LỜI CẢM ƠN

Trước hết em xin được bày tỏ lòng biết ơn sâu sắc đối với thầy giáo hướng dẫn PGS.TS. Ngô Quốc Tạo, Viện Công Nghệ Thông Tin -Viện Khoa Học & Công Nghệ Việt Nam đã tận tình giúp đỡ, chỉ bảo em trong thời gian vừa qua và đã dành rất nhiều thời gian quý báu để giúp em hoàn thành đề tài được giao. Em xin chân thành cảm ơn thầy PGS, TS. Đỗ Năng Toàn, Viện Công Nghệ Thông Tin – Viện Khoa Học & Công Nghệ Việt Nam, người đã cho em niềm đam mê về lĩnh vực Xử lý ảnh.

Em xin gửi lời cảm ơn đến các Thầy cô giáo trong Khoa Công nghệ thông tin, trường Đại Học Dân Lập Hải Phòng đã giảng dạy chúng em trong suốt quãng thời gian qua, cung cấp cho chúng em những kiến thức chuyên môn cần thiết và quý báu giúp chúng em hiểu rõ hơn các lĩnh vực đã nghiên cứu để hoàn thành đề tài được giao .

Cuối cùng, em xin cảm ơn các bạn bè và gia đình đã động viên cổ vũ, đóng góp ý kiến cho em trong suốt quá trình học cũng như làm tốt nghiệp, giúp em hoàn thành đề tài đồ án đúng thời hạn.

Hải Phòng, tháng 7 năm 2010

Sinh viên

Nguyễn Thị Lan

MỤC LỤC

PHẦN MỞ ĐẦU

Xử lý ảnh là một nghành khoa học còn tương đối mới mẻ so với nhiều nghành khoa học khác. Tuy nhiên, hiện nay nghành khoa học này đang tiến những bước dài và đang dần khẳng định là một trong những nghành khoa học không thể thiếu được trong các lĩnh vực ứng dụng công nghệ thông tin.

Trong Xử lý ảnh việc nhận dạng và phân lớp các đối tượng đòi hỏi rất nhiều quá trình xử lý khác nhau, trong đó một công cụ không thể thiếu được đó là việc phát hiện biên. Do đó biên đóng một vị trí hết sức cơ bản trong phân tích ảnh, biên tạo nên khuôn dạng của đối tượng. Biên là ranh giới giữa một đối tượng và nền hay là đường ranh giới phân biệt giữa hai đối tượng kề nhau. Điều này có nghĩa là nếu như các biên của đối tượng được xác định chính xác thì các đối tượng cũng được định vị và các thuộc tính cơ bản của đối tượng như diện tích, chu vi và hình dạng cũng có thể tính được.

Có nhiều phương pháp phát hiện biên khác nhau. Chúng đều dựa trên cơ sở là sự thay đổi đột ngột về độ sáng của điểm ảnh.

Hiện nay, các phương pháp phát hiện biên nâng cao được xây dựng trên cơ sở phân tích lý thuyết chặt chẽ về mô hình toán học của biên và nhiễu. Cách phát hiện biên không còn đơn giản như trước nữa, chúng sử dụng một loạt các kỹ thuật phức tạp như kỹ thuật loại trừ các điểm không cực đại (nonmaximum suppress), kỹ thuật phân ngưỡng trễ (hyteresis thresholding), kỹ thuật phân ngưỡng cục bộ…Kết quả là việc tìm biên hiệu quả và chính xác hơn.

Để có thể trình bày các vấn đề này một cách rõ ràng trong đồ án nay, em xin trình bày 5 chương như sau:

Chương I: Một số khái niệm cơ bản trong Xử lý ảnh. Chương này trình bày tổng quát về Xử lý ảnh và các khái niệm sẽ dùng trong đồ án này.

Chương II: Các phương pháp phát hiện biên cổ điển. Dùng các toán tử đạo hàm để tìm biên. Tiếp theo là kỹ thuật dò biên tổng quát.

Chương III: Phương pháp phát hiện biên dựa vào phép toán hình thái. Hai phép toán hình thái cơ bản là: Dilation và Erosion.

Chương IV: Một số phương pháp phát hiện biên nâng cao. Chương này đề cập đến 3 phương pháp tìm biên nâng cao đó là phương pháp Canny, Shen-Castan, Marr-Hildreth. Tiếp theo là ứng dụng của biên.

Chương V: Cài đặt và đánh giá một số thuật toán trong phương pháp phát hiện biên bằng ngôn ngữ Virtual C++.

Kết luận:

Phụ lục:

Khi bắt tay vào việc nghiên cứu đề tài này, em đã cố gắng hết sức để hoàn thành công việc được giao, song điều kiện về thời gian và trình độ còn hạn chế nên em không thể không tránh khỏi được những thiếu sót. Em rất mong được sự góp ý của thầy giáo hướng dẫn, thầy giáo phản biện cũng như các thầy cô giáo và bạn bè trong Khoa Công Nghệ Thông Tin, qua đó em đã rút ra được những kinh nghiệm thực tế và bổ ích để sau này em có thể xây dựng được một chương trình hoàn chỉnh hơn.

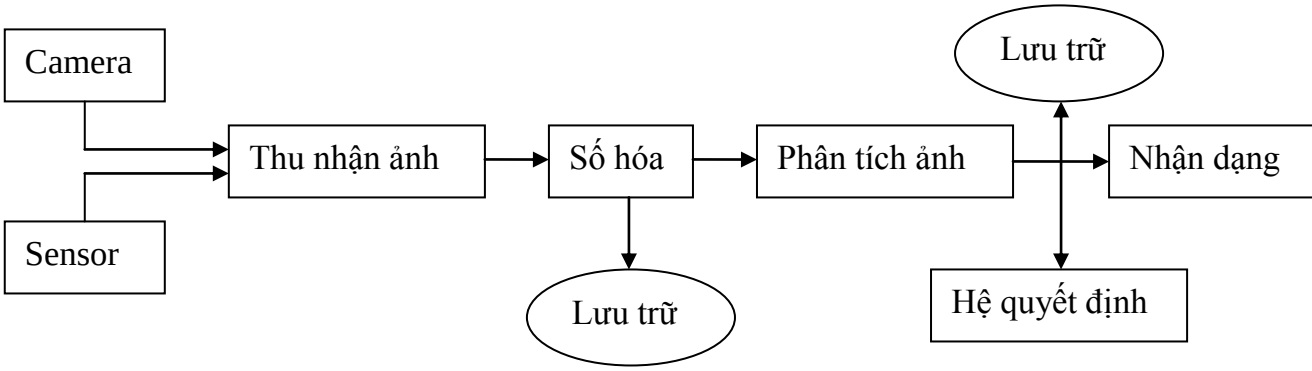
CHƯƠNG I: MỘT SỐ KHÁI NIỆM CƠ BẢN TRONG XỬ LÝ ẢNH

1.1. Tổng quan về xử lý ảnh

1.1.1. Xử lý ảnh

Xử lý ảnh là một lĩnh vực khoa học gồm tất cả những gì liên quan đến việc thao tác ảnh nhằm đưa ra được ảnh như mong muốn.

Xử lý ảnh liên quan đến các hình ảnh đã có, trong khi đó đồ họa máy tính liên quan đến việc tổng hợp hình ảnh thực hoặc ảo trên máy tính. Ngoài ra trong đồ họa đối tượng là hai hoặc ba chiều, còn trong xử lý ảnh có thể là nhiều hơn.



Hình 1: Sơ đồ tổng quát của một hệ thống nhận dạng trong xử lý ảnh.

Trong sơ đồ trên thì ảnh cần được xử lý thông qua hệ thống thu nhận ảnh. Hệ thống thu nhận ảnh này bao gồm các thiết bị chụp như camera, máy quét scanner, máy chụp hình…

Ảnh sau khi thu nhận được qua hệ thống thu nhận, ảnh sẽ được lấy mẫu và số hóa, sau đó sẽ được phân tích theo các loại ảnh. Có rất nhiều loại ảnh chúng được lưu trữ dưới các file khác nhau như: file Bitmap, file PCX, file Gif…Tuy nhiên trong phần đồ án này em chỉ hiển thị ảnh dưới dạng file Bitmap. Ảnh sau khi phân tích sẽ được lưu trữ và tùy theo từng ứng dụng cụ thể mà chọn ra cách thích hợp để phân tích.

Vì vậy: Mục đích của xử lý ảnh là:

- Biến đổi ảnh và làm cho ảnh đẹp

• Tự động phân tích nhận dạng ảnh hay đoán nhận ảnh và đánh giá các nội dung của ảnh.

1.1.2. Ảnh và điểm ảnh

Trong quá trình số hóa người ta biến đổi tín hiệu liên tục thành tín hiệu rời rạc thông qua quá trình lấy mẫu và lượng tử hóa. Do vậy điểm ảnh có thể xem như sự biểu diễn về cường độ sáng hay một dấu hiệu nào đó của ảnh tại một tọa độ nào đó. Ảnh còn là tập hợp các điểm ảnh.

1.1.3. Mức xám (Gray level)

Mức xám là sự mã hóa tương ứng một cường độ sáng của mỗi điểm ảnh với một giá trị là số và là kết quả của quá trình lượng tử hóa. Cách mã hóa thường dùng là 16, 32, hay 64 mức. Mã hóa 256 mức là thông dụng nhất do kỹ thuật vì 2^8 =256(0,1…255) nên với 256 mức thì mọi pixel được mã hóa bởi 8 bit.

1.1.4. Pixel (Picture element)

Là phần tử ảnh, điểm ảnh. Ảnh trong thực tế là ảnh liên tục về không gian độ sáng. Để có thể xử lý ảnh bằng máy tính cần phải tiến hành số hóa, người ta biến đổi tín hiệu liên tục sang tín hiệu rời rạc thông qua quá trình lấy mẫu (rời rạc hóa về không gian) và lượng hóa thành phần giá trị mà về nguyên tắc bằng mắt thường không phân biệt được hai điểm kề nhau. Do vậy một điểm ảnh là tập hợp các pixel, mỗi pixel gồm một cặp tọa độ x, y và màu. Một pixel có thể lưu trữ trên 1, 4, 8 hay 24 bit.

1.1.5. Biểu diễn ảnh

Trong biểu diễn ảnh, người ta dùng các phần tử đặc trưng của ảnh là pixel. Có thể xem một hàm hai biến chứa các thông tin như biểu diễn của ảnh, việc xử lý ảnh số yêu cầu ảnh phải được mã hóa và lượng tử hóa. Việc lượng tử hóa ảnh là chuyển đổi tín hiệu tương tự sang tín hiệu số của một ảnh đã lấy mẫu sang một số hữu hạn mức xám.

Một số mô hình thường được dùng trong xử lý ảnh, mô hình toán, mô hình thống kê.

1.1.6. Tăng cường và khôi phục ảnh

Tăng cường ảnh là bước quan trọng tạo tiền đề cho xử lý ảnh, gồm một loạt các kỹ thuật như: lọc độ tương phản, khử nhiễu, nổi màu…

Khôi phục ảnh là nhằm loại bỏ các suy giảm trong ảnh.

1.1.7. Biến đổi ảnh

Trong thuật ngữ biến đổi ảnh thường được dùng để nói đến một lớp các ma trận đơn vị và các kỹ thuật dùng để biến đổi ảnh. Một số loại biến đổi được dùng như: biến đổi Fourier, Sin, Cosin, Hadamard, tích Kronecker, biến đổi Karhumen Loeve…

1.1.8. Phân tích ảnh

Liên quan đến việc xác định các độ đo định lượng của một ảnh để đưa ra một mô tả đầy đủ về ảnh. Các kỹ thuật được sử dụng ở đây nhằm mục đích xác định biên của ảnh.

1.1.9. Nhận dạng ảnh

Là quá trình liên quan đến việc mô tả các đối tượng mà người ta muốn đặc tả nó. Quá trình nhận dạng thường đi sau quá trình trích chọn các đặc tính chủ yếu của đối tượng.

Có hai kiểu mô tả đối tượng đó là: mô tả tham số (nhận dạng theo tham số) và mô tả theo cấu trúc (nhận dạng theo cấu trúc).

1.1.10. Nén ảnh

Dữ liệu ảnh cũng như các dữ liệu khác cần phải lưu trữ hay truyền đi trên mạng, lượng thông tin để biểu diễn cho một ảnh là rất lớn . Do đó làm giảm lượng thông tin hay nén dữ liệu là một nhu cầu cần thiết.

1.2. Các định dạng cơ bản trong xử lý ảnh

Trong quá trình xử lý ảnh, một ảnh thu nhận vào máy tính phải được mã hóa. Hình ảnh khi lưu trữ dưới dạng tệp tin sẽ được số hóa. Một số dạng ảnh đã được chuẩn hóa như: ảnh GIF, BMP, PCX, IMG, TIFF…

• **Ảnh IMG**: Là ảnh đen trắng, phần đầu của ảnh có 16 byte chứa các thông tin cần thiết, ảnh IMG được nén theo từng dòng. Mỗi dòng bao gồm các gói (pack). Các dòng giống nhau cũng nén thành một gói.

• **Ảnh PCX:** Định dạng ảnh PCX là một trong những định dạng ảnh cổ điển nhất, nó thường được dùng để lưu trữ ảnh, nó sử dụng phương pháp mã loại dài RLE (Run-Length-Encoded) để nén dữ liệu ảnh, quá trình nén và giải nén được thực hiện trên từng dòng ảnh.

• **Ảnh TIFF**: Là ảnh mà dữ liệu chứa trong tệp thường được tổ chức thành các nhóm dòng (cột) quét của dữ liệu ảnh.

• **Ảnh GIF** (Graphics Interchanger Format): Với định dạng ảnh GIF những vướng mắc mà các định dạng khác gặp phải khi số trong ảnh tăng lên không còn nữa. Dạng ảnh GIF cho chất lượng cao độ phân giải đồ họa cũng đạt cao, cho phép hiển thị trên hầu hết các phần cứng.

• **Ảnh BMP** (Windows Bitmap): Là một định dạng tập tin hình ảnh khá phổ biến, cấu trúc tập tin ảnh bao gồm 4 phần:

•• Bitmap Header (14 bytes): giúp nhận dạng tập tin bitmap.

•• Bitmap Information (40 bytes): lưu một số thông tin chi tiết giúp hiển thị ảnh.

•• Color Palette (4*x bytes), x là số màu của ảnh: định nghĩa các màu sẽ được sử dụng trong ảnh.

•• Bitmap Data**:** lưu dữ liệu ảnh.

1.3. Một số khái niệm cơ bản trong phát hiện biên

1.3.1. Khái niệm biên

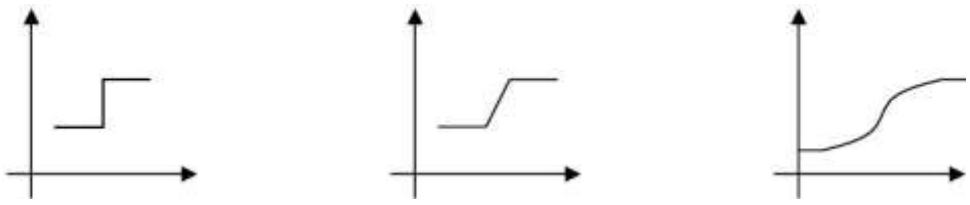
Biên là ranh giới giữa một đối tượng và nền hay là đường ranh giới phân biệt giữa hai đối tượng kề nhau. Do đó biên đóng một vị trí hết sức cơ bản trong phân tích ảnh, một điểm ảnh có thể là biên nếu ở đó có sự thay đổi đột ngột về mức xám. Tập hợp các điểm biên tạo thành biên hay đường bao của ảnh. Do đó một điểm có thể gọi là biên nếu đó là điểm đen và có ít nhất một điểm trắng lân cận.

