**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI TP. HỒ CHÍ MINH**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ ĐIỆN, ĐIỆN TỬ**



**MÔN HỌC**

**Chuyên đề hệ thống giao thông thông minh**

**CHỦ ĐỀ:**

**XÂY DỰNG HỆ THỐNG NHẬN DIỆN BIỂN SỐ XE.**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn: ThS. Vũ Đình Long** | |
| **Sinh viên thực hiện:**   1. **Hồ Thiên Phúc** 2. **Nguyễn Minh Phú** 3. **Nguyễn Lê Huy Tâm** 4. **Nguyễn Ngọc Khánh Tân** 5. **Hồ Lý Minh Phước** | **MSSV:**  **080206013420**  **084205002407**  **056206011188**  **079206001772**  **074206000514** |
| **Lớp: 7480201903613** | |

**TP. Hồ Chí Minh, Ngày 28 Tháng 01 Năm 2026**

MỤC LỤC

[DANH MỤC BẢNG BIỂU V](#_Toc220599244)

[THUẬT NGỮ VIẾT TẮT VI](#_Toc220599245)

[LỜI MỞ ĐẦU 1](#_Toc220599246)

[LỜI CẢM ƠN 2](#_Toc220599247)

[NHẬN XÉT CỦA GIẢNG VIÊN 3](#_Toc220599248)

[BẢNG PHÂN CÔNG CÔNG VIỆC VÀ MỨC ĐỘ HOÀN THÀNH 4](#_Toc220599249)

[CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI 5](#_Toc220599250)

[**1.1. Bối cảnh thực tế và giải pháp 5**](#_Toc220599251)

[**1.1.1. Bối cảnh thực tế 5**](#_Toc220599252)

[**1.1.2. Giải pháp công nghệ 5**](#_Toc220599253)

[**1.2. Vấn đề cần giải quyết (Problem Statement) 5**](#_Toc220599254)

[**1.3. Mục tiêu và Yêu cầu tổng quan (Objectives & Requirements) 6**](#_Toc220599255)

[**1.3.1. Mục tiêu kỹ thuật (Measurable Objectives) 6**](#_Toc220599256)

[**1.3.2. Chỉ tiêu đánh giá (KPIs) 6**](#_Toc220599257)

[**1.4. Hạn chế thực tế (Practical Limitations) 7**](#_Toc220599258)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT & PHÂN TÍCH YÊU CẦU 8](#_Toc220599259)

[**2.1. Cơ sở lý thuyết 8**](#_Toc220599260)

[**2.1.1. Computer Vision trong bài toán nhận dạng biển số xe 8**](#_Toc220599261)

[**2.1.2. Hệ thống cảm biến (Sensor Systems) 8**](#_Toc220599262)

[**2.1.3. Hệ thống truyền thông và giao tiếp dữ liệu 8**](#_Toc220599263)

[**2.1.4. Mô hình phát hiện và lý thuyết dòng giao thông (tham khảo) 9**](#_Toc220599264)

[**2.2. Phân tích yêu cầu hệ thống 9**](#_Toc220599265)

[**2.2.1. Yêu cầu chức năng (Functional Requirements – FR) 9**](#_Toc220599266)

[**2.2.2. Yêu cầu phi chức năng (Non-Functional Requirements – NFR) 9**](#_Toc220599267)

[**2.2.3. Đối tượng sử dụng và kịch bản sử dụng (Actors & User Scenarios) 10**](#_Toc220599268)

[CHƯƠNG 3. THIẾT KẾ HỆ THỐNG 11](#_Toc220599269)

[**3.1. Tổng quan kiến trúc hệ thống (System Architecture) 11**](#_Toc220599270)

[**3.2. Kiến trúc CV Pipeline 11**](#_Toc220599271)

[**3.2.1. Input 11**](#_Toc220599272)

[**3.2.2. Preprocessing (Tiền xử lý ảnh) 11**](#_Toc220599273)

[**3.2.5. Character Segmentation (Phân tách ký tự) 12**](#_Toc220599274)

[**3.2.6. Character Recognition (Nhận dạng ký tự) 12**](#_Toc220599275)

[**3.3. Data Flow Diagram (DFD) 12**](#_Toc220599276)

[**3.3.1. Mô tả luồng dữ liệu 12**](#_Toc220599277)

[**3.3.2. DFD mức khái quát 13**](#_Toc220599278)

[**3.4. Sơ đồ Component / Module 13**](#_Toc220599279)

[**3.5. Kiến trúc Edge–Cloud (Edge–Cloud Architecture) 14**](#_Toc220599280)

[**3.5.1. Luật quyết định (Decision Logic) 14**](#_Toc220599281)

[**3.5.2. Thuật toán sử dụng 14**](#_Toc220599282)

[CHƯƠNG 4: HIỆN THỰC VÀ TRIỂN KHAI 15](#_Toc220599283)

[**4.1. Công cụ và Công nghệ sử dụng 15**](#_Toc220599284)

[**4.1.1. Ngôn ngữ lập trình 15**](#_Toc220599285)

[**4.1.2. Thư viện xử lý ảnh và Computer Vision 15**](#_Toc220599286)

[**4.1.3. Thuật toán Machine Learning 15**](#_Toc220599287)

[**4.1.4. Các thư viện hỗ trợ khác 16**](#_Toc220599288)

[**4.1.5. Môi trường phát triển 16**](#_Toc220599289)

[**4.2. Mô tả các Module chính của Hệ thống 16**](#_Toc220599290)

[**4.2.1. Module Tiền xử lý ảnh (Preprocess.py) 16**](#_Toc220599291)

[**4.2.2. Module Sinh dữ liệu huấn luyện (GenData.py) 17**](#_Toc220599292)

[**4.2.3. Module Nhận dạng từ ảnh tĩnh (Image\_test2.py) 17**](#_Toc220599293)

[**4.2.4. Module Nhận dạng từ video (Video\_test2.py) 19**](#_Toc220599294)

[**4.3. Sơ đồ luồng thực thi (Flowchart) 19**](#_Toc220599295)

[**4.3.1. Luồng chính của hệ thống: 20**](#_Toc220599296)

[**4.3.2. Các điểm quyết định (Decision Points): 21**](#_Toc220599297)

[**4.3.3. Tối ưu hóa hiệu năng: 21**](#_Toc220599298)

[**4.4. Ánh xạ từ Thiết kế sang Code (Design to Code Mapping) 21**](#_Toc220599299)

[**4.4.1. Cấu trúc thư mục dự án: 22**](#_Toc220599300)

[**4.4.2. Code snippets quan trọng: 22**](#_Toc220599301)

[CHƯƠNG 5. KIỂM THỬ – ĐÁNH GIÁ 25](#_Toc220599302)

[**1. Test cases (nhiều bối cảnh: sáng/tối, đông/vắng, occlusion…) 25**](#_Toc220599303)

[**2. Metrics (accuracy, FPS, latency…) 25**](#_Toc220599304)

[**3. Hạn chế 25**](#_Toc220599305)

[**4. Điều kiện triển khai thực tế 25**](#_Toc220599306)

[**5. Tiềm năng cải tiến 25**](#_Toc220599307)

[CHƯƠNG 6. KẾT LUẬN & HƯỚNG PHÁT TRIỂN 26](#_Toc220599308)

[**1. Summary essence của hệ thống 26**](#_Toc220599309)

[**2. Ý nghĩa đối với ITS 26**](#_Toc220599310)

[**3. Hướng mở rộng: 26**](#_Toc220599311)

[TÀI LIỆU THAM KHẢO 27](#_Toc220599312)

# DANH MỤC HÌNH ẢNH

[Hình 1 – Nhận diện bảng số xe 1 26](#_Toc220601846)

[Hình 2 – Nhận diện bảng số xe 2 26](#_Toc220601847)

