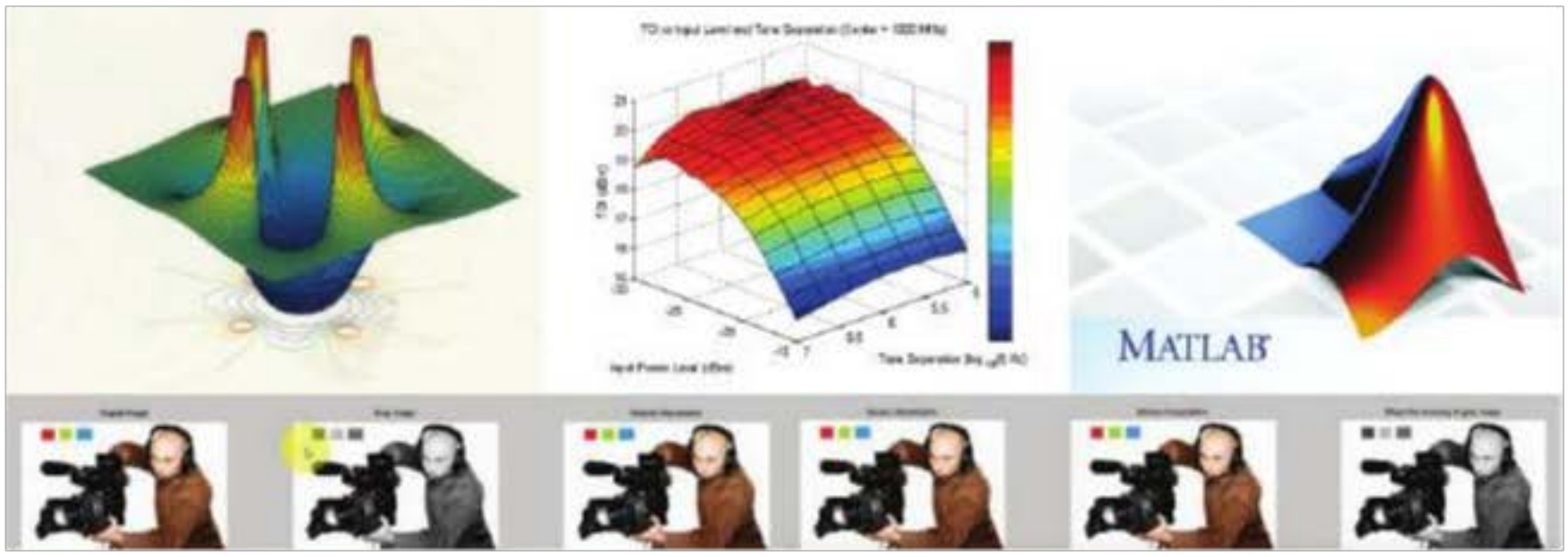
BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT THÀNH PHỐ HỒ CHÍ MINH

ISO 9001:2000

TS. NGUYỄN THANH HẢI



GIÁO TRÌNH

XỬ LÝ ẢNH

(Giáo trình dành cho sinh viên ngành Điện - Điện tử)

NHÀ XUẤT BẢN
ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH

https://tieulun.hopto.org

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT
THÀNH PHỐ HỒ CHÍ MINH

----------**----------

TS. NGUYỄN THANH HẢI

GIÁO TRÌNH
XỬ LÝ ẢNH

(Ngành Điện - Điện Tử)

NHÀ XUẤT BẢN ĐẠI HỌC QUỐC GIA
THÀNH PHỐ HỒ CHÍ MINH – 2014

https://tieulun.hopto.org

GIÁO TRÌNH
XỬ LÝ ẢNH

Nhà xuất bản ĐHQG-HCM và tác giả/đối tác liên kết giữ bản quyền ©

Copyright © by VNU-HCM Publishing House and author/co-partnership

All rights reserved

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TPHCM

Xuất bản năm 2014

https://tieulun.hopto.org

LỜI NÓI ĐẦU

Sự ra đời của xử lý ảnh và những ứng dụng của nó là rất cần thiết cho cuộc sống. Xử lý ảnh đã có từ rất lâu và đã được vận dụng trong những lĩnh vực như dân sự, quân sự, y tế và nhiều lĩnh vực khác. Tuy nhiên, mới chỉ khoảng một thập niên trở lại đây, kiến thức xử lý ảnh mới được đưa vào để giảng dạy tại một vài trường đại học tại Việt Nam nhưng cũng chỉ hạn chế ở cấp bậc cao học, còn bậc đại học thì cũng chỉ mới có ở một vài trường lớn tại Thành phố Hồ Chí Minh và Hà Nội. Tuy nhiên, với việc phát triển rất nhanh của xã hội, nhu cầu áp dụng công nghệ xử lý ảnh vào cuộc sống ngày càng lớn, do vậy, những năm gần đây, bậc đại học tại các trường cũng đưa chương trình xử lý ảnh vào để dạy cho sinh viên. Hiện nay, có nhiều đề tài liên quan đến xử lý ảnh được các sinh viên đại học, học viên cao học và các nghiên cứu sinh nghiên cứu và thực hiện.

Nhằm phục vụ cho việc giảng dạy những kiến thức cơ bản cũng như một số gợi ý về ứng dụng xử lý ảnh, giáo trình *Xử lý ảnh* được biên soạn chủ yếu là xử lý ảnh số với những ví dụ được viết bằng Matlab ra đời. Giáo trình này nhằm mục đích giúp sinh viên đại học và học viên cao học có những kiến thức cơ bản về xử lý ảnh cũng như áp dụng kết quả sau khi xử lý ảnh cho những ứng dụng cụ thể nhằm phát triển những ứng dụng thực tế.

Giáo trình này gồm những phần như sau:

Chương 1: giới thiệu về xử lý ảnh cũng như môi trường Matlab để xử lý cho ra những kết quả mong muốn.

Chương 2: trình bày những kiến thức cơ bản về ảnh và những toán tử liên quan cho xử lý ảnh số.

Chương 3: giới thiệu về những phép biến đổi ảnh như biến đổi Fourier và biến đổi Wavelet.

Chương 4: trình bày kỹ thuật lọc ảnh trong miền tần số và không gian.

Chương 5: giới thiệu phép tăng cường ảnh vận dụng phép xử lý điểm và xử lý mặt nạ.

Chương 6: trình bày kỹ thuật phân đoạn ảnh và dò biên.

Chương 7: trình bày một vài thuật toán liên quan đến trích đặc trưng và nhận dạng.

https://tieulun.hopto.org

Mặc dù tác giả đã cố gắng viết và chỉnh sửa nhưng chắc không thể tránh những thiếu sót, rất mong nhận được những đóng góp quý báu từ người đọc và các chuyên gia trong lĩnh vực xử lý ảnh.

Địa chỉ liên lạc:

Khoa Điện - Điện tử, Đại học Sư phạm Kỹ thuật TP.HCM

Email: nthai@hcmute.edu.vn

https://tieulun.hopto.org

MỤC LỤC

MỤC LỤC 5
LIỆT KÊ THUẬT NGỮ VIẾT TẮT 8

CHƯƠNG 1: GIỚI THIỆU 9
1.1. HỆ THỐNG XỬ LÝ ẢNH 9
1.2. XỬ LÝ ẢNH SỐ DÙNG MATLAB 13
1.2.1. Môi trường làm việc trong MATLAB 13
1.2.2. MATLAB Editor 14
1.2.3. Trợ giúp trong MATLAB 14
1.2.4. Thực hành xử lý ảnh cơ bản với MATLAB 15
BÀI TẬP CHƯƠNG 1 23

CHƯƠNG 2: KHÁI NIỆM CƠ BẢN 24
2.1. ĐIỂM ẢNH VÀ ẢNH SỐ 24
2.1.1. Điểm ảnh 24
2.1.2. Phân loại ảnh 25
2.1.3. Độ phân giải 28
2.1.4. Điểm ảnh lân cận 28
2.1.5. Sự liền kề, kết nối, vùng và đường bao 29
2.1.6. Đo lường khoảng cách 31
2.2. TOÁN TỬ ẢNH 32
2.2.1. Biến đổi âm bản 34
2.2.2. Biến đổi hàm log 34
2.2.3. Biến đổi dạng lũy thừa 36
BÀI TẬP CHƯƠNG 2 39

CHƯƠNG 3: PHÉP BIẾN ĐỔI ẢNH 40
3.1. BIẾN ĐỔI FOURIER RỜI RẠC 40
3.1.1. Biến đổi Fourier rời rạc 42
3.1.2. Tính toán và biểu diễn biến đổi 2-DFT trong MATLAB 44
3.2. BIẾN ĐỔI WAVELET 49

https://tieulun.hopto.org

3.2.1. Biến đổi Fourier thời gian ngắn 49
3.2.2. Phân tích Wavelet 52
3.2.2.1. Phân tích đa phân giải 52
3.2.2.2. Wavelet 53
3.2.2.3. Biến đổi Wavelet liên tục 54
3.2.2.4. Biến đổi Wavelet rời rạc 56
3.2.2.4.1. Giàn lọc 56
3.2.2.4.2. Kỹ thuật tăng và giảm mẫu 57
3.2.2.4.3. Tái cấu trúc tín hiệu 58
3.2.2.4.4. Các giàn lọc đa phân giải 59
3.2.2.4.5. Một số hàm Wavelet thông dụng 64
3.2.2.5. Biến đổi Wavelet trong ảnh số 70
3.2.2.6. Biến đổi Wavelet dùng Wavelet Toolbox 72
3.2.2.7. Một số ứng dụng của phép biến đổi Wavelet 80
BÀI TẬP CHƯƠNG 3 83

CHƯƠNG 4: LỌC ẢNH 84
4.1. LỌC TRONG MIỀN KHÔNG GIAN 84
4.2. LỌC TRONG MIỀN TẦN SỐ 92
4.2.1. Lọc thông thấp 96
4.2.2. Lọc thông cao 101
BÀI TẬP CHƯƠNG 4 104

CHƯƠNG 5: TĂNG CƯỜNG ẢNH 105
5.1. XỬ LÝ ĐIỂM 105
5.1.1. Mở rộng độ tương phản 106
5.1.2. Xử lý lược đồ (Histogram) 109
5.1.2.1. Cân bằng Histogram 111
5.1.2.2. Kỹ thuật phối hợp lược đồ (Histogram matching) 117
5.2. XỬ LÝ MẶT NẠ 121
5.2.1. Các bộ lọc làm mịn ảnh 121
5.2.2. Các bộ lọc làm sắc nét ảnh 126
BÀI TẬP CHƯƠNG 5 131

https://tieulun.hopto.org

CHƯƠNG 6: PHÂN ĐOẠN VÀ TÁCH BIÊN 132
6.1. PHÂN ĐOẠN 132
6.1.1. Phương pháp phân đoạn biên độ chói 132
6.1.2. Phương pháp phân đoạn nhóm 135
6.1.3 Phương pháp phân đoạn vùng 137
6.1.3.1. Công thức cơ bản 137
6.1.3.2. Phân đoạn phát triển vùng 138
6.1.3.3. Tách vùng và ghép vùng 139
6.2. TÁCH BIÊN 141
6.2.1. Tách biên theo đạo hàm bậc một 144
6.2.2. Phát hiện biên theo đạo hàm bậc hai 149
6.2.2.1. Phương pháp Laplace 150
6.2.2.2. Bộ tách biên LoG (Laplacian of Gaussian Detector) 151
6.2.3. Bộ tách biên Canny 152
BÀI TẬP CHƯƠNG 6 153

CHƯƠNG 7: TRÍCH ĐẶC TRƯNG VÀ NHẬN DẠNG 154
7.1. PHÂN TÍCH THÀNH PHẦN CHÍNH (PCA) 154
7.1.1. Giới thiệu 154
7.1.2. Thuật toán PCA 155
7.1.2.1. Chuyển đổi ảnh 155
7.1.2.2. Ảnh trung bình 157
7.1.2.3. Sai biệt ảnh trung bình 157
7.1.2.4. Ma trận hiệp phương sai 158
7.1.2.5. Phép chiếu 159
7.1.2.6. Nhận dạng 160
7.2. MÁY VECTOR HỖ TRỢ (SVM) 160
7.2.1. Siêu phẳng - Hyperplane 160
7.2.2. SVM tuyến tính 161
7.2.3. Các điều kiện Karush-Kuhn-Tucker (KKT) 163
7.2.4. Giải thuật SMO 163
7.3. MẠNG NƠ-RON NHÂN TẠO 166
BÀI TẬP CHƯƠNG 7 170

https://tieulun.hopto.org

LIỆT KÊ THUẬT NGỮ VIẾT TẮT

ALU	Arithmetic Logic Unit
MATLAB	MATrix LABoratory
CRT	Cathode Ray Tube
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
STFT	Short Time Fourier Transform
WT	Wavelet Transform
MRA	MultiResolution Analysis
CWT	Continuous Wavelet Transform
DWT	Discrete Wavelet Transform
FIR	Finite Impulse Response
ILPF	Ideal Lowpass Filter
BLPF	Butterworth Lowpass Filter
GLPF	Gaussian Lowpass Filter
PSNR	Peak Signal to Noise Ratio
PDF	Probability Density Function
CDF	Cumulative Distribution Function
LoG	Laplacian of Gaussian
PCA	Principle Component Analysis
ANN	Artificial Neural Network
SVM	Support Vector Machine
KKT	Karush-Kuhn-Tucker
SMO	Sequential Minimal Optimisation

https://tieulun.hopto.org

Chương 1
GIỚI THIỆU

1.1. HỆ THỐNG XỬ LÝ ẢNH

Trong những năm gần đây, mặc dù còn rất mới mẻ trong lĩnh vực khoa học và công nghệ nhưng xử lý ảnh đang được nghiên cứu và phát triển với tốc độ nhanh chóng bởi các trung tâm nghiên cứu, trường đại học và viện… với rất nhiều ứng dụng trên các lĩnh vực khác nhau. Xử lý ảnh chỉ mới được đưa vào giảng dạy ở bậc đại học Việt Nam trong những năm gần đây dành cho một số ngành học nhất định. Đây là môn học liên quan đến nhiều lĩnh vực và cần nhiều kiến thức cơ sở khác: đại số tuyến tính, xác suất thống kê, xử lý số tín hiệu, trí tuệ nhân tạo…cũng được đề cập và phân tích trong nội dung.

Các phương pháp xử lý ảnh bắt đầu từ các ứng dụng nhằm nâng cao chất lượng bằng các phương pháp phân tích được nghiên cứu trong giai đoạn thiết bị phần cứng bị hạn chế, chẳng hạn như nâng cao độ sáng hay độ phân giải của hình ảnh…Về sau, sự xuất hiện và phát triển mạnh của máy tính đã tạo điều kiện hơn nữa cho quá trình thực hiện các thuật toán xử lý ảnh. Ứng dụng của xử lý ảnh ngày càng được mở rộng sang các lĩnh vực khác, chẳng hạn như viễn thông, điều khiển tự động, giao thông thông minh, kỹ thuật y sinh… Đặc biệt, trong các thiết bị kỹ thuật số có màn hình hiển thị luôn ưu tiên một phần tài nguyên dành cho việc xử lý hình ảnh: máy tính xách tay, camera kỹ thuật số, điện thoại thông minh, tivi thông minh…

Thông thường, xử lý ảnh số bao gồm nhiều vấn đề khác nhau, tuy nhiên, có thể chia thành bốn lĩnh vực chính như hình 1.1: tạo lập ảnh, biểu diễn, phân tích và quản lý. Bên cạnh đó, các thuật toán tăng cường ảnh có thể được xem xét như là một bước tiền hay hậu xử lý trong tất cả các lĩnh vực trên.

- Tạo lập ảnh: bao gồm tất cả các bước từ quá trình thu nhận ảnh đến hình thành một ảnh số dưới dạng ma trận.

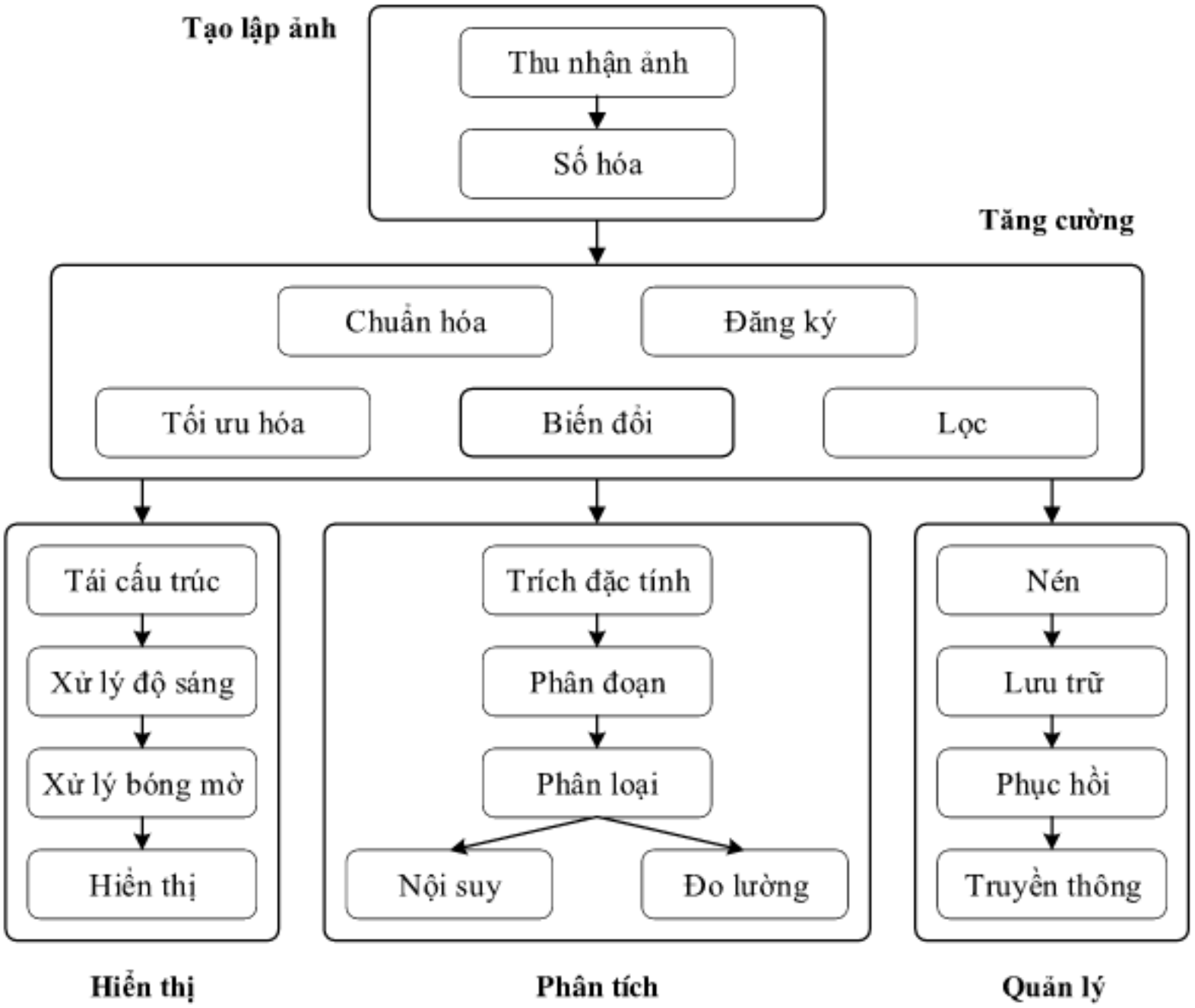
- Hiển thị: tất cả các dạng xử lý trên ma trận để ngõ ra là tối ưu.

- Phân tích: gồm tất cả các bước xử lý được dùng cho quá trình đo lường định lượng. Những bước này yêu cầu trước hết là hiểu biết bản chất và nội dung ảnh, do đó, các tiến trình phân tích ảnh rất rõ ràng và các thuật toán phát triển có thể được chuyển đổi nhanh chóng sang một môi trường ứng dụng khác một cách trực tiếp.

https://tieulun.hopto.org

• Quản lý: các kỹ thuật cung cấp khả năng lưu trữ, truyền thông, phục hồi và truy nhập hết sức hiệu quả. Trong y khoa, việc lấy thông tin hình ảnh từ xa cũng được xem là một phần của lĩnh vực quản lý.

