**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề tài số 10: Xây dựng hệ thống nhận diện biển số xe tự động với OpenCV**

**Giảng viên hướng dẫn: Lương Thị Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210893 | Phạm Mạnh Cương | DCCNTT12.10.3 |
| **2** | 20210763 | Nguyễn Huy Hoàng | DCCNTT12.10.3 |
| **3** | 20210909 | Nguyễn Văn Tuấn | DCCNTT12.10.3 |

**Bắc Ninh, năm 2024**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÀI TẬP LỚN**

**HỌC PHẦN: XỬ LÝ ẢNH VÀ THỊ GIÁC MÁY TÍNH**

**Đề tài số 10: Xây dựng hệ thống nhận diện biển số xe tự động với OpenCV**

**Giảng viên hướng dẫn: Lương Thị Hồng Lan**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Mã sinh viên** | **Sinh viên thực hiện** | **Lớp hành chính** |
| **1** | 20210893 | Phạm Mạnh Cương | DCCNTT12.10.3 |
| **2** | 20210763 | Nguyễn Huy Hoàng | DCCNTT12.10.3 |
| **3** | 20210909 | Nguyễn Văn Tuấn | DCCNTT12.10.3 |

|  |  |
| --- | --- |
|  |  |

**Bắc Ninh, năm 2024**

|  |  |
| --- | --- |
| TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐÔNG Á  **KHOA CÔNG NGHỆ THÔNG TIN** | **KỲ THI KẾT THÚC HỌC PHẦN**  **HỌC KỲ 1, NĂM HỌC 2024** – **2025** |

|  |  |
| --- | --- |
| **PHIẾU CHẤM THI BÀI TẬP LỚN KẾT THÚC HỌC PHẦN**  **Mã đề thi: 10**  **Tên học phần: Xử lý ảnh và thị giác máy tính**  **Lớp Tín chỉ: DCCNTT12.10.3** | |
| **Cán bộ chấm thi 1**  *(Ký và ghi rõ họ tên)* | **Cán bộ chấm thi 2**  *(Ký và ghi rõ họ tên)* |

| **TT** | **TIÊU CHÍ** | **THANG ĐIỂM** | **Phạm Mạnh Cương** | **Nguyễn Huy Hoàng** | **Nguyễn Văn Tuấn** |
| --- | --- | --- | --- | --- | --- |
| 20210893 | 20210763 | 20210909 |
| **1** | **Nội dung báo cáo trên Word đầy đủ** | **3.5** |  |  |  |
| 1.1 | Có bố cục rõ ràng (mục lục, phần mở đầu, nội dung chính, kết luận). | 0,5 |  |  |  |
| 1.2 | Nội dung phân tích rõ ràng, logic. | 0,5 |  |  |  |
| 1.3 | Có dẫn chứng, số liệu minh họa đầy đủ. | 0,5 |  |  |  |
| 1.4 | Ngôn ngữ và trình bày chuẩn, không lỗi chính tả. | 0,5 |  |  |  |
| 1.5 | Có trích dẫn tài liệu tham khảo đúng quy cách. | 0,5 |  |  |  |
| 1.6 | Được trình bày chuyên nghiệp (canh lề, font chữ, khoảng cách dòng hợp lý). | 0,5 |  |  |  |
| 1.7 | Tài liệu đầy đủ, bám sát yêu cầu của đề bài. | 0,5 |  |  |  |
| **2** | **Nội dung thuyết trình đầy đủ** | **1.0** |  |  |  |
| 2.1 | Trình bày tự tin, phát âm rõ ràng, mạch lạc. | 0,5 |  |  |  |
| 2.2 | Nội dung thuyết trình đúng trọng tâm, không lan man. | 0,5 |  |  |  |
| **3** | **Slides báo cáo đầy đủ nội dung + Hỏi đáp** | **3.0** |  |  |  |
| 3.1 | Slides có bố cục rõ ràng (mở đầu, nội dung, kết luận). | 0,5 |  |  |  |
| 3.2 | Thiết kế slides đẹp, chuyên nghiệp (màu sắc, hình ảnh minh họa). | 0,5 |  |  |  |
| 3.3 | Nội dung trên slides ngắn gọn, dễ hiểu, súc tích. | 0,5 |  |  |  |
| 3.4 | Nội dung slides phù hợp với nội dung báo cáo. | 0,5 |  |  |  |
| 3.5 | Trả lời câu hỏi đầy đủ, chính xác. | 0,5 |  |  |  |
| 3.6 | Trả lời câu hỏi tự tin, thuyết phục. | 0,5 |  |  |  |
| **4** | **Code đầy đủ** | **2.5** |  |  |  |
| 1.1 | Code được trình bày rõ ràng, có chú thích đầy đủ. | 0,5 |  |  |  |
| 1.2 | Code chạy đúng, không lỗi. | 0,5 |  |  |  |
| 1.3 | Code tối ưu, không dư thừa. | 0,5 |  |  |  |
| 1.4 | Đáp ứng đầy đủ các yêu cầu chức năng theo đề bài. | 0,5 |  |  |  |
| 1.5 | Có tính sáng tạo hoặc cải thiện so với yêu cầu. | 0,5 |  |  |  |
| **TỔNG ĐIỂM BẰNG SỐ:** | | **10** |  |  |  |
| **TỔNG ĐIỂM BẰNG CHỮ:** | | *Mười tròn* |  |  |  |

# **LỜI CẢM ƠN**

Chúng em xin gửi lời cảm ơn chân thành đến cô Lương Thị Hồng Lan, người đã tận tình chỉ bảo, định hướng và hỗ trợ chúng em trong suốt quá trình thực hiện đề tài. Những ý kiến đóng góp quý báu của thầy/cô không chỉ giúp chúng em hiểu rõ hơn về bài toán mà còn là nguồn động lực lớn để chúng em hoàn thành tốt báo cáo này.

