

ỨNG DỤNG WINDOWS CHỈNH SỬA VÀ TRÍCH XUẤT THÔNG TIN TỪ ẢNH

LUẬN VĂN CỦ NHÂN

Cao Thành Danh – 1912836

Giảng viên hướng dẫn

ThS. Nguyễn Khánh Lợi



ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN – ĐIỆN TỬ, BỘ MÔN VIỄN THÔNG

11 – 2023

Số: _____ /BKĐT

Khoa: **Điện – Điện tử**

Bộ Môn: **Viễn Thông**

NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP

1. Họ và tên: Cao Thành Danh MSSV: 1912836
2. Ngành: Điện – Điện tử Chuyên ngành: Kỹ thuật Điện tử - Truyền thông
3. Đề tài: Ứng dụng windows chỉnh sửa và trích xuất thông tin từ ảnh
4. Nhiệm vụ:
 - Thiết kế mạch ... thiết kế hệ thống ...
 - Đo đặc thực nghiệm ... Mô phỏng hệ thống ...
 - Phân tích và so sánh kết quả ...
5. Ngày giao nhiệm vụ luận văn: 02/10/2023
6. Ngày hoàn thành nhiệm vụ: 04/12/2023
7. Họ và tên người hướng dẫn: Phân hướng dẫn
ThS. Nguyễn Khánh Lợi,
BM Viễn Thông, Khoa Điện – Điện Tử 100%
Nội dung và yêu cầu LVTN đã được thông qua Bộ Môn.

TP.HCM, ngày 04 tháng 12 năm 2023

CHỦ NHIỆM BỘ MÔN

PGS. TS. Hà Hoàng Kha

NGƯỜI HƯỚNG DẪN CHÍNH

ThS. Nguyễn Khánh Lợi

PHẦN DÀNH CHO KHOA, BỘ MÔN:

Người duyệt (chấm sơ bộ):

Đơn vị:

Ngày bảo vệ :

Điểm tổng kết:.....

Nơi lưu trữ luận văn:

LỜI CÁM ƠN

Lời đầu tiên em xin được cảm ơn tập thể các thầy cô của trường Đại học Bách Khoa, các thầy cô của khoa Điện – Điện tử và các thầy cô ở bộ môn Viễn thông vì đã tận tình chỉ dạy và trang bị cho em những kiến thức về các môn đại cương cũng như các môn chuyên ngành để làm nền tảng cho việc thực hiện Đồ án cũng như những kinh nghiệm quý báu để nhóm em vững tin hơn trong môi trường làm việc sau này.

Đặc biệt, em xin chân thành cảm ơn thầy Nguyễn Khánh Lợi, người đã tận tình hướng dẫn, giúp đỡ, định hướng, góp ý và cung cấp ý tưởng cũng như tài liệu tham khảo trong thời gian làm Đồ án.

Vì kiến thức bản thân còn nhiều hạn chế, trong quá trình học tập, hoàn thiện báo cáo này em không tránh khỏi những sai lầm. Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các thầy cô để em có điều kiện bổ sung, nâng cao ý thức của mình, phục vụ tốt hơn công tác thực tế sau này.

Em xin chân thành cảm ơn .

TP. HCM, ngày 04, tháng 12 năm 2023

Cao Thành Danh

LỜI CAM ĐOAN

Tôi tên: Cao Thành Danh là sinh viên chuyên ngành Kỹ thuật Điện tử - Truyền thông, khóa 2019, tại Đại học Quốc gia thành phố Hồ Chí Minh – Trường Đại học Bách Khoa. Tôi xin cam đoan những nội dung sau đây là sự thật: (i) Công trình nghiên cứu này hoàn toàn do chính tôi thực hiện; (ii) Các tài liệu và trích dẫn trong luận văn này được tham khảo từ các nguồn thực tế, có uy tín và độ chính xác cao; (iii) Các số liệu và kết quả của công trình này được tôi tự thực hiện một cách độc lập và trung thực.

TP. HCM, ngày 04, tháng 12 năm 2023

Cao Thành Danh

TÓM TẮT LUẬN VĂN

Tóm tắt luận văn bằng tiếng Việt. Giới hạn trong 1 trang. Nội dung tóm tắt bao gồm: bài toán nghiên cứu (mục tiêu). Phương pháp nghiên cứu. Kết quả nghiên cứu.

ABSTRACT

Tóm tắt luận văn bằng tiếng Anh. Giới hạn trong 1 trang. Nội dung tóm tắt bao gồm: bài toán nghiên cứu (mục tiêu). Phương pháp nghiên cứu. Kết quả nghiên cứu.

MỤC LỤC

LỜI CÁM ƠN	i
LỜI CAM ĐOAN	ii
TÓM TẮT LUẬN VĂN	iii
ABSTRACT	iv
DANH SÁCH BẢNG	vii
DANH SÁCH HÌNH VẼ	viii
DANH SÁCH TỪ VIẾT TẮT	ix
CHƯƠNG 1. GIỚI THIỆU	1
1.1 Đặt vấn đề	1
1.2 Phạm vi và phương pháp nghiên cứu	3
1.2.1 Phạm vi nghiên cứu	3
1.2.2 Phương pháp nghiên cứu	3
1.3 Các đóng góp của luận văn	4
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	5
2.1 Giới thiệu về xử lý ảnh và thị giác máy.....	5
2.1.1 Khái niệm về xử lý ảnh	5
2.1.2 Khái niệm về thị giác máy tính	6
2.1.3 Một số ứng dụng xử lý ảnh và thị giác máy tính	6
2.2 Cơ sở lý thuyết về OpenCV.....	8
2.2.1 Giới thiệu OpenCV	8
2.2.2 Các module phổ biến của OpenCV	9
2.2.3 Ứng dụng của OpenCV	9
2.3 Giới thiệu các kỹ thuật xử lý ảnh	10
2.3.1 Kỹ thuật xử lý cơ bản	10
2.3.2 Vẽ lại vùng ảnh - Inpaint	11
2.3.3 Bộ lọc ảnh – Filter.....	13
2.4 Giới thiệu về OCR:	14
2.4.1 Giới thiệu OCR	14
2.4.2 Phân loại OCR	15
2.4.3 Ứng dụng OCR:	15
2.5 Một số thư viện sử dụng	16
2.5.1 WinUI 3 – .NET Framework 4.8	16
2.5.2 OpenCVSharp	18
2.5.3 Tesseract Python và IronTesseract.....	19

2.5.4 Pilgram Python.....	20
2.5.5 Biểu thức chính quy – Regex.....	21
CHƯƠNG 3. KẾT QUẢ VÀ PHÂN TÍCH.....	22
3.1 Phương pháp tiếp cận.....	22
3.1.1 Thủ nghiệm trên Python	22
3.1.2 Thủ nghiệm trên C#:	26
3.2 Kết quả và phân tích.....	29
3.2.1 Thực hiện Inpaint trong Python	29
3.2.2 Thực hiện Filter trong Python	38
3.2.3 Thực hiện OCR trong Python	42
3.2.4 Thực hiện Inpaint trong C Sharp	51
3.2.5 Thực hiện Filter trong C Sharp	57
3.2.6 Thực hiện OCR trong C Sharp.....	60
3.3 Kết luận chương	62
CHƯƠNG 4. KẾT LUẬN.....	63
4.1 Tóm tắt và kết luận chung	63
4.2 Hướng phát triển.....	63
PHỤ LỤC A.....	65
A.1 Code chương trình Filter Python	65
A.2 Code chương trình xử lý C Sharp	67
TÀI LIỆU THAM KHẢO	72

DANH SÁCH BẢNG

No table of figures entries found.

DANH SÁCH HÌNH VẼ

Hình 2.3-1 Giới thiệu về Chroma**Error! Bookmark not defined.**

DANH SÁCH TỪ VIẾT TẮT

SNR	Signal to noise ratio
Ghi chú	Ghi các từ viết tắt sử dụng trong luận văn vào bảng này theo thứ tự alphabet

CHƯƠNG 1. GIỚI THIỆU

1.1 Đặt vấn đề

Trong thời đại số hóa ngày nay, việc chỉnh sửa ảnh và trích xuất thông tin từ hình ảnh đã trở nên vô cùng phổ biến và quan trọng. Cùng với sự phát triển của công nghệ, các công cụ chỉnh sửa ảnh đã trở nên dễ dàng sử dụng hơn bao giờ hết, cho phép người dùng tạo ra những tác phẩm sáng tạo, tinh chỉnh hình ảnh theo ý thích của mình. Đồng thời, việc trích xuất thông tin từ hình ảnh cũng vô cùng quan trọng, đặc biệt trong các lĩnh vực như trí tuệ nhân tạo, khoa học dữ liệu, y học, an ninh mạng và nhiều lĩnh vực khác. Tuy nhiên việc trích xuất chỉnh sửa ảnh qua các phần mềm phức tạp và trích xuất thông tin thủ công lại thường gây nhiều khó khăn.

Trong vai trò là một người dùng cần sự đơn giản và hiệu quả trong công việc, cần có một công cụ hỗ trợ các thao tác xử lý liên quan đến hình ảnh nhanh chóng. Cần khắc phục được các nhược điểm thường gặp sau:

- Quá trình xử lý quá phức tạp: Việc chỉnh sửa một bức ảnh như xóa một vật thể, chỉnh màu, cũng như là thay đổi kích thước ảnh cần cài một phần mềm chuyên biệt, quá trình cài đặt lâu và cần phải nghiên cứu thao tác sử dụng nhiều. Việc này gây trở ngại với nhu cầu xử lý nhanh gọn
- Xử lý thủ công mất thời gian: Việc trích xuất thông tin từ ảnh bằng mắt thường đã là một điều tốn rất nhiều thời gian và công sức, bên cạnh đó còn kết hợp với việc nhập liệu và lưu trữ bằng tay khiến các công việc này lặp đi lặp lại quá nhiều lần. Gây lãng phí thời gian và nhân lực cho việc này.
- Không hiệu quả: Việc thực hiện các công việc trên vừa mất nhiều thời gian nhưng đôi lúc lại không thật sự chính xác và kịp thời cho các nhu cầu tức thì.

Trước những thách thức nêu trên, sự kết hợp giữa xử lý ảnh và thị giác máy tính đã mở ra một hướng giải quyết tiềm năng. Cụ thể, việc sử dụng kỹ thuật Inpaint thông qua OpenCV và OCR để phát triển một ứng dụng trên Windows dùng WinUI giúp chỉnh sửa và trích xuất thông tin từ ảnh giúp giải quyết các vấn đề trên một cách hiệu quả.

OpenCV là một thư viện mã nguồn mở rộng lớn và mạnh mẽ dành cho xử lý ảnh và thị giác máy tính. Viết tắt của "Open Source Computer Vision Library," OpenCV cung cấp một loạt các công cụ và chức năng để xử lý ảnh số, nhận diện vật thể, phát hiện đối tượng, xử lý video và nhiều ứng dụng khác trong lĩnh vực thị giác máy tính. OpenCV giúp người dùng dễ dàng thực hiện các tác vụ xử lý ảnh và thị giác máy tính một cách hiệu quả, từ các thao tác cơ bản đến các ứng dụng phức tạp.

OCR là viết tắt của "Optical Character Recognition" (Nhận diện ký tự quang học). Đây là một công nghệ dùng để chuyển đổi các hình ảnh hoặc các tài liệu chứa văn bản, ký tự thành dạng văn bản có thể chỉnh sửa được trên máy tính. Công nghệ OCR có nhiều ứng dụng rộng rãi, từ việc chuyển đổi tài liệu giấy sang dạng điện tử, tạo điều kiện cho việc tìm kiếm và chỉnh sửa văn bản, đến việc xử lý tự động thông tin từ các tài liệu, hóa đơn, bằng lái xe, thẻ ID, và nhiều ứng dụng khác trong các lĩnh vực như y tế, ngân hàng, công nghiệp, văn phòng và quản lý tài liệu.

Mục tiêu của đề tài là phát triển một ứng dụng chỉnh sửa ảnh hiệu quả và trích xuất thông tin nhanh chóng từ ảnh trong các trường hợp sử dụng phổ biến như: Xóa bỏ vật thể, làm mờ vùng ảnh, thực hiện thay đổi màu ảnh qua các filter, thay đổi kích thước ảnh, trích xuất thông tin từ ảnh có chứa văn bản nhanh chóng v.v... Ứng dụng được kỳ vọng sẽ giải quyết các vấn đề trên tiết kiệm thời gian và nâng cao trải nghiệm trong công việc.

Để đạt được mục tiêu của đề tài, chúng ta sẽ tiến hành các bước sau:

1. Nghiên cứu OpenCV và nguyên lý hoạt động của nó.
2. Tìm hiểu các kỹ thuật xử lý ảnh như: Inpaint (xóa vật thể khỏi ngữ cảnh), xử lý màu.
3. Nghiên cứu OCR và cách vận hành trong C Sharp
4. Xây dựng ứng dụng dùng .NET Framework 4.8 sử dụng Framework WinUI 3 để xây dựng ứng dụng trên Windows
5. Thực hiện kiểm tra, đánh giá và cải thiện hiệu quả thực hiện của ứng dụng.

1.2 Phạm vi và phương pháp nghiên cứu

1.2.1 Phạm vi nghiên cứu

Đề tài tập trung vào việc phát triển một ứng dụng hỗ trợ người dùng có nhu cầu chỉnh sửa ảnh và trích xuất thông tin nhanh chóng. Phạm vi của nghiên cứu bao gồm:

Xây dựng giao diện và tương tác người dùng: Thiết kế một giao diện trực quan và thân thiện giúp người dùng tương tác với ứng dụng trên Windows quen thuộc.

Xây dựng kiến trúc ứng dụng: Phát triển kiến trúc phần mềm phù hợp cho việc quản lý hệ thống và xử lý dữ liệu ảnh và văn bản.

Xây dựng các đầu ra phù hợp: Lựa chọn các dạng đầu ra của phần mềm từ ảnh, văn bản dưới nhiều hình thức phù hợp với nhu cầu người dùng.

1.2.2 Phương pháp nghiên cứu

Trong quá trình thực hiện đề tài, chúng ta sẽ áp dụng một loạt các phương pháp để đạt được mục tiêu đã đề ra:

Nghiên cứu và tìm hiểu OpenCV: Đầu tiên, tiến hành nghiên cứu chi tiết về OpenCV, hiểu cách thư viện này hoạt động và khả năng của nó trong việc xử lý ảnh.

Thiết kế giao diện dùng WinUI 3 Framework: Dựa trên giao diện Windows phổ biến tiến hành nghiên cứu sử dụng các thành phần giao diện kết hợp với nhau để ra được giao diện phù hợp. Bên cạnh đó cũng cần chú trọng việc tổ chức thuận tiện cho việc phát triển và bảo trì.

Nghiên cứu thành phần chỉnh sửa ảnh: Sử dụng các thành phần hỗ trợ của OpenCV và pilgram, chúng ta sẽ xây dựng một số tính năng xử lý ảnh có khả năng xử lý yêu cầu xóa vật thể và thay đổi màu ảnh.

Nghiên cứu trích xuất thông tin thông qua OCR: Tìm hiểu về lý thuyết, nguyên lý và cách thức triển khai của OCR. Dựa kiến thức này vào ứng dụng để trích xuất thông tin đơn giản đến phức tạp. Giúp quá trình trích xuất thông tin đỡ tốn quá nhiều chi phí.

Thử nghiệm và đánh giá: Cuối cùng, chúng ta sẽ tiến hành các thử nghiệm với trường hợp xử lý ảnh và trích xuất thông tin thực tế để đánh giá hiệu suất của ứng dụng Windows. Mọi vấn đề gặp phải sẽ được ghi nhận và cải thiện ứng dụng.

1.3 Các đóng góp của luận văn

Các đóng góp chính của đề tài này bao gồm:

- Xử lý ảnh đơn giản, nhanh gọn và chính xác

Ứng dụng Windows trong đề tài sẽ giúp cải thiện trải nghiệm xử lý ảnh đơn. Thay vì phải tốn thời gian và công sức để sử dụng một ứng dụng phức, người dùng có thể tương tác với ứng dụng đơn giản và thuận tiện để xóa vật thể và chỉnh sửa ảnh.

- Sử dụng OCR trích xuất tự động:

Sử dụng kĩ thuật OCR, đề tài này giúp kết hợp xử lý ảnh và thị giác máy để xây dựng một ứng dụng thông minh có khả năng trích xuất thông tin tự động theo nhu cầu của người dùng. Điều này giúp tiết kiệm thời gian và công sức trong việc trích xuất thông tin bằng thủ công.

- Môi trường xây dựng quen thuộc và dễ sử dụng:

Việc lựa chọn xây dựng ứng dụng trên Windows vì đây là một trong những hệ điều hành được sử dụng phổ biến hiện nay. Qua đó người dùng không cần mất quá nhiều thời gian tìm hiểu và vận dụng công việc để mang lại hiệu quả kịp thời.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1 Giới thiệu về xử lý ảnh và thị giác máy

2.1.1 Khái niệm về xử lý ảnh

Xử lý ảnh là một dạng xử lý tín hiệu trong kỹ thuật điện và toán học liên quan đến phân tích, thay đổi và cải thiện chất lượng tín hiệu. Trong việc xử lý tín hiệu này có đầu vào là ảnh và đầu ra cũng là ảnh. Đây là lĩnh vực quan trọng trong khóa học máy tính và công nghệ thông tin, có ứng dụng rộng rãi trong nhiều lĩnh vực từ công nghiệp đến y tế, giải trí và nhiều ngành công nghiệp khác.

- Phân loại xử lý ảnh:
 - + Xử lý ảnh tương tự: Xử lý ảnh tương tự sử dụng các phương pháp và kỹ thuật được áp dụng trực tiếp lên tín hiệu tương tự, ví dụ như các tín hiệu điện áp, dòng điện hoặc sóng. Trong kiểu xử lý này, hình ảnh được xử lý dưới dạng điện bằng cách thay đổi tín hiệu điện. Ví dụ phổ biến bao gồm là hình ảnh truyền hình.
 - + Xử lý ảnh kỹ thuật số: Xử lý ảnh số là việc sử dụng các thuật toán và phương pháp tính toán để xử lý hình ảnh dưới dạng số, thông qua việc biểu diễn ảnh dưới dạng ma trận số.
- So sánh giữa xử lý ảnh tương tự và xử lý ảnh số:
 - + Tín hiệu đầu vào: Xử lý ảnh tương tự có đầu vào là tín hiệu liên tục, không giới hạn về số lượng mẫu, ví dụ như dữ liệu từ camera hoặc các thiết bị cảm biến. Trong khi đó, Xử lý ảnh số có đầu vào là dữ liệu số được lấy từ các thiết bị số hóa như máy ảnh kỹ thuật số, máy quét hoặc bất kỳ nguồn dữ liệu số nào khác.
 - + Xử lý: Xử lý ảnh tương tự thực hiện các phép toán trên tín hiệu tương tự thông qua việc sử dụng các linh kiện điện tử như các bộ lọc, ampli, và các thiết bị điện tử khác. Việc xử lý ảnh số thiền các phép toán trên dữ liệu số bằng cách sử dụng các thuật toán và kỹ thuật tính toán trong máy tính.

2.1.2 Khái niệm về thị giác máy tính

Thị giác máy tính là một lĩnh vực trong khoa học máy tính tập trung vào việc nghiên cứu, phát triển và triển khai các phương pháp và thuật toán để máy tính có khả năng "nhìn" và "hiểu" hình ảnh và video giống như con người. Nó liên quan chặt chẽ đến xử lý ảnh, trí tuệ nhân tạo, và các lĩnh vực khác như thị giác nhân tạo và thị giác máy tính học sâu.

Thị giác máy tính chia thành các thành phần sau:

- + Phân tích hình ảnh: Thị giác máy tính cung cấp các công cụ và kỹ thuật để phân tích và hiểu nội dung trong hình ảnh. Điều này bao gồm việc nhận dạng đối tượng, phát hiện khuôn mặt, xác định vị trí và đặc điểm của các vật thể trong ảnh.
- + Xử lý ảnh: Các phương pháp xử lý ảnh được áp dụng để cải thiện chất lượng, loại bỏ nhiễu, tăng cường đặc điểm và thông tin trong hình ảnh. Công việc này bao gồm việc lọc, cắt tia, cân bằng màu sắc, biến đổi hình dạng, và nhiều kỹ thuật khác.
- + Nhận diện đối tượng và phân loại: Quá trình này liên quan đến việc nhận biết và phân loại các vật thể, đối tượng trong hình ảnh thành các danh mục đã được xác định trước. Điều này có thể bao gồm việc sử dụng mô hình học máy để huấn luyện và dự đoán đối tượng trong ảnh.
- + Tracking (Theo dõi): Theo dõi đối tượng hoặc chuyển động trong chuỗi hình ảnh hoặc video là một phần quan trọng của thị giác máy tính. Công việc này bao gồm việc xác định và theo dõi sự di chuyển của các vật thể qua các khung hình liên tiếp.

Trí tuệ nhân tạo và học sâu (deep learning) đang đóng vai trò quan trọng trong thị giác máy tính, cung cấp các mô hình mạng nơ-ron sâu để giải quyết các vấn đề phức tạp trong việc nhận dạng, phân loại và hiểu biết hình ảnh một cách hiệu quả. Lĩnh vực này đang ngày càng phát triển với nhiều ứng dụng thực tế tiềm năng đối với nhiều ngành công nghiệp khác nhau.

2.1.3 Một số ứng dụng xử lý ảnh và thị giác máy tính

- Nhận dạng khuôn mặt: Computer Vision không chỉ có thể nhận dạng được khuôn mặt mà còn xác định được khuôn mặt đó là ai. Nhận dạng khuôn mặt được dùng nhiều trong các lĩnh vực như: An ninh và giám sát, quản lý danh tính, công nghiệp và sản xuất hàng tiêu dùng.

Ví dụ: Nhận diện khuôn mặt được sử dụng trong hệ thống chấm công của tập đoàn CMC dùng để điểm danh nhân viên thay cho hệ thống chấm công bằng vân tay từ ngày 4 tháng 2 năm 2020. Đây là một ứng dụng nhận dạng khuôn mặt để tránh sự lây lan của virus. Sáng kiến này đã được phát triển và triển khai bởi Viện CIST thuộc Tập đoàn CMC.

- Nhận dạng vật thể: Nhận dạng vật thể được sử dụng rất nhiều trong xử lý công nghiệp, giúp giảm chi phí cho việc giám sát và kiểm tra sản phẩm trong dây truyền sản xuất. Nhận dạng vật thể được sử dụng nhiều trong các lĩnh vực như: Sản xuất linh kiện điện tử, điều khiển tự động hóa robot, ô tô tự hành v.v...

Ví dụ: Nhận dạng vật thể và môi trường xung quanh được nghiên cứu trong phát triển ô tô tự hành. Autopilot hệ thống hỗ trợ người lái tiên tiến do Tesla phát triển từ năm 2013 là một ví dụ điển hình. Autopilot được thiết kế để hỗ trợ lái xe thông qua việc sử dụng cảm biến, máy tính và phần mềm thực hiện các tính toán thị giác máy tính để kiểm soát và tự động hóa một số chức năng lái xe. Cụ thể Autopilot có thể giúp người lái xe đỗ xe tự động, xử lý các tình huống trong không gian hẹp. Còn nhiều tính năng khác giúp người dùng cài thiện trải nghiệm khi lái xe.

- Trích xuất thông tin từ ảnh: Thị giác máy tính có thể nhìn thấy, hiểu và tương tác với thế giới xung quanh thông qua hình ảnh. Có thể được sử dụng để trích xuất thông tin từ ảnh bằng cách áp dụng các kỹ thuật xử lý hình ảnh và học máy. Việc trích xuất thông tin từ ảnh giúp việc lấy thông tin tự động và linh hoạt hơn nhiều so với việc tìm kiếm bằng văn bản thông thường. Áp dụng trong việc tra cứu thông tin, bán lẻ, chăm sóc khách hàng, kiểm duyệt và truy xuất dữ liệu.