Ngược với phân tích ảnh, thường được nghĩ là xử lý ảnh nâng cao, xử lý ảnh cấp thấp thường đề cập đến các kỹ thuật thông thường hay tự động, nghĩa là hoàn toàn có thể nhận biết mà không cần trang bị nhiều kiến thức chuyên sâu về nội dung ảnh. Các thuật toán dạng này có ảnh hưởng không đáng kể đến nội dung của ảnh. Ví dụ, sự mở rộng lược đồ của các tấm phim chụp X-quang giúp tăng cường độ tương phản tương tự như với các tấm ảnh đời thường. Do đó, các phương pháp xử lý ảnh thường có giá trị như các chương trình dành cho tăng cường ảnh.



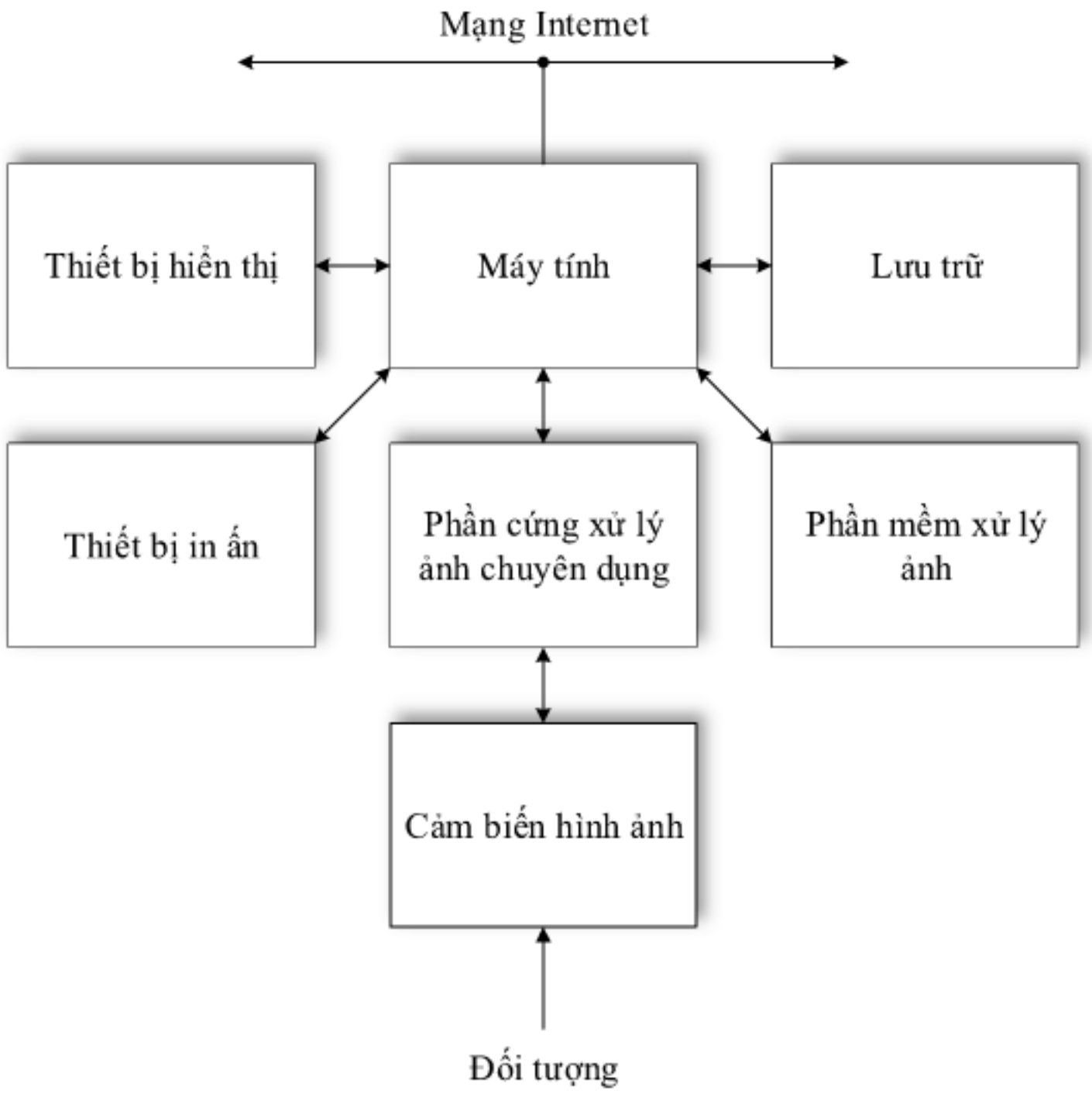
Hình 1.1. *Các lĩnh vực nghiên cứu chính trong xử lý ảnh số*

Các thành phần trong một hệ thống xử lý ảnh

Thông thường, một hệ thống xử lý ảnh có thể được xem như một hệ thống đa năng gồm các thành phần cơ bản như: cảm biến hình ảnh, phần cứng xử lý ảnh chuyên dụng, máy tính, phần mềm xử lý, bộ nhớ, màn hình hiển thị, in ấn và mạng như hình 1.2.

https://tieulun.hopto.org

• *Cảm biến*: hai yếu tố được yêu cầu để tạo lập một tấm ảnh số: đầu tiên là thiết bị vật lý nhạy với năng lượng bức xạ bởi vật thể mà chúng ta mong muốn ghi lại, thứ hai là bộ số hóa, hay còn gọi là thiết bị chuyển đổi ngõ ra của thiết bị cảm biến thành dạng số. Ví dụ, trong các camera số, các cảm biến xuất ngõ ra dạng tín hiệu điện tương ứng với cường độ sáng. Bộ số hóa sẽ chuyển đổi các ngõ ra này sang dạng dữ liệu số.



Hình 1.2. *Các thành phần cơ bản trong hệ thống xử lý ảnh đa dụng*

• *Phần cứng xử lý ảnh chuyên dụng* thường bao gồm các bộ số hóa đã đề cập và phần cứng xử lý các thông số ban đầu khác, chẳng hạn như đơn vị số học (ALU) – chuyên thực hiện các phép toán logic song song với nhiều ngõ vào. Một ví dụ cho thấy cách ALU được dùng để tính trung bình các ảnh nhằm mục đích triệt nhiễu. Dạng phần cứng này cũng thường được gọi là hệ thống phụ trợ đầu cuối, và đặc trưng phân biệt là tốc độ. Hay nói cách khác, các đơn vị thực hiện các chức năng yêu cầu băng thông dữ liệu lớn (số hóa và lấy trung bình các ảnh trong video thường là 30 khung hình/giây) mà các máy tính dùng cho chức năng này không thể điều khiển bằng tay như thông thường.

https://tieulun.hopto.org

- *Máy tính* trong hệ thống xử lý ảnh là các máy tính đa chức năng và chúng có thể là máy tính cá nhân hay thậm chí là siêu máy tính. Trong các ứng dụng cụ thể, các máy tính dân dụng có thể được dùng để đạt được hiệu năng cần thiết, nhưng điều chúng ta quan tâm ở đây là các hệ thống xử lý ảnh đa dụng. Trong những hệ thống này, phần lớn các thiết bị dạng máy tính cá nhân đều có thể phù hợp cho các nhiệm vụ xử lý ảnh ngoại tuyến (off-line).

- *Phần mềm* dành cho xử lý ảnh bao gồm các phần chuyên thực hiện các nhiệm vụ đặc trưng. Các gói này cũng bao gồm các đoạn mã dựng sẵn nhằm giúp cho người dùng có thể tạo ra các module riêng. Nhiều gói phần mềm phức tạp cho phép tích hợp những module này và các câu lệnh trong các phần mềm đa dụng từ ít nhất một ngôn ngữ máy tính.

- *Khả năng lưu trữ* là vấn đề cần quan tâm trong các ứng dụng xử lý ảnh. Một ảnh với kích thước 1024×1024 điểm ảnh, mà trong đó, mỗi điểm ảnh yêu cầu mã hóa 8 bit, thì có thể thấy dung lượng cần thiết cho việc lưu trữ là một megabytes trong trường hợp ảnh chưa được nén. Vì thế, với hàng triệu tấm ảnh thì việc cung cấp không gian lưu trữ trở thành một thách thức lớn. Các kiểu lưu trữ thường rơi vào một trong ba dạng cơ bản sau: lưu trữ tạm thời trong thời gian xử lý, lưu trữ trực tuyến với khả năng trích xuất nhanh chóng và lưu trữ dùng cơ sở dữ liệu.

- *Thiết bị hiển thị* thường được dùng hiện nay là màn hình tivi (dạng phẳng). Các màn hình được điều khiển bởi các ngõ ra hình ảnh và bo mạch hiển thị đồ họa mà xét về cơ bản thì chúng là một phần trong hệ thống máy tính. Trong các trường hợp khác, thiết bị hiển thị còn có khả năng hiển thị dạng lập thể theo không gian ba chiều.

- *Các thiết bị in ấn* dành cho việc ghi lại các hình ảnh bao gồm máy in laser, máy chụp dạng phim, các thiết bị cảm biến nhiệt, máy in phun, hay các thiết bị số như đĩa quang hay đĩa nén. Trong tất cả các dạng kể trên, phim cung cấp độ phân giải cao nhất, nhưng giấy được xem như môi trường thông dụng nhất. Để hiển thị, hình ảnh được đưa lên các tấm phim trong suốt hay trong môi trường số hóa nếu sử dụng các máy chiếu dạng phim.

- *Mạng* gần như là chức năng mặc định trong bất kỳ hệ thống máy tính hiện nay. Do một lượng lớn dữ liệu thuộc các ứng dụng xử lý ảnh, vấn đề cần xem xét ở đây chính là băng thông. Với một mạng dành riêng, nó không thật sự là vấn đề, nhưng truyền thông với các trang điều khiển qua Internet thì không phải lúc nào cũng hiệu quả. May mắn thay, trường hợp này đang được nâng cấp bằng cáp quang và các công nghệ băng rộng khác.

https://tieulun.hopto.org

1.2. XỬ LÝ ẢNH SỐ DÙNG MATLAB

MATLAB (MATrix LABoratory), một ngôn ngữ mở, từ lâu đã trở thành ngôn ngữ lập trình dành riêng cho các công việc yêu cầu tính toán nhiều với khả năng tính toán, hiển thị, lập trình trong môi trường rất dễ sử dụng. Có thể xét một số chức năng chính của MATLAB như:

- Toán học và các phép tính
- Phát triển thuật toán
- Thu thập dữ liệu
- Mô hình hóa, mô phỏng và tạo bản mẫu
- Phân tích dữ liệu, khảo sát và hiển thị
- Đồ họa khoa học và kỹ thuật
- Phát triển ứng dụng, bao gồm các giao diện người dùng

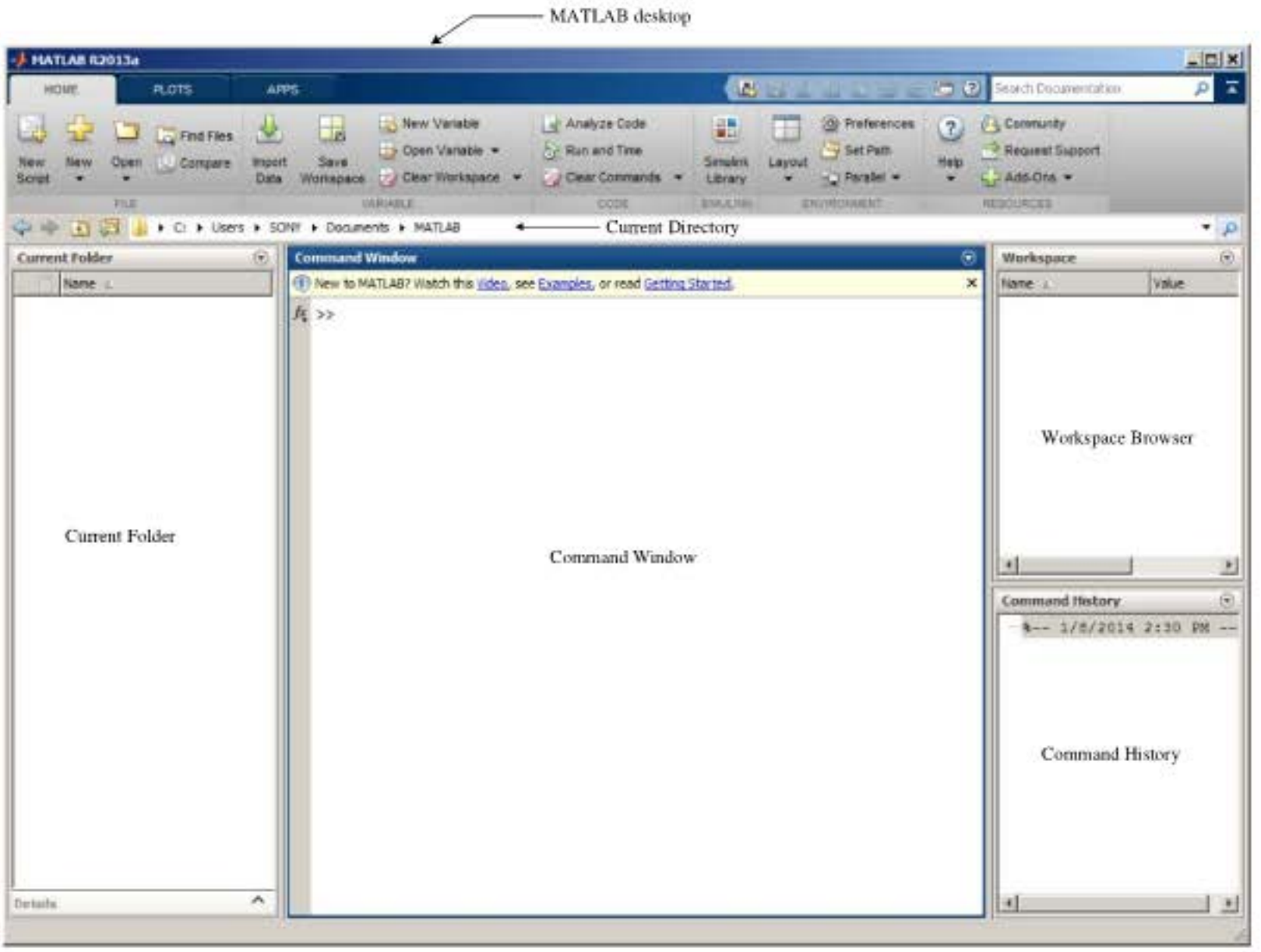
MATLAB là một hệ thống tương tác với các thành phần dữ liệu cơ bản là các mảng không yêu cầu phải định kích thước. Điều này cho phép trình bày các giải pháp dành cho nhiều vấn đề tính toán trong kỹ thuật, đặc biệt là bao gồm việc biểu diễn ma trận, thường thì phải chia nhỏ thời gian thực hiện đối với các chương trình không hỗ trợ tương tác như C hay Fortran.

Trong môi trường đại học, MATLAB được xem như công cụ tính toán tiêu chuẩn cho các khóa học cơ bản cũng như nâng cao về toán học, kỹ thuật và khoa học. Hơn nữa, MATLAB còn là công cụ tính toán dành cho các nghiên cứu, phân tích và phát triển. MATLAB thường được bổ sung bằng các hộp công cụ (toolbox) nhằm giải quyết các vấn đề theo hướng ứng dụng, trong đó có Image Processing Toolbox. Đây là tập các hàm MATLAB (hay còn gọi là file M) được biên soạn dành cho việc giải quyết từng vấn đề cụ thể liên quan đến xử lý ảnh. Các toolbox khác đôi khi được dùng hỗ trợ, chẳng hạn như Signal Processing, Neural Network, Fuzzy Logic hay Wavelet Toolbox.

1.2.1. Môi trường làm việc trong MATLAB

Cửa sổ làm việc chính trong MATLAB luôn là MATLAB desktop như hình 1.3, với các cửa sổ nhỏ bao gồm: the Command Window, the Workspace Browser, the Current Directory Window, the Command History, the Current Folder và nhiều cửa sổ hiển thị hình ảnh là Figure Windows (chỉ cho phép hiển thị hình ảnh và đồ thị).

https://tieulun.hopto.org



Hình 1.3. Cửa sổ làm việc trong MATLAB và các thành phần cơ bản

1.2.2. MATLAB Editor

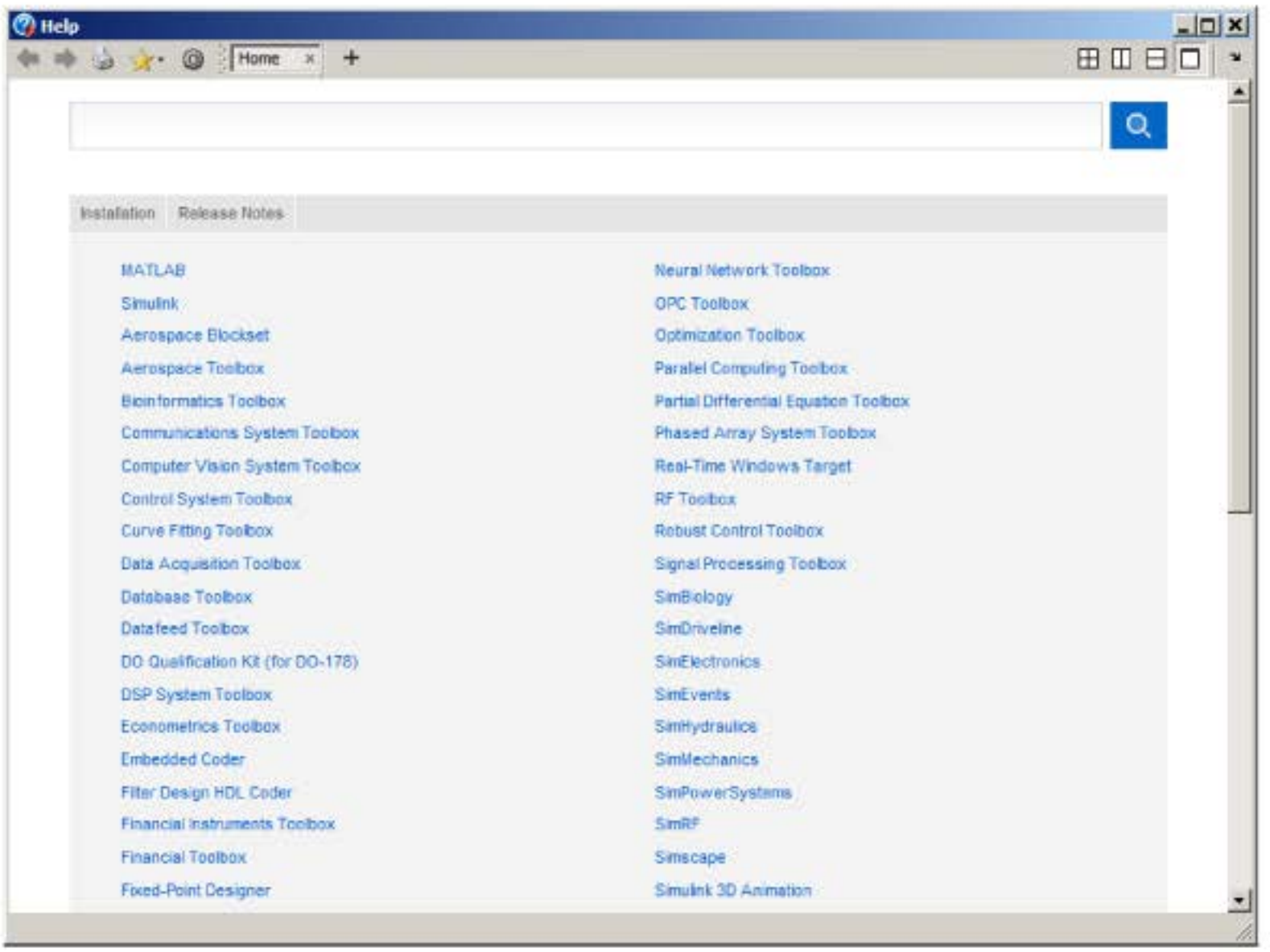
MATLAB Editor bao gồm phần biên soạn dưới dạng text để tạo M-file và bộ debugger – gỡ rối dạng đồ họa. Phần biên soạn này có thể xuất hiện trong cửa sổ của chính nó hay dưới dạng một cửa sổ con bên trong desktop. Với các M-file thường được đánh dấu bởi phần đuôi `.m`, chẳng hạn như `baitap.m`. Cửa sổ soạn thảo MATLAB có nhiều menu với các chức năng khác nhau như lưu, xem trước, hay gỡ rối file. Do nó thực hiện một vài kiểm tra đơn giản và thường dùng các màu khác nhau giúp phân biệt các thành phần của đoạn mã nên bộ biên soạn này được đề xuất như là công cụ cho tạo mới và chỉnh sửa các hàm trong MATLAB. Để mở nó, gõ lệnh `edit` ngay tại cửa sổ Command Window. Tương tự, có thể gõ `edit filename` để mở tập tin đã có sẵn bên cửa sổ Current Directory.

