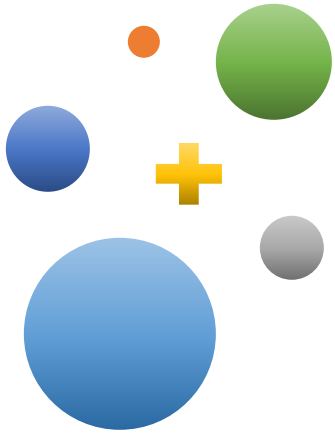
ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HOC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

Báo cáo đồ án

Automatic license plate recognition (ALPR)



GVHD: Võ Hoài Việt

Mục lục

A Thành viên nhóm:

Tên nhóm: Fusion

STT	MSSV	Họ tên	Email
1	1612174	Phùng Tiến Hào	tienhaophung@gmail.com
2	1612269	Võ Quốc Huy	voquochuy304@gmail.com
3	1612272	Trần Nhật Huy	nhathuy13598@gmail.com

B Mức độ hoàn thành:

STT	Nội dung	Mức độ hoàn thành (%)
1	Phát hiện biển số (Plate detection)	100
2	Kiểm tra biển số (Validation)	100
3	Nhận dạng kí tự trong biển số (Character recognition)	100
Tổng cộng:		**100**

C Báo cáo:

I. Tổng quan cách cài đặt:

Các bước thực hiện hiện:

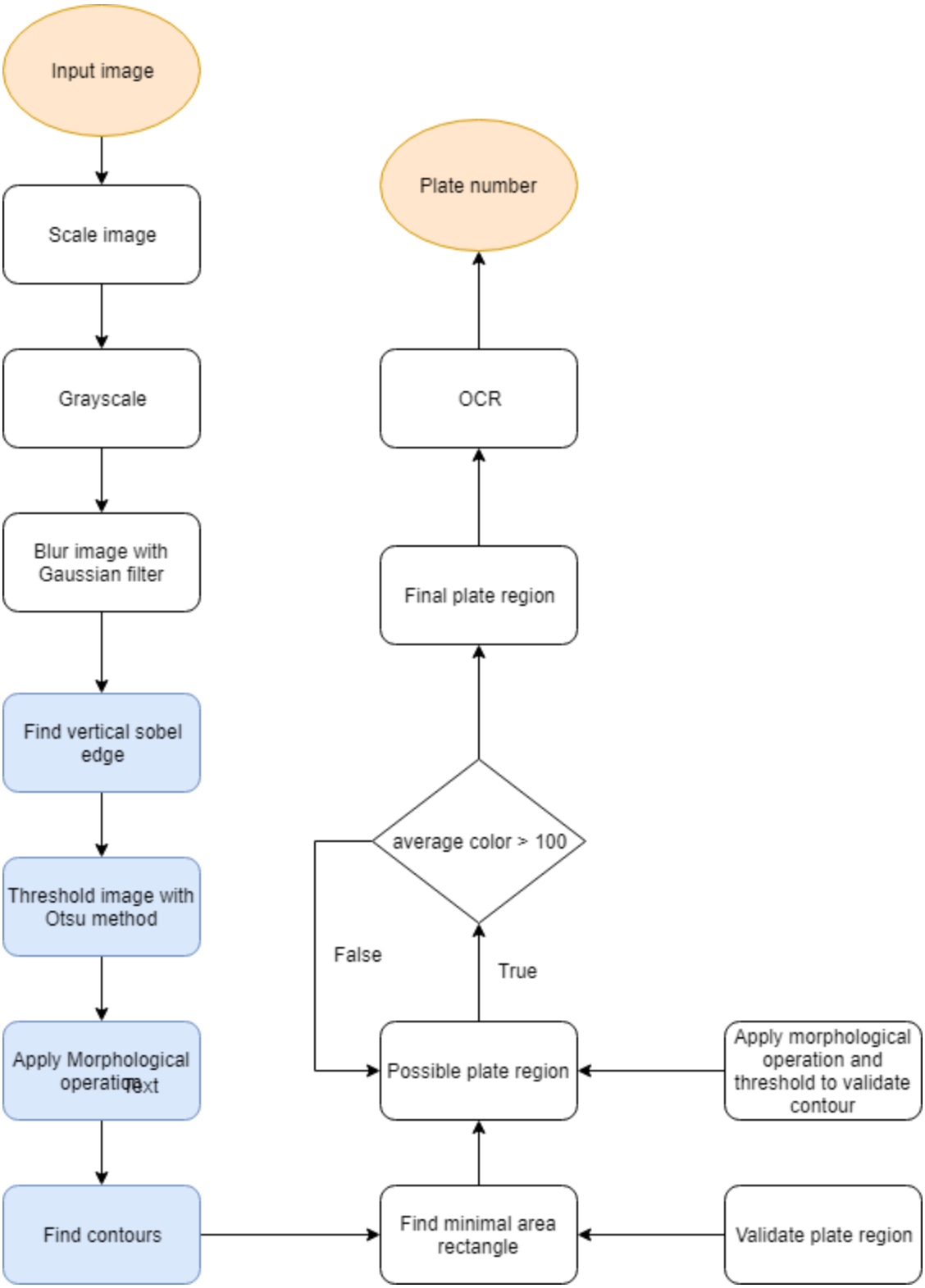


Figure 1 Lưu đồ hoạt động của hệ thống

Chi tiết hơn:

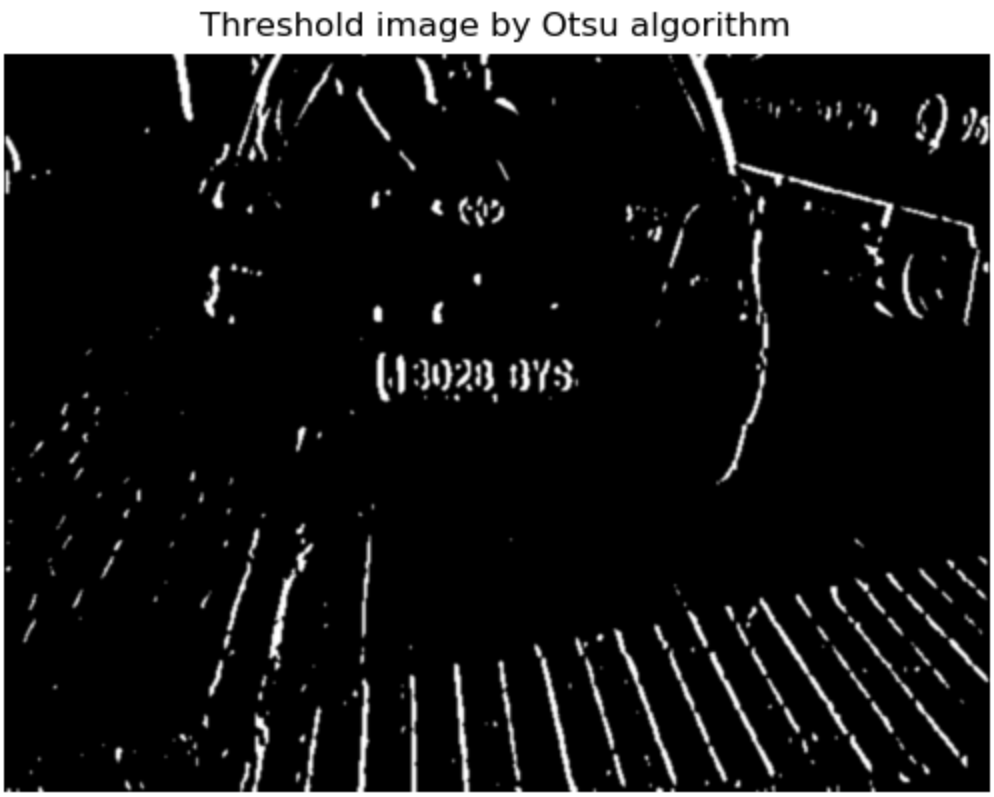
1) Scale ảnh về một kích thước chung: có width = 600 và height = (600 * original_height)/original_width