Ví dụ: Google Lens là một ứng dụng di động sử dụng kỹ thuật nhận dạng hình ảnh (Image Recognition Technology) do Google phát triển từ năm 2017. Dùng để tìm kiếm

và trích xuất thông tin từ ảnh vô cùng hiệu quả. Hiện đang được tích hợp nhiều trong các ứng dụng như Google Photos, Google Assistant và trong năm 2023 là Bard AI. Các tính huống rất thông dụng như người đi du lịch có thể xác định địa điểm họ đi tên là gì, tên gọi cụ thể của các vật thể mà họ gặp, thậm chí quen thuộc là việc lấy dữ liệu chữ từ một bức ảnh, giúp bạn trích xuất và sử dụng văn bản một cách nhanh chóng và linh hoạt. Google Lens vẫn đang được phát triển, nhưng nó đã trở thành một công cụ hữu ích cho nhiều người dùng. Với sự phát triển của thị giác máy tính, Google Lens có khả năng trở thành một công cụ quan trọng trong tương lai.

2.2 Cơ sở lý thuyết về OpenCV

2.2.1 Giới thiệu OpenCV

Thư viện OpenCV được bắt đầu xây dựng bởi Intel năm 1999 bởi Gary Bradsky. OpenCV viết tắt cho Open Source Computer Vision Library. OpenCV là thư viện nguồn mở hàng đầu cho Computer Vision và Machine Learning, và hiện có thêm tính năng tăng tốc GPU cho các hoạt động theo real-time. [Link](#)

OpenCV là hỗ trợ đắc lực cho lĩnh vực thị giác máy tính (Computer Vision), xử lý ảnh và máy học. Nó được phát hành theo giấy phép [BSD](#), do đó nó hoàn toàn miễn phí cho cả học thuật và thương mại. OpenCV được thiết kế để tính toán hiệu quả và với sự tập trung nhiều vào các ứng dụng thời gian thực. Nó bao gồm một bộ công cụ mạnh mẽ cho các nhiệm vụ như:

- Xử lý hình ảnh: OpenCV cung cấp một loạt các hàm để xử lý hình ảnh, chẳng hạn như lọc, thay đổi kích thước, và trích xuất đặc trưng.
- Nhận dạng hình ảnh: OpenCV cung cấp một loạt các thuật toán để nhận dạng các đối tượng trong hình ảnh, chẳng hạn như khuôn mặt, biển báo giao thông, và sản phẩm.
- Theo dõi đối tượng: OpenCV cung cấp các thuật toán để theo dõi các đối tượng trong video, chẳng hạn như người, xe, và vật thể.
- Trí tuệ nhân tạo: OpenCV tích hợp với các thư viện trí tuệ nhân tạo, chẳng hạn như TensorFlow và PyTorch, để cung cấp các khả năng nhận dạng hình ảnh và theo dõi đối tượng tiên tiến hơn.

2.2.2 Các module phổ biến của OpenCV

OpenCV có cấu trúc module, nghĩa là gói bao gồm một số thư viện liên kết tĩnh (static libraries) hoặc thư viện liên kết động (shared libraries). Một số module phổ biến có sẵn như sau:

- + Core functionality (core) – module nhỏ gọn để xác định cấu trúc dữ liệu cơ bản, bao gồm mảng đa chiều dày đặc và nhiều chức năng cơ bản được sử dụng bởi tất cả các module khác.
- + Image Processing (imgproc) – module xử lý hình ảnh gồm cả lọc hình ảnh tuyến tính và phi tuyến (linear and non-linear image filtering), phép biến đổi hình học (chỉnh size, afin và warp phối cảnh, ánh xạ lại dựa trên bảng chung), chuyển đổi không gian màu, biến đổi, và nhiều cái khác.
- + Video Analysis (video) – module phân tích video bao gồm các tính năng ước tính chuyển động, tách nền, và các thuật toán theo dõi vật thể.
- + Camera Calibration and 3D Reconstruction (calib3d) – thuật toán hình học đa chiều cơ bản, hiệu chuẩn máy ảnh single và stereo (single and stereo camera calibration), dự đoán kiểu dáng của đối tượng (object pose estimation), thuật toán thư tín âm thanh nổi (stereo correspondence algorithms) và các yếu tố tái tạo 3D.
- + 2D Features Framework (features2d) – phát hiện các đặc tính nổi bật của bộ nhận diện, bộ truy xuất thông số, thông số đối chọi.
- + Object Detection (objdetect) – phát hiện các đối tượng và mô phỏng của các hàm được định nghĩa sẵn – predefined classes (vd: khuôn mặt, mắt, cốc, con người, xe hơi,...).
- + High-level GUI (highgui) – giao diện dễ dùng để thực hiện việc giao tiếp UI đơn giản.
- + Video I/O (videoio) – giao diện dễ dùng để thu và mã hóa video.
- + GPU – Các thuật toán tăng tốc GPU từ các modun OpenCV khác.

Và còn rất nhiều module đặc trưng hỗ trợ thuật toán phức tạp hơn cho xử lý ảnh và trí tuệ nhân tạo khác.

2.2.3 Ứng dụng của OpenCV

OpenCV được sử dụng rộng rãi trong nhiều ứng dụng khác nhau, bao gồm:

- Công nghệ nhận dạng: OpenCV được sử dụng trong các ứng dụng nhận dạng khuôn mặt, nhận dạng biển báo giao thông, và nhận dạng sản phẩm.
- Xử lý video: OpenCV được sử dụng trong các ứng dụng xử lý video, chẳng hạn như phát hiện chuyển động và theo dõi đối tượng.
- Robotics: OpenCV được sử dụng trong các ứng dụng robotics, chẳng hạn như định vị và lập bản đồ.
- Công nghệ thực tế ảo và thực tế tăng cường: OpenCV được sử dụng trong các ứng dụng thực tế ảo và thực tế tăng cường, chẳng hạn như theo dõi chuyển động và nhận dạng đối tượng.

2.3 Giới thiệu các kỹ thuật xử lý ảnh

2.3.1 Kỹ thuật xử lý cơ bản

Trước khi tìm hiểu về các kỹ thuật xử lý ảnh số, tôi xin nhắc lại một vài khái niệm về ảnh số. Ảnh số là hình ảnh được biểu diễn và lưu trữ dưới dạng dữ liệu số. Thay vì sử dụng các phim hoặc bản ghi hóa học như trong ảnh phim, ảnh số được tạo ra bằng cách chụp ảnh thông qua các thiết bị kỹ thuật số như máy ảnh số, điện thoại thông minh hoặc máy quét. Thành phần nhỏ nhất của ảnh số là pixel, mỗi pixel lưu trữ thông tin màu sắc và vị trí cụ thể trong ảnh.

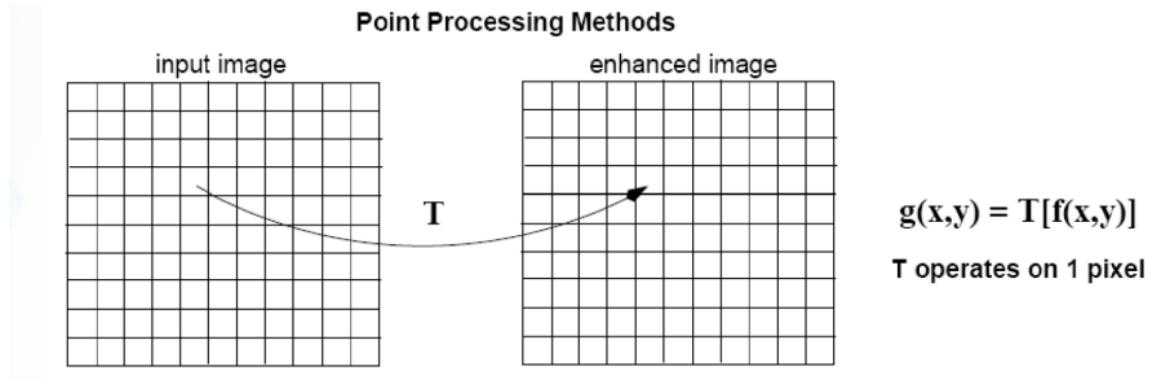
4	4	4	1
5	4	6	2
8	6	8	2
1	2	3	3

Hình 2.3-1 Minh họa về ảnh số

Với ưu điểm dễ dàng xử lý và chỉnh sửa của ảnh số, có nhiều kỹ thuật dùng để xử lý ảnh số khác nhau được ra đời. Tạm chia làm hai nhóm chính sau:

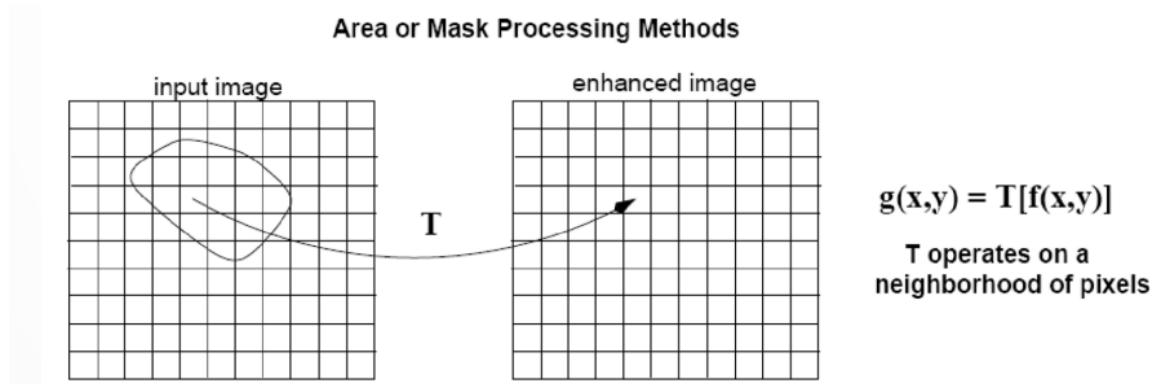
2.3.1.1 Kĩ thuật biến đổi miền không gian:

Các kỹ thuật xử lý ảnh biến đổi miền không gian bao gồm: phép toán điểm và phép toán lân cận.



Hình 2.3-2 Phép toán điểm trên ảnh

Các phép toán điểm sử dụng phổ biến trong xử lý ảnh như:



Hình 2.3-3 Phép toán vùng lân cận trên ảnh

2.3.2 Vẽ lại vùng ảnh - Inpaint

Thuật toán Inpaint là một thuật toán xử lý ảnh được sử dụng để loại bỏ các đối tượng không mong muốn khỏi ảnh. Thuật toán này được phát triển bởi Alexei Efros và Vladimir Kolmogorov vào năm 2007. Nó đặc biệt hữu ích trong việc khôi phục các bức ảnh cũ có thể bị trầy xước ở các cạnh hoặc vết mực trên đó. Chúng có thể được loại bỏ bằng kỹ thuật số thông qua phương pháp này. Tính năng vẽ trong ảnh hoạt động bằng cách thay thế các pixel bị hỏng bằng các pixel tương tự với các pixel lân cận, do đó, làm cho chúng không dễ thấy và giúp chúng hòa trộn tốt với nền.

Thuật toán Inpaint hoạt động theo nguyên tắc sau:

- Thuật toán sẽ tìm kiếm các vùng ảnh xung quanh vùng chứa đối tượng cần loại bỏ.
- Sau đó, thuật toán sẽ sử dụng một mô hình hồi quy để dự đoán các giá trị pixel ở vùng ảnh bị thiếu.
- Cuối cùng, thuật toán sẽ thay thế các giá trị pixel dự đoán cho các giá trị pixel thực tế ở vùng ảnh bị thiếu.

Bước 1: Tìm kiếm các vùng ảnh xung quanh vùng chứa đối tượng cần loại bỏ

Để tìm kiếm các vùng ảnh xung quanh vùng chứa đối tượng cần loại bỏ, thuật toán Inpaint sử dụng một hàm kiểm tra ranh giới. Hàm kiểm tra ranh giới này sẽ đánh giá mức độ giống nhau giữa các giá trị pixel ở hai vùng ảnh liền kề. Nếu mức độ giống nhau cao, thì hai vùng ảnh đó được coi là liền kề.

Bước 2: Sử dụng mô hình hồi quy để dự đoán các giá trị pixel ở vùng ảnh bị thiếu

Để dự đoán các giá trị pixel ở vùng ảnh bị thiếu, thuật toán Inpaint sử dụng một mô hình hồi quy. Mô hình hồi quy này được xây dựng dựa trên các giá trị pixel ở các vùng ảnh xung quanh vùng chứa đối tượng cần loại bỏ.

Bước 3: Thay thế các giá trị pixel dự đoán cho các giá trị pixel thực tế ở vùng ảnh bị thiếu

Sau khi dự đoán được các giá trị pixel ở vùng ảnh bị thiếu, thuật toán Inpaint sẽ thay thế các giá trị pixel thực tế ở vùng ảnh bị thiếu bằng các giá trị pixel dự đoán.

2.3.2.1 Ưu điểm

Ưu điểm của thuật toán Inpaint:

- Dễ sử dụng và nhanh chóng.
- Có thể loại bỏ các đối tượng có kích thước lớn và phức tạp.
- Có thể loại bỏ các đối tượng có màu sắc và độ tương phản khác nhau với các vùng ảnh xung quanh.

2.3.2.2 Nhược điểm

Nhược điểm của thuật toán Inpaint:

- Có thể gây ra hiện tượng răng cưa ở các cạnh của vùng ảnh bị thiếu.
- Có thể không loại bỏ hoàn toàn các đối tượng có màu sắc và độ tương phản khác nhau với các vùng ảnh xung quanh.

2.3.2.3 Ứng dụng

Thuật toán Inpaint với những ưu điểm trên có các ứng dụng phổ biến trong thực tế để loại bỏ các vết xước, vết ô, hoặc các đối tượng không mong muốn khỏi ảnh. Ví dụ, thuật toán Inpaint có thể được sử dụng để loại bỏ các đường dây điện, khách du lịch, các tòa nhà khỏi ảnh phong cảnh hoặc các văn bản khỏi hình ảnh.

2.3.3 *Bộ lọc ảnh – Filter*

Trong xử lý ảnh, Bộ lọc - Filter trong xử lý ảnh là một kỹ thuật được sử dụng để thay đổi các giá trị pixel trong một hình ảnh. Filter có thể được sử dụng để loại bỏ nhiễu, tăng cường các cạnh, thay đổi độ sáng và độ tương phản của hình ảnh, hoặc tạo ra các hiệu ứng đặc biệt. Filter là những phép biến đổi miền không gian. Dưới đây là các dạng filter thông dụng trong xử lý ảnh:

- **Lọc làm mờ (Blur filter):** Lọc làm mờ được sử dụng để làm mịn hình ảnh bằng cách giảm độ tương phản giữa các pixel. Lọc làm mờ có thể được sử dụng để loại bỏ nhiễu hoặc tạo ra hiệu ứng mờ.
- **Lọc làm sắc nét (Sharpen filter):** Lọc tăng cường cạnh được sử dụng để làm nổi bật các cạnh trong hình ảnh. Lọc tăng cường cạnh có thể được sử dụng để cải thiện độ sắc nét của hình ảnh hoặc để tìm các đối tượng trong hình ảnh.
- **Lọc phát hiện cạnh (Edge detection filter):** Nhận diện và làm nổi bật các ranh giới hoặc cạnh trong hình ảnh bằng cách sử dụng các thuật toán phát hiện cạnh như Sobel, Prewitt, Canny, v.v.
- **Lọc giảm nhiễu (Noise reduction filter):** Loại bỏ hoặc giảm thiểu nhiễu trong hình ảnh, giúp cải thiện chất lượng của nó.
- **Lọc hình thái học (Morphological filter):** Thường được sử dụng để thực hiện các phép toán như co, mở rộng, lấp đầy, xóa bỏ nhiễu hoặc kết hợp các khu vực trong hình ảnh.

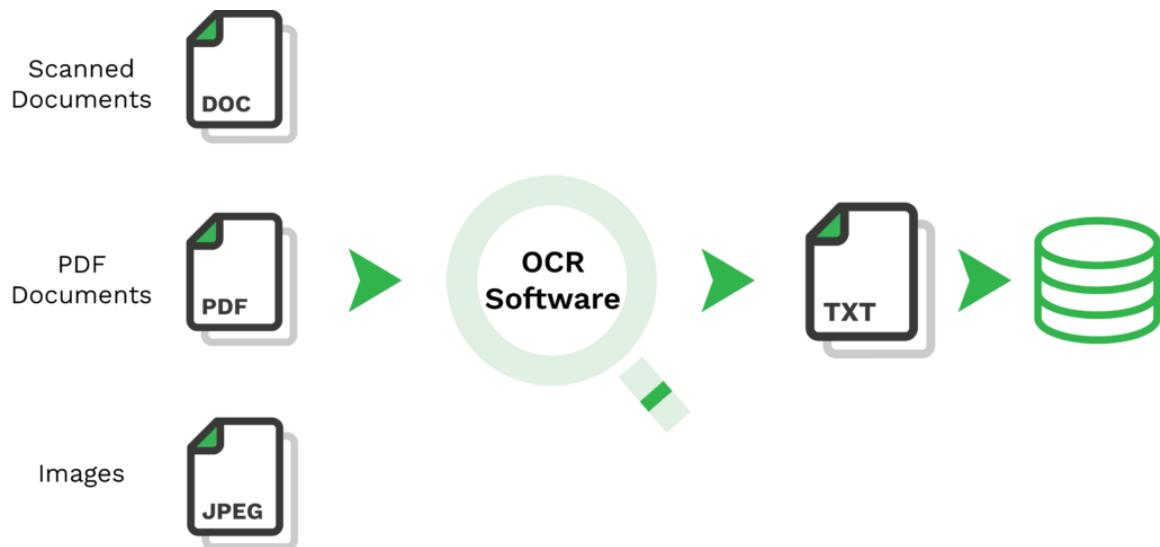
- Lọc màu (Color adjustment filter): Lọc màu được sử dụng để thay đổi màu sắc của hình ảnh. Lọc màu có thể được sử dụng để tạo ra các hiệu ứng đặc biệt, chẳng hạn như hiệu ứng đèn trăng hoặc hiệu ứng màu cổ điển.

Filter là một công cụ mạnh mẽ có thể được sử dụng để cải thiện chất lượng của hình ảnh hoặc để tạo ra các hiệu ứng đặc biệt. Filter có thể được áp dụng cho hình ảnh bằng cách sử dụng phần mềm xử lý ảnh. Có nhiều phần mềm xử lý ảnh khác nhau có sẵn, chẳng hạn như Adobe Photoshop, GIMP và Paint.NET. Công thư viện xử lý ảnh như OpenCV, MATLAB, Photoshop, và nhiều thư viện mã nguồn mở khác cung cấp nhiều loại filter và thuật toán xử lý ảnh cho phép người dùng thực hiện các thay đổi trên hình ảnh dễ dàng thông qua các filter này. Sử dụng filter đúng cách có thể cải thiện chất lượng hình ảnh, loại bỏ nhiễu và nâng cao các đặc điểm mong muốn.

Trong đồ án hiện tại, đề tài sẽ tiến hành sử dụng Lọc màu – Color adjustment filter với sự hỗ trợ của thư viện Pilgram Python để thực hiện xử lý và chỉnh sửa màu sắc cho hình ảnh.

2.4 Giới thiệu về OCR:

2.4.1 Giới thiệu OCR



Hình 2.4-1 Giới thiệu về OCR

OCR là viết tắt của Optical Character Recognition, có nghĩa là nhận dạng ký tự quang học. Đây là một công nghệ máy tính cho phép chuyển đổi hình ảnh văn bản thành định dạng văn bản mà máy có thể đọc được.

Quy trình OCR bao gồm các bước sau:

Chuẩn bị hình ảnh: Hình ảnh văn bản cần được chuẩn bị trước khi có thể được nhận dạng bởi OCR. Điều này có thể bao gồm việc loại bỏ các yếu tố không cần thiết, chẳng hạn như đường viền, nền và các ký hiệu.

Phân đoạn: Hình ảnh văn bản được phân đoạn thành các ký tự riêng lẻ. Điều này được thực hiện bằng cách sử dụng các thuật toán phân tích hình ảnh để xác định các cạnh và góc của các ký tự.

Phân loại: Mỗi ký tự được phân loại thành một ký tự cụ thể. Điều này được thực hiện bằng cách sử dụng một cơ sở dữ liệu các ký tự mẫu.

2.4.2 *Phân loại OCR*

Có hai loại chính của OCR:

OCR đánh máy: Loại OCR này được sử dụng để nhận dạng văn bản được đánh máy. Văn bản đánh máy thường có độ chính xác cao hơn văn bản viết tay.

OCR viết tay: Loại OCR này được sử dụng để nhận dạng văn bản viết tay. Văn bản viết tay thường có độ chính xác thấp hơn văn bản đánh máy.

2.4.3 *Ứng dụng OCR:*

OCR được sử dụng trong nhiều ứng dụng khác nhau, bao gồm:

Số hóa tài liệu: OCR được sử dụng để chuyển đổi các tài liệu giấy thành định dạng kỹ thuật số. Điều này giúp dễ dàng lưu trữ, truy cập và chia sẻ tài liệu.

Tự động hóa quy trình: OCR có thể được sử dụng để tự động hóa các quy trình, chẳng hạn như xử lý hóa đơn và nhập dữ liệu.

Tìm kiếm thông tin: OCR có thể được sử dụng để tìm kiếm thông tin trong các tài liệu văn bản.

Tăng khả năng tiếp cận: OCR có thể được sử dụng để tăng khả năng tiếp cận của thông tin cho người khiếm thị hoặc người có vấn đề về thị lực.

2.5 Một số thư viện sử dụng

2.5.1 WinUI 3 – .NET Framework 4.8

.NET là nền tảng phát triển phần mềm được phát triển bởi Microsoft. Cung cấp một tập hợp các công cụ, thư viện và framework để giúp các nhà phát triển tạo ra các ứng dụng web, ứng dụng máy tính, ứng dụng di động và các loại ứng dụng khác.

.NET – A unified platform



Hình 2.5-1 .NET Hỗ trợ đa nền tảng

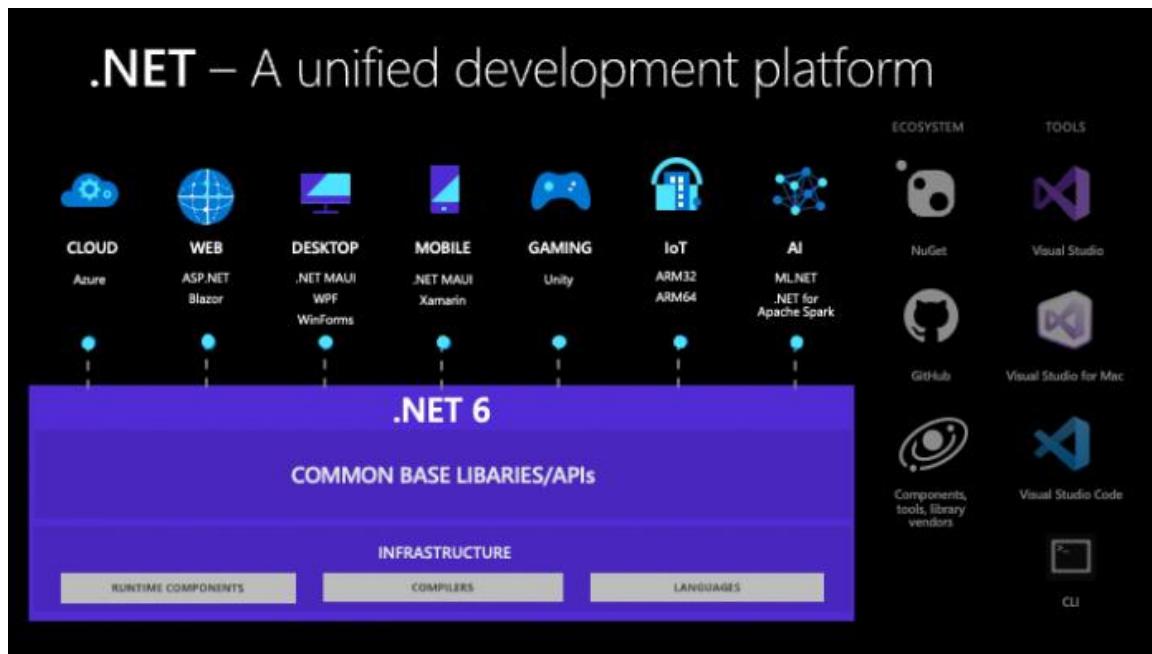
.NET được thiết kế để là một nền tảng đa nền tảng các ứng dụng được phát triển trên .NET có thể chạy trên nhiều nền tảng khác nhau, bao gồm Windows, macOS, Linux và Android.