Chúng em cũng xin cảm ơn Nhà trường và Khoa Công nghệ Thông tin đã tạo điều kiện thuận lợi, cung cấp các nguồn tài liệu và môi trường học tập lý tưởng, giúp chúng em tiếp cận với các công nghệ hiện đại phục vụ nghiên cứu. Bên cạnh đó, sự động viên và hỗ trợ từ gia đình, bạn bè luôn là điểm tựa tinh thần vững chắc để chúng em vượt qua mọi khó khăn trong quá trình thực hiện đề tài.

Cuối cùng, chúng em chân thành cảm ơn cộng đồng phát triển mã nguồn mở, đặc biệt là các tác giả của OpenCV và EasyOCR, những công cụ quan trọng đã giúp chúng em triển khai hệ thống một cách hiệu quả. Chúng em mong rằng với sự nỗ lực của mình, đề tài này sẽ mang lại giá trị thiết thực và nhận được những ý kiến đóng góp quý báu từ thầy cô và các bạn.

Chúng em xin chân thành cảm ơn!

# **LỜI NÓI ĐẦU**

Trong thời đại công nghệ 4.0, việc ứng dụng trí tuệ nhân tạo và xử lý ảnh vào đời sống ngày càng trở nên phổ biến và đóng vai trò quan trọng trong nhiều lĩnh vực như giao thông, y tế, và quản lý thông tin. Một trong những ứng dụng nổi bật là hệ thống nhận diện biển số xe tự động, giúp nâng cao hiệu quả và tính chính xác trong quản lý phương tiện giao thông.

Đề tài *"Xây dựng hệ thống nhận diện biển số xe tự động với OpenCV"* không chỉ mang tính thực tiễn cao mà còn giúp chúng em hiểu rõ hơn về các kỹ thuật xử lý ảnh và nhận diện ký tự. Báo cáo này được thực hiện nhằm trình bày toàn bộ quá trình phân tích, thiết kế, và triển khai hệ thống, cũng như đánh giá kết quả thực nghiệm trên bộ dữ liệu thực tế.

Chúng em hy vọng rằng báo cáo này sẽ mang lại một cái nhìn rõ ràng và chi tiết về quy trình xây dựng một hệ thống nhận diện biển số xe, đồng thời là một tài liệu tham khảo hữu ích cho những ai quan tâm đến lĩnh vực xử lý ảnh và thị giác máy tính.

# **MỤC LỤC**

[LỜI CẢM ƠN 5](#_Toc184605649)

[LỜI NÓI ĐẦU 6](#_Toc184605650)

[MỤC LỤC 7](#_Toc184605651)

[DANH MỤC HÌNH ẢNH 8](#_Toc184605652)

[CHƯƠNG I: CƠ SỞ LÝ THUYẾT 9](#_Toc184605653)

[1.1. Khái niệm biểu số xe 9](#_Toc184605654)

[1.2. Xử lý ảnh và OpenCV 10](#_Toc184605655)

[1.3. Phát hiện vị trí tách biển số xe 11](#_Toc184605656)

[1.3.1. Chuyển ảnh xám 11](#_Toc184605657)

[1.3.2. Giảm nhiễu bang bộ lọc Gaus 11](#_Toc184605658)

[1.3.3. Phát hiện cạnh Canny (Canny Edge Detetion) 12](#_Toc184605659)

[1.3.4. Lọc biển số với contour 13](#_Toc184605660)

[1.4. Nhận diện kí tự 14](#_Toc184605661)

[1.4.1. Tổng quan về lý thuyết 14](#_Toc184605662)

[CHƯƠNG II: XÂY DỰNG HỆ THỐNG 15](#_Toc184605663)

[2.1. Bài toán nói chung 15](#_Toc184605664)

[2.2. Các kỹ thuật sử dụng trong bài toán nhận dạng biển số xe 17](#_Toc184605665)

[2.2.1. Xử lý ảnh (Preprocessing) 17](#_Toc184605666)

[2.2.2. Phát hiện vùng biển số xe (Region Detection) 20](#_Toc184605667)

[2.2.3. Nhận dạng ký tự (Optical Character Recognition - OCR) 21](#_Toc184605668)

[2.2.4. Tra cứu tỉnh/thành phố từ mã biển số 21](#_Toc184605669)

[2.3. Ngôn ngữ lập trình và các thư viện sử dụng 22](#_Toc184605670)

[2.3.1. Python 22](#_Toc184605671)

[2.3.2. Các thư viện 22](#_Toc184605672)

[2.4. Xây dựng hệ thống nhận diện biển số xe 23](#_Toc184605673)

[2.4.1. Tiền xử lý ảnh 23](#_Toc184605674)

[2.4.2. Phát hiện vùng biển số 24](#_Toc184605675)

[2.4.3. Nhận diện ký tự 24](#_Toc184605676)

[2.4.4. Giao diện người dùng 25](#_Toc184605677)

[CHƯƠNG III: THỰC NGHIỆM 26](#_Toc184605678)

[3.1. Dữ liệu sử dụng 26](#_Toc184605679)

[3.2. Kết quả thực nghiệm 26](#_Toc184605680)

[KẾT LUẬN 30](#_Toc184605681)

[TÀI LIỆU THAM KHẢO 31](#_Toc184605682)

# **DANH MỤC HÌNH ẢNH**

[Hình 1: Ảnh minh họa chương trình nhận diện biển số xe 16](#_Toc184603619)

[Hình 2: Ảnh minh họa ảnh gốc 18](#_Toc184603620)

[Hình 3: Ảnh minh họa chuyển đổi sang thang xám 19](#_Toc184603621)

[Hình 4: Ảnh minh họa sau khi phát hiện biên 19](#_Toc184603622)

[Hình 5: Ảnh minh họa phát hiện vùng biển số và khoanh vùng 20](#_Toc184603623)

[Hình 6: Ảnh minh họa kết quả nhận dạng ký tự từ biển số 21](#_Toc184603624)

[Hình 7: Ảnh biển số rõ nét 26](#_Toc184603625)

[Hình 8: Ảnh biển số mờ hoặc điều kiện ánh sáng yếu 27](#_Toc184603626)

[Hình 9: Kết quả mã tỉnh/thành phố đầy đủ ký tự 27](#_Toc184603627)

