ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN
Chủ đề:
EDGE DETECTION
& PENCIL PHOTO SKETCHING

Sinh viên

Võ Nhật Thanh -19522245

Lê Vinh Quang-19522093

Trần Trung Tín-19522351

MỤC LỤC

I.GIỚI THIỆU

Trong những năm gần đây Computer Vision là một lĩnh vực mang tính khoa học và công nghệ và là đề tài được nhiều người quan tâm nghiên cứu phát triển, ứng dụng. Edge detection (phát hiện cạnh) là một bài toán nổi trội trong lĩnh vực này và được ứng dụng vào nhiều lĩnh vực trong cuộc sống như: lĩnh vực hình ảnh y tế (medical imaging), phát hiện và nhận dạng đối tượng, hệ thống camera giám sát, hệ thống điều khiển giao thông… Kỹ thuật này là bước tiền xử lý quan trọng trong hầu hết các hệ thống xử lý ảnh, kết phân vùng tốt sẽ giúp cho quá trình xử lý về sau đạt hiệu quả cao hơn nhằm tiết kiệm về chi phí tính toán, thời gian cũng như tăng độ chính xác của các ứng dụng trên.

Trong đồ án này chúng tôi tìm hiểu và giúp các bạn hiểu rõ một phần nhỏ trong bài toán phát hiện cạnh và phương pháp phác thảo ảnh. Đầu vào của bài toán sẽ là một bức ảnh màu sau đó sẽ được chuyển thành ảnh mức xám, sau đó sẽ là tính toán gradient , sử dụng kernel và kết quả trả về là một ảnh gồm các cạnh được xử lí.

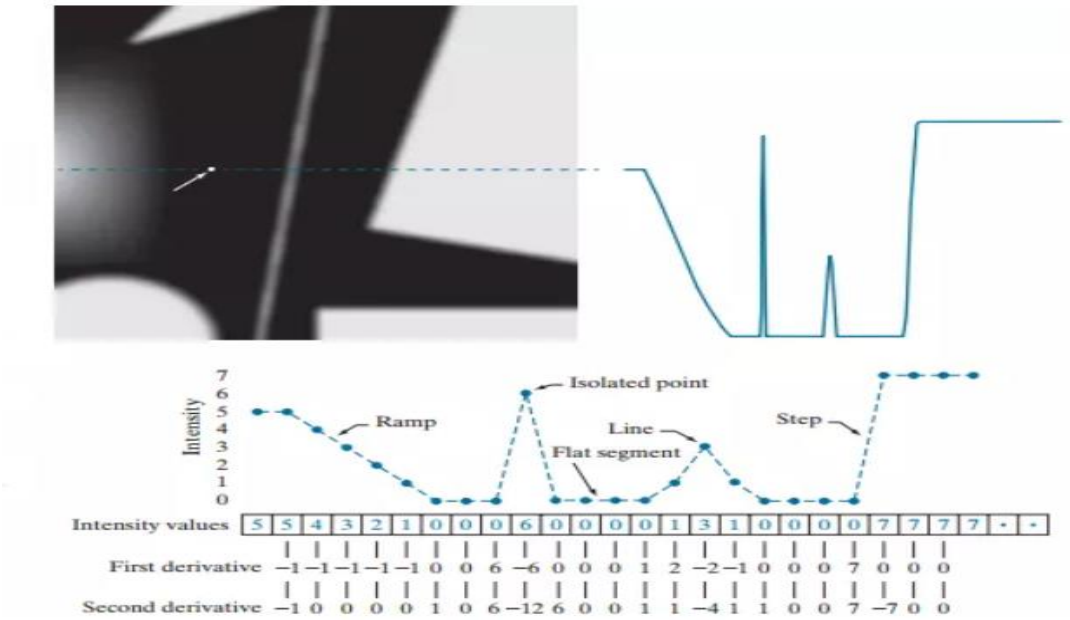
II. EDGE DETECTION

1. 2.1. Edge?

Điểm cạnh: Một điểm ảnh được coi là điểm cạnh nếu có sự thay đổi nhanh hoặc đột ngột về mức xám (hoặc màu).

Ví dụ: Trong ảnh nhị phân, điểm đen được gọi là điểm biên nếu lân cận của nó có ít nhất một điểm trắng.

Cạnh/đường bao (boundary): Là tập hợp các điểm biên liên tiếp.