Ngoài ra, .Net được cấu thành bởi các thành phần chính sau với sự hỗ trợ đa dạng cho các lập trình viên. Ngôn ngữ lập trình đã dạng bao gồm C#, Visual Basic, F#, C++ và Python. Tương ứng với từng ngôn ngữ lập trình, .Net còn hỗ trợ rất nhiều bộ thư viện cung cấp các chức năng và khả năng xây dựng cho các nhà phát triển. Bên cạnh đó, Framework do .Net hỗ trợ cung cấp các giải pháp và quy trình giúp nhà phát triển ứng dụng từ đầu đến khi ra được sản phẩm.

Về phần thiết kế giao diện, .NET hỗ trợ rất nhiều thư viện cho việc thiết kế giao diện người dùng trên các ứng dụng như: Windows desktop, UWP(Universal Windows Platforms), WinUI, MAUI... Trong đó WinUI là thư viện giúp xây dựng giao diện có sự kết hợp và kế thừa của Windows desktop và UWP. Cùng với sự kết hợp với Fluent Design System, một ngôn ngữ thiết kế hiện đại, trẻ trung mang lại trải nghiệm sâu sắc mà

đa chiều hơn bằng cách sử dụng các kỹ thuật thiết kế đa chiều mới. Vì vậy WinUI 3 đang là lựa chọn mới nhất cho thiết kế giao diện cho các ứng dụng Windows hiện nay.

Với sức mạnh là hệ điều hành Windows vô cùng phổ biến hiện nay, để tài lựa chọn sử dụng .NET Framework 4.8 để xây dựng ứng dụng trên nền tảng Windows. Giúp tạo sự quen thuộc trong sử dụng để có được hiệu quả sử dụng phần mềm nhanh nhất. Và sử dụng WinUI 3 để thiết kế giao diện ứng dụng.



Hình 2.5-2 .NET Framework 4.8 (.NET 6.0) – A unified development platform

2.5.1.1 Ưu điểm:

Tham vọng của Microsoft phát triển .NET trong việc cung cấp cho các nhà phát triển một nền tảng để giải quyết mọi loại vấn đề. .NET mang trong mình các ưu điểm nổi bật như:

Hỗ trợ đa ngôn ngữ: .NET là một nền tảng hỗ trợ đa ngôn ngữ nhờ vào CLR (The Common Language Runtime) giúp biên dịch các ngôn ngữ khác nhau về mã máy. Từ đó giúp phát triển kết hợp nhiều hơn một ngôn ngữ trong một ứng dụng.

Đa nền tảng: .NET có cung cấp một thành phần quan trọng là .NET Core. Thành phần này giúp tạo một môi trường để chạy các chương trình được xây dựng bằng .NET trên các nền tảng khác như Windows, macOS, Linux. Giúp tiết kiệm chi phí phát triển, tăng khả năng tiếp cận của ứng dụng và tăng tính linh hoạt.

Trình quản lý thư viện .Nuget: NuGet là trình quản lý thư viện chính thức cho nền tảng nhà phát triển .NET. NuGet lưu trữ một hệ sinh thái gói không lồ tại nuget.org và cung cấp các công cụ để giúp các nhà phát triển tạo, xuất bản và sử dụng các thư viện .NET.

Mã nguồn mở: .NET sử dụng giấy phép bản quyền sử dụng các loại phần mềm mã nguồn mở là MIT (the Massachusetts Institute of Technology). Điều này giúp .NET không ngừng được phát triển nhờ sự đóng góp của cộng đồng lập trình viên lớn.

Giao diện hiện đại: Sử dụng WinUI 3 với ngôn ngữ thiết kế Fluent Design tạo nên tính sáng tạo, linh hoạt và hiện đại trong thiết kế giao diện người dùng. Làm nâng cao tối đa trải nghiệm người dùng khi sử dụng ứng dụng.

2.5.1.2 Ứng dụng chính:

Các ứng dụng chính của WinUI 3 và .NET Framework 4.8:

- Sử dụng .NET Framework để thực hiện xây dựng kiến trúc phần mềm. Lập trình hướng đối tượng, mô hình sự kiện, thiết kế các hàm sự kiện và các hàm xử lý, xử lý đã luồng.
- Sử dụng WinUI thiết kế giao diện người dùng đầu vào, giúp nâng cao trải nghiệm người dùng. Thiết kế với các thành phần giao diện đơn giản, hiện đại và dễ sử dụng.
- Hiển thị kết quả xử lý các hướng xử lý khác nhau từ ảnh, văn bản, và có thể copy vào bộ nhớ đệm.
- Xuất các dạng hình ảnh và thông tin trực tiếp trên máy tính cá nhân.
- Dễ dàng cài đặt và sử dụng.

2.5.2 *OpenCVSharp*

OpenCV là một thư viện được xây dựng bằng ngôn ngữ lập trình C và C++. Ví dụ để sử dụng trong C#, hay trong .NET thì chúng ta cần phải đóng gói thư viện này và viết lại một wrapper. Điều này giúp chúng ta có thể sử dụng các tính năng thị giác máy của OpenCV dưới dạng các APIs.

2.5.2.1 Ưu điểm:

Các ưu điểm cơ bản của OpenCVSharp:

Tương tự OpenCV: Các chức năng của OpenCV tương đối hoàn thiện trong bản wrapper này. Các tính năng hoạt động tương đối ổn định

Sử dụng ngôn ngữ C#: Có thể lập trình bằng ngôn ngữ C#, giúp phát triển các ứng dụng thị giác máy tính trên Windows được dễ dàng hơn.

2.5.2.2 Ứng dụng chính:

Các ứng dụng chính của OpenCV Sharp:

- Thực hiện cá thao tác xử lý ảnh chính: Inpaint ảnh, thực hiện vẽ các mặt nạ cho việc Inpaint.
- Trích xuất ảnh dưới dạng bitmap: Có thể thử xử lý ảnh một cách tự nhiên trong hệ điều hành dưới dạng ảnh bitmap.
- Xử lý Inpaint cho ảnh với các mask được chọn.
- Xử lý có hiệu suất ổn định so với Python.

2.5.3 *Tesseract Python* và *IronOCR*

Nhận dạng ký tự quang học(OCR - Optical Character Recognition) là quá trình chuyển đổi một hình ảnh văn bản thành định dạng văn bản mà máy có thể đọc được. OCR hỗ trợ nhận dạng các dạng văn bản viết tay hoặc đánh máy. Trong đó văn bản viết tay có độ chính xác thấp hơn, khó nhận dạng hơn so với văn bản đánh máy.

Có rất nhiều thư viện hỗ trợ kỹ thuật OCR, được phát triển bởi các nhà phát triển độc lập, các tổ chức nghiên cứu và các công ty phần mềm. Một số thư viện phổ biến giúp nhận dạng các ký tự quang học như:

- Tesseract: Tesseract là một thư viện OCR mã nguồn mở được phát triển bởi Google. Tesseract là một thư viện mạnh mẽ và chính xác, có thể nhận dạng nhiều loại văn bản khác nhau.
- OpenCV: OpenCV là một thư viện xử lý hình ảnh mã nguồn mở do Intel nghiên cứu cung cấp các công cụ xử lý ảnh mạnh mẽ. OpenCV có thể được sử dụng để xây dựng các hệ thống OCR.

- Google Cloud Vision API: Google Cloud Vision API là một dịch vụ đám mây của Google cung cấp các chức năng nhận dạng hình ảnh, bao gồm cả OCR. Google Cloud Vision API có thể được sử dụng để nhận dạng văn bản từ hình ảnh trực tuyến hoặc hình ảnh được lưu trữ trên đám mây.

Trong phạm vi đề tài của đồ án, chúng tôi sử dụng thư viện pytesseract cho python, và IronOCR cho C# để thử nghiệm và xây dựng các tính năng của ứng dụng của đề tài.

2.5.3.1 Ưu điểm:

Ưu điểm chung của hai thư viện pytesseract và IronOCR:

Hỗ trợ đa dạng các chuẩn ảnh: Hỗ trợ các chuẩn ảnh phổ biến như .JPEG, .PNG, .BMP, .TIFF và các chuẩn ảnh khác.

Dễ dàng cài đặt: Đối với IronOCR không cần phải cài phần mềm Tesseract vào hệ thống để sử dụng mà chỉ cần cài gói thư viện từ Nuget là có thể sử dụng được.

Cú pháp đơn giản: Cú pháp của cả hai thư viện này tương đối ngắn gọn là có thể đưa vào xử lý với nhu cầu cơ bản.

2.5.3.2 Ứng dụng chính:

Các ứng dụng chính của Tesseract Python và IronOCR:

- Pytesseract: dùng để thực hiện thử nghiệm các tính năng cơ bản trên python, sau đó tiến hành xây dựng các luồng xử lý cơ bản trên đây.
- IronOCR: sử dụng để xây dựng kiến trúc xử lý ảnh riêng biệt cho ứng dụng trong C#.
- Hỗ trợ đa dạng định dạng ảnh: Thực hiện hỗ trợ xử lý các chuẩn ảnh khác nhau tương ứng với từng nhu cầu người dùng.
- Hỗ trợ nhiều ngôn ngữ: IronOCR được xây dựng riêng cho các ngôn ngữ cụ thể để tăng hiệu quả nhận dạng. Trong đồ án này thực hiện xây dựng cho ba ngôn ngữ chính là Việt, Anh, Trung.

2.5.4 *Pilgram Python*

Pilgram là một thư viện Python được sử dụng để làm việc với ảnh. Thư viện này cho phép bạn thực hiện các thao tác filter – thay đổi màu ảnh bằng cách sử dụng mã nguồn Python. Pilgram dựa trên thư viện CSSgram, một thư viện mã nguồn mở cho việc chỉnh sửa ảnh trên Instagram.

Với Pilgram, bạn có thể thực hiện các tác vụ như thay đổi màu cho ảnh với nhiều định dạng được xây dựng sẵn như: _1977, aden, brannan, brooklyn, clarendon, earlybird, gingham, hudson, inkwell, kelvin, lark, lofi, maven, mayfair, moon, nashville, perpetua, reyes, rise, slumber, stinson, toaster, valencia, walden, willow, xpro2.

2.5.4.1 Ứng dụng chính:

- Thực hiện hiện các điều chỉnh màu sắc cho ảnh theo nhu cầu sử dụng

2.5.5 Biểu thức chính quy – Regex

Biểu thức chính quy – (Regular expressions hay Regex) là một chuỗi miêu tả một bộ các chuỗi khác, theo những quy tắc cú pháp nhất định. Biểu thức chính quy thường được dùng trong các trình biên tập văn bản và các tiện ích tìm kiếm và xử lý văn bản dựa trên các mẫu được quy định. Nhiều ngôn ngữ lập trình cũng hỗ trợ biểu thức chính quy trong việc xử lý chuỗi, chẳng hạn như Perl có bộ máy mạnh mẽ để xử lý biểu thức chính quy được xây dựng trực tiếp trong cú pháp của chúng.

Các thao tác có thể thực hiện được của Regex trong xử lý văn bản như:

So khớp chuỗi (Matching): So khớp là quá trình tìm kiếm một chuỗi theo một mẫu cụ thể. Ví dụ, một biểu thức chính quy có thể được sử dụng để kiểm tra xem một chuỗi có chứa một từ cụ thể không. Ví dụ: '/[0-9]+/' sẽ so khớp với bất kỳ chuỗi nào có ít nhất một chữ số.

Tìm kiếm và thay thế (Search and Replace): Regex cho phép tìm kiếm chuỗi theo mẫu cụ thể và thay thế chúng bằng chuỗi khác. Ví dụ: '/apple/' sẽ tìm kiếm từ "apple" trong chuỗi và có thể thay thế nó bằng "orange".

Thao tác với ký tự đặc biệt (Special Characters): Regex sử dụng các ký tự đặc biệt để biểu thị các mẫu như ký tự đầu dòng, ký tự kết thúc dòng, các nhóm ký tự, ký tự đại diện cho ký tự trắng, ký tự số,. Ví dụ: '^' biểu thị ký tự đầu dòng, '\$' biểu thị ký tự kết thúc dòng, '.' biểu thị bất kỳ ký tự nào, '\d' biểu thị ký tự số.

Nhóm và phân đoạn (Grouping and Capturing): Regex cho phép nhóm các mẫu để xử lý chúng cùng nhau và thực hiện việc ghi nhớ các phần của chuỗi phù hợp với mẫu. Ví dụ: `'/(abc)+/'` sẽ so khớp với chuỗi "abcabcabc" hoặc "abc", `'/(\d{3})-(\d{4})/'` sẽ so khớp với số điện thoại trong định dạng "123-4567".

Bắt đầu và kết thúc (Anchors): Regex cung cấp các ký tự để chỉ ra vị trí bắt đầu và kết thúc của chuỗi hoặc dòng. Ví dụ: `'^regex'` sẽ so khớp với chuỗi bắt đầu bằng "regex", `'regex$'` sẽ so khớp với chuỗi kết thúc bằng "regex".

Quy tắc về lựa chọn (Quantifiers): Regex cho phép xác định số lượng xuất hiện của một mẫu. Ví dụ: `'a*/'` sẽ so khớp với chuỗi "aaaa" hoặc không có "a" nào, `'a+/'` sẽ so khớp với ít nhất một "a".

2.5.5.1 Ứng dụng chính:

- Hỗ trợ trích xuất thông tin theo các mẫu thông dụng như: email, số điện thoại, số tiền, ngày tháng năm, một dãy số bất kỳ, một chuỗi bắt đầu bằng một vài kí tự cụ thể.

CHƯƠNG 3. KẾT QUẢ VÀ PHÂN TÍCH

3.1 Phương pháp tiếp cận

3.1.1 *Thử nghiệm trên Python*

3.1.1.1 Thực hiện Inpaint Image:

Quá trình thực hiện triển khai thuật toán Inpaint trên Python có 2 hình thức tiếp cận chính dùng OpenCV và InpaintAnything, thực hiện các bước sau:

Bước 1: Tiến hành thu thập dữ liệu ảnh thử nghiệm cho thuật toán Inpaint phân làm 3 cấp độ khác nhau:

- Cơ bản (Simple): Đôi tượng thực hiện Inpaint đơn giản và kích thước nhỏ, nền có cấu trúc đơn giản tương đối đồng nhất.
- Trung bình (Intermediate): Đôi tượng thực hiện Inpaint kích thước trung bình, nền có cấu trúc tương đối đồng nhất.

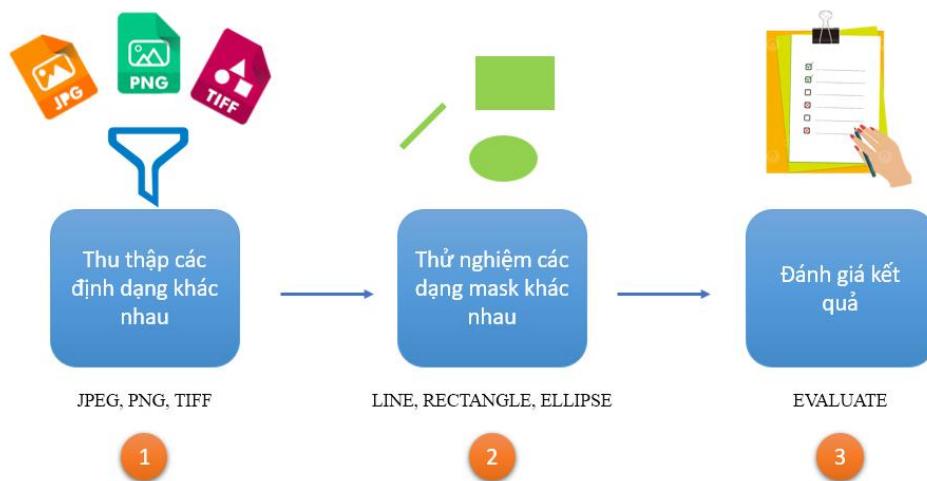
- Phức tạp (Complex): Đối tượng thực hiện Inpaint kích thước trung bình hoặc lớn, nền có cấu trúc phức tạp.

Bước 2: Thực hiện lần lượt 2 quá trình tiếp cận khác nhau:

- OpenCV: Thuật toán Inpaint trong OpenCV quy trình sau:
 - Thực hiện tải ảnh xử lý vào.
 - Tạo mặt nạ (mask) bằng các hình dạng khác nhau gồm: hình chữ nhật, hình elip, và các điểm liên tục trên hình ảnh có bán kính nhỏ.
 - Thực hiện quá trình Inpaint ảnh từ ảnh đầu vào và mặt nạ vừa tạo.
 - Lưu kết quả vừa thực hiện.
- Inpaint Anything: Thuật toán Inpaint Anything quy trình sau:
 - Thực hiện tải ảnh xử lý vào.
 - Thực hiện đánh dấu điểm chọn vào đối tượng so với ảnh đầu vào để xác định đối tượng trong hình ảnh.
 - Thực hiện Inpaint Anything để remove vật thể khỏi bức ảnh một cách hiệu quả với ngữ cảnh.
 - Lưu kết quả vừa thực hiện.

Bước 3: Thực hiện so sánh kết quả, đánh giá và nhận xét hai thuật toán xử lý trên.

Bước 4: Thực hiện viết chương trình Python cho thuật toán, sử dụng cho việc gọi chương trình Python thông qua Terminal trong C Sharp.



Hình 3.1-1 Flow xử lý Inpaint trong Python

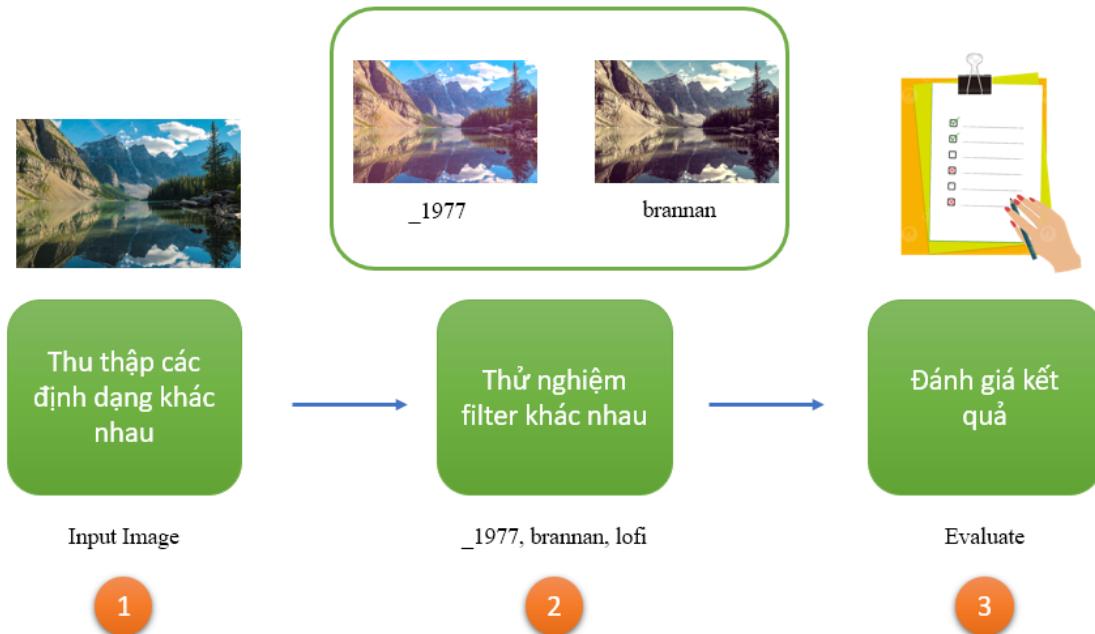
3.1.1.2 Filter – Chính sửa màu ảnh

Tiến hành sử dụng thư viện Pilgram trong Python để thực hiện thay đổi màu ảnh

Bước 1: Thu thập dữ liệu từ các định dạng ảnh khác nhau như: JPEG, PNG, TIFF, WEBP, BMP, SVG, GIFF. Thử nghiệm mức độ hỗ trợ của thư viện Pilgram.

Bước 2: Thử nghiệm thực nghiệm các loại Filter khác nhau có các ảnh.

Bước 3: Đánh giá thời gian xử lý và kết quả của ảnh trước và sau khi xử lý.



Hình 3.1-2 Flow xử lý chỉnh sửa ảnh trong Python

3.1.1.3 OCR

Tiến hành trích xuất văn bản dùng kĩ thuật OCR trong Python thực hiện theo các bước sau:

Bước 1: Tiến hành thu thập dữ liệu ảnh thử nghiệm cho thuật toán OCR phân làm 3 cấp độ khác nhau:

- Cơ bản (Simple): Chữ đen và nền trắng (có độ tương phản tốt), không có ảnh hưởng nhiều bởi ngũ cản trong ảnh.
- Trung bình (Intermediate): Chữ và nền có độ tương phản tốt, tuy nhiên có ảnh hưởng với yếu tố ngũ cản
- Phức tạp (Complex): Chữ và nền không phân biệt rõ, số lượng chi tiết nhiều và không theo cấu trúc cố định, có yếu tố ngũ cản.

Bước 2: Thực hiện lần lượt 2 quá trình tiếp cận khác nhau:

- Pytesseract: Thuật toán OCR trong Tesseract Python quy trình sau: [Link](#), [Install Pytesseract](#) ; [Tesseract Wraper](#)
 - Thực hiện tải ảnh xử lý vào.

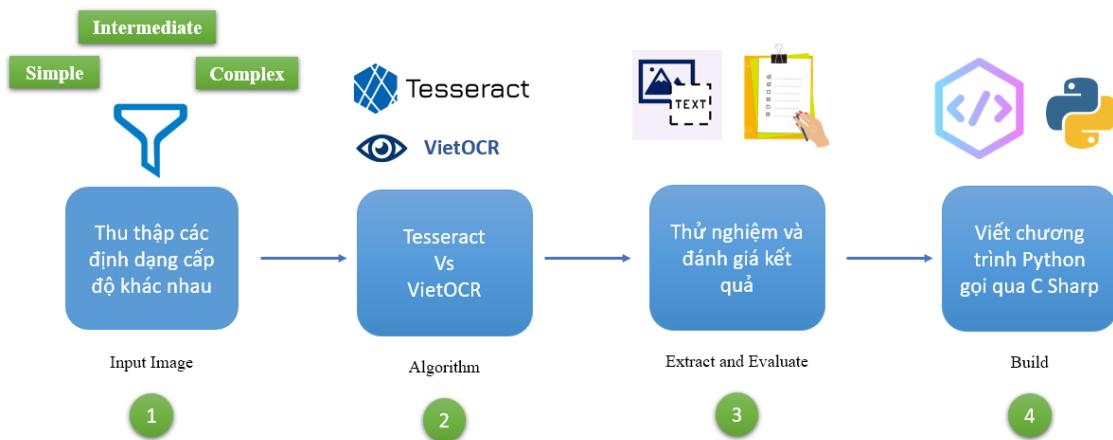
- Thực hiện quá trình trích xuất thông tin từ ảnh và lưu trữ. Hình thức sử dụng là toàn bộ ảnh và từng vùng ảnh.
 - Thực hiện với các ngôn ngữ khác nhau: tiếng Việt, tiếng Anh, tiếng Trung.

