TRƯỜNG ĐẠI HỌC HÀNG HẢI VIỆT NAM

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

HỌC PHẦN: XỬ LÝ ẢNH

Đề tài: Xây dựng chương trình xử lí ảnh chạy trên thiết bị di động cho phép xử lý ảnh với thao tác lọc ảnh và ghi kết quả thành file

Giảng viên hướng dẫn: ***Ts. Nguyễn Hữu Tuân***

***Sinh viên thực hiện*:**

Phạm Quang Long – MSV: 86847

Hải phòng,ngày 14 tháng 4 năm 2023

MỤC LỤC

LỜI MỞ ĐẦU

Xử lý ảnh là quá trình sử dụng các thuật toán và phần mềm để xử lý, phân tích và trích xuất thông tin từ hình ảnh. Với sự phát triển của công nghệ, xử lý ảnh trở nên ngày càng phổ biến và được áp dụng rộng rãi trong nhiều lĩnh vực, bao gồm cả trong thiết bị động như điện thoại thông minh và máy tính bảng.

Trong thời đại hiện nay, xử lý ảnh đang được áp dụng rộng rãi trong các ứng dụng di động như camera điện thoại, ứng dụng chụp ảnh và video, cũng như trong các ứng dụng tìm kiếm, xác nhận khuôn mặt, quét mã QR và chụp hình selfie. Các thuật toán nhận dạng khuôn mặt, phân tích cảm xúc và tăng cường ảnh cũng được áp dụng trong các ứng dụng này để cải thiện chất lượng và độ chính xác của ảnh.

Ngoài ra, xử lý ảnh cũng được áp dụng trong các thiết bị động như xe tự hành, robot hút bụi và máy bay không người lái. Các thuật toán xử lý ảnh giúp cho các thiết bị này có thể nhận dạng, phân tích và đáp ứng với môi trường xung quanh một cách chính xác và hiệu quả.

Tóm lại, xử lý ảnh đóng vai trò quan trọng trong nhiều lĩnh vực và đang được áp dụng rộng rãi trong thiết bị động như điện thoại thông minh, máy tính bảng, xe tự hành và robot. Các thuật toán và công nghệ xử lý ảnh tiếp tục được nghiên cứu và phát triển để cải thiện hiệu suất và độ chính xác của các thiết bị này trong tương lai.

Chương 1 Giới thiệu

1.1 Mục đích xử lí ảnh

+ Xử lý ảnh (XLA) là đối tượng nghiên cứu của lĩnh vực thị giác máy, là quá trình biến đổi từ một ảnh ban đầu sang một ảnh mới với các đặc tính và tuân theo ý muốn của người sử dụng. Xử lý ảnh có thể gồm quá trình phân tích, phân lớp các đối tượng, làm tăng chất lượng, phân đoạn và tách cạnh, gán nhãn cho vùng hay quá trình biên dịch các thông tin hình ảnh của ảnh.

+ Cũng như xử lý dữ liệu bằng đồ hoạ, xử lý ảnh số là một lĩnh vực của tin học ứng dụng. Xử lý dữ liệu bằng đồ họa đề cập đến những ảnh nhân tạo, các ảnh này được xem xét như là một cấu trúc dữ liệu và được tạo bởi các chương trình. Xử lý ảnh số bao gồm các phương pháp và kỹ thuật biến đổi, để truyền tải hoặc mã hoá các ảnh tự nhiên. Mục đích của xử lý ảnh gồm:

- Biến đổi ảnh làm tăng chất lượng ảnh.
- Tự động nhận dạng ảnh, đoán nhận ảnh, đánh giá các nội dung của ảnh

+ Nhận biết và đánh giá các nội dung của ảnh là sự phân tích một hình ảnh thành những

phần có ý nghĩa để phân biệt đối tượng này với đối tượng khác, dựa vào đó ta có thể mô tả cấu trúc của hình ảnh ban đầu. Có thể liệt kê một số phương pháp nhận dạng cơ bản như nhận dạng ảnh của các đối tượng trên ảnh, tách cạnh, phân đoạn hình ảnh,... Kỹ thuật này được dùng nhiều trong y học (xử lý tế bào, nhiễm sắc thể), nhận dạng chữ trong văn bản

Lý thuyết xử lí ảnh bao gồm các chủ đề như là sau đây:

Xử lý ảnh số: Là quá trình chuyển đổi các hình ảnh từ định dạng tín hiệu analog sang dạng số để có thể xử lý trên máy tính. Quá trình này bao gồm lấy mẫu, lượng tử hóa, mã hóa và giải mã.

Xử lý hình ảnh: Là quá trình phân tích, biến đổi và trích xuất thông tin từ các hình ảnh. Các kỹ thuật xử lý hình ảnh bao gồm lọc, biến đổi Fourier, biến đổi wavelet, xử lý nhiễu, phát hiện cạnh, trích xuất đặc trưng và phân loại hình ảnh.