Các loại cạnh:

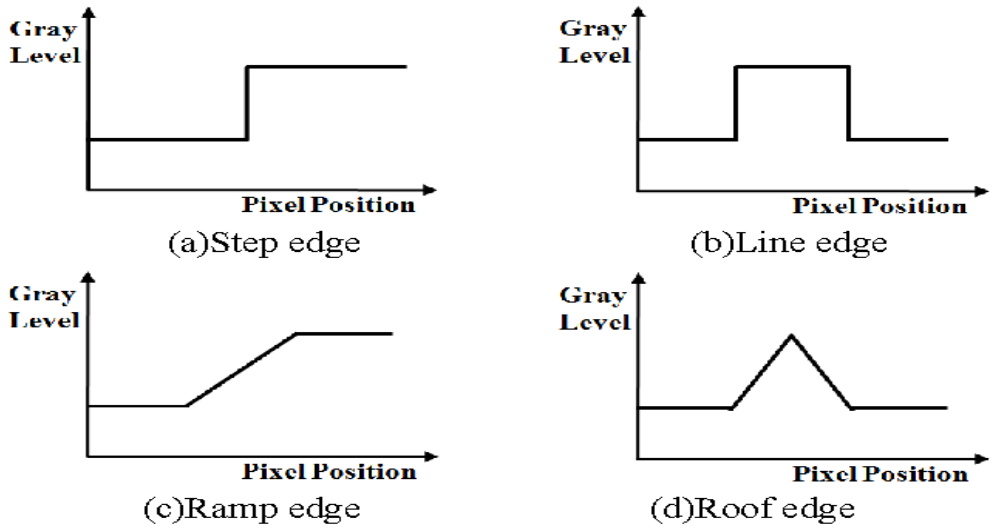


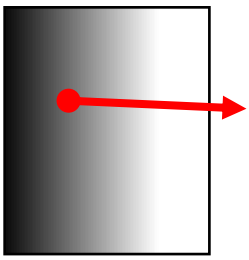
Figure 1. Four types of edge

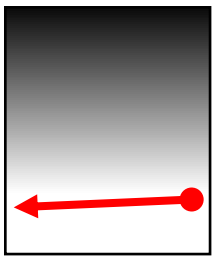
2. 2.2. Gradient

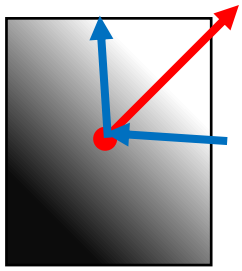
Gradient là một vectơ có các thành phần biểu thị tốc độ thay đổi mức xám của điểm ảnh (theo hai hướng *x*, *y* đối với ảnh 2 chiều) :

Gradient theo hướng (theo phương nằm ngang)

Gradient theo hướng (theo phương nằm đứng)







Độ lớn $|\nabla f(x,y)| = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2}$

Thông thường, trong miền rời rạc, độ lớn của gradient được tính xấp xỉ.

Công thức tính xấp xỉ đạo hàm

$$\frac{\partial f(x,y)}{\partial x} = f'_x(x,y) \approx \frac{f(x+dx,y) - f(x,y)}{dx},$$

$$\frac{\partial f(x,y)}{\partial y} = f'_y(x,y) \approx \frac{f(x,y+dy) - f(x,y)}{dy}$$

Trong đó *dx*, *dy* là khoảng cách giữa 2 điểm kế cận theo hướng x,y tương ứng (thực tế chọn *dx*=*dy*=1)

Khi nhắc đến sự thay đổi đổi thì ta thường nhắc tới Gradient. Gradient của hàm cho biết hàm tăng mạnh như thế nào:

Ví dụ với hàm 1 chiều: f(x) = x^2thì Gradient của nó được ký hiệu và tính như sau:

Grad(x) = $\frac{\partial f(x)}{\partial (x)} = 2x$

- Grad (2) = 4 chỉ ra hướng tăng của hàm là bên phải
- Grad (-1) = - 2 chỉ ra hướng tăng của hàm nằm ở bên trái

1.2 2.3. Sobel filter

-Robert filter

$$|\nabla f(x,y)| \approx \sqrt{(z_6 - z_5)^2 + (z_8 - z_5)^2}$$

$$= \sqrt{(z_5 - z_8)^2 + (z_5 - z_6)^2} \quad = \sqrt{([z_5\ z_8\ ] \otimes [1\ -1\ ])^2 + ([z_5\ z_6\ ] \otimes [1\ -1\ ])^2}$$

Công thức trên xấp xỉ độ lớn của gradient sử dụng

hai Robert kernels là: $H_1 = [1\ -1\ ]$ và $H_2 = [1\ -1\ ]$

Hay có thể viết: $|\nabla f(x,y)| \approx \sqrt{(f \otimes H_1)^2 + (f \otimes H_2)^2}$

Một công thức xấp xỉ độ lớn của gradient tốt hơn là:

$$|\nabla f(x,y)|$$

$$= \sqrt{\left((z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)\right)^2 + \left((z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)\right)^2}$$

$$= \sqrt{([z_1\ z_2\ z_3\ z_4\ z_5\ z_6\ z_7\ z_8\ z_9\ ] \otimes [-1\ 0\ 1\ -2\ 0\ 2\ -1\ 0\ 1\ ])^2 + ([z_1\ z_2\ z_3\ z_4\ z_5\ z_6\ z_7\ z_8\ z_9\ ] \otimes}$$

Công thức xấp xỉ trên sử dụng hai Sobel kernels là:

$G_x = [-1\ 0\ 1\ -2\ 0\ 2\ -1\ 0\ 1\ ]$ và $G_y = [-1\ -2\ -1\ 0\ 0\ 0\ 1\ 2\ 1\ ]$

Là phương pháp dò tìm cạnh ảnh phụ thuộc vào hướng và đối xứng trục, thường sử dụng các Sobel kernels có kích thước 3×3**.**

Sobel kernel xác định Gradient chiều ngang (Xác định cạnh đứng)

$G_x = [-1\ 0\ 1\ -2\ 0\ 2\ -1\ 0\ 1\ ]$ $\qquad SV = I \otimes G_x$

Sobel kernel xác định Gradient chiều đứng (Xác định cạnh ngang)

$G_y = [-1\ -2\ -1\ 0\ 0\ 0\ 1\ 2\ 1\ ]$ $SH = I \otimes G_y$

Mỗi điểm của ảnh kết quả được xác định bằng độ lớn của Gradient

$$D(x,y) = \sqrt{SH(x,y)^2 + SV(x,y)^2}$$

Kích thước ảnh kết quả

$$\frac{N-3}{1} + 1 = N - 2 < N$$

Sử dụng Zero-padding để giữ nguyên kích thước của ảnh kết quả so với ảnh gốc

Thêm "viền số 0" bao quanh ảnh gốc:

[0 0 10 10 10 0 0 10 10 10 0 0 10 10 10 0 0 10 10 10 0 0 10 10 10]

[0 0 0 0 0 0 0 0 0 0 10 10 10 0 0 0 0 10 10 10 0 0 0 0 10 10 10 0 0 0 0 10 10 10

Kích thước ảnh kết quả:

$$\frac{N-3+2\times 1}{1} + 1 = N$$

3. 2.4. Laplacian filter

3 loại hạt nhân thường dùng:

$$H_1 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \quad H_2 = \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{pmatrix} \quad H_3 = \begin{pmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{pmatrix}$$

Nhược điểm :

- Nhạy cảm với nhiễu
- Các đường biên thu được thường kém ổn định

Khắc phục :

- Dùng thêm hàm Gauss để giảm nhiễu cho ảnh (làm trơn ảnh)

Ví dụ các bước trên Python :

- Tải ảnh lên
- Khử nhiễu bằng bộ lọc Gauss và chuyển ảnh sang ảnh xám

```
src = cv.GaussianBlur(src, (7, 7),0)
src_gray = cv.cvtColor(src, cv.COLOR_BGR2GRAY)
```

- Áp dụng Laplacian filter

```
dst = cv.Laplacian(src_gray, ddepth, ksize=kernel_size)
```

4. 2.5. Canny edge detection

Phương pháp này do John Canny ở phòng thí nghiệm MIT khởi xướng vào năm 1986. Canny đã đưa ra một tập hợp các ràng buộc mà một phương pháp phát hiện biên phải đạt được. Ông đã trình bày một phương pháp tối ưu nhất để thực hiện được các ràng buộc đó. Và phương pháp này được gọi là phương pháp Canny. Bây giờ chúng ta sẽ tìm hiểu hoạt động và triển khai của thuật toán này

Các bước cơ bản liên quan đến thuật toán này:

- Giảm nhiễu bằng bộ lọc Gaussian
- Tính toán độ dốc dọc theo trục ngang và trục dọc
- Non-Maximum Suppression
- Định ngưỡng độ trễ

1. Giảm nhiễu bằng cách sử dụng bộ lọc Gaussian

Bước này là cực kỳ quan trọng trong canny edge detection. Nó sử dụng bộ lọc Gaussian để loại bỏ nhiễu khỏi hình ảnh, đó là bởi vì nhiễu này có thể được coi là các cạnh do sự thay đổi cường độ đột ngột của máy dò cạnh. Tổng các phần tử trong hạt nhân Gaussian là 1, do đó hạt nhân phải được chuẩn hóa trước khi áp dụng dưới dạng tích chập cho hình ảnh.

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

Trong đoạn code này, chúng ta sử dụng một nhân có kích thước 5 X 5 và sigma=1.4, sẽ làm mờ hình ảnh và loại bỏ nhiễu khỏi nó.

```
# Noise reduction step
img = cv2.GaussianBlur(img, (5, 5), 1.4)
```

2. Tính gradient

Khi hình ảnh được làm mịn, các đạo hàm Ix và Iy được tính theo trục x và y. Nó có thể được thực hiện bằng cách sử dụng tích chập các hạt nhân Sobel-Feldman với hình ảnh như đã cho:

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, K_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}.$$

Hạt nhân Sobel

Sử dụng 2 hạt nhân: 1 hạt nhân để thay đổi theo chiều ngang và một hạt nhân

để thay đổi theo chiều dọc.