○ VietOCR: Thuật toán OCR được thực hiện với Model transformer của VietOCR quy trình sau:

 - Thực hiện tải ảnh xử lý vào.
 - Thực hiện quá trình trích xuất thông tin từ ảnh. Hình thức sử dụng là toàn bộ ảnh và từng vùng ảnh.
 - Thực hiện với các ngôn ngữ: tiếng Việt, tiếng Anh

Bước 3: Thực hiện so sánh kết quả, đánh giá và nhận xét hai thuật toán xử lý trên.

Bước 4: Thực hiện viết chương trình Python cho thuật toán, sử dụng cho việc gọi chương trình Python thông qua Terminal trong C Sharp.



Hình 3.1-3 Flow trích xuất thông tin từ ảnh

3.1.2 Thủ nghiệm trên C#:

Tiến hành thực hiện xây dựng ứng dụng trên C# bằng .NET Framework 4.8 và thiết kế giao diện bằng WinUI 3, thực hiện các bước xử lý sau.

3.1.2.1 Thực hiện Inpaint Image trên C#:

Qua trình thực hiện Inpaint Image trong C# tiến hành các bước sau:

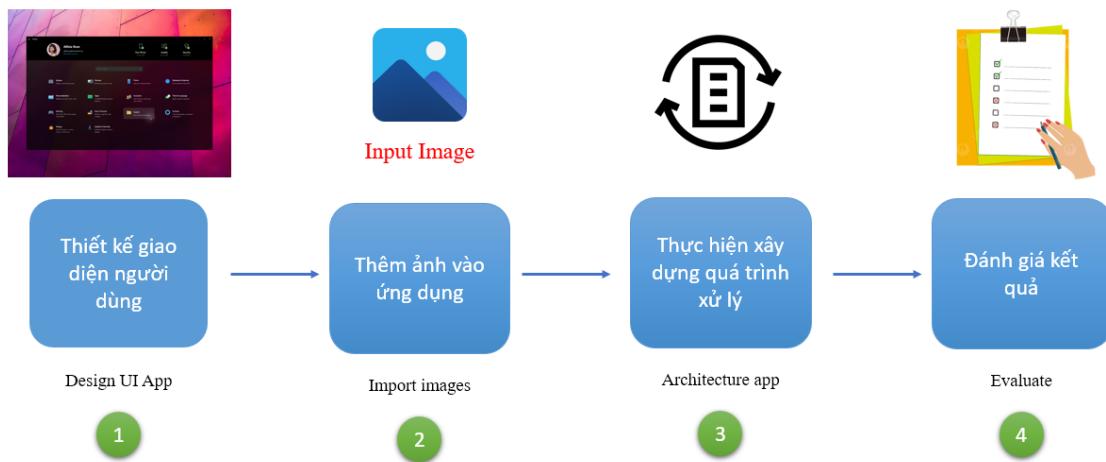
Bước 1: Thực hiện thiết kế giao diện người dùng sử dụng bao gồm: Danh sách ảnh sử dụng ListView, từng ảnh đơn lẻ được hiển thị bằng Canvas, Thiết kế các bảng chọn mặt nạ thông dụng đường thẳng có độ dày, hình chữ nhật, hình elip.

Bước 2: Xây dựng quá trình thêm ảnh vào ứng dụng kết hợp với xử lý đa luồng – Multi Threads. Giúp việc thêm ảnh vào ứng dụng không bị khóa giao diện người dùng, tránh được được tình trạng ứng dụng không thể tương tác trong khoảng thời gian nhất định

Bước 3: Thực hiện phần xử lý lỗi:

- Thực hiện nhận mặt nạ xử lý Inpaint vào trước bằng thao tác trên màn hình, sau đó chuyển đổi và lưu trữ dưới dạng bitmap Image mà OpenCVSharp hỗ trợ riêng bằng cách lấy tọa độ của các điểm được vẽ trên Canvas.
- Thực hiện Inpaint Image ngay sau khi vẽ xong và tiến hành lưu trữ các ảnh được xử lý dưới dạng một danh sách các ảnh thuận tiện cho việc thực hiện các thao tác như quay lại (Undo), tiến tới (Redo) và lưu ảnh với kích thước khác nhau về sau:
- Xây dựng tính năng Undo, Redo: Thực hiện các thao tác quay lại, thuận tiện hơn cho việc chỉnh sửa ảnh, giúp các thao tác có thể tùy chỉnh dễ dàng.
- Thực hiện xây dựng phần lưu trữ ảnh xuống dưới dạng file khác, lúc này sẽ thuận tiện hơn cho việc lưu trữ ảnh sau khi xử lý.

Bước 4: Tiến hành đánh giá các bước xử lý và cải thiện thực hiện Inpaint ảnh.



Hình 3.1-4 Flow xử lý Inpaint trong ứng dụng C#

3.1.2.2 Filter – Chính sửa màu ảnh

Tiến hành sử dụng thư viện Pilgram trong Python để thực hiện thay đổi màu ảnh, sau đó chúng ta thực hiện gọi các lệnh này qua Command trong C#. Thực hiện các bước tiến hành sau:

Bước 1: Thực hiện lấy dữ liệu ảnh bằng đường dẫn ảnh được thêm vào ứng dụng

Bước 2: Thực hiện xây dựng bảng chọn với các dạng filter chỉnh sửa màu khác nhau của ảnh.

Bước 3: Sau khi nhận được filter được chọn, tiến hành gọi lệnh thực thi qua code Python trong dự án để tiến hành thực hiện bước điều chỉnh màu cho ảnh.

Bước 4: Tiến hành lưu ảnh dưới dạng tạm, lưu các thông tin của ảnh gốc.

Bước 5: Sau đó tiến hành load lại ảnh vào trong ứng dụng cho người dùng xem.

Bước 6: Tiến hành đánh giá kết quả xử lý của ảnh.

3.1.2.3 OCR – IronOCR

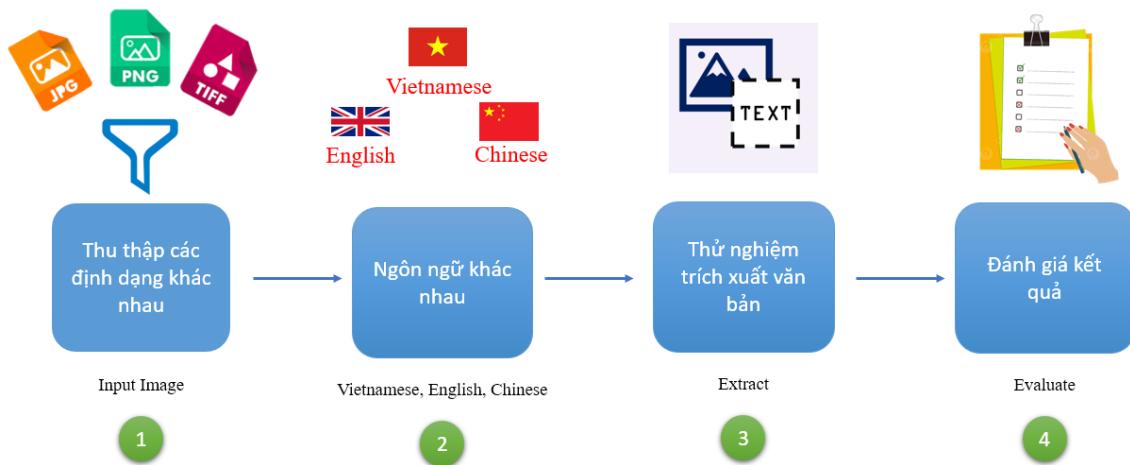
Tiến hành trích xuất văn bản dùng kỹ thuật OCR trong Python qua thư viện pytesseract thực hiện theo các bước sau:

Bước 1: Thực hiện lấy dữ liệu ảnh bằng đường dẫn ảnh được thêm vào ứng dụng

Bước 2: Thực hiện lựa chọn ngôn ngữ được thực hiện: tiếng Việt, tiếng Anh, tiếng Trung.

Bước 3: Thủ nghiệm trích xuất văn bản từ ảnh.

Bước 4: Đánh giá kết quả của ảnh trước và sau khi xử lý.



Hình 3.1-5 Flow trích xuất thông tin từ ảnh trong C Sharp

3.2 Kết quả và phân tích

3.2.1 Thực hiện Inpaint trong Python

Quá trình thực hiện triển khai thuật toán Inpaint trên Python có 2 hình thức tiếp cận chính dùng OpenCV và InpaintAnything đạt được kết quả sau:

Bước 1: Tiến hành thu thập dữ liệu ảnh thử nghiệm cho thuật toán Inpaint phân làm 3 cấp độ khác nhau:

- Cơ bản (Simple): Đối tượng thực hiện Inpaint đơn giản và kích thước nhỏ, nền có cấu trúc đơn giản tương đối đồng nhất.

Simple

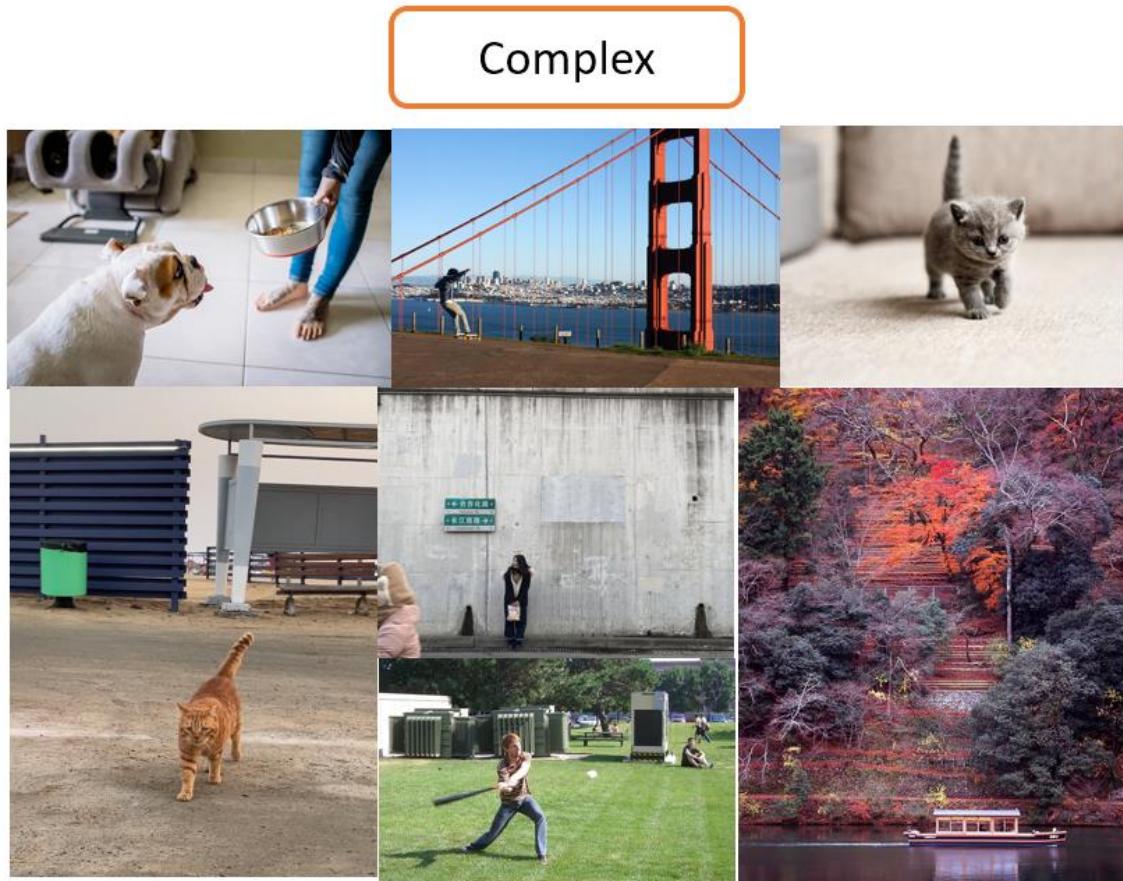


- Trung bình (Intermediate): Đối tượng thực hiện Inpaint kích thước trung bình, nền có cấu trúc tương đối đồng nhất.

Intermediate

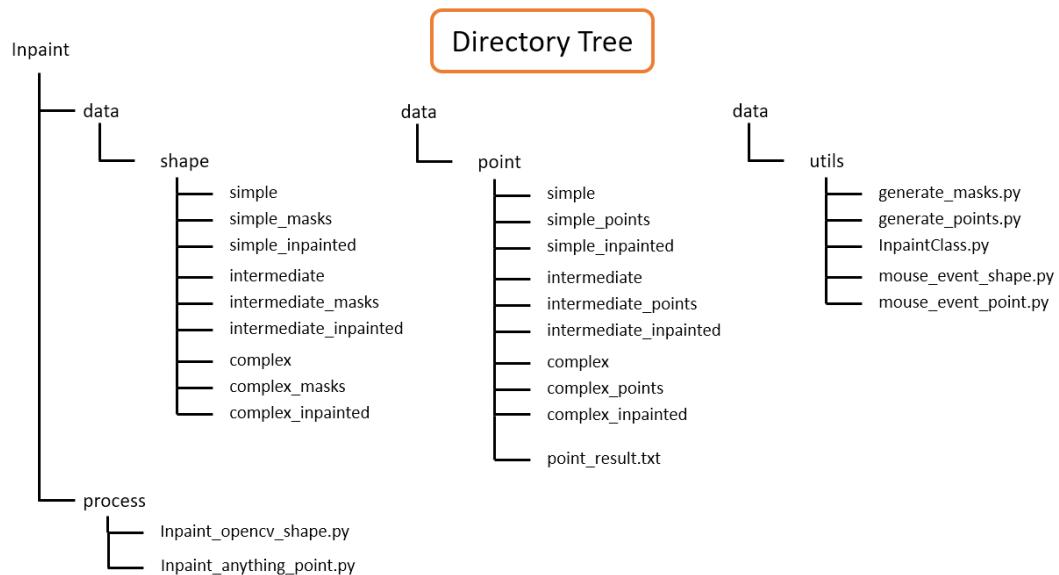


- Phức tạp (Complex): Đối tượng thực hiện Inpaint kích thước trung bình hoặc lớn, nền có cấu trúc phức tạp.



Bước 2: Thực hiện lần lượt 2 quá trình tiếp cận khác nhau:

Trước hết tổ chức cây thư mục sau:



Trong đó:

- ‘data’: Chứa dữ liệu đầu vào xử lý và kết quả xử lý bao gồm masks, point, inpainted theo các mức độ phức tạp khác nhau. Trong hình thức tiếp cận point, ‘point_result.txt’ chứa dữ liệu cho phần xử lý inpaint có cấu trúc như sau:

'inputPath | outputFolder | coordinate (x, y)

Với x là tọa độ trục đứng, y là tọa độ trục ngang có đơn vị nhỏ nhất là 1 pixel.

```

1  point\simple\simple_0.jpg|point\simple_inpainted|(280, 108)
2  point\simple\simple_1.jpg|point\simple_inpainted|(410, 778)
3  point\simple\simple_2.jpg|point\simple_inpainted|(522, 716)
4  point\simple\simple_3.jpg|point\simple_inpainted|(352, 623)
5  point\simple\simple_4.jpg|point\simple_inpainted|(45, 48)
6  point\simple\simple_5.jpg|point\simple_inpainted|(691, 214)

```

- ‘utils’: Chứa các chương trình được Python hỗ trợ quá trình xử lý Inpaint
 - ‘generate_masks.py’: Thực hiện tạo mask cho tương ứng từng ảnh, sau đó kết quả sẽ được lưu vào folder tương ứng với cú pháp sau:

f'shape\{level\}_masks\{level\}_{index}.jpg'

Với:

- ‘level’: bao gồm các giá trị {‘simple’, ‘intermediate’, ‘complex’}
- ‘index’: Số thứ tự của ảnh tương ứng với ảnh đầu vào.
- ‘generate_points.py’: Thực hiện xác định point, vị trí của đối tượng thực hiện inpaint cho tương ứng từng ảnh, sau đó kết quả sẽ được ghi vào ‘point_result.txt’
- ‘inpaintClass.py’: Để thuận tiện cho quá trình xử lý, đề tài thực hiện tạo các class để tổ chức có thê thông và xử lý rõ ràng theo lập trình hướng đối tượng như sau:
 - “Mode”: Đây là kiểu dữ liệu enum, gồm ba chế độ: RECTANGLE, ELIP, POLYGON
 - “Point”: Một điểm trên ảnh với tọa độ là x, y.
 - “Shape”: gồm hai tham số, mode lưu trữ dạng hình của mask, param lưu trữ tập hợp các điểm dùng để vẽ một hình.

- “Mask”: Mỗi đuôi tượng lưu trữ một mask, gồm rất nhiều hình dạng(Shape) khác nhau. Có hàm thực hiện tạo mask sau khi tạo đối tượng, giúp tạo và lưu mask ra ngoài.

Sau đó chúng ta thực hiện 2 bước tiếp cận như sau:

- OpenCV: Thuật toán Inpaint trong OpenCV quy trình sau, thực hiện chạy chương trình của file ‘generate_masks.py’:
 - Thực hiện tải ảnh xử lý vào.

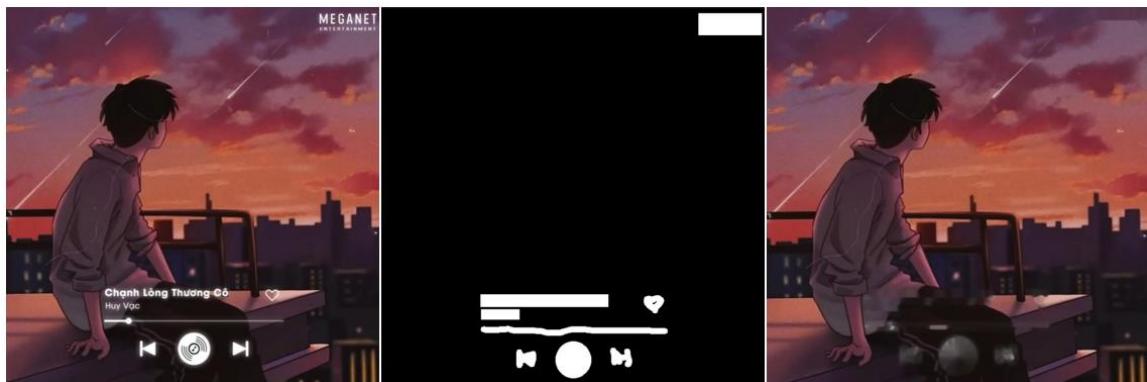
```
images = [  
    r'shape\simple\simple_0.jpg',  
    r'shape\simple\simple_1.jpg',  
    r'shape\simple\simple_2.jpg',  
    r'shape\simple\simple_3.jpg',  
    r'shape\simple\simple_4.jpg',  
    r'shape\simple\simple_5.jpg',  
    r'shape\intermediate\intermediate_0.jpg',  
    r'shape\intermediate\intermediate_1.jpg',  
    r'shape\intermediate\intermediate_2.jpg',  
    r'shape\intermediate\intermediate_3.jpg',  
    r'shape\intermediate\intermediate_4.jpg',  
    r'shape\intermediate\intermediate_5.jpg',  
    r'shape\intermediate\intermediate_6.jpg',  
    r'shape\intermediate\intermediate_7.jpg',  
    r'shape\intermediate\intermediate_8.jpg',  
    r'shape\complex\complex_0.jpg',  
    r'shape\complex\complex_1.jpg',  
    r'shape\complex\complex_2.jpg',  
    r'shape\complex\complex_3.jpg',  
    r'shape\complex\complex_4.jpg',  
    r'shape\complex\complex_5.jpg',  
    r'shape\complex\complex_6.jpg',  
]
```

- Tạo mặt nạ (mask) bằng các hình dạng khác nhau gồm: hình chữ nhật, hình elip, và các điểm liên tục trên hình ảnh có bán kính nhỏ.



Quá trình thực hiện tạo mặt nạ (mask) được thực hiện bằng tương tác người dùng, với các phím sau: phím ‘r’ thực hiện vẽ hình chữ nhật với 2 điểm là góc trái trên và góc phải dưới ; phím ‘e’ thực hiện vẽ hình elip với 2 điểm là góc trái trên và góc phải dưới sau đó tự động tính các thông số còn lại của elip; phím ‘p’ thực hiện vẽ các điểm hình tròn liên tục với đường kính nhỏ; phím ‘n’ thực hiện nhảy đến hình ảnh kế tiếp, và phím ‘q’ dùng để kết thúc quá trình tạo mask.

- Thực hiện quá trình Inpaint ảnh từ ảnh đầu vào và mặt nạ vừa tạo



- (Kết quả trên là thực hiện với cấp độ đơn giản nhất)

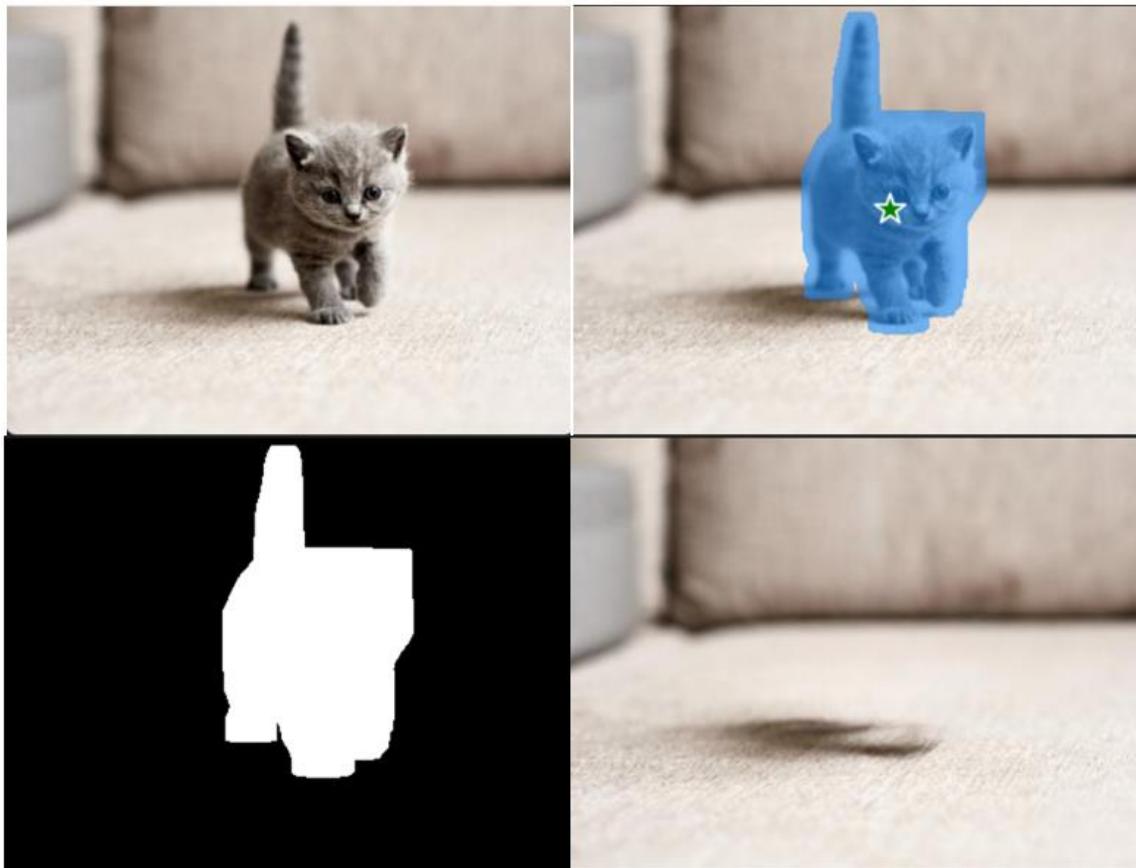
- Lưu kết quả vừa thực hiện.