[Hình 3 - Bảng số xe mẫu 1 27](#_Toc220601848)

[Hình 4 - Kết quả 1 27](#_Toc220601849)

[Hình 5 - Bảng số xe hơi mẫu 1 28](#_Toc220601850)

[Hình 6 - Kết quả bảng số xe hơi mẫu 1 28](#_Toc220601851)

[Hình 7 - Bảng số xe mẫu 2 29](#_Toc220601852)

[Hình 8 - Nhận diện bảng số xe mẫu 2.1 29](#_Toc220601853)

[Hình 9 - Nhận diện bảng số xe mẫu 2.2 30](#_Toc220601854)

[Hình 10 - Kết quả bảng số xe mẫu 2.1 31](#_Toc220601855)

[Hình 11 - Kết quả bảng số xe mẫu 2.2 31](#_Toc220601856)

[Hình 12 - Ảnh chụp trắng đen để check kết quả mẫu xe 2 32](#_Toc220601857)

[Hình 13 - Ảnh phóng to kết quả mẫu xe 2 33](#_Toc220601858)

[Hình 14 - Bảng số xe hơi mẫu 2 33](#_Toc220601859)

[Hình 15 - Kết quả bảng số xe hơi mẫu 2 34](#_Toc220601860)

[Hình 16 - Nhận diện bảng số xe mẫu 3 34](#_Toc220601861)

[Hình 17 - Kết quả bảng số xe mẫu 3.1 35](#_Toc220601862)

[Hình 18 - Kết quả bảng số xe mẫu 3.2 36](#_Toc220601863)

# DANH MỤC BẢNG BIỂU

[Bảng 1: Hai mô hình phổ biến trong ITS 12](#_Toc220546162)

[Bảng 2: Các thuật toán sử dụng trong các module dự án 13](#_Toc220546163)

[Bảng 3: Các điểm quyết định 19](#_Toc220546164)

[Bảng 4: Mô tả các thành phần thiết kế của dự án 20](#_Toc220546165)

# THUẬT NGỮ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tiếng Anh** | **Tiếng Việt** | **Viết Tắt** |
| **1** | Intelligent Transportation Systems | Hệ thống giao thông thông minh | ITS |
| **2** | Automatic License Plate Recognition | Nhận diện biển số xe tự động | ALPR |
| **3** | Computer Vision | Thị giác máy tính | CV |
| **4** | K-Nearest Neighbors | Thuật toán K láng giềng gần nhất | KNN |
| **5** | Data Flow Diagram | Sơ đồ luồng dữ liệu | DFD |
| **6** | Key Performance Indicator | Chỉ số đánh giá hiệu suất | KPI |
| **7** | Optical Character Recognition | Nhận dạng ký tự quang học | OCR |
| **8** | Region of Interest | Vùng quan tâm | ROI |
| **9** | Traffic Management Center | Trung tâm quản lý giao thông | TMC |

# LỜI MỞ ĐẦU

Trong bối cảnh đô thị hóa nhanh chóng và sự gia tăng mạnh mẽ của các phương tiện giao thông cá nhân, bài toán quản lý giao thông và kiểm soát phương tiện đang trở thành một thách thức lớn đối với các đô thị hiện đại. Tại Việt Nam, các mô hình quản lý truyền thống trong bãi giữ xe, cổng ra vào khu dân cư, trường học hay cơ quan vẫn còn phụ thuộc nhiều vào thao tác thủ công, gây tốn kém nhân lực, giảm hiệu quả vận hành và tiềm ẩn nhiều rủi ro sai sót.

Sự phát triển của **Hệ thống giao thông thông minh (Intelligent Transportation Systems – ITS)** đã mở ra hướng tiếp cận mới, trong đó các công nghệ như **Computer Vision,** **Machine Learning** và **xử lý ảnh số** đóng vai trò trung tâm trong việc tự động hóa quá trình giám sát và quản lý phương tiện. Một trong những ứng dụng tiêu biểu và có tính thực tiễn cao của ITS là **hệ thống nhận diện biển số xe tự động (Automatic License Plate Recognition – ALPR).**

Xuất phát từ nhu cầu thực tế đó, nhóm thực hiện đề tài **“Xây dựng hệ thống nhận diện biển số xe”** trong khuôn khổ môn họcChuyên đề Hệ thống Giao thông Thông minh*.* Đề tài tập trung nghiên cứu và hiện thực một hệ thống nhận diện biển số xe Việt Nam dựa trên các kỹ thuật xử lý ảnh truyền thống kết hợp với thuật toán học máy K-Nearest Neighbors (KNN), nhằm xây dựng một giải pháp có tính ứng dụng, phù hợp với điều kiện phần cứng phổ thông và phục vụ mục tiêu học tập, nghiên cứu.

Thông qua đề tài này, nhóm mong muốn củng cố kiến thức về ITS, Computer Vision và Machine Learning, đồng thời hiểu rõ hơn quy trình thiết kế – triển khai một hệ thống nhận diện trong thực tế, làm nền tảng cho các nghiên cứu và ứng dụng nâng cao trong tương lai.

# LỜI CẢM ƠN

Nhóm xin gửi lời cảm ơn chân thành đến **Giảng viên hướng dẫn – ThS. Vũ Đình Long**, người đã tận tình hướng dẫn, định hướng và đóng góp nhiều ý kiến chuyên môn quý báu trong suốt quá trình thực hiện đề tài “Xây dựng hệ thống nhận diện biển số xe”*.* Những góp ý và sự hỗ trợ của thầy là cơ sở quan trọng giúp nhóm hoàn thiện nội dung học thuật cũng như triển khai hệ thống đúng định hướng của môn học.

Nhóm cũng xin cảm ơn **Trường Đại học Giao thông Vận tải TP. Hồ Chí Minh, Viện Công nghệ Thông tin và Điện, Điện tử** đã tạo điều kiện học tập, cung cấp môi trường và tài liệu cần thiết để nhóm có thể tiếp cận và nghiên cứu các công nghệ trong lĩnh vực Hệ thống Giao thông Thông minh (ITS).

Cuối cùng, nhóm xin cảm ơn các bạn trong nhóm đã hợp tác, hỗ trợ lẫn nhau trong quá trình nghiên cứu, lập trình và hoàn thiện báo cáo. Mặc dù đã có nhiều cố gắng, báo cáo khó tránh khỏi những thiếu sót, nhóm rất mong nhận được sự góp ý của thầy cô để đề tài được hoàn thiện hơn.

# NHẬN XÉT CỦA GIẢNG VIÊN

TP. Hồ Chí Minh, Ngày … Tháng … Năm 20…

Chữ ký giảng viên

# BẢNG PHÂN CÔNG CÔNG VIỆC VÀ MỨC ĐỘ HOÀN THÀNH

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và Tên** | **Nhiệm vụ** | **Phần trăm hoàn thành** |
| **1** | Nguyễn Lê Huy Tâm | Code GenData, Check code, soạn nội dung trên github, soạn nội dung Words phần 5+6. | 100% |
| **2** | Hồ Thiên Phúc | Code video test, soạn nội dung Words phần 1. | 100% |
| **3** | Nguyễn Minh Phú | Code Preprocess, soạn nội dung Words phần 3, soạn báo cáo Words. | 100% |
| **4** | Nguyễn Ngọc Khánh Tân | Kiếm data, soạn nội dung Words phần 2. | 100% |
| **5** | Hồ Lý Minh Phước | Code Image test, soạn nội dung Words phần 4. | 100% |

**CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI**

## **1.1. Bối cảnh thực tế và giải pháp**

## **1.1.1. Bối cảnh thực tế**

Hiện nay, số lượng phương tiện cá nhân (xe máy, ô tô) tại Việt Nam gia tăng nhanh chóng, gây áp lực lớn lên cơ sở hạ tầng giao thông tĩnh, đặc biệt là tại các bãi đỗ xe trường học, chung cư, siêu thị và tòa nhà văn phòng.