Phân tích hình ảnh: Là quá trình tìm kiếm, phát hiện và trích xuất thông tin quan trọng từ các hình ảnh. Các kỹ thuật phân tích hình ảnh bao gồm phát hiện đối tượng, theo dõi đối tượng, nhận dạng khuôn mặt, nhận dạng biển số xe và nhận dạng vật thể.

Ứng dụng xử lý ảnh: Là việc sử dụng các kỹ thuật xử lý ảnh để giải quyết các vấn đề thực tế trong các lĩnh vực như y học, công nghiệp, an ninh, nông nghiệp, điện ảnh và trò chơi điện tử.

Trong thời đại hiện nay, lý thuyết xử lý ảnh đóng vai trò quan trọng trong nhiều lĩnh vực khác nhau và đang được sử dụng rộng rãi trong các ứng dụng thực tế.

⇒ Vì thế, việc xây dựng ứng dụng lọc ảnh trên thiết bị di động coi là một phần quan trọng, một số những ứng dụng vào trong thực tế như các ứng dụng chỉnh sửa ảnh, lọc ảnh, ... cho phép người dùng tương tác, làm tăng tính thẩm mỹ của ảnh.

1.2 Phương hướng xây dựng

+ Xây dựng chương trình xử lí ảnh trên Python

+ Xây dựng chương trình đọc/lưu ảnh trên Android Studio

+ Xây dụng chương trình giải nén, mã hóa ảnh thông qua thư viện trung gian trong Android Studio để Python đọc được ảnh trên di động

1.3 Kết quả

Việc xử lý ảnh trên thiết bị di động nhằm xử lí ảnh đầu vào và ảnh sau khi áp dụng xử lý sẽ nhận được và lưu trong file kết quả

Chương 2 CƠ SỞ LÝ THUYẾT

2.1 Lí thuyết

Lọc ảnh trên miền không gian là quá trình thay đổi giá trị của các điểm ảnh trong không gian hình ảnh để cải thiện chất lượng của ảnh.

+ Một bộ lọc ảnh bao gồm:

(1) 1 mặt nạ (thường là 1 hình chữ nhật nhỏ)

(2) 1 phép toán được định nghĩa trước thực hiện trên các điểm ảnh được phủ bởi mặt nạ

+ Quá trình lọc ảnh gồm các bước sau:

(1) Xác định điểm trung tâm của mặt nạ

(2) Tại điểm (x,y) đang xét (trung với tâm mặt nạ) thực hiện các phép toán lọc trên các điểm lân cận(bị mặt nạ bao phủ)

(3) Ghi nhận kết quả phép lọc là giá trị mức xám của điểm ảnh (x,y) trong ảnh đầu ra

(4) Lần lượt trượt mặt nạ tới những điểm chưa xét.Lặp lại bước (2).

(5) Quá trình lọc kết thúc khi điêm trung tâm của mặt nạ lần lượt thăm hết từng điểm ảnh của ảnh đầu vào , ta thu được kết quả ảnh đã được lọc ở đầu ra

Ví dụ Tại điểm (x,y) nào của ảnh, đáp ứng g(x,y) của bộ lọc là tổng các tích của hệ số lọc và giá tri các điem ảnh bao phủ bởi kernel(mặt nạ,nhân)

$$G(x,y) = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + w(0,0)f(x,y) + .. + w(1,1)f(x+1,y+1)$$

Dựa vào phép toán áp dụng trên, ta áp dụng được trên các bộ lọc, tại bài báo cáo em xin trình bày gồm các bộ lọc trong miền không gian và áp dụng bằng các phép toán đã được trình bày ở trên mà cụ thể qua thiết bị di động

2.2 Bộ lọc trong miền không gian:

2.1.1 Khái niệm:

Bộ lọc trong miền không gian (hay còn gọi là bộ lọc không gian) là một kỹ thuật xử lý tín hiệu được sử dụng trong xử lý ảnh và video. Nó thực hiện bằng cách áp dụng một bộ lọc (hay còn gọi là kernel) trên toàn bộ hoặc một phần của ảnh hoặc video, tạo ra một ảnh hoặc video mới với các tính chất khác nhau.

Bộ lọc trong miền không gian thường được sử dụng để thực hiện các tác vụ như làm mịn, làm nét, phát hiện cạnh, tăng cường độ tương phản, và phân tích hình ảnh. Các bộ lọc có thể được thiết kế để tìm kiếm các đặc trưng cụ thể trong ảnh hoặc video, giúp xác định các đối tượng, vật thể, hoặc các đặc điểm khác trong hình ảnh.

