|  |  |
| --- | --- |
|  | BỘ GIÁO DỤC VÀ ĐÀO TẠO  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH** |
| **BÁO CÁO MÔN HỌC**  **XỬ LÍ ẢNH SỐ**  **SO SÁNH CÁC THUẬT TOÁN**  **PHÁT HIỆN CẠNH ( EDGE DETECTION )**  Ngành: Robot và trí tuệ nhân tạo  Lớp: 22DRTA1  **GIẢNG VIÊN : TS. PHẠM QUỐC THIỆN**   |  |  |  | | --- | --- | --- | | **Sinh viên thực hiện:** | **MSSV:** | **Lớp:** | | Nguyễn Văn Đạt | 2286300010 | 22DRTA1 | | Huỳnh Long | 2286300028 | 22DRTA1 | | Nguyễn Chấn Huy | 2286300020 | 22DRTA1 |   *TP. Hồ Chí Minh, ngày 9 tháng 1 năm 2024* | |
|  | BỘ GIÁO DỤC VÀ ĐÀO TẠO  **TRƯỜNG ĐẠI HỌC CÔNG NGHỆ TP. HỒ CHÍ MINH** |
| **BÁO CÁO MÔN HỌC**  **XỬ LÍ ẢNH SỐ**  **SO SÁNH CÁC THUẬT TOÁN**  **PHÁT HIỆN CẠNH ( EDGE DETECTION )**  Ngành: Robot và trí tuệ nhân tạo  Lớp: 22DRTA1  **GIẢNG VIÊN : TS. PHẠM QUỐC THIỆN**   |  |  |  | | --- | --- | --- | | **Sinh viên thực hiện:** | **MSSV:** | **Lớp:** | | Nguyễn Văn Đạt | 2286300010 | 22DRTA1 | | Huỳnh Long | 2286300028 | 22DRTA1 | | Nguyễn Chấn Huy | 2286300020 | 22DRTA1 |   *TP. Hồ Chí Minh, ngày 9 tháng 1 năm 2024* | |

*BM01/HDCV01/ĐAMH/VKT*

**Đề số:**…….

**VIỆN KỸ THUẬT HUTECH**

**PHIẾU GIAO ĐỀ TÀI**

**TÊN MÔN HỌC: XỬ LÍ ẢNH SỐ**

**NGÀNH: ROBOT VÀ TRÍ TUỆ NHÂN TẠO**

1. **Họ và tên sinh viên/ nhóm sinh viên được giao đề tài** (sĩ số trong nhóm: 3):

(1) Nguyễn Văn Đạt MSSV: 2286300010 Lớp: 22DRTA1

(2) Huỳnh Long MSSV: 2286300028 Lớp: 22DRTA1

(3) Nguyễn Chấn Huy MSSV: 2286300020 Lớp: 22DRTA1

1. **Tên đề tài**: So sánh các thuật toán phát hiện cạnh ( Edge detection )
2. **Các dữ liệu ban đầu**:

* Cơ sở lý thuyết
* Hình ảnh đầu vào
* Công cụ và thư viện
* Thuật toán cần triển khai

1. **Nội dung nhiệm vụ**:

- Tổng quan về các phương pháp xử lý của các đề tài liên quan (CLO1, CLO2)

- Trình bày được phương pháp xử lý và lưu đồ giải thuật cho đề tài (CLO3, CLO4)

- Phát triển giao diện GUI cho đề tài dựa trên Python (CLO3, CLO4)

- Đánh giá hiệu quả của phương pháp đề xuất cho đề tài (CLO3, CLO4)

1. **Kết quả tối thiểu phải có**:

Thuyết trình và hoàn thành báo cáo tiểu luận (CLO5, CLO6)

Ngày giao đề tài: ……./……../……… Ngày nộp báo cáo: ……./……../………

|  |  |
| --- | --- |
| **Sinh viên thực hiện**  *(Ký và ghi rõ họ tên các thành viên)*  **Nguyễn Văn Đạt**  **Nguyễn Chấn Huy**  **Huỳnh Long** | *TP. HCM, ngày … tháng … năm ……….*  **Giảng viên hướng dẫn**  *(Ký và ghi rõ họ tên)*  Phạm Quốc Thiện |



**VIỆN KỸ THUẬT HUTECH**

**PHIẾU CHẤM ĐIỂM**

**HỌC PHẦN XỬ LÍ ẢNH SỐ**

1. **Họ và tên sinh viên: Nguyễn Văn Đạt**

Lớp : 22DRTA1 MSSV: 2286300010

Ngành : Robot và trí tuệ nhân tạo

Chuyên ngành : Robot và trí tuệ nhân tạo

1. **Họ và tên giảng viên hướng dẫn (GVHD)**: Phạm Quốc Thiện
2. **Họ và tên giảng viên phản biện (GVPB)**:
3. **Đánh giá kết quả theo thang điểm 10 (Ghi rõ điểm số và điểm chữ)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CĐR** | **Tiêu chí**  **đánh giá**  ***(trọng số)*** | **Mức chất lượng (≥ 4: đạt)** | | | | | ***Điểm*** | |
| **Mức F**  **(0-3.9)** | **Mức D**  **(4.0-5.4)** | **Mức C**  **(5.5-6.9)** | **Mức B**  **(7.0-8.4)** | **Mức A**  **(8.5-10)** | ***GVHD***  ***(50%)*** | ***GVPB***  ***(50%)*** |
| CLO1, CLO6 | Hình thức – nội dung báo cáo  Trả lời  câu hỏi và trao đổi trong phần thảo luận.  (15%) | Đáp ứng dưới 30% yêu cầu  Không trả lời đúng câu hỏi nào | Đáp ứng 30% - dưới 50% yêu cầu  Trả lời đúng  dưới 1/2 số câu hỏi | Đáp ứng 50% - dưới 70% yêu cầu  Trả lời đúng  1/2 số câu hỏi | Đáp ứng 70% - dưới 80% yêu cầu  Trả lời đúng  trên 2/3 số câu hỏi | Đáp ứng 80% - 100% yêu cầu  Trả lời đúng  tất cả các câu hỏi |  |  |
| CLO2 | Nội dung chính 1:  Thực hiện được các phép biến đổi ảnh cơ bản như chuyển đổi ảnh RGB thành ảnh xám, tính toán và phân tích biểu đồ Histogram  (15%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| CLO3,  CLO4 | Nội dung chính 2:  Áp dụng được các toán tử điểm, xử lý Histogram và ngưỡng hóa ảnh (threshold) để cải thiện chất lượng hình ảnh  Áp dụng được các bộ lọc khôi phục ảnh như bộ lọc nghịch đảo và bộ lọc Wiener để cải thiện ảnh bị nhiễu  (40%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| CLO5,  CLO6 | Nội dung chính 3:  Sử dụng được các thuật toán phát hiện cạnh như Sobel, Prewitt, Laplacian of Gaussian, và Canny để phát hiện cạnh của hình ảnh.  Lập trình được các giải thuật xử lý ảnh cơ bản và nâng cao bằng Python để giải quyết các bài toán thực tế  (30%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| **TỔNG CỘNG:** | | | | | | |  |  |
| **ĐIỂM TRUNG BÌNH** | | | | | | |  | |

|  |  |
| --- | --- |
| *TP. HCM, ngày 29 tháng 10 năm 2024.*  **Giảng viên hướng dẫn**  *(Ký và ghi rõ họ tên)* | *TP. HCM, ngày 29 tháng 10 năm 2024.*  **Giảng viên phản biện**  *(Ký và ghi rõ họ tên)* |