- InpaintAnything: Thuật toán Inpaint Anything quy trình sau:

- Thực hiện tải ảnh xử lý vào như trên.
- Thực hiện đánh dấu điểm chọn vào đối tượng so với ảnh đầu vào để xác định đối tượng trong hình ảnh.



- Thực hiện Inpaint Anything để remove vật thể khỏi bức ảnh một cách hiệu quả với ngũ canh.

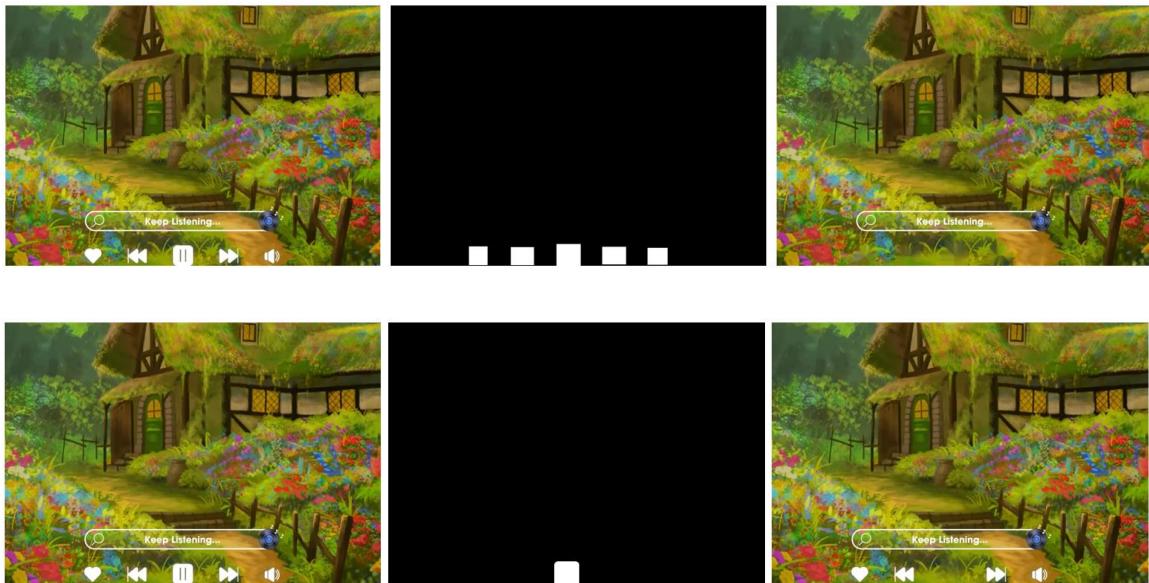


- Lưu kết quả vừa thực hiện.

Bước 3: So sánh kết quả, đánh giá và nhận xét hai thuật toán xử lý trên.

Thực hiện so sánh với mức độ phức tạp khác nhau để đánh giá thuật toán:

- Cơ bản (Simple)



Cả hai thuật toán thực hiện quá trình Inpaint để mang lại hiệu quả nhất định. Đối với thuật toán dùng OpenCV thì thực hiện xóa vật thể quá tốt trong tình huống đơn giản. Trong khi đó thuật toán Inpaint Anything cũng có thể làm điều tương tự, tuy nhiên lại chỉ có thể thực hiện chọn một đối tượng trong một phần xử lý mà chưa thực hiện nhận dạng được các đối tượng lân cận tốt.

Đánh giá: Thuận toán dùng OpenCV phù hợp hơn trong việc xử lý các chi tiết nhỏ, có độ tương phản tốt với nền và nền có cấu trúc đồng nhất. Còn thuật toán Inpaint Anything thì chưa phát huy tốt trong trường hợp các chi tiết đối tượng là nhỏ và thiết liên kết với nhau.

- Trung bình (Intermediate)

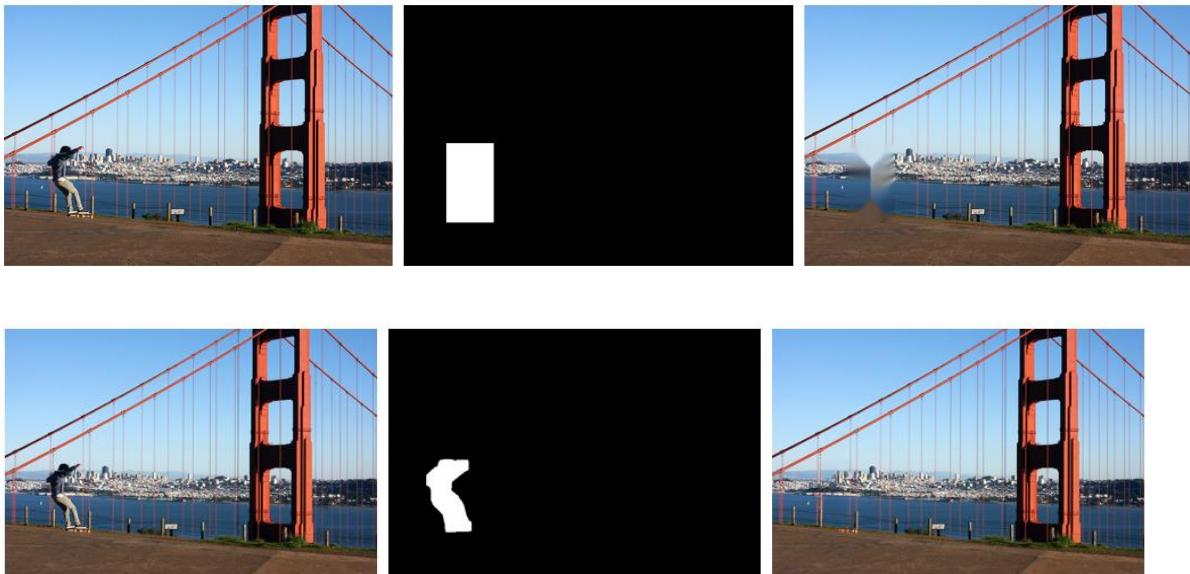




Cả hai thuật toán thực hiện quá trình Inpaint đều mang lại hiệu quả nhất định. Đối với thuật toán dùng OpenCV thì thực hiện xóa vật không quá tốt trong mức độ trung bình. Trong khi đó thuật toán Inpaint Anything xử lý rất tốt trong trường hợp đơn vật thể, thực hiện phân đoạn ảnh để tạo mask hiệu quả, và kết quả của thuật toán Inpaint Anything là vượt trội hơn hẳn.

Đánh giá: Thuật toán dùng OpenCV gặp khó khăn trong việc xử lý các chi tiết trung bình, có độ tương kém với nền và nền có cấu trúc không đồng nhất. Còn thuật toán Inpaint Anything thì phát huy tốt trong trường hợp các chi tiết đối tượng là trung bình và phân biệt rõ với ngữ cảnh. Bên cạnh đó chất lượng xử lý Inpaint Anything là vô cùng vượt trội.

- Phức tạp (Complex)



Cả hai thuật toán thực hiện quá trình Inpaint đều mang lại hiệu quả nhất định. Đối với thuật toán dùng OpenCV thì thực hiện xóa vật không quá tốt trong mức độ phức tạp. Trong khi đó thuật toán Inpaint Anything xử lý rất tốt trong trường hợp đơn vật thể, thực

hiện phân đoạn ảnh để tạo mask hiệu quả, và kết quả của thuật toán Inpaint Anything là vượt trội hơn hẳn.

Đánh giá: Thuật toán dùng OpenCV gặp khó khăn trong việc xử lý các chi tiết phức tạp, có độ tương kẽm với nền và nền có cấu trúc không đồng nhất. Còn thuật toán Inpaint Anything thì phát huy tốt trong trường hợp các chi tiết đối tượng là trung bình và phân biệt rõ với ngũ cành. Bên cạnh đó chất lượng xử lý Inpaint Anything là vô cùng vượt trội.

Và cuối cùng hai thuật toán đều có các ưu nhược điểm khác nhau, sau đây là đánh giá về 2 thuật toán;

- Inpaint – OpenCV:
 - Ưu điểm: Thuật tiện cho việc xử lý nhiều đối tượng cùng một lúc, có thể thực hiện được các dạng mask khác nhau giúp quá trình xử lý vô cùng linh hoạt, bên cạnh đó kết quả xử lý nhanh. Đạt hiệu quả nhất với các ảnh ở cấp độ cơ bản (Simple) với chi tiết nhỏ, độ tương phản cao và cấu trúc nền là đồng nhất
 - Nhược điểm: Cần phải thực hiện xác định mask một cách chính xác và vừa đủ, không thể thực hiện Inpaint với các chi tiết quá phức tạp và nền có cấu trúc không đồng nhất. Kết quả thực hiện có thể ảnh hưởng nhiều đến ảnh sau xử lý khi chi tiết là lớn so với bức ảnh.
- Inpaint Anything:
 - Ưu điểm: Xử lý có độ chính xác cao đối với cấp độ khác nhau từ cơ bản đến phức tạp. Hiệu quả ảnh sau xử lý là rất tốt so với thuật toán Inpaint OpenCV. Thực hiện nhận dạng được các đối tượng có tích chất tương đồng và tạo mask phù hợp với ngũ cành nhất.
 - Nhược điểm: Thực hiện chỉ một đối tượng ở một thời điểm, quá trình thực hiện tương đối lâu hơn. Tránh các đối tượng xung quanh có tính chất khá tương đồng với đối tượng được chọn có thể khiến đối tượng bị nhầm lẫn

Bước 4: Thực hiện viết chương trình Python cho thuật toán, sử dụng cho việc gọi chương trình Python thông qua Terminal trong C Sharp.

3.2.2 Thực hiện Filter trong Python

Thực hiện Filter trong Python thực hiện các bước sau:

Bước 1: Tiến hành thu thập dữ liệu ảnh ở các định dạng khác nhau:



Thực hiện filter trên các chuẩn ảnh khác nhau để xem thư viện hỗ trợ chuẩn ảnh nào, Dùng filter Kelvin để thực hiện lọc màu ảnh để xem kết quả mang lại.



Dựa vào kết quả trên, thư viện Pilgram thực hiện tốt đối với các chuẩn ảnh như: JPEG, PNG, TIFF, WEBP, BMP. Và chưa thực hiện tốt đối với chuẩn ảnh GIFF khi không đảm bảo được chi tiết ảnh động của chuẩn ảnh này. Cuối cùng thư viện không hỗ trợ chuẩn ảnh SVG.

Bước 2: Thực hiện xử lý với các ảnh đầu vào khác nhau với các filter sau:

- _1977

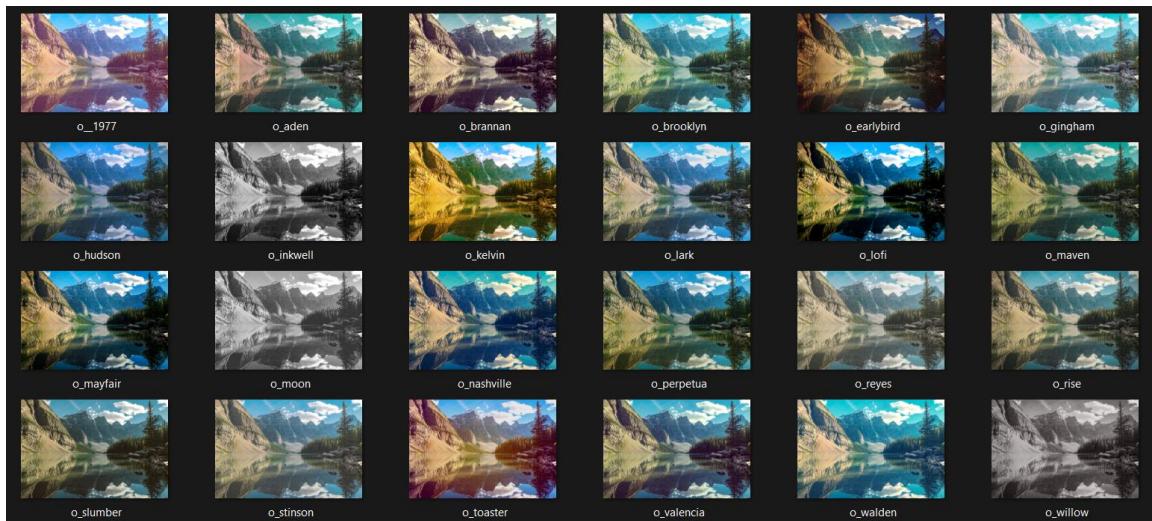
- aden
- brannan
- brooklyn
- clarendon
- earlybird
- gingham
- hudson
- inkwell
- kelvin
- lark
- lofi
- maven
- mayfair
- moon
- nashville
- perpetua
- reyes
- rise
- slumber
- stinson
- toaster
- valencia
- walden
- willow
- xpro2

Thực hiện các thuật toán Filter với ảnh đầu vào là các chuẩn ảnh trên với đầu vào là ảnh JPEG, một chuẩn ảnh phổ biến hiện nay.



Hình 3.2-1 Ảnh đầu vào Filter thực nghiệm trên Python

Sau quá trình thực hiện chạy code tự động, kết quả thử nghiệm chúng ta được kết quả sau:

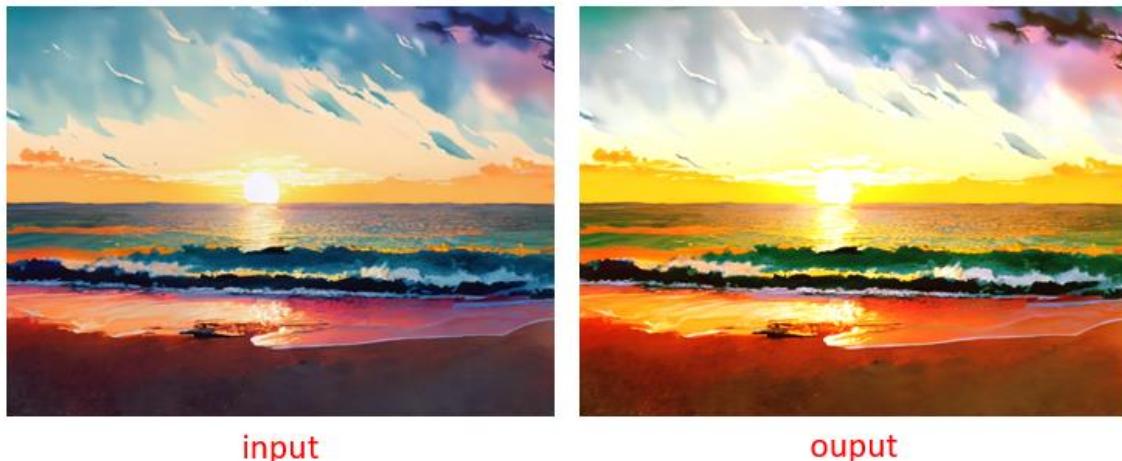


Hình 3.2-2 Đầu ra của ảnh sau chỉnh sửa màu trong Python

Việc thực hiện filter tất cả các dạng cho ảnh thực hiện tốt, và đạt được những màu sắc dạng.

Bước 3: Tiến hành đánh giá thời gian xử lý và kết quả thuật toán.

Thực hiện xử lý ảnh với chuẩn ảnh BMP, đạt kết quả như sau:



```
(filter) D:\Workspace\University\Report_DATN\Report_processes\Filter\data>python filter_multi_types.py  
DONE!  
⌚ Start at: 2023-12-12 10:53:33.414827  
⌚ End at: 2023-12-12 10:53:33.448327  
⌚ Duration: 0:00:00.033500
```

Time = 0.034s/img

Thời gian đo đạt được cho việc xử lý filter của một bức ảnh rơi vào khoảng $0.034s/image$ cho một bức ảnh. Với thời gian xử lý và kết quả như trên là lý tưởng cho việc đáp ứng thời gian thực của ứng dụng.

3.2.3 Thực hiện OCR trong Python

Tiến hành trích xuất văn bản dùng kỹ thuật OCR trong Python thực hiện theo các bước sau:

Bước 1: Tiến hành thu thập dữ liệu ảnh thử nghiệm cho thuật toán OCR phân làm 3 cấp độ khác nhau:

- Cơ bản (Simple):

[1] Computer Organization and Architecture 8th Edition, William Stallings,2011
 [2] Bài giảng kiến trúc máy tính – Viện CNTT-DH BKHN
 [3] Andrew S. Tanenbaum, Structured Computer Organization, 5rd Edition, Prentice- Hall International Edition, 2006.
 [4] Các nguồn tham khảo khác từ internet

Kiến trúc tập lệnh (Instruction Set Architecture): bao gồm: tập lệnh, biểu diễn dữ liệu, các cơ chế vào ra, kỹ thuật đánh địa chỉ
 • Tập lệnh: tập hợp các chuỗi số nhị phân mã hóa cho các thao tác mà máy tính có thể thực hiện
 • Các kiểu dữ liệu: các kiểu dữ liệu mà máy tính có thể xử lý

¹Machine Learning Department, Carnegie Mellon University
²Department of Computer Science, Princeton University
 agu@cs.cmu.edu, tri@tridao.me

simple_0

simple_1

simple_2

Every table is broken up into smaller entities called fields. The fields in the Customers table consist of CustomerID, CustomerName, ContactName, Address, City, PostalCode and Country. A field is a column in a table that is designed to maintain specific information about every record in the table.

A record, also called a row, is each individual entry that exists in a table. For example, there are 91 records in the above Customers table. A record is a horizontal entity in a table.

simple_3

Semicolon after SQL Statements?

Some database systems require a semicolon at the end of each SQL statement.

Semicolon is the standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.

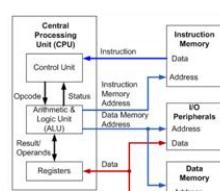
In this tutorial, we will use semicolon at the end of each SQL statement.

simple_4

○ Trung bình (Intermediate):



intermediate_0



intermediate_1

Loại lệnh	A	B	C
CPI theo loại lệnh	1	2	3
IC trong dây lệnh 1	2	1	2
IC trong dây lệnh 2	4	1	1

- Dãy lệnh 1: IC = 2+1+2=5 => N1= 2x1+ 1x2+ 2x3= 10
- Dãy lệnh 2: IC = 4+1+1=6 => N2 = 4 x1+ 1x2+ 1x3= 9

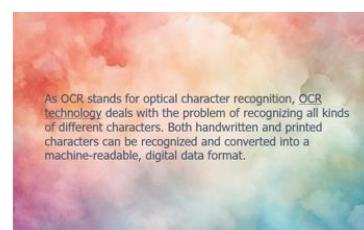
$$CPI_1 = \frac{10}{5} = 2$$

$$CPI_2 = \frac{9}{6} = 1.5$$

intermediate_2



intermediate_3



intermediate_4

○ Phức tạp (Complex):



complex_0



complex_1

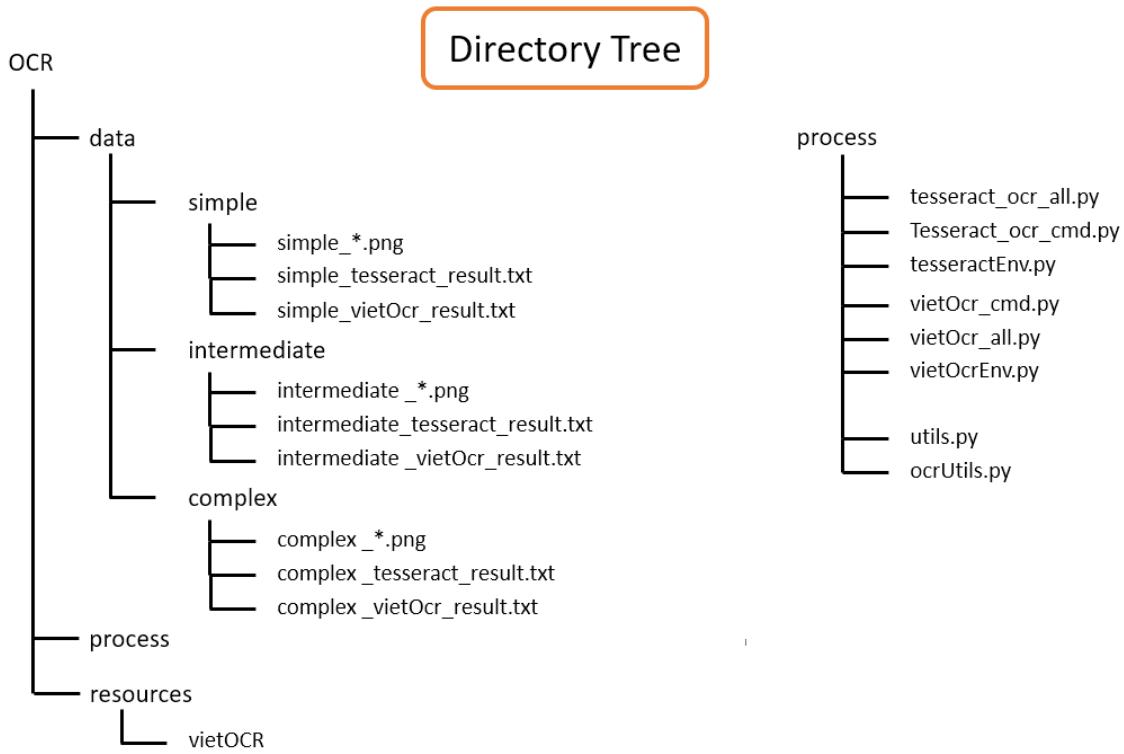


complex_2



Bước 2: Thực hiện lần lượt 2 quá trình tiếp cận nhau:

Trước hết tổ chức cây thư mục sau:



Trong đó:

- ‘data’: Là thư mục chứa dữ liệu phân làm 3 cấp độ {‘simple’, ‘intermediate’, ‘complex’}. Trong mỗi thư mục có chứa ảnh, kết quả xử lý của các ảnh bằng thư viện tesseract và vietOcr theo cú pháp sau:

$f'{{level}}_{{library}}_result.txt'$

- ‘process’: Chứa các code xử lý chính cho việc trích xuất thông tin từ ảnh. Mỗi thư viện sử dụng tiến hành viết ba file xử lý gồm:

$f'{{library}}_all.py'$

$f'{{library}}_cmd.py'$

$f'{{library}}_Env.py'$

Với $f'{{library}}_all.py'$, thực hiện chức năng xử lý toàn bộ ảnh trong các thư mục chưa dữ liệu; $f'{{library}}_cmd.py'$ thực hiện chức năng gọi và truyền tham số bằng command line, giúp việc kích hoạt môi trường và gọi chương trình qua C Sharp được thuận tiện. Cuối cùng là $f'{{library}}_Env.py$ thực hiện gọi quá trình xử lý ở bất cứ đâu trong máy tính, giúp việc gọi chương trình Python được linh hoạt hơn.