1.2.3. Trợ giúp trong MATLAB

Phương pháp truyền thống để truy cập vào phần Help của MATLAB mà với từng cửa sổ riêng lẻ hoặc bằng cách nhấp vào biểu tượng dấu chấm hỏi (?) trên thanh công cụ hoặc gõ `helpbrowser` ngay bên trong cửa sổ Command Window. Một cách đơn giản và nhanh chóng khác là nhấn phím F1 khi ở bất kỳ cửa cổ nào của MATLAB. Phần Help

https://tieulun.hopto.org

gồm hai phần nhỏ: ô dành cho tìm kiếm thông tin và ô dành hiển thị thông tin tìm được như hình 1.4.



Hình 1.4. *Giao diện cửa sổ Help của MATLAB*

Bên trong giao diện cửa sổ Help, người dùng có thể tùy chọn việc tìm kiếm theo hai cách: (1) có thể gõ vào nội dung cần tham khảo ô tìm kiếm, (2) tìm theo từng Toolbox với các chủ đề biết trước (thường được sử dụng như là một phương pháp tự học).

1.2.4. Thực hành xử lý ảnh cơ bản với MATLAB

Ví dụ 1.1: *Đọc, hiển thị và lấy thông tin ảnh.*

```
f=imread('peppers.png');
Info_f=imfinfo('peppers.png');
```

Ngõ ra của hàm `imfinfo` cho phép truy vấn thông tin của ảnh. Với ảnh `peppers.png`, kết quả hiển thị tại cửa sổ Command Window:

```
Info_I =
            Filename: [1x65 char]
         FileModDate: [1x20 char]
            FileSize: 287677
              Format: 'png'
       FormatVersion: []
```


https://tieulun.hopto.org

```
                    Width: 512
                   Height: 384
                 BitDepth: 24
                ColorType: [1x9 char]
FormatSignature: [1x8 double]
                 Colormap: []
                Histogram: []
InterlaceType: 'none'
              Transparency: 'none'
    SimpleTransparencyData: []
           BackgroundColor: []
           RenderingIntent: []
            Chromaticities: []
                     Gamma: []
               XResolution: []
               YResolution: []
            ResolutionUnit: []
                   XOffset: []
                   YOffset: []
                OffsetUnit: []
          SignificantBits: []
              ImageModTime: [1x26 char]
                     Title: []
                    Author: []
               Description: [1x13 char]
                 Copyright: [1x29 char]
              CreationTime: []
                  Software: []
                Disclaimer: []
                   Warning: []
                    Source: []
                   Comment: []
                 OtherText: []
```

Hàm `imread` cho phép đọc được nhiều định dạng ảnh khác nhau trong máy tính (tham khảo bảng 1.1) với tên file phải bao gồm cả phần mở rộng được đặt trong dấu nháy đơn ('). Ảnh sau khi đọc sẽ được lưu dưới dạng ma trận có tên `f` (người dùng hoàn toàn có thể dùng biến khác

https://tieulun.hopto.org

nhau miễn không trùng lắp với những biến đã có trước đó). Hàm `imread` có thể đọc một file ảnh với đường dẫn cụ thể.

```
imread('D:\Baitap\baitap.png);
```

Bảng 1.1. *Một số định dạng ảnh được hỗ trợ bởi hàm* `imread`.

Tên định dạng	Mô tả	Phần mở rộng
TIFF	Tagged Image File Format	.tif,.tiff
JPEG	Joint Photographic Experts Group	.jpg,.jpeg
JPEG 2000	Joint Photographic Experts Group	.jpg,.jpeg
GIF	Graphics Interchange Format	.gif
BMP	Windows Bitmap	.bmp
PNG	Portable Network Graphics	.png
XWD	X Window Dump	.xwd
HDF4	Hierarchical Data Format	.hdf
ICO	Icon File	.ico
PCX	Windows Paintbrush	.pcx
PGM	Portable Graymap	.pgm
PPM	Portable Pixmap	.ppm
RAS	Sun Raster	.ras

❖ **Lưu ý:**

- Nên sử dụng hàm `clear all` để xóa các tham số trong cửa sổ Workspace khi bắt đầu một chương trình mới để tránh việc nhầm lẫn với tham số của chương trình trước đó.
- Kết quả của hàm `imfinfo` là một cấu trúc có nhiều trường, vì thế có thể truy cập bất kỳ trường nào bằng cách trích xuất cụ thể. Ví dụ:

```
Info_f.Filename
Info_f.FileSize
Info_f.Format
...
```

- Truy cập nhanh kích thước cửa ảnh hay của ma trận, có thể dùng hàm `size` hoặc lấy thông tin của ma trận bằng hàm `whos`

```
[M N]=size(f);
whos f;
```


https://tieulun.hopto.org

Ví dụ 1.2: *Hiển thị ảnh trong môi trường MATLAB*

```
imshow(f);
```

❖ **Lưu ý:**

- Ảnh sẽ được trình bày dưới dạng một hình (Figure) trong MATLAB. Để hiển thị đầy đủ các ảnh liên tiếp thì cần sử dụng hàm `figure` trước khi dùng `imshow` để đảm bảo hiển thị đầy đủ các ảnh mong muốn. Cấu trúc đơn giản dành cho việc hiển thị nhiều ảnh với nhiều cửa sổ hình khác nhau:

```
figure; imshow(f);
```

Hình 1.5. *Ảnh 'peppers.png' được hiển thị bằng cách dùng hàm* `imshow`

- Có thể hiển thị nhiều ảnh trong cùng một hình với hàm `subplot` dùng chỉ định vị trí thứ tự cửa sổ hiển thị.

```
A=imread('rice.png');
B=imread('cameraman.tif');
C=imread('trees.tif');
figure;
subplot(1,3,1)
imshow(A)
subplot(1,3,2)
```


https://tieulun.hopto.org

```
imshow(B)
subplot(1,3,3)
imshow(C)
```

Hình 1.6. *Hiển thị nhiều ảnh trên cùng một hình*

Ví dụ 1.3: *Lưu ma trận ảnh thành file đồ họa trong Window*

```
clear all;
f=imread('peppers.png');
f_gs=rgb2gray(f);
imwrite(f_gs,'pepper_gray.png','png');
```

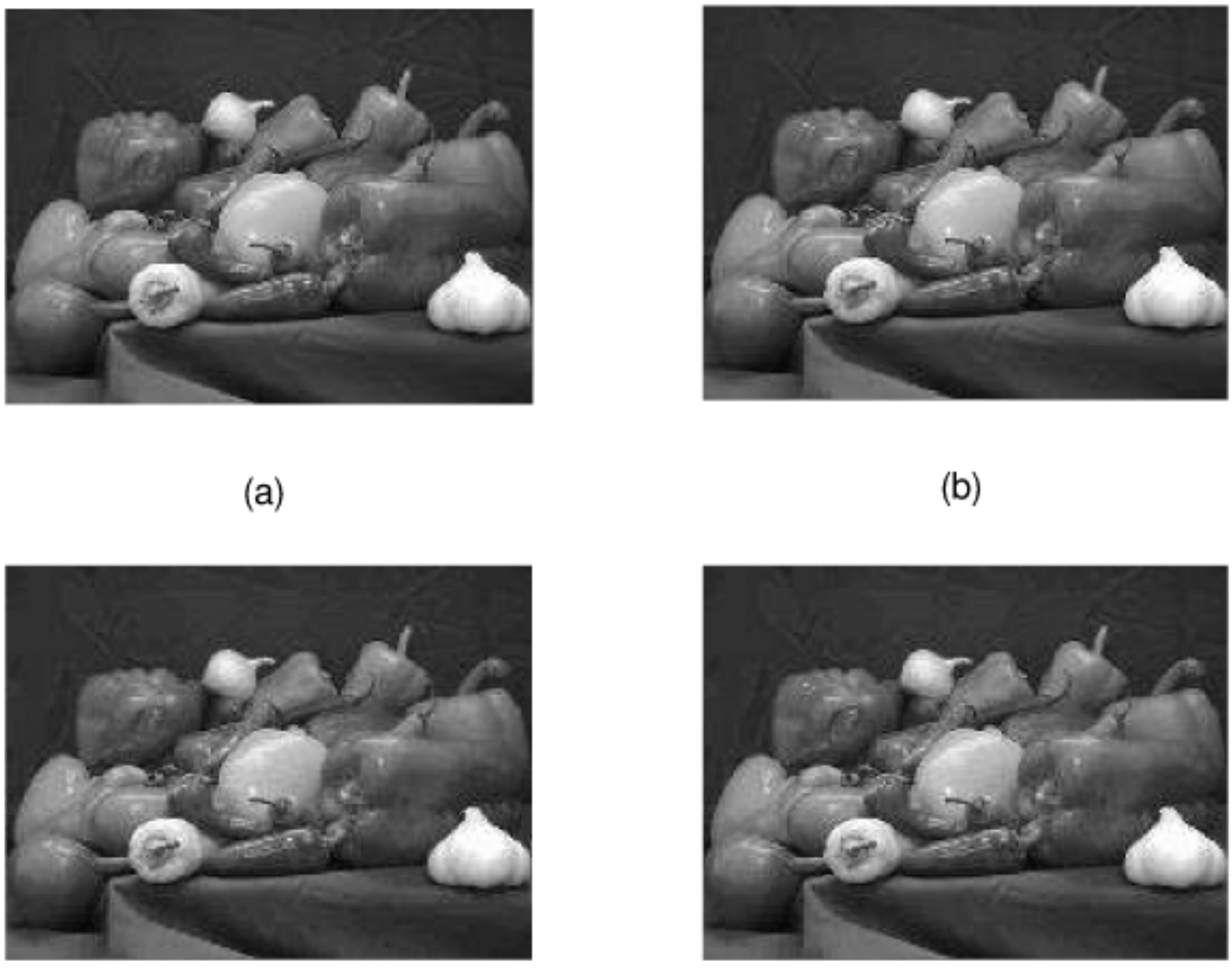
❖ **Lưu ý:**

- Khi dùng hàm `imwrite` cần chỉ rõ tên file và định dạng file hình ảnh. File ảnh sau khi lưu dưới dạng file đồ họa sẽ lưu trong thư mục hiện hành (xem bên cửa sổ Current Window). Hoàn toàn có thể lưu với đường dẫn chỉ rõ như khi dùng hàm `imread`.
- Hàm `imwrite` cho phép lưu ảnh dưới nhiều định dạng khác nhau như tif, gif, jpg, bmp…

```
imwrite(f_gs,'pepper_gray.tif','tif');
imwrite(f_gs,'pepper_gray.png','png);
imwrite(f_gs,'pepper_gray.jpg','jpg','Quality',50);
```

Khi lưu ảnh dưới định dạng JPEG thì cần chỉ rõ hệ số chất lượng của ảnh ngõ ra. Hệ số này tương ứng với tỷ lệ nén ảnh với kỹ thuật nén có tổn hao JPEG.

https://tieulun.hopto.org



Hình 1.7. *Kết quả lưu ảnh dưới định dạng JPEG theo các tỷ số nén khác nhau: (a) 80%; (b) 60%; (c) 40% và (d) 20%*

Hoàn toàn có thể tính được tỷ lệ nén dựa trên tỷ số dung lượng lưu trữ của ảnh gốc trước và sau khi nén (thông thường đây là tỷ số giữa kích thước ảnh lưu trên bộ nhớ máy tính). Trong MATLAB, tỷ số này được tính dựa trên các thông số của ảnh. Ví dụ, với ảnh trong hình 1.7(d) sau khi truy xuất thông tin dùng hàm `imfinfo`:

```
Infor =

           Filename: [1x51 char]
        FileModDate: [1x20 char]
           FileSize: 7237
             Format: 'jpg'
      FormatVersion: ''
              Width: 512
             Height: 384
           BitDepth: 8
```


https://tieulun.hopto.org

```
      ColorType: 'grayscale'
FormatSignature: ''
NumberOfSamples: 1
   CodingMethod: 'Huffman'
  CodingProcess: 'Sequential'
        Comment: {}
```

Trong đó, FileSize là kích thước của ảnh dưới dạng byte. Kích thước của ảnh gốc khi chưa được nén được tính theo công thức:

$$I_{bytes} = Width \times Height \times BitDepth / 8 \quad (1.1)$$

Theo đó, tỷ lệ nén được tính như sau:

$$CompressionRatio = I_{bytes} / FileSize \quad (1.2)$$

Tỷ số nén dành cho ảnh trong hình 1.7(d) được tính cụ thể như sau:

$$CompressionRatio = (512 \times 512 \times 8/8)/11831 \approx 22.15 \quad (1.3)$$

Tùy vào từng ứng dụng cụ thể mà yêu cầu chất lượng của ảnh sau khi nén vẫn phải được đảm bảo. Ngoài việc giảm được dung lượng cần thiết trong quá trình lưu trữ thì nén cũng giúp việc truyền tải thông tin dưới dạng hình ảnh nhanh hơn nhiều so với khi chưa được nén. Người đọc cần tham khảo thêm các thuộc tính khi dùng hàm `imwrite` với từng định dạng khác nhau. Một ví dụ đối với việc lưu ảnh dưới dạng JPEG luôn có thể tùy chọn một số thuộc tính như trong bảng 1.2.

Bảng 1.2. *Các thuộc tính có thể tùy chỉnh khi dùng hàm* `imwrite` *ứng với định dạng JPEG*

Thông số	Giá trị	Mặc định
`'Bitdepth'`	Đại lượng vô hướng chỉ ra độ sâu số hay có thể hiểu là lượng bit dành mã hóa 1 pixel. Với ảnh trắng đen thì có thể là 8, 12 hay 16; còn với ảnh màu thì chỉ có thể là 8 hoặc 12.	8 dành cho ảnh trắng đen 8 dành cho ảnh màu
`'Comment'`	Là một ma trận các ký tự. Hàm imwrite ghi từng hàng của ngõ vào như là một chú thích trong file JPEG	Trống
`'Mode'`	Chỉ rõ dạng nén được sử dụng: 'lossy' hay 'lossless'	'lossy'

https://tieulun.hopto.org

'Quality'	Là một số nguyên từ 0 đến 100, thông số này càng lớn nghĩa là ảnh ngõ ra chất lượng càng cao (ảnh ít bị suy hao do ảnh hưởng của nén) và kích thước ảnh cũng lớn theo.	75

Ví dụ 1.4: *Thay đổi kích thước ảnh*

```
clear all;
f=imread('peppers.png');
f_gs=rgb2gray(f);
f_256=imresize(f,0.5);
f_128=imresize(f_gs,[128 128]);
```

Hàm `imresize` cho phép thay đổi kích thước của ảnh bằng cách chỉ rõ kích thước ảnh ngõ ra [width height] (trong ví dụ trên là `[128 128]`) hay hệ số tỷ lệ (`0.5`). Hàm `imresize` dùng được với ảnh ngõ vào là ảnh đen trắng và ảnh màu.

Với ảnh đen trắng:

```
size(f_128)
ans =
  128   128
```

Với ảnh màu:

```
size(f_256)
ans =
  256   256   3
```

Mặc dù chúng ta thường làm việc với các biến kiểu `integer` nhưng, trong MATLAB, giá trị các điểm ảnh không bị giới hạn bởi chỉ với kiểu `integer`. Bảng 1.3 trình bày các kiểu dữ liệu khác nhau được hỗ trợ bởi MATLAB và Image ProcessingToolbox.

Bảng 1.3. *Các kiểu dữ liệu được hỗ trợ trong MATLAB*

Tên	Mô tả
double	Các giá trị có dấu chấm động có độ chính xác kép nằm trong khoảng xấp xỉ -10^{308} đến 10^{308} (dùng 8 byte để biểu diễn cho một phần tử)
uint8	Số nguyên không dấu trong khoảng [0 255] (1 byte cho một phần tử)
uint16	Số nguyên không dấu trong khoảng [0 65535] (2 byte cho một phần tử)
uint32	Số nguyên không dấu trong khoảng [0 4294967295] (4 byte cho một phần tử)

https://tieulun.hopto.org

int8	Số nguyên có dấu trong khoảng [-128 127] (1 byte cho một phần tử)
int16	Số nguyên có dấu trong khoảng [-32768 32767] (2 byte cho một phần tử)
int32	Số nguyên có dấu trong khoảng [-2147483648 2147483648] (4 byte cho một phần tử)
single	Các giá trị có dấu chấm động với độ chính xác đơn nằm trong khoảng xấp xỉ-10^{38} đến 10^{38} (dùng 4 bytes để biểu diễn cho một phần tử)
char	Kiểu ký tự (2 byte cho một phần tử)
logical	Giá trị logic là 0 hay 1 (1 byte cho một phần tử)

Như vậy, ta có thể thao tác trên một hay nhiều ảnh với sự hỗ trợ rất thuận tiện của phần mềm MATLAB. Các ví dụ vừa trình bày nêu những hướng cơ bản để giúp người đọc tiếp cận vào việc xử lý ảnh dùng MATLAB. Các chương về sau sẽ phân tích các phương pháp trong xử lý ảnh.

BÀI TẬP CHƯƠNG 1

1.1. Trình bày tên các hàm hỗ trợ xử lý ảnh trong Image Processing Toolbox.

1.2. Mở ảnh 'peppers.png' và lưu ảnh này thành các định dạng khác nhau. So sánh kích thước file các định dạng này.

1.3. Mở ảnh 'cameraman.tif' và thay đổi kích thước ảnh này. Hiển thị lại ảnh đã bị thay đổi kích thước và nhận xét.

https://tieulun.hopto.org

Chương 2
KHÁI NIỆM CƠ BẢN

Trong chương này, các khái niệm cơ bản về điểm ảnh, ảnh và các toán tử trên ảnh được trình bày.

2.1. ĐIỂM ẢNH VÀ ẢNH SỐ

2.1.1. Điểm ảnh

Ảnh số là một tập hợp của nhiều điểm ảnh, hay còn gọi là pixel. Mỗi điểm ảnh biểu diễn một màu sắc nhất định (hay độ sáng với ảnh đen trắng) tại một điểm duy nhất, có thể xem một điểm ảnh giống như một chấm nhỏ trong một tấm ảnh màu. Bằng phương pháp đo lường và thống kê một lượng lớn các điểm ảnh, chúng ta hoàn toàn có thể tái cấu trúc các điểm ảnh này thành một ảnh mới gần giống với ảnh gốc. Có thể nói pixel gần giống như các phần tử có cấu trúc hạt trên một ảnh thông thường nhưng được sắp xếp theo từng hàng và cột và chứa các thông tin khác nhau.