[Hình 10: Kết quả khi mã tỉnh/thành phố mất đi 1 phần ký tự 28](#_Toc184603628)

[Hình 11: Ảnh đầu vào 28](#_Toc184603629)

[Hình 12: Ảnh đã chọn ROI 28](#_Toc184603630)

[Hình 13: Kết quả nhận diện 29](#_Toc184603631)

# **CHƯƠNG I: CƠ SỞ LÝ THUYẾT**

## **Khái niệm biểu số xe**

Tại Việt Nam, biển số xe cơ giới (còn gọi là biển kiểm soát xe hoặc biển số) là tấm biển được gắn trên mỗi xe cơ giới. Biển số này do cơ quan công an cấp, trong khi xe quân sự do Bộ Quốc phòng cấp khi xe được mua mới hoặc chuyển nhượng. Biển số được làm từ hợp kim nhôm sắt, có hình chữ nhật hoặc gần vuông, trên đó in các chữ cái và con số (các biển số xe dân sự không dùng các chữ cái I, J, O, Q, W, và chữ R chỉ dùng cho xe rơ-moóc và sơ-mi rơ-moóc). Những ký hiệu này cho biết vùng và địa phương quản lý, và khi tra cứu trên máy tính, còn có thể xác định được danh tính của người chủ hay đơn vị đã mua xe, thời gian mua xe nhằm phục vụ công tác an ninh. Đặc biệt, trên biển số còn có hình Quốc huy Việt Nam được dập nổi.

Tiêu chuẩn kích thước: Ở mỗi quốc gia thường có tiêu chuẩn kích thước khác nhau, nhưng tại Việt Nam, tỷ lệ kích thước của các biển số gần như tương đồng. Biển số xe có hai loại với kích thước như sau: loại dài có chiều cao 110 mm và chiều dài 470 mm; loại ngắn có chiều cao 200 mm và chiều dài 280 mm. Do đó, tỷ lệ chiều cao/chiều rộng được giới hạn như sau: 3.5 ≤ cao/rộng ≤ 6.5 (đối với biển một hàng) và 0.8 ≤ cao/rộng ≤ 1.5 (đối với biển hai hàng).

Số lượng ký tự trên biển số xe nằm trong khoảng từ 7 đến 9. Chiều cao của các chữ và số là 80 mm, chiều rộng là 40 mm. Từ những đặc điểm này, ta có thể thiết lập các thông số và quy định để chọn lọc những đối tượng phù hợp mà ta cần

## **1.2. Xử lý ảnh và OpenCV**

Xử lý ảnh là một nhánh con của xử lý tín hiệu số, tập trung vào việc thao tác và phân tích hình ảnh kỹ thuật số. Đây là một lĩnh vực khoa học phát triển mạnh mẽ trong những năm gần đây. Xử lý ảnh gồm 4 lĩnh vực chính:

- Xử lý nâng cao chất lượng ảnh: Cải thiện độ rõ nét, loại bỏ nhiễu, điều chỉnh độ sáng, độ tương phản…

- Nhận dạng ảnh: Phân loại và xác định các đối tượng trong ảnh.

- Nén ảnh: Giảm kích thước tệp ảnh để lưu trữ và truyền tải hiệu quả hơn.

- Truy vấn ảnh: Tìm kiếm hình ảnh dựa trên các đặc điểm cụ thể.

Sự phát triển của xử lý ảnh mang lại rất nhiều lợi ích trong cuộc sống của con người. Hiện nay xử lý ảnh đã được áp dụng một cách rộng rãi trong đời sống như:

- Photoshop: Chỉnh sửa ảnh chuyên nghiệp.

- Nén ảnh: Tiết kiệm dung lượng lưu trữ.

- Nhận dạng biển số xe, nhận dạng khuôn mặt: Hỗ trợ anh ninh, giao thông.

- Nhận dạng chữ viết: Chuyển đổi văn bản từ ảnh sang định dạng kỹ thuật số.

- Xử lý ảnh thiên văn, ảnh y tế: Phân tích dữ liệu khoa học.

OpenCV (Open Computer Vision) là thư viện mã nguồn mở hàng đầu cho xử lý ảnh, thị giác máy tính và máy học. OpenCV đươc lập trình dùng C/C++ nên có tốc độ xử lý hay tính toán rất nhanh, có thể sử dụng với các ứng dụng liên quan đến thời gian thực. Opencv có các interface cho C/C++, Python Java vì vậy hỗ trợ được cho Window, Linux, MacOs lẫn Android, iOS OpenCV sỡ hữu một cộng đồng với hơn 47 nghìn người dùng và số lượng tải về đã vượt quá 6 triệu lần. Opencv có rất nhiều ứng dụng như:

- Nhận dạng ảnh

- Xử lý hình ảnh

- Phục hồi hình ảnh/video

- Thực tế ảo

- Các ứng dụng khá

## **1.3. Phát hiện vị trí tách biển số xe**

### **1.3.1. Chuyển ảnh xám**

Ảnh xám (Gray Scale) là một loại ảnh mà bên trong có các màu là các sắc thái của màu xám với 256 cấp độ xám thay đổi từ màu đen đến màu trắng và có giá trị trong khoảng từ 0 đến 255, hay nói cách khác là cần có 8 bits hay 1 byte để có thể biểu diễn điểm ảnh này. Chúng ta cần việc phân biệt ảnh xám với các ảnh khác vì việc ảnh xám cung cấp ít thông tin hơn cho mỗi pixel. Với các ảnh màu thông thường thì đối với mỗi pixel sẽ có 3 trường thông tin còn đối với ảnh xám thì chỉ có 1 trường thông tin, việc chung ta dùng ảnh xám để giảm khối lượng thông tin sẽ giúp tăng tốc độ xử lý, đồng thời đơn giản hóa giải thuật nhưng vẫn đảm bảo được các tác vụ cần thiết.