Sau khi áp dụng Kernel này, chúng ta có thể sử dụng độ lớn của gradient và góc để tiếp tục xử lý bước này. Độ lớn và góc có thể được tính như sau:

$$|G| = \sqrt{I_x^2 + I_y^2},$$
$$\theta(x, y) = arctan\left(\frac{I_y}{I_x}\right)$$

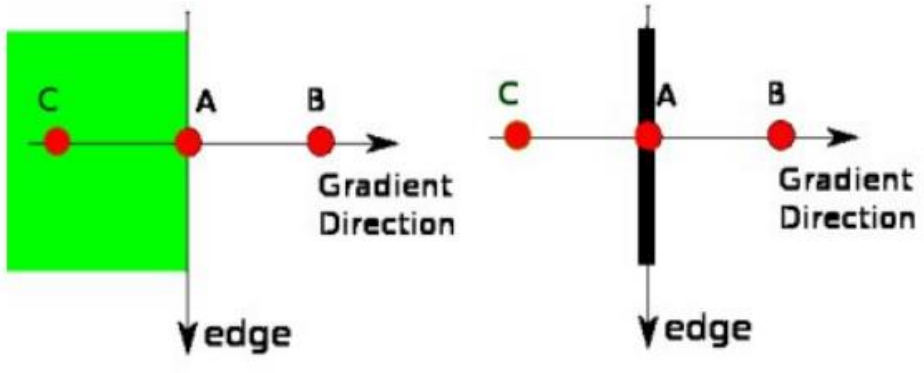
Độ lớn và góc của gradient

Độ lớn của gradient được sử dụng để đo mức độ thay đổi của tương quang ảnh.

Góc của gradient được sử dụng để xác định hướng thay đổi của cường độ đang hướng đến.

3. Non-Maximum Suppression

Sau khi nhận được độ lớn và hướng của gradient, quá trình quét toàn bộ hình ảnh được thực hiện để loại bỏ bất kỳ pixel không mong muốn nào có thể không tạo thành cạnh. Đối với điều này, tại mỗi pixel được kiểm tra xem nó có phải là cực đại cục bộ trong vùng lân cận của nó theo hướng gradient hay không. Nếu là cực đại lân cận, ta sẽ ghi nhận sẽ giữ pixel đó lại. Còn nếu pixel tại đó không phải là cực đại lân cận, ta sẽ set độ lớn gradient của nó về zero. Ta chỉ so sánh pixel trung tâm với 2 pixel lân cận theo **hướng gradient.**



hình ảnh

Điểm A nằm trên cạnh (theo phương thẳng đứng). Gradient hướng bình thường đối với cạnh. Điểm B và C theo hướng gradient. Vì vậy, điểm A được kiểm tra

với điểm B và C để xem nó có tạo thành cực đại cục bộ hay không. Nếu là cực đại cục bộ nó được xem xét cho giai đoạn tiếp theo, nếu không nó sẽ bị loại bỏ (đưa về 0).

Tóm lại, kết quả ta nhận được là một ảnh nhị phân với "các cạnh mỏng".

4. Định ngưỡng độ trễ

Ngay cả sau khi áp dụng Non-Maximum Suppression, chúng ta có thể cần phải loại bỏ các vùng của hình ảnh không phải là cạnh về mặt kỹ thuật, nhưng vẫn phản hồi dưới dạng các cạnh sau khi tính toán độ lớn gradient và dụng Non-Maximum Suppression.

Để bỏ qua những vùng này của hình ảnh, chúng ta cần xác định hai ngưỡng: T_{upper} và T_{lower}.

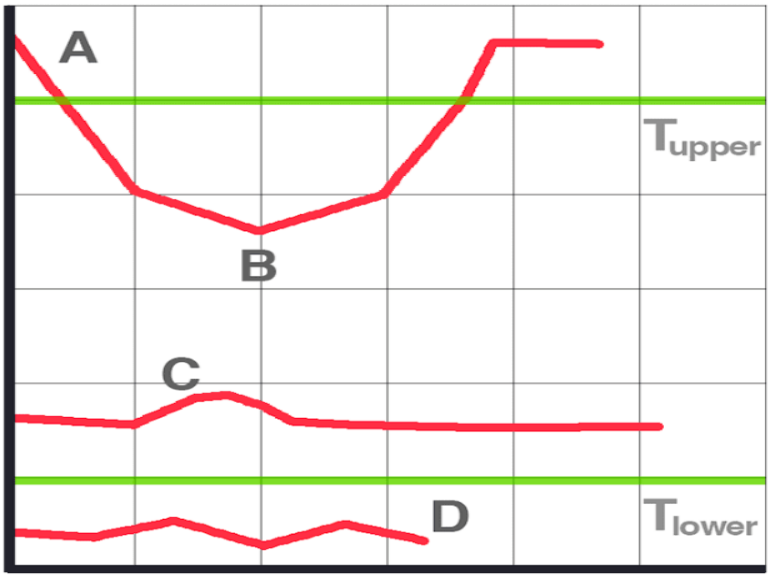
Bất kỳ giá trị gradient nào $G > T_{upper}$ chắc chắn là cạnh.