Ảnh được biểu diễn dưới dạng một ma trận hai chiều với các pixel được xác định bởi cặp tọa độ (x, y), trong đó, giá trị của pixel tại tọa độ nhất định biểu diễn độ sáng (ảnh đen trắng) hay màu nhất định (ảnh màu). Giá trị độ sáng được số hóa trong xử lý ảnh được gọi là giá trị mức xám. Với một ảnh sau khi được lấy mẫu để cho ra một ảnh số với kích thước xác định gồm có M hàng và N cột, ta nói rằng ảnh có kích thước $M \times N$ và được biểu diễn dưới dạng ma trận như sau:

$$f(x, y) = \begin{bmatrix} f(1,1) & \cdots & f(1,N) \\ \vdots & \ddots & \vdots \\ f(M,1) & \cdots & f(M,N) \end{bmatrix} \tag{2.1}$$

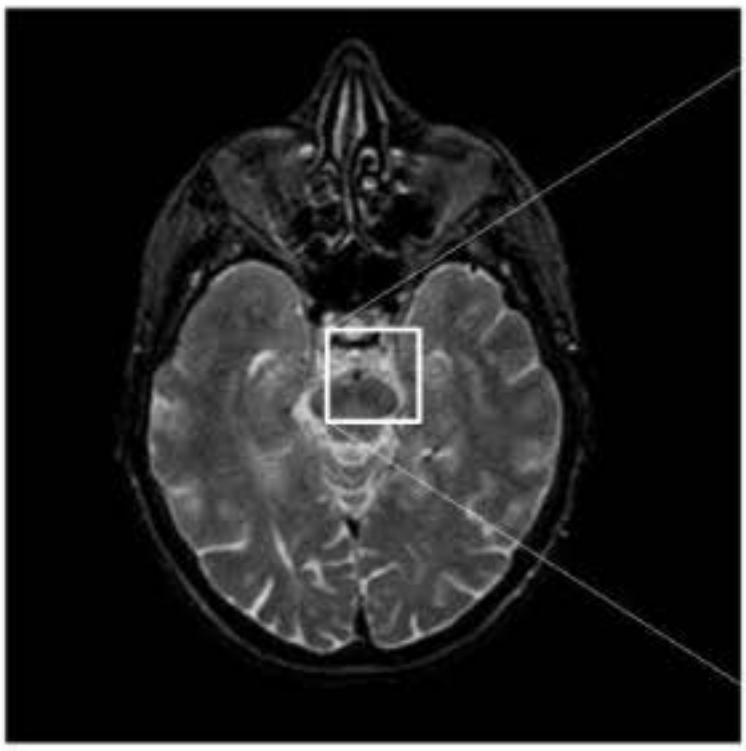
Trong đó:

- $f(x, y)$ – giá trị của pixel tại vị trí (x, y)
- M – số hàng của ảnh số đang xét
- N – số cột của ảnh số đang xét

Đối với ảnh đen trắng, giá trị mức xám thường nằm trong khoảng $0 \le f(x, y) \le G-1$, trong đó G thường được biểu diễn theo lũy thừa của 2. Trong một ví dụ cụ thể, ảnh đen trắng được biểu diễn dưới dạng ma trận

https://tieulun.hopto.org

hai chiều cùng giá trị mức xám cụ thể tương ứng với từng pixel như trong hình 2.1.



173	145	141	170	159	146	165	211
124	99	121	173	174	158	182	223
78	94	120	144	140	140	145	163
116	139	127	115	149	160	129	112
130	129	104	97	131	146	146	142
88	81	73	76	86	86	104	147
72	70	74	80	61	61	72	115
68	71	73	72	75	75	72	75

Hình 2.1. *Biểu diễn ảnh dưới dạng ma trận hai chiều.*

Mỗi điểm ảnh tương ứng với một phần của một đối tượng vật lý trong thế giới ba chiều. Đối tượng này được mô tả bởi một vài nguồn sáng mà trong đó chúng được phản chiếu một phần và hấp thụ một phần bởi vật thể. Phần phản chiếu có thể thu được bằng các cảm biến để mô tả lại khung cảnh tương ứng và nó được ghi lại như là đặc trưng của điểm ảnh, hay nói cách khác, các giá trị này phụ thuộc vào từng loại cảm biến được dùng để phản ánh khung cảnh từ nguồn sáng phản chiếu. Cần phải lưu ý là giá trị độ sáng của những điểm ảnh khác nhau chỉ mang tính tương đối với nhau và rõ ràng là không có ý nghĩa trong điều kiện tuyệt đối. Vì vậy, giá trị điểm ảnh giữa các ảnh khác nhau chỉ nên được so sánh nếu chúng được chuẩn hóa bằng một cách nào đó để loại bỏ sự khác nhau từ các tiến trình vật lý khác nhau.

2.1.2. Phân loại ảnh

Có hai dạng quan trọng trong ảnh số được dùng với nhiều mục đích khác nhau là ảnh màu và ảnh đen trắng (hay còn gọi là ảnh xám). Trong đó, ảnh màu được cấu trúc từ các pixel màu trong khi ảnh đen trắng được xây dựng từ các pixel có giá trị mức xám khác nhau.

Hình 2.2. *Minh họa về sự thay đổi mức xám.*

Ảnh đen trắng: với một ảnh đen trắng được xây dựng từ nhiều pixel mà tại đó biểu diễn một giá trị nhất định tương ứng với một mức xám.

https://tieulun.hopto.org

Những mức xám này trải dài trong một khoảng từ đen sang trắng với bước nhảy rất mịn, thông thường là 256 mức xám khác nhau theo tiêu chuẩn. Do mắt người chỉ có thể phân biệt một cách rõ ràng với khoảng 200 mức xám khác nhau nên vì thế hoàn toàn có thể nhận xét sự thay đổi liên tục các mức xám như minh họa ở hình 2.2.

Ảnh màu: Một ảnh màu thường được tạo thành từ nhiều pixel mà trong đó mỗi pixel được biểu diễn bởi ba giá trị tương ứng với các mức trong các kênh màu đỏ (Red), xanh lá (Green) và xanh dương (Blue) tại một vị trí cụ thể. Các kênh màu Red, Green và Blue (trong không gian màu RGB) là những màu cơ bản mà từ đó có thể tạo ra các màu khác nhau bằng phương pháp pha trộn. Với việc chuẩn hóa 256 (2^8) mức cho từng kênh màu chính, từ đó có thể thấy một pixel màu có thể biểu diễn được một trong $(2^8)^3$=16777216 màu khác nhau. Từ đó có thể thấy rằng với 1 pixel thì chỉ cần 1 byte cho việc lưu trữ đối với ảnh đen trắng và 3 bytes đối với ảnh màu.Vì thế với cùng một ảnh có kích thước nhất định, dung lượng của ảnh màu lưu trên bộ nhớ luôn lớn hơn gấp ba lần dung lượng dành cho ảnh đen trắng trong trường hợp không sử dụng các kỹ thuật nén ảnh.

Việc tính toán dung lượng của ảnh với đầy đủ mức xám ($G=2^8=256$) có thể được biểu diễn như sau:

$$C_{gs} = m \times n \tag{2.2}$$

$$C_{rgb} = 3 \times (m \times n) \tag{2.3}$$

Trong đó:

- C_{gs} – dung lượng của ảnh xám (tính theo byte).
- C_{rgb} – dung lượng của ảnh màu (tính theo byte).
- m – kích thước theo chiều ngang của ảnh.
- n – kích thước theo chiều dọc của ảnh.
- $(m \times n)$ – tổng số pixel của ảnh.

Ảnh nhị phân: chỉ sử dụng duy nhất một bit để biểu diễn một pixel. Do một bit chỉ có thể xác lập hai trạng thái là đóng và mở hay 1 và 0 tương ứng với hai màu là đen và trắng. Do đặc trưng trên mà ảnh nhị phân ít khi được sử dụng trong thực tế.

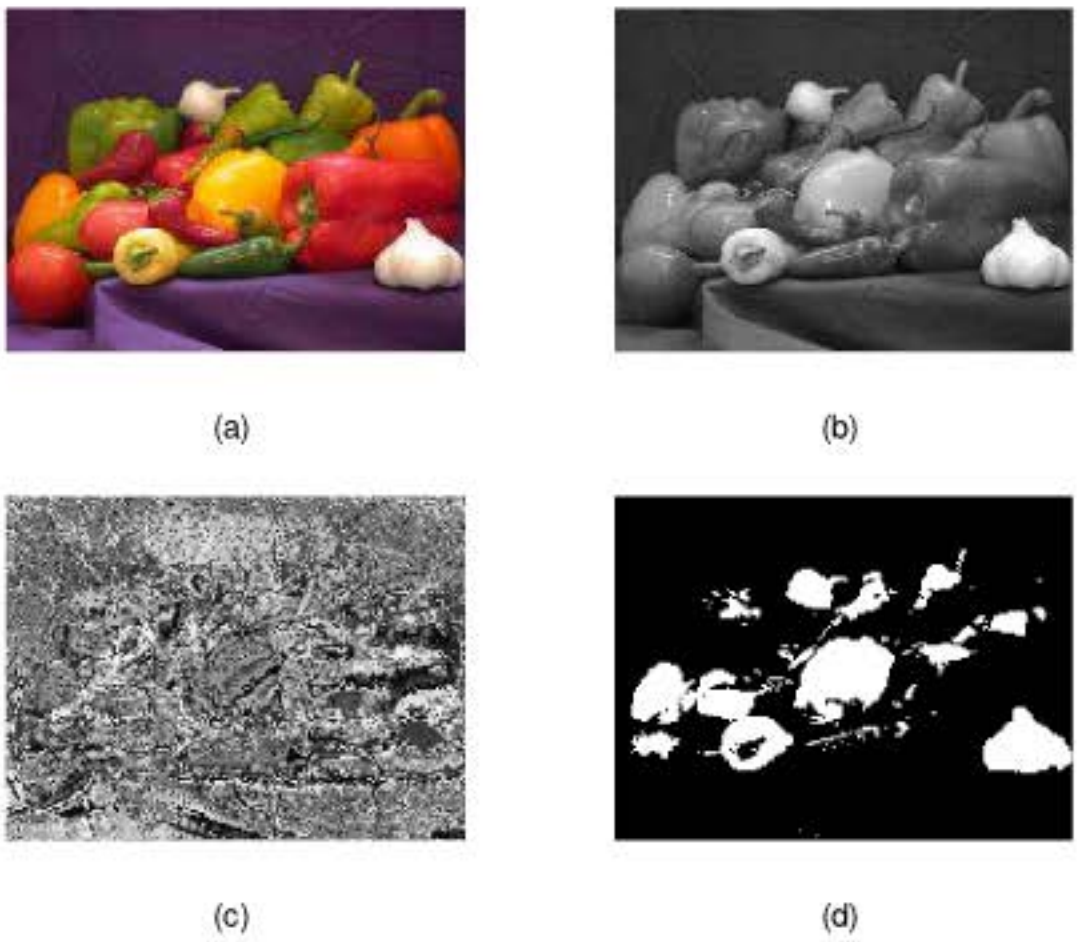
Ảnh chỉ số (indexed): Một vài ảnh màu (hay đen trắng) được tạo thành từ một bảng màu có sẵn bị giới hạn, điển hình thường dùng là tập 256 màu khác nhau. Những ảnh này được gọi là ảnh màu chỉ số hóa

https://tieulun.hopto.org

(indexed) do dữ liệu dành cho mỗi pixel bao gồm chỉ số có sẵn chỉ rõ màu trong tập có sẵn ứng với pixel đang xem xét.

Ví dụ 2.1: Thực hiện chuyển ảnh màu RGB thành các kiểu ảnh xám, nhị phân và chỉ số.

```
f=imread('peppers.png');
fg=rgb2gray(f);
fi=rgb2ind(f,256);
fb=im2bw(f);
subplot(2,2,1)
imshow(f)
xlabel('(a)')
subplot(2,2,2)
imshow(fg)
xlabel('(b)')
subplot(2,2,3)
imshow(fi)
xlabel('(c)')
subplot(2,2,4)
imshow(fb)
xlabel('(d)')
```



Hình 2.3. *Các loại ảnh khác nhau: (a) Ảnh màu; (b) Ảnh xám; (c) Ảnh chỉ số; (d) Ảnh nhị phân*

https://tieulun.hopto.org

2.1.3. Độ phân giải

Với cùng một ảnh, càng nhiều điểm được lấy mẫu thì ảnh chụp càng chi tiết. Mật độ điểm ảnh trong một ảnh được xem như độ phân giải của chính nó. Ảnh có độ phân giải càng cao thì ảnh càng chứa nhiều thông tin. Nói một cách chính xác, khi giữ ảnh tại cùng một kích thước vật lý thì ảnh trở nên sắc nét hơn và chi tiết hơn nếu độ phân giải cao hơn như hình 2.4.

Ví dụ 2.2: Thực hiện thay đổi độ phân giải ảnh

```
f=imread('cameraman.tif');
f128=imresize(f, [128 128]);
f64=imresize(f, [64 64]);
f32=imresize(f, [32 32]);
subplot(2,2,1)
imshow(f)
xlabel('(a)')
subplot(2,2,2)
imshow(f128,'InitialMagnification','fit')
xlabel('(b)')
subplot(2,2,3)
imshow(f64,'InitialMagnification','fit')
xlabel('(c)')
subplot(2,2,4)
imshow(f32,'InitialMagnification','fit')
xlabel('(d)')
```

Hình 2.4. *Ảnh với các độ phân giải khác nhau: (a) 256x256; (b) 128x128; (c) 64x64; (d) 32x32*

2.1.4. Điểm ảnh lân cận

Với một điểm ảnh p tại tọa độ (x, y), luôn có 4 điểm ảnh xung quanh theo phương ngang và dọc với tọa độ được xác định cụ thể sau:

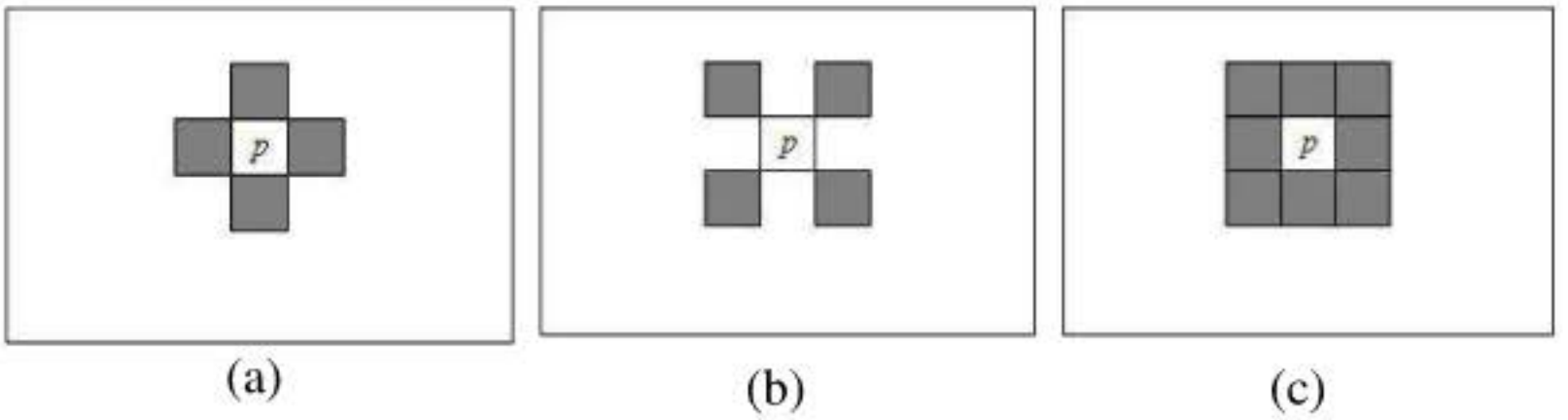
$$(x+1, y), \quad (x-1, y), \quad (x, y+1), \quad (x, y-1) \qquad (2.4)$$

https://tieulun.hopto.org

Tập các điểm ảnh này được gọi là lân cận 4 của p, gọi tắt là $N_4(p)$. Ngoài ra, một dạng lân cận nữa là theo đường chéo được gọi tắt là $N_D(p)$ với các tọa độ được cho sau:

$$(x+1, y+1), \quad (x+1, y-1), \quad (x-1, y+1), \quad (x-1, y-1) \qquad (2.5)$$

Kết hợp hai dạng lân cận này thì chúng ta có dạng lân cận 8 điểm ảnh, gọi tắt là $N_8(p)$. Lưu ý nhỏ là những khái niệm này không áp dụng cho các đường biên của ảnh. Quan sát mô tả ở hình 2.5 để hiểu rõ hơn về các dạng lân cận này.



Hình 2.5. *Các dạng lân cận cơ bản: (a) Lân cận 4 điểm ngang dọc; (b) Lân cận 4 điểm chéo; (c) Lân cận 8 điểm bao quanh.*

2.1.5. Sự liền kề, kết nối, vùng và đường bao

Gọi V là tập các giá trị mức xám dùng để xác định sự liền kề. Với ảnh nhị phân, $V=\{1\}$ nếu chúng ta muốn xét sự liền kề của những điểm ảnh có giá trị là 1. Với ảnh đen trắng, gọi tắt là ảnh xám, có thể là tập các giá trị mong muốn. Chẳng hạn, giá trị có thể có trong ảnh xám nằm trong khoảng [0, 255], do đó V có thể là nhiều tập con chứa 256 giá trị mức xám. Trong khuôn khổ tài liệu này, ta xét ba dạng liền kề sau:

(a) Liền kề 4: Hai điểm ảnh p và q với giá trị nằm trong tập V thuộc liền kề 4 nếu q nằm trong tập $N_4(p)$.

(b) Liền kề 8: Hai điểm ảnh p và q với giá trị nằm trong tập V thuộc liền kề 8 nếu q nằm trong tập $N_8(p)$.

(c) Liền kề m (liền kề hỗn hợp): Hai điểm ảnh p và q với giá trị nằm trong tập V, thuộc liền kề m nếu:

i. q nằm trong tập $N_4(p)$, hay

ii. q nằm trong tập $N_D(p)$ và $N_4(p) \cap N_4(q)$ không tồn tại bất kỳ điểm ảnh nào thuộc tập V.

Xét một ví dụ cụ thể với ảnh nhị phân cùng tập $V=\{1\}$ trong hình 2.6(a). Dễ dàng thấy được sự không rõ ràng trong việc xem xét tính liền kề của những điểm ảnh ở hàng thứ hai và ba trong hình 2.6(b). Nhưng nếu sử dụng đặc trưng liền kề m thì liền kề giữa các điểm ảnh rõ ràng và

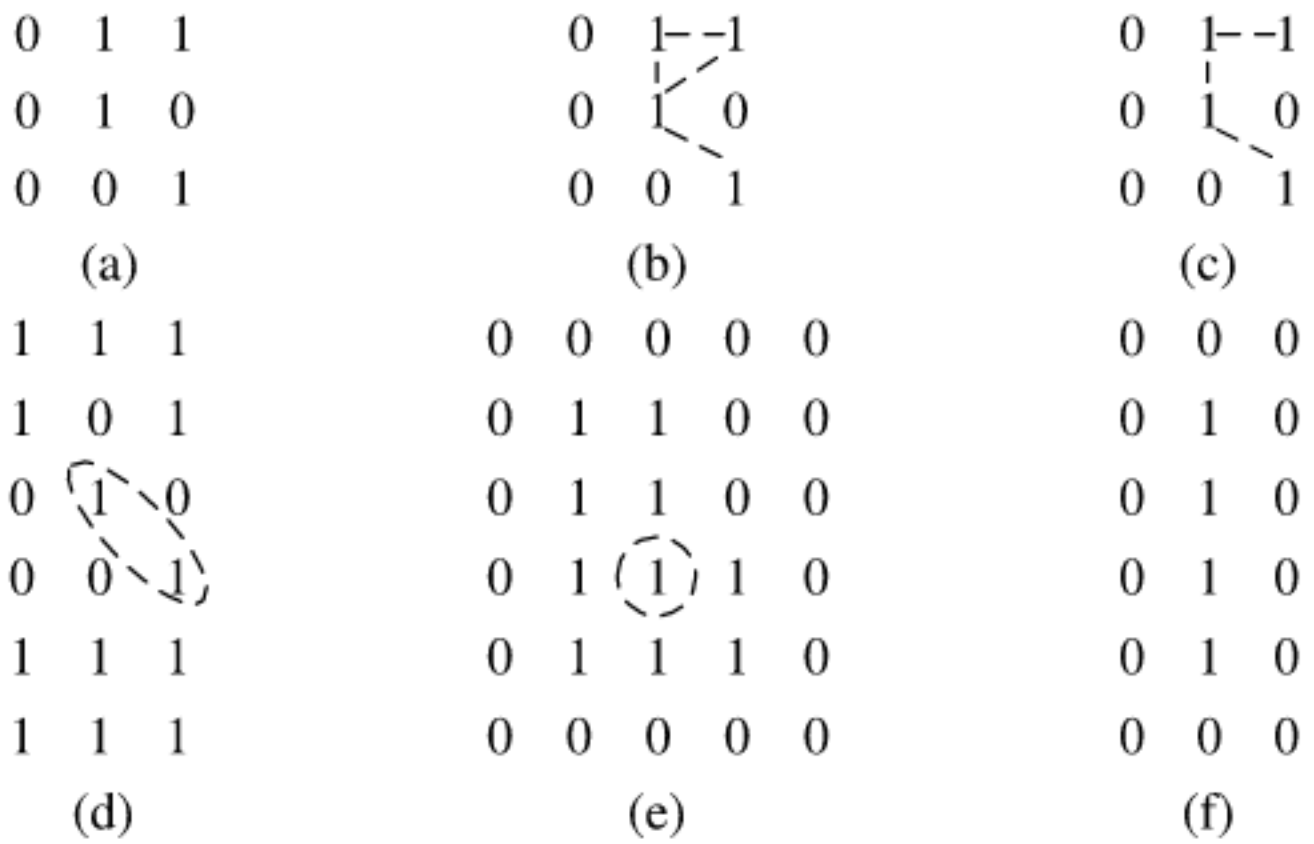
https://tieulun.hopto.org

được mô tả bằng đường đứt nét như trong hình 2.6(c). Giả sử các điểm ảnh liền kề vẽ nên một đường cong mà điểm đầu và điểm cuối trùng nhau thì khi đó ta có một đường kín hay còn gọi là closed path.

