Faculty of Engineering
Cairo University

CPMS446 - Image Processing and Computer Vision

Enhanced License Plate Recognition and Gate Access System

—

Team Member	ID
Abd-Elrahman Adel	1190140
Hossam Nabil	1190097
Nada Tarek	1180504
Nourhan Ahmed	1190423

Table of Contents

Project Overview

The project aims to detect license plates in car images and utilize an OCR (Optical Character Recognition) system to recognize Arabic letters and numbers on the license plates. The system is designed to extract license plates from images, perform pre-processing operations, and apply a trained OCR model for character recognition. It also detects the colours of the Egyptian car licenses.

Methodology

Utilize image processing techniques for license plate localization and apply OCR to recognize Arabic letters and numbers on the license plate. Ready tools and language used were; OpenCV for image processing, EasyOCR for Arabic character recognition accuracy check, Skimage, and Python as the programming language.

Algorithms Used

1 Image Preprocessing:

- **Grayscale Conversion (cv2.cvtColor)**: The conversion to grayscale simplifies the image, reducing the dimensionality while retaining essential information. Grayscale is important for character recognition, and it helps in reducing computational complexity.
- **Morphological Operations:**
    - **Morphological Closing (cv2.morphologyEx):** Closing is applied to connect gaps in the license plate characters. It helps in smoothing the contours and preparing the image for further processing.
    - **Morphological Opening (cv2.morphologyEx):** Opening is used to remove small objects and details from the background while preserving the overall structure. It aids in eliminating noise and fine textures, enhancing subsequent processing steps.
    - **Morphological Erosion (cv2.erode):** Erosion erodes away the boundaries of the foreground object, helping to separate connected characters and reduce

character thickness. It is useful in preparing the image for character segmentation.

    - **Morphological Dilation (cv2.dilate):** Dilation is applied to expand the boundaries of the foreground object, reconnecting broken characters and filling gaps. It is beneficial for improving the continuity of characters.
    - **Morphological Blackhat:** Blackhat operation enhances dark regions in an image, highlighting the characters and details on the license plate.

- **Thresholding (cv2.threshold):** Thresholding is used to convert the grayscale image into a binary image, where pixel values above a certain threshold become white, and below become black. This simplifies character segmentation.
- **Adaptive Thresholding (cv2.adaptiveThreshold):** Adaptive thresholding (Otsu's method) is used to handle variations in lighting conditions, enhancing the binarization process.

2 Character Extraction:

- **Contours (cv2.findContours):** Contours are used to identify and isolate individual characters on the license plate. Each contour represents a distinct region in the image. This was also used in colour extraction where we searched for max contour by area to extract the most dominant colour in the plate.
- **Sobel Operator (cv2.Sobel):** The Sobel operator is used to detect vertical edges in the license plate image, emphasizing the boundaries of characters.
- **Bounding Rectangle (cv2.boundingRect)**: Bounding rectangles are fitted around each contour to create regions of interest (ROIs) containing potential characters. These rectangles serve as a basis for character extraction.
- **Character Filtering**: Filtering contours based on area, aspect ratio, and location helps eliminate noise and non-character regions, focusing on potential characters on the license plate.
- **Connected Components (cv2.connectedComponents):** Connected components are identified to group pixels that belong to the same object, assisting in locating and segmenting license plate regions.
- **Character Cropping:** Extracting characters based on the bounding rectangles allows for the isolation of individual characters for further processing or recognition.

- **Saving Extracted Characters:** Saving the extracted characters as separate images facilitates manual verification, dataset creation, and potential use in training machine learning models.

3 Data Preparation for Training:

- **Copying Training Data (shutil.copy):** Creating a structured dataset involves copying character images into class-specific folders. This structured dataset is crucial for training a supervised machine learning model.

4 Machine Learning Model Training (K-Nearest Neighbors):

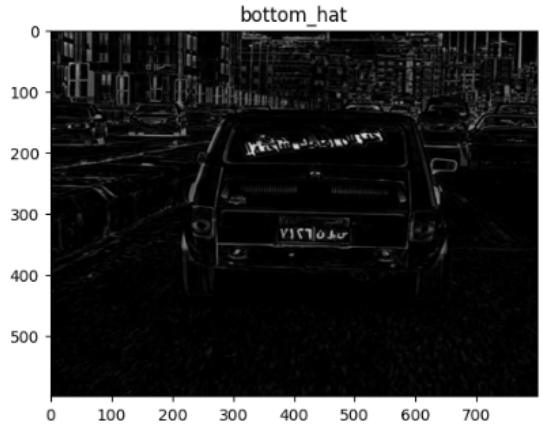
- **Data Loading:** Loading character images and labels prepares the data for training.
- **Data Normalization**: Normalizing pixel values ensures that the model is not sensitive to the scale of input features, improving convergence during training.
- **Train-Test Split (train_test_split):** Splitting the dataset into training and validation sets allows for model evaluation on unseen data.
- **K-Nearest Neighbors Classifier (neighbours.KNeighborsClassifier):** The choice of KNN is based on its simplicity and effectiveness in recognizing patterns in data. It is used as a baseline model for character recognition.
- **Model Evaluation (classification_report):** Evaluating the model on a validation set provides insights into its performance and helps in identifying potential issues.
- **Saving Trained Model (joblib.dump):** Saving the trained model allows for later use without the need to retrain, streamlining the deployment process.