Các bộ lọc trong miền không gian thường được xác định bằng cách chọn các tham số như kích thước của kernel, hệ số tham số, và hướng của kernel. Các bộ lọc khác nhau có thể được áp dụng để thực hiện các tác vụ khác nhau trên ảnh hoặc video

2.1.2 Một số bộ lọc:

+ Bộ lọc mờ Guassian:

Bộ lọc Gaussian là một loại bộ lọc thông thấp được sử dụng trong xử lý tín hiệu và xử lý ảnh để làm mờ hoặc làm trơn ảnh bằng cách loại bỏ các chi tiết nhỏ và nhiễu. Bộ lọc Gaussian được đặt tên theo nhà toán học Carl Friedrich Gauss, người đã phát triển hàm Gaussian, một hàm mượt và liên tục.

Cách thức hoạt động của bộ lọc Gaussian là sử dụng ma trận Gaussian kernel (hay còn gọi là ma trận kernel) để làm trơn ảnh. Ma trận kernel là một ma trận bán kính n x n (n là số lẻ) với các giá trị được tính bằng hàm Gaussian. Khi bộ lọc Gaussian được áp dụng lên ảnh, mỗi điểm ảnh sẽ được tính trung bình trọng số của các điểm ảnh lân cận xung quanh nó, trong đó trọng số được xác định bằng giá trị của ma trận kernel. Kết quả là ảnh được làm mờ và nhiễu được giảm đi, tạo ra một bức ảnh mượt và dễ nhìn hơn. Bộ lọc Gaussian thường được sử dụng trong các ứng dụng xử lý ảnh như nhận diện khuôn mặt, phát hiện đối tượng và xử lý ảnh y tế.

Công thức:

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

+ Bộ lọc Linear Sharpening

Bộ lọc Unsharp Masking là một kỹ thuật xử lý ảnh phổ biến được sử dụng để làm tăng độ nét và độ tương phản của ảnh. Bộ lọc này hoạt động bằng cách tạo ra một bản sao mờ của ảnh gốc, sau đó trừ bản sao mờ này từ ảnh gốc để tạo ra một ảnh kết quả có độ nét và độ tương phản tốt hơn. Là bộ lọc làm mịn đồng thời làm mất đi cá thành phần tần số cao của ảnh gốc nên một bản sắc nét của ảnh có thể nhận được bằng cách cộng ảnh gốc với hiệu ứng giữa nó và ảnh bị mờ theo công thức:

$$I_{sharp} = I + \gamma(I - I_{smooth})$$

+ Ứng dụng của linear sharpening:

Tăng độ nét và độ tương phản của ảnh: Linear sharpening có thể được sử dụng để làm rõ các chi tiết trong ảnh, tăng độ nét và độ tương phản của ảnh. Điều này giúp làm rõ các đối tượng trong ảnh và cải thiện chất lượng ảnh.

Cải thiện chất lượng ảnh được chụp bằng điện thoại di động hoặc máy ảnh kém chất lượng: Linear sharpening có thể giúp cải thiện chất lượng của ảnh được chụp bằng các thiết bị di động hoặc máy ảnh kém chất lượng. Nó giúp làm rõ các chi tiết và tăng độ tương phản của ảnh, giúp ảnh trở nên sắc nét hơn.

Cải thiện chất lượng ảnh được scan từ hình ảnh cũ: Linear sharpening có thể được sử dụng để cải thiện chất lượng của ảnh được scan từ hình ảnh cũ. Khi scan hình ảnh cũ, ảnh có thể bị mờ hoặc không rõ ràng. Linear sharpening có thể giúp làm rõ các chi tiết trong ảnh và tăng độ tương phản, giúp ảnh trở nên rõ ràng và sắc nét hơn.

Tạo hiệu ứng nghệ thuật: Linear sharpening cũng có thể được sử dụng để tạo ra các hiệu ứng nghệ thuật trên các bức ảnh. Khi áp dụng kỹ thuật này với mức độ tăng cường độ nét đáng kể, ảnh có thể trở nên rất sắc nét và tạo ra một cảm giác động đậm hoặc chuyển động.

+ Bộ lọc Sắc nét ảnh

Bộ lọc sắc nét ảnh (sharpen filter) là một loại bộ lọc trong xử lý ảnh được sử dụng để làm tăng độ sắc nét của ảnh. Bộ lọc này thường được sử dụng để cải thiện chất lượng hình ảnh bị mờ hoặc không rõ nét.

Cách thức hoạt động của bộ lọc sắc nét ảnh là tăng độ tương phản của các cạnh trong ảnh bằng cách áp dụng một kernel hoặc mặt nạ phù hợp lên ảnh gốc. Kernel này thường có các giá trị âm và dương, với giá trị tại trung tâm lớn hơn các giá trị xung quanh. Bằng cách này, các cạnh trong ảnh sẽ được tăng độ sắc nét và trở nên rõ ràng hơn.

Tuy nhiên, bộ lọc sắc nét ảnh cũng có một số hạn chế. Nếu áp dụng quá nhiều, nó có thể gây ra hiện tượng nhiễu và các chi tiết không mong muốn trong ảnh.