- ‘resources’: Chứa phần xử lý chính gồm model pretrained và trọng số của model sử dụng cho việc trích xuất văn bản từ ảnh.
 - Pytesseract: Thuật toán OCR trong Tesseract Python quy trình sau: [Link](#), [Install Pytesseract](#) ; [Tesseract Wraper](#).
 - Cơ bản (Simple):

[1] Computer Organization and Architecture 8th Edition, William Stallings, 2011

[2] Bài giảng kiến trúc máy tính – Viện CNTT-ĐH BK HN

[3] Andrew S. Tanenbaum, Structured Computer Organization, 5rd Edition, Prentice- Hall International Edition, 2006.

[4] Các nguồn tham khảo khác từ internet

simple_0

```
data > simple > simple_tesseract_result.txt
 1  #1
 2  simple_0.png
 3  |
 4  [1] Computer Organization and Architecture 8th Edition, William
 5  Stallings, 2011
 6
 7  [2] Bài giảng kiến trúc máy tính - Viện CNTT-ĐH BK HN
 8
 9  [3] Andrew S. Tanenbaum, Structured Computer Organization, 5rd
10  Edition, Prentice- Hall International Edition, 2006.
11
12  [4] Các nguồn tham khảo khác từ internet
13
14  |
15  #1
```

simple_0_result

100%

¹Machine Learning Department, Carnegie Mellon University
²Department of Computer Science, Princeton University
 agu@cs.cmu.edu, tri@tridao.me

simple_3

```

data > simple > simple_tesseract_result.txt
43  #4
44  simple_3.png
45  |
46  "Machine Learning Department, Carnegie Mellon University
47  "Department of Computer Science, Princeton University
48  agu@cs.cmu.edu, tri@tridao.me
49
50  |
51  #4
  
```

simple_3_result

98%

Nhận xét: Thư viện Tesseract thực hiện trích xuất ảnh trong điều kiện lý tưởng, cấp độ cơ bản, thực hiện rất tốt và đạt được kết quả chính xác cao. Tuy nhiên thư viện khó có thể nhận dạng các chi tiết quá nhỏ, vậy cần thực hiện hiện phóng to vùng ảnh trước khi đưa vào xử lý để tăng hiệu quả trích xuất thông tin.

- Trung bình (Intermediate):



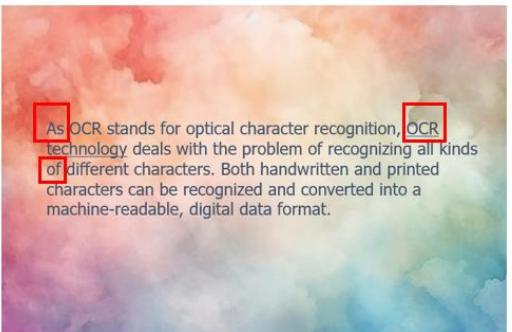
intermediate_3

```

data > intermediate > intermediate_tesseract_result.txt
94  #4
95  intermediate_3.png
96  |
97  Giá trị cốt lõi
98
99  Đặt khách hàng làm trọng tâm, VinFast Sán phẩm đằng cấp, giá
100  không ngừng sáng tạo để tạo ra các sản | tốt, hậu mãi vượt trội.
101  phẩm đằng cấp và trải nghiệm xuất sắc
102  cho mọi người.
103
104  |
105  #4
  
```

intermediate_3_result

80%



intermediate_4

```
data > intermediate > intermediate_tesseract_result.txt
106 #5
107 intermediate_4.png
108 |
109 "
110 #5
111 intermediate_4.png
112 |
113 "
114 #5
115
116
117 |
118 #5
```

Intermediate_4_result

90%

Nhận xét: Thư viện Tesseract thực hiện trích xuất ảnh trong điều kiện bị ảnh hưởng bởi ngũ cẩm như các hàng không liền kề nhau và không tương phản tốt, cấp độ trung bình. Thực hiện nhận dạng thiếu và sai ký tự.

- Phức tạp (Complex):

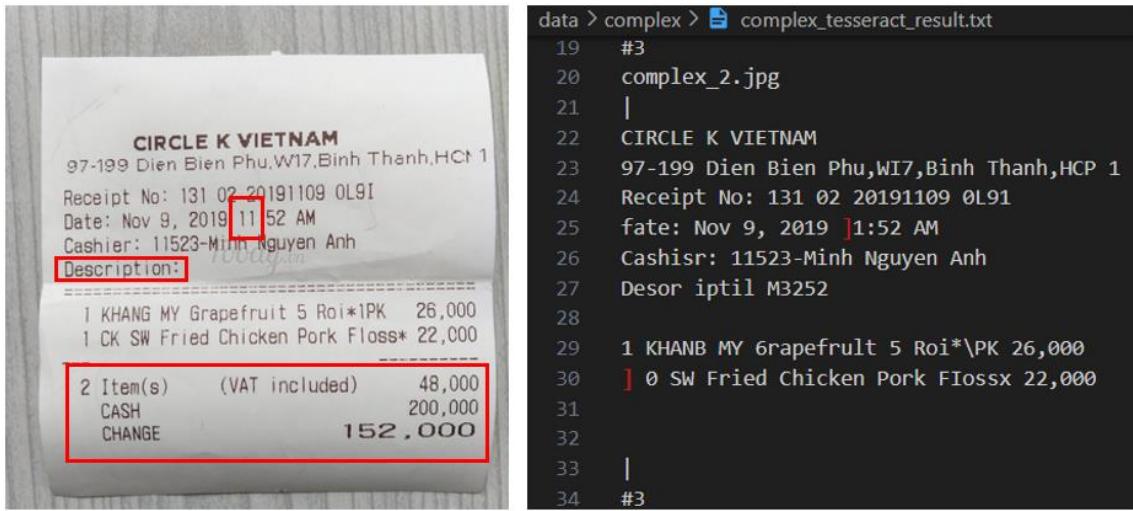


complex_0

```
data > complex > complex_tesseract_result.txt
1 #1
2 complex_0.png
3 |
4 Agency
5
6 Agency
7
8 000 123 456 789 'www.website.com
9
10
11 |
12 #1
```

Complex_0_result

30%



complex_2

complex_2_result

50%

Nhận xét: Thư viện Tesseract thực hiện trích xuất ảnh trong điều kiện bị ảnh hưởng bởi ngũ cẩm, màu sắc không đồng đều, các dòng chữ không ngay ngắn, các kí tự nhỏ, cấp độ phức tạp. Thực hiện nhận dạng thiếu và sai ký tự xảy ra nhiều hơn.

- VietOCR: Thuật toán OCR được thực hiện với Model transformer của VietOCR quy trình sau:

- Cơ bản (Simple):

- [1] Computer Organization and Architecture 8th Edition, William Stallings, 2011
- [2] Bài giảng kiến trúc máy tính – Viện CNTT-ĐH BKHN
- [3] Andrew S. Tanenbaum, Structured Computer Organization, 5rd Edition, Prentice- Hall International Edition, 2006.
- [4] Các nguồn tham khảo khác từ internet

simple_0

```

data > simple > simple_vietocr_result.txt
1  #1
2  D:\Workspace\University\Report_DATN\Report_processes\OCR\data\simple\simple_0.png
3  |
4  036000000095
5  |
6  #1

```

simple_0_result

0%

¹Machine Learning Department, Carnegie Mellon University
²Department of Computer Science, Princeton University
 agu@cs.cmu.edu, tri@tridao.me

simple_3

```
data > simple > simple_vietOcr_result.txt
19  #4
20  D:\Workspace\University\Report_DATN\Report_processes\OCR\data\simple\simple_3.png
21  |
22  "Mightine Learing The ""Communice Comple Assition Stationary"""
23  |
24  #4
```

simple_3_result

0%

Nhận xét: Thư viện VietOCR thực hiện trích xuất ảnh trong điều kiện lý tưởng, cấp độ cơ bản, không thể trích xuất văn bản từ hình ảnh. Vậy thư viện VietOCR không thể xử lý ảnh với thông tin dạng nhiều dòng.

- Trung bình (Intermediate):



intermediate_3

```
data > intermediate > intermediate_vietOcr_result.txt
19  #4
20  D:\Workspace\University\Report_DATN\Report_processes\OCR\data\intermediate\intermediate_3.png
21  |
22  0
23  |
24  #4
```

intermediate_3_result

0%

Nhận xét: Thư viện VietOCR thực hiện trích xuất ảnh trong điều kiện trung bình không thể trích xuất văn bản từ hình ảnh.

- Phức tạp (Complex):



complex_0

```
data > complex > complex_vietOcr_result.txt
1  #1
2  D:\Workspace\University\Report_DATN\Report_processes\OCR\data\complex\complex_0.png
3  |
4  Expection
5  |
6  #1
```

Complex_0_result

0%



```
data > complex > complex_vietOcr_result.txt
13  #3
14  D:\Workspace\University\Report_DATN\Report_processes\OCR\data\complex\complex_2.jpg
15  |
16  0380000099
17  |
18  #3
19  #4
```

complex_2

complex_2_result

0%

Nhận xét: Thư viện VietOCR thực hiện trích xuất ảnh trong điều kiện bị ảnh hưởng bởi ngũ cẩm, màu sắc không đồng đều, các dòng chữ không ngay ngắn, các kí tự nhỏ, cấp độ phức tạp. Không thể trích xuất văn bản từ hình ảnh.

Bước 3: Thực hiện so sánh kết quả, đánh giá và nhận xét hai thuật toán xử lý trên

Đánh giá:

- Tesseract: Là thư viện thực hiện tốt với xử lý đơn ảnh, trong điều kiện lý tưởng. Tuy nhiên với các điều kiện trung bình và phức tạp thì thư viện thực hiện trích xuất thông tin có sai và thiếu các kí tự. Đặc biệt là văn bản bị có

màu kém tương phản so với nền và kí tự không theo các hàng dạng thẳng và liền mạch. Vậy thư viện Tesseract thực hiện tốt trong điều kiện lý tưởng, có thể xử lý toàn bộ bức ảnh và từng vùng ảnh khác nhau.

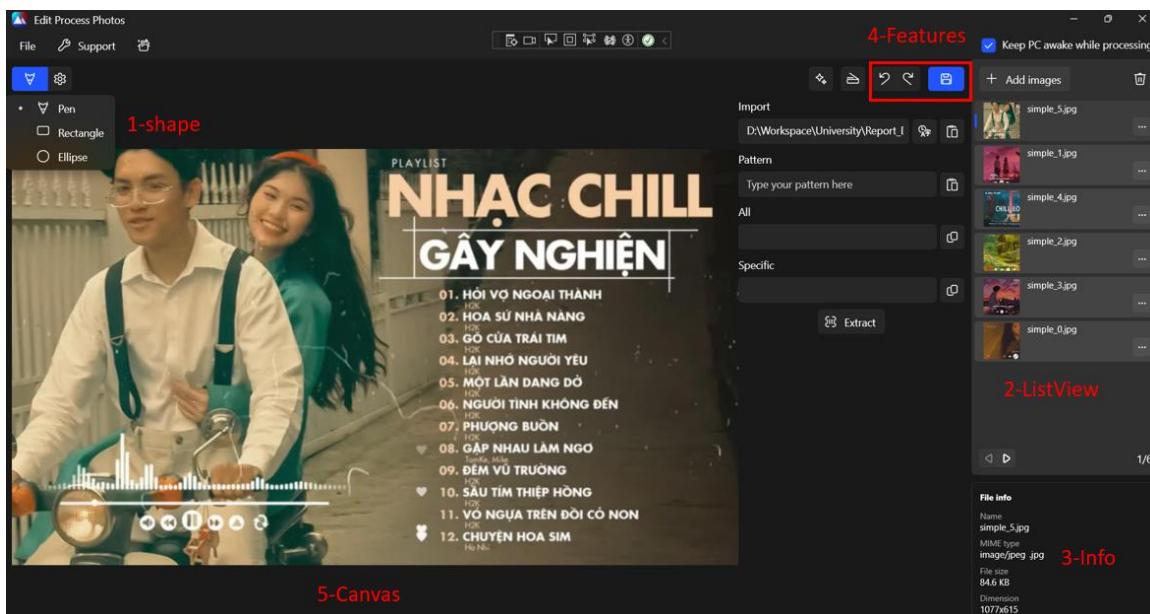
- VietOCR: Thực hiện không thể thực hiện trích xuất văn bản trong tình huống ảnh gồm nhiều dòng kí tự. Vì vậy VietOCR chỉ thích hợp với việc trích xuất các dòng kí tự riêng lẻ. Thuật toán này sẽ phát huy hiệu quả khi xử lý với việc phân đoạn vùng chứa văn bản sau đó tiến hành trích xuất thông tin từ các vùng đó. Vì vậy cần tiền xử lý trước khi sử dụng thuật toán này.

Bước 4: Thực hiện viết chương trình Python cho thuật toán, sử dụng cho việc gọi chương trình Python thông qua Terminal trong C Sharp.

3.2.4 Thực hiện Inpaint trong C Sharp

Qua trình thực hiện Inpaint Image trong C# tiến hành các bước sau:

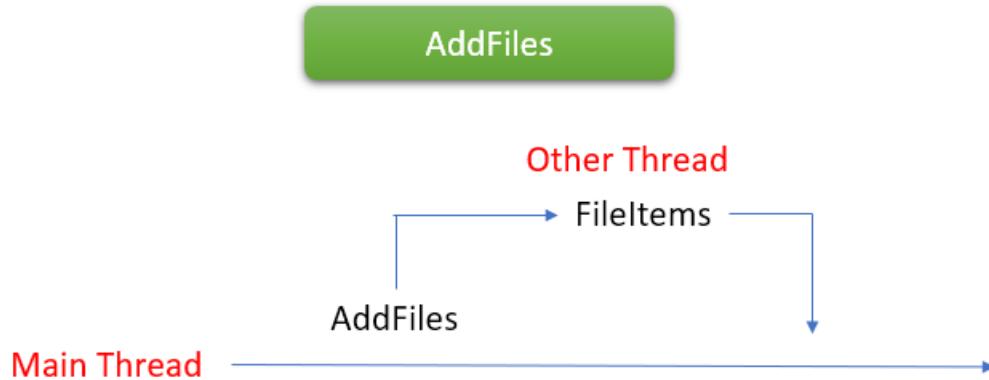
Bước 1: Thực hiện thiết kế giao diện người dùng bao gồm: Danh sách ảnh sử dụng ListView, từng ảnh đơn lẻ được hiển thị bằng Canvas, Thiết kế các bảng chọn mặt nạ thông dụng đường thẳng có độ dày, hình chữ nhật, hình elip.



Quá trình Inpaint trong C Sharp với hình thức tiếp cận là tạo mask bằng các hình dạng khác nhau, trong đó:

1. Shape: Một bảng chọn dạng mask sẽ được tạo là dạng RECTANGLE, ELIP, POLYGON. Trong đó POLYGON, là chuỗi các điểm nối lại với nhau.
2. ListView: Vùng hiển thị danh sách các ảnh được lưu trữ trong một phiên xử lý.
3. Info: Hiện các thông tin chi tiết về ảnh như: tên file, kích thước file, và độ phân giải của file.
4. Features:
 - a. Undo: Khôi phục quá trình Inpaint vừa thực hiện.
 - b. Redo: Thực hiện lại quá trình Inpaint.
 - c. Save: Thực hiện lưu ảnh với kích thước nhất định sau khi xử lý.

Bước 2: Xây dựng quá trình thêm ảnh vào ứng dụng kết hợp với xử lý đa luồng – Multi Threads. Giúp việc thêm ảnh vào ứng dụng không bị khóa giao diện người dùng, tránh được được tình trạng ứng dụng không thể tương tác trong khoảng thời gian nhất định



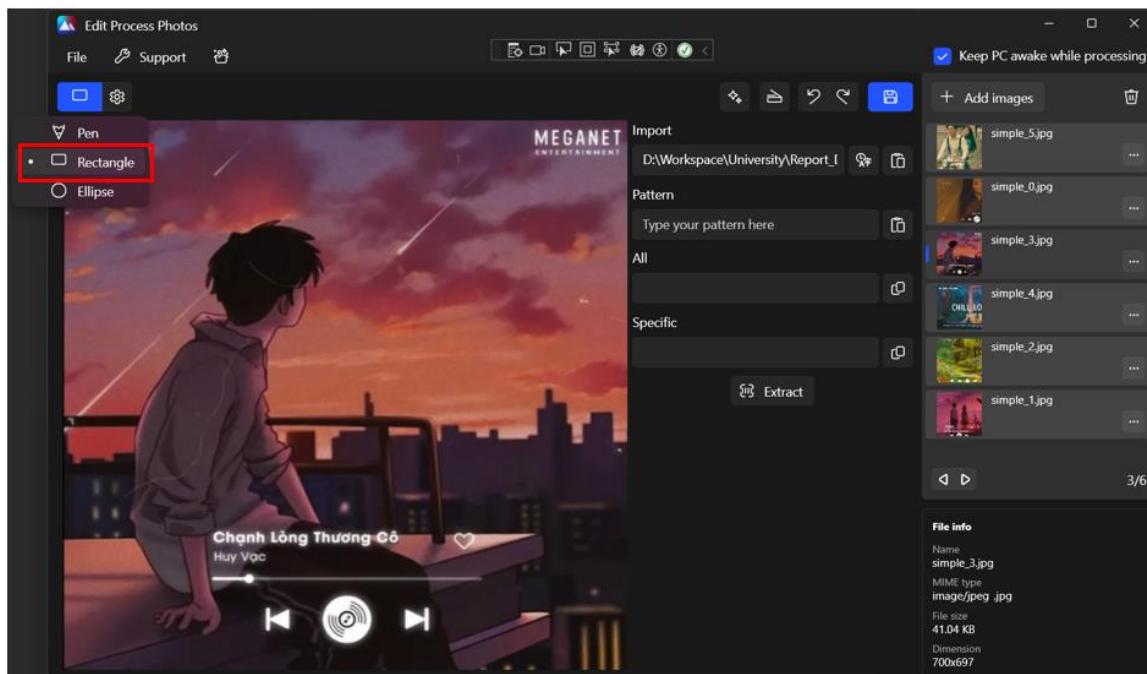
Bước 3: Thực hiện phân xử lý lõi:

- Thực hiện nhận mặt nạ xử lý Inpaint vào trước bằng thao tác trên màn hình, sau đó chuyển đổi và lưu trữ dưới dạng bitmap Image mà OpenCVSharp hỗ trợ riêng bằng cách lấy tọa độ của các điểm được vẽ trên Canvas.

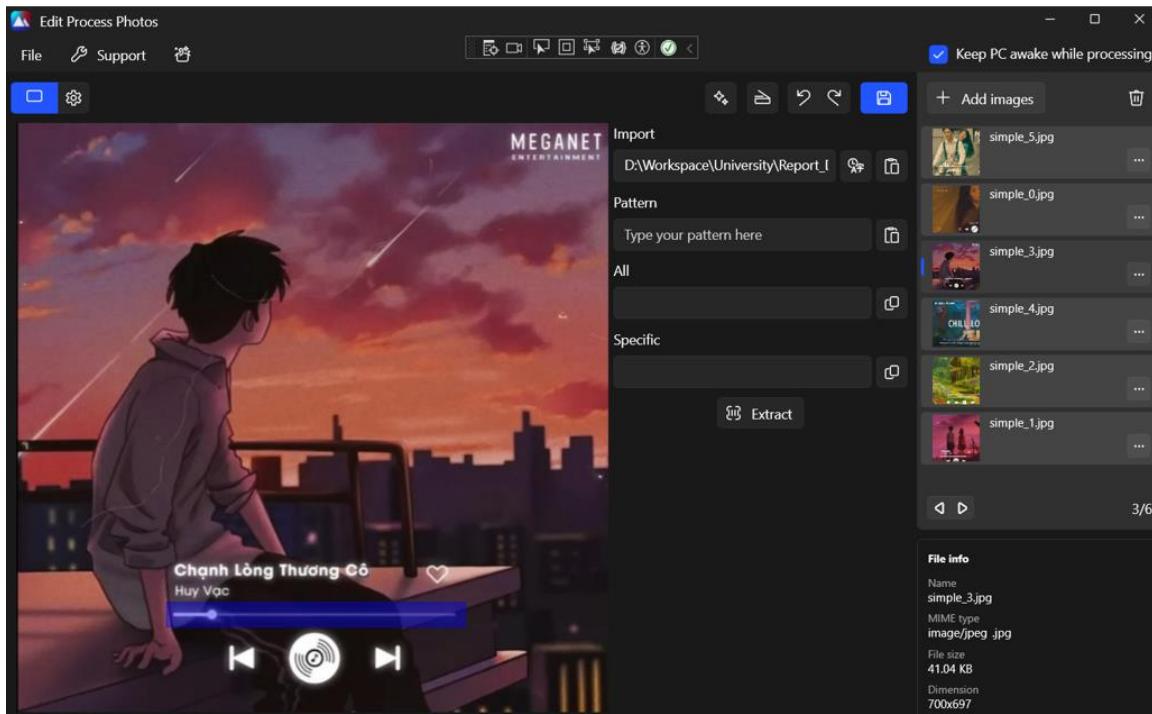
- Thực hiện Inpaint Image ngay sau khi vẽ xong và tiến hành lưu trữ các ảnh được được xử lý dưới dạng một danh sách các ảnh thuận tiện cho việc thực hiện các thao tác như quay lại (Undo), tiến tới (Redo) và lưu ảnh với kích thước khác nhau về sau:
- Xây dựng tính năng Undo, Redo: Thực hiện các thao tác quay lại, thuận tiện hơn cho việc chỉnh sửa ảnh, giúp các thao tác có thể tùy chỉnh dễ dàng.
- Thực hiện xây dựng phần lưu trữ ảnh xuống dưới dạng file khác, lúc này sẽ thuận tiện hơn cho việc lưu trữ ảnh sau khi xử lý.

Bước 4: Tiến hành đánh giá các bước xử lý và cải thiện thực hiện Inpaint ảnh.