Với đồ án này nhóm em sẽ thực hiện việc chuyển ảnh xám từ hệ màu HSV thay vì RGB vì trong không gian màu HSV chúng ta sẽ có ba giá trị chính là: Vùng màu (Hue), độ bão hòa (Saturation), cường độ sáng (Value). Do đó không gian màu HSV sẽ có thể thích nghi tốt hơn đối với sự ảnh hưởng của ánh sáng từ môi trường ngoài. Khi thực hiện chuyển đổi, ảnh xám mà ta cần là một ma trận với các giá trị cường độ sáng được tách ra từ hệ màu HSV

### **1.3.2. Giảm nhiễu bang bộ lọc Gaus**

**a), Nhiễu**

Nhiễu có thể hiểu căn bản là các chấm hạt nhỏ nằm trên hình ảnh. Nhiễu có thể làm biến dạng các chi tiết trong ảnh khiến cho chất lượng ảnh thấp.

Trên thực tế có nhiều loại nhiễu, nhưng người ta thường chia làm ba loại: nhiễu cộng, nhiễu nhân và nhiễu xung. Các tính chất của nhiễu thường là phù hợp với tần số cao và dựa trên cơ sở lý thuyết của các bộ lọc là chỉ truyền những tín hiệu ứng với tần số nhất định đi qua, đó là lý do nhiều người thường sử dụng bộ lọc thông thấp hoặc trung bình

**b), Bộ lọc Gauss (Gauss filter)**

Bộ lọc Gausssian có thể coi là bộ lọc hữu ích nhất, được sử dùng bằng cách tích chập ảnh đầu vào với một mặt nạ Gauss sau đó cộng chúng lại để được ảnh đầu ra.

Ý tưởng được đưa ra là giá trị tại mỗi điểm ảnh sẽ khá phụ thuộc vào những điểm ảnh nằm gần hơn là các điểm ảnh ở xa. Trọng số của sự phụ thuộc được lấy theo hàm Gaussian (cũng được sử dụng trong quy luật phân phối chuẩn)

Giả sử như ta có ảnh một chiều. Điểm ảnh nằm tại trung tâm sẽ có trọng số lớn nhất. Các điểm ảnh nằm tại những vị trí càng xa trung tâm sẽ có trọng số giảm dần khi mà khoảng cách từ các điểm đó đến vị trí trung tâm tăng lên. Vì vậy các điểm càng có vị trí gần trung tâm thì sẽ càng có nhiều đóng góp hơn đến giá trị điểm ngay trung tâm

### **1.3.3. Phát hiện cạnh Canny (Canny Edge Detetion)**

Trong một hình ảnh sẽ có nhiều bộ phận cấu thành như: vùng trơn, góc/cạnh và nhiễu. Thành phần cạnh có trong ảnh mang đặc trưng quan trọng, thông thường là thuộc đối tượng trong ảnh. Vì vậy, để có thể phát hiện cạnh trong ảnh, có nhiều giải thuật khác nhau như toán tử Sobel, toán tử Prewitt, Zero crossing .... và ở đây nhóm em sẽ chọn giải thuật Canny vì hương pháp này ra đời sau và được cải tiến dựa trên các phương pháp khác nên ít bị tác động của nhiễu và có khả năng phát hiện các biên yếu. Giải pháp này sẽ có 4 bước chính:

1. Giảm nhiễu (Noise reduction)

2. Tính toán Gradient (Gradient calculation)

3. Loại bỏ những điểm không phải là cực đại (Non-maximum suppression)

4. Lọc ngưỡng (Double threshold)

**a), Giảm nhiễu**

Để giảm nhiễu hay làm mờ ảnh ta sẽ dùng một mặt nạ Gauss với kích thước 5x5. Với kích thước 5x5 thì thường hoạt động phù hợp với giải thuật Canny.

**b), Tính toán Gradient**

Đầu tiên ta dùng phương pháp sobel với 2 ma trận Sobel X và Sobel Y (3x3) để tính đạo hàm Gx và Gy

A white background with black and white clouds

Description automatically generated

Khi tìm được gradient và hướng ta sẽ làm tròn về 4 hướng chính đó là: ngang (0 độ), chéo bên phải (45 độ), dọc (90 độ) và cuối cùng chéo bên trái (135 độ).



**c), Loại bỏ những điểm không phải cực đại**

Đến bước này, ta sẽ dùng một bộ lọc 3x3 lần lượt cho chạy qua các điểm ảnh trên ảnh gradient. Trong quá trình lọc, xem xét xem độ lớn gradient của điểm ảnh trung tâm có phải là lớn nhất so với các gradient ở vị trí xung quanh. Nếu đã là cực đại thì sẽ giữ điểm ảnh đó. Còn nếu điểm ảnh tại đó không phải lớn nhất so với lân cận, ta sẽ đưa độ lớn gradient của nó về không. Ta chỉ thực hiện so sánh pixel trung tâm với 2 pixel lân cận theo hướng của gradient.

**d), Lọc Ngưỡng**

Khi lọc ngưỡng là ta sẽ xét các điểm ảnh dương năm trên mặt nạ nhị phân từ kết quả của bước trước. Nếu giá trị gradient lớn hơn ngưỡng max\_val thì pixel đó chắc chắn là cạnh. Nếu giá trị gradient nhỏ hơn ngưỡng min\_val thì ta sẽ loại bỏ. Còn nếu các pixel có giá trị gradient nằm trong 2 ngưỡng trên và dưới ta sẽ kiểm tra xem nó có nằm liền kề với những pixel có gradient lớn hơn ngưỡng max hay không. Nếu thực sự liền kề thì ta sẽ giữ lại pixel đó, còn nếu không liền kề bất cứ pixel cạnh nào thì ta sẽ loại. Khi thực hiện xong bước này ta có thể sử dụng thêm một bước xử lý loại bỏ nhiễu nếu muốn.

### **1.3.4. Lọc biển số với contour**

a), **Một số phương pháp tìm contour**

* **Thuật toán Square Tracing**

Bắt đầu xét từ pixel ngoài cùng bên trái phía dưới rồi từ từ đi lên cho tới khi gặp pixel có giá trị bằng 1 thì sẽ bắt đầu di chuyển theo phương pháp bên dưới:

- Nếu gặp Pixel có giá trị bằng 1 thì rẽ trái.

- Nếu gặp Pixel có giá trị bằng 0 thì rẽ phải.