1.3.2. Tại sao phải tìm biên

Tìm biên là làm nổi bật lên được những điểm ảnh mà tại đó có sự biến đổi lớn về giá trị độ sáng so với các điểm xung quanh, thực chất đây là công đọan quan trọng trong công việc phân tích ảnh, bởi vì do các nguyên nhân khác nhau làm cho ảnh bị suy biến, do vậy phải tăng cường và khôi phục lại ảnh.

Tìm biên còn chính là đi tìm được các đường bao quanh của đối tượng, quá trình định vị các điểm biên trong khi mà biên lại làm tăng độ tương phản giữa biên và nền, cho đến khi biên có thể được nhìn thấy một cách dễ dàng.

Hiện nay có nhiều định nghĩa về đường biên, mỗi định nghĩa được sử dụng trong một số trường hợp nhất định. Điển hình gồm ba loại đường biên chính: đường biên lý tưởng, đường biên bậc thang, đường biên thực(không trơn).



Hình 2: Các đường biên

a, Đường biên lý tưởng b, Đường biên bậc thang c, Đường biên thực

• **Đường biên lý tưởng**

Đường biên lý tưởng được định nghĩa là sự thay đổi giá trị cấp xám tại một vị trí xác định. Nếu sự thay đổi cấp xám giữa các vùng trong ảnh càng lớn thì đường biên càng dễ dàng nhận ra. Trong trường hợp này sự thay đổi này lại diễn ra tại một điểm nên đường biên có độ rộng là một điểm ảnh và vị trí của đường biên chính là vị trí thay đổi cấp xám.

• **Đường biên bậc thang**

Đường biên bậc thang xuất hiện khi sự thay đổi cấp xám trải rộng qua nhiều điểm ảnh. Vị trí của đường biên được xem như vị trí chính giữa của đường nối giữa cấp xám thấp và cấp xám cao. Tuy nhiên đây chỉ là đường thẳng trong toán học, từ khi ảnh được kỹ thuật số hoá thì đường đó không còn là đường thẳng mà thành những đường không trơn.

• **Đường biên thực** (không trơn)

Đường biên thực xuất hiện khi sự thay đổi cấp xám tại nhiều điểm ảnh nhưng không trơn.

1.3.3. Các khái niệm về nhiễu

Trong thực tế không bao giờ xảy ra trường hợp không có nhiễu, mà nhiễu xuất hiện hoàn toàn ngẫu nhiên nên không thể dự đoán nhiễu một cách chính xác. Tuy nhiên dựa trên những ảnh hưởng của nhiễu gây ra trên ảnh ta có thể mô tả nhiễu theo độ lệch tiêu chuẩn và giá trị trung bình của nó. Trong phân tích ảnh chúng ta quan tâm đến hai kiểu nhiễu chính đó là:

a, Nhiễu độc lập tín hiệu (Nhiễu cộng)

Là một tập ngẫu nhiên các mức xám độc lập với dữ liệu ảnh. Tức là được cộng thêm vào các điểm ảnh để được ảnh bị nhiễu. Loại nhiễu này thường xuất hiện khi ảnh được truyền bằng điện tử từ nơi này đến nơi kia. Nếu ảnh A là ảnh không có nhiễu, N là nhiễu xuất hiện trong quá trình truyền ảnh thì kết quả cho một ảnh $A^{'}$ có nhiễu như sau: $A^{'} = A+N$.

Với A và N độc lập nhau. Giả thiết nhiễu N có thể có thuộc tính thống kê bất kỳ, nhưng giả sử chung là tuân theo luật phân bố chuẩn với trung bình bằng 0 và độ lệch tiêu chuẩn cho trước.

Việc tạo các bức ảnh bị nhiễu nhân tạo với các đặc trưng cho trước không phải là quá khó khăn. Những bức ảnh như thế là công cụ rất cần thiết cho việc kiểm chứng các giải thuật phát hiện biên.

b, Nhiễu phụ thuộc tín hiệu (signal-dependent noise) (Nhiễu nhân)

Trong trường hợp này các mức giá trị nhiễu tại mỗi điểm ảnh là một hàm mức xám tại điểm đó. Loại nhiễu này thường khó ước lượng nhưng thường ít quan trọng hơn và ta có thể kiểm soát được nếu ta tiến hành lấy mẫu ảnh một cách thích hợp.

Trong trường hợp ảnh được nhập một cách chính xác thì sẽ kiểm soát được nhiễu. Nếu ảnh A là ảnh không có nhiễu, N là nhiễu xuất hiện trong quá trình truyền ảnh thì kết quả cho một ảnh $A^{'}$ có nhiễu như sau: $A^{'} = A * N$

1.3.4. Quy trình phát hiện biên

B1: Do ảnh ghi được thường có nhiễu, vì vậy cần lọc bỏ nhiễu theo các phương pháp đã tìm hiểu ở trên.

B2: Làm nổi biên sử dụng các toán tử phát hiện biên.

B3: Định vị biên (chú ý rằng kỹ thuật làm nổi biên gây tác dụng phụ là gây nhiễu làm một số biên giả xuất hiện do vậy cần loại bỏ biên giả).

B4: Liên kết và trích chọn biên.

1.4. Các phương pháp đánh giá thuật toán phát hiện biên

Sau khi đưa ra một phương pháp tìm biên, sẽ rất tốt nếu như đánh giá được độ thành công của từng thuật toán. Nói chung không có cách nào để đánh giá được điều này. Tuy nhiên những ước lượng về phương pháp tìm biên có thể thu được bằng việc xét những lỗi mà một thuật toán tìm biên có thể mắc phải. Một thuật toán tìm biên có thể mắc phải các lỗi sau:

• Lỗi âm: Một thuật toán tìm biên có thể không thông báo một biên trong khi nó tồn tại

• Lỗi dương: Một thuật toán tìm biên có thể thông báo về một biên trong khi nó không tồn tại. Điều này có thể do nhiễu hoặc do việc thiết kế thuật toán sơ sài hoặc do quá trình phân ngưỡng.

• Vị trí của một điểm biên có thể bị nhầm. Trong các ảnh thử nghiệm ta biết được số lượng và vị trí của các điểm biên, biết được số lượng và kiểu nhiễu nên việc áp dụng các phương pháp phát hiện biên và các ảnh này sẽ cho ta một sự đánh giá gần đúng và hiệu quả của các phương pháp phát hiện biên.

Sau đây sẽ giới thiệu hai phương pháp đánh giá đó là: phương pháp Pratt và phương pháp Kitchen-Rosenfeld

1.4.1. Đánh giá Pratt

Dựa vào những phân tích trên, năm 1978 Pratt đã đề xuất ra hàm :

$$E1= \frac{\sum_{i=1}^{I_A}\left(\frac{1}{1+\alpha\, d(i)^2}\right)}{\max(I_A, I_i)}$$

Trong đó: $\mathbf{I_A}$: số lượng điểm biên tìm được bằng phương pháp tìm biên

$\mathbf{I_i}$: số lượng các điểm biên thực sự trên ảnh thử nghiệm

d(i): là khoảng cách giữa điểm thứ i trên ảnh thử nghiệm và các điểm tìm biên bởi phương pháp

$\alpha = 1/9$ là hằng số do Pratt đưa ra

Giá trị E là một hàm của khoảng cách giữa vị trí biên được đo và vị trí trên thực tế, tuy nhiên nó chỉ liên quan gián tiếp với lỗi âm và lỗi dương.

1.4.2. Đánh giá Kitchen-Rosenfeld

Năm 1981 Kitchen và Rosenfeld giới thiệu một mô hình đánh giá dựa trên tính liên kết cục bộ của biên. Phương pháp này đo độ lệch giữa một điểm biên với các điểm biên láng giềng, chứ không quan tâm vào vị trí thực tế của biên, do vậy nó

là sự bổ sung cho phương pháp Pratt. Bước đầu tiên là xác định một hàm tính toán xem, liệu có xuất hiện một điểm biên lân cận bên trái hay không.

Nếu láng giềng K là một điểm biên.

$$L_K = \begin{cases} a(d, d_K) \le a\left(\frac{k\pi}{4}, d + \frac{\pi}{2}\right) \\ 0 \end{cases}$$

Trường hợp khác.

Trong đó: **d** : hướng của biên tại điểm đang xét

d_0: hướng của biên tại điểm láng giềng phải

d_1: hướng của láng giềng phải trên tương tự ngược chiều kim đồng hồ với những điểm liên quan

a: là hàm đo độ lệch góc giữa hai góc bất kì

$$a(\alpha, \beta) = \frac{\pi - |\alpha - \beta|}{\pi}$$

Một hàm tương tự để đo hướng liên tục về phía bên phải của điểm biên đang xét:

Nếu láng giềng K là một điểm biên.

$$R_K = \begin{cases} a(d, d_K) \le a\left(\frac{k\pi}{4}, d + \frac{\pi}{2}\right) \\ 0 \end{cases}$$

Trường hợp khác.

Độ đo sự tiếp tục được tính bằng trung bình các giá trị lớn nhất của L(K) và R(K) và được gọi là C. Biên phải là một đường mảnh có độ rộng bằng một pixel. Các đường biên có độ rộng lớn hơn một pixel là những biên có sự xuất hiện của lỗi âm, có thể là do giải thuật đã phát hiện ra nhiều hơn một lần đối với cùng một biên thực. Độ mảnh T là 6 điểm ảnh của một vùng ảnh kích thước 3x3, tâm là điểm ảnh đang xét và bỏ qua tâm vùng (điểm đang xét) và hai điểm ảnh được tìm thấy bởi L(K) và R(K) là các điểm biên. Ước lượng tổng thể của phương pháp phát hiện biên là:

$$E_2 = \gamma C + (1 - \gamma) T$$

Trong đó : γ là một hằng số và ở đây sử dụng: $\gamma = 0{,}8$

CHƯƠNG II: CÁC PHƯƠNG PHÁP PHÁT HIỆN BIÊN CỔ ĐIỂN

2.1. Cơ sở về các phép toán tìm biên

2.1.1. Khái niệm

Tìm biên là đi tìm các đường bao quanh các đối tượng trong ảnh. Trong thực tế ảnh thường đi kèm theo nhiễu, vì vậy tìm biên là công việc rất khó và hầu như trước khi sử dụng các thuật toán tìm biên, phải trải qua một bước tiền xử lý đó là quá trình loại bỏ nhiễu. Cơ sở của phương pháp tìm biên, đó là quá trình biến đổi độ lớn về giá trị độ sáng của các điểm ảnh khi đi qua biên. Điều đó có nghĩa là điểm biên là điểm tại đó có sự biến đổi về lớn về giá trị độ sáng. Trong khi đó, sự biến đổi về giá trị độ sáng của các điểm thuộc một đối tượng là nhỏ. Như vậy, nếu chúng ta làm nổi lên được những điểm ảnh mà lại có sự biến đổi lớn về giá trị độ sáng của các vùng là đều, thì có nghĩa là làm nổi được biên của ảnh. Đây chính là cơ sở của các thuật toán tìm biên xuất phát từ những cơ sở này, có hai phương pháp phát hiện biên tổng quát đó là: phương pháp tìm biên trực tiếp và phương pháp tìm biên gián tiếp.

• Phương pháp tìm biên trực tiếp: là phương pháp làm nổi biên dựa vào sự biến thiên về giá trị độ sáng của điểm ảnh. Kỹ thuật này chủ yếu dùng phát hiện biên ở đây là kỹ thuật đạo hàm, nếu lấy đạo hàm bậc nhất của ảnh ta có phương pháp Gradient, Sobel, Prewitt, nếu lấy đạo hàm bậc hai ta có kỹ thuật Laplace. Các phương pháp này sẽ được mô tả ở dưới.

• Phương pháp tìm biên gián tiếp: Nếu bằng cách nào đó ta phân được ảnh thành các vùng, trong vùng đó những điểm ảnh có tính chất tương tự nhau, khi đó ranh giới giữa các vùng đó được gọi là biên.

Như vậy: Phương pháp phát hiện biên trực tiếp và gián tiếp là hai bài toán đối ngược nhau, khi biết các vùng sẽ tìm được biên và ngược lại. Tuy nhiên trong một số trường hợp không thể làm ngược lại (Biên hở).

Tuy nhiên, phương pháp tìm biên trực tiếp thường sử dụng có hiệu quả đối với các ảnh ít chịu ảnh hưởng của nhiễu, song nếu như sự biến thiên độ sáng không đột ngột thì phương pháp này tỏ ra kém hiệu quả. Phương pháp tìm biên gián tiếp giải quyết tốt trong trường hợp này, tuy nhiên cài đặt tương đối khó khăn. Với các phương pháp tìm biên trực tiếp, có hai dạng sau:

• Phương pháp tìm biên dùng bộ lọc tuyến tính: Phương pháp này dựa trên phép toán xử lý lân cận cục bộ hoặc xử lý tổng thể. Xử lý lân cận là sử dụng các ma trận hệ số lọc kích thước nhỏ, còn xử lý tổng thể là thực hiện trên toàn ảnh và có thể coi như sử dụng ma trận hệ số lọc có kích thước bằng kích thước của ảnh cụ thể như các phương pháp Gradient, Sobel, Prewitt, Laplace…

• Phương pháp tìm biên phi tuyến: Phương pháp này không dựa trên phép lọc tuyến tính mà sử dụng các phép toán phi tuyến như phép toán lựa chọn, so sánh được áp dụng trong phương pháp hình chóp, Kisrch…

Sau khi đưa ra các khái niệm cơ bản phục vụ cho việc xây dựng một phương pháp tách cạnh, một câu hỏi được đặt ra làm thế nào để có thể tìm ra được cạnh. Một cạnh như đã nói ở trên là sự thay đổi cấp xám hay thay đổi chu tuyến màu. Nếu ta mô tả sự thay đổi trên sơ đồ nêu trên thì sẽ thấy các mức xám có đường cong biến thiên rất ngẫu nhiên và phức tạp. Điều này dẫn ta tới kết luận rằng: Mỗi cạnh đều có một giá trị Gradient nhất định. Khi Gradient tại điểm trên đường cong biến thiên là lớn hay nói cách khác sự thay đổi cấp xám là lớn thì khi đó một cạnh được nhận dạng để xác định ra cạnh kia thì khi đó ta có thể dùng toán tử đạo hàm (còn gọi là phương pháp phát hiện biên trực tiếp) được thiết kế để nhận dạng những nơi có sự thay đổi cường độ lớn, hoặc dùng phương pháp gần giống với phương pháp đối sánh mẫu (template-matching) gọi là phương pháp dựa mẫu, nơi mà cạnh được mô hình hóa bởi một mẫu nhỏ.

2.1.2. Toán tử đạo hàm

Nếu như cạnh được định nghĩa bằng sự thay đổi cấp xám, thì một toán tử nhạy cảm với sự thay đổi này sẽ là một toán tử tách cạnh, đó chính là toán tử đạo hàm. Toán tử đạo hàm là tốc độ thay đổi của một hàm số, tốc độ thay đổi cấp xám trong một ảnh là lớn nếu gần cạnh và nhỏ trong các khu vực có cấp xám là hằng số.

Trong trường hợp ảnh được biểu diễn hai chiều, ta phải xét sự thay đổi cấp xám theo nhiều hướng. Cũng vì lý do này mà đạo hàm riêng theo hai hướng x, y được sử dụng. Việc đánh giá hướng của cạnh có thể thu được bằng việc dùng đạo hàm x, y như những thành phần hướng cạnh và tính vector tổng. Toán tử liên quan đến đạo hàm là Gradient và nếu ảnh được coi như là một hàm hai biến A(x, y) thì Gradient được định nghĩa là:

$\nabla A(x, y)= \left(\frac{\partial A}{\partial x}, \frac{\partial A}{\partial y}\right)$ đây là một vector hai chiều.