Tuy nhiên, quy trình quản lý tại phần lớn các bãi giữ xe hiện nay vẫn còn mang tính thủ công hoặc bán tự động:

* **Phương thức cũ:** Sử dụng vé giấy, ghi phấn lên xe, hoặc nhân viên bảo vệ phải nhập biển số bằng tay vào máy tính.
* **Hệ quả:** Tốn kém nhân lực, thời gian chờ đợi lâu gây ùn tắc cục bộ giờ cao điểm, dễ xảy ra nhầm lẫn, thất thoát doanh thu và khó khăn trong việc tra cứu/tìm kiếm xe khi cần thiết.

## **1.1.2. Giải pháp công nghệ**

Trong bối cảnh đó, công nghệ Nhận diện biển số xe tự động (Automatic License Plate Recognition - ALPR) đóng vai trò then chốt để giải quyết bài toán trên. Thay vì sử dụng các mô hình Deep Learning nặng nề đòi hỏi phần cứng GPU đắt tiền, đồ án này tiếp cận theo hướng Xử lý ảnh (Computer Vision) kết hợp với thuật toán Học máy cơ bản (Machine Learning).

Cụ thể, hệ thống sẽ tận dụng tính cá nhân hóa của biển số xe làm định danh duy nhất, sử dụng thư viện OpenCV để trích xuất đặc trưng hình ảnh và thuật toán KNN (K-Nearest Neighbors) để nhận dạng ký tự. Đây là bước tiếp cận căn bản, phù hợp cho việc học tập và triển khai trên các thiết bị máy tính cá nhân thông thường mà vẫn đảm bảo hiệu quả cho bài toán kiểm soát xe ra vào.

## **1.2. Vấn đề cần giải quyết (Problem Statement)**

Đề tài tập trung giải quyết các vấn đề cụ thể sau để thay thế quy trình thủ công:

1. **Tự động hóa quy trình kiểm soát:** Loại bỏ thao tác ghi chép/nhập liệu thủ công của bảo vệ. Hệ thống phải tự động phát hiện xe, chụp ảnh và đọc biển số. 1
2. **Xử lý đa dạng loại biển số Việt Nam:**

* Hỗ trợ cả biển số 1 dòng (xe ô tô con đời mới) và biển số 2 dòng (xe máy, xe tải).
* Xử lý được các ký tự chữ và số theo chuẩn font biển số Việt Nam.

1. **Tối ưu hóa trên phần cứng phổ thông:** Xây dựng giải thuật nhận diện nhẹ, không yêu cầu GPU rời mạnh mẽ, có thể chạy mượt mà trên laptop sinh viên hoặc máy tính văn phòng cấu hình trung bình (sử dụng CPU).
2. **Giải quyết các thách thức về xử lý ảnh:**

* Tách biển số khỏi nền phức tạp.
* Xử lý ảnh trong điều kiện ánh sáng thay đổi, ảnh bị nghiêng hoặc nhiễu hạt.

## **1.3. Mục tiêu và Yêu cầu tổng quan (Objectives & Requirements)**

## **1.3.1. Mục tiêu kỹ thuật (Measurable Objectives)**

Xây dựng thành công ứng dụng (Application) có khả năng thực hiện quy trình xử lý khép kín sau:

1. **Đầu vào (Input):** Hình ảnh hoặc Video stream từ Camera (Webcam/Camera điện thoại/IP Camera).
2. **Quy trình Xử lý (Processing Pipeline):**

* **Tiền xử lý:** Chuyển đổi ảnh xám (Grayscale), tăng độ tương phản (Top Hat/Black Hat), giảm nhiễu (Gaussian Blur).
* **Phát hiện vị trí (Detection):** Sử dụng thuật toán phát hiện cạnh **Canny** và tìm đường bao **Contour** để khoanh vùng biển số dựa trên tỷ lệ kích thước (Aspect Ratio) và diện tích.
* **Phân đoạn (Segmentation):** Cắt rời từng ký tự, xử lý xoay ảnh (Deskewing) để ký tự thẳng hàng.
* **Nhận diện (Recognition):** Sử dụng mô hình **KNN** đã được huấn luyện để phân loại từng ký tự thành mã ASCII (chữ hoặc số).

3**. Đầu ra (Output):** Chuỗi ký tự biển số (Text) hiển thị trên giao diện và lưu vào cơ sở dữ liệu/file log.

## **1.3.2. Chỉ tiêu đánh giá (KPIs)**

* Tỷ lệ phát hiện (Detection Rate): Hệ thống khoanh vùng đúng vị trí biển số trong khung hình (đạt > 80% trong điều kiện tiêu chuẩn).
* Độ chính xác nhận diện (Recognition Accuracy): Nhận diện đúng các ký tự trên biển số. Mục tiêu giảm thiểu tỷ lệ nhận diện sai các cặp ký tự dễ nhầm (như 8-B, 0-D, 1-7).
* Thời gian xử lý: Đảm bảo tốc độ xử lý chấp nhận được cho bài toán bãi xe (thời gian chờ < 2 giây/xe).

## **1.4. Hạn chế thực tế (Practical Limitations)**

Do đặc thù của phương pháp KNN và Xử lý ảnh truyền thống, cùng với giới hạn về thiết bị của đồ án sinh viên, hệ thống tồn tại một số hạn chế sau:

1. **Hạn chế về Thuật toán (KNN & Contour):**

* **Nhạy cảm với ánh sáng:** Thuật toán Canny và Thresholding hoạt động kém hiệu quả khi bị chói sáng (đèn pha xe chiếu ngược), bóng đổ mạnh hoặc quá tối.
* **Nhiễu nền:** Dễ nhận diện nhầm các chi tiết có hình dáng giống biển số (như biển quảng cáo, decal dán trên xe, lưới tản nhiệt) nếu không lọc kỹ.
* **Độ chính xác:** KNN có độ chính xác thấp hơn so với các mô hình Deep Learning (như CNN, YOLO) khi xử lý các ký tự bị mờ, dính nhau hoặc biến dạng.

1. **Hạn chế về Môi trường & Camera:**

* **Góc chụp:** Yêu cầu camera phải đặt ở góc tương đối chính diện. Nếu biển số bị nghiêng quá nhiều (> 15-20 độ), thuật toán tìm Contour hình chữ nhật và tách ký tự sẽ bị lỗi.
* **Chất lượng ảnh đầu vào:** Ảnh từ webcam hoặc camera điện thoại (như Samsung J7 Prime trong thử nghiệm) có thể bị nhòe (motion blur) nếu xe di chuyển nhanh. Hệ thống yêu cầu xe phải dừng hoặc đi rất chậm.
* **Vật thể che khuất:** Không thể nhận diện nếu biển số bị che khuất một phần (bởi bùn đất, khung biển số, ốc vít che mất nét chữ).

1. **Phạm vi áp dụng:** Hệ thống phù hợp cho mô hình **kiểm soát cổng (Gate Control)** nơi môi trường được kiểm soát tốt, không phù hợp cho bài toán giám sát giao thông tốc độ cao trên đường phố (Free-flow traffic).

# CHƯƠNG 2. CƠ SỞ LÝ THUYẾT & PHÂN TÍCH YÊU CẦU

## **2.1. Cơ sở lý thuyết**

## **2.1.1. Computer Vision trong bài toán nhận dạng biển số xe**

Computer Vision (thị giác máy tính) là lĩnh vực nghiên cứu giúp máy tính có khả năng “nhìn” và hiểu nội dung từ hình ảnh hoặc video. Trong đề tài xây dựng hệ thống nhận dạng biển số xe, Computer Vision đóng vai trò cốt lõi, đặc biệt ở các khâu phát hiện và trích xuất thông tin từ hình ảnh camera.