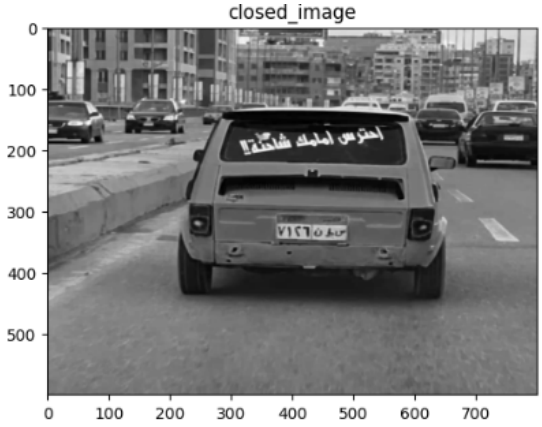
Experiment Results and Analysis

1) First, we resize the image for a consistent size and display the original image after resizing to (800, 600).

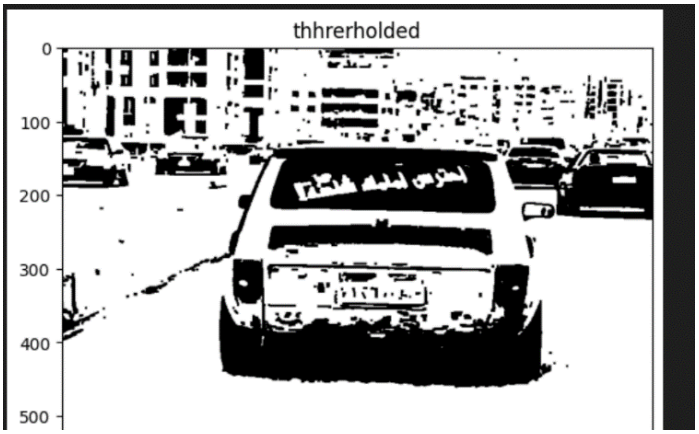
2) Then convert the image to grayscale (image_gray) and apply the Morphological Blackhat operation to enhance dark regions (bottom_hat). We chose to use a rectangular structuring element because the car plate that we are trying to enhance the image to extract is rectangular. The Blackhat operation is followed by Morphological Closing to smoothen contours (closed_image).

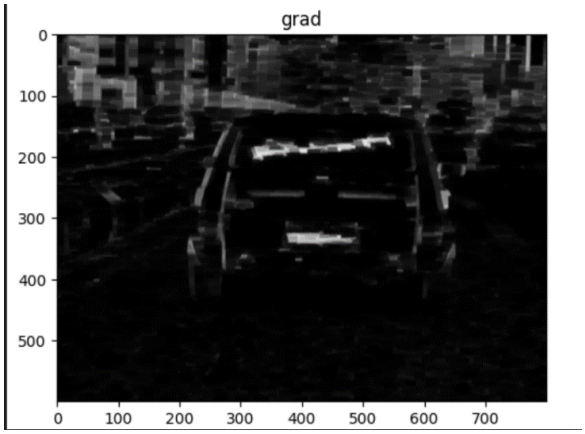
```
kernel_rect = cv2.getStructuringElement(cv2.MORPH_RECT, (13, 5))
bottom_hat = cv2.morphologyEx(image_gray, cv2.MORPH_BLACKHAT, kernel_rect)
kernel_closing = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3))
closed_image = cv2.morphologyEx(image_gray, cv2.MORPH_CLOSE, kernel_closing)
```



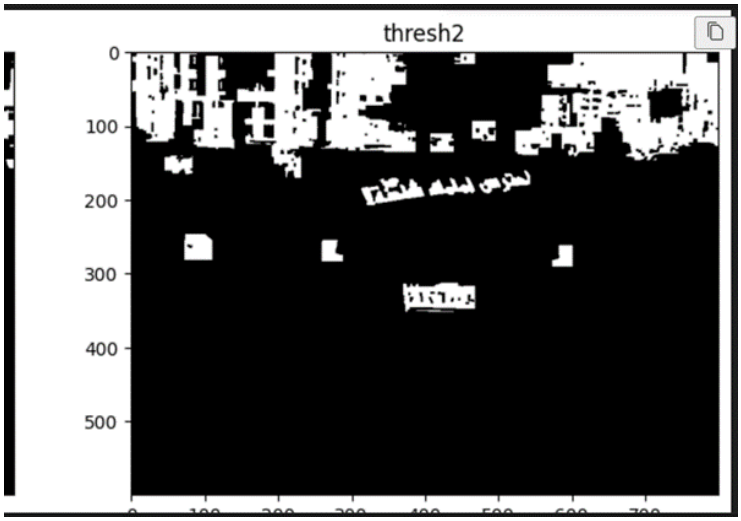


3) Then we apply thresholding using Otsu's method (thresholded) on the image followed by Sobel operator and Gradient Normalization to detect vertical edges (gradX). Also normalise gradients to the range 0-255.