Với một ảnh, ta có thể chia thành nhiều phần nhỏ *S* chứa số lượng điểm ảnh như nhau. Ta nói hai điểm ảnh *p* và *q* là được kết nối với nhau trong tập *S* nếu tồn tại một đường giữa chúng gắn kết toàn bộ điểm ảnh trong *S*. Với bất kỳ điểm ảnh *p* nào bên trong *S*, tập các điểm ảnh được kết nối với *p* thì được gọi là thành phần liên kết. Nếu chỉ có duy nhất một thành phần liên kết, thì tập *S* được gọi là tập liên kết. Ví dụ như trong hình 2.6(d), hai vùng R_i và R_j (bit 1) là liền kế với nhau chỉ khi áp dụng phép liền kề 8.

Đường bao (hay còn gọi là đường viền) của vùng *R* là tập các điểm liền kề với các điểm nằm trong phần bù của *R*. Nói cách khác, đường viền của một vùng là tập các điểm ảnh nằm trong vùng đó có ít nhất một điểm ảnh nền liền kề. Trong hình 2.6(e), điểm được khoanh tròn không thuộc đường bao của vùng điểm ảnh có giá trị 1 nếu dùng dạng liền kề 4 giữa vùng và nền của nó.

Đường bao cũng được phân biệt thành hai dạng: đường bao bên trong và bên ngoài. Như trong hình 2.6(f), đường bao bên trong của vùng điểm ảnh có giá trị 1 được xác định là chính nó. Đường bao này không thỏa mãn yêu cầu về đường bao kín như đã đề cập trước đó.



Hình 2.6.*(a) Tổ hợp các điểm ảnh. (b) Mô tả sự kết nối khi dùng dạng liền kề 8. (c) Dùng dạng liền kề m. (d) Hai vùng được xem là liền kề nếu dùng dạng liền kề 8. (e) Điểm được khoanh tròn là một phần đường bao các điểm ảnh có giá trị 1 nếu dạng liền kề 8 giữa nền và vùng được dùng. (f) Đường bao bên trong của vùng điểm ảnh có giá trị 1 không tạo nên một vòng kín, nhưng luôn có đường bao bên ngoài.*

https://tieulun.hopto.org

2.1.6. Đo lường khoảng cách

Giả sử với ba điểm ảnh được cho trước là p, q và z, với tọa độ được xác định (x, y), (s, t), (v,w) tương ứng. D là khoảng cách với các điều kiện:

(a) $D(p,q) \geq 0$ với $D(p,q)=0$ khi $p=q$

(b) $D(p,q)=D(q,p)$

(c) $D(p,z) \leq D(p,q)+D(q,z)$

Khoảng cách Euclid giữa hai điểm p và q được tính như sau:

$$D_e(p,q) = [(x-s)^2 + (y-t)^2]^{1/2} \tag{2.6}$$

Để đo được khoảng cách này, các điểm ảnh có khoảng cách nhỏ hơn hay bằng với giá trị r từ tọa độ (x, y) thì đều là những điểm nằm trong vòng tròn có bán kính r với góc tọa độ là (x, y).

Khoảng cách D_4 giữa điểm p và q được định nghĩa như sau:

$$D_4(p,q) = |x-s| + |y-t| \tag{2.7}$$

Trong trường hợp này, các điểm ảnh có khoảng cách D_4 từ tọa độ (x, y) nhỏ hơn hay bằng với giá trị r từ tâm của hình thoi có tọa độ (x, y). Lấy ví dụ, các pixel với khoảng cách $D_4 \leq 2$ từ tọa độ trung tâm là (x, y) tạo thành đường bao với khoảng cách không đổi:

$$\begin{array}{ccccc} & & 2 & & \\ & 2 & 1 & 2 & \\ 2 & 1 & 0 & 1 & 2 \\ & 2 & 1 & 2 & \\ & & 2 & & \end{array}$$

Các điểm ảnh có khoảng cách $D_4 = 1$ là lân cận với điểm có tọa độ (x, y).

Khoảng cách D_8 giữa hai điểm p và q được xác định như sau:

$$D_8(p,q) = \max(|x-s|, |y-t|) \tag{2.8}$$

Trong trường hợp này, các điểm ảnh có kích thước D_8 từ điểm gốc có tọa độ (x, y) nhỏ hơn hay bằng với giá trị r tạo thành hình vuông có tâm đặt tại gốc tọa độ (x, y). Ví dụ, những điểm ảnh với kích thước $D_8 \leq 2$ từ tọa độ (x, y) tạo thành đường bao với khoảng cách không đổi:

https://tieulun.hopto.org

2	2	2	2	2
2	1	1	1	2
2	1	0	1	2
2	1	1	1	2
2	2	2	2	2

Các điểm ảnh có khoảng cách $D_8 = 1$ là lân cận với điểm có tọa độ (x, y).

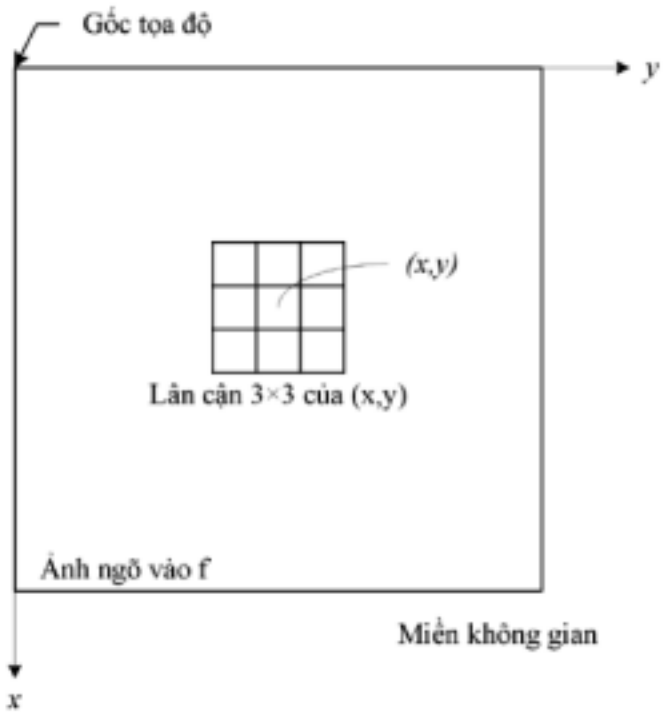
2.2. TOÁN TỬ ẢNH

Tất cả các kỹ thuật xử lý ảnh số được trình bày trong phần này đều được thực hiện trên miền không gian. Qua đây, người học sẽ thực hiện một số bài tập về xử lý ảnh số từ dễ đến khó trên miền không gian. Tất nhiên, các thuật toán này đều đạt được hiệu quả trong tính toán và yêu cầu ít tài nguyên hơn.

Xử lý trên miền không gian mà chúng ta đề cập trong chương này có thể được mô tả bằng biểu thức dạng tổng quát như sau:

$$g(x, y) = T[f(x, y)] \tag{2.9}$$

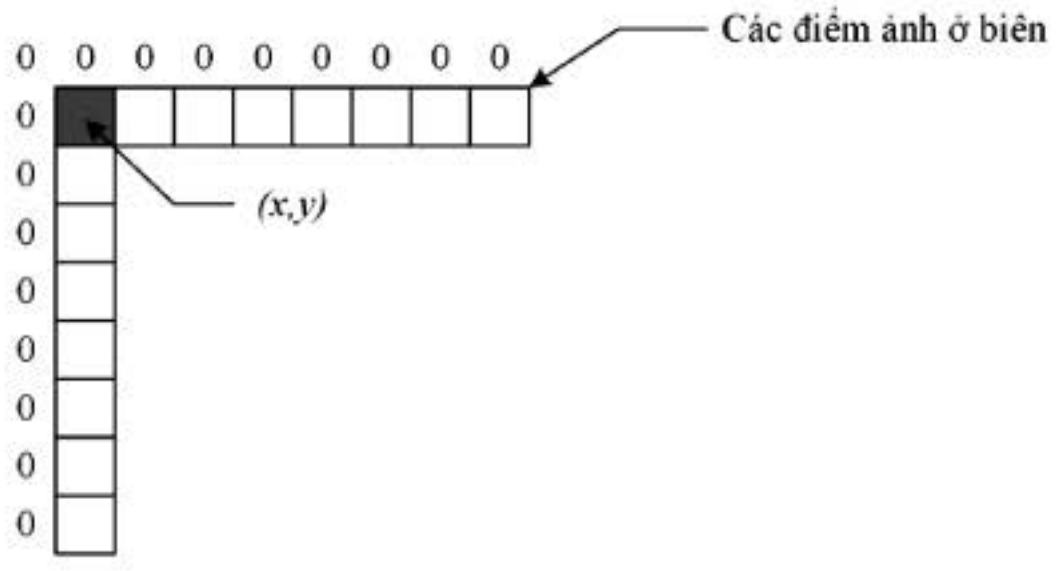
với $f(x, y)$ là ảnh ngõ vào, $g(x, y)$ là ảnh ngõ ra và T là toán tử dành cho f thông qua các điểm ảnh lân cận của (x, y). Toán tử này có thể áp dụng cho ảnh đơn hoặc cho một tập ảnh. Hình 2.7 mô tả việc thực hiện hàm biến đổi (2.6) trên một ảnh đơn với điểm ảnh (x, y) là bất kỳ và lân cận của (x, y) có dạng hình vuông với kích thước nhỏ hơn ảnh gốc.



Hình 2.7.*Lân cận 3×3 của điểm ảnh (x, y) của ảnh đơn trong miền không gian. Các điểm lân cận này được dịch chuyển từ điểm ảnh này sang điểm ảnh khác liên tiếp để tạo ra ảnh mới tuân theo toán tử T*

https://tieulun.hopto.org

Tiến trình xử lý được thực hiện bằng cách di chuyển các lân cận này theo từng điểm ảnh và áp dụng toán tử T tại điểm ảnh đang xét. Do đó, với bất kỳ điểm ảnh (x, y), giá trị của ảnh ngõ ra g tại những điểm ảnh tương ứng sẽ là kết quả của việc áp dụng toán tử T cho các điểm ảnh lân cận với tọa độ trung tâm là (x, y) trong ảnh ngõ vào f. Ví dụ, xét toán tử trung bình cho điểm ảnh bất kỳ (x, y) với lân cận dạng 3×3, giá trị mức xám của ảnh ngõ ra tại điểm (x, y) có giá trị được tính bằng trung bình giá trị của 8 điểm ảnh xung quanh và điểm ảnh (x, y). Việc xử lý được thực hiện từ trái sang phải theo chiều ngang và từ trên xuống dưới theo chiều dọc của ảnh. Lưu ý với các điểm ảnh ở biên thì các điểm ảnh lân cận có giá trị là 0 đối với các lân cận không tồn tại như hình 2.8.



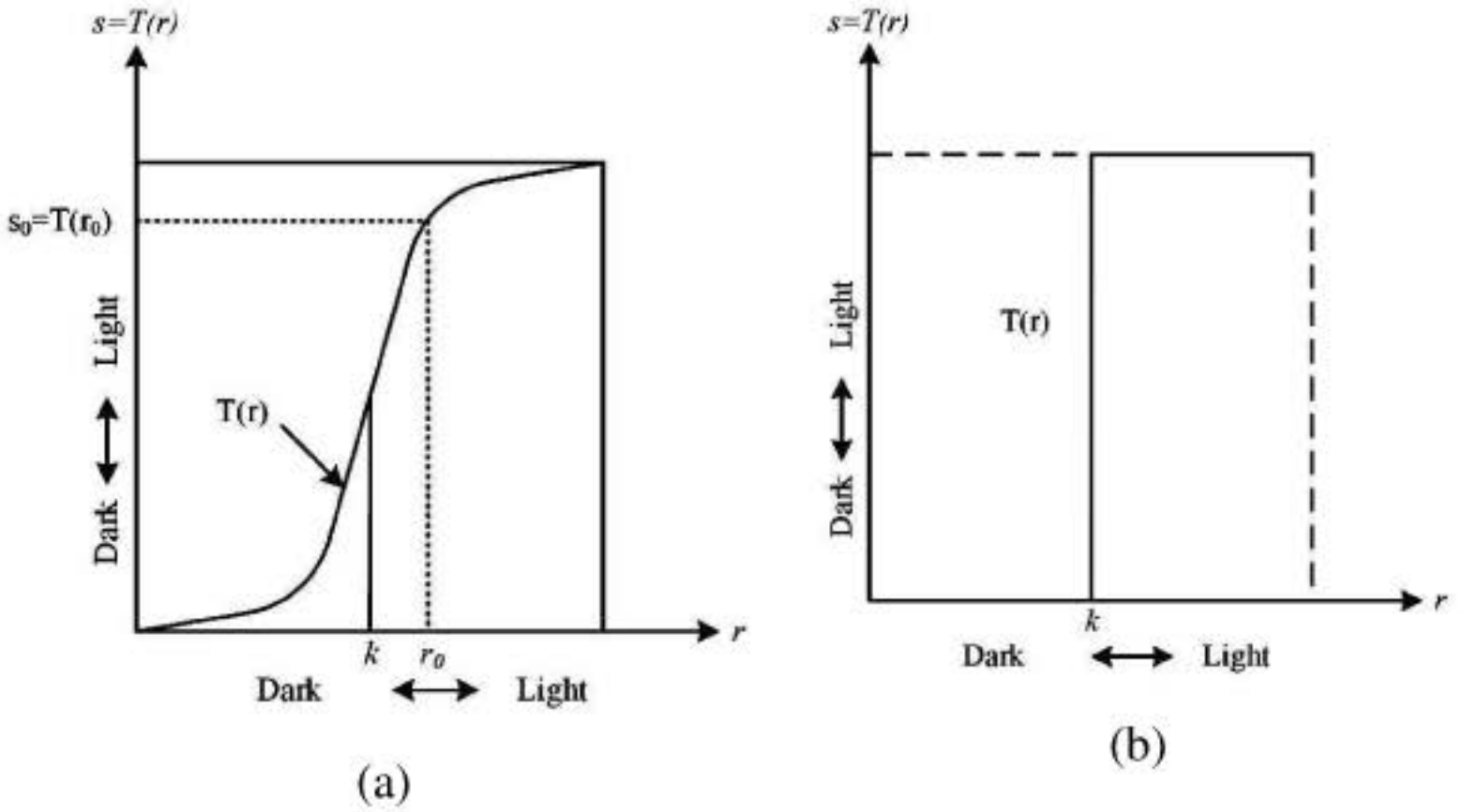
Hình 2.8. *Đối với các điểm ảnh ở biên, khi áp dụng toán tử chuyển đổi mức xám thì các điểm ảnh không tồn tại được xem như có giá trị là 0.*

Trong nhiều tài liệu, việc áp dụng toán tử với sự hỗ trợ của các điểm ảnh lân cận được gọi là lọc không gian và toán tử được cho trước đôi khi còn được gọi là bộ lọc không gian (hay mặt nạ không gian, nhân, mẫu, hay cửa sổ lọc). Lân cận nhỏ nhất có thể tồn tại có kích thước là 1×1. Trong trường hợp này, ảnh ngõ ra phụ thuộc vào ảnh ngõ f tại điểm ảnh (x, y) và toán tử T trong biểu thức (2.9) trở thành hàm chuyển đổi cường độ hay hàm chuyển đổi mức xám (đôi khi còn gọi là ánh xạ mức xám):

$$s = T(r) \tag{2.10}$$

với s và r là các biến biểu diễn cường độ của ảnh ngõ ra g và ngõ vào f tại điểm ảnh (x, y). Ví dụ, nếu $T(r)$ có dạng như trong hình 2.9(a), kết quả của việc biến đổi giá trị mức xám của ảnh ngõ f vào sẽ cho ra ảnh ngõ ra g với độ tương phản cao hơn bằng cách làm tối thêm những điểm ảnh có mức xám nhỏ hơn k và làm sáng hơn những điểm ảnh có mức xám lớn hơn k.

https://tieulun.hopto.org



Hình 2.9. *Hàm biến đổi cường độ điểm ảnh (a) Hàm mở rộng độ tương phản. (b) Hàm lấy ngưỡng.*

2.2.1. Biến đổi âm bản

Âm bản của ảnh với cường độ mức xám nằm trong khoảng $[0, L-1]$ đạt được bằng cách sử dụng biến đổi âm tính được biểu diễn bởi biểu thức (2.11):

$$s = L - 1 - r \tag{2.11}$$

Việc đảo ngược mức xám trong ảnh với phương pháp này sẽ tạo ra một ảnh âm bản tương ứng. Dạng xử lý này đặc biệt phù hợp với các ứng dụng tăng cường các chi tiết trắng hay xám trong các vùng tối trong ảnh, đặc biệt là khi các vùng đen chiếm phần lớn diện tích.