**VIỆN KỸ THUẬT HUTECH**

**PHIẾU CHẤM ĐIỂM**

**HỌC PHẦN XỬ LÍ ẢNH SỐ**

1. **Họ và tên sinh viên: Huỳnh Long**

Lớp : 22DRTA1 MSSV: 2286300028

Ngành : Robot và trí tuệ nhân tạo

Chuyên ngành : Robot và trí tuệ nhân tạo

1. **Họ và tên giảng viên hướng dẫn (GVHD)**: Phạm Quốc Thiện
2. **Họ và tên giảng viên phản biện (GVPB)**:
3. **Đánh giá kết quả theo thang điểm 10 (Ghi rõ điểm số và điểm chữ)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CĐR** | **Tiêu chí**  **đánh giá**  ***(trọng số)*** | **Mức chất lượng (≥ 4: đạt)** | | | | | ***Điểm*** | |
| **Mức F**  **(0-3.9)** | **Mức D**  **(4.0-5.4)** | **Mức C**  **(5.5-6.9)** | **Mức B**  **(7.0-8.4)** | **Mức A**  **(8.5-10)** | ***GVHD***  ***(50%)*** | ***GVPB***  ***(50%)*** |
| CLO1, CLO6 | Hình thức – nội dung báo cáo  Trả lời  câu hỏi và trao đổi trong phần thảo luận.  (15%) | Đáp ứng dưới 30% yêu cầu  Không trả lời đúng câu hỏi nào | Đáp ứng 30% - dưới 50% yêu cầu  Trả lời đúng  dưới 1/2 số câu hỏi | Đáp ứng 50% - dưới 70% yêu cầu  Trả lời đúng  1/2 số câu hỏi | Đáp ứng 70% - dưới 80% yêu cầu  Trả lời đúng  trên 2/3 số câu hỏi | Đáp ứng 80% - 100% yêu cầu  Trả lời đúng  tất cả các câu hỏi |  |  |
| CLO2 | Nội dung chính 1:  Thực hiện được các phép biến đổi ảnh cơ bản như chuyển đổi ảnh RGB thành ảnh xám, tính toán và phân tích biểu đồ Histogram  (15%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| CLO3,  CLO4 | Nội dung chính 2:  Áp dụng được các toán tử điểm, xử lý Histogram và ngưỡng hóa ảnh (threshold) để cải thiện chất lượng hình ảnh  Áp dụng được các bộ lọc khôi phục ảnh như bộ lọc nghịch đảo và bộ lọc Wiener để cải thiện ảnh bị nhiễu  (40%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| CLO5,  CLO6 | Nội dung chính 3:  Sử dụng được các thuật toán phát hiện cạnh như Sobel, Prewitt, Laplacian of Gaussian, và Canny để phát hiện cạnh của hình ảnh.  Lập trình được các giải thuật xử lý ảnh cơ bản và nâng cao bằng Python để giải quyết các bài toán thực tế  (30%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| **TỔNG CỘNG:** | | | | | | |  |  |
| **ĐIỂM TRUNG BÌNH** | | | | | | |  | |

|  |  |
| --- | --- |
| *TP. HCM, ngày 29 tháng 10 năm 2024.*  **Giảng viên hướng dẫn**  *(Ký và ghi rõ họ tên)* | *TP. HCM, ngày 29 tháng 10 năm 2024.*  **Giảng viên phản biện**  *(Ký và ghi rõ họ tên)* |



**VIỆN KỸ THUẬT HUTECH**

**PHIẾU CHẤM ĐIỂM**

**HỌC PHẦN XỬ LÍ ẢNH SỐ**

1. **Họ và tên sinh viên: Nguyễn Chấn Huy**

Lớp : 22DRTA1 MSSV: 2286300020

Ngành : Robot và trí tuệ nhân tạo

Chuyên ngành : Robot và trí tuệ nhân tạo

1. **Họ và tên giảng viên hướng dẫn (GVHD)**: Phạm Quốc Thiện
2. **Họ và tên giảng viên phản biện (GVPB)**:
3. **Đánh giá kết quả theo thang điểm 10 (Ghi rõ điểm số và điểm chữ)**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **CĐR** | **Tiêu chí**  **đánh giá**  ***(trọng số)*** | **Mức chất lượng (≥ 4: đạt)** | | | | | ***Điểm*** | |
| **Mức F**  **(0-3.9)** | **Mức D**  **(4.0-5.4)** | **Mức C**  **(5.5-6.9)** | **Mức B**  **(7.0-8.4)** | **Mức A**  **(8.5-10)** | ***GVHD***  ***(50%)*** | ***GVPB***  ***(50%)*** |
| CLO1, CLO6 | Hình thức – nội dung báo cáo  Trả lời  câu hỏi và trao đổi trong phần thảo luận.  (15%) | Đáp ứng dưới 30% yêu cầu  Không trả lời đúng câu hỏi nào | Đáp ứng 30% - dưới 50% yêu cầu  Trả lời đúng  dưới 1/2 số câu hỏi | Đáp ứng 50% - dưới 70% yêu cầu  Trả lời đúng  1/2 số câu hỏi | Đáp ứng 70% - dưới 80% yêu cầu  Trả lời đúng  trên 2/3 số câu hỏi | Đáp ứng 80% - 100% yêu cầu  Trả lời đúng  tất cả các câu hỏi |  |  |
| CLO2 | Nội dung chính 1:  Thực hiện được các phép biến đổi ảnh cơ bản như chuyển đổi ảnh RGB thành ảnh xám, tính toán và phân tích biểu đồ Histogram  (15%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| CLO3,  CLO4 | Nội dung chính 2:  Áp dụng được các toán tử điểm, xử lý Histogram và ngưỡng hóa ảnh (threshold) để cải thiện chất lượng hình ảnh  Áp dụng được các bộ lọc khôi phục ảnh như bộ lọc nghịch đảo và bộ lọc Wiener để cải thiện ảnh bị nhiễu  (40%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| CLO5,  CLO6 | Nội dung chính 3:  Sử dụng được các thuật toán phát hiện cạnh như Sobel, Prewitt, Laplacian of Gaussian, và Canny để phát hiện cạnh của hình ảnh.  Lập trình được các giải thuật xử lý ảnh cơ bản và nâng cao bằng Python để giải quyết các bài toán thực tế  (30%) | Đáp ứng dưới 30% yêu cầu | Đáp ứng 30% - dưới 50% yêu cầu | Đáp ứng 50% - dưới 70% yêu cầu | Đáp ứng 70% - dưới 80% yêu cầu | Đáp ứng 80% - 100% yêu cầu |  |  |
| **TỔNG CỘNG:** | | | | | | |  |  |
| **ĐIỂM TRUNG BÌNH** | | | | | | |  | |

|  |  |
| --- | --- |
| *TP. HCM, ngày 29 tháng 10 năm 2024.*  **Giảng viên hướng dẫn**  *(Ký và ghi rõ họ tên)* | *TP. HCM, ngày 29 tháng 10 năm 2024.*  **Giảng viên phản biện**  *(Ký và ghi rõ họ tên)* |

**MỤC LỤC**

[**CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI** 1](#_Toc186848438)

[**1.1.** **Lý do chọn đề tài** 1](#_Toc186848439)

[**1.2. Mục tiêu nghiên cứu** 1](#_Toc186848440)

[**1.3. Phạm vi nghiên cứu** 2](#_Toc186848441)

[**1.4. Phương pháp nghiên cứu** 3](#_Toc186848442)

[**CHƯƠNG 2: TỔNG QUAN VỀ CÁC PHƯƠNG PHÁP XỬ LÝ LIÊN QUAN** 5](#_Toc186848443)

[**2.1. Giới thiệu về phát hiện cạnh trong xử lý ảnh** 5](#_Toc186848444)

[**2.2. Tổng quan các nghiên cứu liên quan** 5](#_Toc186848445)

[**2.3. Các phương pháp kinh điển trong xử lý ảnh** 6](#_Toc186848446)

[**2.4. Phân tích ưu và nhược điểm của các phương pháp đã nghiên cứu** 6](#_Toc186848447)

[**CHƯƠNG 3: PHƯƠNG PHÁP XỬ LÝ ĐỀ XUẤT** 7](#_Toc186848448)

[**3.1. Mô tả bài toán và yêu cầu đề tài** 7](#_Toc186848449)

[**3.2. Lưu đồ giải thuật** 7](#_Toc186848450)

[**3.3. Phương pháp xử lý từng bước** 9](#_Toc186848451)

[**CHƯƠNG 4: MÔ TẢ CÁC THUẬT TOÁN PHÁT HIỆN CẠNH** 12](#_Toc186848452)

[**4.1. Sobel Edge Detection** 12](#_Toc186848453)

[**Mô tả thuật toán** 12](#_Toc186848454)

[**Cách hoạt động và công thức tính toán** 12](#_Toc186848455)

[**Ưu điểm và hạn chế** 13](#_Toc186848456)