- Tiếp đi cho tới khi quay lại pixel ban đầu thì dừng lại.

* **Thuật toán Moore-Neighbor**

Đối với thuật toán này thì sẽ có một ít sự khác biệt so với thuật toán Square Tracking. Nói cụ thể hơn là khi gặp được pixel có giá trị bằng 1 đầu tiên (pixel start) thì chúng ta sẽ cho trở lại pixel trước đó, tiếp theo là di chuyển vòng qua các pixel thuộc 8-connected theo chiều kim đồng hồ đến khi gặp pixel khác có giá trị bằng 1. Điều kiện kết thúc thì sẽ tương tự với thuật toán Square Tracking.

* **Thuật toán Suzuki’s Tracing**

Thuật toán Suzuki’s Tracing là một thuật toán mà thư viện OpenCV sử dụng, cùng với khả năng xác định biên của vật thể như các phương pháp nêu trên thì phương pháp Suzuki’s Tracing còn có thể phân biệt được đường biên ngoài (Outer) hay là đường biên trong (Hole) của vật thể.

**b), Lọc biển số**

Để lọc ra biển số thì ban đầu ta làm xấp xỉ contour như hình đa giác và chỉ chọn ra những đa giác có 4 cạnh. Điều này nghĩa là khi ta xấp xỉ contour bộ nhớ chỉ ghi lại những vị trí các đỉnh của đa giác đó thành một mảng. Các cạnh của đa giác sẽ bằng với số đỉnh và bằng với chiều dài của mảng đó.

Sau đó ta thực hiện việc tính toán tỉ lệ cao/rộng và diện tích phù hợp cho biển số rồi tiếp tục lưu lại hết các biển số có trong hình dưới dạng là tọa độ các đỉnh.

Cuối cùng thì thực hiện cắt ảnh của biển số từ những tọa độ vị trí đã biết để sử dụng cho việc sau là “Tách các kí tự trong biển số”. Chú ý là ở đây ta sẽ xử lý trên ảnh nhị phân để giúp máy tính chạy nhanh hơn, tốn ít thời gian hơn

## **1.4. Nhận diện kí tự**

### **1.4.1. Tổng quan về lý thuyết**

**Thư viện EasyOCR**

EasyOCR là một thư viện mã nguồn mở dùng để nhận diện ký tự quang học (OCR - Optical Character Recognition), được phát triển bởi Jaided AI và xây dựng dựa trên công nghệ PyTorch. Thư viện này nổi bật với khả năng hỗ trợ hơn 80 ngôn ngữ, bao gồm cả tiếng Việt, giúp nhận diện chữ in, chữ viết tay trên hình ảnh hoặc tài liệu với độ chính xác cao. EasyOCR sử dụng mạng nơ-ron nhân tạo để tối ưu quá trình nhận diện ký tự, hoạt động hiệu quả ngay cả khi hình ảnh bị méo, chữ bị nghiêng hoặc chất lượng thấp. Quy trình của thư viện bao gồm:

- Tiền xử lý ảnh.

- phát hiện vùng chứa ký tự.

- nhận diện ký tự.

- xuất kết quả dưới dạng danh sách văn bản kèm độ tin cậy.

Với giao diện lập trình đơn giản, người dùng chỉ cần cài đặt thư viện qua pip và sử dụng vài dòng lệnh Python để trích xuất văn bản từ hình ảnh. EasyOCR được ứng dụng rộng rãi trong các lĩnh vực như nhận diện biển số xe, trích xuất văn bản từ tài liệu, hay phân tích văn bản từ ảnh chụp màn hình, mang lại hiệu quả cao trong việc xử lý hình ảnh và dữ liệu.

# **CHƯƠNG II: XÂY DỰNG HỆ THỐNG**

## **2.1. Bài toán nói chung**

Nhận diện đối tượng trong ảnh là một lĩnh vực quan trọng trong Xử lý ảnh và Thị giác máy tính, với mục tiêu xác định và phân loại các đối tượng cụ thể trong hình ảnh hoặc video. Đây là một bài toán có ứng dụng rộng rãi, từ giám sát an ninh, y tế, giao thông, đến tự động hóa trong công nghiệp.

Ví dụ minh họa bài toán nhận dạng:

* Nhận diện khuôn mặt người trong hình ảnh chụp đám đông.
* Nhận diện biển số xe
* Phát hiện biển báo giao thông trên đường.
* Xác định các sản phẩm trên băng chuyền trong dây chuyền sản xuất.

Hình ảnh minh họa:  
A screenshot of a computer

Description automatically generated

Hình 1: Ảnh minh họa chương trình nhận diện biển số xe

**Các công việc chính trong bài toán nhận dạng đối tượng trong ảnh**

Để nhận dạng một đối tượng trong ảnh, hệ thống cần thực hiện các bước cơ bản sau:

* **Thu thập dữ liệu (Data Collection):**
  + Thu thập các hình ảnh hoặc video chứa đối tượng cần nhận diện từ nhiều nguồn khác nhau (camera, internet, v.v.).
  + Đảm bảo dữ liệu đủ đa dạng về ánh sáng, góc độ, kích thước và môi trường.
* **Tiền xử lý ảnh (Image Preprocessing):**
  + Xử lý dữ liệu ảnh để chuẩn bị cho các bước tiếp theo, bao gồm:
    - Chuyển đổi ảnh về dạng xám (Grayscale).
    - Cân bằng độ sáng, tương phản (Histogram Equalization).
    - Lọc nhiễu (Noise Reduction).
    - Cắt ảnh (Cropping) để tập trung vào đối tượng chính.
* **Phát hiện đối tượng (Object Detection):**
  + Sử dụng các kỹ thuật phát hiện đối tượng để xác định vị trí đối tượng trong ảnh.
  + Ví dụ:
    - Sử dụng các phương pháp dựa trên ngưỡng (Thresholding).
    - Sử dụng mô hình học sâu như YOLO, Faster R-CNN, hoặc SSD.
* **Trích xuất đặc trưng (Feature Extraction):**
  + Trích xuất các đặc trưng quan trọng từ đối tượng, ví dụ:
    - Đường biên (Edges).
    - Đặc điểm hình dạng (Shape).
    - Văn bản (Text) hoặc ký tự nếu có.
* **Nhận dạng (Recognition):**
  + Áp dụng các thuật toán nhận dạng để phân loại đối tượng hoặc giải mã thông tin.
  + Với biển số xe, bước này bao gồm nhận diện ký tự và chuyển chúng thành chuỗi văn bản.
* **Hậu xử lý và hiển thị kết quả (Post-Processing and Result Display):**
  + Xử lý kết quả nhận dạng để đưa ra thông tin hữu ích.
  + Ví dụ: Hiển thị chuỗi biển số xe trên màn hình hoặc lưu kết quả vào cơ sở dữ liệu.