Tất nhiên ảnh không phải là một hàm, và cũng không thể được vi phân hóa bằng cách thông thường. Thực chất chúng ta sử dụng sai phân để tính xấp xỉ đạo hàm vì ảnh số là tín hiệu rời rạc nên đạo hàm không tồn tại. Một sự xấp xỉ đơn giản nhất là toán tử ∇_1:

$$\nabla_{x1}A(x, y) = A(x, y) - A(x-1, y)$$

$$\nabla_{y1}A(x, y) = A(x, y) - A(x, y-1)$$

Giả thiết trong trường hợp này các cấp xám biến đổi giữa các điểm ảnh là tuyến tính, do đó mà giá trị đạo hàm là độ nghiêng của đường thẳng. Các phương pháp sử dụng đạo hàm bậc nhất làm việc khá tốt khi mà độ sáng thay đổi rõ nét. Khi mức xám thay đổi chậm, miền chuyển tiếp trải rộng, phương pháp này hiệu quả hơn là phương pháp sử dụng đạo hàm bậc hai. Khi đó toán tử ∇_2 xấp xỉ với các đạo hàm bậc hai:

$$\nabla^2_{x2}A = A(x +1, y) - A(x-1, y)$$

$$\nabla^2_{y2}A = A(x, y +1) - A(x, y-1)$$

Toán tử này đối xứng với điểm ảnh (x, y), mặc dù nó không xét đến giá trị của điểm ảnh tại (x, y). Bất cứ toán tử nào được dùng để tính Gradient thì vector kết quả cũng chứa đựng thông tin về độ lớn và hướng của Gradient tại điểm đó. Độ lớn của vector Gradient là độ dài của cạnh huyền tam giác phía phải có hai cạnh còn lại là ∇_x, ∇_y điều này phản ánh độ lớn của cạnh tại bất cứ điểm ảnh xác định nào. Hướng của cạnh tại mỗi điểm ảnh là góc tạo bởi cạnh huyền với các trục x, y. Khi độ lớn gradient được tính bằng công thức sau:

$$G_{max}= \sqrt{\left(\left(\frac{\partial A}{\partial x}\right)^2+\left(\frac{\partial A}{\partial y}\right)^2\right)}$$

Và hướng của cạnh xấp xỉ bằng: G_{dir} = arctg $\left[\frac{\frac{\partial A}{\partial y}}{\frac{\partial A}{\partial x}}\right]$

Độ lớn của cạnh sẽ là một số thực và thường được làm tròn thành số nguyên. Điểm ảnh nào mà có gradient vượt quá giá trị ngưỡng cho trước thì nói rằng đó là điểm cạnh, còn những điểm khác không phải. Hai toán tử tách cạnh được đánh giá ở đây sẽ sử dụng giá trị trung vị trong khoảng các cấp xám như một ngưỡng. Để tìm hiểu kỹ về các vấn đề trên em xin lần lượt trình bày các phương pháp tìm biên ở trong phần sau.

2.2. Phương pháp tìm biên dựa trên kĩ thuật lọc tuyến tính

Kỹ thuật lọc tuyến tính thực chất là quá trình xếp chồng ảnh đầu vào với một hạt nhân xếp chồng tương ứng. Các hạt nhân xếp chồng này được xây dựng trên cơ sở hai phép toán cơ bản, đó là phương pháp Gradient và phương pháp Laplace. Về lý thuyết mỗi phương pháp xác định biên này thường trải qua các giai đoạn chính sau:

• Loại bỏ nhiễu

• Chọn toán tử thực hiện

• Chọn phương pháp xác định điểm biên

• Liên kết các điểm biên

Sau đây các phép toán tìm biên sẽ được trình bày

2.2.1. Phương pháp đạo hàm bậc nhất Gradient

Phương pháp Gradident là phương pháp dò biên cục bộ dựa vào cực đại của đạo hàm. Theo định nghĩa Gradient là một vector có các thành phần biểu thị tốc độ thay đổi giá trị của điểm ảnh theo hai hướng x và y. Các thành phần của Gradient được tính bởi công thức sau:

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

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

Với dx, dy là các khoảng cách giữa các điểm theo hướng x và hướng y (tính theo số điểm). Trong thực tế người ta hay dùng với dx= dy =1.

Với một ảnh liên tục f(x,y), các đạo hàm riêng của nó cho phép xác định vị trí cực đại cục bộ theo hướng của biên. Khi đó Gradient của một ảnh liên tục được biểu diễn bởi một hàm f(x,y) dọc theo r với góc θ được định nghĩa theo tọa độ cực.

Ta có: $f(x, y) = f(r.\cos\theta, r.\sin\theta)$

$$\frac{df}{dr} = \frac{\partial f}{\partial x}.\frac{dx}{dr} + \frac{\partial f}{\partial x}.\frac{dy}{dr} = -f_x . \cos\theta + f_y . \sin\theta$$

$$\frac{df}{d\theta} = \frac{\partial f}{\partial x}.\frac{dx}{d\theta} + \frac{\partial f}{\partial y}.\frac{dy}{d\theta} = -f_x . r.\sin\theta + f_y .r.\cos\theta$$

Hướng xảy ra khi: $\frac{df}{d\theta} = 0$

$$\Leftrightarrow -f_x . r\sin\theta + f_y . r\cos\theta = 0$$

$$\Leftrightarrow \operatorname{tg}\theta = \frac{fx}{fy} \qquad \Rightarrow \theta = \operatorname{arctg}\frac{fx}{fy}$$

Và $\frac{df}{d\theta}\max = \sqrt{fx^2 + fy^2}$.

Khi đó θ_r là hướng của biên.

Tuy ta nói lấy đạo hàm của ảnh, nhưng thực ra chỉ là mô phỏng và xấp xỉ đạo hàm bằng kĩ thuật nhân chập (cuộn). Do ảnh số là tín hiệu rời rạc, do vậy đạo hàm không tồn tại.

Các bước tiến hành như sau:

1. Dùng 2 mặt nạ là: $H_1 = \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix}$ và $H_2 = \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}$

2. Với mỗi điểm ảnh I(x, y) ta tính Gradient I(x, y):

$$\text{Gradient I(x, y)} = \sqrt{I(x,y) \otimes H1^2 + I(x,y) \otimes H2^2}$$

3. Tìm các điểm biên I'(x, y) = Gradient I(x, y)

Nếu Gradient I(x, y) > θ thì I'(x, y) = 1

Gradient I(x, y) <= θ thì I'(x, y) = 0

Việc xấp xỉ đạo hàm bậc nhất theo các hướng x, y được thực hiện thông qua 2 mặt nạ nhân chập tương ứng sẽ cho ta các kỹ thuật phát hiện biên khác nhau. Phương pháp này có thể tạo ra một số mặt nạ khác bằng cách sử dụng kỹ thuật lấy đạo hàm trái, phải, trung tâm.

Kỹ thuật Prewitt:

$$P_1 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad P_2 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

Kỹ thuật Sobel:

$$S_1 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad S_2 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Các bước tiến hành như sau:

1. Với mỗi điểm ảnh I(x, y) ta tính:

$I_{S1} = I(x,y) \otimes S_1$

$I_{S2} = I(x,y) \otimes S_2$

$I_{S(x,y)} = |I_{S1}| + |I_{S2}|$

2. Tìm các điểm biên I'(x, y) = $I_{S(x,y)}$

Nếu $I_{S(x,y)} > \theta$ thì I'(x, y) = 1

$I_{S(x,y)} <= \theta$ thì I'(x, y) = 0

2.2.2. Phương pháp đạo hàm bậc 2 Laplace

Các phương pháp lấy đạo hàm bậc 1 ở trên làm việc khá tốt khi mà độ sáng thay đổi rõ nét. Khi mức xám thay đổi chậm, miền chuyển tiếp trải rộng thì phương pháp cho hiệu quả hơn đó là phương pháp sử dụng đạo hàm bậc 2 mà trong phần trên gọi là phương pháp Laplace. Toán tử Laplace được định nghĩa như sau:

$$\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \qquad (7)$$

Ta có:

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}\right)$$

$$\approx \frac{\partial}{\partial x}\,(f(x, y) - f(x-1, y))$$

$$\approx (f(x+1, y) - f(x, y)) - (f(x, y) - f(x-1, y))$$

$$= f(x+1, y) - 2f(x, y) + f(x-1, y)$$

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial y}\right)$$

$$\approx \frac{\partial}{\partial y}\,(f(x, y) - f(x, y-1))$$

$$\approx (f(x, y+1) - f(x, y)) - (f(x, y) - f(x, y-1))$$

$$= f(x, y+1) - 2f(x, y) + f(x, y-1)$$

Do đó: $\nabla^2 f = f(x+1, y) + f(x, y+1) - 4f(x, y) + f(x-1, y) + f(x, y-1)$

Toán tử Laplace dùng nhiều kiểu mặt nạ khác nhau để xấp xỉ rời rạc đạo hàm bậc hai. Dưới đây là các kiểu mặt nạ hay dùng:

$$L_1 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix} \qquad L_2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

$$L_3 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad L_4 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Các bước tiến hành như sau:

1. Với mỗi điểm ảnh I(x, y) ta tính:

   $I_{biên}(x, y) = |I(x, y) \otimes L|$

2. Tìm các điểm biên $I'(x, y) = I_{biên}(x, y)$

   Nếu $I_{biên}(x, y) > \theta$ thì $I'(x, y) = 1$

   $I_{biên}(x, y) <= \theta$ thì $I'(x, y) = 0$

2.3. Một số phương pháp tìm biên phi tuyến

Các toán tử để tìm biên đã trình bày ở trên là các toán tử tuyến tính tổng quát. Với một ảnh đầu vào qua toán tử này ta tạo ảnh đầu ra mà mỗi điểm của nó là tổ hợp tuyến tính của các điểm đầu vào với các hệ số của bộ lọc tương ứng. Như vậy không có quá trình trung gian, so sánh, sắp xếp, lựa chọn hay các phép tính phân chia phức tạp khác. Trong phần này sẽ giới thiệu một số kỹ thuật cơ bản theo phương pháp này.

2.3.1. Phương pháp tìm biên theo hình chóp (pyramid edge detection)

Thông thường trong một số trường hợp ảnh bao gồm nhiều các đường biên trong đó có đường biên dài, có đường biên ngắn, có đường biên hoặc không có đường biên. Vấn đề ở đây cần loại bỏ đi một số đường biên trong ảnh không đáng quan tâm đối với người sử dụng, đó là các đường biên ngắn, đường biên mờ và đường biên không được kết nối với nhau, trong khi cần làm nổi được những đường biên thực sự, đó là những đường biên đậm và dài. Một giải pháp đó là phương pháp tìm biên theo hình chóp. Phương pháp này được định nghĩa như sau:

• Ảnh gốc được chia làm 4 phần bởi chia đôi độ dài mỗi chiều. Mỗi giá trị điểm ảnh trong ảnh nhỏ một phần tư mới là trung bình cộng của bốn điểm ảnh tương ứng trong ảnh gốc theo công thức sau:

$$I_{new}\left(\frac{m}{2},\frac{n}{2}\right)=\frac{1}{4}\left[I(m,n)+I(m+1,n)+I(m,n+1)+I(m+1,n+1)\right]$$

• Cứ thế lặp lại cho đến khi ảnh mới được tạo ra, với biên thực đã được phát hiện, còn các biên khác thì mất.

• Sau đó sử dụng phương pháp tìm biên đối với ảnh nhỏ nhất, tại các điểm biên đã được phát hiện đó lại sử dụng phương pháp tìm biên đối với bốn điểm tương ứng trong ảnh lớn.

• Quá trình cứ tiếp tục được thực hiện cho đến khi đường biên thực của ảnh gốc được xuất hiện

Quá trình thực hiện này sẽ làm mờ đi những đường biên ngắn và mờ và làm nổi lên những đường biên thực. Thật vậy sau mỗi bước tìm biên từ ảnh nhỏ, ở đây ta chỉ xét đến các điểm tương ứng với những điểm đã là biên của ảnh nhỏ mà bỏ qua các điểm không phải là biên mặc dù những điểm này vẫn có thể tạo ra biên đối với 4 điểm tương ứng của nó trong ảnh lớn. Ta xét ví dụ sau:

Từ ảnh gốc I_{new} (chia 4) (B)

$$I=\begin{bmatrix}1&1&2&1\\1&1&2&1\\2&2&3&2\\3&1&2&1\end{bmatrix} \qquad \Rightarrow\begin{bmatrix}1&1.5\\2&2.25\end{bmatrix}$$

Nếu chọn ngưỡng là 1.5 thì ta có biên :

Biên (B) là: $\begin{bmatrix}0&1\\1&1\end{bmatrix}$

Với mỗi biên của B ta lại tìm biên với 4 điểm tương ứng của nó trên ảnh gốc, khi đó sẽ được biên thực là C:

Biên thực (C) là: $\begin{bmatrix}*&*&1&1\\*&*&1&0\\1&1&1&1\\1&0&1&1\end{bmatrix}$

2.3.2 Phương pháp toán tử tìm biên la bàn Kirsch.

Toán tử này được xây dựng trên cơ sở cửa số (3 x 3, 5 x 5, ….)

Trong trường hợp của cửa sổ 3 x 3, nó được mô tả như sau: Mỗi điểm ảnh đầu ra là giá trị lớn nhất trong 8 kết quả xếp chồng của cửa sổ (a) dưới đây với ma trận ảnh. Sau mỗi lần xếp chồng ta quay cửa sổ lọc này đi một góc 45^0 và lấy làm cửa sổ xếp chồng cho lần sau.

Lần 1 dùng cửa sổ xếp chồng sau:

$$H_1 = \begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \qquad (a)$$

Lần 2 quay cửa sổ đi một góc 45^0 thu được mặt nạ:

$$H_2 = \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} \qquad (b)$$

…….

Lần thứ 8 sau lần quay đầu tiên (a) đi một góc 315^0 là:

$$H_8 = \begin{bmatrix} 5 & 5 & -3 \\ 5 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix}$$

Giá trị lớn nhất trong 8 lần xếp chồng tương ứng với điểm ảnh nào đó sẽ cho kết quả điểm ảnh đầu ra khi đó phép toán được mô tả như sau:

$$h(p) = [\max\{|5^* \sum_{i=0}^{2} F_{(i \oplus i)} - 3^* \sum_{i=3}^{7} F_{(i \oplus j)}|\}]$$

Ở đây phép toán a, b được định nghĩa là phần dư của phép chia (a+b) cho 8 và $F_{(j)}$ là trị số của phần dư thứ j của cửa sổ lọc. Thứ tự các phần tử của cửa sổ lọc theo chiều kim đồng hồ, riêng phần tử thứ 9 là phần tử trung tâm của cửa sổ.

2.4. Kỹ thuật dò biên tổng quát

2.4.1. Các khái niệm cơ bản

• Ảnh và điểm ảnh

Ảnh là một mảng số thực 2 chiều (I_{ij}) có kích thước ($m \times n$), trong đó mỗi phần tử $I_{ij}(i = 0,...,m; j = 0,...,n)$ biểu thị mức xám của ảnh tại (i,j) tương ứng.

Ảnh được gọi là nhị phân nếu các giá trị I_{ij} chỉ nhận giá trị 0 hoặc 1.

Ở đây ta chỉ xét tới ảnh nhị phân vì ảnh bất kỳ có thể đưa về dạng nhị phân bằng cách phân ngưỡng. Ta ký hiệu A là tập các điểm 1 (điểm vùng) và $\overline{A}$ là tập các điểm 0 (điểm nền).