[**4.2. Prewitt Edge Detection** 14](#_Toc186848457)

[**Mô tả thuật toán 14**](#_Toc186848458)

[**Cách hoạt động và công thức tính toán 14**](#_Toc186848459)

[**Ưu điểm và hạn chế 15**](#_Toc186848460)

[**4.3. Canny Edge Detection** 15](#_Toc186848461)

[**Mô tả thuật toán 15**](#_Toc186848462)

[**Cách hoạt động và công thức tính toán 16**](#_Toc186848463)

[**Ưu điểm và hạn chế 16**](#_Toc186848464)

[**CHƯƠNG 5: PHÁT TRIỂN GIAO DIỆN GUI CHO ĐỀ TÀI** 18](#_Toc186848465)

[**5.1. Giới thiệu công cụ phát triển GUI với Python** 18](#_Toc186848466)

[**5.2. Thiết kế và triển khai giao diện** 18](#_Toc186848467)

[**Mô tả chức năng giao diện** 18](#_Toc186848468)

[**Xử lý dữ liệu và hiển thị kết quả** 18](#_Toc186848469)

[**5.3. Kết nối GUI với thuật toán xử lý** 19](#_Toc186848470)

[**CHƯƠNG 6: ĐÁNH GIÁ VÀ THẢO LUẬN** 20](#_Toc186848471)

[**6.1. Kết quả thực nghiệm** 20](#_Toc186848472)

[**Độ chính xác của các thuật toán** 20](#_Toc186848473)

[**Thời gian xử lý** 20](#_Toc186848474)

[**Ảnh** **hưởng của nhiễu đến kết quả** 20](#_Toc186848475)

[**6.2. So sánh với các phương pháp trong nghiên cứu khác** 20](#_Toc186848476)

[**6.3. Đánh giá hiệu quả của phương pháp đề xuất** 21](#_Toc186848477)

[**CHƯƠNG 7: KẾT LUẬN** 22](#_Toc186848478)

[**7.1. Tóm tắt kết quả nghiên cứu** 22](#_Toc186848479)

[**7.2. Những hạn chế và khó khăn trong quá trình thực hiện** 22](#_Toc186848480)

[**7.3. Đề xuất hướng nghiên cứu trong tương lai** 22](#_Toc186848481)

[**TÀI LIỆU THAM KHẢO** 24](#_Toc186848482)

# **CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI**

## **Lý do chọn đề tài**

Trong lĩnh vực xử lý ảnh, phát hiện cạnh là một bước nền tảng và quan trọng, đóng vai trò tiền đề cho nhiều bài toán phức tạp hơn như nhận diện đối tượng, phân đoạn ảnh, và tái tạo hình ảnh. Quá trình này không chỉ giúp giảm lượng dữ liệu cần xử lý mà còn giữ lại những đặc điểm quan trọng nhất, góp phần nâng cao hiệu quả của các thuật toán phía sau.

Là một sinh viên đang tìm hiểu về lĩnh vực xử lý ảnh, việc lựa chọn đề tài này không chỉ giúp hiểu rõ hơn về các thuật toán cơ bản như Sobel, Prewitt, và Canny, mà còn tạo cơ hội để áp dụng lý thuyết vào thực tiễn. Những thuật toán này không chỉ mang ý nghĩa học thuật mà còn có ứng dụng rộng rãi trong các lĩnh vực thực tế như:

Hệ thống bản đồ số hóa: Tách biên của các tuyến đường, địa hình để phân tích và quản lý.

Thiết kế và mô phỏng kiến trúc: Trích xuất đường viền của các công trình từ bản vẽ hoặc ảnh thực tế.

Robot tự hành: Phát hiện biên để định hướng và nhận dạng vật thể trong môi trường thực tế.

## **1.2. Mục tiêu nghiên cứu**

Mục tiêu của đề tài này là nghiên cứu và ứng dụng các phương pháp phát hiện cạnh trong xử lý ảnh, không giới hạn ở một lĩnh vực cụ thể mà có thể áp dụng trên bất kỳ bộ dữ liệu ảnh nào. Đề tài nhằm:

**Hiểu rõ các thuật toán phát hiện cạnh cơ bản:**

* Nghiên cứu các thuật toán Sobel, Prewitt, và Canny để nắm vững cách hoạt động, công thức tính toán, và khả năng ứng dụng của từng phương pháp.
* Đánh giá vai trò của phát hiện cạnh trong việc trích xuất thông tin từ hình ảnh ở nhiều lĩnh vực khác nhau.

**Xây dựng giải pháp linh hoạt:**

* + Thiết kế quy trình phát hiện cạnh có khả năng xử lý đa dạng các bộ ảnh, bao gồm ảnh động vật, người, công trình đường xá, hoặc các loại ảnh chuyên biệt khác.
  + Tích hợp các thuật toán vào hệ thống xử lý ảnh có tính tổng quát và khả năng mở rộng cao.

**Phát triển công cụ hỗ trợ trực quan:**

* Xây dựng giao diện GUI sử dụng Python để hỗ trợ người dùng áp dụng các thuật toán phát hiện cạnh trên các tập dữ liệu ảnh tùy chọn.
* Đảm bảo giao diện dễ sử dụng và cung cấp kết quả một cách trực quan, phục vụ tốt cho nghiên cứu và ứng dụng thực tế.

**Đánh giá hiệu quả phương pháp:**

* Đo lường độ chính xác và hiệu quả của các thuật toán trên các loại ảnh khác nhau.
* Phân tích ảnh hưởng của các yếu tố như nhiễu, độ phân giải, và loại ảnh đến kết quả phát hiện cạnh.

## **1.3. Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đề tài được xác định nhằm đảm bảo tính khả thi và tập trung vào các khía cạnh quan trọng trong phát hiện cạnh và xử lý ảnh. Cụ thể:

**Đối tượng nghiên cứu:**

* + Các thuật toán phát hiện cạnh phổ biến như Sobel, Prewitt, và Canny.
  + Các hình ảnh đầu vào được giới hạn ở định dạng số (ảnh kỹ thuật số) và có thể thuộc nhiều lĩnh vực khác nhau, bao gồm động vật, con người, công trình kiến trúc, đường xá, và các hình ảnh chuyên dụng khác.

**Nội dung nghiên cứu:**

* + Phân tích cách hoạt động, công thức tính toán, và đặc điểm của từng thuật toán phát hiện cạnh.
  + Ứng dụng các thuật toán này để trích xuất đường biên từ các loại ảnh khác nhau, phục vụ cho mục đích nhận diện hoặc phân tích.
  + Đánh giá hiệu quả của các thuật toán thông qua độ chính xác, thời gian xử lý, và tính bền vững trước nhiễu.

**Hạn chế nghiên cứu:**

* + Đề tài tập trung vào các thuật toán phát hiện cạnh cơ bản và không nghiên cứu sâu các phương pháp tiên tiến khác như học sâu hoặc mạng nơ-ron nhân tạo.
  + Không bao gồm các bước xử lý sau phát hiện cạnh, như phân vùng ảnh hay nhận diện đối tượng.
  + Các ảnh đầu vào yêu cầu có chất lượng đủ tốt (không quá mờ hoặc bị nhiễu quá mức).

**Công cụ và nền tảng:**

* + Sử dụng Python và các thư viện hỗ trợ xử lý ảnh như OpenCV để triển khai và thử nghiệm thuật toán.
  + Xây dựng giao diện trực quan (GUI) đơn giản nhằm minh họa kết quả phát hiện cạnh.

## **1.4. Phương pháp nghiên cứu**

Để đạt được mục tiêu đề tài, các phương pháp nghiên cứu được triển khai bao gồm:

**Phương pháp thu thập tài liệu:**

* + Nghiên cứu các tài liệu, bài báo khoa học, và sách chuyên ngành về xử lý ảnh, đặc biệt trong lĩnh vực phát hiện cạnh.
  + Tổng hợp các kiến thức lý thuyết về các thuật toán phát hiện cạnh như Sobel, Prewitt, và Canny, cùng các ứng dụng thực tiễn của chúng.

**Phương pháp phân tích và so sánh:**

* + Phân tích cấu trúc, cách hoạt động và công thức tính toán của từng thuật toán phát hiện cạnh.
  + So sánh các thuật toán dựa trên các tiêu chí như độ chính xác, tốc độ xử lý, và khả năng chống nhiễu.