## **2.2. Các kỹ thuật sử dụng trong bài toán nhận dạng biển số xe**

Bài toán nhận dạng biển số xe sử dụng một số kỹ thuật trong xử lý ảnh và nhận dạng ký tự. Dưới đây là mô tả chi tiết từng kỹ thuật, bao gồm tư tưởng, minh họa, và ưu nhược điểm:

### **2.2.1. Xử lý ảnh (Preprocessing)**

* **Tư tưởng kỹ thuật:** Xử lý ảnh nhằm chuẩn hóa và làm nổi bật các đặc điểm quan trọng để nhận diện biển số dễ dàng hơn. Gồm các bước:
  + **Chuyển đổi sang thang xám (Grayscale):** Loại bỏ thông tin màu, chỉ giữ lại cường độ sáng, giảm kích thước dữ liệu.
  + **Làm mờ (Blurring):** Dùng Gaussian Blur để làm mượt ảnh, giảm nhiễu, giúp xác định viền tốt hơn.
  + **Phát hiện biên (Edge Detection):** Dùng thuật toán Canny để tìm các đường biên, làm nổi bật vùng có khả năng chứa biển số xe.
* **Hình minh họa:**
  + Ảnh gốc:

A license plate on a motorcycle

Description automatically generated

Hình 2: Ảnh minh họa ảnh gốc

* Ảnh thang xám:

A license plate on a motorcycle

Description automatically generated

Hình 3: Ảnh minh họa chuyển đổi sang thang xám

* Ảnh sau khi phát hiện biên:

A black and white image of a sign

Description automatically generated

Hình 4: Ảnh minh họa sau khi phát hiện biên

* **Ưu điểm:**
  + Giảm nhiễu, làm nổi bật các đặc điểm quan trọng.
  + Chuẩn bị dữ liệu tốt hơn cho các bước xử lý sau.
* **Nhược điểm:**
  + Độ chính xác bị ảnh hưởng nếu ảnh đầu vào có nhiều nhiễu.
  + Phụ thuộc vào tham số của các thuật toán (như Gaussian Blur và Canny).

### **2.2.2. Phát hiện vùng biển số xe (Region Detection)**

* **Tư tưởng kỹ thuật:** Dùng thuật toán tìm đường viền (**Contours**) để phát hiện các vùng hình chữ nhật, từ đó xác định khu vực chứa biển số xe.
  + Tìm các đường viền lớn nhất trong ảnh.
  + Kiểm tra các đường viền có hình dạng là tứ giác (biển số xe thường có dạng hình chữ nhật).
* **Hình minh họa:**
  + Vùng biển số được phát hiện và khoanh vùng:

A license plate on a motorcycle

Description automatically generated

Hình 5: Ảnh minh họa phát hiện vùng biển số và khoanh vùng

* **Ưu điểm:**
  + Hiệu quả cao khi ảnh đầu vào có độ tương phản tốt.
  + Không cần dữ liệu huấn luyện.
* **Nhược điểm:**
  + Độ chính xác giảm nếu ảnh bị che khuất, biển số bị méo.
  + Phụ thuộc nhiều vào chất lượng ảnh.

### **2.2.3. Nhận dạng ký tự (Optical Character Recognition - OCR)**

* **Tư tưởng kỹ thuật:** Sử dụng thư viện **EasyOCR** để nhận diện các ký tự trên vùng biển số đã được cắt.
  + OCR dựa trên các mô hình học sâu để nhận diện ký tự trong hình ảnh.
* **Hình minh họa:**
  + Kết quả nhận dạng ký tự từ biển số:

A screenshot of a computer

Description automatically generated

Hình 6: Ảnh minh họa kết quả nhận dạng ký tự từ biển số

* **Ưu điểm:**
  + Dễ sử dụng, hỗ trợ nhiều ngôn ngữ.
  + Độ chính xác cao với ảnh rõ ràng.
* **Nhược điểm:**
  + Hiệu suất giảm với ảnh có chất lượng thấp hoặc biển số bị che khuất.
  + Tốn tài nguyên tính toán hơn các phương pháp truyền thống.

### **2.2.4. Tra cứu tỉnh/thành phố từ mã biển số**

* **Tư tưởng kỹ thuật:** Lấy hai chữ số đầu tiên từ kết quả nhận diện ký tự (biển số xe), sau đó tra cứu trong từ điển mã tỉnh/thành phố để xác định địa phương.
* **Ưu điểm:**
  + Dễ triển khai, không cần thuật toán phức tạp.
  + Tăng tính trực quan trong ứng dụng.
* **Nhược điểm:**
  + Phụ thuộc hoàn toàn vào độ chính xác của OCR.
  + Không nhận diện được nếu biển số bị sai ký tự hoặc không đầy đủ.

## **2.3. Ngôn ngữ lập trình và các thư viện sử dụng**

### **2.3.1. Python**

Python là một ngôn ngữ lập trình bậc cao, đa năng và dễ sử dụng. Python được lựa chọn để xây dựng hệ thống vì các ưu điểm:

* **Dễ đọc và dễ học**: Cú pháp đơn giản, thân thiện với người dùng, phù hợp với người mới bắt đầu.
* **Hỗ trợ nhiều thư viện mạnh mẽ**: Python cung cấp các thư viện hỗ trợ xử lý ảnh, học máy, và giao diện người dùng như OpenCV, EasyOCR, PyQt6, giúp đơn giản hóa quá trình phát triển.
* **Khả năng mở rộng**: Python có thể tích hợp dễ dàng với các công nghệ khác và có cộng đồng hỗ trợ rộng lớn.
* **Hiệu suất cao với thư viện tối ưu**: Mặc dù Python không nhanh như C++ nhưng các thư viện như OpenCV (được viết bằng C++) vẫn đảm bảo hiệu suất tốt trong xử lý ảnh thời gian thực.