Hiện nay, các mô hình học sâu (Deep Learning) được sử dụng rộng rãi trong bài toán nhận dạng biển số, tiêu biểu là các mô hình thuộc họ YOLO (You Only Look Once). YOLO cho phép phát hiện đối tượng trong ảnh theo thời gian thực với độ trễ thấp, phù hợp cho các hệ thống giám sát giao thông. Mô hình này thường được sử dụng để xác định vị trí biển số xe trong khung hình.

Bên cạnh detection, kỹ thuật segmentation có thể được áp dụng để tách chính xác vùng biển số khỏi nền ảnh, giúp cải thiện chất lượng dữ liệu đầu vào cho bước nhận dạng ký tự. Ngoài ra, trong các hệ thống camera video, kỹ thuật tracking (theo dõi đối tượng) giúp theo dõi một phương tiện xuyên suốt nhiều khung hình, tránh việc nhận dạng trùng lặp hoặc bỏ sót.

Sau khi biển số được phát hiện, hệ thống sẽ sử dụng các mô hình nhận dạng ký tự quang học (OCR) để chuyển hình ảnh biển số thành chuỗi ký tự tương ứng.

## **2.1.2. Hệ thống cảm biến (Sensor Systems)**

Trong hệ thống nhận dạng biển số xe, camera là cảm biến chính được sử dụng để thu thập dữ liệu hình ảnh. Camera có thể được lắp đặt tại cổng ra vào, bãi giữ xe, trạm thu phí hoặc các nút giao thông. Chất lượng camera (độ phân giải, tốc độ khung hình, khả năng hoạt động trong điều kiện thiếu sáng) ảnh hưởng trực tiếp đến độ chính xác của hệ thống.

Ngoài camera, trong các hệ thống giao thông thông minh quy mô lớn, radar hoặc LiDAR có thể được sử dụng để hỗ trợ phát hiện phương tiện, đo khoảng cách hoặc tốc độ. Tuy nhiên, trong phạm vi đề tài này, nhóm tập trung chủ yếu vào dữ liệu hình ảnh từ camera do tính phổ biến và chi phí triển khai thấp hơn.

## **2.1.3. Hệ thống truyền thông và giao tiếp dữ liệu**

Sau khi biển số xe được nhận dạng, dữ liệu cần được truyền đến hệ thống xử lý trung tâm hoặc cơ sở dữ liệu. Các giao thức truyền thông nhẹ như MQTT thường được sử dụng trong các hệ thống IoT nhờ ưu điểm về độ trễ thấp và khả năng hoạt động ổn định trong môi trường mạng không đồng đều.

Ngoài ra, trong các hệ thống giao thông thông minh nâng cao, công nghệ V2X (Vehicle-to-Everything) cho phép phương tiện giao tiếp với hạ tầng giao thông và các phương tiện khác. Mặc dù V2X không phải là trọng tâm của đề tài, nhưng đây là hướng phát triển tiềm năng khi hệ thống nhận dạng biển số được tích hợp vào hạ tầng giao thông thông minh.

## **2.1.4. Mô hình phát hiện và lý thuyết dòng giao thông (tham khảo)**

Trong một số ứng dụng mở rộng, dữ liệu biển số xe có thể được dùng để phân tích lưu lượng giao thông, thống kê tần suất di chuyển hoặc phát hiện các hành vi bất thường. Khi đó, các mô hình phân tích dòng giao thông (traffic flow theory) có thể được áp dụng để đánh giá mật độ xe, thời gian lưu thông hoặc xu hướng ùn tắc. Tuy nhiên, trong phạm vi đề tài này, phần lý thuyết này chỉ mang tính tham khảo.

## **2.2. Phân tích yêu cầu hệ thống**

## **2.2.1. Yêu cầu chức năng (Functional Requirements – FR)**

Hệ thống nhận dạng biển số xe cần đáp ứng các chức năng chính sau:

* FR1: Thu thập hình ảnh hoặc video từ camera giám sát.
* FR2: Phát hiện phương tiện và vị trí biển số xe trong khung hình.
* FR3: Trích xuất vùng biển số và nhận dạng ký tự biển số.
* FR4: Phân loại và lưu trữ thông tin biển số vào cơ sở dữ liệu.
* FR5: Đếm số lượng phương tiện dựa trên biển số nhận dạng được.
* FR6: Cảnh báo hoặc thông báo khi phát hiện biển số nằm trong danh sách đặc biệt (xe bị cấm, xe vi phạm, xe trong danh sách theo dõi).
* FR7: Hiển thị kết quả nhận dạng cho người dùng thông qua giao diện hệ thống.

## **2.2.2. Yêu cầu phi chức năng (Non-Functional Requirements – NFR)**

Bên cạnh các chức năng chính, hệ thống cần đảm bảo các yêu cầu phi chức năng sau:

* NFR1: Độ chính xác cao trong nhận dạng biển số, đặc biệt trong điều kiện ánh sáng thay đổi
* NFR2: Độ trễ thấp, đảm bảo hệ thống có thể hoạt động gần thời gian thực.
* NFR3: Tính ổn định và tin cậy khi vận hành liên tục trong thời gian dài.
* NFR4: Khả năng mở rộng, cho phép tích hợp thêm camera hoặc mở rộng phạm vi giám sát.
* NFR5: Bảo mật dữ liệu, đảm bảo thông tin biển số và người dùng không bị truy cập trái phép.

## **2.2.3. Đối tượng sử dụng và kịch bản sử dụng (Actors & User Scenarios)**

Các đối tượng chính tương tác với hệ thống bao gồm:

* Người lái xe: phương tiện của người dùng được hệ thống tự động nhận dạng khi đi qua khu vực giám sát.
* Trung tâm quản lý giao thông (Traffic Management Center – TMC): giám sát, theo dõi và phân tích dữ liệu biển số xe.
* Nhân viên vận hành hệ thống: quản lý camera, kiểm tra dữ liệu và xử lý các tình huống cảnh báo.
* Người đi bộ (gián tiếp): được hưởng lợi từ việc quản lý giao thông hiệu quả và an toàn hơn.

**Kịch bản sử dụng tiêu biểu**:  
Khi một phương tiện đi vào khu vực giám sát, camera ghi nhận hình ảnh. Hệ thống tự động phát hiện biển số, nhận dạng ký tự và lưu thông tin vào cơ sở dữ liệu. Nếu biển số thuộc danh sách cảnh báo, hệ thống sẽ gửi thông báo đến trung tâm quản lý để xử lý kịp thời.

**CHƯƠNG 3. THIẾT KẾ HỆ THỐNG**

**3.1. Tổng quan kiến trúc hệ thống (System Architecture)**

Hệ thống nhận diện biển số xe được thiết kế theo mô hình **pipeline xử lý tuần tự**, bao gồm ba khối chính: **Đầu vào (Input) → Xử lý (Processing) → Đầu ra (Output)**.  
Kiến trúc này phù hợp với các hệ thống ITS (Intelligent Transportation Systems) yêu cầu xử lý dữ liệu thị giác theo thời gian thực.

Trong phạm vi đề tài, hệ thống tập trung vào xử lý ảnh truyền thống kết hợp học máy cơ bản, không sử dụng các mô hình học sâu phức tạp, nhằm đảm bảo khả năng triển khai trên phần cứng phổ thông.