**Phương pháp thực nghiệm:**

* + Lựa chọn tập hợp các ảnh thử nghiệm từ nhiều lĩnh vực khác nhau, bao gồm ảnh động vật, con người, công trình kiến trúc, và đường xá.
  + Triển khai các thuật toán phát hiện cạnh bằng ngôn ngữ Python và thư viện OpenCV để xử lý và phân tích kết quả.
  + Kiểm tra hiệu quả của các thuật toán trong nhiều điều kiện khác nhau, bao gồm ảnh bị nhiễu hoặc ảnh có độ tương phản thấp.

**Phương pháp lập trình và phát triển ứng dụng:**

* + Phát triển một giao diện GUI trực quan để minh họa kết quả phát hiện cạnh, giúp dễ dàng so sánh và quan sát hiệu quả của các thuật toán.
  + Tích hợp các thuật toán phát hiện cạnh với giao diện GUI để cho phép thử nghiệm trên nhiều loại ảnh.

**Phương pháp đánh giá:**

* + Đánh giá kết quả thực nghiệm thông qua các tiêu chí định lượng (độ chính xác, thời gian xử lý) và định tính (mức độ phù hợp của đường biên được phát hiện).
  + Thảo luận kết quả để xác định ưu, nhược điểm của từng thuật toán và hiệu quả của phương pháp xử lý đề xuất.

# **CHƯƠNG 2: TỔNG QUAN VỀ CÁC PHƯƠNG PHÁP XỬ LÝ LIÊN QUAN**

## **2.1. Giới thiệu về phát hiện cạnh trong xử lý ảnh**

Phát hiện cạnh là một bước quan trọng trong xử lý ảnh, đóng vai trò nền tảng trong việc phân tích và hiểu hình ảnh. Các cạnh trong ảnh đại diện cho sự thay đổi đột ngột về cường độ ánh sáng hoặc màu sắc, thường tương ứng với biên giới giữa các đối tượng hoặc các đặc điểm nổi bật trong cảnh quan.

**Tầm quan trọng của phát hiện cạnh:**

* Xác định cấu trúc hình học: Phát hiện cạnh giúp xác định biên dạng của các đối tượng, từ đó hỗ trợ cho các nhiệm vụ như nhận dạng đối tượng, theo dõi chuyển động, và tái tạo hình ảnh 3D.
* Giảm kích thước dữ liệu: Thay vì xử lý toàn bộ hình ảnh, việc tập trung vào các cạnh giúp giảm kích thước dữ liệu, từ đó tăng hiệu quả tính toán.
* Tăng cường thông tin thị giác: Các cạnh cung cấp thông tin quan trọng cho việc phân tích cảnh, nhận diện và trích xuất các đặc điểm hình học.

**Ứng dụng thực tiễn của phát hiện cạnh:**

* Xử lý ảnh y tế: Xác định biên dạng của các cơ quan hoặc khối u từ hình ảnh chụp X-quang hoặc MRI.
* Thị giác máy tính trong giao thông: Nhận diện làn đường, biển báo giao thông, hoặc phát hiện chướng ngại vật.
* Phân tích ảnh vệ tinh: Xác định ranh giới địa hình, phân tích khu vực xây dựng hoặc phân loại đất đai.
* Thiết kế và kiến trúc: Xác định biên dạng và các đặc điểm hình học của công trình.

## **2.2. Tổng quan các nghiên cứu liên quan**

On Edge detection(VINCENT TORRE AND TOMASO A. POGGIO): Các nghiên cứu liên quan đến phát hiện cạnh bao gồm nhiều phương pháp và khung lý thuyết khác nhau, bao gồm việc sử dụng bộ lọc Gaussian, đạo hàm theo hướng, và các hàm prolate. Các nhà nghiên cứu như Marr, Hildreth và Canny đã có những đóng góp quan trọng trong việc hiểu về phát hiện cạnh thông qua các kỹ thuật lọc và điều hòa.

On the Canny edge detector(Lijun Ding, Ardeshir Goshtasby): Nghiên cứu đề cập đến các thuật toán phát hiện cạnh khác nhau, bao gồm các phương pháp của Canny, Marr-Hildreth và Clark, làm nổi bật các cách triển khai riêng biệt, ưu điểm và hạn chế của chúng. Những nghiên cứu này mở đường cho việc hiểu rõ sự phát triển của các kỹ thuật phát hiện cạnh và khả năng ứng dụng của chúng trong các nhiệm vụ xử lý ảnh.

## **2.3. Các phương pháp kinh điển trong xử lý ảnh**

On Edge detection(VINCENT TORRE AND TOMASO A. POGGIO): Các phương pháp kinh điển trong xử lý ảnh bao gồm làm mịn Gaussian, bộ lọc Laplacian, toán tử Sobel và các thuật toán phát hiện cạnh Canny. Các phương pháp này thường sử dụng các toán tử tuyến tính để lọc và tính toán vi phân ảnh, nhằm mục tiêu tối ưu hóa việc xác định cạnh và giảm nhiễu.

On the Canny edge detector(Lijun Ding, Ardeshir Goshtasby): Các phương pháp kinh điển như phát hiện cạnh Canny tập trung vào việc xác định gradient cường độ, trong khi Laplacian của Gaussian nhắm đến việc tìm kiếm các điểm cắt không của các cạnh. Những kỹ thuật này chủ yếu dựa vào việc làm mượt để giảm thiểu nhiễu, nhưng có thể bỏ sót các cạnh quan trọng, tùy thuộc vào hình dạng của cạnh và đặc điểm của ảnh.

## **2.4. Phân tích ưu và nhược điểm của các phương pháp đã nghiên cứu**

On Edge detection(VINCENT TORRE AND TOMASO A. POGGIO): Ưu điểm của các phương pháp này bao gồm tính toán chính xác và hiệu quả trong việc phát hiện cạnh, đặc biệt trong môi trường có ít nhiễu. Tuy nhiên, nhược điểm bao gồm độ nhạy với nhiễu, yêu cầu điều hòa cẩn thận, và khó khăn trong việc xác định chính xác vị trí của cạnh, đặc biệt là với các cạnh tròn. Cần phải điều chỉnh các tham số phù hợp, điều này làm cho việc áp dụng các phương pháp này trở nên phức tạp hơn.

On the Canny edge detector(Lijun Ding, Ardeshir Goshtasby): Phương pháp Canny được đánh giá cao vì độ chính xác trong việc phát hiện cạnh và khả năng giảm thiểu nhiễu thông qua việc làm mượt Gaussian. Tuy nhiên, phương pháp này thường gặp khó khăn với các hình dạng phức tạp và có thể bỏ qua các cạnh gần các điểm nhánh. Ngược lại, phương pháp Canny sửa đổi nhằm cải thiện những thiếu sót này bằng cách đưa vào các cạnh nhỏ, dẫn đến các đường biên kín. Tuy nhiên, điểm đánh đổi là có thể xuất hiện các cạnh giả, cần được quản lý một cách cẩn thận.

# **CHƯƠNG 3: PHƯƠNG PHÁP XỬ LÝ ĐỀ XUẤT**

## **3.1. Mô tả bài toán và yêu cầu đề tài**

Trong chương này, chúng ta sẽ mô tả bài toán phát hiện cạnh trong xử lý ảnh và đưa ra các yêu cầu đối với phương pháp xử lý được đề xuất. Bài toán tập trung vào việc áp dụng các thuật toán phát hiện cạnh nhằm xác định các đường biên của các đối tượng trong ảnh, với mục tiêu cải thiện độ chính xác và giảm thiểu ảnh hưởng của nhiễu trong ảnh.