Bất kỳ giá trị gradient nào $G < T_{lower}$ không phải là cạnh, vì vậy loại bỏ ngay.

Và bất kỳ giá trị gradient nào nằm trong phạm vi $T_{lower} < G < T_{upper}$ cần phải được xem xét rằng:

1. Nếu giá trị gradient cụ thể được kết nối với một **cạnh mạnh** (tức là $G > T_{upper}$), sau đó đánh dấu pixel là một cạnh.
2. Nếu pixel gradient không được kết nối với một cạnh mạnh, thì loại bỏ nó.

Ngưỡng trễ thực sự được giải thích trực quan tốt hơn:



- Ở trên cùng của biểu đồ, chúng ta có thể thấy rằng **A** là một cạnh chắc chắn, vì $A > T_{upper}$.
- **B** cũng là một cạnh, mặc dù $B < T_{upper}$ kể từ khi nó được kết nối với một lợi thế cạnh mạnh.
- **C** không phải là một cạnh kề từ $C < T_{upper}$ và không được kết nối với một cạnh mạnh.
- Cuối cùng, **D** không phải là một cạnh vì $D < T_{lower}$ và tự động bị loại bỏ.

Đặt các phạm vi ngưỡng này không phải lúc nào cũng là một quá trình nhỏ.

Nếu phạm vi ngưỡng quá rộng, thì chúng ta sẽ nhận được nhiều cạnh giả thay vì chỉ cần tìm thấy cấu trúc và đường biên của một đối tượng trong hình ảnh.

Tương tự, nếu phạm vi ngưỡng quá chặt chẽ, chúng ta sẽ không tìm thấy nhiều cạnh và có thể có nguy cơ bỏ lỡ hoàn toàn cấu trúc/đường viền của đối tượng.

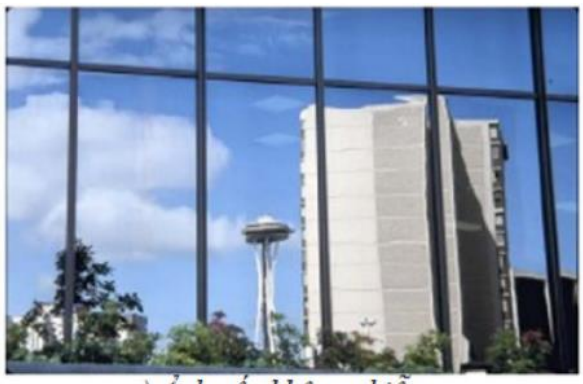
Hình ảnh đầu vào

Hình ảnh đầu ra

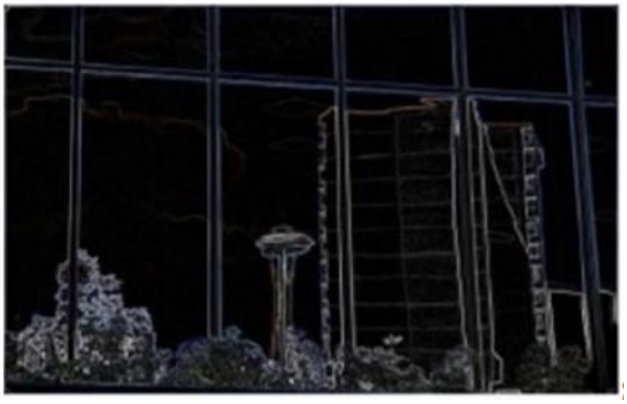
5. 2.6. So sánh 2 pp Sobel và Canny

1. Đối với ảnh không nhiễu

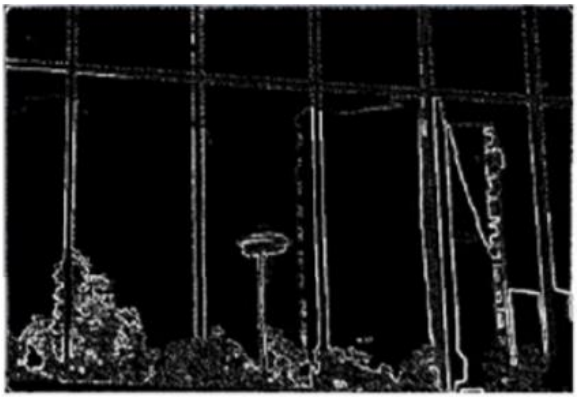
Cả hai phương pháp đều cho kết quả tốt. Song phương pháp phát hiện biên Sobel cho biên rõ nét nhưng lớn. Phương pháp Canny do quá trình làm trơn ảnh nên từ một ảnh không nhiễu, các biên mờ bớt đi và to ra. Do vậy biên ảnh trong phương pháp Canny lớn nhưng lại không đầy đủ. Đối với loại ảnh này khi tìm biên không áp dụng phương pháp Canny.