### **2.3.2. Các thư viện**

**PyQt6**:

* **Mục đích**: Tạo giao diện người dùng đồ họa (GUI) cho hệ thống nhận diện biển số xe.
* **Các thành phần sử dụng**:
  + QApplication, QMainWindow: Quản lý ứng dụng và cửa sổ chính.
  + QFileDialog: Hỗ trợ chọn tệp ảnh hoặc video từ máy tính.
  + QPixmap, QImage: Hiển thị ảnh trên giao diện.
  + loadUi: Nạp file .ui (thiết kế giao diện) vào ứng dụng.
  + QtCore: Hỗ trợ quản lý tín hiệu, luồng dữ liệu, và xử lý thời gian thực.

**OpenCV**:

* **Mục đích**: Xử lý và phân tích ảnh/video để trích xuất vùng chứa biển số xe.
* **Các tính năng sử dụng**:
  + Đọc ảnh và video (cv2.imread, cv2.VideoCapture).
  + Tiền xử lý ảnh: chuyển đổi màu sắc (RGB, grayscale), lọc nhiễu, làm mịn ảnh (cv2.cvtColor, cv2.GaussianBlur).
  + Xác định và cắt vùng biển số dựa trên phát hiện đường biên (cv2.findContours, cv2.boundingRect).

**EasyOCR**:

* **Mục đích**: Nhận diện ký tự từ vùng biển số xe đã cắt.
* **Lý do sử dụng**:
  + Hỗ trợ tiếng Việt, phù hợp với các biển số xe Việt Nam.
  + Dễ sử dụng, không yêu cầu cấu hình phức tạp.
* **Cách hoạt động:**
  + Khởi tạo đối tượng Reader.
  + Nhận diện văn bản từ ảnh với reader.readtext.

**Các thư viện hỗ trợ khác**:

* **os**: Quản lý đường dẫn và tệp tin.
* **sys**: Xử lý các lệnh liên quan đến hệ thống (thoát chương trình, kiểm tra môi trường).
* **numpy**: Xử lý dữ liệu dưới dạng ma trận, hỗ trợ các phép toán liên quan đến pixel ảnh.

## **2.4. Xây dựng hệ thống nhận diện biển số xe**

### **2.4.1. Tiền xử lý ảnh**

**a), Chuyển đổi ảnh xám**

* Ảnh gốc được chuyển đổi từ định dạng màu RGB (hoặc BGR trong OpenCV) sang ảnh xám để giảm số lượng kênh màu từ 3 xuống 1.
* Điều này giúp tập trung vào các chi tiết cần thiết như biên và vùng tương phản cao.

|  |
| --- |
| grayscale = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) |

**b), Làm Mờ Ảnh (Gaussian Blur)**

* Kỹ thuật Gaussian Blur được sử dụng để giảm nhiễu trong ảnh, giúp làm mịn và loại bỏ các chi tiết không cần thiết như hạt nhiễu hoặc vùng không đồng đều.
* Kích thước kernel phù hợp được chọn (ví dụ: (5, 5)) để đạt hiệu quả làm mờ tối ưu.

|  |
| --- |
| blurred = cv2.GaussianBlur(grayscale, (5, 5), 0) |

**c), Phát Hiện Cạnh (Canny Edge Detection)**

* Áp dụng thuật toán phát hiện cạnh Canny để xác định các biên sắc nét trong ảnh.
* Các ngưỡng (threshold) được tùy chỉnh để phát hiện các đường viền chính xác.

|  |
| --- |
| edged = cv2.Canny(blurred, 10, 200) |

### **2.4.2. Phát hiện vùng biển số**

**a), Lọc Theo Contours**

* Sử dụng kỹ thuật tìm contour trong OpenCV để phát hiện các vùng có đường biên rõ nét.
* Các vùng có hình dạng chữ nhật và kích thước phù hợp được ưu tiên để nhận diện là biển số.

|  |
| --- |
| contours, \_ = cv2.findContours(edged, cv2.RETR\_TREE, cv2.CHAIN\_APPROX\_SIMPLE)  for c in contours:  perimeter = cv2.arcLength(c, True)  approximation = cv2.approxPolyDP(c, 0.02 \* perimeter, True)  if len(approximation) == 4: # Hình chữ nhật  number\_plate\_shape = approximation  break |

**b), Cắt Vùng Biển Số**

* Sau khi xác định được vùng biển số, sử dụng hàm cv2.boundingRect() để cắt và trích xuất vùng biển số.

|  |
| --- |
| (x, y, w, h) = cv2.boundingRect(number\_plate\_shape)  number\_plate = grayscale[y:y + h, x:x + w] |

### **2.4.3. Nhận diện ký tự**

**a), EasyOCR**

* Thư viện **EasyOCR** được sử dụng để nhận diện ký tự trên vùng biển số đã được trích xuất.
* Kỹ thuật OCR dựa trên mạng neural tiên tiến, hỗ trợ nhận diện văn bản đa ngôn ngữ.

|  |
| --- |
| reader = Reader(['en'])  detection = reader.readtext(number\_plate) |

**b), Hiển thị kết quả**

* Sau khi nhận diện, kết quả được hiển thị dưới dạng văn bản.
* Dựa trên hai ký tự đầu tiên, hệ thống tra cứu mã tỉnh/thành phố và hiển thị thông tin tương ứng.

|  |
| --- |
| license\_plate\_text = detection[0][1] # Biển số chỉ có một hàng province\_code = license\_plate\_text[:2] # Lấy 2 chữ số đầu làm mã tỉnh/thành phố province\_name = province\_dict.get(province\_code, "Không xác định") |