**Pipeline tổng quát:**

Camera / Video/ Ảnh → Preprocessing→ License Plate Detection→ Plate Alignment → Character Segmentation→ Character Recognition (KNN) → Display / Log

**3.2. Kiến trúc CV Pipeline**

Hệ thống sử dụng Hệ thống được xây dựng dựa trên Computer Vision Pipeline truyền thống, gồm các bước xử lý chính sau:

**3.2.1. Input**

Nguồn dữ liệu: Ảnh tĩnh, Video giám sát, Video offline phục vụ huấn luyện và kiểm thử  
Định dạng: [.jpg], [.png], [.mp4], [.avi],…

**3.2.2. Preprocessing (Tiền xử lý ảnh)**

Mục tiêu của bước tiền xử lý là nâng cao chất lượng ảnh, giảm nhiễu và làm nổi bật vùng biển số.

Các kỹ thuật được sử dụng:

* Chuyển đổi sang ảnh xám (Grayscale)
* Tăng độ tương phản bằng Top Hat và Black Hat
* Gaussian Blur để giảm nhiễu
* Adaptive Threshold để nhị phân hóa ảnh
* Canny Edge Detection để phát hiện biên
* Morphological Closing để kết nối các cạnh

**3.2.3. License Plate Detection (Phát hiện biển số)**

Hệ thống phát hiện biển số dựa trên phân tích contour trong ảnh đã tiền xử lý.

Tiêu chí lựa chọn contour biển số:

* Diện tích nằm trong ngưỡng cho phép
* Tỷ lệ chiều rộng / chiều cao phù hợp với biển số Việt Nam
* Contour có hình dạng gần chữ nhật (4 đỉnh)

Contour thỏa mãn tốt nhất sẽ được chọn làm vùng biển số và cắt ra từ ảnh gốc.

**3.2.4. Plate Alignment (Căn chỉnh biển số)**

Do ảnh hưởng của góc đặt camera, biển số có thể bị nghiêng. Hệ thống tiến hành tính toán góc nghiêng dựa trên các điểm biên của biển số và xoay ảnh để đưa biển số về phương ngang.

*Mục đích:*

* Tăng độ chính xác khi phân tách ký tự
* Giảm lỗi nhận dạng

**3.2.5. Character Segmentation (Phân tách ký tự)**

Sau khi căn chỉnh, biển số được xử lý để tách từng ký tự riêng lẻ.

Các bước chính:

* Nhị phân hóa vùng biển số
* Tìm contour của các ký tự
* Lọc contour theo tỷ lệ chiều cao/chiều rộng và diện tích
* Sắp xếp ký tự theo thứ tự từ trái sang phải

Output: tập các ảnh ký tự riêng lẻ.

**3.2.6. Character Recognition (Nhận dạng ký tự)**

Hệ thống sử dụng thuật toán **K-Nearest Neighbors (KNN)** để nhận dạng ký tự.

Quy trình:

* Chuẩn hóa ảnh ký tự về kích thước 20×30 pixels
* Chuyển ảnh thành vector 1 chiều (600 phần tử)
* So sánh với dữ liệu huấn luyện bằng khoảng cách Euclidean
* Chọn K = 3 láng giềng gần nhất
* Xác định nhãn ký tự bằng voting

**3.3. Data Flow Diagram (DFD)**

**3.3.1. Mô tả luồng dữ liệu**

1. Camera / Image / Video cung cấp dữ liệu đầu vào
2. Dữ liệu được tiền xử lý
3. Phát hiện và trích xuất vùng biển số
4. Căn chỉnh biển số
5. Phân tách ký tự
6. Nhận dạng ký tự bằng KNN
7. Hiển thị kết quả và lưu log

**3.3.2. DFD mức khái quát**

[Input Image / Video]

↓

[Preprocessing]

↓

[Plate Detection]

↓

[Plate Alignment]

↓

[Character Segmentation]

↓

[Character Recognition (KNN)]

↓

[Display / Log]

**3.4. Sơ đồ Component / Module**

Hệ thống được thiết kế theo kiến trúc module hóa:

1. Input Module: Đọc ảnh/video
2. Preprocessing Module (Preprocess.py)
3. Plate Detection Module
4. Alignment Module
5. Character Segmentation Module
6. KNN Recognition Module
7. Output Module (hiển thị và lưu kết quả)

**3.5. Kiến trúc Edge–Cloud (Edge–Cloud Architecture)**

Hệ thống áp dụng mô hình **Edge–Cloud** phổ biến trong ITS:

**3.5.1. Luật quyết định (Decision Logic)**

Hệ thống sử dụng các luật IF–THEN đơn giản:

* IF không tìm thấy contour thỏa mãn → bỏ qua frame
* IF số ký tự hợp lệ < ngưỡng → không hiển thị kết quả
* IF ký tự nhận dạng không thuộc [0–9, A–Z] → loại bỏ

**3.5.2. Thuật toán sử dụng**

| **Thành phần** | **Thuật toán** |
| --- | --- |
| Tiền xử lý | Grayscale, Threshold, Canny |
| Phát hiện biển số | Contour-based Detection |
| Căn chỉnh | Geometric Rotation |
| Phân tách ký tự | Contour Filtering |
| Nhận dạng | KNN (K=3) |
| Logic | Rule-based |

Bảng 2: Các thuật toán sử dụng trong các module dự án

# CHƯƠNG 4: HIỆN THỰC VÀ TRIỂN KHAI

## **4.1. Công cụ và Công nghệ sử dụng**

Hệ thống nhận dạng biển số xe Việt Nam được xây dựng dựa trên các công nghệ và thư viện hiện đại, đảm bảo hiệu suất xử lý và độ chính xác cao. Dưới đây là chi tiết các công cụ được sử dụng trong dự án.

## **4.1.1. Ngôn ngữ lập trình**

**Python 3.x:** Được chọn làm ngôn ngữ chính cho dự án nhờ vào tính linh hoạt, cộng đồng lớn và hệ sinh thái thư viện phong phú trong lĩnh vực xử lý ảnh và machine learning. Python cho phép phát triển nhanh chóng và dễ dàng tích hợp các thư viện khác nhau.

## **4.1.2. Thư viện xử lý ảnh và Computer Vision**

**OpenCV (Open Source Computer Vision Library):** Thư viện xử lý ảnh mạnh mẽ và phổ biến nhất hiện nay, cung cấp hơn 2500 thuật toán tối ưu cho các tác vụ computer vision. Trong dự án, OpenCV được sử dụng cho:

* Đọc và xử lý ảnh/video từ nhiều nguồn khác nhau
* Chuyển đổi không gian màu (RGB, Grayscale, HSV)
* Các phép biến đổi hình thái học (morphological operations): erosion, dilation, opening, closing
* Phát hiện cạnh với thuật toán Canny Edge Detection
* Tìm và vẽ contours để xác định vùng biển số
* Các phép biến đổi hình học: xoay, co giãn, căn chỉnh ảnh
* Áp dụng các bộ lọc: Gaussian blur, bilateral filter để giảm nhiễu

**NumPy:** Thư viện tính toán khoa học, xử lý mảng đa chiều hiệu quả. Được sử dụng để:

* Thao tác với mảng pixel của ảnh
* Tính toán các phép biến đổi toán học trên ma trận ảnh
* Xử lý và chuẩn hóa dữ liệu đầu vào cho mô hình KNN

## **4.1.3. Thuật toán Machine Learning**

**K-Nearest Neighbors (KNN):** Thuật toán học có giám sát đơn giản nhưng hiệu quả cho bài toán phân loại ký tự. KNN hoạt động dựa trên nguyên lý "những điểm gần nhau có xu hướng thuộc cùng một lớp".