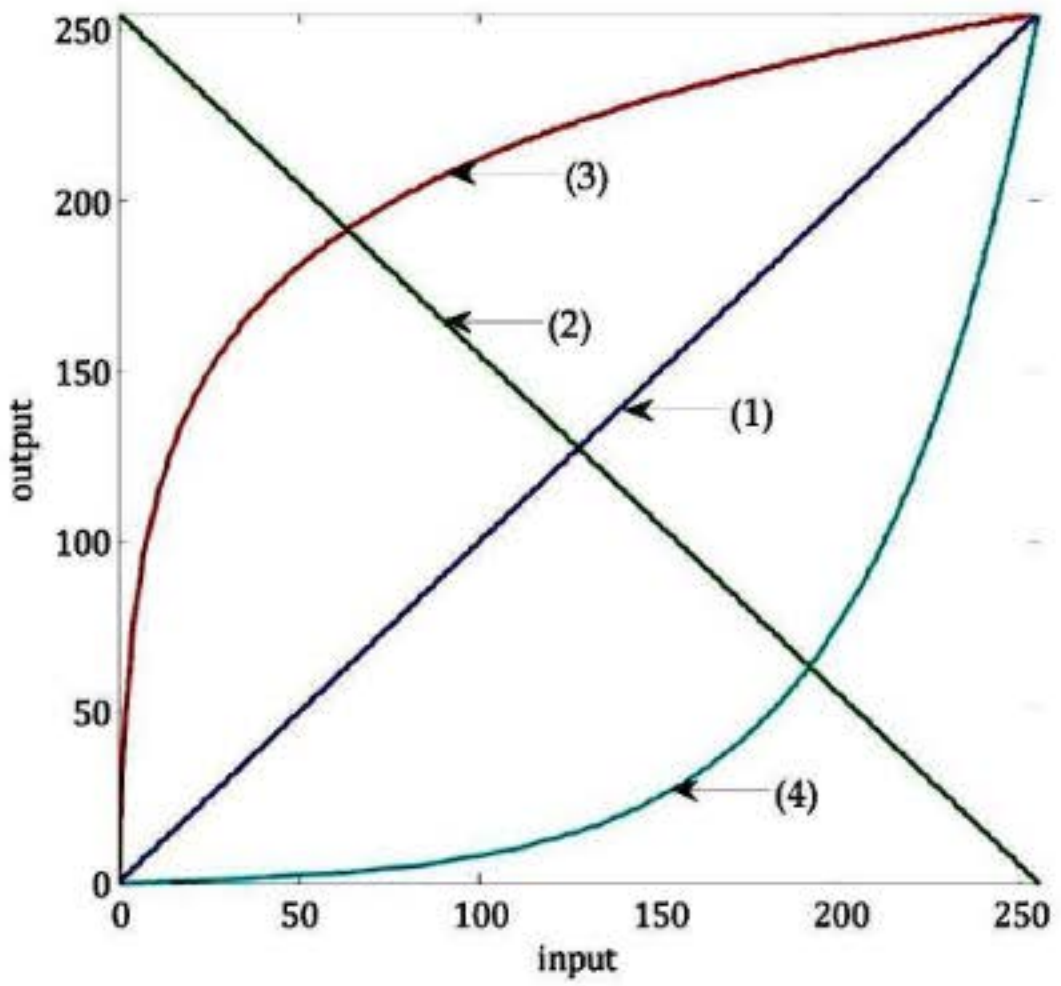
2.2.2. Biến đổi hàm log

Dạng cơ bản của biến đổi hàm log như sau:

$$s = c\log(1 + r) \tag{2.12}$$

với c là hằng số và giả định rằng $r \geq 0$. Hình dạng đường cong (3), (4) như trong hình 2.10 cho thấy hàm biến đổi này sẽ ánh xạ một khoảng hẹp cường độ mức xám thấp của ảnh ngõ vào trở nên rộng hơn và ngược lại với cường độ mức xám cao. Điều này cho phép chúng ta có thể mở rộng các giá trị của điểm ảnh tối trong ảnh ngõ vào. Có thể suy ra tác động ngược lại khi dùng làm logarit ngược.

https://tieulun.hopto.org



Hình 2.10. *Một số hàm biến đổi cường độ cơ bản: (1) Bất biến; (2) Hàm biến đổi âm bản; (3) Hàm log; (4) Hàm log ngược*

Ví dụ 2.3: Thực hiện biến đổi mức xám dùng các hàm biến đổi âm bản và logarit

```
clear all;
f1=imread('cameraman.tif');
f=double(f1);
g_neg=255-f;
g_neg=uint8(g_neg);
g_log=255/log2(256)*log2(f+1);
g_log=uint8(g_log);
subplot(1,3,1)
imshow(f1)
xlabel('(a)')
subplot(1,3,2)
imshow(g_neg)
xlabel('(b)')
subplot(1,3,3)
imshow(g_log)
xlabel('(c)')
```


https://tieulun.hopto.org

(a) (b) (c)

Hình 2.11. *Thực hiện biến đổi ảnh: (a) Ảnh gốc. (b) Ảnh biến đổi âm bản. (c) Ảnh biến đổi hàm log*

Lưu ý: Ma trận ảnh ngõ vào có kiểu dữ liệu mặc định là `uint8`. Do đó với các phép tính, chẳng hạn như logarit, thì yêu cầu phải chuyển về dạng `double`. Và sau khi thực hiện các toán tử trong hàm biến đổi thì cần phải chuyển ngược về kiểu `uint8` như ban đầu. Giá trị của hằng số c giúp chuẩn hóa ảnh với giá trị mức xám cho phép trong khoảng [0 255].

2.2.3. Biến đổi dạng lũy thừa

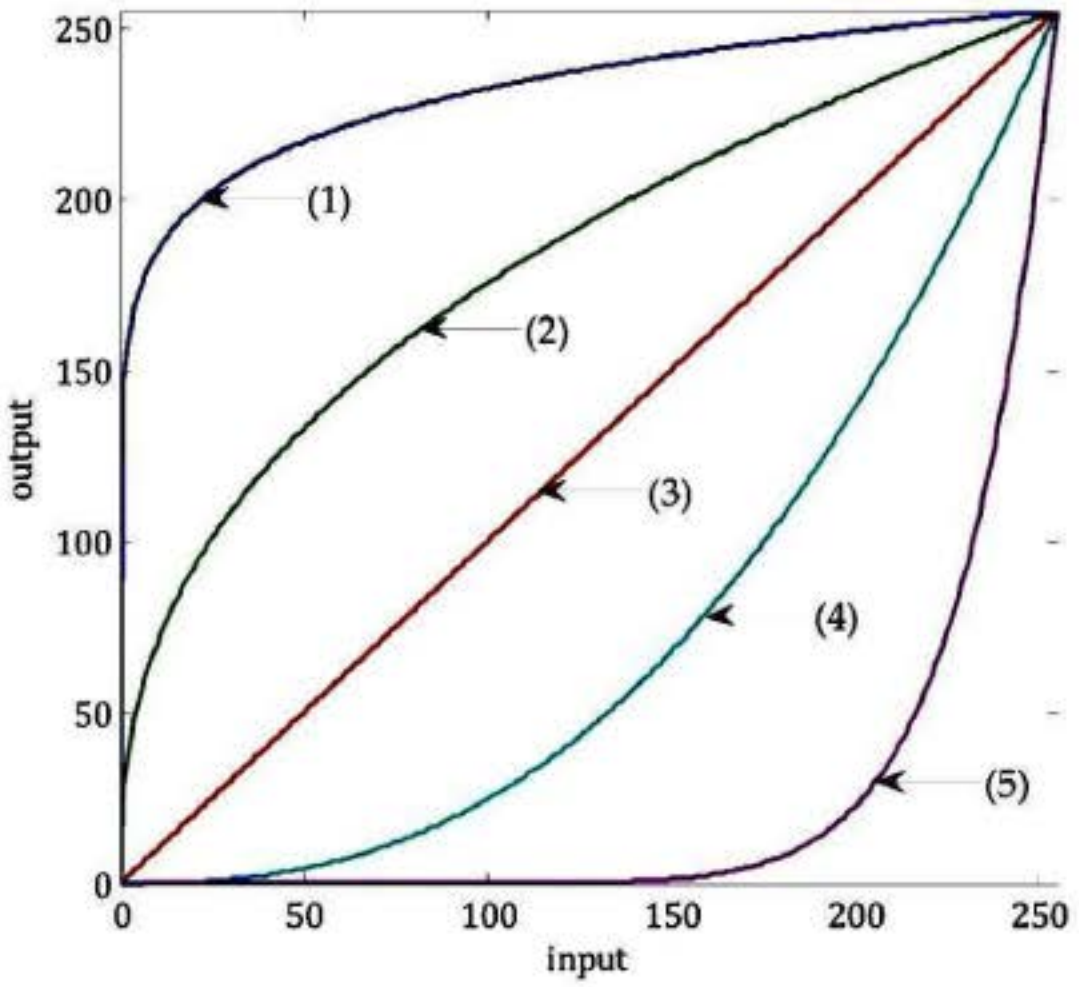
Dạng cơ bản của hàm biến đổi lũy thừa được biểu diễn như sau:

$$s = cr^{\gamma} \tag{2.13}$$

với c và γ là các hằng số dương. Đôi khi biểu thức (2.13) có thể được viết dưới dạng $s = c(r + \varepsilon)^{\gamma}$ cho phép điều chỉnh độ lệch (trong trường hợp ngõ vào là 0). Hình 2.12 biểu diễn giá trị mức xám ngõ ra theo ngõ vào với cùng một giá trị hằng số c, trong khi γ thay đổi. Không giống như hàm biến đổi `log`, việc thay đổi giá trị của γ cho phép đạt được các đường cong ánh xạ khác nhau. Những đường cong với giá trị $\gamma > 1$ có tác động trái ngược với các đường cong ứng với $\gamma < 1$, và biểu thức (2.13) sẽ trở về dạng biến đổi bất biến trong trường hợp $c = \gamma = 1$.

Để thuận tiện, số mũ trong biểu thức lũy thừa (2.13) được gọi là hệ số gamma và tiến trình sử dụng đáp ứng dạng lũy thừa này cũng được gọi là hiệu chỉnh gamma (gamma correction).Ví dụ như các thiết bị dạng CRT (ống tia âm cực) có đáp ứng cường độ theo điện áp là dạng hàm mũ, với số mũ thay đổi xấp xỉ trong khoảng 1.8 đến 2.5. Trong hình 2.12, đường cong có $\gamma = 2.5$ có xu hướng tạo ra ảnh tối hơn.

https://tieulun.hopto.org



Hình 2.12. *Hàm biến đổi lũy thừa: (1)* $\gamma = 0.10$*; (2)* $\gamma = 0.40$*; (3)* $\gamma = 1.00$*; (4)* $\gamma = 2.50$*; (5)* $\gamma = 10.0$

Ví dụ 2.4: Ảnh gốc hình 2.13(a) được đưa vào biểu diễn lên trên màn hình dạng CRT như trong hình 2.13(b). Ảnh khi được thể hiện lên trên màn hình sẽ chịu ảnh hưởng tương ứng với khi dùng hàm lũy thừa $\gamma = 2.5$. Để khắc phục hiện tượng này, trước khi mang hiển thị trên màn hình CRT, ảnh cần một bước tiền xử lý với hàm lũy thừa $\gamma = 0.4$ như trong hình 2.13(c). Do đó, khi hiển thị ảnh lên trên màn hình như trong hình 2.13(d) thì ta được ảnh gần giống với ảnh gốc, không bị lệch màu như hình 2.13(b). So sánh giữa ảnh gốc với ảnh sau khi hiệu chỉnh gamma để thấy được ứng dụng của hàm biến đổi lũy thừa.

```
clear all;
f1=imread('cameraman.tif');
f=double(f1)/255;
gamma1=2.5;
g1=255*f.^gamma1;
g1=uint8(g1);
gamma2=0.4;
g2=255*f.^gamma2;
g2=uint8(g2);
g3=(double(g2)/255).^gamma1;
g3=255*g3;
```


https://tieulun.hopto.org

```
g3=uint8(g3);
subplot(2,2,1)
imshow(f1)
xlabel('(a)')
subplot(2,2,2)
imshow(g1)
xlabel('(b)')
subplot(2,2,3)
imshow(g2)
xlabel('(c)')
subplot(2,2,4)
imshow(g3)
xlabel('(d)')
```

Hình 2.13. *Ứng dụng hiệu chỉnh gamma trên các màn hình CRT: (a) Ảnh gốc; (b) Hiển thị ảnh gốc trên màn hình CRT; (c) Hiệu chỉnh gamma với* $\gamma = 0.4$*; (d) Hiển thị ảnh sau khi hiệu chỉnh trên màn hình CRT.*

Ví dụ 2.5: Hiệu chỉnh gamma dùng hàm `imadjust` trong Toolbox

```
clear all
f1=imread('cameraman.tif');
f=double(f1);
```

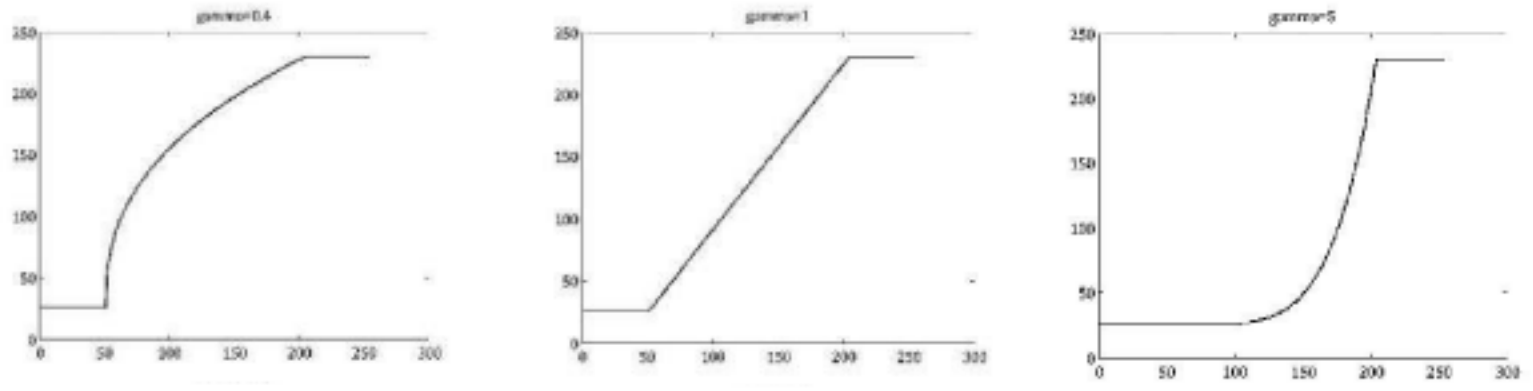

https://tieulun.hopto.org

```
f=f/255;

g1=imadjust(f,[0.2 0.8],[0.1 0.9],0.4);
g1=uint8(g1*255);


g2=imadjust(f,[0.2 0.8],[0.1 0.9]);
g2=uint8(g2*255);

g3=imadjust(f,[0.2 0.8],[0.1 0.9],5);
g3=uint8(g3*255);
```

Lưu ý: Cú pháp của hàm `imadjust` như sau:

```
J = imadjust(I,[low_in; high_in],[low_out; high_out],gamma)
```

Trong đó các thông số `[low_in; high_in]`, `[low_out; high_out]` phải có giá trị nằm trong khoảng [0 1]. Vì vậy trước khi dùng cần phải chuẩn hóa cường độ mức xám của ảnh ngõ vào. Thực hiện biến đổi ngược lại và chuyển kiểu dữ liệu để thu được ảnh ngõ ra và cho phép hiển thị. Hàm biến đổi tương ứng với các ảnh ngõ ra trong *ví dụ 2.5* được biểu diễn như trong hình 2.14.



(a) (b) (c)

Hình 2.14. *Các hàm ánh xạ trong ví dụ 2.5: (a) gamma=0.4;(b) gamma=1; (c) gamma=5*

BÀI TẬP CHƯƠNG 2

2.1. Thực hiện mở ảnh 'pout.tif' và 'peppers.png'. Xoay hai ảnh này sang trái/phải 90°. Hiển thị ảnh đã xoay.

2.2. Mở ảnh 'peppers.png'. Giảm độ chói của tất cả các điểm ảnh đi 50 nếu điểm có độ chói lớn hơn 100. Quan sát ảnh ngõ ra và nhận xét.

2.3. Thực hiện biến đổi âm bản cho ảnh 'coins.png'.

2.4. Giải thích việc hiệu chỉnh gamma và cho ví dụ.

https://tieulun.hopto.org

Chương 3
PHÉP BIẾN ĐỔI ẢNH

Trong chương này, hai phép biến đổi cơ bản là Fourier và Wavelet được trình bày. Song song đó các ví dụ về áp dụng các loại biến đổi này lên một ảnh cũng được triển khai.

3.1. BIẾN ĐỔI FOURIER RỜI RẠC

Cho $f(t)$ là một hàm liên tục với biến t có chu kỳ T được biểu diễn dưới dạng là tổng của các hàm sine và cosine với các hệ số xấp xỉ. Và tổng này, được biết như là chuỗi Fourier có dạng như sau:

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{j\frac{2\pi n}{T}t} \tag{3.1}$$

với

$$c_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t)e^{-j\frac{2\pi n}{T}t}dt\,; n = 0, \pm 1, \pm 2, \ldots \tag{3.2}$$

là các hệ số. Thật sự biểu thức (3.1) là dạng triển khai của hàm sine và cosine theo như trong công thức Euler $e^{j\theta} = \cos\theta + j\sin\theta$

Trong xử lý tín hiệu, xung đơn vị $\delta(t)$ được định nghĩa như sau:

$$\delta(t) = \begin{cases} \infty & ; t = 0 \\ 0 & ; t \neq 0 \end{cases} \tag{3.3}$$

và có đáp ứng

$$\int_{-\infty}^{\infty} \delta(t)dt = 1 \tag{3.4}$$

Xung đơn vị chỉ có thể được quan sát tại thời điểm $t = 0$ với biên độ vô cực, do đó, nó có đặc trưng chọn lọc thông tin với tích phân như sau:

$$\int_{-\infty}^{\infty} f(t)\delta(t)dt = f(0) \tag{3.5}$$

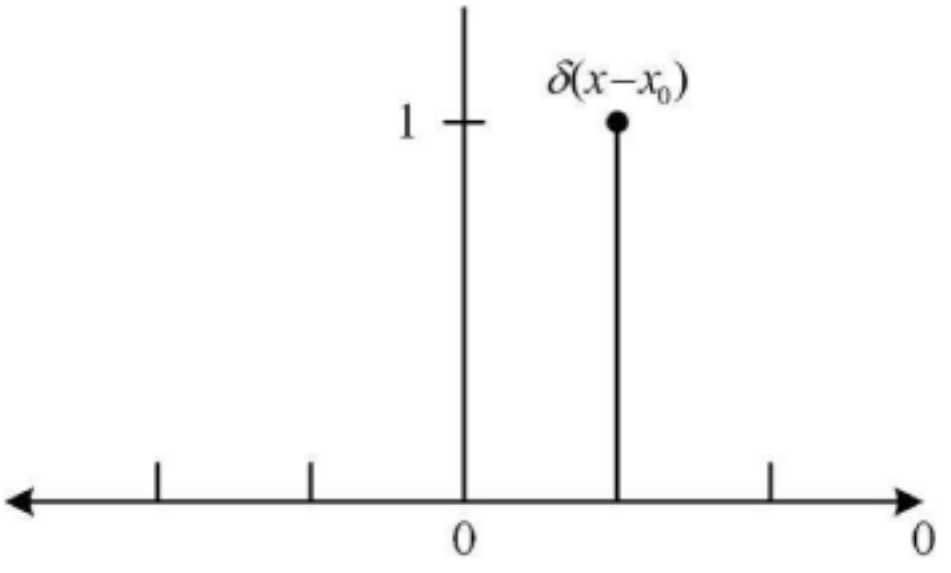
Biểu thức (3.5) chỉ đơn giản giữ lại thành phần $f(t)$ tại thời điểm $t = 0$. Từ đây, việc trích xuất thành phần tín hiệu $f(t)$ tại thời điểm t_0 chỉ đơn giản thay đổi vị trí của xung đơn vị $\delta(t - t_0)$. Trong trường hợp này, với đặc trưng chọn lọc:

https://tieulun.hopto.org

$$\int_{-\infty}^{\infty} f(t)\delta(t-t_0)dt = f(t_0) \tag{3.6}$$

Ví dụ, nếu hàm $f(t) = \cos(t)$, sử dụng xung đơn vị $\delta(t-\pi)$ trong biểu thức (3.6) sẽ cho kết quả $f(\pi) = \cos(\pi) = -1$.

Hình 3.1. *Xung đơn vị rời rạc tại vị trí* $\boldsymbol{x = x_0}$. *Biến* $\boldsymbol{x}$ *là rời rạc và* $\boldsymbol{\delta}$ *bằng 0 tại bất kỳ điểm nào mà* $\boldsymbol{x = x_0}$



Còn đối với biến rời rạc x, trong trường hợp này, ta có xung đơn vị rời rạc tương ứng $\delta(x)$ cũng thực hiện nhiệm vụ tương ứng với $\delta(t)$ như trong hệ thống liên tục. Cụ thể, nó được định nghĩa như sau:

$$\delta(x) = \begin{cases} 1 & ; x = 0 \\ 0 & ; x \neq 0 \end{cases} \tag{3.7}$$

Từ đó, biểu thức (3.6) tương ứng dưới dạng rời rạc được viết lại như sau:

$$\sum_{x=-\infty}^{\infty} f(x)\,\delta(x-x_0) = f(x_0) \tag{3.8}$$

Với một hàm $f(t)$ liên tục trên miền thời gian, biểu thức biến đổi Fourier $F(\mu)$ được viết gọn như sau:

$$F(\mu) = \int_{-\infty}^{\infty} f(t)e^{-j2\pi\mu t}\,dt \tag{3.9}$$

Sử dụng công thức Euler, ta có

$$F(\mu) = \int_{-\infty}^{\infty} f(t)[\cos(2\pi\mu t) - j\sin(2\pi\mu t)]\,dt \tag{3.10}$$

Ngược lại, cho $F(\mu)$, hàm $f(t)$ đạt được bằng phép biến đổi ngược Fourier:

https://tieulun.hopto.org

$$f(t) = \int_{-\infty}^{\infty} F(\mu) e^{j2\pi\mu t} d\mu \tag{3.11}$$

Nếu $f(t)$ là thực thì biến đổi Fourier của nó có dạng phức. Do chỉ còn mỗi biến tần số tồn tại sau khi tích phân nên thông thường người ta còn gọi biến đổi Fourier là biến đổi tần số.