2) Gray-scale image: để thực hiện tìm biên cạnh và phục vụ cho các bước sau

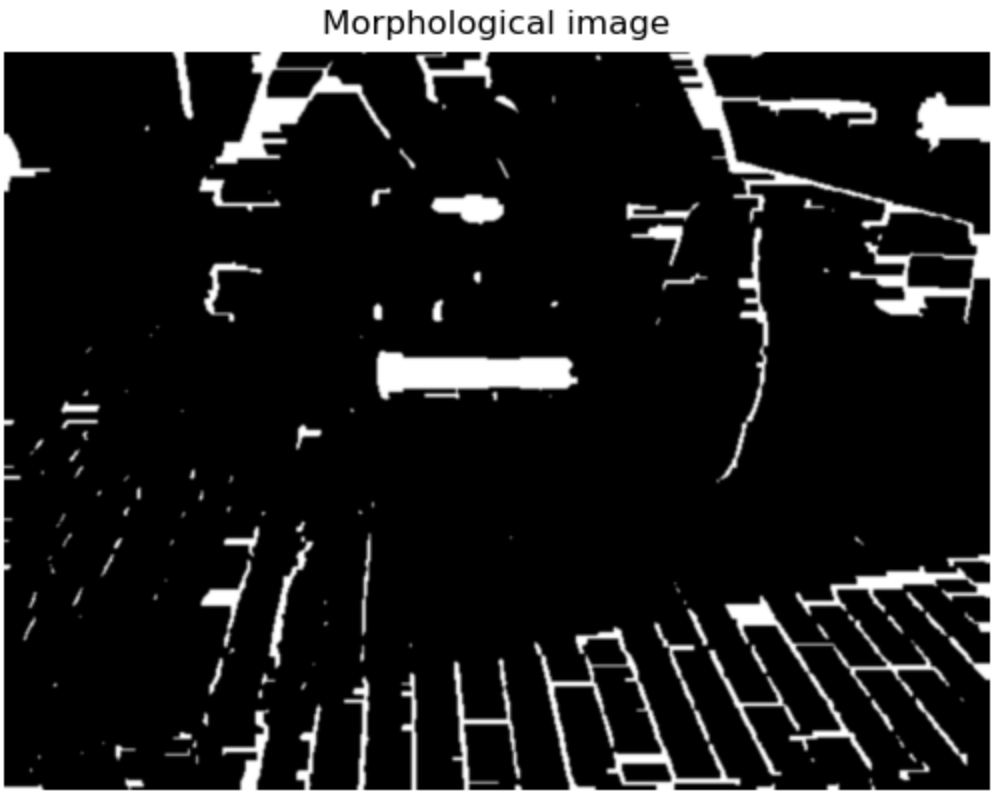
3) Làm mờ ảnh với Gaussian filter để giảm bớt nhiễu

4) Tìm biên cạnh dọc bằng Sobel filter

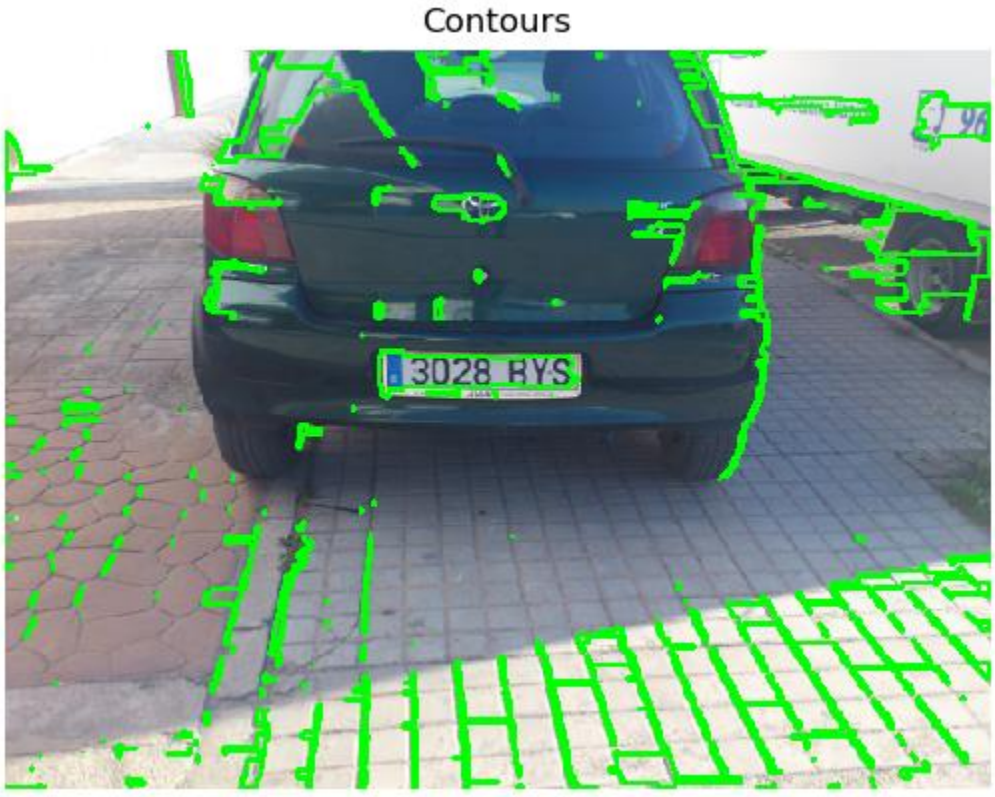
5) Threshold ảnh bằng phương pháp Otsu để nó tự tìm một ngưỡng tối ưu