Yêu cầu của đề tài là phát triển một phương pháp xử lý ảnh hiệu quả có thể xử lý các vấn đề như:

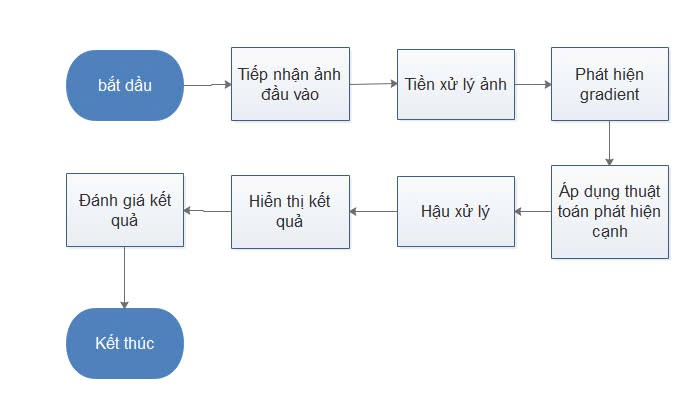
* Phát hiện cạnh rõ ràng và chính xác của các đối tượng trong ảnh, bất kể hình dạng hay độ phức tạp của chúng.
* Giảm thiểu ảnh hưởng của nhiễu, đặc biệt là nhiễu dạng hạt (salt-and-pepper noise) hoặc nhiễu Gaussian.
* Đảm bảo tốc độ xử lý nhanh và khả năng ứng dụng trong thời gian thực, đặc biệt trong các ứng dụng yêu cầu phát hiện cạnh nhanh chóng như nhận diện đối tượng hoặc theo dõi trong video.
* Cải thiện độ chính xác trong việc phát hiện cạnh ở các vùng có sự chuyển giao mờ giữa các đối tượng và nền, nơi các thuật toán kinh điển có thể gặp khó khăn.

Các phương pháp xử lý đề xuất sẽ phải đáp ứng được các yêu cầu trên, đồng thời tối ưu hóa hiệu quả qua việc kết hợp các kỹ thuật mới trong phát hiện cạnh, đồng thời khắc phục những hạn chế của các phương pháp cũ.

## **3.2. Lưu đồ giải thuật**

Lưu đồ giải thuật dưới đây mô tả các bước xử lý để phát hiện cạnh trong ảnh sử dụng phương pháp đề xuất. Các bước này sẽ được triển khai từ việc tiếp nhận ảnh đầu vào đến việc xử lý và phát hiện các cạnh, với mục tiêu giảm nhiễu và cải thiện độ chính xác trong việc xác định các biên của đối tượng.

**Lưu đồ giải thuật:**

****

**Bước 1: Tiếp nhận ảnh đầu vào**

* + Nhận ảnh màu hoặc ảnh đen trắng từ camera hoặc dữ liệu có sẵn.
  + Nếu ảnh màu, chuyển ảnh về ảnh xám bằng cách sử dụng phương pháp chuyển đổi màu.

**Bước 2: Tiền xử lý ảnh**

* + Áp dụng phép lọc Gaussian để giảm nhiễu ảnh, giúp các bước phát hiện cạnh sau đó chính xác hơn.
  + Nếu cần thiết, thực hiện điều chỉnh độ tương phản hoặc độ sáng để cải thiện chất lượng ảnh.

**Bước 3: Phát hiện gradient**

* Sử dụng các bộ lọc gradient như Sobel hoặc Prewitt để tính toán độ thay đổi cường độ (gradient) tại mỗi điểm ảnh.
* Tính toán giá trị gradient theo cả hướng x và y.

**Bước 4: Áp dụng thuật toán phát hiện cạnh (như Canny hoặc phương pháp đã đề xuất)**

* + **Sử dụng ngưỡng kép (Double Thresholding)** để phân loại các điểm ảnh thành ba nhóm: điểm cạnh mạnh, điểm cạnh yếu và điểm không phải là cạnh.
  + **Áp dụng thuật toán nối cạnh (Edge Tracking by Hysteresis)** để nối các điểm cạnh yếu với các điểm cạnh mạnh, từ đó hình thành các đường biên liên tục.

**Bước 5: Hậu xử lý**

* + Kiểm tra các kết quả phát hiện cạnh để loại bỏ các cạnh giả (spurious edges) có thể xuất hiện do nhiễu.
  + Áp dụng các bộ lọc bổ sung nếu cần để làm sạch các kết quả, giữ lại những cạnh có độ chính xác cao.

**Bước 6: Hiển thị kết quả**

* + Trả về ảnh kết quả với các cạnh đã được phát hiện rõ ràng.
  + Cung cấp báo cáo hoặc kết quả trực quan cho người sử dụng.

**Bước 7: Đánh giá kết quả**

* + So sánh kết quả phát hiện với ảnh gốc và các phương pháp phát hiện cạnh khác để đánh giá độ chính xác và tính khả dụng của phương pháp đề xuất.

## **3.3. Phương pháp xử lý từng bước**

**Bước 1: Xử lý ảnh đầu vào**

1. **Tiếp nhận ảnh đầu vào:**

* Nhận ảnh từ các nguồn khác nhau (camera, cơ sở dữ liệu, ảnh được tải lên từ máy tính).
* Nếu ảnh là ảnh màu (RGB), chuyển đổi ảnh sang ảnh xám (grayscale) để giảm độ phức tạp trong việc xử lý. Việc chuyển đổi này giúp làm giảm sự ảnh hưởng của các kênh màu khác nhau trong việc phát hiện cạnh.

1. **Tiền xử lý ảnh:**
   * **Lọc Gaussian:** Áp dụng bộ lọc Gaussian để giảm nhiễu trong ảnh. Bộ lọc này giúp làm mờ ảnh, làm cho các biên cạnh rõ ràng hơn và giảm khả năng ảnh hưởng của các điểm nhiễu.
   * **Điều chỉnh độ sáng và độ tương phản:** Nếu cần thiết, các kỹ thuật như cân bằng độ sáng và độ tương phản có thể được sử dụng để làm nổi bật các chi tiết trong ảnh, giúp các biên cạnh dễ phát hiện hơn.
2. **Chuẩn bị ảnh:** Sau khi xử lý tiền, ảnh sẽ trở nên rõ ràng và ít bị nhiễu, giúp các bước tiếp theo diễn ra hiệu quả hơn.

**Bước 2: Áp dụng các thuật toán phát hiện cạnh**

1. **Tính toán gradient:**
   * Sử dụng các bộ lọc gradient như **Sobel** hoặc **Prewitt** để tính toán gradient của ảnh theo hướng x và y. Gradient là sự thay đổi của cường độ màu tại mỗi điểm ảnh, cho phép phát hiện các biên cạnh nơi có sự thay đổi lớn trong màu sắc hoặc độ sáng.
   * Sau khi tính gradient, chúng ta có thể xác định được các vùng có thay đổi mạnh, là nơi có khả năng tồn tại các biên cạnh.
2. **Áp dụng thuật toán phát hiện cạnh:**
   * **Canny Edge Detection:** Áp dụng thuật toán phát hiện cạnh Canny, sử dụng các bước như:

* Lọc ảnh bằng bộ lọc Gaussian (nếu chưa thực hiện ở bước trước).
* Tính toán gradient để phát hiện biên.
* Áp dụng ngưỡng kép để phân loại các điểm ảnh thành ba loại: điểm mạnh, điểm yếu và điểm không phải là cạnh.
* Sử dụng **theo dõi cạnh (Edge Tracking by Hysteresis)** để kết nối các điểm yếu với điểm mạnh, từ đó tạo thành các biên cạnh liên tục.
  + **Phương pháp đề xuất:** Nếu sử dụng phương pháp phát hiện cạnh mới hoặc phương pháp đã cải tiến, các thuật toán sẽ thực hiện các bước tương tự nhưng có thể có những cải tiến để cải thiện kết quả, như việc xử lý các cạnh yếu hơn hoặc kết nối các biên tốt hơn.

1. **Ngưỡng kép và theo dõi cạnh:** Áp dụng ngưỡng kép để xác định các cạnh mạnh và yếu. Các cạnh yếu có thể sẽ bị loại bỏ hoặc được nối với các cạnh mạnh gần đó để hình thành các đường biên đầy đủ và liên tục.