**Ưu điểm của KNN trong dự án:**

* Đơn giản, dễ hiểu và dễ triển khai
* Không cần quá trình training phức tạp
* Phù hợp với dataset nhỏ và trung bình
* Hoạt động tốt với dữ liệu có ranh giới rõ ràng
* Có thể cập nhật model dễ dàng khi thêm dữ liệu mới

**Tham số sử dụng:**

* K = 3: Số lượng láng giềng gần nhất được xem xét
* Distance metric: Euclidean distance
* Kích thước ảnh ký tự chuẩn hóa: 20×30 pixels

## **4.1.4. Các thư viện hỗ trợ khác**

**imutils:** Cung cấp các hàm tiện ích để xử lý ảnh như resize, rotate, translate một cách đơn giản hơn.

**matplotlib:** Hiển thị và trực quan hóa kết quả xử lý ảnh, vẽ biểu đồ phân tích.

**scikit-image:** Cung cấp thêm các thuật toán xử lý ảnh nâng cao.

## **4.1.5. Môi trường phát triển**

**IDE:** Visual Studio Code, PyCharm.

**Version Control:** Git, GitHub ([Sleepy2608/ITS\_Doc-bien-so-xe: Final-term assignment for the Intelligent Transportation Systems (ITS) course focusing on the development of a vehicle license plate recognition application.](https://github.com/Sleepy2608/ITS_Doc-bien-so-xe))

**Hệ điều hành:** Windows 10/11, Linux (Ubuntu).

**Python Version:** Python 3.8 trở lên.

## **4.2. Mô tả các Module chính của Hệ thống**

Hệ thống được thiết kế theo kiến trúc modular, chia thành các module độc lập với chức năng rõ ràng, dễ dàng bảo trì và mở rộng. Dưới đây là mô tả chi tiết từng module:

## **4.2.1. Module Tiền xử lý ảnh (Preprocess.py)**

Module này chứa tất cả các hàm xử lý ảnh cơ bản, đóng vai trò quan trọng trong việc chuẩn bị dữ liệu đầu vào cho các bước tiếp theo.

**Các hàm chính:**

**1. extract\_value() - Trích xuất kênh V từ HSV:**

Chuyển đổi ảnh từ không gian màu RGB sang HSV và trích xuất kênh Value (độ sáng). Kênh V giúp tách biệt thông tin độ sáng với màu sắc, hữu ích trong điều kiện ánh sáng thay đổi.

**2. maximize\_contrast() - Tăng độ tương phản:**

Sử dụng các phép toán hình thái học Top Hat và Black Hat để tăng cường độ tương phản giữa biển số và môi trường xung quanh. Điều này làm nổi bật vùng biển số, giúp dễ dàng phát hiện trong các bước sau.

* Top Hat = Original – Opening
* Black Hat = Closing – Original
* Result = Gray + Top Hat - Black Hat

**3. preprocess() - Xử lý tổng hợp:**

Kết hợp nhiều kỹ thuật xử lý ảnh:

1. Chuyển sang ảnh xám (Grayscale)

2. Tăng độ tương phản với maximize\_contrast()

3. Làm mờ Gaussian để giảm nhiễu

4. Ngưỡng hóa thích ứng (Adaptive Threshold) để nhị phân hóa ảnh

5. Phát hiện cạnh Canny

6. Đóng hình thái học để kết nối các cạnh gần nhau

### **4.2.2. Module Sinh dữ liệu huấn luyện (GenData.py)**

Module này cho phép người dùng tạo dữ liệu huấn luyện cho thuật toán KNN bằng cách gán nhãn thủ công cho các ký tự.

***Quy trình hoạt động:***

1. Đọc ảnh training\_chars.png chứa các ký tự mẫu

2. Phân tách từng ký tự dựa trên contour

3. Hiển thị từng ký tự và yêu cầu người dùng nhập nhãn

4. Chuẩn hóa ảnh ký tự về kích thước 20×30 pixels

5. Chuyển đổi ma trận ảnh thành vector 1D (600 phần tử)

6. Lưu mã ASCII của ký tự vào classifications.txt

7. Lưu vector đặc trưng vào flattened\_images.txt

**Output Files:**

* classifications.txt: Chứa mã ASCII của các ký tự (36 ký tự: 0-9, A-Z)
* flattened\_images.txt: Chứa vector đặc trưng tương ứng (mỗi hàng 600 giá trị)

### **4.2.3. Module Nhận dạng từ ảnh tĩnh (Image\_test2.py)**

Module chính xử lý ảnh tĩnh, thực hiện toàn bộ pipeline từ phát hiện biển số đến nhận dạng ký tự.

**Pipeline xử lý:**

**Bước 1: Đọc và tiền xử lý ảnh đầu vào**

Ảnh được đọc từ thư mục data/image/ và qua các bước tiền xử lý cơ bản.

**Bước 2: Phát hiện biển số (License Plate Detection)**

Quy trình phát hiện:

* Áp dụng hàm preprocess() để có ảnh nhị phân với các cạnh rõ nét
* Tìm tất cả contours trong ảnh
* Lọc contours dựa trên các tiêu chí:
* Diện tích: 1000 < area < 50000 pixels
* Tỷ lệ chiều rộng/chiều cao: 1.2 < ratio < 6.0
* Số đỉnh của contour xấp xỉ: 4 (hình chữ nhật)
* Chọn contour có diện tích lớn nhất thỏa mãn điều kiện
* Cắt vùng biển số từ ảnh gốc

**Bước 3: Căn chỉnh biển số (Plate Alignment)**

Biển số thường bị nghiêng do góc chụp. Module tính toán góc nghiêng và xoay lại:

# Tính góc nghiêng từ 2 đỉnh dưới  
 dy = abs(y2 - y1)  
 dx = abs(x2 - x1)  
 angle = math.degrees(math.atan(dy/dx))  
 # Xoay ảnh về phương ngang  
 if y1 < y2:  
 angle = -angle  
 rotated = imutils.rotate\_bound(plate, angle)

**Bước 4: Phân tách ký tự (Character Segmentation)**

Các bước phân tách:

* Chuyển biển số sang ảnh xám và nhị phân hóa
* Tìm contours của các ký tự (vùng màu trắng)
* Lọc contours dựa trên:
* Tỷ lệ chiều cao/chiều rộng: 1.2 < ratio < 6.0
* Diện tích tối thiểu để loại nhiễu
* Sắp xếp các ký tự theo thứ tự từ trái sang phải
* Vẽ bounding box xung quanh mỗi ký tự

**Bước 5: Nhận dạng ký tự với KNN**

Quy trình nhận dạng mỗi ký tự:

* Chuẩn hóa ảnh ký tự về kích thước 20×30 pixels
* Chuyển đổi thành vector 1D (flatten)
* So sánh với tất cả mẫu trong flattened\_images.txt bằng Euclidean distance
* Chọn K=3 mẫu gần nhất
* Xác định nhãn dựa trên voting (đa số)
* Trả về ký tự tương ứng với mã ASCII

**Bước 6: Tổng hợp và hiển thị kết quả**

Ghép các ký tự đã nhận dạng thành chuỗi biển số hoàn chỉnh. Hiển thị ảnh gốc với biển số được đánh dấu và chuỗi kết quả.

### **4.2.4. Module Nhận dạng từ video (Video\_test2.py)**

Module này mở rộng khả năng xử lý từ ảnh tĩnh sang video thời gian thực, xử lý từng frame và tối ưu hiệu năng.