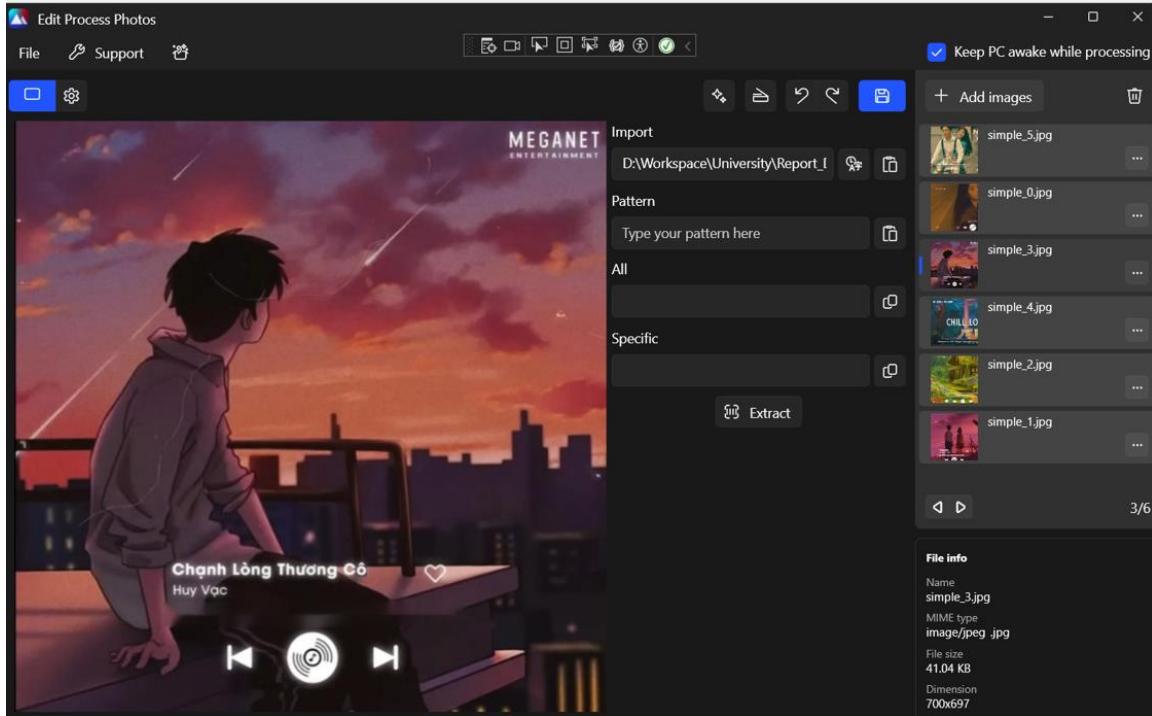
Thực hiện chọn danh sách ảnh đưa vào ứng dụng, sau đó tiến hành Inpaint theo quy trình sau:



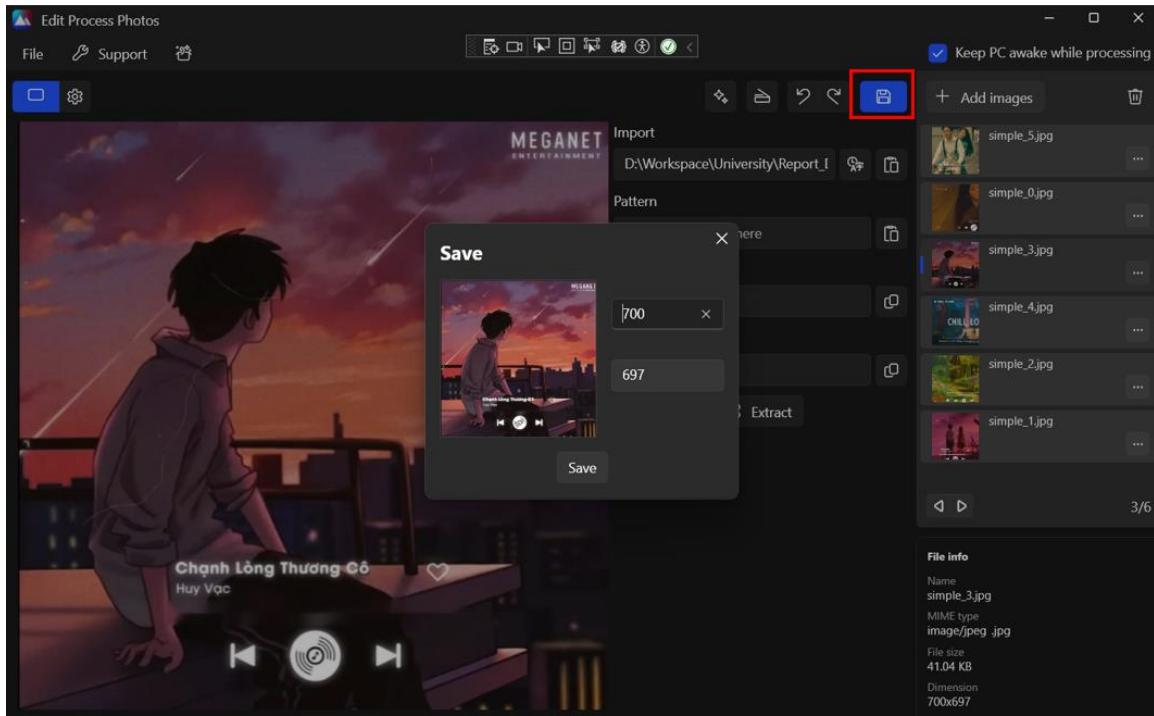
Thực hiện Click vào nút Shape Menu trên màn hình, ứng dụng sẽ hiện ra bảng chọn. Chọn Rectangle để sử dụng tạo mask là hình chữ nhật.



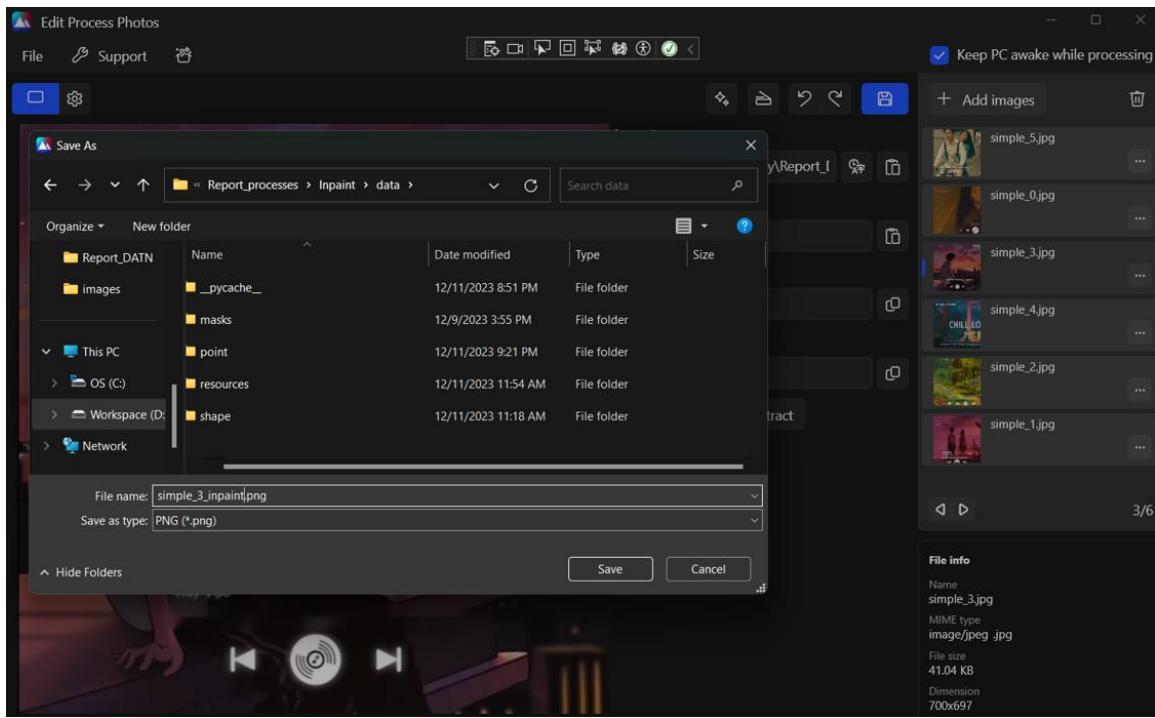
Sau đó thực hiện chọn vùng muốn xử lý Inpaint trong ảnh, bằng cách clickdown vào góc trái trên của hình chữ nhật, và di chuyển tới góc phải dưới và clickup. Quá trình này đã tạo được mask, Quá trình Inpaint sẽ được diễn ra ngay lúc đó.



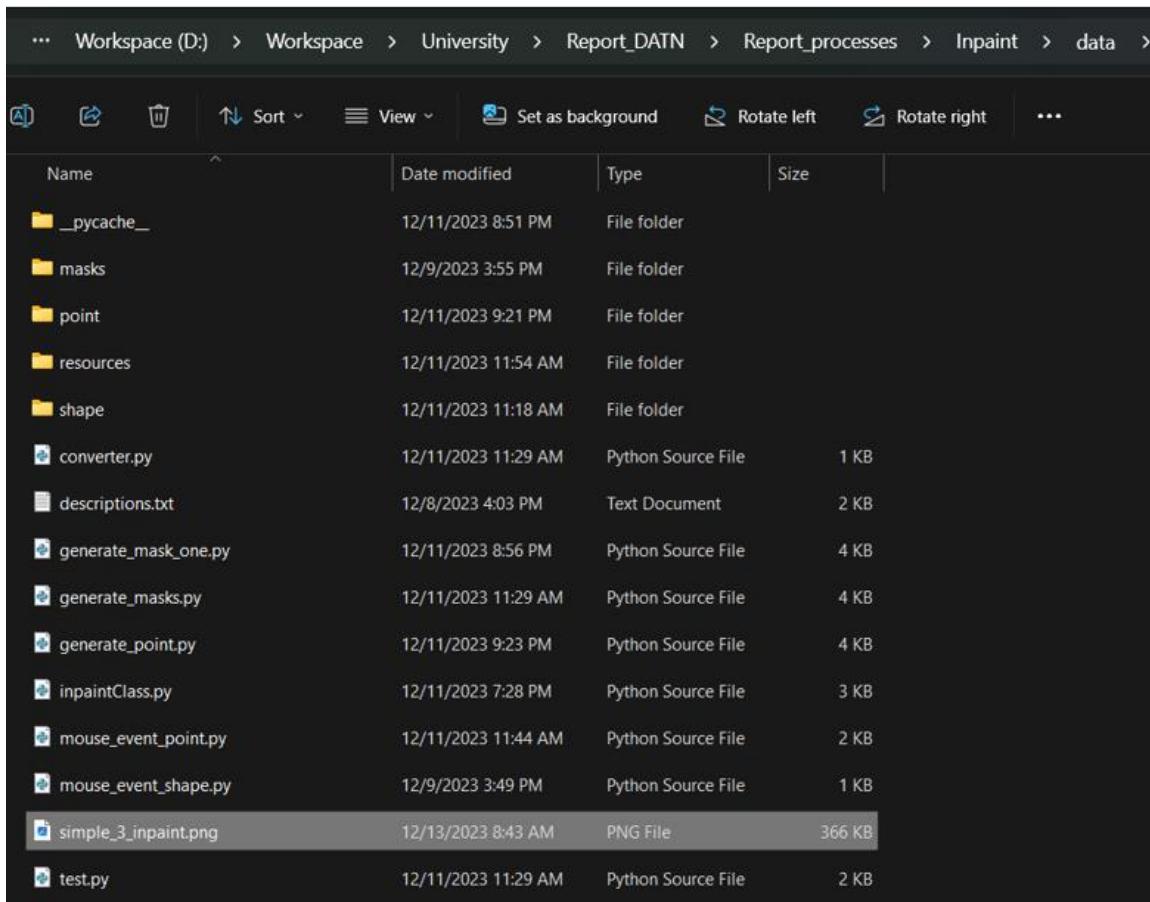
Và ảnh trên là kết quả sau khi xử lý, cho ra kết quả tốt với chi tiết đơn giản.



Và có thể tiếp tục thực hiện Inpaint đến khi nào quá trình xử ảnh trên kết thúc. Tiếp hành lưu sau xử lý, thực hiện Click vào Save Button. Một Dialog sẽ hiện ra cho việc điều chỉnh kích thước của hình ảnh, chức năng này có thể đảm bảo được tỷ lệ của ảnh. Sau đó click vào Save Button trên Dialog để thực hiện chọn nơi lưu ảnh.



Thực hiện chọn nơi lưu ảnh và đặt tên file muốn lưu.



The screenshot shows a Windows File Explorer window with the following directory path: ... > Workspace (D:) > Workspace > University > Report_DATN > Report_processes > Inpaint > data >. The window displays a list of files and folders:

Name	Date modified	Type	Size
__pycache__	12/11/2023 8:51 PM	File folder	
masks	12/9/2023 3:55 PM	File folder	
point	12/11/2023 9:21 PM	File folder	
resources	12/11/2023 11:54 AM	File folder	
shape	12/11/2023 11:18 AM	File folder	
converter.py	12/11/2023 11:29 AM	Python Source File	1 KB
descriptions.txt	12/8/2023 4:03 PM	Text Document	2 KB
generate_mask_one.py	12/11/2023 8:56 PM	Python Source File	4 KB
generate_masks.py	12/11/2023 11:29 AM	Python Source File	4 KB
generate_point.py	12/11/2023 9:23 PM	Python Source File	4 KB
inpaintClass.py	12/11/2023 7:28 PM	Python Source File	3 KB
mouse_event_point.py	12/11/2023 11:44 AM	Python Source File	2 KB
mouse_event_shape.py	12/9/2023 3:49 PM	Python Source File	1 KB
simple_3_inpaint.png	12/13/2023 8:43 AM	PNG File	366 KB
test.py	12/11/2023 11:29 AM	Python Source File	2 KB

Sau quá trình trình trên việc xử lý Inpaint, hoàn thành và app sẽ tự động mở thư mục người dùng lưu ảnh để tiện cho quá trình kiểm tra như trên.

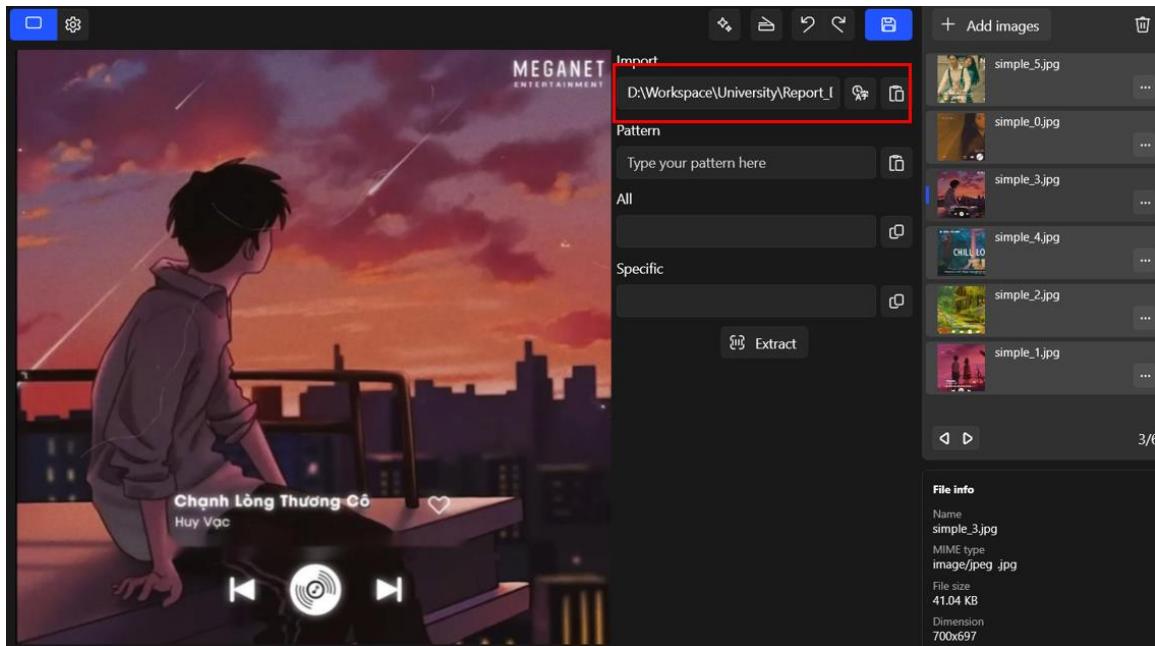
Nhận xét:

Thuật toán Inpaint được xử lý trong C Sharp dùng OpenCV là rất tốt, đạt được những yêu cầu ban đầu đề ra. Đáp ứng được độ chính xác và thời gian đáp ứng nhanh cho người dùng.

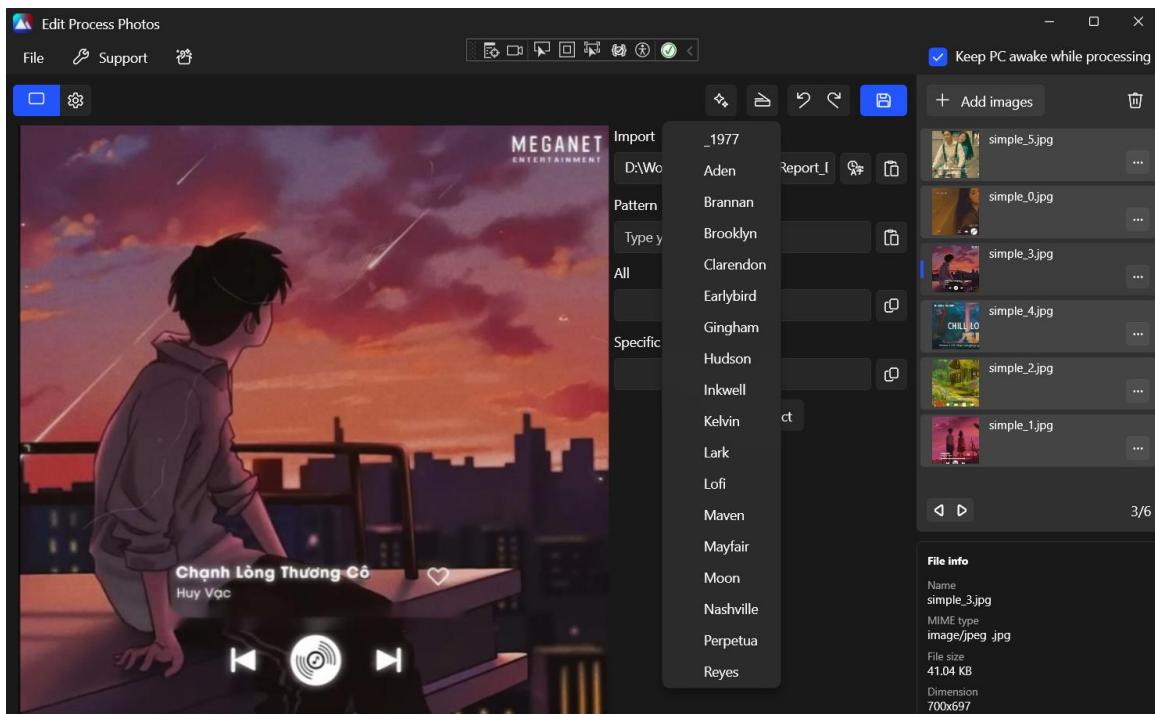
3.2.5 Thực hiện Filter trong C Sharp

Tiến hành sử dụng thư viện Pilgram trong Python để thực hiện thay đổi màu ảnh, sau đó chúng ta thực hiện gọi các lệnh này qua Command trong C#. Thực hiện các bước tiến hành sau:

Bước 1: Thực hiện lấy dữ liệu ảnh bằng đường dẫn ảnh được thêm vào ứng dụng.



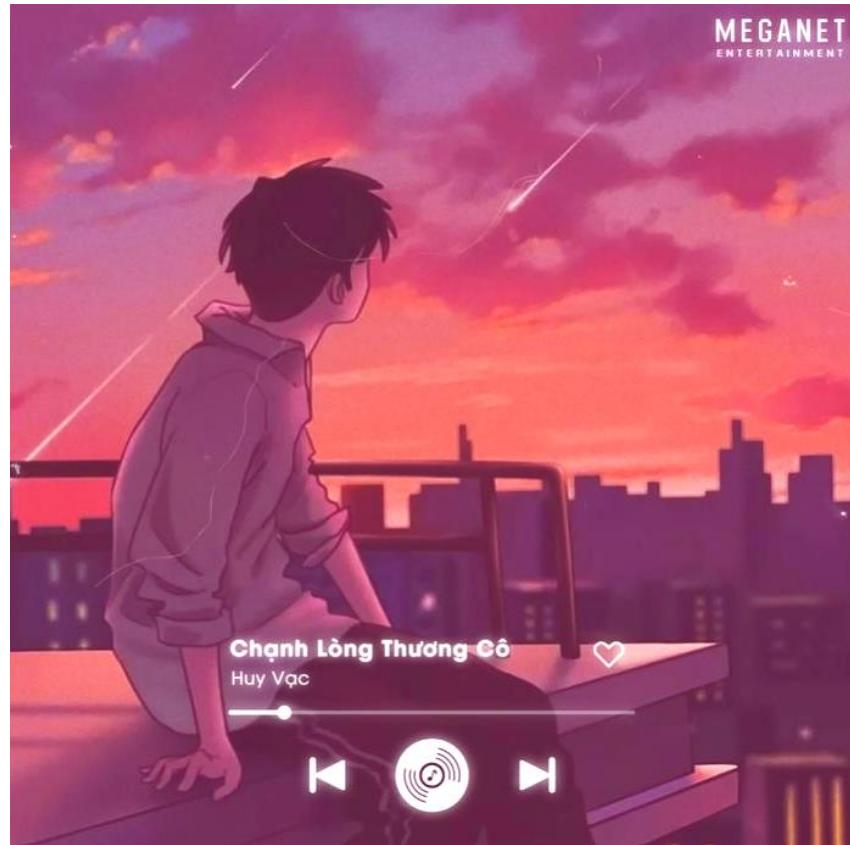
Bước 2: Thực hiện xây dựng bảng chọn với các dạng filter chỉnh sửa màu khác nhau của ảnh.



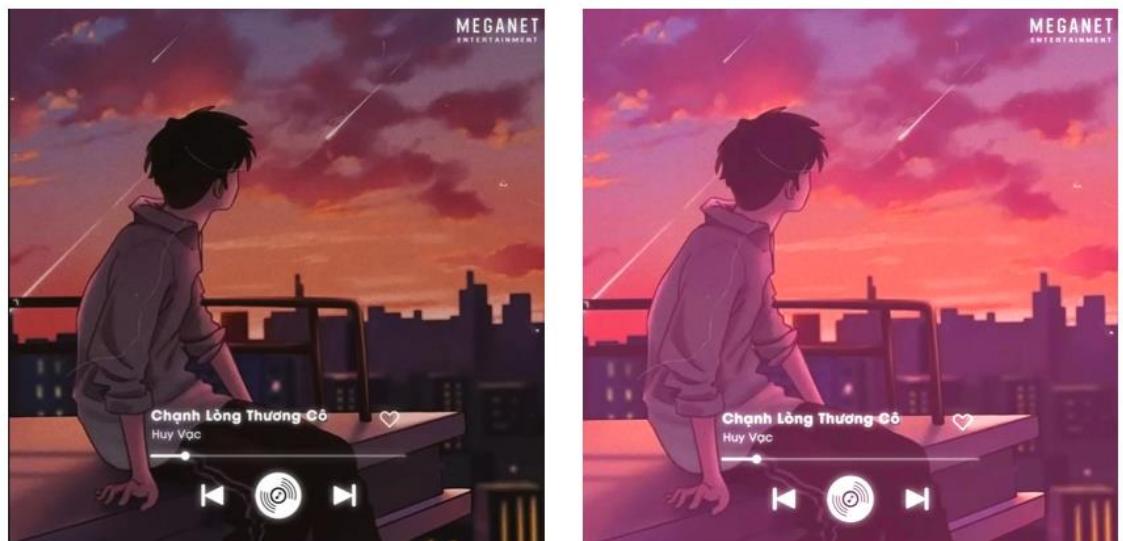
Bước 3: Sau khi nhận được filter được chọn, tiến hành gọi lệnh thực thi qua code Python trong dự án để tiến hành thực hiện bước điều chỉnh màu cho ảnh.

Bước 4: Tiến hành lưu ảnh dưới dạng tạm, lưu các thông tin của ảnh gốc.

Bước 5: Sau đó tiến hành load lại ảnh vào trong ứng dụng cho người dùng xem.



Bước 6: Tiến hành đánh giá kết quả xử lý của ảnh.