Xem như đã biết, tích chập hai tín hiệu được viết theo sau:

$$f(t) \star h(t) = \int_{-\infty}^{\infty} f(\tau) h(t-\tau) d\tau \tag{3.12}$$

Khi đó biến đổi Fourier của tích chập hai tín hiệu có dạng:

$$\begin{aligned} \mathrm{F}\{f(t) \star h(t)\} &= \int_{-\infty}^{\infty} f(\tau)\left[H(\mu) e^{-j2\pi\mu\tau}\right] d\tau \\ &= H(\mu) \int_{-\infty}^{\infty} f(\tau) e^{-j2\pi\mu\tau} d\tau \\ &= H(\mu) F(\mu) \end{aligned} \tag{3.13}$$

Có thể thấy biến đổi Fourier của tích chập hai tín hiệu là tích của Fourier của hai tín hiệu riêng lẻ. Hay nói cách khác, $f(t) \star h(t)$ và $H(\mu)F(\mu)$ là một cặp biến đổi Fourier. Kết quả này là một phần lý thuyết tích chập và có thể được viết như sau:

$$f(t) \star h(t) \Leftrightarrow H(\mu) F(\mu) \tag{3.14a}$$

Cần lưu ý vế phải đạt được bằng phép biến đổi Fourier vế trái, trong khi đó, vế trái cũng thu được bằng phép biến đổi Fourier ngược vế phải. Phần còn lại của lý thuyết tích chập có thể được biểu diễn dưới dạng biểu thức sau:

$$f(t) h(t) \Leftrightarrow H(\mu) \star F(\mu) \tag{3.14b}$$

3.1.1. Biến đổi Fourier rời rạc

Gọi $\tilde{f}(t)$ là hàm liên tục được lấy mẫu với khoảng ΔT đồng nhất, biến đổi Fourier của $\tilde{f}(t)$ có dạng $\tilde{F}(\mu)$

$$\tilde{F}(\mu) = \int_{-\infty}^{\infty} \tilde{f}(t) e^{-j2\pi\mu t}\, dt \tag{3.15}$$

Trong đó $\tilde{f}(t) = \sum_{n=-\infty}^{\infty} f(t)\delta(t - n\Delta T)$. Thế $\tilde{f}(t)$ vào (3.15), ta có

$$\tilde{F}(\mu) = \sum_{n=-\infty}^{\infty} f_n e^{-j2\pi\mu\Delta T} \tag{3.16}$$

https://tieulun.hopto.org

Mặc dù f_n là hàm rời rạc nhưng Fourier của nó $\tilde{F}(\mu)$ là liên tục và tuần hoàn với chu kỳ $1/\Delta T$. Và vì thế chỉ cần xét đặc trưng $\tilde{F}(\mu)$ trong một chu kỳ.

Giả sử ta lấy M mẫu trong khoảng thời gian $\mu = 0$ và $\mu = 1/\Delta T$. Khi đó ta hoàn toàn có thể suy luận được rằng μ bao gồm các giá trị:

$$\mu = \frac{m}{M\Delta T} \qquad m = 0, 1, 2, \dots, M-1 \tag{3.17}$$

Thay kết quả (3.16) vào (3.17) thu được F_m

$$F_m = \sum_{n=1}^{M-1} f_n e^{-j2\pi mn/M} \qquad m = 0, 1, 2, \dots, M-1 \tag{3.18}$$

Đây là dạng mở rộng của biến đổi Fourier rời rạc, trong đó $\{f_n\}$ gồm M mẫu được lấy từ hàm liên tục $f(t)$. Do đó, kết quả của biến đổi Fourier rời rạc, tập $\{F_m\}$, cũng gồm M giá trị rời rạc dạng phức. Một cách ngược lại, chúng ta hoàn toàn có thể thu được tập mẫu $\{f_n\}$ với phép biến đổi Fourier rời rạc.

$$F_m = \sum_{n=1}^{M-1} f_n e^{-j2\pi mn/M} \qquad m = 0, 1, 2, \dots, M-1 \tag{3.19}$$

Trong môi trường đa chiều, cụ thể là 2 chiều đối với ảnh số, thì hàm biến đổi Fourier phải thỏa mãn hai biến số tương ứng với chiều ngang và dọc của ảnh. Cụ thể, biến đổi Fourier rời rạc 2-D được thực hiện theo biểu thức sau:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{3.20}$$

Với $f(x,y)$ là ảnh số có kích thước $M \times N$. Như ở dạng 1-D, biểu thức (3.20) thỏa mãn toàn bộ các biến u và v trong khoảng $u = 0, 1, 2, \dots, M-1$ và $v = 0, 1, 2, \dots, N-1$. Từ dạng Fourier $F(u,v)$, chúng ta thu được hàm $f(x,y)$ bằng hàm biến đổi ngược Fourier rời rạc:

$$f(x,y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} F(u,v) e^{-j2\pi\left(\frac{ux}{M}+\frac{vy}{N}\right)} \tag{3.21}$$

Với $x = 0, 1, 2, \dots, M-1$ và $y = 0, 1, 2, \dots, N-1$. Hai biểu thức (3.20) và (3.21) tạo thành cặp biến đổi Fourier rời rạc 2-D. Giá trị của $F(u,v)$ trong biểu thức (3.20) đôi khi còn được xem như là các hệ số Fourier ở dạng triển khai.

https://tieulun.hopto.org

Giá trị biến đổi tại gốc của miền tần số (nghĩa là hệ số $F(0,0)$) được gọi là thành phần một chiều DC của biến đổi Fourier. Thuật ngữ này có thể gọi là thành phần có tần số bằng không. Không khó để chứng minh $F(0,0)$ bằng MN lần giá trị trung bình của $f(x,y)$. Ngay cả khi hàm $f(x,y)$ là dạng số thực thì biến đổi Fourier của nó luôn là dạng phức. Phương pháp cơ bản của việc phân tích hàm biến đổi một cách trực quan chính là tính toán phổ của nó (nghĩa là biên độ của hàm F(u,v) và biểu diễn nó như là một ảnh độc lập. Bằng việc giả sử $R(u,v)$ và $I(u,v)$ biểu diễn các thành phần thực và phần ảo của $F(u,v)$, phổ Fourier được định nghĩa như sau:

$$|F(u,v)| = [R^2(u,v) + I^2(u,v)]^{1/2} \tag{3.22}$$

Góc pha của của biến đổi được xác định như sau:

$$\Phi(u,v) = tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right] \tag{3.23}$$

Hai hàm này có thể được sử dụng để biểu diễn $F(u,v)$ dưới dạng hệ tọa độc cực thông thường cho một đại lượng:

$$F(u,v) = |F(u,v)|e^{-j\Phi(u,v)} \tag{3.24}$$

Phổ công suất được định nghĩa như là bình phương của độ lớn

$$\begin{aligned} P(u,v) &= |F(u,v)|^2 \\ &= R^2(u,v) + I^2(u,v) \end{aligned} \tag{3.25}$$

3.1.2. Tính toán và biểu diễn biến đổi 2-DFT trong MATLAB

Dựa trên công thức (3.20), biến đổi Fourier rời rạc theo thời gian có thể được thực hiện thông qua các phép tính thông thường. Xét ví dụ với một ma trận cụ thể để hiểu rõ hơn về phương pháp tính biến đổi Fourier.

Ví dụ 3.1: Thực hiện biến đổi Fourier không dùng hàm có sẵn trong MATLAB

```
clear all;clc
a=zeros(4,4);
a(2:3,2:3)=1;
for u=1:size(a,1)
    for v=1:size(a,2)
        for x=1:size(a,1)
for y=1:size(a,2)
                phi=(2*pi*(u-1)*(x-1)+2*pi*(v-1)*(y-1))/4;
```


https://tieulun.hopto.org

```
                p(x,y)=complex(cos(phi),-sin(phi));
end
        end
        F(u,v)=sum(sum(p.*a));
    end
end
```

Cho ma trận a có kích thước 4×4 với các giá trị cụ thể sau:

$$a = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Biến đổi Fourier của a với việc tính toán từng giá trị cụ thể tại từng vị trí, chẳng hạn như với $u = 2$ và $v = 1$.

$$\begin{aligned} F(2,1) &= e^{-j2\pi\left(\frac{1}{4}\right)} + e^{-j2\pi\left(\frac{1}{4}\right)} + e^{-j2\pi\left(\frac{2}{4}\right)} + e^{-j2\pi\left(\frac{2}{4}\right)} \\ &= -j - j - 1 - 1 \\ &= -2 - 2j \end{aligned}$$

Cần lưu ý trong MATLAB, vị trí của ma trận được xem xét với giá trị từ 1 trở đi, do đó, theo như công thức (3.20) thì cần có sự điều chỉnh nhỏ. Cụ thể, trong đoạn code MATLAB trên, các giá trị của u và v hay của x và y khi tính toán góc phi đều phải điều chỉnh lại sao cho phù hợp. Hàm `complex` được dùng biểu diễn giá trị dưới dạng số phức. Kết quả biến đổi Fourier đầy đủ của a được tính:

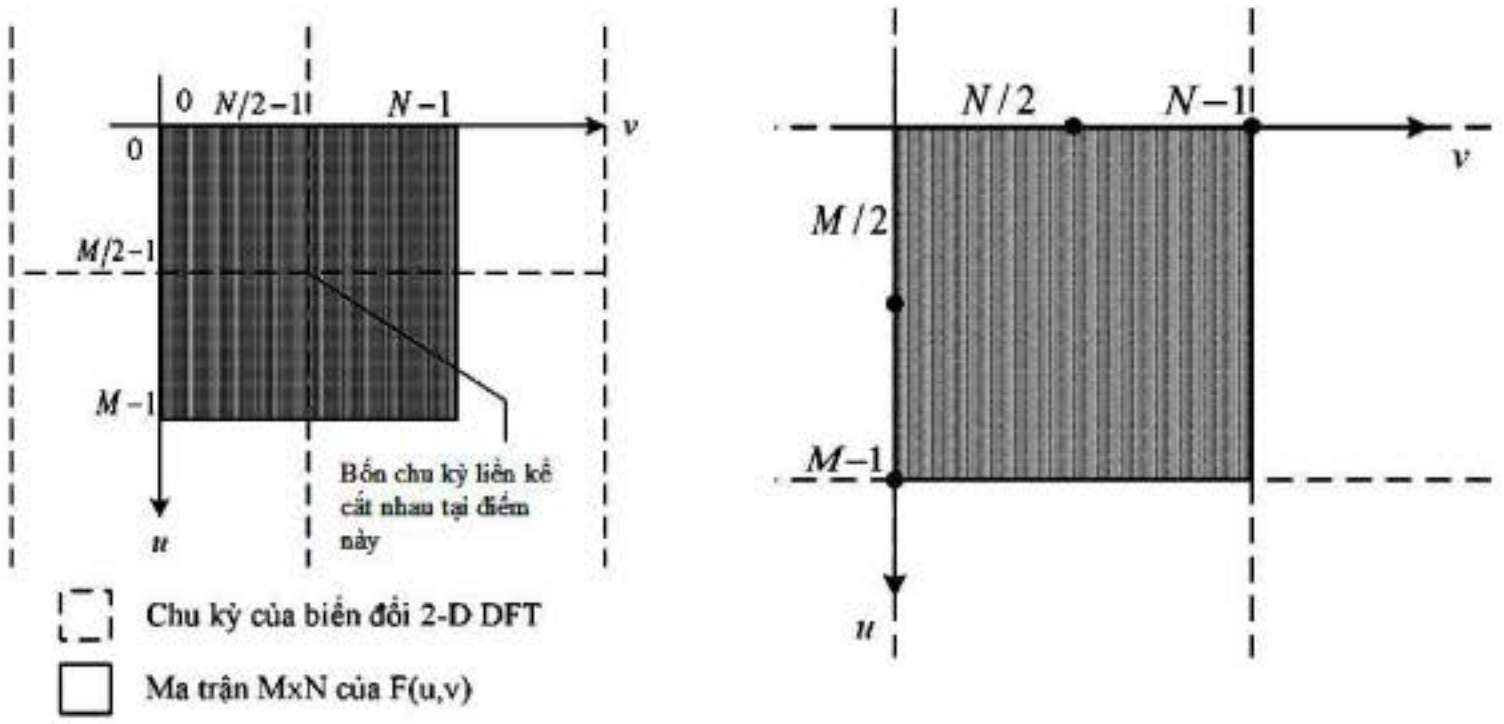
$$\mathrm{A} = \begin{bmatrix} 4 + j0 & -2 - j2 & 0 + j0 & -2 + j2 \\ -2 - j2 & 0 + j2 & 0 + j0 & 2 + j0 \\ 0 + j0 & 0 + j0 & 0 + j0 & 0 + j0 \\ -2 + j2 & 0 + j0 & 0 + j0 & 0 - j2 \end{bmatrix}$$

Trong MATLAB, biến đổi DFT và biến đổi ngược DFT được thực hiện bằng phương pháp sử dụng thuật toán biến đổi nhanh Fourier FFT. Biến đổi FFT của ảnh f với kích thước $M \times N$ được thực hiện bởi hàm `fft` trong toolbox với cú pháp như sau

```
F=fft2(f)
```

Hàm này sẽ trả về biến đổi Fourier cũng với kích thước $M \times N$ có dữ liệu được sắp xếp như trong hình 3.2(a), khi đó, gốc tọa độ nằm ở góc trên bên trái của ma trận và chu kỳ kết thúc tại trung tâm của hình chữ nhật tần số.

https://tieulun.hopto.org



(a) (b)

Hình 3.2. *(a) Phổ Fourier có kích thước $M \times N$ và chỉ ra bốn góc tư của các chu kỳ tiếp giáp nhau trong phổ. (b) Phổ có được bằng cách nhân hàm $f(x, y)$ với $(-1)^{x+y}$ trước khi tính biến đổi Fourier. Khi đó trọn vẹn 1 chu kỳ được xem xét bằng cách thực hiện biến đổi Fourier thông thường.*

Cần lưu ý, tập dữ liệu cần được mở rộng với các giá trị bằng không khi sử dụng biến đổi Fourier để lọc ảnh. Trong trường hợp này, cú pháp được cho như sau:

```
F=fft2(f,P,Q)
```

Với cú pháp này, `fft2` mở rộng ngõ vào với số lượng giá trị không cho phép sao cho hàm kết quả có kích thước $P \times Q$.

Phổ biên độ Fourier đạt được bằng cách sử dụng hàm abs:

```
S=abs(F)
```

Hàm `abs` tính biên độ (căn bậc hai của tổng bình phương phần thực và ảo) của từng phần tử trong mảng. Hình 3.3(b) trình bày phổ Fourier, trong đó, các điểm sáng tại các góc có biên độ lớn nhất tương ứng với 4 góc phần tư của 4 chu kỳ tiếp giáp nhau.

Phân tích phổ bằng cách biểu diễn nó như một ảnh độc lập được xem là một khía cạnh quan trọng trong miền tần số. Hàm `fftshift` trong Toolbox được sử dụng để chuyển gốc tọa độ của biến đổi đến trung tâm của hình chữ nhật hay nói đúng hơn thì nó thực hiện chức năng tương tự việc nhân $(-1)^{x+y}$. Cú pháp của hàm `fftshift` được đưa ra như sau:

```
Fc=fftshift(F)
```


https://tieulun.hopto.org

với F là biến đổi Fourier đã được tính toán bằng hàm `fft2` và Fc là biến đổi trung tâm. Hàm `fftshift` thực hiện hoán đổi các gốc phần tư của F như trong hình 3.3(c), khi đó, việc tái sắp xếp các góc phần tư này làm cho các điểm sáng đều được đặt tại vị trí trung tâm và cho phép hiển thị trọn vẹn một chu kỳ hoàn chỉnh. Hình 3.3(d) biểu diễn phổ như hình 3.3(c) nhưng được tăng cường bằng biến đổi log.

***Ví dụ 3.2**:* Thực hiện biến đổi Fourier cho ảnh và tính phổ của nó

```
clear all;
close all;
f=imread('TestDFT.tif');
F=fft2(f);
S=abs(F);
Fc=fftshift(F);
Sc=abs(Fc);
S2=log(1+abs(Fc));
figure;
subplot(2,2,1);imshow(f);
subplot(2,2,2);imshow(S);
subplot(2,2,3);imshow(Sc);
subplot(2,2,4);imshow(S2);
```

Ta có thể viết gọn hai hàm tính Fourier và chuyển đổi vị trí phổ bằng một dòng lệch gộp tương đương `fftshift(fft2(f))`, tuy nhiên, sẽ là sai hoàn toàn nếu dùng `fft2(fftshift(f))`.

Ngược với hàm `fftshift` là hàm `ifftshift` sẽ chuyển các góc phần tư về vị trí ban đầu, hàm này có cú pháp như sau:

```
F=ifftshift(Fc)
```

Cuối cùng, để biến đổi Fourier ngược, hàm `ifft` được sử dụng với cú pháp như sau:

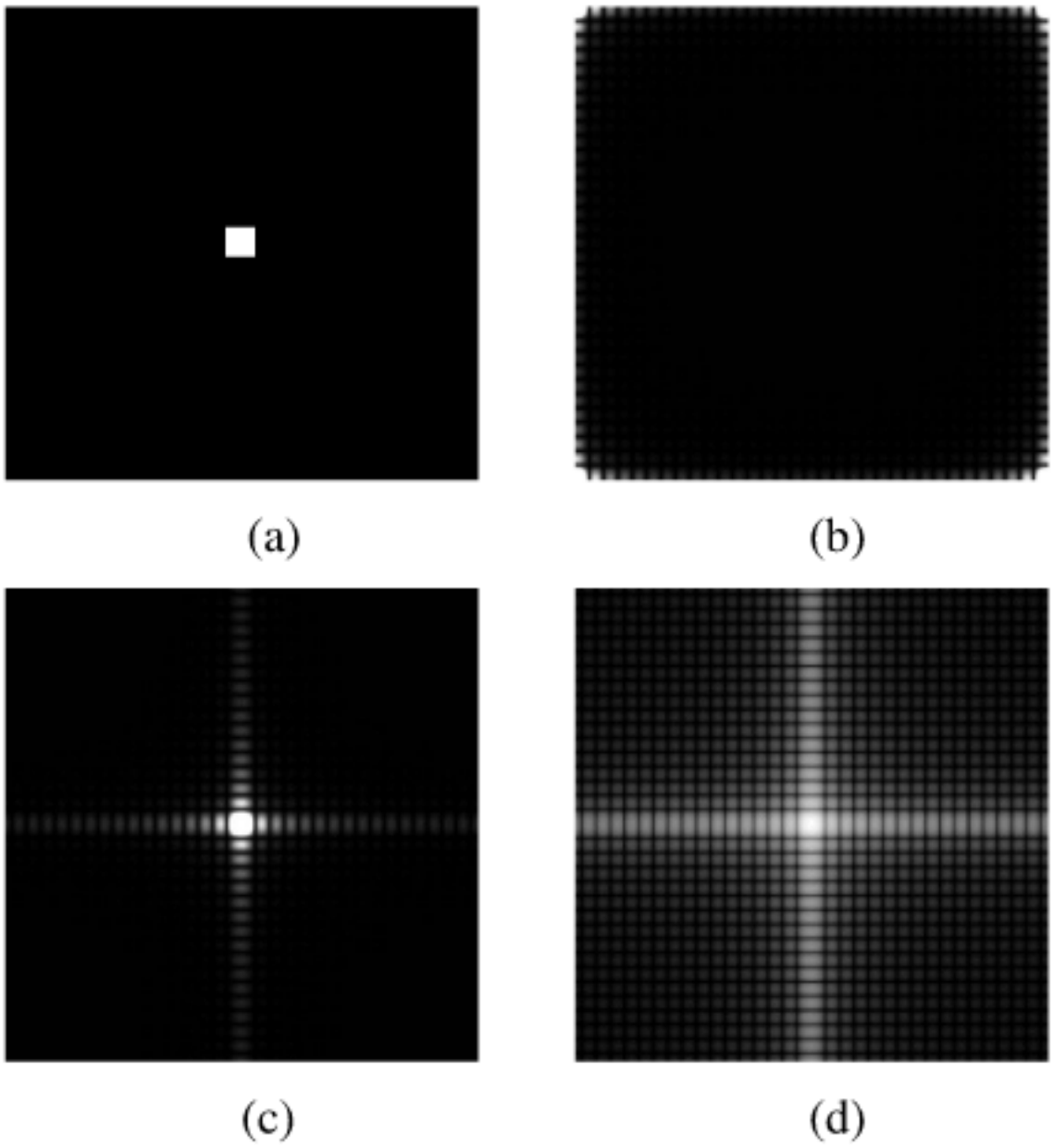
```
f=ifft2(F)
```

Với F là biến đổi Fourier sau khi đã chuyển các góc phần tư về vị trí ban đầu, f là ảnh trên miền không gian. Nếu ngõ vào tính toán F là số thực thì tương ứng ngõ ra cũng là dạng số thực. Tuy nhiên, trong thực tế, hàm `ifft2` thường có một vài thành phần ảo rất nhỏ là kết quả từ việc làm tròn vốn là đặc trưng trong tính toán dấu chấm động. Do đó, tốt nhất

https://tieulun.hopto.org

là ta sẽ lấy phần thực của kết quả sau cùng hay có thể viết gọn trong cùng một dòng lệnh như sau

```
f=real(ifft2(F))
```