a) Ảnh gốc không nhiễu



Sobel



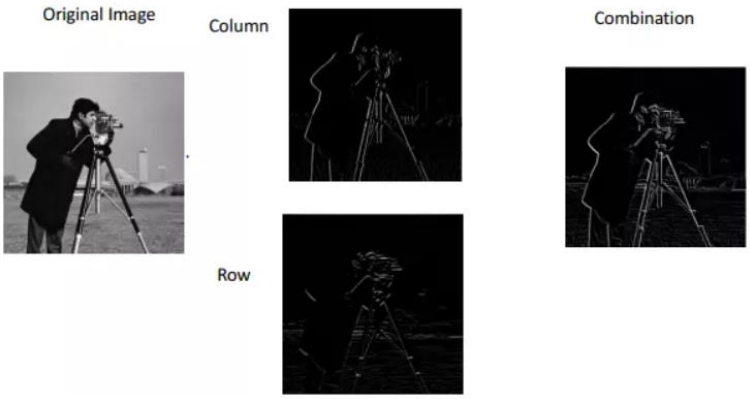
Canny

2. Đối với ảnh có nhiều cạnh

Khi phát hiện biên, các cạnh không quan trọng nên được loại bỏ. Ở đây, phương pháp Sobel vẫn phát hiện được biên nhưng các biên mở, không được rõ nét, do trong ảnh có những vùng có mức xám thấp, sự thay đổi giữa các mức xám nhỏ. Chính vì vậy mà ảnh qua phương pháp Laplace cho kết quả rõ nét hơn (do phương pháp này sử dụng phương pháp đạo hàm bậc hai, các điểm biên là các điểm cắt không). Tuy vậy do ảnh có nhiều điểm biên nhỏ nên các biên ảnh ở trên qua phương này nhiều và rối, chúng ta nên loại bỏ các điểm biên thừa. Còn đối với phương pháp Canny, do quá trình "Non-Maximum Suppression" và do quá trình áp dụng ngưỡng mà các điểm biên phụ bị loại bớt đi, cá biên chính được giữ lại nên biên rõ nét hơn. Đối với ảnh có nhiều mức xám nhỏ, sự biến thiên các mức xám là thấp ta nên sử dụng phương pháp Laplace, song nếu ảnh đó có quá nhiều biên thì ta nên sử dụng phương pháp Canny để loại bỏ bớt các cạnh không cần quan tâm.

2.7. Ưu điểm và hạn chế của các pp

1. Sobel filter



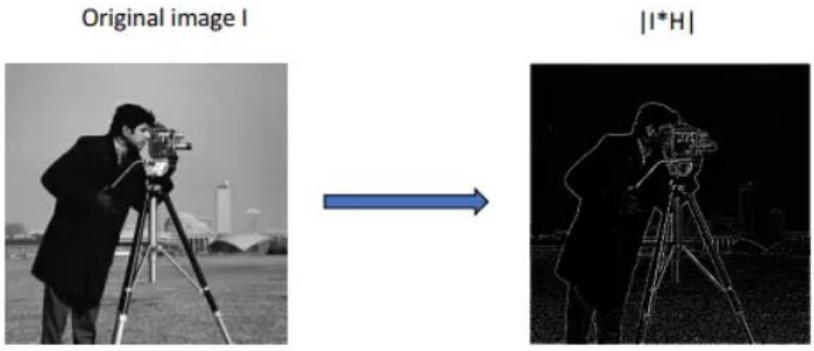
Ưu điểm:

1. Tính toán đơn giản và hiệu quả về thời gian.
2. Rất dễ dàng tìm kiếm các cạnh nhẵn.

Hạn chế:

1. Các điểm hướng theo đường chéo không phải lúc nào cũng được bảo toàn.
2. Rất nhạy cảm với độ nhiễu.
3. Không chính xác lắm trong phát hiện cạnh.
4. Phát hiện với các cạnh dày và thô không cho kết quả thích hợp.

2. Laplace filter



Ưu điểm:

1. Dễ dàng phát hiện các cạnh và các hướng khác nhau của chúng.
2. Có các đặc điểm cố định ở tất cả các hướng.

Hạn chế:

2. Rất nhạy cảm với độ nhiễu.
3. Lỗi bản địa hóa có thể nghiêm trọng ở các cạnh cong.
4. Nó tạo ra các phản hồi nhiễu không tương ứng với các cạnh, được gọi là "các cạnh giả".

3. Canny filter

Ưu điểm:

1. Nó có bản địa hóa tốt.
2. Nó trích xuất các tính năng hình ảnh mà không làm thay đổi các tính năng.
3. Ít nhạy cảm với độ nhiễu.

Hạn chế:

1. Có sai số giao nhau.
2. Tính toán phức tạp và tốn thời gian.

2.8. Hạn chế của việc áp dụng edge detection

- Kích thước của đầu ra sẽ bị thu hẹp.
- Mất nhiều thông tin có giá trị, đặc biệt là từ các cạnh của hình ảnh đầu vào.