• Các điểm 4 và 8-láng giềng

Giả sử (i,j) là một điểm ảnh, các điểm 4-láng giềng là các điểm kề trên, dưới, trái, phải của điểm (i, j) :

$$N_4 = \{(i',j') : |i-i'|+|j-j'| = 1\},$$

Và những điểm 8-láng giềng gồm:

$$N_8 = \{(i',j') : \max(|i-i'|+|j-j'|) = 1\}.$$

Trên hình 1 các điểm P_0, P_2, P_4, P_6 là các 4-láng giềng của điểm P, còn các điểm P_0, P_1, P_2, P_3, P_4, P_5, P_6, P_7 là các 8-láng giềng của P.

P_3	P_2	P_1
P_4	P	P_0
P_5	P_6	P_7

Hình 3. Ma trận 8-láng giềng kề nhau

• Đối tượng ảnh

Cho 1 dãy P_1, P_2…… P_n là một dãy liên thông khi $P_n \equiv P_1$.

4 - liên thông: Khi các P là liên thông 4 láng giềng và nó là một láng giềng

$$P_0, P_2, P_4, P_6, P_0$$

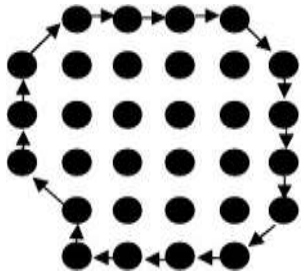
8 - liên thông: Khi nó là một chu trình và các P là liên thông 8 láng giềng

$P_0, P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_0$

• Chu tuyến của một đối tượng ảnh

Chu tuyến của một đối tượng ảnh là dãy các điểm của đối tượng ảnh $P_1,\ldots,P_n$ sao cho P_i và P_{i+1} là các 8-láng giềng của nhau (i=1,...,n-1) và P_1 là 8-láng giềng của $P_n, \forall$ i $\exists$ Q không thuộc đối tượng ảnh và Q là 4-láng giềng của P_i (hay nói cách khác $\forall$ i thì Pi là biên 4). Kí hiệu $<P_1P_2\ldots P_n>$.

Tổng các khoảng cách giữa hai điểm kế tiếp của chu tuyến là độ dài của chu tuyến và kí hiệu Len(C) và hướng $P_i P_{i+1}$ là hướng chẵn nếu P_i và P_{i+1} là các 4 – láng giềng (trường hợp còn lại thì P_iP_{i+1} là hướng lẻ).



Hình 4. Ví dụ về chu tuyến của đối tượng ảnh

2.4.2. Các kỹ thuật dò biên

a, Kỹ thuật Freeman

Tư tưởng: Xuất phát từ một điểm bất kỳ, nếu gặp trắng thì rẽ trái, gặp đen thì rẽ phải.

Cải tiến kỹ thuật trên (Đỗ Ngọc Kỷ - 1992) đưa ra: Gặp trắng rẽ trái, gặp đen lùi lại và rẽ phải.

b, Kỹ thuật nới lỏng

Tạo ra các khớp điểm ảnh với nhau, các khớp này chính là biên của ảnh và chúng thỏa mãn điều kiện hai điểm P, Q tạo thành khớp khi và chỉ khi:

P, Q là 4 láng giềng

$| I_P - I_Q | \geq \theta$ (θ là ngưỡng)

c, Kỹ thuật dò biên theo cặp nền - vùng

Biểu diễn đối tượng ảnh theo chu tuyến thường dựa trên các kỹ thuật dò biên. Có hai kỹ thuật dò biên cơ bản. Kỹ thuật thứ nhất xét ảnh biên thu được từ ảnh vùng sau một lần duyệt như một đồ thị, sau đó áp dụng các thuật toán duyệt cạnh đồ thị. Kỹ thuật thứ hai dựa trên ảnh vùng, kết hợp đồng thời quá trình dò biên và tách biên. Ở đây ta quan tâm cách tiếp cận thứ hai.

Trước hết, giả sử ảnh được xét chỉ bao gồm một vùng ảnh 8-liên thông A, được bao bọc bởi một vành đai các điểm nền. Dễ thấy A là một vùng 4-liên thông chỉ là một trường hợp đối ngẫu với trường hợp trên.

Về cơ bản, các thuật toán dò biên trên một vùng đều bao gồm các bước sau:

- Xác định điểm biên xuất phát
- Dự báo và xác định điểm biên tiếp theo
- Lặp bước 2 cho đến khi gặp điểm xuất phát

Do xuất phát từ những tiêu chuẩn và định nghĩa khác nhau về điểm biên, và quan hệ liên thông, các thuật toán dò biên cho ta các đường biên mang các sắc thái rất khác nhau.

Kết quả tác động của toán tử dò biên lên một điểm biên r_i là điểm biên r_{i+1} (8-láng giềng của r_i). Thông thường các toán tử này được xây dựng như một hàm đại số Boolean trên các 8-láng giềng của r_i. Mỗi cách xây dựng các toán tử đều phụ thuộc vào định nghĩa quan hệ liên thông và điểm biên. Do đó sẽ gây khó khăn cho việc khảo sát các tính chất của đường biên. Ngoài ra, vì mỗi bước dò biên đều phải kiểm tra tất cả các 8-láng giềng của mỗi điểm nên thuật toán thường kém hiệu quả. Để khắc phục các hạn chế trên, thay vì sử dụng một điểm biên ta sử dụng cặp điểm biên (một thuộc A, một thuộc $\overline{A}$), các cặp điểm này tạo nên tập nền vùng, kí hiệu là NV và phân tích toán tử dò biên thành 2 bước:

- Xác định cặp điểm nền vùng tiếp theo

- Lựa chọn điểm biên

Trong đó bước thứ nhất thực hiện chức năng của một ánh xạ trên tập NV lên NV và bước thứ hai thực hiện chức năng chọn điểm biên.

Thuật toán dò biên tổng quát

Bước 1: Xác định cặp nền-vùng xuất phát

Bước 2: Xác định cặp nền-vùng tiếp theo

Bước 3: Lựa chọn điểm biên

Bước 4: Nếu gặp lại cặp xuất phát thì dừng, nếu không quay lại bước 2

Việc xác định cặp nền-vùng xuất phát được thực hiện bằng cách duyệt ảnh lần lượt từ trên xuống dưới và từ trái qua phải rồi kiểm tra điều kiện lựa chọn cặp nền-vùng. Do việc chọn điểm biên chỉ mang tính chất quy ước, nên ta gọi ánh xạ xác định cặp nền-vùng tiếp theo là toán tử dò biên.

Định nghĩa 6 [Toán tử dò biên]

Giả sử T là một ánh xạ như sau: $T: NV \rightarrow NV$

$(b, r) \mapsto (b', r')$

Gọi T là một toán tử dò biên cơ sở nếu nó thoả mãn điều kiện: b',r' là các 8-láng giềng của r.

Giả sử $(b,r) \in NV$; gọi K(b,r) là hàm chọn điểm biên. Biên của một dạng $\Im$ có thể định nghĩa theo một trong ba cách:

Tập những điểm thuộc Acó mặt trên NV, tức là K(b,r)= r

Tập những điểm thuộc $\overline{A}$ có trên NV, tức là K(b,r)= b

Tập những điểm ảo nằm giữa cặp nền-vùng, tức là K(b,r) là những điểm nằm giữa hai điểm b và r.

Cách định nghĩa thứ ba tương ứng mỗi cặp nền-vùng với một điểm biên. Còn đối với cách định nghĩa thứ nhất và thứ hai một số cặp nền-vùng có thể có chung một điểm biên.

CHƯƠNG III: PHƯƠNG PHÁP PHÁT HIỆN BIÊN DỰA VÀO PHÉP TOÁN HÌNH THÁI

3.1. Các phép toán hình thái cơ bản

Hình thái là thuật ngữ chỉ sự nghiên cứu về cấu trúc hay hình học topo của đối tượng trong ảnh. Phần lớn các phép toán hình thái được định nghĩa từ hai phép toán cơ bản là phép "giãn " (Dilation) và phép "co" (Erosion). Các phép toán này được định nghĩa như sau:

Cho I (x, y) là một ảnh, T (i, j) là một ma trận mẫu.

a, Định nghĩa Dilation

Với ảnh nhị phân: $D(I) = I \oplus T = \{ (x + i), (y + j) \}$

Với ảnh đa cấp xám: $D_{(x, y)} I = I \oplus T(x, y) = Max(I (x-i, y-j) + T(i, j))$

Hoặc $D(I) = Max (I (x-i, y-j) + T' (i, j))$ với $T' = Rot_{180} (T)$

b, Định nghĩa Erosion

Với ảnh nhị phân: $E(I) = I \ominus T= \{ (x - i), (y - j) \}$

Với ảnh đa cấp xám: $E_{(x, y)} I = I \ominus T(x, y) = Min (I (x-i, y-j) - T(i, j))$

c, Định nghĩa OPEN

Phép toán mở (OPEN) của I theo T là tập hợp các điểm của ảnh I sau khi đã co và giãn nở liên liếp theo T.

$$D (E(I)) = (I \ominus T) \oplus T$$

d, Định nghĩa CLOSE

Phép toán đóng (CLOSE) của I theo cấu trúc T là tập hợp các điểm của ảnh I sau khi đã giãn nở và co liên liếp theo T.

$$E (D(I))= (I \oplus T) \ominus T$$

e, Xét ví dụ

Cho ảnh: Ma trận mẫu:

$$I = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \qquad T = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

• Tính D(I) = ?

Ta có: $D(I) = I \oplus T = \{ (x + i), (y + j) \} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$

• Tính E(I) = ?

Ta có: $E(I) = I \ominus T = \{ (x - i), (y - j) \} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$

• Tính OPEN(I) = ?

$$OPEN(I) = D(E(I)) = (I \ominus T) \oplus T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

• Tính CLOSE(I) = ?

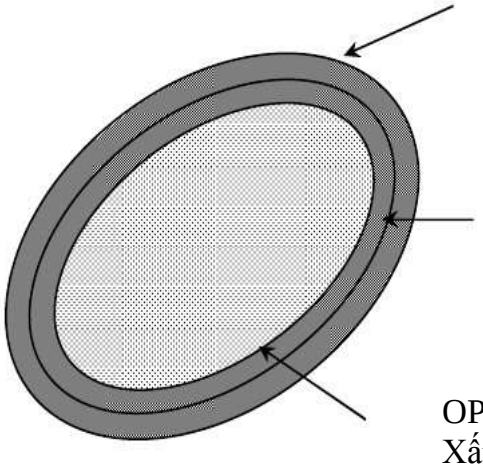
$$CLOSE(I) = E(D(I)) = (I \oplus T) \ominus T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

3.2. Thuật toán phát hiện biên dựa vào phép toán hình thái

Giả thiết E (D(I))= (I $\oplus$ T)$\ominus$ T có thể xem như là một tập xấp xỉ trên của tập I theo mẫu T.

Và D (E(I)) = (I $\ominus$ T) $\oplus$ T thể xem như là một tập xấp xỉ dưới của tập I theo mẫu T.

Từ đó, tập CLOSE (I, T) \ OPEN (I,T) có thể được xem như là xấp xỉ biên của tập I theo mẫu T và quá trình xấp xỉ biên của I theo T được ký hiệu là: I © T.



CLOSE (I, T) = (I $\oplus$ T)$\ominus$ T
Xấp xỉ trên của I (chứa I)

I © T = CLOSE (I, T) \ OPEN (I,T)
Xấp xỉ biên của I theo mẫu T

OPEN (I, T) = (I $\ominus$ T) $\oplus$ T
Xấp xỉ dưới của I (thuộc I)

Để có được kết quả chính xác, việc xác định biên cần phải được thực hiện một cách đối xứng. Do đó người ta thường định nghĩa một dãy các phần tử cấu trúc.

$\{T\} = \{ T^i, 1 \leq i \leq n\}$

Trong đó T^i và T^{i-1} được quay đi một góc và được sử dụng lần lượt theo trình tự

$$I © \{T\} = = \bigcup_{i=1}^{n} (I © T^i)$$

Thuật toán phát hiện biên:

Vào: Ma trận ảnh I và dãy mẫu T = $\{ T^i, 1 \leq i \leq n\}$

Ra: Biên của đối tượng theo mẫu T

Phương pháp:

Bước 1: Tính $I \copyright T^i \quad \forall i = 1 \ldots n$

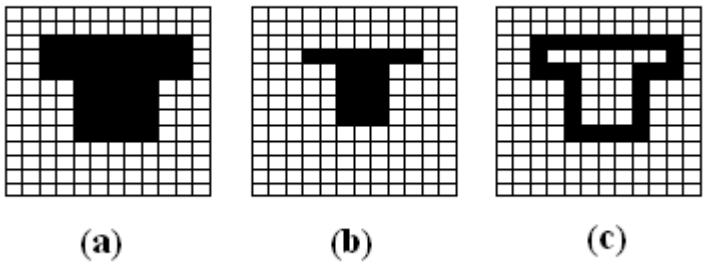
Bước 2: Tính $\bigcup_{i=1}^{n} (I \copyright T^i)$

3.3. Ứng dụng của các phép toán hình thái trong nhận dạng biên ảnh

Biên (hay đường biên) có thể hiểu đơn giản là các đường bao của các đối tượng trong ảnh chính là ranh giới giữa đối tượng và nền. Việc xem ranh giới là phần được tạo lập bởi các điểm thuộc đối tượng và thuộc nền cho phép ta xác định biên dựa trên các phép toán hình thái.

Những điểm ảnh trên biên của một đối tượng là những điểm ảnh trên biên mà có ít nhất một điểm ảnh lân cận thuộc nền. Do bởi lân cận nền cụ thể là không biết trước mà phải tìm, vả lại không thể tạo ra được một cấu trúc đơn mà cho phép phép co hoặc phép dãn dò ra biên, mặc dầu rằng trong thực tế một phép co bởi phần tử cấu trúc đơn giản chính xác là có thể xoá những điểm biên. Mặt khác ta lại có thể áp dụng điều này để thiết kế một phép toán hình thái dò biên. Biên có thể được tách ra bằng cách sử dụng một phép co và ảnh được co sau đó được trừ đi bởi ảnh gốc. Tương tác này sẽ để lại cho ta những điểm ảnh mà được co, đó chính là biên. Điều này được viết như sau:

$$B(I) = (I + T) - (I - T) = CLOSE (I, T) - OPEN (I, T)$$



Hình 5 : Kết quả làm mảnh
a) Ảnh ban đầu
b) Áp dụng Erosion
c) Ảnh ban đầu – đi ảnh đã biến đổi

CHƯƠNG IV: MỘT SỐ PHƯƠNG PHÁP PHÁT HIỆN BIÊN NÂNG CAO

Ngoài các phương pháp phát hiện biên đã trình bày trong chương 2, người ta cũng áp dụng một số phương pháp khác phức tạp và hiệu quả hơn. Các phương pháp này sử dụng mô hình toán học của biên. Một số phương pháp tiêu biểu như: phương pháp Canny, Shen- Castan, Marr- Hildreth…Dưới đây sẽ trình bày một cách tóm lược một số phương pháp.

4.1. Phương pháp Canny

4.1.1. Cơ sở lý thuyết của thuật toán

a, Nguyên lý của thuật toán

Năm 1986, phương pháp này do Canny ở phòng thí nghiệm MIT khởi xướng. Canny đã đưa ra tập hợp các mục tiêu của một phương pháp phát hiện biên và đưa ra một phương pháp tối ưu để thực hiện các mục tiêu đó. Phương pháp này gọi là phương pháp Canny.

Canny đưa ra ba điểm chính mà một phương pháp phát hiện biên phải xác định được đó là:

Mức lỗi: Phương pháp phải làm sao chỉ có hiệu quả đối với các điểm biên, phải tìm ra tất cả các biên và không có đường biên nào bị bỏ sót.