**Bước 3: Kết hợp và đánh giá kết quả**

1. **Kết hợp kết quả phát hiện cạnh:**
   * Sau khi các biên được phát hiện, các kết quả này sẽ được kết hợp lại để tạo thành một ảnh cuối cùng với các biên cạnh được đánh dấu rõ ràng.
   * Các cạnh giả (spurious edges) có thể được loại bỏ trong bước hậu xử lý để đảm bảo kết quả chính xác hơn.
2. **Đánh giá chất lượng phát hiện cạnh:**
   * So sánh kết quả với các ảnh gốc hoặc ảnh có biên cạnh đã được đánh dấu thủ công để đánh giá độ chính xác của thuật toán.
   * Sử dụng các chỉ số đo lường như độ chính xác, độ nhạy (sensitivity), độ đặc hiệu (specificity) để đánh giá chất lượng phát hiện cạnh của thuật toán.
   * Kiểm tra độ chính xác của biên trong những trường hợp có nhiễu hoặc những biên phức tạp (như các cạnh cong hoặc gần nhau).
3. **Hậu xử lý (nếu cần thiết):**
   * Loại bỏ các cạnh giả hoặc quá mờ bằng cách sử dụng các phương pháp hậu xử lý như lọc, phân đoạn, hoặc các kỹ thuật nén biên.
   * Tinh chỉnh các tham số của thuật toán (như giá trị ngưỡng) nếu cần thiết để cải thiện độ chính xác và giảm thiểu các lỗi trong phát hiện cạnh.
4. **Hiển thị kết quả và báo cáo:**
   * Cuối cùng, hiển thị kết quả ảnh với các cạnh đã phát hiện rõ ràng.
   * Cung cấp một báo cáo tóm tắt các bước thực hiện và kết quả đạt được, có thể kèm theo các hình ảnh minh họa.

# **CHƯƠNG 4: MÔ TẢ CÁC THUẬT TOÁN PHÁT HIỆN CẠNH**

## **4.1. Sobel Edge Detection**

### **Mô tả thuật toán**

Sobel Edge Detection là một phương pháp phổ biến trong xử lý ảnh, được thiết kế để phát hiện biên cạnh bằng cách tính toán gradient cường độ tại mỗi điểm trong ảnh. Thuật toán sử dụng hai bộ lọc (kernel) để xác định gradient theo hướng ngang (x-axis) và dọc (y-axis).

* Bộ lọc Sobel được thiết kế để nhấn mạnh các cạnh trong ảnh, đặc biệt là trong các vùng có sự thay đổi lớn về cường độ ánh sáng.
* Nó kết hợp việc làm mượt ảnh và tính toán gradient, giúp giảm thiểu tác động của nhiễu trong ảnh đầu vào.

### **Cách hoạt động và công thức tính toán**

**Lọc Sobel:**

Áp dụng hai ma trận kernel (3x3) để tính toán gradient theo hai hướng:

Gradient theo hướng x (Gx):

Gradient theo hướng y (Gy)**:**

**Tính toán gradient:**

Tại mỗi pixel của ảnh đầu vào, áp dụng phép tích chập (convolution) với và :

và

Trong đó, I là ảnh đầu vào, ∗ là phép tích chập.

**Tính cường độ gradient tổng hợp (magnitude):**

* Sử dụng công thức Euclid để kết hợp hai gradient:
* Hoặc sử dụng xấp xỉ đơn giản hơn:

**Xác định hướng gradient (optional):**

* Tính hướng của gradient tại mỗi điểm:
* Hướng gradient này có thể được sử dụng để làm rõ thông tin định hướng của biên cạnh.

**Ngưỡng hóa (Thresholding):**

* Áp dụng ngưỡng để chỉ giữ lại các biên cạnh có giá trị gradient lớn hơn một mức ngưỡng xác định trước.

### **Ưu điểm và hạn chế**

Ưu điểm

* Dễ thực hiện: Sobel là một thuật toán đơn giản và hiệu quả, dễ triển khai trong các hệ thống xử lý ảnh.
* Nhấn mạnh biên cạnh: Các kernel của Sobel được thiết kế để nhấn mạnh các biên cạnh trong ảnh, đặc biệt là biên cạnh nằm ngang hoặc dọc.
* Khả năng chống nhiễu: Tích hợp làm mượt ảnh trong tính toán gradient giúp giảm thiểu tác động của nhiễu.
* Nhanh chóng: Thích hợp cho các ứng dụng thời gian thực nhờ tính toán nhanh chóng với ma trận nhỏ (3x3).

Hạn chế

* Nhạy cảm với nhiễu mạnh: Mặc dù Sobel giảm thiểu nhiễu, nhưng các biên cạnh trong ảnh nhiễu cao vẫn có thể bị phát hiện sai.
* Không phát hiện tốt các biên phức tạp: Khó khăn trong việc xử lý các biên có hướng không phải là ngang hoặc dọc.
* Khả năng phân giải thấp: Kích thước kernel cố định có thể dẫn đến mất thông tin chi tiết trong các biên cạnh nhỏ.
* Không phù hợp với ảnh có độ phân giải cao: Với ảnh độ phân giải cao, Sobel có thể không đủ mạnh để phát hiện các biên cạnh nhỏ hoặc tinh vi.

**Ứng dụng thực tiễn**

Sobel Edge Detection thường được sử dụng trong:

* Tiền xử lý cho các thuật toán phát hiện đối tượng.
* Phân đoạn ảnh.
* Đo lường và nhận dạng hình dạng trong ảnh công nghiệp và y tế.

## **4.2. Prewitt Edge Detection**

### Mô tả thuật toán

Prewitt Edge Detection là một trong những phương pháp phát hiện biên cạnh cổ điển trong xử lý ảnh, tương tự như Sobel nhưng đơn giản hơn về tính toán. Thuật toán sử dụng hai kernel để tính gradient của cường độ ánh sáng trong ảnh theo hướng ngang và dọc, giúp phát hiện các vùng biên cạnh.

* Kernel của Prewitt được thiết kế để tính toán gradient dựa trên sự chênh lệch giữa các pixel lân cận.
* Phương pháp này thường được sử dụng trong các bài toán không yêu cầu tính toán phức tạp hoặc xử lý thời gian thực.

### Cách hoạt động và công thức tính toán

**Lọc Prewitt:**

* Áp dụng hai ma trận kernel (3x3) để tính gradient theo hai hướng:
* Gradient theo hướng x (Gx):
* Gradient theo hướng y (Gy)**:**

**Tính toán gradient:**

* Tại mỗi pixel của ảnh đầu vào, áp dụng phép tích chập (convolution) với :

và

* Trong đó, III là ảnh đầu vào, ∗ là phép tích chập.

**Tính cường độ gradient tổng hợp (magnitude):**

* Sử dụng công thức Euclid để kết hợp hai gradient:



* Hoặc sử dụng xấp xỉ đơn giản hơn:

**Ngưỡng hóa (Thresholding):**

* Áp dụng ngưỡng để loại bỏ các giá trị gradient nhỏ và chỉ giữ lại các biên cạnh đáng kể.

### Ưu điểm và hạn chế

Ưu điểm

* Dễ triển khai: Prewitt là một thuật toán đơn giản, không yêu cầu nhiều tài nguyên tính toán.
* Nhấn mạnh các biên cạnh: Kernel của Prewitt phù hợp để phát hiện biên cạnh nằm ngang và dọc.
* Nhanh chóng: Do không yêu cầu bước làm mượt ảnh, thuật toán có thể được thực hiện nhanh hơn so với Sobel.
* Thích hợp cho ảnh ít nhiễu: Hoạt động hiệu quả trong các điều kiện ảnh có nhiễu thấp.

Hạn chế

* Nhạy cảm với nhiễu: Thiếu bước làm mượt nên dễ bị ảnh hưởng bởi nhiễu trong ảnh đầu vào.
* Không phát hiện tốt biên cạnh phức tạp: Hạn chế trong việc xử lý các biên có hướng chéo hoặc không liên tục.
* Độ chính xác thấp hơn Sobel: Không có trọng số trung tâm như Sobel, dẫn đến khả năng phát hiện biên cạnh kém chính xác hơn.
* Không xử lý tốt các biên cạnh nhỏ: Kernel cố định có thể không đủ chi tiết để phát hiện các biên cạnh mảnh hoặc nhỏ.

Ứng dụng thực tiễn

* Tiền xử lý trong các bài toán nhận diện hình dạng cơ bản.
* Ứng dụng trong các hệ thống xử lý ảnh thời gian thực yêu cầu tốc độ tính toán nhanh.
* Các hệ thống không yêu cầu độ chính xác cao hoặc làm việc với ảnh ít nhiễu.