Vì vậy, để đạt được kết quả tốt nhất, cần sử dụng một số kỹ thuật khác nhau để điều chỉnh thông số của bộ lọc và điều chỉnh cường độ của nó cho phù hợp với ảnh cụ thể. và nó sử dụng các bộ lọc dùng nhân có tổng các phần tử bằng 1:

-1	-1	-1
-1	9	-1
-1	-1	-1

1	1	1
1	-7	1
1	1	1

+ Bộ lọc Emboss(chạm nổi)

Bộ lọc (emboss) dập nổi, còn được gọi là bộ lọc chênh lệch hướng, tăng cường các cạnh theo hướng của (các) mặt nạ tích chập đã chọn. Khi bộ lọc chạm nổi được áp dụng, ma trận bộ lọc được tính toán tích chập với cùng một diện tích hình vuông trên ảnh gốc.

Vì vậy, nó liên quan đến một lượng lớn tính toán khi kích thước hình ảnh hoặc kích thước mặt nạ bộ lọc dập nổi lớn. Bộ lọc dập nổi lặp lại phép tính như được mã hóa trong ma trận bộ lọc cho mọi pixel trong ảnh; quy trình tự so sánh các pixel lân cận trên hình ảnh, để lại dấu nơi phát hiện thấy sự thay đổi rõ rệt về giá trị pixel.

Theo cách này, các dấu tạo thành một đường theo đường viền của đối tượng. Quá trình tạo ra một hình ảnh nổi với các cạnh được đánh dấu.
Một số nhân (kernel) được sử dụng để tạo thành hiệu ứng:

-1	-1	0
-1	0	1
0	1	1

-2	-1	0
-1	1	1
0	1	2

-1	-1	-1	-1	0
-1	-1	-1	0	1
-1	-1	0	1	1
-1	0	1	0	1
0	1	1	1	1

Ứng dụng của emboss có thể kể đến như là:

- Tạo hiệu ứng nổi bật trên ảnh: Bộ lọc Emboss filter được sử dụng để tạo ra hiệu ứng nổi bật trên ảnh, giúp làm nổi bật các đường viền và tạo ra các chi tiết 3D trên hình ảnh.
- Xử lý ảnh trong đồ họa máy tính: Bộ lọc này có thể được sử dụng để tạo ra hiệu ứng trên các đối tượng trong đồ họa máy tính như vật liệu kim loại, đồ vật, vật thể và cả nhân vật.

Bộ lọc Motion blur

Là bộ lọc trong những loại bộ lọc được sử dụng trong xử lý ảnh. Nó được thiết kế để mô phỏng hiệu ứng mờ khi chụp ảnh chuyển động. Khi một đối tượng chuyển động nhanh trên khung hình, nó có thể làm cho hình ảnh trông mờ và không rõ ràng.
Bộ lọc motion blur được sử dụng để xử lý ảnh bị mờ do chuyển động bằng cách tạo ra các hiệu ứng mờ giả tạo, từ đó giúp cho ảnh trông rõ ràng và sắc nét hơn. Bộ lọc này sử dụng một số kỹ thuật tính toán để làm mờ ảnh, bao gồm việc áp dụng bộ lọc Gaussian và thực hiện biến đổi Fourier trên ảnh.
Các bộ lọc motion blur thường được sử dụng trong các ứng dụng xử lý ảnh, ví dụ như xử lý ảnh chụp chuyển động hoặc xử lý video để tạo ra hiệu ứng chuyển động mượt mà và tự nhiên.
Kernel của motion blur là một ma trận số được sử dụng để thực hiện bộ lọc motion blur trên ảnh. Kernel này được thiết kế để tạo ra hiệu ứng mờ theo hướng chuyển động. Kernel thường có kích thước lẻ và được xác định bởi một tham số gọi là độ dài của hiệu ứng mờ.
Kernel motion blur thường có dạng một ma trận vuông kích thước lẻ, có giá trị trung tâm cao hơn các giá trị ở các vị trí xung quanh, như một hình chữ T. Các giá trị ở các vị trí xung quanh của trung tâm của kernel thường giảm dần và có thể được tính toán bằng cách sử dụng một hàm mũ hoặc một hàm tuyến tính.

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \left(\frac{1}{9}\right)$$

+ Một số ứng dụng của motion blur:
Tạo hiệu ứng chuyển động: Motion blur có thể được sử dụng để tạo ra hiệu ứng chuyển động trong các bức ảnh. Điều này giúp tăng tính động đậm và tạo ra một ảnh động trong suốt thời gian.
Tạo hiệu ứng nghệ thuật: Motion blur cũng có thể được sử dụng để tạo ra hiệu ứng nghệ thuật trên các bức ảnh, cho phép tạo ra các bức ảnh mờ hoặc mịn.