Định vị: Khoảng cách giữa các điểm biên được tìm thấy trong giải thuật và biên trong thực tế phải càng nhỏ càng tốt.

Hiệu xuất: Không được phép chỉ ra nhiều biên trong khi chỉ có một biên tồn tại. Canny giả thiết rằng nhiễu trong ảnh tuân theo phân bố Gauss, đồng thời ông cho rằng một phương pháp tìm biên thực chất là một bộ lọc nhân xoắn có khả năng làm mịn nhiễu và định vị được cạnh. Vấn đề là làm sao để tìm ra được một bộ lọc như vậy đồng thời phải thỏa mãn tối ưu nhất ba tiêu chuẩn đã đặ ra ở trên.

b, Nội dung của thuật toán

Về mặt một chiều, đáp ứng của bộ lọc f với biên G được tính bởi tích phân chập:

$$H = \int_{-w}^{w} G\ (-x)\, f(x)\, dx$$

ở đây ta giả sử đáp ứng của bộ lọc ở ngoài khoảng [-w, w] là bằng 0. Ba tiêu chuẩn trên được biểu diễn toán học như sau:

$$\text{SNR} = \frac{A\left|\int_{-w}^{0} - f(x)\, dx\right|}{n_0\sqrt{\int_{-w}^{w} f^2(x)\, dx}}$$

$$\text{Localization} = \frac{A\left|f'(0)\right|}{n_0\sqrt{\int_{-w}^{w} f'^2\, dx}}$$

$$X_{ZC} = \sqrt{\frac{\left(\int_{-w}^{w} f'^2(x)\, dx\right)}{\int_{-w}^{w} f''^2(x)\, dx}}$$

Giá trị SNR là tỉ số giữa tín hiệu ra so với nhiễu (output signal to noise ratio) hay còn gọi là tỉ xuất lỗi (error rate). Giá trị này càng lớn càng tốt bởi ta cần nhiều tín hiệu và ít nhiễu.

Giá trị Localization là nghịch đảo của khoảng cách từ biên tìm được cho tới biên thực. Giá trị này cũng càng lớn càng tốt bởi nó đồng nghĩa với khoảng cách từ biên tìm được cho tới biên thực càng bé càng tốt.

Giá trị X_{ZC} là một ràng buộc. Nó là khoảng cách trung bình giữa của các điểm 0 và f' và nó có nghĩa rằng trong một vùng nhỏ, sẽ không có nhiều đáp ứng của f đối với cùng một biên.

Canny đã xây dựng một bộ lọc f làm cực đại hóa được tích SNR xLocalization và thỏa mãn ràng buộc X_{ZC}. Vì kết quả quá phức tạp để có thể biểu diễn giải tích, Canny đã đưa ra một xấp xỉ có hiệu quả, là đạo hàm bậc nhất của hàm Gaussian. Nhắc lại rằng hàm Gaussian có dạng sau:

$$G(x) = e^{\frac{-x^2}{2\sigma^2}}$$

Đạo hàm bậc nhất của Gaussian theo x của G(x):

$$G'(x) = \left(-\frac{x}{\sigma^2}\right) e^{\left(\frac{-x^2}{2\sigma^2}\right)}$$

Hàm Gaussian trong không gian hai chiều có dạng sau:

$$G(x, y) = \sigma^2 e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

Và G có đạo hàm theo cả hai hướng x, y. Xấp xỉ của bộ lọc tối ưu Canny trong phát hiện biên là G', vì thế cách nhân chập ảnh đầu vào với G' ta được ảnh E đã được cải thiện biên, thậm chí cả trong trường hợp có nhiễu.

Việc nhân xoắn dễ dàng thực hiện, tuy nhiên nó phải trả một giá trị tính toán khá đắt, đặc biệt là nhân xoắn với mảng hai chiều. Tuy nhiên một phép nhân xoắn với mảng hai chiều Gaussian có thể chia thành hai phép nhân xoắn với mặt nạ Gaussian một chiều. Việc vi phân có thể thực hiện bằng phép tính chập với mảng một chiều tạo nên hai ảnh: Một là thành phần x của việc nhân xoắn với G' và cái còn lại là thành phần y.

4.1.2 . Mô tả thuật toán

a. Các bước của thuật toán

Giải thuật phát hiện biên Canny được trình bày như sau:

1. Đọc ảnh I cần xử lý

2. Tạo một mặt nạ G để nhân xoắn với I. Độ lệch tiêu chuẩn của mặt nạ này chính là tham số để tách cạnh.

3. Tạo một mặt nạ cho đạo hàm bậc nhất của Gassian theo hướng x, y và gọi là Gx, Gy và giá trị vẫn được giữ như ở bước 2.

4. Nhân xoắn ảnh I cùng với G dọc theo các hàng tạo ảnh thành phần x gọi là I_x và theo các cột tạo ra ảnh I_y.

5. Nhân xoắn I_x với Gx để sinh ra I'_x: thành phần x của I được nhân xoắn với đạo hàm của Gaussian, và nhân xoắn I_y với Gy để tạo ra I'_y.

6. Nếu lúc này bạn muốn xem kết quả, thành phần x, y phải được kết hợp khi đó độ lớn tại điểm (x, y) được tính như sau:

$$M(x, y) = \sqrt{I'_x(x, y)^2 + I'_y(x, y)^2}$$

Độ lớn được tính theo kiểu đã tính đối với Gradient.

b. Giải thích thuật toán

Phần cài đặt thuật toán này có trong phần cuối cùng của đề tài này, sau đây là phần giải thích về chương trình.

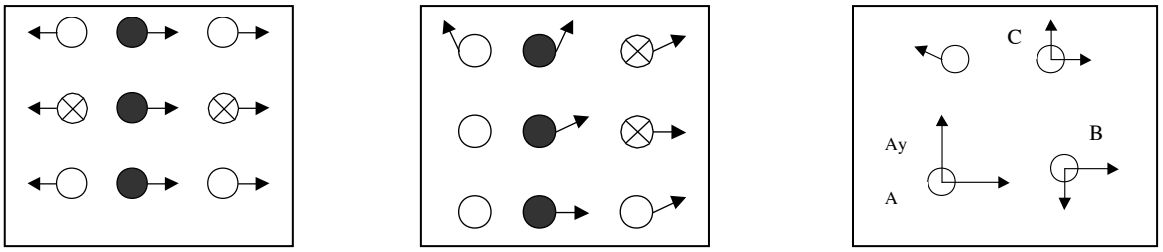
Chương trình chính là mở file ảnh và đọc nó, đồng thời đọc các tham số như độ lệch tiêu chuẩn.

Sau đó gọi hàm Canny, đây là hàm thực hiện các tính toán chủ yếu. Việc mà Canny làm đầu tiên là tính mặt nạ lọc Gaussian (Gauss) và đạo hàm mặt nạ lọc Gauss (dGau). Kích thước của mặt nạ được sử dụng phụ thuộc vào độ lệch tiêu chuẩn. Chương trình này tự động tính kích thước của mặt nạ.

Tiếp theo là tính nhân xoắn như trong bước 4, để làm mịn ảnh. Hàm separable_convolution làm việc này, đầu vào của hàm là ảnh và mặt nạ, hàm trả lại thành phần x, y của phép nhân xoắn (được gọi là smx và smy). Tiếp theo phép nhân xoắn ở bước 5 được thực hiện bởi gọi hàm dxy_ separable_convolution hai lần, một lần cho x và một lần cho y. Lúc này các ảnh thành phần x, y (được gọi là dx, dy) được nhân xoắn với G', hàm norm thực hiện việc này. Cho tới lúc này ta có hai ảnh được làm nổi cạnh theo chiều ngang và dọc.

Để lọc ra các điểm cạnh Canny đề xuất ra phương pháp nonmaximum suppresion. Ý tưởng cơ bản của quá trình này là: mỗi điểm cạnh có một hướng gắn liền với chúng, độ lớn của Gradient tại các điểm cạnh lớn hơn của các điểm láng giềng khác của nó. Khi đó những giao điểm không không phải là cực đại địa phương được bỏ đi.

Hình 6 minh họa như sau:



(a) (b) (c)

Hình 6 giải thích quá trình này bằng việc sử dụng hình học. Hình này chỉ ra một vùng có kích thước 3*3 bao quanh một điểm cạnh, cạnh trong trường hợp này là dọc. Mũi tên chỉ ra hướng Gradient tại mỗi điểm cạnh và độ dài mũi tên tỉ lệ với độ lớn của Gradient. Ở đây quá trình nonmaximum có nghĩa rằng điểm chính giữa (điểm đang xét) phải có Gradient lớn hơn Gradient của các láng giềng nằm trên phương Gradient của điểm đang xét, đó là hai điểm được đánh dấu là $\otimes$. Thực vậy từ điểm chính giữa đi theo hướng của Gradient cho đến khi gặp một điểm khác cùng hướng, đó là láng giềng thứ nhất. Bây giờ lại bắt đầu từ điểm trung tâm đi theo hướng ngược lại cho đến khi gặp điểm có Gradient cùng với hướng này, đây là láng giềng thứ hai. Di chuyển từ điểm láng giềng này sang điểm láng giềng kia ngang qua cạnh vì thế độ lớn của Gradient sẽ lớn nhất tại điểm cạnh.

Đây là trường hợp rõ ràng: hướng của Gradient là ngang và các điểm láng giềng là các điểm ở bên trái và bên phải. Thật không may là điều này không xảy ra. Nếu hướng Gradient là tùy ý, thì theo hướng đó thường dẫn ta tới giữa hai điểm. Vậy làm sao có thể ước lượng được giá trị Gradient tại một điểm so với các điểm láng giềng. Giá trị của điểm mà theo hướng Gradient ta bắt gặp không thể biết trước được, tuy nhiên có thể ước lượng được từ Gradient của các điểm láng giềng. Giả thiết rằng sự thay đổi của Gradient là một hàm liên tục. Thêm vào đó giả thiết rằng sự thay đổi Gradinet gữa hai điểm bất kỳ là một hàm tuyến tính thì Gradient tại bất

kỳ chỗ nào giữa các điểm cũng có thể được xấp xỉ bằng một phếp nội suy tuyến tính.

Trường hợp tổng quát được chỉ ra ở hình (b) thì Gradient của các điểm bây giờ là khác nhau, và đi theo Gradient từ điểm trung tâm sẽ đưa ta vào giữa những điểm được đánh dấu là x theo hướng ngược lại sẽ đưa chúng ta vào giữa những điểm được đánh dấu là y. Ta xét trường hợp chỉ có những điểm được đánh dấu x ở hình (c), các trường hợp khác tương tự. Điểm có tên là A, ta xét những điểm B, C là những điểm láng giềng. Vector thành phần của Gradient tại A là Ax, Ay, quy ước tương tự với B, C.

Mỗi điểm nằm trên đường lưới có tọa độ nguyên. Điều này có nghĩa rằng điểm A, B khác nhau một đơn vị khoảng cách theo hướng x. Ta phải xác định rằng đường lưới nào sẽ bị cắt đầu tiên khi đi từ A theo hướng Gradient. Sau đó độ lớn Gradient sẽ được nội suy tuyến tính từ 2 điểm nằm trên đường lưới và vị trí giao điểm (Px, Py). Trong hình (c) giao điểm được đánh dấu ' + ' và nằm giữa B, C. Độ lớn Gradient tại điểm này được đánh giá như sau:

$G = (P_y - Cy)$ Norm (C) + ($B_y - P_y$) Norm(B)

Trong đó hàm Norm tính độ lớn Gradient

Tất cả các điểm ở trong ảnh vừa được lọc đều được xử lý theo cách này, độ lớn của Gradient được đánh giá tại hai vị trí hai bên của điểm cạnh, và độ lớn của Gradient của điểm cạnh phải lớn hơn độ lớn Gradient của các láng giềng. Trong trường hợp tổng quát có 8 trường hợp chính để kiểm tra. Trên thực tế có một số cách tính tắt vì tính hiệu quả nhưng phương pháp trên là phương pháp cơ bản nhất trong tất cả các thuật toán của Canny. Hàm nonmax_supress tính độ lớn tạicác điểm dựa trên phương pháp này và đặt các giá trị bằng 0 trừ khi điểm đó là cực đại địa phương.

Sau khi thực hiên xong các bước của thuật toán, ta được ảnh cuối cùng là ảnh đa cấp xám. Vậy cần xác định điểm nào là điểm cạnh, điểm nào là không. Như một

bước mở rộng thêm, Canny khuyến cáo quá trình phân nghưỡng nên sử dụng hiện tượng trễ. Phân nghưỡng trễ sử dụng một nghưỡng cao Th và một nghưỡng thấp Tl. Bất cứ điểm nào trong ảnh có giá trị lớn hơn Th thì được coi là điểm cạnh và được đánh dấu. Sau đó, bất cứ điểm nào là láng giềng của điểm cạnh này và có giá trị lớn hơn Tl cũng được coi như điểm cạnh và cũng được đánh dấu. Việc đánh dấu các láng giềng cũng có thể được làm như trong quá trình phân ngưỡng trễ.

4.2. Phương pháp Shen - Castan

Như đã nói ở trên, Canny đã định nghĩa một tập hợp các tiêu chuẩn dành cho việc tách cạnh và các tiêu chuẩn này rất hợp lý và đầy đủ. Tuy nhiên không có một lý do gì để nghĩ rằng đó là phương pháp tối ưu nhất. Điều này có nghĩa rằng khái niệm tối ưu chỉ là một khái niệm tương đối và rất có thể có một phương pháp tối ưu hơn. Sau đây ta sẽ khảo sát một thuật toán cũng chạy tốt trong nhiều trường hợp khác nhau của ảnh đó là phương pháp phát hiện biên Shen- Castan.

4.2.1. Cơ sở lý thuyết của thuật toán

a, Nguyên lý của thuật toán

Shen và Castan có cùng quan điểm với Canny về một dạng thức chung trong phát hiện điểm biên. Tuy nhiên họ đưa ra một hàm tối ưu khác, đó là khái niệm tối thiểu hóa (theo một chiều).

$$C^2_N = \frac{4\int_0^{\infty} f^2(x)\,dx \cdot \int_0^{\infty} f'^2(x)\,dx}{f^4(0)}$$

Nói một cách khác hàm mà làm cực tiểu C_N là bộ lọc làm mịn tối ưu cho việc phát hiện biên. Tuy nhiên, Shen và Castan lại đề cập đến việc thuật toán sẽ nhận ra nhiều cạnh trong khi chỉ có một cạnh tồn tại.

b, Nội dung của thuật toán

Hàm lọc tối ưu mà họ đưa ra được gọi là bộ lọc hàm mũ đối xứng vô hạn ISEF (Infinine Symmetric Exponential Filter) :

$$f(x) = \frac{P}{2} e^{-P|x|}$$

Theo Shen và Castan, bộ lọc này cho tỉ lệ giữa tín hiệu và nhiễu tốt hơn so với bộ lọc của Canny và cho giá trị Localization tốt hơn. Điều này có thể là bởi vì trong thuật toán Canny bộ lọc tối ưu được xấp xỉ bằng đạo hàm của bộ lọc Gauss, trong khi đó Shen và Castan sử dụng bộ lọc tối ưu một cách trực tiếp, hoặc cũng có thể là do những tiêu chuẩn tối ưu khác nhau được thể hiện khác nhau trong thực tế. Tuy nhiên Shen - Castan lại không đưa ra những tiêu chuẩn đa đáp ứng (multiple-reponse), nên rất có thể phương pháp của họ có thể sinh ra nhiễu và làm mờ biên.

