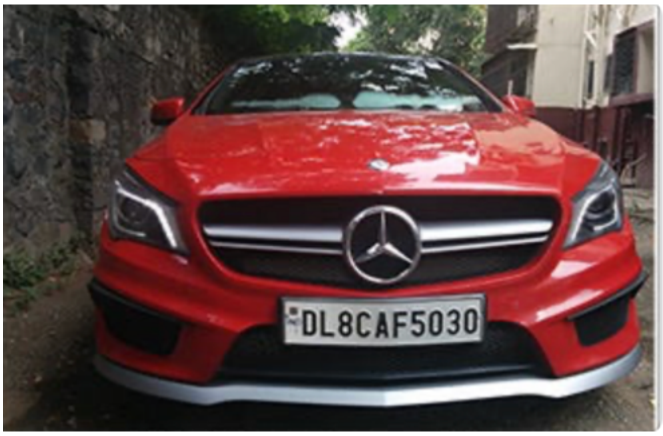


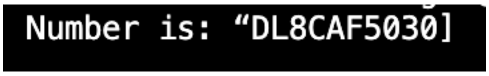
```
Number is: MH–14–TC–928
```



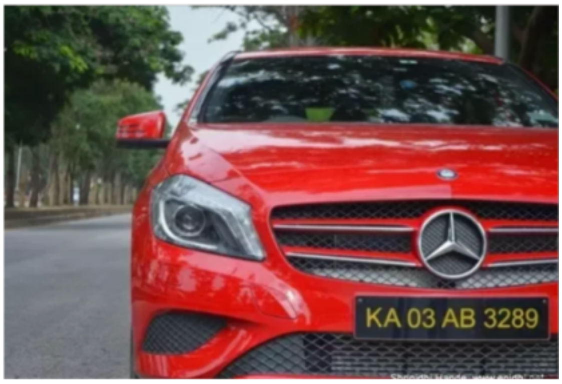


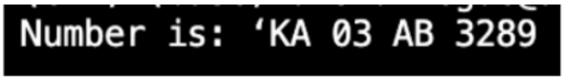
```
Number is: <KL 63B 4246
```



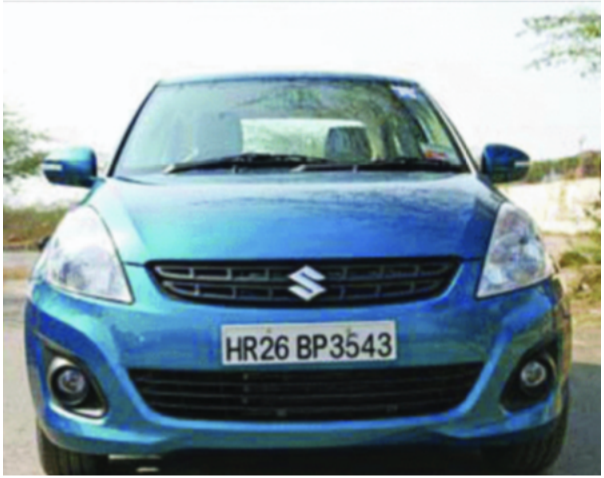


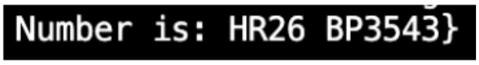
```
Number is: “DL8CAF5030]
```





```
Number is: ‘KA 03 AB 3289
```





```
Number is: HR26 BP3543}
```