Tăng tính chân thực: Motion blur cũng có thể được sử dụng để tăng tính chân thực của các bức ảnh. Ví dụ, trong các bức ảnh đua xe hoặc thể thao, motion blur có thể tạo ra hiệu ứng của sự di chuyển nhanh chóng và động đậm của các vật thể.Giảm độ nhiễu: Motion blur có thể được sử dụng để giảm độ nhiễu trong các bức ảnh. Khi ảnh chụp trong điều kiện ánh sáng thấp hoặc tốc độ chậm, độ nhiễu có thể xuất hiện. Sử dụng motion blur có thể giúp làm mờ các vết nhiễu này và cải thiện chất lượng của ảnh.

+ Bộ lọc song phương

Là một kỹ thuật xử lý ảnh được sử dụng để làm mờ ảnh mà vẫn giữ lại các đặc tính cục bộ của ảnh. Kỹ thuật này được sử dụng để giảm nhiễu trong ảnh, cải thiện chất lượng ảnh và làm mịn ảnh.

Bộ lọc được thực hiện bằng cách tính trung bình trọng số của các điểm ảnh trong một cửa sổ. Hàm trọng số của bộ lọc này được tính bằng cách sử dụng hai thông số: độ lớn của độ khác biệt trong giá trị của các điểm ảnh và khoảng cách giữa các điểm ảnh. Khi độ lớn khác biệt trong giá trị của các điểm ảnh nhỏ, hàm trọng số có giá trị lớn hơn, và ngược lại. Khi khoảng cách giữa các điểm ảnh nhỏ, hàm trọng số cũng có giá trị lớn hơn, và ngược lại.

Công thức được sử dụng với bộ lọc song phương đó là:

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma s}(\|p - q\|) G_{\sigma r}(|I_p - I_q|) I_q$$

Trong đó với W_p (trọng số) trong công thức sau:

$$W_p = \sum_{q \in S} G_{\sigma s}(\|p - q\|) G_{\sigma r}(|I_p - I_q|)$$

Ứng dụng của bộ lọc song phương

- Giảm nhiễu ảnh: Bộ lọc bilateral filter có thể được sử dụng để giảm nhiễu trong ảnh mà không làm mất các đặc tính cục bộ của ảnh.
- Làm mịn ảnh: Bộ lọc này cũng có thể được sử dụng để làm mịn ảnh, giúp giảm các chi tiết không mong muốn trong ảnh.

- Tăng cường độ sáng và độ tương phản: Bộ lọc bilateral filter có thể được sử dụng để tăng cường độ sáng và độ tương phản của ảnh một cách tối ưu.
- Bộ lọc bilateral filter có nhiều ứng dụng trong xử lý ảnh và đồ họa máy tính. Dưới đây là một số ứng dụng phổ biến của nó

2.3 Các bước xử lý ảnh và lưu file trên thiết bị di động:

+ Lưu hình ảnh hoặc tệp file ảnh cần được xử lý dưới dạng Bitmap và mã hóa chúng sang dạng mảng byte

+ Đọc mảng byte và tiếp tục chuyển sang dạng chuỗi để Python có thể đọc được, việc chuyển sang chuỗi được sử dụng thông qua thư viện trung gian

+ Vì ảnh được đại diện bằng một ma trận NumPy. Mỗi phần tử của ma trận đại diện cho một pixel trong ảnh và có thể có các giá trị từ 0 đến 255 đại diện cho mức xám của pixel đó. Kiểu dữ liệu của ma trận thường là `uint8`, tức là kiểu số nguyên không dấu có kích thước 8 bit. Tuy nhiên, nếu ảnh là ảnh màu, ma trận sẽ có 3 kênh (đại diện cho các kênh màu đỏ, xanh lá cây và xanh lam), và kiểu dữ liệu sẽ là `uint8` với kích thước 24 bit (8 bit cho mỗi kênh). Do đó, Sau khi chuỗi đã được đưa sang String mà Python có thể đọc được, tiến hành việc xử lý ảnh thông qua việc giải mã bằng thư viện Opencv và base64.b64decode trong Python

+ Kết quả thu được hình ảnh sau khi đã được xử lý bằng các phương pháp lọc kể trên, lưu ảnh dưới dạng PNG hoặc JPEG theo đường dẫn bộ nhớ trong của điện thoại

+ Để điện thoại có thể hiện thị ảnh, sử dụng đường dẫn đã cho phép lưu sau khi xử lý ảnh ở Python, đọc file ảnh hoặc ảnh và tiếp tục chuyển sang dạng byte thông qua việc nén ảnh, sau đó Bitmap sẽ tiến hành đọc mảng byte này và hiển thị ảnh trên điện thoại

+ Tạo file/thư mục lưu ở bộ nhớ ngoài(external storage) , sau khi Bitmap đã hiển thị được ảnh, để lưu ảnh thì cần phải nén Bitmap ở dạng PNG hoặc JPEG.Sau khi đã nén xong, cấp quyền và Tạo đối tượng File để lưu ảnh, ảnh sẽ được lưu lại thông qua các đường dẫn thông qua phương thức compress() sẽ chuyển đổi ảnh thành định dạng JPEG và lưu vào bộ nhớ ngoài(external storage).