Bộ lọc ISEF trong không gian hai chiều có công thức:

$$f(x, y) = a. e^{-P(|x|+|y|)}$$

Công thức này có thể được áp dụng vào ảnh theo cách coi nó như là đạo hàm của bộ lọc Gaussian, như là lọc một chiều theo hướng x, sau đó theo hướng y. Tuy nhiên Shen và Castan đã phát triển bộ lọc của họ thành bộ lọc hồi tiếp một chiều.

Hàm bộ lọc f nói trên là một hàm số thực liên tục. Nó có thể được viết lại dưới dạng rời rạc và lấy mẫu như sau:

$$f[i, j] = \frac{1-b}{1+b}^{|x|+|y|}$$

Để nhân chập ảnh như bộ lọc này, trước hết ta tiến hành lọc đệ quy theo hướng x và thu được r [i, j] như sau:

$$y_1[i, j] = \frac{1-b}{1+b} I[i, j] + by_1[i, j-1]$$

$$y_2[i, j] = b.\frac{1-b}{1+b} I[i, j] + by_1[i+1, j]$$

$$r [i, j] = y_1[i, j] + y_2[i, j+1]$$

Với i = 1…..M và j = 1…….N

Cùng với các điều kiện biên:

$$I[i, j] = 0$$

$$y_1[i, 0] = 0$$

$$y_2[i, M+1] = 0$$

Sau đó tiến hành lọc trên r[i, j] theo hướng y, sẽ tạo ra kết quả cuối cùng của bộ lọc: y[i, j] như sau:

$$y_1[i, j] = \frac{1-b}{1+b} I[i, j] + by_1[i - 1, j]$$

$$y_2[i, j] = b.\frac{1-b}{1+b} I[i, j] + by_1[i+1, j]$$

$$y[i, j] = y_1[i, j] + y_2[i + 1, j]$$

Với i = 1…..M và j = 1…….N

Với các điều kiện biên:

$$I[0, j] = 0$$

$$y_1[0, j] = 0$$

$$y_2[N+ 1, j] = 0$$

Do sử dụng lọc đệ quy nên tốc độ nhanh hơn rất nhiều so với nhân chập.

Sau khi được lọc ảnh , vấn đền đặt ra làm sao phải phát hiện được các điểm biên. Biên được nhận dạng bằng việc tìm các giao điểm không của toán tử Laplace.

4.2.2 Hoạt động thuật toán

a, Các bước của thuật toán

Dựa trên những phân tích ở trên ta có thể đưa ra một thuật toán tìm biên như sau:

1. Đọc ảnh từ tệp cần xử lý.
2. Lọc ảnh bằng phương pháp lọc đệ quy theo công thức ở trên.

3. Tìm các giao điểm không sau khi áp dụng toán tử Laplace.

4. Thực hiện quá trình phân nghưỡng.

b, Giải thích thuật toán

Việc đọc ảnh từ tệp được thực hiện bằng thủ tục Input_Bitmap.

Sau khi đọc ảnh ta lọc ảnh ở bước 2 của thuật toán bằng việc dùng đệ quy hàm ISEF. Việc lọc được thực hiện bằng hàm ISEF, hàm ISEF _ vert lọc theo dọc và ISEF _horiz lọc theo ngang. Giá trị b là tham số để lọc và được nhập bởi người dùng.

Việc tiếp theo là phải tìm ra các điểm cạnh tương ứng với bước 3 trong thuật toán, để tìm được các điểm cạnh ta áp dụng toán tử Laplace rồi tìm các giao điểm không. Tuy nhiên theo Shen và Castan, một sự xấp xỉ toán tử Laplace có thể thu được một cách nhanh chóng bằng việc lấy ảnh gốc trừ ảnh đã làm mịn và tạo ảnh nhị phân. Thật vậy, nếu ảnh lọc là S và ảnh gốc là I ta có:

$$S[\,i, j\,] - I[\,i, j\,] \approx \frac{1}{4a^2} I(\,i, j) \,.\, \nabla^2 f[\,i, j]$$

Ảnh kết quả B = S - 1 được gọi là giới hạn dài Laplace (band- limited Laplacian) của ảnh. Từ đó ta tính ảnh nhị phân Laplace(Binary Laplacian Image) BLI bằng cách đặt các điểm ảnh ó giá trị dương trong B thành giá trị 1 và những điểm ảnh còn lại thành giá trị 0 (được thực hiện bởi hàm compute_bli trong phần mã nguồn). Các điểm nằm trên đường biên giữa các vùng trong BLI có thể được coi là những điểm biên, hoặc có thể áp dụng một số phương pháp cải thiện ảnh để nâng cao chất lượng đường biên cụ thể như sau:

Việc nâng cấp đầu tiên là việc sử dụng phương pháp loại bỏ giao điểm không lỗi (false zero-crossing supression), tương tự như phép giới hạn không cực đại(nonmaximum suppression) trong phương pháp Canny. Tại mỗi điểm biên, đạo hàm bậc hai tại điểm này sẽ là giao điểm không. Điều này có nghĩa rằng Gradient tại điểm đó là cực đại hoặc cực tiểu. Nếu dấu đạo hàm bậc hai thay đổi từ (+) sang (-)

thì giao điểm không đó được gọi là giao điểm không dương. Và nếu nó thay đổi từ (-) sang (+) thì được gọi là giao điểm không âm. Giả thiết rằng những giao điểm không dương sẽ có Gradient dương, những giao điểm không âm sẽ có Gradient âm. Tất cả các giao điểm không khác đều là sai và không được gọi là điểm cạnh. Việc làm này được thực hiện bởi hàm is_candidate_edge trong phần chương trình ISEF.

Trong trường hợp ảnh gốc bị nhiễu rất nặng, việc áp dụng một hàm ngưỡng chuẩn sẽ có thể không phù hợp. Các điểm biên có thể được lấy ngưỡng áp dụng một hàm nghưỡng tổng thể cho Gradient , nhưng Shen và Castan đề xuất một phương pháp gọi là phù hợp Gradient (adaptive gradient method). Lấy một cửa sổ độ rộng cố định W, đặt nó vào các điểm biên trong BLI sao cho trọng tâm của cửa sổ trùng với tâm của điểm biên. Nếu đây thực sự là một điểm biên, thì cửa sổ sẽ chia làm 2 phần có mức xám khác nhau được tách biệt bởi một biên (đường bao zero- crossing). Xấp xỉ Gradient tốt nhất tại điểm này chính là sự sai khác giữa hai mức xám của hai vùng, trong đó một vùng tương ứng với các điểm có giá trị 0 trong BLI , phần kia có giá trị các điểm là 1.

Cuối cùng là việc phân nghưỡng trễ tương tự như trong thuật toán Canny.

4.3. Phương pháp phát hiện biên Marr- Hildreth

4.3.1. Cơ sở lý thuyết chung

Theo Marr (1980): " Mục tiêu của xử lý ảnh là tạo ra điểm cốt yếu nhưng lại có được sự diễn tả đầy đủ nhất về một bức ảnh, nhằm có thể sử dụng để xác định được độ phản xạ và độ sáng của bề mặt, cùng với hướng và khoảng cách của chúng đối với người xem ".

Sự diễn tả ở mức thấp nhất được ông gọi là phác họa cơ bản (primal sketch) là thành phần chính của các biên.

Marr đưa ra một giải thuật phát hiện biên mô tả như sau:

1. Nhân chập ảnh I với một hàm Gaussian 2 chiều.
2. Tính Laplace của ảnh sau khi đã nhân chập, giá trị này gọi là L.

3. Điểm biên là những điểm có sự đi qua điểm 0 trong L (zero crossing) (2 giá trị đối xứng nhau qua điểm đó trong L là đối dấu nhau).

Hàm G được sử dụng trong tích chập trên là một hàm Gaussian 2 chiều :

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

Để tiến hành nhân chập trên một ảnh số, ta phải thu nhận hàm Gaussian nói trên thành một ảnh 2 chiều. Sau khi đã nhân chập ta áp dụng toán tử Laplace:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

Giá trị này có thể được tính bằng cách sử dụng sai phân. Tuy nhiên số thứ tự thực hiện là không quan trọng, cho nên ta có thể tính Laplace có giá trị hàm Gaussian, sau đó thu nhận thành ảnh giá trị hàm này, tạo ra một mặt nạ nhân chập có thể được áp dụng và cho cùng một kết quả như trên. Giá trị Laplace của hàm Gaussian (LoG) là:

$$\nabla^2 G_\sigma = \left(\frac{r^2 - 2\sigma^2}{\sigma^4} \right).e$$

Zero crossing tại điểm P có nghĩa là giá trị của 2 điểm lân cận đối xứng nhau trên một hướng nào đó là đối dấu nhau. Ví dụ: nếu biên tại điểm P là đối xứng nhau thì điểm nằm bên trái của P sẽ khác dấu với điểm nằm bên phải P. Như vậy có 4 trường hợp phải kiểm tra: trên/ dưới, trái/ phải và hai đường chéo. Việc kiểm tra này phải được thực hiện trên mọi điểm ảnh trong Laplace của hàm Gaussian.

4.3.2. Mô tả thuật toán

Trên cơ sở lý thuyết đã trình bày ta có thể đưa ra các bước cho một phép phát hiện biên sử dụng phương pháp Marr- Hildreth như sau:

1. Đọc ảnh cần xử lý I.

2. Dựa vào giá trị độ lệch tiêu chuẩn σ, ta xây dựng một ma trận Gaussian theo công thức trên. Kích thước của ma trận được tính theo số kích thước

tối đa của ma trận sao cho phần tử cuối cùng của ma trận có giá trị lớn hơn giá trị ε nào đó.

3. Tính ma trận vuông Laplace của Gaussian (LoG) dựa trên ma trận Gaussian ở trên theo công thức đã cho.

4. Nhân chập ma trận ảnh với LoG, được ma trận biên độ Gradient (MaG).

5. Tính ma trận Zero crossing với ma trận MaG vừa th được. Các điểm có thể là biên là các điểm có giá trị 1 trong ma trận Zero crossing.

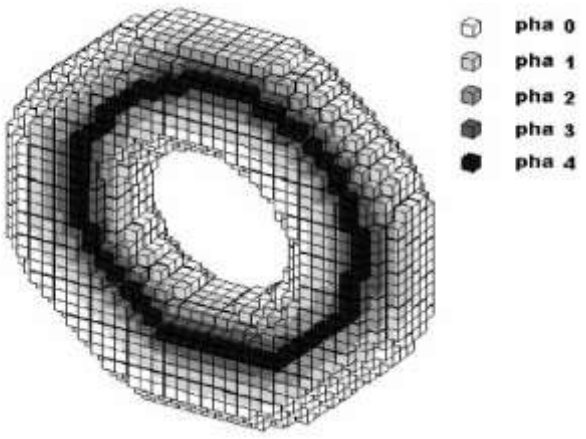
6. Lấy ngưỡng và hiển thị ảnh.

ỨNG DỤNG CÁC PHƯƠNG PHÁP PHÁT HIỆN BIÊN

Sau khi bắt tay vào việc tìm hiểu các phương pháp và các thuật toán phát hiện biên, em đã đưa ra được ứng dụng của các thuật toán trên. Các kết quả của ứng dụng này bao gồm: Tìm xương dựa trên làm mảnh ảnh, phát hiện góc nghiêng văn bản, phát hiện và nhận dạng đối tượng chuyển động…Tuy nhiên do thời gian và trình độ còn hạn chế, trong phần cuối của đồ án này em xin trình bày tóm tắt ứng dụng của biên trong việc tìm xương dựa trên làm mảnh.

Xương được coi như là hình dạng cơ bản của đối tượng với số ít các điểm ảnh, ta có thể lấy được các thông tin cơ bản của đối tượng thông qua xương.

Thuật toán tìm xương dựa trên làm mảnh là một trong những thuật toán quan trọng trong xử lý ảnh. Thuật toán làm mảnh được phân loại dựa vào phương pháp xử lý các điểm, thường chia thành 2 loại: Thuật toán làm mảnh song song và thuật toán làm mảnh tuần tự. Thực chất của quá trình làm mảnh là việc xoá dần các biên theo một thứ tự và một điều kiện xoá nào đó để thu được xương của đối tượng.



Hình 7: Minh họa quá trình làm mảnh

Thuật toán làm mảnh tổng quát bao gồm các bước cơ bản sau:

Bước 1: Dò biên theo thuật toán dò biên chuẩn trên. Nhằm phát hiện tất cả các đường biên của đối tượng.

Bước 2: Với mỗi đường biên. Kiểm tra tính thỏa mãn của điểm biên theo các thông tin khai báo trước và kiểm tra điểm biên nếu thỏa mãn điều kiện xoá thì xóa đi.

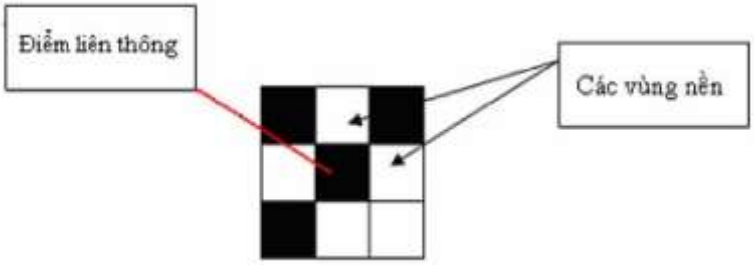
Bước 3: Nếu không còn điểm biên nào được đánh dấu xoá thì dừng, ngược lại thì quay lại bước 1.

Sự khác biệt của thuật toán chính là điều kiện xóa, với các điều kiện xóa khác nhau sẽ cho ta các thuật toán làm mảnh khác nhau. Mỗi thuật toán có một số ưu điểm và nhược điểm khác nhau.

Một số thông tin khai báo trước cơ bản:

a) Điều kiện đảm bảo tính liên thông

Ta đã biết một điểm ảnh là điểm liên thông khi ít nhất 2 điểm nền không gần nhau trong 8-láng giềng. Điều này đảm bảo rằng các điểm biên trong điều kiện xoá không bị xoá đi các thành phần liên thông.



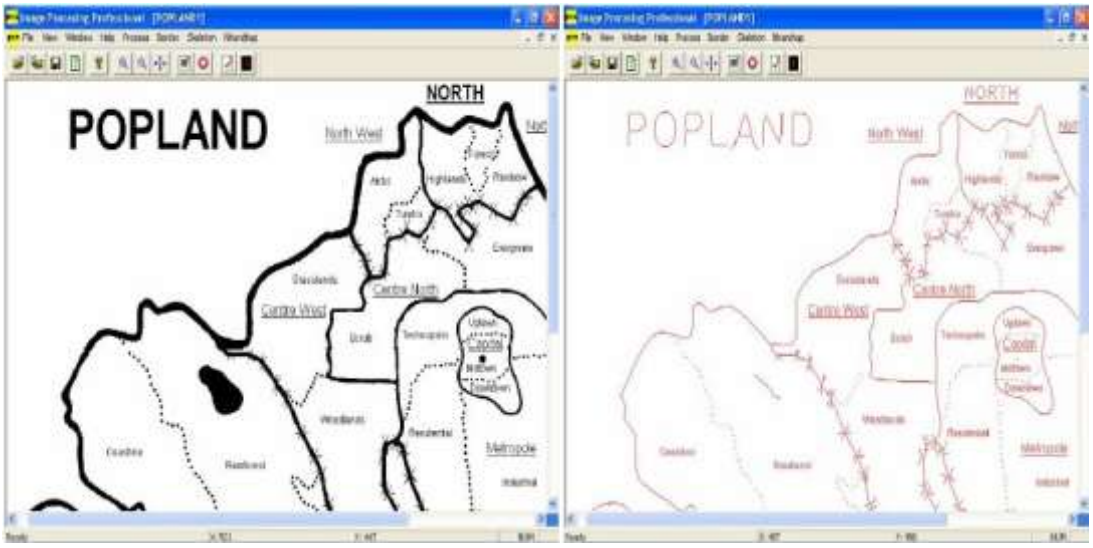
Hình 8: Điểm liên thông

b) Điều kiện đảm bảo xương có dạng mảnh

Một điểm biên được gọi là điểm xương dạng mảnh nếu thoả mãn:

- Điểm ảnh có điểm láng giềng treo góc khi điểm láng giềng góc này có mầu đối tượng hai láng giềng kề nó màu nền.
- Điểm cô lập là điểm không có láng giềng nào hay nói cách khác đối tượng chỉ có một điểm.
- Điểm cuối của đường thẳng là điểm có duy nhất một láng giềng.
- Điểm liên kết là điểm mà khi ta bỏ nó đi thì mất tính liên thông.