Before

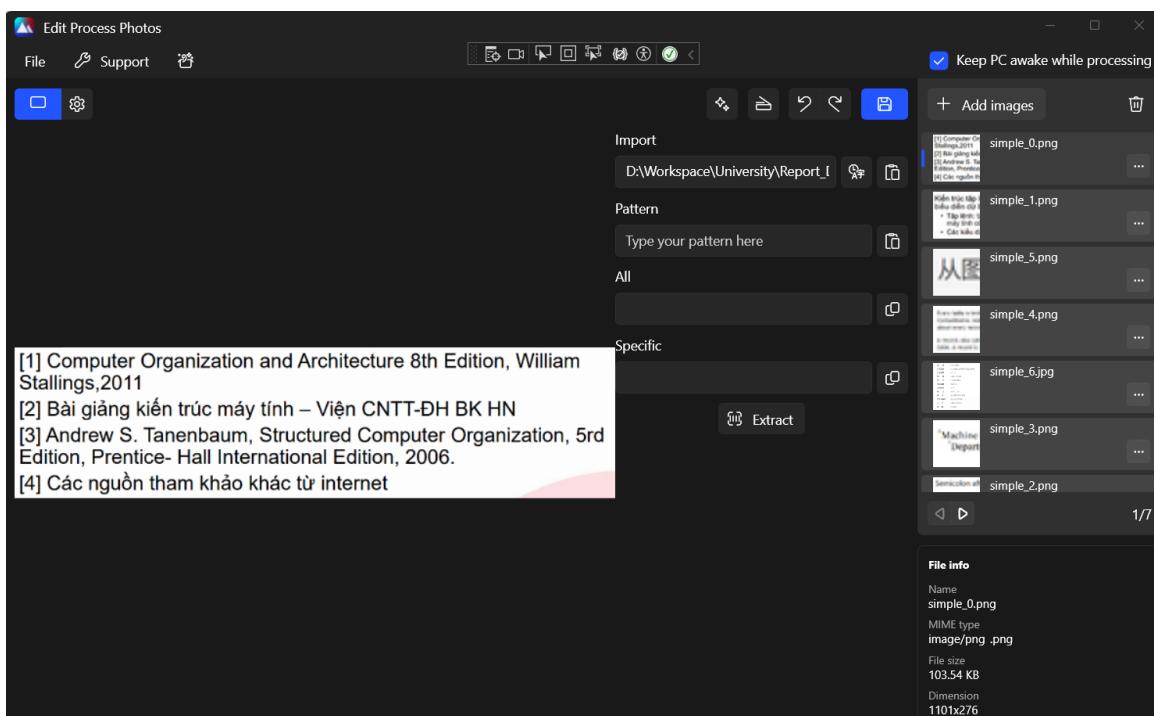
After

Nhận xét: Vậy việc thực hiện lọc màu cho ảnh trong ứng dụng đạt được kết quả như mong đợi, thời gian xử lý nhanh và đa dạng các bộ lọc màu khác nhau.

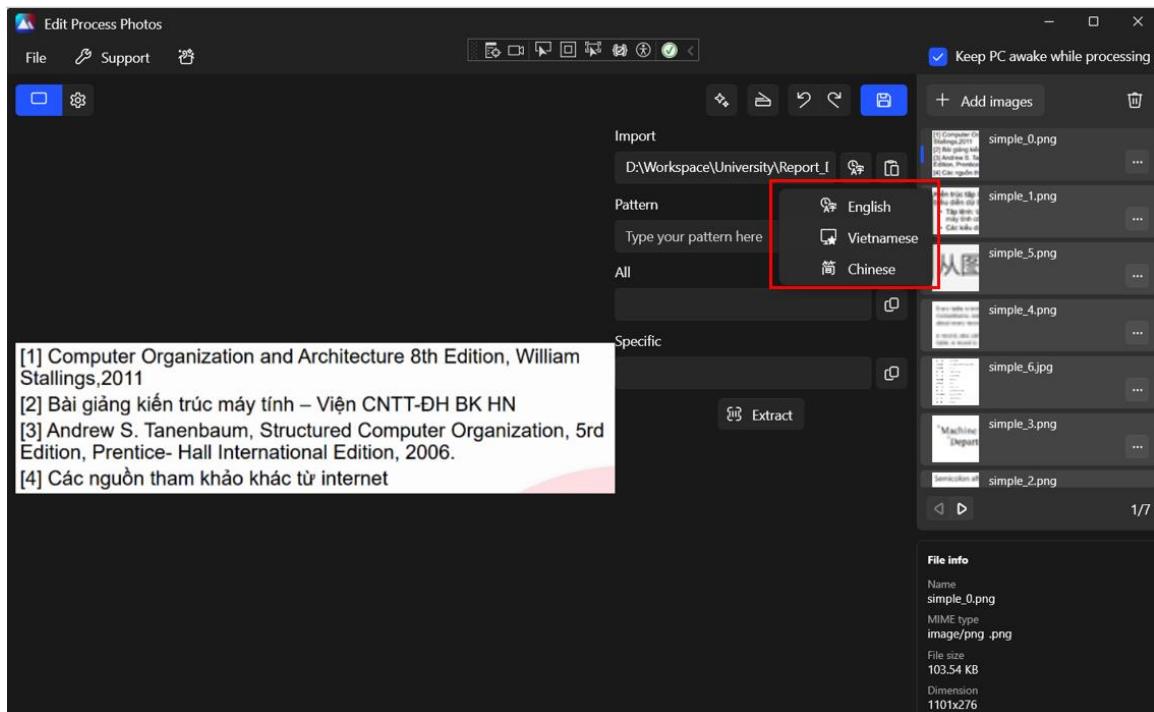
3.2.6 Thực hiện OCR trong C Sharp

Tiến hành trích xuất văn bản dùng kỹ thuật OCR trong C Sharp qua thư viện IronOCR, có xử lý lỗi dựa trên Tesseract 5.0, thực hiện theo các bước sau:

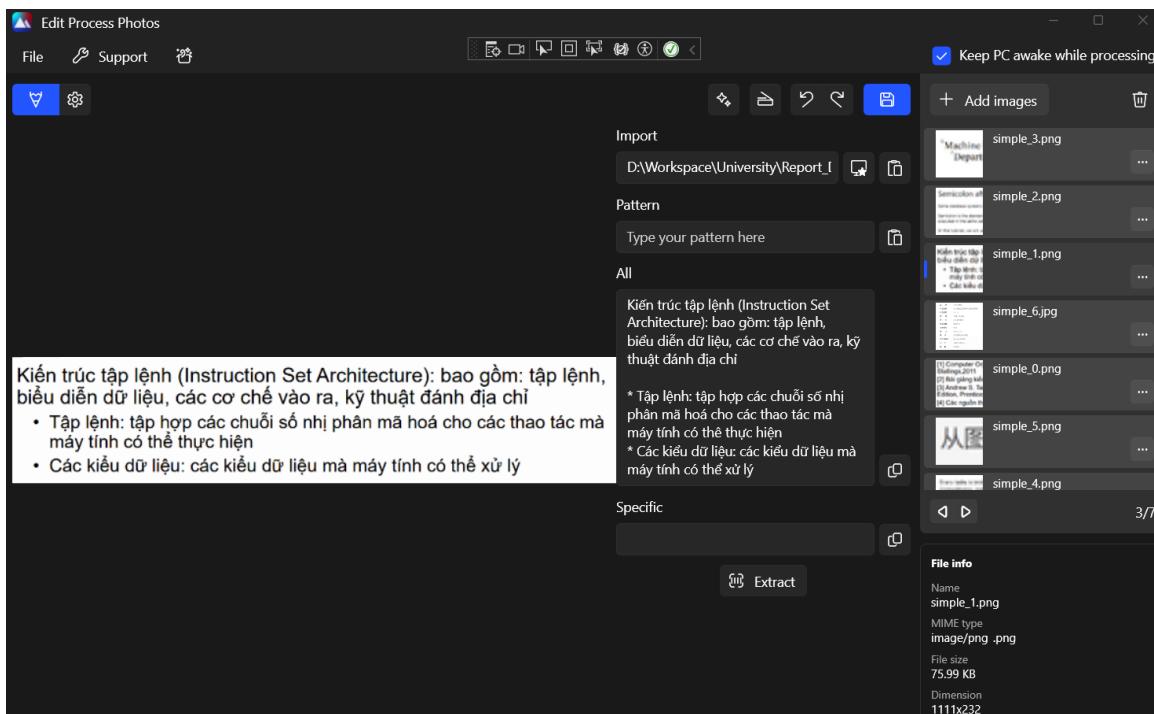
Bước 1: Thực hiện lấy dữ liệu ảnh bằng đường dẫn ảnh được thêm vào ứng dụng.



Bước 2: Thực hiện lựa chọn ngôn ngữ tiếng Việt để thử nghiệm, người dùng có thể lựa chọn tiếng Anh và tiếng Trung.



Bước 3: Thủ nghiệm trích xuất văn bản từ ảnh.



Mặc định ứng dụng sẽ trích xuất toàn bộ chữ từ ảnh và lưu vào TextBox ‘All’. Còn nếu người dùng sử dụng Pattern Regex thì có thể trích xuất đúng dạng chuỗi kí tự mà mình muốn từ thông tin trích xuất từ ảnh.

Bước 4: Đánh giá kết quả của ảnh trước và sau khi xử lý.

Nhận xét: Việc trích xuất thông tin từ ảnh trong ứng dụng hoạt động tốt trong điều kiện lý tưởng, vẫn đáp ứng được nhu cầu cơ bản của người dùng. Tuy nhiên với các trường hợp phức tạp hợp việc trích xuất thông tin bị biết hoặc bị sai giá trị.

3.3 Kết luận chương

Sau quá trình thực hiện nghiên cứu và triển khai các hình thức tiếp cận khác nhau với các kĩ thuật xử lý của ứng dụng như: Inpaint, Filter, OCR. Đạt được các kết quả như sau:

Thuật toán Inpaint xử lý trong Python: OpenCV, Inpaint Anything thì thực hiện xóa vật thể quá tốt trong tình huống đơn giản. Thuật toán dùng OpenCV phù hợp hơn trong việc xử lý các chi tiết nhỏ, có độ tương phản tốt với nền và nền có cấu trúc đồng nhất. Còn thuật toán Inpaint Anything thì chưa phát huy tốt trong trường hợp các chi tiết đối tượng là nhỏ và thiết liên kết với nhau. Để đáp ứng nhu cầu của đề tài tiến hành sử dụng Inpaint OpenCv trong ứng dụng Windows.

Inpaint OpenCV trong ứng dụng Windows đã đáp ứng được tính đơn giản, dễ dùng, thời gian đáp ứng nhanh như nhu cầu đã đề ra.

Filter Python đáp ứng được các yêu cầu đề ra. Thời gian đo đạc được cho việc xử lý filter của một bức ảnh rơi vào khoảng $0.034s/image$ cho một bức ảnh.

Filter trong ứng dụng Windows có thể đáp ứng được nhu cầu tương tự, nhưng việc xử lý File còn nhiều vấn đề xoanh quanh như: Việc xử lý lưu file tạm, xử lý các threads khác nhau giúp không bị khóa UI, và thực hiện tải lại ảnh vào ứng dụng.

Thuật toán OCR được triển khai trong Python và C Sharp để đáp ứng được nhu cầu đề ra. Việc xử lý OCR trong ứng dụng Windows còn gặp khó khăn cho việc xử lý các ngữ cảnh phức tạp.

CHƯƠNG 4. KẾT LUẬN

4.1 Tóm tắt và kết luận chung

Đề tài xây dựng ứng dụng chỉnh sửa và trích xuất thông tin từ ảnh trên Windows đã đáp ứng được các câu hỏi đề ra. Nghiên cứu và tìm hiểu OpenCV trong việc thực hiện các thao tác xử lý ảnh, Inpaint. Thiết kế giao diện dùng WinUI 3 Framework, tổ chức thuận tiện cho việc phát triển và bảo trì xây dựng trên nền tảng Windows. Thực hiện nghiên cứu các thành phần hỗ trợ xử lý ảnh OpenCV và pilgram, filter. Nghiên cứu trích xuất thông tin thông qua OCR. Và ứng dụng cũng đáp ứng được các nhu cầu đề ra của đề tài: Độ chính xác, thời gian đáp ứng nhanh, đơn giản cho việc sử dụng.

Tuy nhiên đề tài chưa thể xử lý tốt trong các trường hợp tương đối phức tạp. Thuật toán Inpaint chưa thể xóa các vật thể có chi tiết nhiều và nền không có cấu trúc đồng nhất. Chỉnh sửa ảnh còn chưa đa dạng thao tác xử lý như: thay đổi độ tương phản, độ sáng, ... Thực hiện trích xuất thông tin ảnh còn chưa thực hiện tốt trong tình huống các văn bản có độ tương phản kém và bị lồng vào ngữ cảnh nhiều.

4.2 Hướng phát triển

Đề tài xây dựng ứng dụng chỉnh sửa và trích xuất thông tin từ ảnh trên Windows. Đã đáp ứng những câu hỏi nghiên cứu đề ra của đề tài. Tuy nhiên cần có sự cải tiến để đáp ứng độ chính xác cao hơn xong các bước xử lý.

Xử lý Inpaint có thể sử dụng thuật toán Inpaint Anything để có thể xóa các vật thể phức tạp hơn trong nhiều ngữ cảnh đa dạng hơn. Việc xử lý này giúp tăng tính đáp ứng của ứng dụng.

Xử lý chỉnh sửa ảnh có thể bổ sung cho việc xử lý Filter bằng các hoạt động xử lý thông dụng như: Điều chỉnh độ sáng, cắt ảnh, xoay ảnh.

Trích xuất thông tin dùng thư viện Tesseract có thể đạt độ chính xác tốt hơn nếu thực hiện thêm tính năng khoanh vùng các ô chữ nhật cụ thể để có thể trích xuất thông tin chính xác hơn.

PHỤ LỤC A

(Ghi chú: Code chương trình có thể để vào phụ lục A và các chứng minh toán học hỗ trợ có thể để vào phụ lục B)

A.1 Code chương trình Filter Python

```
import os
import sys
import PIL.Image
import pilgram

from enum import Enum
from utils.utils import Utils

class FilterType(Enum):
    _1977 = 1
    aden = 2
    brannan = 3
    brooklyn = 4
    clarendon = 5
    earlybird = 6
    gingham = 7
    hudson = 8
    inkwell = 9
    kelvin = 10
    lark = 11
    lofi = 12
    maven = 13
    mayfair = 14
    moon = 15
    nashville = 16
    perpetua = 17
    reyes = 18
    rise = 19
    slumber = 20
    stinson = 21
    toaster = 22
    valencia = 23
    walden = 24
```

```
willow = 25
xpro2 = 26

def FilterImage(filterType: FilterType, src, dest):
    img = PIL.Image.open(src)

    if filterType == FilterType._1977:
        pilgram._1977(img).save(dest)
    elif filterType == FilterType.aden:
        pilgram.aden(img).save(dest)
    elif filterType == FilterType.brannan:
        pilgram.brannan(img).save(dest)
    elif filterType == FilterType.brooklyn:
        pilgram.brooklyn(img).save(dest)
    elif filterType == FilterType.clarendon:
        pilgram.clarendon(img).save(dest)
    elif filterType == FilterType.earlybird:
        pilgram.earlybird(img).save(dest)
    elif filterType == FilterType.gingham:
        pilgram.gingham(img).save(dest)
    elif filterType == FilterType.hudson:
        pilgram.hudson(img).save(dest)
    elif filterType == FilterType.inkwell:
        pilgram.inkwell(img).save(dest)
    elif filterType == FilterType.kelvin:
        pilgram.kelvin(img).save(dest)
    elif filterType == FilterType.lark:
        pilgram.lark(img).save(dest)
    elif filterType == FilterType.lofi:
        pilgram.lofi(img).save(dest)
    elif filterType == FilterType.maven:
        pilgram.maven(img).save(dest)
    elif filterType == FilterType.mayfair:
        pilgram.mayfair(img).save(dest)
    elif filterType == FilterType.moon:
        pilgram.moon(img).save(dest)
    elif filterType == FilterType.nashville:
        pilgram.nashville(img).save(dest)
    elif filterType == FilterType.perpetua:
        pilgram.perpetua(img).save(dest)
    elif filterType == FilterType.reyes:
        pilgram.reyes(img).save(dest)
    elif filterType == FilterType.rise:
        pilgram.rise(img).save(dest)
    elif filterType == FilterType.slumber:
```

```

        pilgram.slumber(img).save(dest)
    elif filterType == FilterType.stinson:
        pilgram.stinson(img).save(dest)
    elif filterType == FilterType.toaster:
        pilgram.toaster(img).save(dest)
    elif filterType == FilterType.valencia:
        pilgram.valencia(img).save(dest)
    elif filterType == FilterType.walden:
        pilgram.walden(img).save(dest)
    elif filterType == FilterType.willow:
        pilgram.willow(img).save(dest)
    elif filterType == FilterType.xpro2:
        pilgram.xpro2(img).save(dest)

def main():
    for filter in FilterType:
        FilterImage(filterType= filter, src= r'assets\mountain.jpg', dest=
r'outputs\o' + '_' + filter.name + r'.jpg')

if __name__ == "__main__":
    try:
        main()
    except KeyboardInterrupt:
        print("Interrupted")
        try:
            sys.exit(0)
        except SystemExit:
            os._exit(0)

```

A.2 Code chương trình xử lý C Sharp

Thiết kế Filter:

```

public enum FilterType
{
    _1977,
    Aden,
    Brannan,
    Brooklyn,
    Clarendon,

```

```
    Earlybird,
    Gingham,
    Hudson,
    Inkwell,
    Kelvin,
    Lark,
    Lofi,
    Maven,
    Mayfair,
    Moon,
    Nashville,
    Perpetua,
    Reyes,
    Rise,
    Slumber,
    Stinson,
    Toaster,
    Valencia,
    Walden,
    Willow,
    Xpro2,
}

public Dictionary<FilterType, RadioMenuFlyoutItem> FILTERS;

public enum StatusType
{
    Ready,
    Loading,
    Loaded,
    LoadFailed,
    Processing,
    Processed,
    ProcessFailed,
}
```

Xử lý Input File ảnh vào trong ứng dụng:

```
public void AddFiles(IReadOnlyList<string> paths, Action startAction =
null, Action<StatusType> endAction = null, Action<List<ThumbnailItem>>
itemAction = null)
{
    if (Utils.Any(_status, StatusType.Loading, StatusType.Processing))
return;
```

```
    Status = StatusType.Loading;
    startAction?.Invoke();

    var hasCorrupted = false;
    var hasEpsAndCannotRead = false;

    List<ThumbnailItem> items = new();
    var coreCount = Environment.ProcessorCount;
    object locked = new();

    var start = FileItems.Count == 0;

    IProgress<StatusType> progress = new Progress<StatusType>(status =>
    {
        if (status == StatusType.Loaded)
        {
            if (hasEpsAndCannotRead)
                MainWindow.Inst.ShowMissingLibDialog();
            else if (hasCorrupted)
                MainWindow.Inst.ShowMessageTeachingTip(null, string.Empty,
                _resourceLoader.GetString("LoadCorruptedSomeFiles"));
        }

        Status = status;
        endAction?.Invoke(status);

        if (start && FileItems.Count > 0)
            FileListView.SelectedIndex = 0;
    });

    IProgress<List<ThumbnailItem>> itemProgress = new
    Progress<List<ThumbnailItem>>(items =>
    {
        lock (locked)
        {
            FileItems.AddRange(items);
            itemAction?.Invoke(items);
        }
    });

    _ = Task.Run(() =>
    {
        try
        {
            var packages = paths.Chunk(256);
```

```
foreach (var package in packages)
{
    var pathChunksPerPackage = package.Chunk(coreCount);

    lock (locked)
    {
        items.Clear();
    }

    foreach (var pathChunks in pathChunksPerPackage)
    {
        Parallel.ForEach(pathChunks, path =>
        {
            try
            {
                var isFilePath = Utils.IsFilePath(path);
                if (isFilePath.GetValueOrDefault(false) &&
!FileItems.Any(i => i.InputFilePath == path))
                {
                    var imageMeta =
ImageMagickUtils.GetMagickImageMeta(path);

                    if
(Profile.IsAcceptedInputFormat(imageMeta?.Format))
                    {
                        lock (locked)
                        {
                            items.Add(new() { InputInfo =
new(path) });
                        }
                    }
                    else throw new();
                }
            }
            catch (MissingLibException mle)
            {
                if (mle.Message == "eps")
                    hasEpsAndCannotRead = true;
                else
                    hasCorrupted = true;
            }
            catch (Exception)
            {
                hasCorrupted = true;
            }
        });
    }
}
```

```
        }
    });

    itemProgress.Report(items);
}

progress.Report(StatusType.Loaded);
}
catch (Exception)
{
    progress.Report(StatusType.LoadFailed);
}
});
}
```

Chương trình xử lý phần trích xuất văn bản ra khỏi ảnh:

```
private void OCRButton_Click(object sender, RoutedEventArgs e)
{
    if (sender is not Control control) return;
    if (InputPath.Text == "") return;

    string pattern = InputPattern.Text;
    string inputPath = InputPath.Text;

    IronTesseract IronOcr = new();
    IronOcr.Language = (LangType)LanguageButton.Tag switch
    {
        LangType.English     => OcrLanguage.English,
        LangType.Vietnamese => OcrLanguage.Vietnamese,
        LangType.Chinese     => OcrLanguage.ChineseSimplifiedFast,
        _                     => OcrLanguage.English,
    };

    OCROutput.Text = IronOcr.Read(inputPath).Text;

    if (OCROutput.Text != "" && pattern != "")
    {
        Regex regex = new Regex(pattern);
        Match match = regex.Match(OCROutput.Text);
        if (match.Success)
            OCRSpecificOutput.Text = match.Value;
    }
}
```

TÀI LIỆU THAM KHẢO

- [1] S. S. S. R. Rautela, "Chatbot: A Bridge Between Technology and Learn," 2022 *International Conference on Cyber Resilience (ICCR)*, 2022.
- [2] R. Mursalzade. "Introducing Answer Bot." <https://www.zendesk.com/blog/introducing-answer-bot/> (accessed).
- [3] S. Singapore. "Sephora Virtual Artist | Sephora Singapore." <https://www.sephora.sg/pages/virtual-artist> (accessed).
- [4] "Babylon." <https://www.babylonhealth.com/> (accessed).
- [5] "Erica - Bank of America Mobile Banking app." <https://promotions.bankofamerica.com/digitalbanking/mobilebanking/erica> (accessed).
- [6] "Expedia launches conversational trip planning powered by ChatGPT to inspire members to dream about travel in new ways." <https://www.expediagroup.com/investors/news-and-events/financial-releases/news/news-details/2023/Chatgpt-Wrote-This-Press-Release--No-It-Didnt-But-It-Can-Now-Assist-With-Travel-Planning-In-The-Expedia-App/default.aspx> (accessed).
- [7] K. Johnson. "Recruitment chatbot Mya automates 75% of hiring process." <https://venturebeat.com/business/recruitment-chatbot-my-a-automates-75-of-hiring-process/> (accessed).
- [8] "Survey Sparrow Home page." <https://surveysparrow.com/> (accessed).
- [9] K. Brown. "Something Bothering You? Tell It to Woebot." <https://www.nytimes.com/2021/06/01/health/artificial-intelligence-therapy-woebot.html> (accessed).
- [10] Đ. Hung. "ChatGPT: Những quan điểm "sáng" và "tối" trong quá trình "tự động hóa" tri thức." <https://ictvietnam.vn/chatgpt-nhung-quan-diem-sang-va-toi-trongqua-trinh-tu-dong-hoa-tri-thuc-56494.html> (accessed).
- [11] N. S. Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, "Attention Is All You Need," 2017.
- [12] T. Vu. "Embedding." https://machinelearningcoban.com/tabml_book/ch_embedding/embedding.html#:~:text=Embedding%20l%C3%A0%20m%E1%BB%99t%20k%E1%BB%99%20thu%E1%BA%ADt,m%E1%BB%99t%20vector%20d%E1%BA%A1ng%20one%2Dhot. (accessed).

- [13] A. K. Reddy. "Guide to Chroma DB | A Vector Store for Your Generative AI LLMs." <https://www.analyticsvidhya.com/blog/2023/07/guide-to-chroma-db-a-vector-store-for-your-generative-ai-llms/> (accessed).
- [14] "Embeddings." <https://docs.trychroma.com/embeddings> (accessed).
- [15] I. G. Nils Reimers, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," *EMNLP 2019*, 2019.