**Đặc điểm:**

* Đọc video từ file hoặc camera (độ phân giải khuyến nghị: 1920×1080, 24fps)
* Xử lý từng frame như một ảnh độc lập
* Tối ưu hóa tốc độ xử lý để đạt real-time
* Hiển thị kết quả trực tiếp trên video
* Có thể lưu video kết quả
* Xử lý trường hợp không phát hiện được biển số (skip frame)

cap = cv2.VideoCapture('video.mp4')  
 while cap.isOpened():  
 ret, frame = cap.read()  
 if not ret:  
     break  
 # Xử lý frame (tương tự Image\_test2.py)  
 result\_frame = process\_frame(frame)  
   
 # Hiển thị kết quả  
 cv2.imshow('License Plate Detection', result\_frame)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
 cap.release()

## **4.3. Sơ đồ luồng thực thi (Flowchart)**

Dưới đây là sơ đồ luồng chi tiết mô tả quá trình xử lý từ đầu vào đến kết quả cuối cùng của hệ thống:

**4.3.1. Luồng chính của hệ thống:**

START → Nhận đầu vào (ảnh/video)

Đọc và load ảnh vào bộ nhớ

Tiền xử lý ảnh (Preprocess module)

├─ Chuyển sang grayscale

├─ Tăng tương phản

├─ Gaussian blur

├─ Adaptive threshold

└─ Canny edge detection

Tìm contours trong ảnh

Lọc contours theo tiêu chí (diện tích, tỷ lệ, hình dạng)

Có tìm thấy biển số? [Decision Point]

├─ Không → Hiển thị "No plate detected" → END

└─ Có → Tiếp tục

Trích xuất vùng biển số

Tính góc nghiêng và xoay biển số

Phân tách ký tự:

├─ Chuyển sang binary

├─ Tìm contours của ký tự

├─ Lọc theo tỷ lệ H/W

└─ Sắp xếp từ trái sang phải

Load KNN model (classifications.txt, flattened\_images.txt)

FOR mỗi ký tự:

├─ Chuẩn hóa về 20×30 pixels

├─ Flatten thành vector 1D

├─ Tính khoảng cách với tất cả mẫu

├─ Chọn K=3 láng giềng gần nhất

├─ Voting để xác định nhãn

└─ Thêm ký tự vào chuỗi kết quả

Ghép chuỗi ký tự thành biển số hoàn chỉnh

Vẽ bounding box và text lên ảnh gốc

Hiển thị kết quả

END

**4.3.2. Các điểm quyết định (Decision Points):**

|  |  |  |
| --- | --- | --- |
| **Decision Point** | **Điều kiện** | **Hành động** |
| **Phát hiện biển số** | area > 1000 AND  ratio between 1.2-6.0 AND  approx = 4 vertices | Tiếp tục xử lý hoặc bỏ qua |
| **Lọc ký tự** | ratio between 1.2-6.0 AND  area > threshold | Chấp nhận hoặc loại bỏ |
| **Xác định ký tự** | distance to K nearest  neighbors | Gán nhãn ký tự |

Bảng 3: Các điểm quyết định

## **4.3.3. Tối ưu hóa hiệu năng:**

* Xử lý ảnh ở độ phân giải phù hợp (không quá lớn)
* Sử dụng Region of Interest (ROI) để giảm vùng xử lý
* Cache KNN model data để không phải load lại mỗi frame
* Skip frames nếu không phát hiện được biển số
* Sử dụng numpy vectorization thay vì loops

## **4.4. Ánh xạ từ Thiết kế sang Code (Design to Code Mapping)**

Mô tả cách các thành phần thiết kế trong Chương 3 được hiện thực thành code cụ thể:

|  |  |  |
| --- | --- | --- |
| **Thành phần Thiết kế** | **Module/File Code** | **Functions/Classes** |
| **Module Tiền xử lý** | Preprocess.py | extract\_value()  maximize\_contrast()  preprocess() |
| **Module Phát hiện biển số** | Image\_test2.py  Video\_test2.py | detect\_plate()  find\_contours()  filter\_contours() |
| **Module Căn chỉnh** | Image\_test2.py | calculate\_angle()  rotate\_plate() |
| **Module Phân tách ký tự** | Image\_test2.py  Video\_test2.py | segment\_characters()  find\_char\_contours()  sort\_contours() |
| **Module Nhận dạng KNN** | GenData.py  Image\_test2.py | train\_knn()  predict\_char()  load\_knn\_data() |
| **Module Quản lý I/O** | Image\_test2.py  Video\_test2.py | read\_image()  read\_video()  save\_result() |
| **Module Hiển thị** | Image\_test2.py  Video\_test2.py | draw\_results()  show\_image()  display\_video() |

Bảng 4: Mô tả các thành phần thiết kế của dự án

**4.4.1. Cấu trúc thư mục dự án:**

ITS\_Doc-bien-so-xe/  
 ├── data/  
 │   ├── image/          # Thư mục chứa ảnh test  
 │   └── video/          # Thư mục chứa video test  
 ├── result/             # Thư mục lưu kết quả  
 ├── Preprocess.py       # Module tiền xử lý  
 ├── GenData.py          # Module sinh training data  
 ├── Image\_test2.py      # Main: xử lý ảnh  
 ├── Video\_test2.py      # Main: xử lý video  
 ├── training\_chars.png  # Ảnh chứa ký tự mẫu  
 ├── classifications.txt # File lưu nhãn KNN  
 ├── flattened\_images.txt # File lưu features KNN  
 └── README.md           # Hướng dẫn sử dụng

**4.4.2. Code snippets quan trọng:**

**Snippet 1: Hàm tiền xử lý chính (Preprocess.py)**

def preprocess(img\_original):  
 img\_gray = cv2.cvtColor(img\_original, cv2.COLOR\_BGR2GRAY)  
 # Tăng tương phản  
 img\_contrast = maximize\_contrast(img\_gray)  
 # Giảm nhiễu  
 img\_blur = cv2.GaussianBlur(img\_contrast, (5, 5), 0)  
 # Nhị phân hóa thích ứng  
 img\_thresh = cv2.adaptiveThreshold(  
     img\_blur, 255,  
     cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,  
     cv2.THRESH\_BINARY\_INV, 19, 9  
 )  
 # Phát hiện cạnh  
 img\_edges = cv2.Canny(img\_thresh, 100, 200)  
   
 # Morphological closing  
 kernel = cv2.getStructuringElement(cv2.MORPH\_RECT, (3, 3))  
 img\_closed = cv2.morphologyEx(img\_edges, cv2.MORPH\_CLOSE, kernel)  
 return img\_closed

**Snippet 2: Phát hiện biển số (Image\_test2.py)**

def detect\_plate(img):  
 img\_processed = preprocess(img)  
 # Tìm contours  
 contours, \_ = cv2.findContours(  
     img\_processed,  
     cv2.RETR\_EXTERNAL,  
     cv2.CHAIN\_APPROX\_SIMPLE  
 )  
 for contour in contours:  
     area = cv2.contourArea(contour)  
     # Lọc theo diện tích  
     if 1000 < area < 50000:  
         # Tính tỷ lệ  
         x, y, w, h = cv2.boundingRect(contour)  
         ratio = w / float(h)  
 # Kiểm tra tỷ lệ và số đỉnh  
         if 1.2 < ratio < 6.0:  
             approx = cv2.approxPolyDP(  
                 contour, 0.02 \* cv2.arcLength(contour, True), True  
             )  
             if len(approx) == 4:  
                 # Tìm thấy biển số  
                 plate = img[y:y+h, x:x+w]  
                 return plate, (x, y, w, h)  
 return None, None

**Snippet 3: Nhận dạng ký tự với KNN (Image\_test2.py)**

def recognize\_char(char\_img, knn\_model):  
# Chuẩn hóa kích thước  
 char\_resized = cv2.resize(char\_img, (20, 30))  
# Flatten thành vector 1D  
 char\_flattened = char\_resized.reshape(1, 600).astype(np.float32)  
   
# Dự đoán với KNN  
 ret, result, neighbors, dist = knn\_model.findNearest(  
     char\_flattened, k=3  
 )  
# Chuyển đổi mã ASCII thành ký tự  
 char\_predicted = chr(int(result[0][0]))  
 return char\_predicted  
 # Load KNN model  
 def load\_knn\_model():  
 npa\_classifications = np.loadtxt(  
     "classifications.txt", np.float32  
 )  
 npa\_flattened\_images = np.loadtxt(  
     "flattened\_images.txt", np.float32  
 )  
 knn = cv2.ml.KNearest\_create()  
 knn.train(npa\_flattened\_images, cv2.ml.ROW\_SAMPLE, npa\_classifications)  
 return knn

# CHƯƠNG 5. KIỂM THỬ – ĐÁNH GIÁ

1. **Test cases (nhiều bối cảnh: sáng/tối, đông/vắng, occlusion…)**

Ảnh chụp ban ngày (ở trong tệp result).