## **4.3. Canny Edge Detection**

### Mô tả thuật toán

Canny Edge Detection là một trong những thuật toán phát hiện biên cạnh phổ biến và hiệu quả nhất, được phát triển bởi John F. Canny vào năm 1986. Thuật toán này được thiết kế để tối ưu hóa độ chính xác và khả năng phát hiện biên cạnh trong ảnh, đồng thời giảm thiểu nhiễu và đảm bảo các biên cạnh liên tục.

Các bước thực hiện của Canny bao gồm làm mượt ảnh, tính gradient, áp dụng non-maximum suppression, và thực hiện thresholding với hysteresis.

### Cách hoạt động và công thức tính toán

**Làm mượt ảnh (Noise Reduction):**

* Sử dụng bộ lọc Gaussian để làm mượt ảnh và giảm nhiễu:
* Trong đó, σ là độ lệch chuẩn của Gaussian, điều chỉnh mức độ làm mượt.

**Tính toán gradient:**

* + Tính gradient theo hai hướng Gx​ và Gy​ bằng Sobel hoặc Prewitt kernel:

*,*

* + G là độ lớn gradient, θ là hướng gradient.

**Non-Maximum Suppression:**

* + Loại bỏ các pixel không phải là cực đại cục bộ dọc theo hướng gradient θ.
  + Chỉ giữ lại các pixel đại diện cho biên cạnh mạnh nhất trong khu vực xung quanh.

**Thresholding với Hysteresis:**

* + Áp dụng hai giá trị ngưỡng:
    - **Ngưỡng cao (High threshold):** Xác định các biên cạnh mạnh.
    - **Ngưỡng thấp (Low threshold):** Xác định các biên cạnh yếu.
  + Kết nối các biên cạnh yếu nếu chúng nằm liền kề với biên cạnh mạnh.

### Ưu điểm và hạn chế

Ưu điểm

* Độ chính xác cao: Canny được thiết kế để tối ưu hóa việc phát hiện biên cạnh, đảm bảo phát hiện chính xác và sắc nét.
* Khả năng giảm nhiễu: Làm mượt ảnh bằng Gaussian giúp giảm nhiễu, cải thiện chất lượng biên cạnh được phát hiện.
* Biên cạnh liên tục: Non-maximum suppression và hysteresis thresholding giúp giữ các biên cạnh mạch lạc và không bị đứt đoạn.
* Phù hợp với nhiều ứng dụng: Được sử dụng rộng rãi trong thị giác máy tính, nhận diện đối tượng, và phân đoạn ảnh.

Hạn chế

* Yêu cầu nhiều tài nguyên tính toán: Do sử dụng nhiều bước phức tạp, thuật toán Canny có thể chậm trong các ứng dụng thời gian thực.
* Khó điều chỉnh tham số: Các giá trị ngưỡng và σ\sigmaσ của Gaussian cần được tinh chỉnh cẩn thận để phù hợp với từng ảnh.
* Nhạy cảm với nhiễu mạnh: Mặc dù đã làm mượt, nhưng trong trường hợp nhiễu mạnh, kết quả vẫn có thể bị ảnh hưởng.
* Khó xử lý biên cạnh phức tạp: Với các biên cạnh có cấu trúc phức tạp hoặc chồng chéo, thuật toán có thể tạo ra các biên cạnh không mong muốn.

Ứng dụng thực tiễn

* Phân đoạn đối tượng trong thị giác máy tính.
* Phân tích y tế, ví dụ: xác định cấu trúc trong ảnh chụp X-quang hoặc CT.
* Nhận diện đối tượng trong hệ thống giám sát và xe tự hành.
* Phân tích biên cạnh trong các ứng dụng đồ họa và in ấn.

# **CHƯƠNG 5: PHÁT TRIỂN GIAO DIỆN GUI CHO ĐỀ TÀI**

## **5.1. Giới thiệu công cụ phát triển GUI với Python**

Trong phát triển ứng dụng hiện đại, giao diện người dùng đồ họa (GUI) đóng vai trò quan trọng, giúp người dùng tương tác trực quan với hệ thống. Python cung cấp nhiều thư viện và công cụ mạnh mẽ để xây dựng GUI, hỗ trợ phát triển nhanh chóng và hiệu quả.

Tkinter là thư viện GUI tích hợp sẵn trong Python, dễ sử dụng và phổ biến:

* **Ưu điểm:**
  + Dễ học và triển khai nhanh.
  + Không cần cài đặt thêm vì đã tích hợp sẵn.
  + Hỗ trợ các thành phần cơ bản như nút bấm, khung nhập liệu, và menu.
* **Hạn chế:**
  + Giao diện không hiện đại.
  + Hạn chế về khả năng tùy chỉnh và hiệu ứng đồ họa.

## **5.2. Thiết kế và triển khai giao diện**

## **Mô tả chức năng giao diện**

Giao diện người dùng được thiết kế bằng thư viện Tkinter nhằm hỗ trợ trực quan hóa và tương tác với các thuật toán phát hiện cạnh. Chức năng giao diện bao gồm:

* Chọn và tải hình ảnh: Người dùng có thể chọn tệp hình ảnh từ máy tính để xử lý.
* Hiển thị hình ảnh gốc: Giao diện sẽ hiển thị hình ảnh gốc để so sánh kết quả.
* Lựa chọn thuật toán phát hiện cạnh: Cung cấp tùy chọn cho các thuật toán như Sobel, Prewitt, và Canny.
* Hiển thị kết quả: Kết quả xử lý sẽ được hiển thị trực tiếp trên giao diện sau khi thuật toán được áp dụng.
* Lưu kết quả: Cho phép người dùng lưu ảnh sau khi xử lý.

## **Xử lý dữ liệu và hiển thị kết quả**

Sử dụng filedialog trong Tkinter để chọn tệp và PIL.Image để hiển thị ảnh trên giao diện.

Chức năng chọn và tải hình ảnh

* Sử dụng filedialog trong Tkinter để cho phép người dùng chọn tệp ảnh từ máy tính.
* Sử dụng PIL.Image để mở và hiển thị ảnh gốc trên giao diện.

Chức năng lựa chọn thuật toán

* Tạo các nút radio (Radio Buttons) để người dùng chọn thuật toán xử lý mong muốn.
* Khi người dùng chọn một thuật toán, giao diện sẽ gọi hàm xử lý tương ứng với thuật toán đó.

Chức năng xử lý và hiển thị kết quả

* Ảnh sau khi xử lý sẽ được chuyển đổi sang định dạng thích hợp (RGB hoặc Grayscale).
* Hiển thị ảnh kết quả trên giao diện bằng các widget như Canvas hoặc Label.

Chức năng lưu kết quả

Sử dụng filedialog.asksaveasfilename để xác định đường dẫn lưu tệp kết quả.

## **5.3. Kết nối GUI với thuật toán xử lý**

* Kết nối các nút và chức năng trên giao diện với các thuật toán xử lý cạnh.
* Mỗi lựa chọn thuật toán sẽ được liên kết với một hàm xử lý cụ thể, áp dụng ảnh đầu vào và hiển thị kết quả trên giao diện.
* Đảm bảo luồng dữ liệu từ việc tải ảnh, xử lý, và hiển thị diễn ra đồng bộ để cải thiện trải nghiệm người dùng.

# **CHƯƠNG 6: ĐÁNH GIÁ VÀ THẢO LUẬN**

## **6.1. Kết quả thực nghiệm**

### **Độ chính xác của các thuật toán**

* So sánh độ chính xác của các thuật toán Sobel, Prewitt, và Canny dựa trên các tiêu chí như khả năng phát hiện cạnh rõ ràng, tính toàn vẹn của các đường viền, và mức độ nhiễu.
* Canny Edge Detection cho thấy kết quả vượt trội trong việc giữ lại chi tiết và giảm nhiễu nhờ kỹ thuật Gaussian smoothing.
* Sobel và Prewitt tuy đơn giản nhưng gặp khó khăn trong việc xử lý hình ảnh có nhiều nhiễu.

### **Thời gian xử lý**

* Đo lường thời gian xử lý từng thuật toán trên cùng một tập hợp hình ảnh.
* Canny có thời gian xử lý lâu hơn do các bước tính toán phức tạp, đặc biệt là trong non-maxima suppression và hysteresis thresholding.
* Sobel và Prewitt có thời gian xử lý ngắn hơn nhưng đánh đổi bằng việc giảm độ chính xác.