III. PENCIL PHOTO SKETCHING

6. 3.1 Khái niệm

Phác thảo còn được biết đến với cái tên gọi khác là "sketch". Đây là một trong những nghệ thuật đồ hoạ bằng tay đơn giản được thực hiện như một bước đệm cơ bản nhất trước khi người hoạ sĩ vẽ nên tác phẩm của chính họ. Phác thảo thể hiện những nét vẽ nhanh, có thể chưa được trau chuốt cẩn thận.

Để có được một bản phác thảo bằng bút chì (nghĩa là bản vẽ đen trắng) của khung máy ảnh, chúng tôi sẽ sử dụng hai kỹ thuật hòa trộn hình ảnh, được gọi là **Dodge** và **Burn**. Các thuật ngữ này đề cập đến các kỹ thuật được sử dụng trong quá trình in trong nhiếp ảnh truyền thống; các nhiếp ảnh gia sẽ điều chỉnh thời gian phơi sáng của một vùng nhất định của bản in trong phòng tối để làm sáng hoặc tối nó. Dodging làm sáng một hình ảnh, trong khi đốt cháy làm tối nó.

Các khu vực không được phép thay đổi đã được bảo vệ bằng mask. Ngày nay, hiện tại các chương trình chỉnh sửa hình ảnh, chẳng hạn như Photoshop và Gimp, cung cấp các cách để bắt chước các hiệu ứng này trong hình ảnh kỹ thuật số. Ví dụ: mask vẫn được sử dụng để bắt chước hiệu ứng của việc thay đổi thời gian phơi sáng của hình ảnh, trong đó các vùng của mặt nạ có giá trị cường độ tương đối sẽ hiển thị hình ảnh nhiều hơn, do đó làm sáng hình ảnh. OpenCV không cung cấp một hàm gốc để thực hiện các kỹ thuật này, nhưng với một chút hiểu biết sâu sắc và một vài thủ thuật, chúng ta sẽ có được cách triển khai hiệu quả của riêng mình có thể được sử dụng để tạo ra hiệu ứng phác thảo bút chì đẹp mắt.

7. 3.2 Mục đích và ứng dụng

1. Giúp mọi người hiểu rõ hơn về cách các họa sĩ tạo ra các bức vẽ bằng bút chì, đây là là một trong ngôn ngữ hình ảnh cơ bản nhất để nhận thức trừu tượng của con người về cảnh thiên nhiên
2. Hiển thị cảnh thực theo cách phi thực tế, nghệ thuật trong nhiều trường hợp là điều mong muốn trong ngành công nghiệp game và điện ảnh vì cách làm cho hình ảnh trong ít chân thực hơn, Vì thế, máy tính giúp tiết kiệm rất nhiều sức lao động, việc sản xuất một phân đoạn video phi thực tế sẽ yêu cầu các nghệ sĩ vẽ nhiều bức tranh bằng tay.
    - Có thể biến những bức ảnh thông thường do chính họ chụp thành tác phẩm nghệ thuật độc đáo sẽ mang lại nhiều niềm vui cho cuộc sống của mọi người

8. 3.3 Các bước chuyển đổi hình ảnh sang phác thảo bút chì

Bước 1: Chuyển sang hình ảnh màu xám

Bây giờ, bước tiếp theo là chuyển đổi hình ảnh **có màu** hoặc **RGB** thành hình ảnh **đen trắng** bằng phương thức **cvtColor ()**.

Tại sao chúng ta chuyển đổi hình ảnh màu thành hình ảnh đen trắng bởi vì hình ảnh đen trắng chỉ có một lớp nhưng hình ảnh màu có ba lớp đó là R **(Đỏ)**, G **(Xanh lá cây)** và B **(Xanh lam)**. Vì vậy, để tiết kiệm thời gian, chúng tôi sẽ sử dụng hình ảnh đen trắng.

Mask= [0.2989, 0.5870, 0.1140]

https://www.w3.org/Graphics/Color/sRGB

grey_img=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

Bước 2: Đảo ngược hình ảnh

Chúng tôi có thể đảo ngược hình ảnh đơn giản bằng cách trừ đi 255, vì hình ảnh thang độ xám là hình ảnh 8 bit hoặc có tối đa 256 tông màu.

invert_img=cv2.bitwise_not(grey_img)

#invert_img=255-grey_img

Bước 3: Làm mờ hình ảnh

Bây giờ chúng ta làm mờ hình ảnh bị đảo ngược. Làm mờ được thực hiện bằng cách áp dụng bộ lọc Gaussian cho hình ảnh bị đảo ngược. Chìa khóa ở đây là phương sai của hàm Gaussian hoặc sigma.

Áp dụng tính năng làm mờ Gaussian cho hình ảnh. Đối số thứ hai của hàm là kích thước hạt nhân, nếu phải là một cặp số lẻ.

Kích thước hạt nhân lớn hơn, hình ảnh sẽ bị mờ hơn và nó sẽ mất đi những nét tinh tế.