Chương 3 CÀI ĐẶT VÀ THỬ NGHIỆM

3.1 Công cụ và thư viện sử dụng:

3.1.1 Thư viện sử dụng

OpenCV (Open Source Computer Vision Library) là một thư viện mã nguồn mở chuyên dụng trong lĩnh vực xử lý ảnh và thị giác máy tính. OpenCV được phát triển bằng C++ và có sẵn các liên cho nhiều ngôn ngữ như Python, Java và C#. OpenCV cung cấp nhiều công cụ và thuật toán để xử lý ảnh và video, bao gồm cả xử lý ảnh số, phân tích hình ảnh, nhận dạng đối tượng, theo dõi chuyển động, tìm kiếm và phát hiện các đối tượng trong ảnh.

OpenCV được sử dụng rộng rãi trong các ứng dụng thị giác máy tính như: xử lý ảnh y tế, giám sát an ninh, định vị và xác định đối tượng, xử lý ảnh và video cho các ứng dụng di động và trí tuệ nhân tạo. OpenCV được phát triển bởi Intel vào năm 2000 và hiện nay đã trở thành một trong những thư viện xử lý ảnh phổ biến nhất trên thế giới.

+ Thư viện Chaquopy:

Chaquopy là một thư viện được phát triển dành riêng cho ứng dụng Python trên nền tảng Android. Thư viện này cho phép bạn viết mã Python và chạy trực tiếp trên Android Studio, không cần phải chuyển đổi sang mã Java hoặc Kotlin. Chaquopy được viết bằng ngôn ngữ Java và được phát triển bởi Chaquo Limited.

Các tính năng chính của Chaquopy bao gồm:

- Hỗ trợ đầy đủ Python 3.x, bao gồm các thư viện phổ biến như numpy, pandas, opencv, tensorflow, và các thư viện khác.
- Hỗ trợ các thư viện C/C++ thông qua JNI (Java Native Interface).
- Cho phép tạo các ứng dụng Python độc lập hoàn toàn trên thiết bị Android.
- Hỗ trợ các tính năng đa luồng và quản lý tài nguyên hiệu quả.

Sử dụng Chaquopy giúp giảm thiểu thời gian và công sức cần thiết để phát triển các ứng dụng xử lý dữ liệu phức tạp trên Android. Chaquopy cũng được sử dụng rộng rãi trong các ứng dụng y tế, kỹ thuật, và khoa học.

Thư viện base64 trong Python là một thư viện chuẩn để mã hóa và giải mã các chuỗi dữ liệu dạng base64. Base64 là một phương thức mã hóa được sử dụng để mã hóa các dữ liệu nhị phân thành các ký tự ASCII có thể đọc được.

Thư viện base64 cung cấp các hàm sau:

- `base64.b64encode(s)`: Mã hóa dữ liệu chuỗi s thành base64.
- `base64.b64decode(s)`: Giải mã base64 thành dữ liệu nhị phân.

3.1.2 Công cụ:

+ Android Studio: Android Studio là một môi trường phát triển tích hợp (IDE) dành cho việc phát triển ứng dụng Android. Được phát triển bởi Google, Android Studio được xây dựng trên nền tảng phát triển mã nguồn mở IntelliJ IDEA và có các tính năng và công cụ đáp ứng đầy đủ yêu cầu phát triển ứng dụng Android.

Android Studio cung cấp cho người lập trình Android một số tính năng quan trọng như:

- Trình soạn thảo mã nguồn và giao diện đồ họa để thiết kế các giao diện người dùng (UI) và các thành phần khác cho ứng dụng.
- Các công cụ phân tích mã nguồn, giúp tìm kiếm và sửa lỗi mã nguồn hiệu quả.
- Trình quản lý dự án cho phép người lập trình quản lý mã nguồn và các tài nguyên liên quan đến dự án.
- Trình mô phỏng để kiểm tra và thử nghiệm ứng dụng trên các thiết bị ảo hoặc thiết bị thực.
- Công cụ phát triển ứng dụng cho các nền tảng khác nhau như Android Wear, Android TV, và Android Auto.

Android Studio được sử dụng rộng rãi trong cộng đồng lập trình Android và được đánh giá là một trong những IDE tốt nhất để phát triển ứng dụng Android hiện nay.