6) Áp dụng morphological operation, cụ thể là phép “Clossing”. Tức là trước tiên sẽ làm giản nỡ (dilation) ảnh sau đó làm xói mòn (erosion) ảnh để giảm nhiễu và xóa các lỗ chấm đen trên đối tượng. Với kích thước kernel = 19x3 (đây là kích thước thực nghiệm) chỉ áp dụng để nhận diện biển số xe ô tô với tối đa 8 kí tự.



7) Tìm contours để phân vùng ảnh



8) Tìm các bounding box hình chữ nhật (có tính góc nghiêng không quá 30 độ)

9) Kiểm tra và xác thực các bounding box:
Các công viên liên quan:
- Kiểm ra màu trắng có chiếm đáng kể trong vùng đó không
- Kiểm tra tỉ lệ khung hình cho phần vùng biển số:
  - Kích thước biển ô tô của ngước ngoài (cụ thể châu Tây Ban Nha hoặc Châu Âu): chiều rộng 52 cm, chiều cao 11 cm.
  - Do đó, tỉ lệ khung hình: 52/11 = 4.7272 với độ lỗi chấp nhận là 40%.
  - Dao động của tỉ lệ khung hình là min = 4.7272*(1 – 0.4), max = 4.7272*(1 + 0.4)
- Kiểm tra diện tích vùng trong khoảng từ: min = 15*aspect_ratio*15, max = 125*aspect_ratio*125
- Kiêm tra góc quay: Nếu bounding box xoay quá 30 độ thì bị loại.

10) Lợi dụng phần nền biển số là màu trắng, tính cường độ màu trung bình của phần ảnh biển số đó xem có gần trắng hay không (> 100)

11) Nhận dạng kí tự trong biển số bằng thư viện mở Tesseract – được phát triển bởi Google

II. Yêu cầu của hệ thống:

Để đảm bảo hệ thống hoạt động tốt và chính xác thì chất lượng ảnh đầu vào đóng vai trò rất quan trọng. Cụ thể:

- Ảnh phải được chụp trực diện và cách xa khoảng 1.5-2 (m).
- Điều kiện sáng đầy đủ
- Ảnh bị nhiễu ít
- Góc nghiêng vừa phải

Ngôn ngữ sử dụng: Python

Các thư viện cần thiết để chạy chương trình:

- Thư viện OpenCV
- Thư viện PIL
- Thư viện Matplotlib
- Thư viện Numpy
- Thư viện Pytesseract

III. Kết quả thực nghiệm:

Mẫu dữ liệu test gồm 66 ảnh trong nhiều điều kiện khác nhau như: độ sáng, nhiễu, ảnh chụp góc nghiêng,…

Công việc	Số lượng mẫu bị thất bại	Độ chính xác (%)
Phát hiện biển số	9	86.36
Nhận dạng kí tư	45	30

Một số ảnh phát hiện và nhận dạng kí tự đúng:

Plate number recognition

Correct plates

Plate number recognition

Correct plates

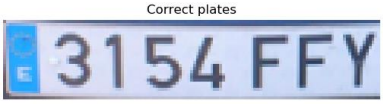


Plate number recognition

Correct plates

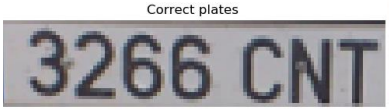
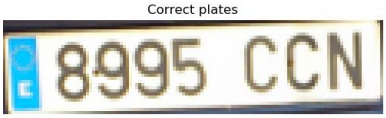


Plate number recognition

Correct plates



Một số ảnh phát hiện sai hoặc nhận dạng sai (hoặc không ra kết quả: None)

Plate number recognition

Correct plates

Plate number recognition

Correct plates

Nhận xét:

- Kết quả phát hiện biển số vẫn khá thi nhưng chưa thực sự chính xác vì có trường hợp đã phát hiện ra vùng ảnh là biển số nhưng khi qua bước kiểm thực lần nữa thì đã bị loại. Điều này nhóm sẽ cải tiến bằng mô hình SVM để phân lớp và xác định biển số.
- Đa số ảnh bị nhận dạng kí tự sai hoặc không thể nhận dạng kí tự được.
- Bên cạnh đó, nhóm sẽ cố gắng thay thế phương pháp truyền thống bằng việc dùng mô hình mạng deep learning YOLO để giúp phát hiện biển số tốt hơn và chạy với thời gian thực.
- Hiện tại, công việc nhận dạng kí tự được thực hiện bởi thư viên Tesseract của google. Nhóm không hiểu ví sao kết quả nhận dạng lại quá thấp. Điều này trong đợt cải tiến sắp tới nhóm sẽ cố gắng cải thiện.

D Hướng dẫn sử dụng:

Chương trình được chạy bằng command line.

Cú pháp:

python <Ten chuong trinh.py> -i <Input image> -o <Option: 0 hoặc 1>

Ý nghĩa:

- <Ten chuong trinh.py>: file chạy chương trình

- <Input image>: đường dẫn đến ảnh input
- <Option>: Lựa chọn hiển thị tất cả các bước làm hay không. Nếu có thì nhập 1, không thì nhập 0. Mặc định chương trình sẽ để là 0 khi bạn không nhập giá trị này.

Lưu ý: Chương trình dùng argparse để giúp command line trở nên trực quan và dễ sử dụng hơn khi truyền tham sao. Bạn có thể nhập lệnh help để trở giúp:

python plate_detection.py -h

```
(opencv-env) E:\K16\Junior\TGMT\ALPR-project>python plate_detection.py -h
usage: plate_detection.py [-h] -i INPUT [-o OPTION]

optional arguments:
  -h, --help            show this help message and exit
  -i INPUT, --input INPUT
                        Path to input image
  -o OPTION, --option OPTION
                        Show step by step
```

Ví dụ: Để chạy chương trình ALPR với tập ảnh ".\test_images\IMG_0378.jpg":

python 1612174_1612269_1612274_Lab03.py -i ".\test_images\IMG_0378.jpg" – o 0

E Tham khảo:

[1] Baggio, D. L. (2012). 5. Number Plate Recognition Using SVM and Neural Networks. In Mastering OpenCV with Practical Computer Vision Projects (6th ed., pp. 161-188). Birmingham, UK: Packt Publishing

[2] https://github.com/kagan94/Automatic-Plate-Number-Recognition-APNR.git

[3] https://github.com/MicrocontrollersAndMore/OpenCV_3_License_Plate_Recognition_Python.git

[4] https://docs.opencv.org/3.4.3/d7/d4d/tutorial_py_thresholding.html

[5] https://docs.opencv.org/3.1.0/d4/d73/tutorial_py_contours_begin.html

[6] https://docs.opencv.org/trunk/d9/d61/tutorial_py_morphological_ops.html

[7] https://docs.opencv.org/3.1.0/dd/d49/tutorial_py_contour_features.html

[8] https://www.pyimagesearch.com/2017/07/10/using-tesseract-ocr-python/

[9] https://cvisiondemy.com/license-plate-detection-with-opencv-and-python/

[10] Dataset: http://www.medialab.ntua.gr/research/LPRdatabase.html?fbclid=IwAR0d_5jeUecAGPX2Dw23p5GFpArKSRAsLSZSD-jOU_RGhGbMOU2JydKsRq8