Áp dụng tiêu chuẩn này kết hợp với thuật toán tìm xương dựa trên làm mảnh theo điều kiện xoá của các thuật toán đã biết để thu được xương có dạng mảnh mà vẫn đảm bảo tính liên thông.



a, Ảnh gốc b, ảnh xương (dạng mảnh)

Hình 9. Xương dạng mảnh đảm bảo một số tiêu chuẩn đề ra

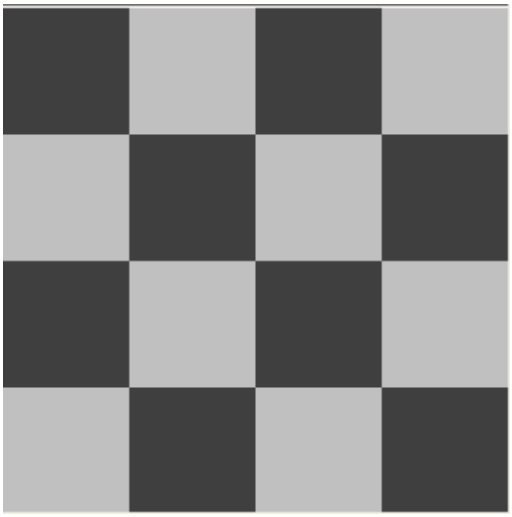
CHƯƠNG V: CÀI ĐẶT VÀ ĐÁNH GIÁ CÁC THUẬT TOÁN

Trong chương cuối này ta sẽ đi cài đặt và đánh giá các phương pháp tìm biên cơ bản như: phương pháp Gradient, Laplace, Sobel. Tiếp theo là đánh giá và so sánh hai phương pháp tìm biên nâng cao là Canny và Shen-Castan, sở dĩ việc đánh giá hai phương pháp này chủ yếu được sử dụng trong các ứng dụng của xử lý ảnh hiện nay. Những chi tiết cài đặt cụ thể cũng được giới thiệu trong chương này.

5.1. Các phương pháp cổ điển

5.1.1. Thuật toán

Ta có ảnh đầu vào như sau:



Từ ảnh gốc ban đầu ta thực hiện các bước sau để tìm biên:

Bước 1: Đọc ảnh, gán giá trị hàng i=0, cột j=0

Bước 2: Lấy giá trị của cửa sổ ảnh trượt với kích thước 3×3 bắt đầu vị trí (i.j)

Bước 3: Tính tích nhân chập với mặt nạ với cửa sổ ảnh

Bước 4: Thực hiện phép phân ngưỡng trên toàn ảnh

Bước 5: Thoát

Các mặt nạ tương ứng với các phương pháp như sau:

Mặt nạ của phương pháp Gradient:

$$H_1 = \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix} \text{ và } \quad H_2 = \begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}$$

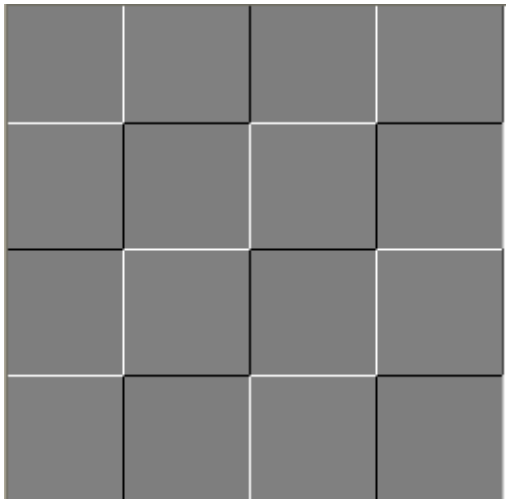
Mặt nạ của phương pháp Sobel:

$$S_1 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad S_2 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

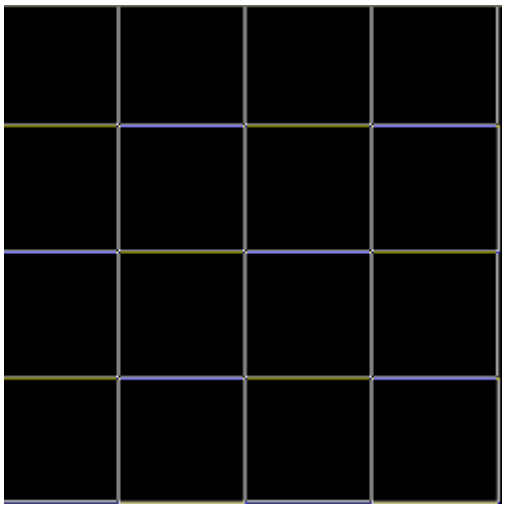
Mặt nạ của phương pháp Laplace:

$$L_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{or} \quad L_2 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad \text{or} \quad L_3 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

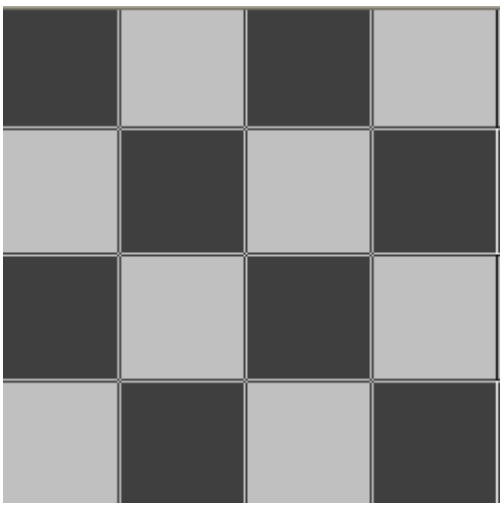
Sau khi thực hiện các thuật toán và áp dụng các mặt nạ vào ảnh gốc ta thu được kết quả sau:



a, Gradient



b, Laplace



c, Sobel

5.1.2. Nhận xét

Sau khi áp dụng vào ảnh đối với phương pháp Gradient (a) ta thấy các biên được tách đã có độ mảnh, nhiễu trong ảnh đã được xử lý tốt, không coi những nơi có nhiều nhiễu là cạnh. Tuy nhiên thuật toán lại không chỉ ra được các cạnh mà tại đó độ thay đổi cấp xám là không lớn. Kết quả là đường biên bị đứt quãng nhiều, gây mất thông tin của ảnh. Nguyên nhân của việc không nhận ra cạnh nào là do phương pháp lấy sai phân tại những biên mà cấp xám thay đổi ít, kết quả lấy sai phân là nhỏ khiến quá trình phân ngưỡng không thể nhận ra những cạnh này.

Đối với phương pháp Sobel đã khắc phục được việc mất cạnh đồng thời cũng xử lý tốt vè nhiễu. Tuy nhiên ảnh thu được lại có độ rộng 1 pixel.

5.2. Phương pháp Canny và phương pháp Shen-Castan

5.2.1. So sánh hai thuật toán

• Phương pháp Canny

- Nhân xoắn ảnh cùng đạo hàm của mặt nạ Gause
- Thực hiện quá trình nonmaximum để loại bỏ các điểm không phải là cực đại
- Phân ngưỡng ảnh bằng quá trình phân ngưỡng trễ.

• Phương pháp Shen - Castan

- Nhân xoắn ảnh với bộ lọc ISEF

- Tính ảnh nhị phân Lacplacian

- Khử các giao điểm không bị lỗi

- Thực hiện phân ngưỡng phù hợp với Gradient

- Phân ngưỡng ảnh bằng quá trình phân ngưỡng trễ

5.2.2. Đánh giá và so sánh hai phương pháp

Khi thực hiện việc nhân xoắn thuật toán Canny sử dụng phương pháp bao gói nên những khu vực gần các đường biên xuất hiện các điểm đen (đôi khi những điểm này bị coi là nhiễu). Trong khi đó thuật toán ISEF sử dụng bộ lọc đệ quy làm cho việc nhân xoắn theo phương pháp bao gói rất khó thực hiện. Trên thực tế không thực hiện các nguyên tắc này, thay vào đó ảnh được nhúng vào một vùng rộng hơn trước khi sử lý. Khi đó kết quả là đường biên của những ảnh này sẽ là trắng tại những nơi mà mặt nạ tích chập vượt quá ảnh.

Từ việc đánh giá trên ta có nhận xét: trong trường hợp nhiều nhiễu phương pháp ISEF tỏ ra đạt kết quả cao hơn phương pháp Canny. Còn trong trường hợp nhiễu ít thì mức độ thành công của hai phương pháp này là xấp xỉ với nhau. Nếu đánh giá một cách tổng thể thì phương pháp ISEF được xếp thứ nhất do độ mảnh của các đường biên tỏ ra trội hơn. Còn phương pháp Canny xếp thứ hai.

Hai phương pháp cho kết quả khá chặt chẽ và gần nhau, trong trường hợp áp dụng trên ảnh thì sự sai lệch kết quả giwuax hai thuật toán lag không đáng kể, nhưng bằng trực giác không thể nhận ra những sai sót này.

Có thể nói kết quả thu được từ hai phương pháp này vượt trội hơn hẳn các phương pháp khác. Những thiếu sót trong các phương pháp như không nhận được đầy đủ cạnh, nhận ra nhiều cạnh trong khi chỉ có một cạnh là tồn tại.

Việc so sánh giữa Canny và ISEF phụ thuộc vào mỗi tham số được chọn cho mỗi trường hợp. Trong một số trường hợp thì Canny tỏ ra vượt trội hơn nhưng trong trường hợp khác thì ISEF lại hiệu quả hơn. Không thể có được một tập hợp tham số tốt nhất cho từng ảnh do vậy những phán quyết cuối cùng là vẫn dành cho người sử

dụng. Mặc dù cho kết quả cao nhưng hai phương pháp trên vẫn còn hạn chế tuy nhiên là không đáng kể.

KẾT LUẬN

Trên đây là toàn bộ quá trình nghiên cứu về " Các kỹ thuật phát hiện biên ảnh ". Do thời gian và trình độ còn hạn chế, đồng thời môn xử lý ảnh là một môn mới mẻ trong nghiên cứu khoa học và lần đầu tiếp xúc với ngôn ngữ lập trình Virtual C ++ chưa được bao lâu, nên còn nhiều phương pháp phát hiện biên chưa được nghiên cứu, do vậy chương trình vẫn còn nhiều thiếu sót.

Trong quá trình nghiên cứu làm đồ án tốt nghiệp em đã được thầy Ngô Quốc Tạo tận tình hướng dẫn và giúp đỡ, đồng thời cũng được các thầy, cô trong khoa CNTT trường ĐHDLHP và bạn bè giúp đỡ để em hoàn thành tốt đề tài của mình.

Tuy nhiên em cũng mong được sự góp ý của các thầy cô giáo, các bạn bè, để giúp em có được một chương trình hoàn thiện hơn.

CÀI ĐẶT CHƯƠNG TRÌNH NGUỒN

1. Phương pháp Gradient

```
//DAO HAM BAC 1 GRADIENT
// De giam thoi gian tinh toan ta dung theo chuan
//  A = |Gx(x,y) + Gy(x,y)|
//  Gx=I(x+1,y) - I(x,y)
// Gy=I(x,y+1) - I(x,y)
// Ta dung Hai mat na H1= { 0, 1, -1,  0}   H2= {-1, 0, 0, -1}.
void CBMPImageView::OnGradien1()
{
// TODO: Add your command handler code here
    CDC *hDC=GetDC();
    int i, j,k,l,t,rv,gv,bv;
    unsigned char r[2][2],g[2][2],b[2][2];
    ::SetCursor(::LoadCursor(NULL, IDC_WAIT));
    for(i=0;i<600;i++)
    for(j=0;j<400;j++)
     {
       rv=gv=bv=0;
       for(k=0;k<2;k++)
       for(l=0;l<2;l++)
         {
            t=hDC->GetPixel(i+k+4,j+l+4);
           b[k][l]=((0xff<<16)&t)>>16;
           g[k][l]=((0xff<<8)&t)>>8;
           r[k][l]=0xff&t;
         }
```

```
//Gradient
        rv=abs(r[2][1]+r[1][2]-2*r[1][1]);
        gv=abs(g[2][1]+g[1][2]-2*g[1][1]);
         bv=abs(b[2][1]+b[1][2]-2*b[1][1]);
        hDC->SetPixel(i,j, RGB(rv,gv,bv));
}
::SetCursor(::LoadCursor(NULL, IDC_ARROW));
}
```

2. Phương pháp Sobel

Đây là phương pháp rất hay dùng trong các bài toán tìm biên, phương pháp này dựa vào các mẫu xếp chồng còn gọi là mẫu xếp chồng Sobel.

Các mẫu xếp chồng là các mặt nạ đối xứng qua trục X hoặc trục Y. Ma trận này là kết quả của các phép tính toán đạo hàm ở trên (Gradient) trong chương trình này sử dụng các ma trận kích thước 3×3. Đó là áp dụng các hạt nhân vào ma trận điểm ảnh sau đó dùng các phép toán nhân chập ảnh và lấy ngưỡng để bớt nhiễu. Trong trường hợp này sẽ áp dụng cả hai hạt nhân trên bằng cách lấy trị tuyệt đối sự biến đổi độ sáng theo hai trục, sau đó lấy giá trị tuyệt đối abs(X) + abs(Y) cuối cùng lấy điểm ngưỡng sẽ thu được ảnh đầu ra mà có đường biên được xác định.

input: Pixels: day pixels

Pallett: bang mau

ColorType:loai mau xuli RED,GREEN,hay BLUE

ColorTableSize:kich thuoc bang mau

nWidth,nHeight:chieu rong va cao cua anh

N,M: kich thuoc cua so tim bien

output: Pallett:bang mau da bi thay doi

```
void Timbien::SOBEL(BYTE *Pixels,BYTE *Pallett, int ColorType,
 int ColorTableSize,int nWidth, int nHeight,int N,int M)
  {
    int WindowSize=N*M;
    int x,y,i,j,s,k,l;
    double R1,R2,R;
    double k1,k2;
     double t2[]={ -1, -2, -1,
                    0,  0,  0,
                    1,  2,  1};
      double t1[]={ -1,  0,  1,
                    -2,  0,  2,
                    -1,  0,  1};
//cap phat bo nho cho cua so
   w=(int*)new int[WindowSize*sizeof(int)];
//chua bang mau tam thoi
    buf=(BYTE*)new char[ColorTableSize*sizeof(BYTE)];
    for(i=0;i<ColorTableSize;i++)
    buf[i]=Pallett[i];
    int N2=N>>1;
    int M2=M>>1;
    for(y=N2;y<nHeight-N2;y++)
for(x=M2;x<nWidth-M2;x++)
{   s=0;
   R1=0;
   R2=0;
   for(j=-N2;j<=N2;j++)
```

```
      for(i=-M2;i<=M2;i++)
        {
           k=(y+j)*nWidth+x+i;   // cua so truot qua ma tran anh
           l=Pixels[k]*4+ColorType;
           R1+=t1[s]*buf[l];      // lay tong cac tich cua gia tri
           R2+=t2[s]*buf[l];     // trong cua so voi gia tri pixel
          s++;                    // trong vung cua so tuong ung
         }
              k1=abs((int)R1);
             k2=abs((int)R2);
   R=abs((int)R1)+abs((int)R2);
   R=(k1>k2)?k1:k2;
   k=y*nWidth+x;
   l=Pixels[k]*4+ColorType;
  pallett[l]=(BYTE)R;
 }
```