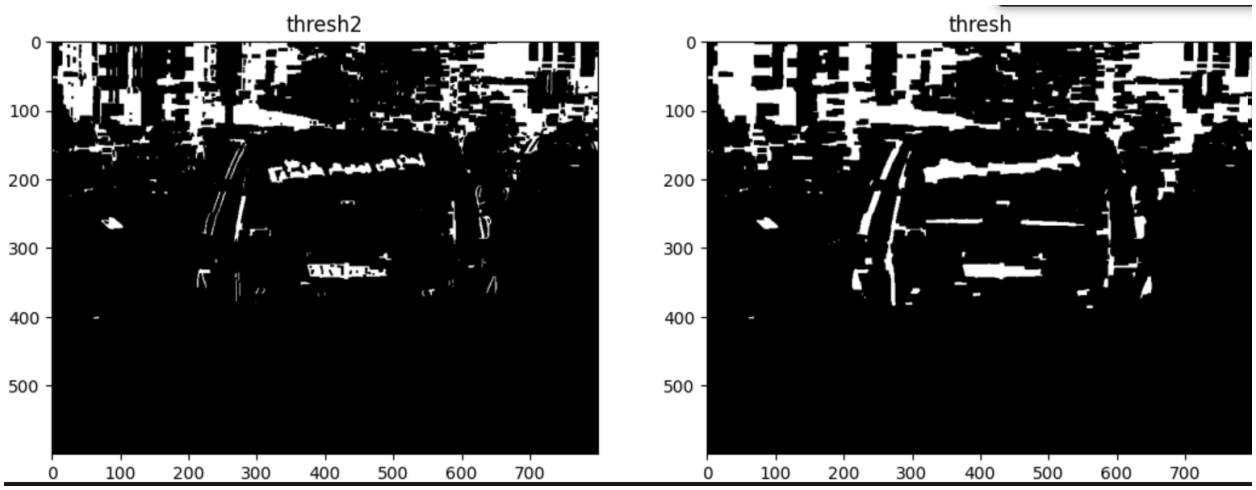


4) Apply Gaussian blur to gradXand Morphological Closing to further enhance contours (thresh).



5) Perform Bitwise AND operation between thresh and thresholded (thresh2). Apply a series of morphological operations (erode, dilate) to refine the result and find contours in the processed image.

Display contours (thresh2 and thresh).



6) Display the final image with the detected license plate outlined (image and plate).

7) Detect Plate Color using color_mask for each possible colour

Performance & Accuracy

1) Plate detection:

- Out of 118 images, 84 images had the license plate contoured, which gives an accuracy of 72 %

2) OCR

- The KNN model was trained on 80% of the available data (10,500 images of Arabic characters extracted from license plates), and then tested using 20% of the data, this yielded an accuracy of **0.95**.
- After using the model to predict the license plate characters that were extracted by the previous modules using image processing; the accuracy was **0.652**.

Work Division Between Team Members

Module Name	Team Member(s)
Image Pre-processing and Plate Region Extraction	Hossam Nabil
	Abd-Elrahman Adel
Character Extraction and Segmentation and Character Recognition (KNN model)	Nada Tarek
	Nourhan Ahmed

Additional Comments

The project successfully achieved its objectives, but there is room for enhancement in terms of real-time processing and handling diverse lighting and environmental conditions.

Level of Variety for Test Cases

The experimental results were tested on a diverse set of car images, considering variations in lighting conditions, image resolutions, and license plate sizes. This helped ensure the robustness of the system.

Choice of Comparison Metric

The comparison metric for evaluating the OCR model's performance was based on classification metrics such as accuracy during the training phase.

System Analysis: Strengths and Weaknesses

Strengths

- License Plate Localization: The system effectively localizes license plates in various images.
- Character Recognition: The OCR model demonstrates good accuracy in recognizing Arabic letters and numbers.

Weaknesses

- Robustness: The system may struggle under challenging lighting and environmental conditions.
- Real-time Processing: The current implementation may not meet real-time processing requirements.

Conclusion

The project successfully developed a license plate detection and OCR system for Arabic characters. While achieving satisfactory results, there is potential for further improvement, especially in terms of real-time processing and robustness.

References

https://docs.opencv.org/

https://github.com/JaidedAI/EasyOCR

https://scikit-learn.org/stable/documentation.html

https://www.researchgate.net/publication/363892739_An_Application_of_Image_Processing_License_Plate_Recognition