### **2.4.4. Giao diện người dùng**

Hệ thống được thiết kế giao diện bằng **Qt Designer** và tích hợp vào ứng dụng Python bằng thư viện **PyQt6**. Giao diện gồm:

* Nút chọn ảnh và hiển thị ảnh đã xử lý.
* Hiển thị biển số và thông tin tỉnh/thành phố nhận diện được.

|  |
| --- |
| self.btn\_anh.clicked.connect(self.open\_and\_process\_image)  self.btn\_chon\_tc.clicked.connect(self.select\_and\_process\_roi) |

# **CHƯƠNG III: THỰC NGHIỆM**

## **3.1. Dữ liệu sử dụng**

Dữ liệu sử dụng trong bài toán nhận diện biển số xe bao gồm các hình ảnh chụp biển số xe từ nhiều nguồn khác nhau, đảm bảo đa dạng về điều kiện ánh sáng, góc chụp và chất lượng hình ảnh.

Các ảnh bảo gồm:

- Biển số xe có chất lượng cao, rõ ràng.

- Biển số xe bị mờ, bị nhiễu hoặc chụp trong điều kiện ánh sáng yếu.

- Biển số xe có nhiều định dạng khác nhau (1 hàng hoặc 2 hàng).

Ngoài ra, bài toán sử dụng thư viện EasyOCR để hỗ trợ nhận diện ký tự từ hình ảnh và một từ điển mã tỉnh/thành phố để xác định vị trí của biển số.

## **3.2. Kết quả thực nghiệm**

**Nhận diện biển số:**

Hệ thống cho kết quả nhận diện tốt với các biển số rõ ràng, nhưng giảm hiệu suất với các biển số bị mở hoặc có ánh sáng không đồng đều. Ví dụ:

- Ảnh rõ nét: Độ chính xác đạt trên 90%:

A screenshot of a white and black license plate

Description automatically generated

Hình 7: Ảnh biển số rõ nét

- Ảnh mờ hoặc điều kiện ánh sáng yếu: Độ chính xác giảm xuống khoảng 60-70%:

A close up of a car

Description automatically generated

Hình 8: Ảnh biển số mờ hoặc điều kiện ánh sáng yếu

**Nhận diện tỉnh/thành phố:**

Hệ thống dựa trên mã biển số xe để xác định tỉnh/thành phố. Kết quả cho thấy:

- Mã tỉnh/thành phố nhận diện chính xác đối với các biển số đầy đủ ký tự:

A motorcycle with a license plate

Description automatically generated

Hình 9: Kết quả mã tỉnh/thành phố đầy đủ ký tự

- Trong một số trường hợp biển số bị mờ hoặc mất một phần ký tự, mã tỉnh/thành phố không nhận diện được:

A license plate on a motorcycle

Description automatically generated

Hình 10: Kết quả khi mã tỉnh/thành phố mất đi 1 phần ký tự

**Minh họa kết quả:**

- Ảnh đầu vào:

The front of a car

Description automatically generated

Hình 11: Ảnh đầu vào

- Ảnh đã chọn ROI (Region of Interest):

A close up of a license plate

Description automatically generated

Hình 12: Ảnh đã chọn ROI

- Kết quả nhận diện:

A white car parked in a parking lot

Description automatically generated

Hình 13: Kết quả nhận diện

Hệ thống hoạt động hiệu quả nhất trong điều kiện ảnh đầu vào rõ ràng và được căn chỉnh đúng tỷ lệ.

# **KẾT LUẬN**

Hệ thống nhận diện biển số xe tự động sử dụng OpenCV đã được triển khai thành công, đáp ứng các mục tiêu đặt ra trong bài tập. Qua quá trình nghiên cứu và xây dựng hệ thống, các kết quả chính có thể được tổng kết như sau:3

1. **Tính khả thi và ứng dụng thực tiễn:**  
   Hệ thống đã chứng minh tính khả thi trong việc áp dụng xử lý ảnh và nhận diện ký tự để tự động phát hiện và trích xuất thông tin từ biển số xe. Đây là nền tảng quan trọng để phát triển các ứng dụng thực tế như quản lý giao thông, kiểm soát ra vào tại bãi đỗ xe, hoặc giám sát an ninh.
2. **Hiệu quả của OpenCV và EasyOCR:**

* **OpenCV** đã thể hiện vai trò vượt trội trong việc xử lý ảnh, phát hiện vùng biển số xe, và tiền xử lý ảnh để tăng chất lượng nhận diện.
* **EasyOCR** cho thấy độ chính xác cao trong việc nhận diện ký tự, ngay cả với các ký tự tiếng Việt và biển số xe có cấu trúc đặc biệt.

1. **Hạn chế và thách thức:**  
   Dù đạt được nhiều kết quả tích cực, hệ thống vẫn gặp một số hạn chế:

* Chất lượng ảnh đầu vào kém (mờ, nhiễu, hoặc ánh sáng không đồng đều) có thể làm giảm độ chính xác của nhận diện.
* Hệ thống cần được tối ưu hơn để xử lý nhanh trong trường hợp áp dụng cho luồng dữ liệu lớn hoặc thời gian thực.

**Hướng phát triển:**

* **Nâng cao độ chính xác:** Áp dụng các kỹ thuật học sâu tiên tiến hơn để cải thiện khả năng nhận diện trong các điều kiện phức tạp.
* **Xử lý thời gian thực:** Tích hợp hệ thống với camera để nhận diện trực tiếp và tối ưu tốc độ xử lý.
* **Mở rộng ứng dụng:** Xây dựng thêm các tính năng như phân loại phương tiện, giám sát hành trình, hoặc liên kết với cơ sở dữ liệu giao thông.

# **TÀI LIỆU THAM KHẢO**

[1]. Lê Mỹ Hà(2019), *Giáo trình thị giác máy tính và ứng dụng,* Nhà xuất bản Đại học quốc gia thành phố Hồ Chí Minh.

[2]. Nguyễn Quang Hoan(2006), *Xử lý ảnh*, Nhà xuất bản Học viện Bưu chính viễn thông.