3. Phương pháp Laplace

input: Pixels: day pixels

Pallett: bang mau

ColorType: loai mau xuli RED,GREEN,hay BLUE

ColorTableSize:kich thuoc bang mau

nWidth,nHeight: chieu rong va cao cua anh

output: Pallett: bang mau da bi thay doi

```
void Timbien::Laplace2(BYTE *Pixels,BYTE *Pallett, int ColorType,int
ColorTableSize, int nWidth, int nHeight)
{
 //Laplace
```

```
    int N=3;
    int M=3;   // kich thuoc cua so
     int WindowSize=N*M;
     int x,y,i,j,s,k,l;
     int q;
    double ws[9]={  -1, -1, -1,
                    -1,  8, -1,
                    -1,-1,-1};
    double w[3][3]={  -1,-1,-1,
                      -1, 8,-1,
                      -1,-1,-1};
    double R;
//buf chua bang mau tam thoi
    buf=(BYTE*)new char[ColorTableSize*sizeof(BYTE)];
    for(i=0;i<ColorTableSize;i++)
   buf[i]=Pallett[i];
   int N2=N>>1;
   int M2=M>>1;
  for(y=N2;y<nHeight-N2;y++)
  for(x=M2;x<nWidth-M2;x++)
   {
       s=0;
       q=0;
       R=0;
       for(j=-N2;j<=N2;j++)
       for(i=-M2;i<=M2;i++)
         {   // cua so truot qua ma tran anh
```

```
            k=(y+j)*nWidth+x+i;
            l=Pixels[k]*4+ColorType;  // Tinh chi so bang mau
            R+=ws[s++]*buf[l];
//R+=(4*w[s,q]-w[s-1,q]-w[s+1,q]-w[s,q-1]-w[s,q+1])*buf[l];
 }
        k=y*nWidth+x;
        l=Pixels[k]*4+ColorType;
       Pallett[l]=(BYTE)R;
}
}
```

4. Phương pháp la bàn

```
void Timbien::Laban(BYTE *Pixels, BYTE *Pallett, int ColorType, int
ColorTableSize, int nWidth, int nHeight, int N, int M)
{
int WindowSize=N*M;
int x,y,i,j,s,k,l;
double R1,R2,R;
double k1,k2;
double t1[]={ 5, 5, 5,
                -3, 0,-3
                -3,-3,-3};
double t2[]={ 5, 5,-3,
                 5, 0,-3,
                -3,-3,-3};
```

```
double t3[]={5,-3,-3,
             5, 0,-3,
             5,-3,-3};
// cap phat bo nho cho cua so
    w=(int*)new int[WindowSize*sizeof(int)];
// chua bang mau tam thoi
 buf=(BYTE*)new char[ColorTableSize*sizeof(BYTE)];
for(i=0;i<ColorTableSize;i++)
buf[i]=Pallett[i];
int N2=N>>1;
int M2=M>>1;
for(y=N2;y<nHeight-N2;y++)
for(x=M2;x<nWidth-M2;x++)
  {  s=0;
     R1=0;
     R2=0;
     for(j=-N2;j<=N2;j++)
     for(i=-M2;i<=M2;i++)
       {// cua so truot qua ma tran anh
             k=(y+j)*nWidth+x+i;
              l=Pixels[k]*4+ColorType;
              R1+=t1[s]*buf[l];  // lay tong cac tich cua gia tri
              R2+=t2[s]*buf[l];   // trong cua so voi gia tri pixel
              s++;
        }
  k1=abs((int)R1);
  k2=abs((int)R2);
```

```
  R=t1[0]*buf[l];
  if(R1>R) R=R1;
  k=y*nWidth+x;
  l=Pixels[k]*4+ColorType;
  Pallett[l]=(BYTE)R;
}
}
```

5. Phương pháp hình chóp

```
void CBMPImageView::OnHinhchop()
{
        CDC *hDC=GetDC();
        int i, j,k,l,t,rv,gv,bv;
        unsigned char r[3][3],g[3][3],b[3][3];
        ::SetCursor(::LoadCursor(NULL, IDC_WAIT));
        for(i=0;i<600;i++)
                for(j=0;j<400;j++)
        {
                rv=gv=bv=0;
                for(k=0;k<3;k++)
                  for(l=0;l<3;l++)
                  {
                        t=hDC->GetPixel(i+k+4,j+l+4);//lay diem anh
                        b[k][l]=((0xff<<16)&t)>>16;
                        g[k][l]=((0xff<<8)&t)>>8;
                        r[k][l]=0xff&t;
                  }
```

```
                rv=r[1][1]/4+r[2][1]/4+r[1][2]/4+r[2][2]/4;
                gv=g[1][1]/4+g[2][1]/4+g[1][2]/4+g[2][2]/4;
                bv=r[1][1]/4+g[2][1]/4+g[1][2]/4+g[2][2]/4;
                hDC->SetPixel(i,j, RGB(rv,gv,bv));
                }
        ::SetCursor(::LoadCursor(NULL, IDC_ARROW));
}
```

6. Các hàm và thủ tục của phương pháp nâng cao

```
void CShenCastan::ISEF_Horiz(float **Src, float **Ret, int nRows, int nCols, float
b)
// Calculate ISEF in x direction
{
        int i, j;
        float b1, b2;
        float **y1=NULL,**y2=NULL,**Temp=NULL;
        COperation op;
        b1 = (float) (1-b)/(1+b);
        b2 = b * b1;
        y1 = op.f2D(nRows, nCols);
        Temp = op.f2D(nRows, nCols);
        y2 = op.f2D(nRows, nCols);
        for (i = 0; i < nRows; i++)
                for (j = 0; j < nCols; j++)
                        Temp[i][j] = Src[i][j];
        // Boundary Conditions
        for (i = 0; i < nRows; i++)
        {
```

```
            y1[i][0] = b1*Temp[i][0];
            y2[i][nCols-1] = b2*Temp[i][nCols-1];
        }
        // Apply the Filter
        for (j = 1; j < nCols; j++)
            for ( i = 0; i < nRows; i++)
            {
                y1[i][j] = b1*Temp[i][j] + b*y1[i][j-1];
            }
        for (j = nCols - 2; j >=0; j--)
            for (i = 0; i < nRows; i++)
            {
                y2[i][j] = b2*Temp[i][j] + b*y2[i][j+1];
            }
// Calculate the Result
        for ( i = 0; i < nRows; i++)
            Ret[i][nCols - 1] = y1[i][nCols-1];
        for ( i = 0; i < nRows; i++)
            for (j = 0; j < nCols - 1; j++)
                Ret[i][j] = y1[i][j] + y2[i][j+1];
        free(y1[0]);free(y1);
        free(y2[0]);free(y2);
        free(Temp[0]); free(Temp);
}
void CShenCastan::ISEF_Vert(float **Src, float **Ret, int nRows, int nCols, float
b)
//Calculate ISEF in y direction
```

```
{
        int i, j;
        float b1, b2;
        float **y1=NULL,**y2=NULL,**Temp=NULL;
        COperation op;
        b1 = (float) (1-b)/(1+b);
        b2 = b * b1;
        y1 = op.f2D(nRows, nCols);
        Temp = op.f2D(nRows, nCols);
        y2 = op.f2D(nRows, nCols);
        for (i = 0; i < nRows; i++)
                for (j = 0; j < nCols; j++)             Temp[i][j] = Src[i][j];
        //Boundary Conditions
        for (j = 0; j < nCols; j++)
        {
                y1[0][j] = b1*Temp[0][j];
                y2[nRows-1][j] = b2*Temp[nRows-1][j];
        }
        //Apply the Filter
        for (i = 1; i < nRows; i++)
                for (j = 0; j < nCols; j++)
                        y1[i][j] = b1*Temp[i][j] + b*y1[i-1][j];
        for (i = nRows-2; i >= 0; i--)
                for (j = 0; j < nCols; j++)
                        y2[i][j] = b2*Temp[i][j] + b*y2[i+1][j];
        //Compute the output
        for (j = 0; j < nCols; j++)
```

```
		Ret[nRows-1][j] = y1[nRows-1][j];
	for (i = 0; i < nRows-1; i ++)
		for (j = 0; j < nCols; j++)
			Ret[i][j] = y1[i][j] + y2[i+1][j];
	free(y1[0]);free(y1);
	free(y2[0]);free(y2);
	free(Temp[0]); free(Temp);
}
-----------------------------------------------------------------------------------------
void CShenCastan::Compute_BLI(float **Src, float **Smoothed, float **Ret, int
nRows, int nCols)
{
	int i,j;
	for (i = 0; i < nRows ; i++)
		for (j = 0; j < nCols; j ++)
		{
			if (Smoothed[i][j] - Src[i][j] > 0) Ret[i][j] = 1;
			else     Ret[i][j] =0;
		}
}
-----------------------------------------------------------------------------------------
void CShenCastan::ISEF(float **Src, float **Smoothed, int nRows, int nCols, float
b)
{
	COperation op;
	float **Temp = NULL;
	Temp = op.f2D(nRows,nCols);
```

```
        ISEF_Horiz(Src,Temp,nRows,nCols,b);
        ISEF_Vert(Temp, Smoothed, nRows, nCols, b);
        free(Temp[0]);
        free(Temp);
}
```

```
void CShenCastan::ShenAutoDetector(CDC &dcMem, float **Mag, float
**Zero_Cross, BITMAP bmp, float b)
{
        int nRows, nCols, i, j;
        COperation op;
        nCols = bmp.bmHeight;
        nRows = bmp.bmWidth;
        float **Src = op.f2D (nRows, nCols);
        //float **Zero_Cross = op.f2D (nRows, nCols);
        float **LastImg = op.f2D (nRows, nCols);
        // Convert ColorImage to Grey Level Image
        for ( i = 0; i < nRows  ; i ++)
                for ( j = 0; j < nCols ; j++)
                {
                        Src[i][j] = (float) ((   GetRValue(dcMem.GetPixel(CPoint(i, j)))
                                                                +
                                        GetGValue(dcMem.GetPixel(CPoint(i, j)))
                                                                +
                                        GetBValue(dcMem.GetPixel(CPoint(i, j))) )/3);
                }
        ISEF (Src,Mag,nRows,nCols,b);
```

```
	Compute_BLI (Src, Mag, Zero_Cross, nRows, nCols);
	for (i = 0; i < nRows; i++)
		for (j = 0; j< nCols; j++)
		{
			Mag[i][j] = Src[i][j];
			if ( (i == 0) || (j == 0) || (i == nRows - 1) || (j == nCols - 1) )
			{
				dcMem.SetPixel (CPoint(i,j),RGB(255,255,255));
				continue;
			}
				if (Is_Candidate_Edge(Zero_Cross,Mag,i,j)==true)
				dcMem.SetPixel (CPoint(i,j), RGB(0,0,0));
		else dcMem.SetPixel (CPoint(i,j),RGB(255,255,255));
		}
	free(Src[0]); free (Src);
//	free(Zero_Cross[0]); free (Zero_Cross);
	free(LastImg[0]); free (LastImg);
}
-----------------------------------------------------------------------------------------------
void CCanny::CannyAutoDetector(CDC &dcMem, float **Mag, float **NonMax,
BITMAP bmp, float sigma)
{
	int nRows, nCols, i, j;
	COperation op;
	nCols = bmp.bmHeight;
	nRows = bmp.bmWidth;
	float **Src = op.f2D (nRows, nCols);
```

```
        float **LastImg = op.f2D (nRows, nCols);
//      float **NonMax= op.f2D (nRows, nCols);
// Convert ColorImage to Grey Level Image
        for ( i = 0; i < nRows  ; i ++)
                for ( j = 0; j < nCols ; j++)
                {
                Src[i][j] = (float) ((  GetRValue(dcMem.GetPixel(CPoint(i, j)))
                                                                +
                                GetGValue(dcMem.GetPixel(CPoint(i, j)))
                                                                +
                                GetBValue(dcMem.GetPixel(CPoint(i, j))) )/3);
                }
// Calculate the Magnitude Image and save this to Mag array
// NonMax is like a Zero-Cross Image, with edge pixels have value 1, else value 0
        Magnitude (Src, Mag, NonMax, nRows, nCols, sigma);
        op.Auto_Threshold (Src,NonMax,nRows,nCols,LastImg);
// Convert Last Img to CDC
        for ( i = 0; i < nRows; i ++)
                for ( j = 0; j < nCols; j++)
                {
                        Mag[i][j] = Src[i][j];
                        if (LastImg[i][j] == 0)
dcMem.SetPixel(CPoint(i,j),RGB(255,255,255));
                        else
                        {
                                if (LastImg[i][j] == 255)
dcMem.SetPixel(CPoint(i,j),RGB(0,0,0));
```

```
                }
            }
        free(Src[0]); free(Src);
        free(LastImg[0]); free(LastImg);
//      free(NonMax[0]); free(NonMax);
}
-----------------------------------------------------------------------------------------------
void CMarr::MarrAutoDetector(CDC &dcMem, float **Mag, float **Zero_Cross,
BITMAP bmp, float sigma)
{
        int nRows, nCols, i, j;
        COperation op;
        nCols = bmp.bmHeight;
        nRows = bmp.bmWidth;
        float **Src = op.f2D (nRows, nCols);
//      float **Zero_Cross = op.f2D (nRows, nCols);
        float **LastImg = op.f2D (nRows, nCols);
// Convert ColorImage to Grey Level Image
        for ( i = 0; i < nRows  ; i ++)
            for ( j = 0; j < nCols ; j++)
            {
                Src[i][j] = (float) ((   GetRValue(dcMem.GetPixel(CPoint(i, j)))
                                                    +
                                    GetGValue(dcMem.GetPixel(CPoint(i, j)))
                                                    +
                                    GetBValue(dcMem.GetPixel(CPoint(i, j))) )/3);
            }
```

```
// Calculate the Gradient Image and save this to Mag array
        Magnitude (Src, Mag, nRows, nCols, sigma);
// Points that have value 1 in Zero_Cross is Zero-crossing points
//      Zero_Crossing(Mag, Zero_Cross, nRows, nCols, sigma);
        Zero_Crossing(Mag, LastImg, nRows, nCols, sigma);
// LastImg is the Image which Points have value 255 is Edge, else have value 0
//      op.Auto_Threshold(Src, Zero_Cross, nRows, nCols, LastImg);
// Convert Last Img to CDC
        for ( i = 0; i < nRows; i ++)
                for ( j = 0; j < nCols; j++)
                {
                        Mag[i][j] = Src[i][j];
                        if (LastImg[i][j] == 0)
dcMem.SetPixel(CPoint(i,j),RGB(255,255,255));
                        else
                        {
                                if (LastImg[i][j] == 1)
dcMem.SetPixel(CPoint(i,j),RGB(0,0,0));
                        }
                }
        free(Src[0]); free (Src);
        free(LastImg[0]); free (LastImg);
}
```

TÀI LIỆU THAM KHẢO

1. Nhập môn xử lý ảnh số: Lương Mạnh Bá – Nguyễn Thanh Thủy.

Nhà xuất bản khoa học kỹ thuật 2000

2. Kỹ thuật xử lý ảnh và video: Nguyễn Kim Sách

Nhà xuất bản khoa học kỹ thuật 1999

3. Giáo trình xử lý ảnh: Phạm Việt Bình - Đỗ Năng Toàn

4. Lập trình VC với MFC

5. Internet