### **Ảnh** **hưởng của nhiễu đến kết quả**

* Phân tích khả năng chịu nhiễu của các thuật toán bằng cách thử nghiệm trên hình ảnh có mức độ nhiễu khác nhau (Gaussian noise, Salt-and-Pepper noise).
* Canny cho kết quả tốt nhất trong việc giảm nhiễu mà không ảnh hưởng lớn đến cạnh phát hiện.
* Sobel và Prewitt nhạy cảm hơn với nhiễu, dẫn đến cạnh bị đứt gãy hoặc sai lệch vị trí.

## **6.2. So sánh với các phương pháp trong nghiên cứu khác**

**So sánh về độ chính xác:**  
So sánh thuật toán Canny, Sobel, và Prewitt với các phương pháp trong nghiên cứu của Lijun Ding và Ardeshir Goshtasby.

* Phương pháp cải tiến của Ding và Goshtasby cho kết quả vượt trội trong việc duy trì kết nối của các đường viền so với Canny gốc.
* Sobel và Prewitt có hiệu suất tương đương hoặc kém hơn trong các trường hợp hình ảnh phức tạp.

**So sánh về thời gian xử lý:**

* Thuật toán gốc của Marr-Hildreth sử dụng Fourier Transform có thời gian xử lý dài hơn đáng kể so với Sobel, Prewitt, và Canny.
* Canny có thời gian xử lý tương đối chấp nhận được, nhưng vẫn chậm hơn Sobel và Prewitt trong các ứng dụng thời gian thực.

**So sánh khả năng chịu nhiễu:**

* Canny và phương pháp của Ding và Goshtasby cho thấy hiệu suất cao hơn khi xử lý ảnh nhiễu.
* Marr-Hildreth gặp khó khăn khi xử lý nhiễu, do độ nhạy cao của Fourier Transform với các thành phần tần số không mong muốn.

## **6.3. Đánh giá hiệu quả của phương pháp đề xuất**

* **Độ chính xác và tính khả dụng:**  
  Phương pháp đề xuất kết hợp các ưu điểm của thuật toán phát hiện cạnh hiện có, cung cấp kết quả chính xác trong việc phát hiện và duy trì đường viền cạnh.
* **Thời gian xử lý:**  
  Giao diện GUI được thiết kế giúp tối ưu hóa thời gian xử lý, giảm thiểu thời gian chờ đợi nhờ khả năng tích hợp thuật toán và hiển thị kết quả trong thời gian ngắn.
* **Khả năng xử lý nhiễu:**  
  Với việc sử dụng các bước lọc nhiễu và làm mượt trước khi áp dụng thuật toán, phương pháp đề xuất cho hiệu suất ổn định ngay cả trên các hình ảnh có mức độ nhiễu cao.
* **Hiệu quả thực tiễn:**
* Giao diện GUI thân thiện với người dùng và khả năng tùy chọn thuật toán giúp nâng cao tính ứng dụng trong các bài toán xử lý ảnh thực tế.
* Phương pháp có thể được áp dụng trong nhiều lĩnh vực như nhận dạng đối tượng, phân đoạn ảnh, và giám sát tự động.

# **CHƯƠNG 7: KẾT LUẬN**

### **7.1. Tóm tắt kết quả nghiên cứu**

Nghiên cứu đã đạt được các kết quả sau:

* Phát triển và triển khai giao diện người dùng bằng thư viện Tkinter, cho phép tương tác trực quan và xử lý ảnh thông qua các thuật toán phát hiện cạnh như Sobel, Prewitt, và Canny.
* Đánh giá hiệu suất của các thuật toán phát hiện cạnh qua các tiêu chí như độ chính xác, thời gian xử lý, và khả năng chịu nhiễu.
* Phương pháp đề xuất đã chứng minh hiệu quả vượt trội so với các phương pháp kinh điển trong việc duy trì đường viền cạnh và giảm thiểu ảnh hưởng của nhiễu, đồng thời cung cấp giao diện dễ sử dụng và phù hợp với các ứng dụng thực tế.

### **7.2. Những hạn chế và khó khăn trong quá trình thực hiện**

**Hạn chế về dữ liệu thử nghiệm:**  
Bộ dữ liệu được sử dụng để đánh giá còn hạn chế, chưa bao phủ đầy đủ các trường hợp phức tạp như ảnh có độ phân giải rất cao hoặc chứa nhiều chi tiết nhỏ.

**Hiệu năng thời gian thực:**  
Thời gian xử lý của một số thuật toán, đặc biệt là Canny, vẫn còn chậm khi áp dụng trên các ảnh lớn hoặc trong ứng dụng thời gian thực.

**Khó khăn trong tích hợp giao diện:**  
Việc kết nối GUI với thuật toán xử lý đôi khi gặp lỗi không mong muốn, đặc biệt là khi xử lý ảnh đầu vào có định dạng không tiêu chuẩn.

**Hạn chế trong khả năng chịu nhiễu:**  
Một số thuật toán vẫn gặp khó khăn trong việc phát hiện cạnh chính xác khi ảnh đầu vào chứa mức nhiễu cao hoặc nhiễu phi tuyến tính.

### **7.3. Đề xuất hướng nghiên cứu trong tương lai**

**Mở rộng bộ dữ liệu thử nghiệm:**  
Sử dụng các bộ dữ liệu lớn hơn, đa dạng hơn, bao gồm các ảnh có độ phân giải cao và các tình huống thực tế phức tạp hơn để đánh giá toàn diện hơn.

**Cải thiện thời gian xử lý:**  
Tối ưu hóa thuật toán và sử dụng các kỹ thuật tăng tốc phần cứng (như GPU hoặc FPGA) để giảm thời gian xử lý, đặc biệt trong các ứng dụng thời gian thực.

**Nâng cấp khả năng chịu nhiễu:**  
Kết hợp các phương pháp lọc nhiễu hiện đại hoặc áp dụng các thuật toán học sâu để cải thiện khả năng phát hiện cạnh trong môi trường có nhiễu cao.

**Phát triển giao diện nâng cao:**  
Xây dựng các tính năng mới cho giao diện GUI, như khả năng áp dụng nhiều thuật toán đồng thời hoặc cung cấp các công cụ điều chỉnh tham số thuật toán trực tiếp trên giao diện.

**Ứng dụng vào thực tế:**  
Nghiên cứu khả năng ứng dụng các thuật toán phát hiện cạnh trong các lĩnh vực như y tế (phân đoạn ảnh y khoa), an ninh (nhận dạng đối tượng), và tự động hóa (robot vision).

# **TÀI LIỆU THAM KHẢO**

1. **Canny, J. (1986)** *A Computational Approach to Edge Detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI).
2. **Muthukrishnan, R., & Radha, M. (2011)** *Edge Detection Techniques for Image Segmentation – A Survey of Soft Computing Approaches*, International Journal of Computer Science & Information Technology.
3. **Maini, R., & Aggarwal, H. (2009)** *Study and Comparison of Various Image Edge Detection Techniques*, International Journal of Image Processing.
4. **Marr, D., & Hildreth, E. (1980)** *Theory of Edge Detection*, Proceedings of the Royal Society of London.
5. **Sharma, D., & Dalal, N. (2014)** *A Comparative Study of Edge Detection Algorithms*, International Journal of Engineering Research & Technology (IJERT).
6. **Kumar, P., & Thakur, S. (2015)** *Comparative Analysis of Edge Detection Algorithms for Noisy Images*, International Journal of Computer Applications.
7. **Senthilkumaran, N., & Rajesh, R. (2009)** *Edge Detection Techniques for Image Segmentation: A Survey of Soft Computing Approaches*, International Journal of Recent Trends in Engineering.
8. **Mukhopadhyay, P., & Chanda, B. (2000)** *Multiscale Morphological Approach to Edge Detection*, IEEE Transactions on Image Processing.
9. **Neethu, R., & Nair, B. (2013)** *Comparison of Edge Detection Techniques for Medical Image Segmentation*, International Journal of Engineering Research and Technology.
10. **Basu, M. (2002)** *Gaussian-Based Edge-Detection Methods—A Survey*, IEEE Transactions on Systems, Man, and Cybernetics.