3.2 Cài đặt chương trình:

+ Lấy ảnh và chuyển về dạng Bitmap:

```
public Bitmap getImage() throws FileNotFoundException {
    //getURI
    Intent data = getIntent();
    String URI = data.getStringExtra("uri_i");
    InputStream in = getContentResolver().openInputStream(Uri.parse(URI));
    Bitmap bm = BitmapFactory.decodeStream(in);
}
```

+Nén ảnh về dạng Bitmap sang dạng byte[]:

```
public byte[] compress(Bitmap k) {
    ByteArrayOutputStream str = new ByteArrayOutputStream();
    k.compress(Bitmap.CompressFormat.PNG, 100, str);
    return str.toByteArray();
}
```

+ Chuyển byte[] sang dạng String :

```
public String getStringImage(Bitmap bitmap) {
    ByteArrayOutputStream b = new ByteArrayOutputStream();
    bitmap.compress(Bitmap.CompressFormat.PNG, 100, b);
    byte[] imageBytes = b.toByteArray();
    String encodedImage = android.util.Base64.encodeToString(imageBytes,
Base64.DEFAULT);
    return encodedImage;
}
```

+ Hàm xử lý ảnh thông qua thư viện Chaquopy như sau đây

```
public FileInputStream BilateralPython(Bitmap bitmap , int typeKernel , int D1)
{
    Python python = Python.getInstance();
    ModuleFile = python.getModule("script5");
    fileDir = (ModuleFile.callAttr("getfileDIRs")).toString();
    numpy_module = python.getModule("numpy");
    String b = getStringImage(bitmap);
    PyObject k = ModuleFile.callAttr("bilateral_filters", b , typeKernel , D1);

    try {
        f = new FileInputStream(fileDir);
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    return f;
}


public byte[] compress(Bitmap k) {
    ByteArrayOutputStream str = new ByteArrayOutputStream();
    k.compress(Bitmap.CompressFormat.PNG, 100, str);
    return str.toByteArray();
}
```

+ Lấy đường dẫn file trên Python như sau:

```
import os
import io
import matplotlib.pyplot as plt
import numpy
from os.path import dirname, join
import base64
from PIL import Image
import cv2
from com.chaquo.python import Python


#getting the file we needing for provided
file_dir = str(Python.getPlatform().getApplication().getFilesDir())
#get the locate of file where we need to store data that
filename = join(dirname(file_dir) , "matrix_plot.png")
filenamenoisy = join(dirname(file_dir) , "noi_plot.png")
```

+ Đọc String và chuyển sang dạng ma trận (ảnh) để xử lý ảnh:

```
    decoded_data = base64.b64decode(data)
    numpy_data = numpy.fromstring(decoded_data,numpy.uint8)
    img = cv2.imdecode(numpy_data,cv2.IMREAD_UNCHANGED)
    sharpened_image = unsharp_mask(img , sigma=b)
```

+ Hàm lưu file ảnh

```
public void Save(Bitmap image_bitmap, String image_name) {
    String root = Environment.getExternalStorageDirectory().toString();
    if (isStoragePermissionGranted()) { // check or ask permission
        File myDir = new File(root, "/saved_images");
        if (!myDir.exists()) {
            myDir.mkdirs();
        }
        String fname = "Image-" + image_name + ".jpg";
        File file = new File(myDir, fname);
        if (file.exists()) {
            file.delete();
        }
        try {
            file.createNewFile(); // if file already exists will do nothing
            FileOutputStream out = new FileOutputStream(file);
            image_bitmap.compress(Bitmap.CompressFormat.JPEG, 90, out);
            out.flush();
            out.close();

        } catch (Exception e) {
            e.printStackTrace();
        }

        MediaScannerConnection.scanFile(getActivity(), new
String[]{file.toString()}, new String[]{file.getName()}, null);
    }
}
```

3.3 Thử nghiệm chương trình

+ Bộ lọc Guassian


+ Bộ lọc Linear Sharpening:

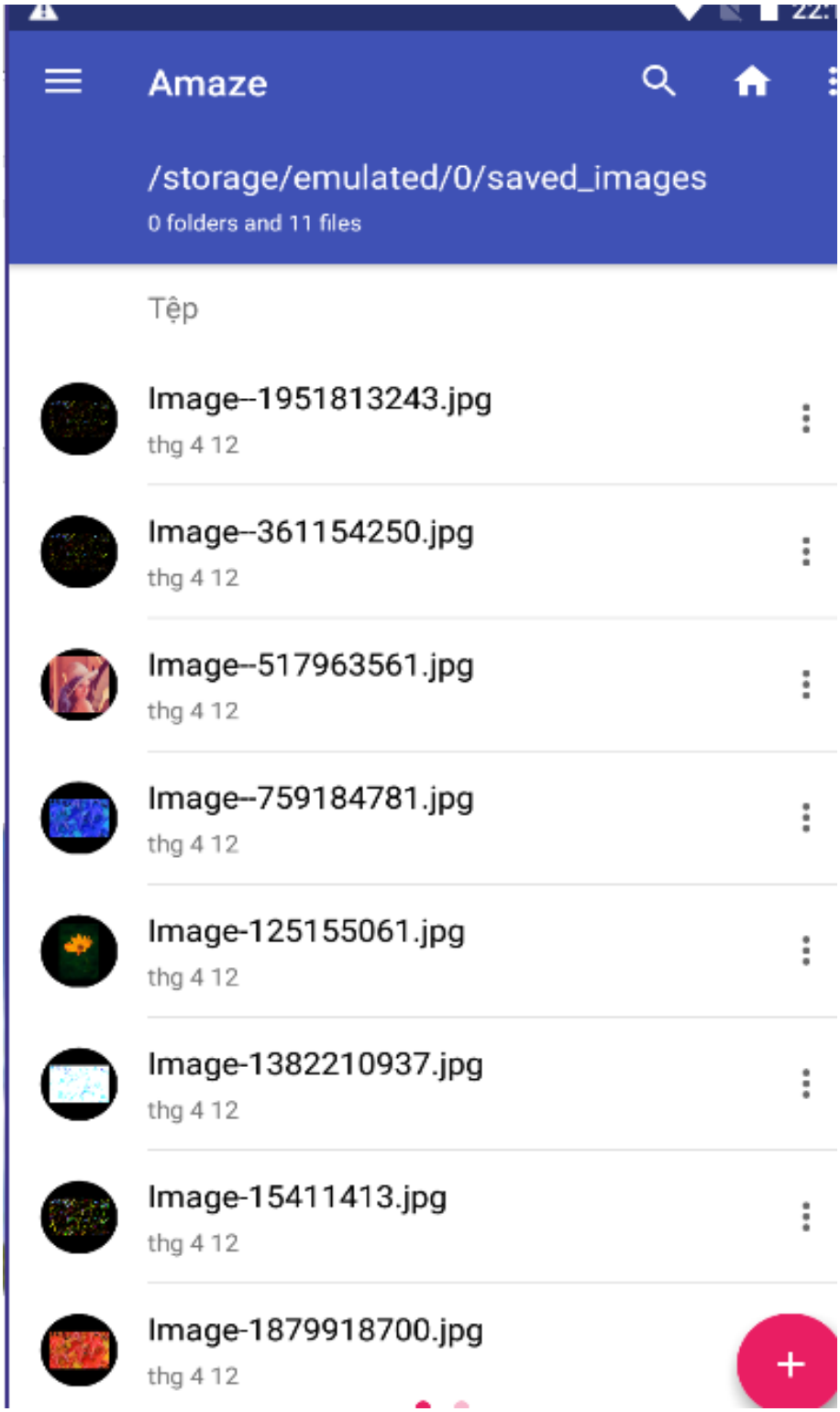

+ Bộ lọc sắc nét


+ Bộ lọc Emboss


+ Bộ lọc Motion blur

+ Bộ lọc song phương

+ File lưu ảnh đã được lọc trên thiết bị di động như sau:



KẾT LUẬN

Thông qua bài báo cáo em nhận được những kết quả để đánh giá cũng như kinh nghiệm và những hạn chế của chương trình trong quá trình thực hiện.

Thông qua việc xây dựng và áp dụng các thuật toán xử lý ảnh phổ biến như: lọc trung bình, lọc Gaussian, lọc trung vị và lọc Linear Sharpening, và lọc bilateral filter, ... Đồng thời, em cũng đã thiết kế giao diện người dùng đơn giản, cho phép người dùng dễ dàng thao tác với các chức năng của chương trình.

Sau khi thực hiện kiểm thử và đánh giá, em nhận thấy rằng chương trình hoạt động ổn định trên các thiết bị di động thông thường và có thể xử lý nhanh các hình ảnh với kích thước và độ phân giải. Tuy nhiên, vẫn còn một số hạn chế cần được cải thiện như tối ưu hóa thời gian xử lý để chương trình có thể hoạt động mượt mà hơn.

Với chức năng lưu ảnh vào file, em đã sử dụng thư viện base64 trong Python để thực hiện việc giải nén và chaquopy thư viện để mà lưu ảnh với định dạng JPEG. Tính năng này giúp người dùng có thể lưu lại những bức ảnh đã được lọc và xử lý bởi chương trình.

Tổng kết lại, em hy vọng rằng chương trình lọc ảnh trên thiết bị di động và lưu ảnh vào file sẽ giúp người dùng có thể dễ dàng chỉnh sửa, cải thiện chất lượng ảnh của mình. Em cũng mong muốn sẽ có những cải tiến và phát triển hơn trong tương lai để mang lại trải nghiệm tốt hơn cho người dùng

Mọi sai sót em mong thầy cô góp ý để ngày càng tốt và hoàn thiện hơn.Em xin chân thành cám ơn thầy cô rất nhiều.