Biển số rõ.

Chụp ở góc nghiêng và thẳng.

Khi quay video có chuyển động nhẹ.

1. **Metrics (accuracy, FPS, latency…)**

Accuracy phát hiện biển số.

Tỉ lệ nhận dạng đúng 100% ký tự.

Thời gian xử lý video offline.

1. **Hạn chế**

Khả năng nhận diện của KNN còn thấp, khi tập dữ liệu quá nhiều sẽ tăng thời gian xử lý vì phải quét hết tập dữ liệu train.

Nhận diện kém với sự phản chiếu của biển số, sự di ảnh, chói sáng từ môi trường ngoài, những biển có phần chữ số không rõ ràng, với biển số xe ô tô (có thể gây nhầm lẫn ký tự có hình dạng tương tự (1–7, 0–6, B–8).

1. **Điều kiện triển khai thực tế**

Ở trên đường đi, đặt camera ở các vị trí như: Biển báo giao thông, đèn giao thông, cây cối lên đường, ...

Camera cố định, ánh sáng ổn định.

1. **Tiềm năng cải tiến**

Cần thay đổi thuật toán nhận diện KNN sang những thuật toán khác tinh vi và phức tạp hơn như CNN, SVM hoặc có thể sử dụng những bộ thư viện đã có sẵn trên thế giới như YOLO, YOLOv3...

Sử dụng camera chuyên dụng cho việc nhận diện biển số xe vì có khả năng chống chịu với sương mù, đêm tối, chói sáng...

Sử dụng các thuật toán xử lý ảnh khác để xác định vị trí biển số tốt hơn như phương pháp biến đổi Hough để nhận diện đường thẳng, xác định bằng màu sắc, những thuật toán làm hạn chế sự di ảnh khi xe đang di chuyển.

1. **Kết quả**



Hình 1 – Nhận diện bảng số xe 1



Hình 2 – Nhận diện bảng số xe 2



Hình 3 - Bảng số xe mẫu 1



Hình 4 - Kết quả 1



Hình 5 - Bảng số xe hơi mẫu 1



Hình 6 - Kết quả bảng số xe hơi mẫu 1



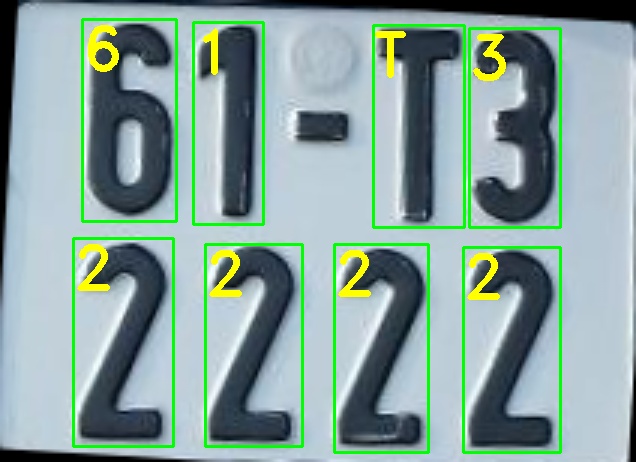
Hình 7 - Bảng số xe mẫu 2



Hình 8 - Nhận diện bảng số xe mẫu 2.1



Hình 9 - Nhận diện bảng số xe mẫu 2.2



Hình 10 - Kết quả bảng số xe mẫu 2.1



Hình 11 - Kết quả bảng số xe mẫu 2.2



Hình 12 - Ảnh chụp trắng đen để check kết quả mẫu xe 2



Hình 13 - Ảnh phóng to kết quả mẫu xe 2



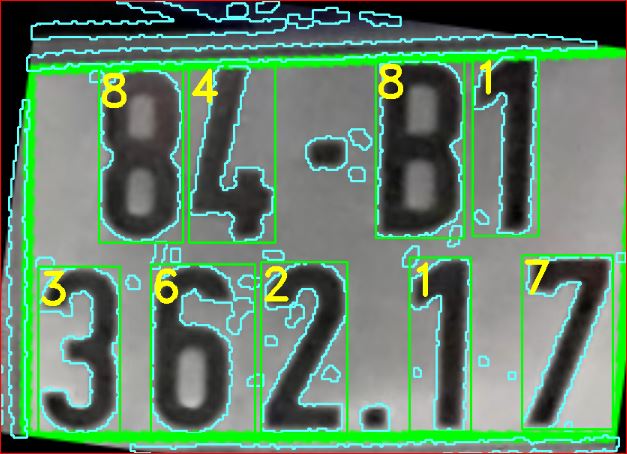
Hình 14 - Bảng số xe hơi mẫu 2



Hình 15 - Kết quả bảng số xe hơi mẫu 2



Hình 16 - Nhận diện bảng số xe mẫu 3



Hình 17 - Kết quả bảng số xe mẫu 3.1



Hình 18 - Kết quả bảng số xe mẫu 3.2

# CHƯƠNG 6. KẾT LUẬN & HƯỚNG PHÁT TRIỂN

1. **Summary essence của hệ thống**

Hệ thống hiện thực thành công bài toán nhận dạng biển số xe Việt Nam dựa trên xử lý ảnh và học máy cơ bản.

1. **Ý nghĩa đối với ITS**

Tự động hóa quản lý bãi đỗ xe.

Nền tảng cho hệ thống giao thông thông minh.

1. **Hướng mở rộng:**

scaling, edge computing, integration, dataset, deployment

Scaling dữ liệu và mô hình.

Tích hợp edge–cloud.

Mở rộng triển khai thực tế.

# TÀI LIỆU THAM KHẢO

[1] S. Du, M. Ibrahim, M. Shehata, W. Badawy, Automatic License Plate Recognition (ALPR): A State-of-the-Art Review, IEEE Transactions on Circuits and Systems for Video Technology, 2013.

Link: [Automatic License Plate Recognition (ALPR): A State-of-the-Art Review | IEEE Journals & Magazine | IEEE Xplore](https://ieeexplore.ieee.org/document/6213519)

[2] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, 4th Edition, Pearson Education, 2018.

Link: [Digital Image Processing, 4e](https://api.pageplace.de/preview/DT0400.9781292223070_A37747583/preview-9781292223070_A37747583.pdf)

[3] Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

Link: [Bishop\_book.pdf](https://www.cs.uoi.gr/~arly/courses/ml/tmp/Bishop_book.pdf)

[4] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, 3rd Edition, Pearson, 2016.

Link: [Artificial Intelligence: A Modern Approach, Global Edition, 4ed](https://api.pageplace.de/preview/DT0400.9781292401171_A41586057/preview-9781292401171_A41586057.pdf)

[5] Bradski, G., The OpenCV Library, Dr. Dobb’s Journal of Software Tools, 2000.

Link: [Learning OpenCV---Computer Vision with the OpenCV Library (Bradski, G.R. et al.; 2008)[On the Shelf] | IEEE Journals & Magazine | IEEE Xplore](https://ieeexplore.ieee.org/document/5233425)

[6] Ministry of Transport Vietnam, Intelligent Transportation Systems Overview, (tài liệu tham khảo ITS).

Link: [Intelligent Transport System in Viet Nam](https://www.unescap.org/sites/default/d8files/event-documents/Viet%20Nam_0.pdf)