Để tạo bản phác thảo, chúng tôi chỉ yêu cầu các đặc điểm nổi bật (các cạnh tương phản) từ hình ảnh.

Đối với hình ảnh nhỏ, kích thước hạt nhân là (3,3), (5,5), v.v. sẽ là đủ, trong khi đối với hình ảnh lớn hơn, kích thước hạt nhân nhỏ không tạo ra bất kỳ tác động nào.

Kích thước hạt nhân thích hợp có thể được chọn bằng phương pháp thử và sai.

blur_img=cv2.GaussianBlur(invert_img, (111,111),0)

Bước 4: Dodge pha trộn hình ảnh bị mờ và thang độ xám

Các **Color Dodge** pha trộn chế độ chia lớp dưới cùng của lớp trên cùng đảo ngược. Điều này làm sáng lớp dưới cùng tùy thuộc vào giá trị của lớp trên cùng. Chúng tôi có hình ảnh bị mờ, làm nổi bật các cạnh đậm nhất.

Vì tất cả các hình ảnh của chúng tôi được đọc bằng Numpy, tất cả các phép tính ma trận đều siêu nhanh.

sketch_img=cv2.divide(grey_img,255-blur_img, scale=256.0)

IV. KẾT LUẬN

Qua quá trình làm đồ án môn học này đã giúp nhóm chúng tôi có có thể hiểu thêm về các phát hiện cạnh và phác thảo ảnh, đặc biệt là về bài toán phát

hiện cạnh theo nhiều phương pháp khác nhau. Hiểu được khái niệm Gradient, phương pháp sobel, phương pháp laplace, phương pháp canny, kỹ thuật này là bước tiền xử lý quan trọng trong hầu hết các hệ thống xử lý ảnh, kết phân vùng tốt sẽ giúp cho quá trình xử lý về sau đạt hiệu quả cao hơn nhằm tiết kiệm về chi phí tính toán, thời gian cũng như tăng độ chính xác của các tác vụ.

- Kết quả đạt được

  -Nắm vững kiến thức nền tảng

  -Kỹ năng code được nâng cao

  -Xây dựng được ứng dụng của photo sketch

  -Kỹ năng làm việc nhóm, quản lí thời gian

STT	Họ và Tên	MSSV	Công việc	Hoàn thành

1	Võ Nhật Thanh	19522245	Tìm hiểu tài liệu. Tập trung lý thuyết Gradient, phương pháp laplace và phương pháp photo sketch. Tìm hiểu code, chạy demo trên colab. Làm slide, thuyết trình và viết báo cáo phần tập trung	9.5/10
2	Lê Vinh Quang	19522093	Tìm hiểu tài liệu. Tập trung vào lý thuyết Edge,kỹ thuật Robert, Sobel. Tìm hiểu code, chạy demo trên colab. Đóng góp ý kiến làm silde. Thuyết trình và viết báo cáo phần tập trung	9.5/10
3	Trần Trung Tín	19522351	Tìm hiểu tài liệu. Tập trung vào lý thuyết Edge detection, kỹ thuật canny và so sánh hiệu quả 2 thuật toán. Tìm hiểu code, chạy demo trên colab. Đóng góp ý kiến làm slide. Thuyết trình và viết báo cáo phần tập trung	9.5/10

BẢNG PHÂN CHIA CÔNG VIỆC

TÀI LIỆU THAM KHẢO

Edge detection:

https://viblo.asia/p/tuan-5-gradient-and-edge-bJzKmOGwl9N#_what-is-an-edge--0

https://www.cse.usf.edu/~r1k/MachineVisionBook/MachineVision.files/MachineVision_Chapter5.pdf

https://www.slideshare.net/EndiMion3/edge-detection-62287272

https://www.geeksforgeeks.org/implement-canny-edge-detector-in-python-using-opencv/?ref=gcse

https://slideplayer.vn/slide/17091492/

https://www.phamduytung.com/blog/2019-12-13-nms/

Photo sketching:

https://www.ri.cmu.edu/wp-content/uploads/2019/01/Li-Mengtian-WACV-2019-Photo-Sketching.pdf?fbclid=IwAR23CojRRVeytH7UIhuGzu0jSX5_V9TChnDcPrBviEOhNd8Bmcav2Dsz8cs

https://towardsdatascience.com/generate-pencil-sketch-from-photo-in-python-7c56802d8acb

https://www.geeksforgeeks.org/convert-image-into-sketch/

https://en.wikipedia.org/wiki/Blend_modes#Dodge_and_burn

https://subscription.packtpub.com/book/data/9781789537147/1/ch01lvl1sec06/creating-pencil-sketches-from-images?fbclid=IwAR2Eq0JODKVAmPLcCWQG4qZ3Y8zrewUYq8PTljQxoLatt15OifGrFlAHMdQ

https://arxiv.org/pdf/2001.02600.pdf?fbclid=IwAR1cclpy2vwd1XCAbJ0vHNT6jAyMQRI93VNvLw_IQUBBSEKjPKXRFEQIUPw