Hình 3.3. *Biến đổi Fourier cho ảnh và phổ của nó:*

(a). Ảnh gốc

(b) Phổ Fourier

(c) Phổ được định vị trí trung tâm

(d) Phổ được tăng cường để hiển thị bằng biến đổi log

Do việc tính FFT trong MATLAB giả định gốc tọa độ của hàm truyền tại vị trí góc trái phía trên của hình chữ nhật tần số, vì vậy, việc tính toán khoảng cách cần phải chú ý. Trong trường hợp này, gốc tọa độ cần phải được sắp xếp lại sao cho góc nằm ngay tại trung tâm bằng cách sử dụng hàm `fftshift` như đã trình bày trước đó. Để cung cấp một sơ đồ lưới dùng tính toán khoảng cách và cho các ứng dụng tương tự, hàm `dftuv` được viết như sau:

```
function [U,V]=dftuv(m,n)
u=0:(m-1);
v=0:(n-1);
idx=find(u>m/2);
u(idx)=u(idx)-m;
idy=find(v>n/2);
v(idy)=v(idy)-n;
[V,U]=meshgrid(v,u);
```


https://tieulun.hopto.org

```
end
```

Để minh họa, ta tính bình phương khoảng cách từ mỗi điểm trong hình chữ nhật có kích thước 8 × 5 đến gốc:

```
[U,V]=dftuv(8,5);
D=U.^2+V.^2
```

D =

0	1	4	4	1
1	2	5	5	2
4	5	8	8	5
9	10	13	13	10
16	17	20	20	17
9	10	13	13	10
4	5	8	8	5
1	2	5	5	2

Giá trị khoảng cách bằng không tại góc trên bên trái và các giá trị lớn nhất tại trung tâm của hình chữ nhật được giải thích như ở hình 3.2 (a). Hiển nhiên, ta có thể dùng hàm `fftshift` để đạt được các giá trị khoảng cách với mối liên hệ đến trung tâm của hình chữ nhật dùng biểu diễn tần số (frequency retangle).

```
fftshift(D)
ans =
```

20	17	16	17	20
13	10	9	10	13
8	5	4	5	8
5	2	1	2	5
4	1	0	1	4
5	2	1	2	5
8	5	4	5	8
13	10	9	10	13

3.2. BIẾN ĐỔI WAVELET

3.2.1. Biến đổi Fourier thời gian ngắn

Hạn chế của biến đổi Fourier trong việc chỉ đưa ra thông tin tần số chung của tín hiệu có thể được giải quyết bằng biến đổi Fourier thời gian

https://tieulun.hopto.org

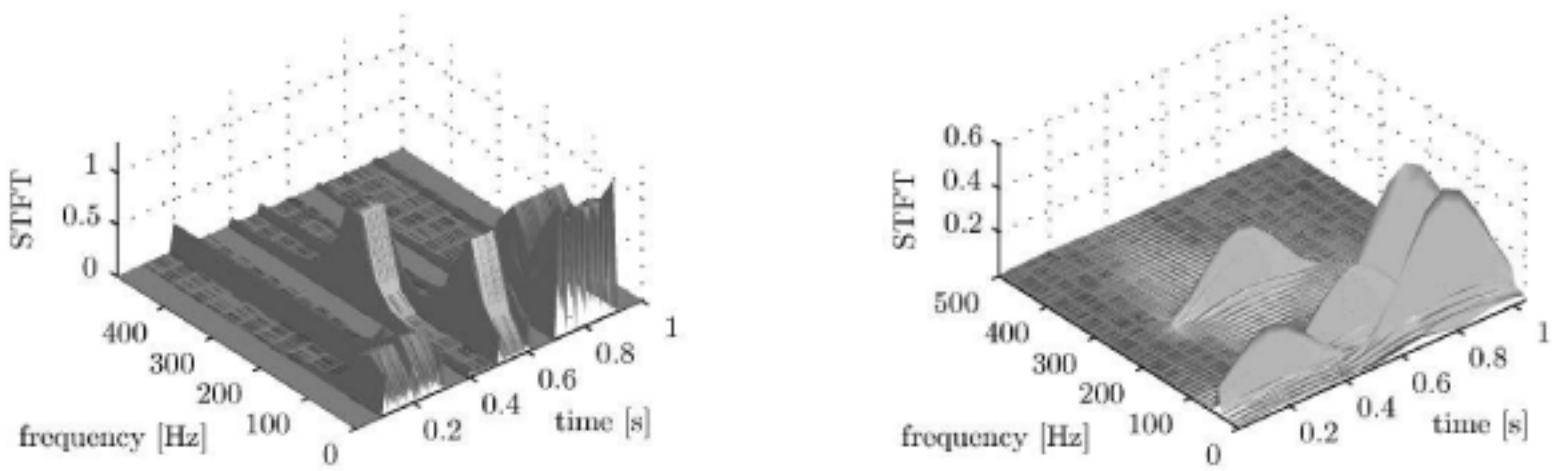
ngắn STFT. Biến đổi STFT cho phép khôi phục lại cả thông tin về mặt tần số và thời gian của tín hiệu bằng cách tính toán biến đổi Fourier của một phần tín hiệu được giới hạn trong một cửa sổ có kích thước cố định, và khi đó, cửa sổ này được dịch chuyển liên tiếp trên trục thời gian. Nói cách khác, tín hiệu $x(t)$ được chia nhỏ bởi các cửa sổ $g(t)$ với tâm của nó được đặt tại thời điểm τ. Bằng cách thực hiện biến đổi Fourier cho từng cửa sổ nhỏ, thông tin về tần số trong từng khoảng nhỏ thời gian của tín hiệu được cung cấp.

Biến đổi STFT được định nghĩa như trong biểu thức (3.26). Ở đây, dấu * biểu diễn toán tử liên hợp phức. Có thể được nhìn nhận đơn giản là biến đổi STFT chỉ đơn giản là biến đổi Fourier của tín hiệu $x(t)$ nhân với cửa sổ $g(t)$:

$$X_{STFT}(\tau, t) = \int_{-\infty}^{\infty} x(t) g^*(t-\tau) e^{-j2\pi f t} dt \tag{3.26}$$

Hiệu năng của biến đổi STFT phụ thuộc vào cửa sổ được chọn $g(t)$. Kích thước cửa sổ nhỏ cho phép thực hiện biến đổi với độ phân giải cao, nhưng bù lại các tần số khác nhau không được phân biệt rõ ràng. Điều này có thể được nhìn nhận như trong hình 3.4(a) với kích thước cửa sổ là 0.03s. Còn với kích thước của sổ dài thì cho độ phân giải kém hơn, nhưng độ phân giải tần số thì tốt hơn, như trong hình 3.4(b) với kích thước cửa sổ là 0.6s. Do đó, không thể thu kết quả phân tích với độ phân giải tốt ở cả mặt tần số lẫn thời gian. Điều này được xem xét thông qua bất đẳng thức Heisenberg: tích của độ phân giải tần số và thời gian là hằng số và việc nó được giới hạn theo như biểu thức:

$$\Delta t \Delta f \geq \frac{1}{4\pi} \tag{3.27}$$



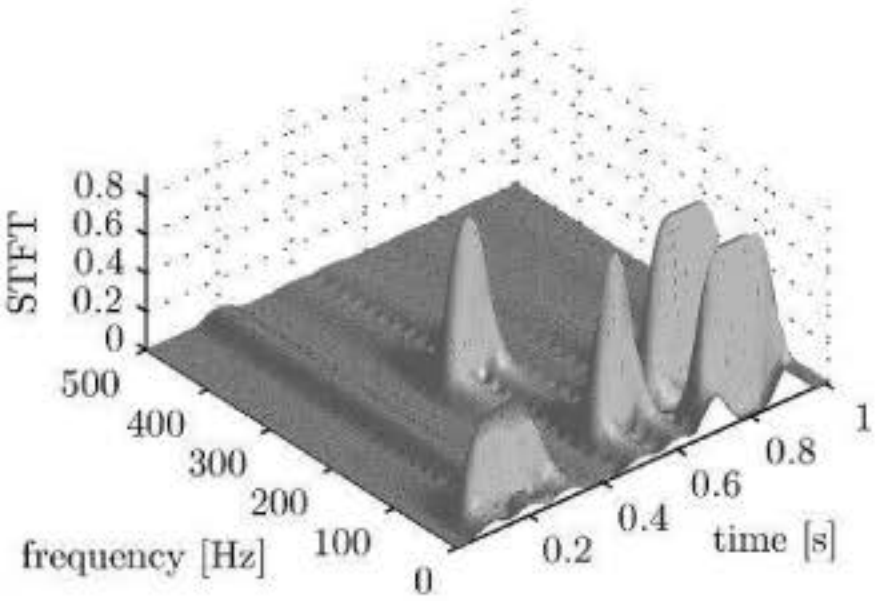
(a) (b)

Hình 3.4. *Phân tích STFT dùng cửa sổ với các kích thước khác nhau*

Sự cân bằng giữa độ phân giải thời gian và tần số được trình bày trong hình 3.5 với kích thước cửa sổ là 0.15s. Với kích thước này, biến

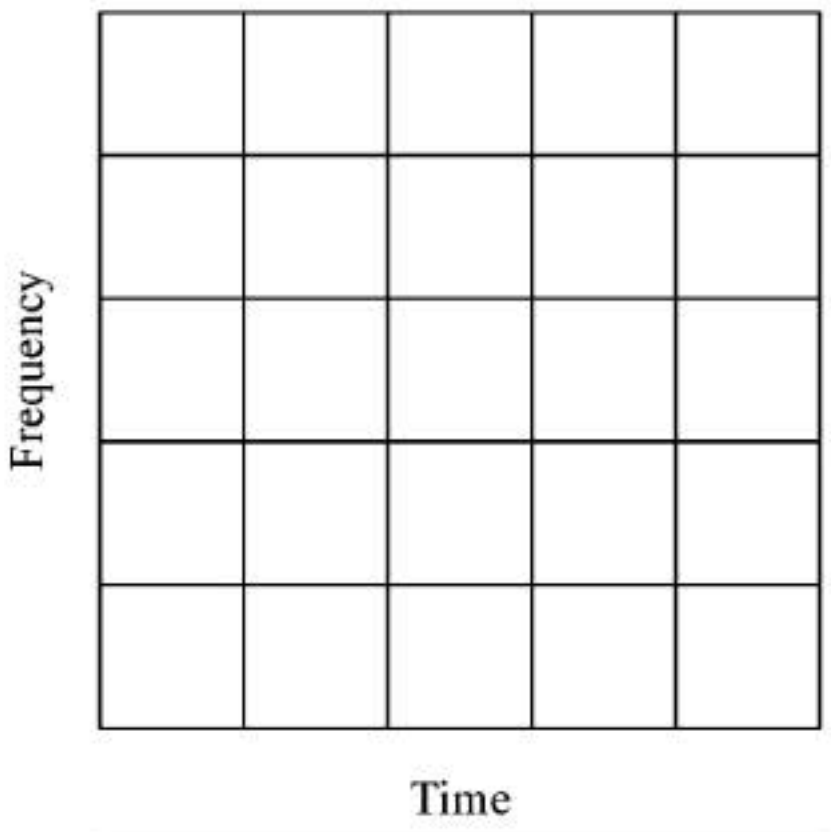
https://tieulun.hopto.org

đổi STFT cho phép biểu diễn độ phân giải thời gian và tần số một cách hợp lý hơn.



Hình 3.5. *Phân tích STFT với kích thước cửa sổ phù hợp.*

Việc tìm ra kích thước cửa sổ ảnh hưởng trực tiếp đến chất lượng của phép phân tích STFT với kích thước cửa sổ là cố định, nghĩa là Δt và Δf là hằng số. Khi đó mặt phẳng thời gian – tần số được chia ra làm nhiều khối có kích thước bằng nhau như trong hình 3.6. Các thành phần tần số thấp thường có chu kỳ thời gian dài, do đó yêu cầu phân tích với độ phân giải tần số cao. Trong khi đó, các thành phần tần số cao thường xuất hiện trong khoảng thời gian ngắn, do đó cần phải thực hiện với độ phân giải thời gian cao.



Hình 3.6. *Không gian phân tích thời gian – tần số cố định*

Sự khác nhau cơ bản giữa biến đổi Wavelet (WT) và STFT, trước nhất là độ rộng của cửa sổ có thể được thay đổi trong biến đổi Wavelet như

https://tieulun.hopto.org

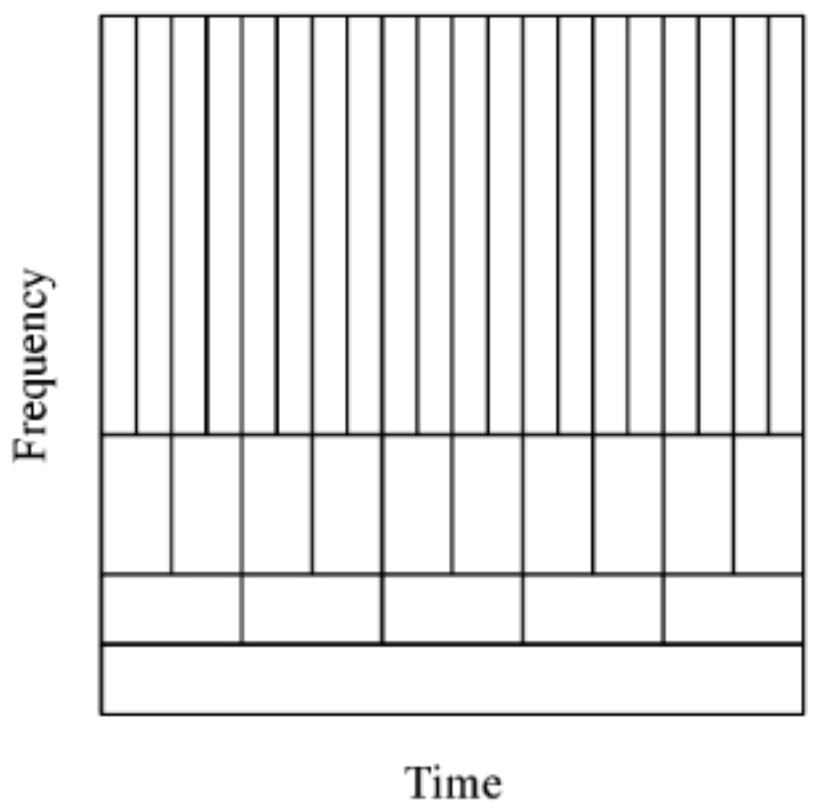
là một hàm dùng phân tích tần số. Thứ hai, hàm phân tích của biến đổi Wavelet có thể được chọn sao cho phù hợp với các dạng tín hiệu khác nhau.

3.2.2. Phân tích Wavelet

Việc phân tích tín hiệu sử dụng biến đổi Fourier và biến đổi STFT không cho ra kết quả tối ưu trong một số trường hợp, do đó, chỉ có thể đạt được kết quả tốt hơn bằng phương pháp phân tích Wavelet. Một ưu điểm có thể nhận thấy trong phương pháp này là khả năng phân tích cục bộ. Phân tích Wavelet cho phép xem xét tín hiệu dưới nhiều khía cạnh khác nhau khi so sánh với các phép phân tích khác. Khi so sánh với biến đổi STFT, phân tích Wavelet thực hiện dựa trên phương pháp phân tích đa phân giải.

3.2.2.1. Phân tích đa phân giải

Vấn đề độ phân giải trong miền thời gian – tần số có thể được giải quyết bằng phép phân tích STFT với các cửa sổ có kích thước cố định. Bằng cách sử dụng phương pháp tiếp cận được gọi là phân tích đa phân giải (Multiresolution Analysis – MRA), nó cho phép phân tích tín hiệu tại nhiều tần số khác nhau với nhiều mức phân giải khác nhau. Sự thay đổi trong độ phân giải được trình bày dưới dạng biểu đồ như trong hình 3.7.



Hình 3.7. *Không gian phân tích thời gian – tần số đa phân giải*

Trong hình 3.7, giả sử rằng các tần số thấp ứng với toàn bộ khoảng thời gian của tín hiệu, nơi mà các tần số cao xuất hiện rất ít trong khoảng thời gian rất ngắn. Đây là dạng thường xảy ra trong các ứng dụng thực tế.

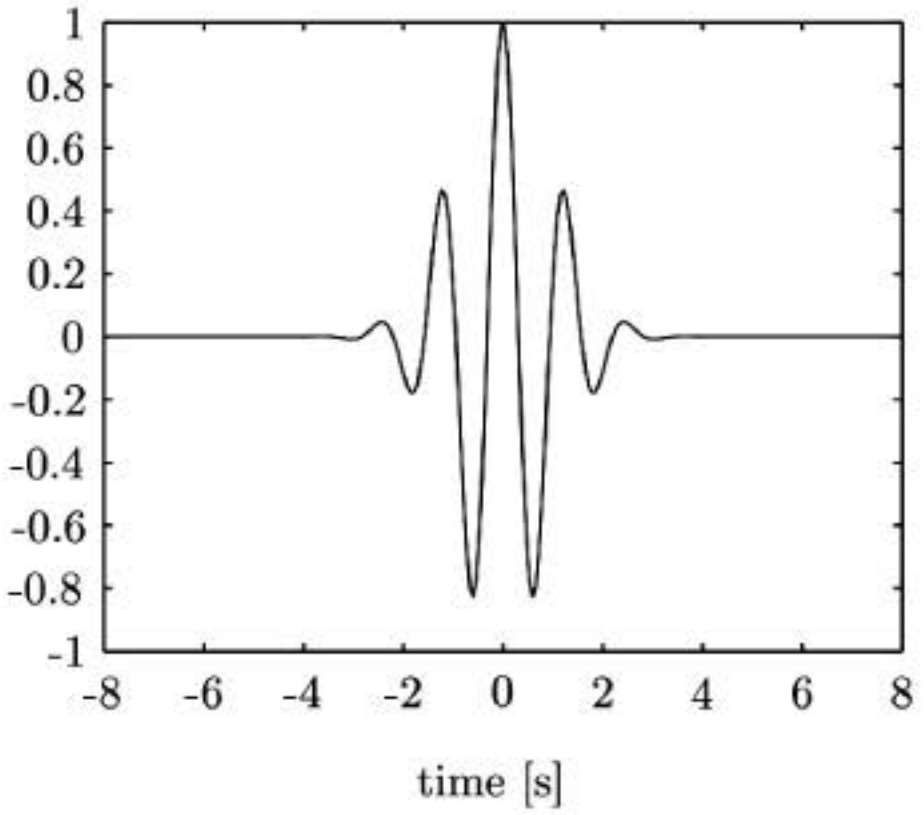
Phân tích Wavelet vốn là xét tính tương quan giữa tín hiệu và hàm Wavelet $\psi(t)$. Tính đồng dạng giữa tín hiệu và hàm phân tích Wavelet được tính toán độc lập trong nhiều khoảng thời gian khác nhau, và kết

https://tieulun.hopto.org

quả được trình bày trong không gian hai chiều. Hàm phân tích Wavelet $\psi(t)$ cũng được xem xét như là hàm Wavelet mẹ.

3.2.2.2. Wavelet

Khi so sánh với biến đổi Fourier, hàm phân tích trong biến đổi Wavelet có thể được chọn tùy ý, không nhất thiết phải là dạng hàm sine. Hàm Wavelet $\psi(t)$ được hiểu là một dạng sóng nhỏ, hay là một dạng dao động dùng để phân biệt các tần số khác nhau trong tín hiệu ngõ vào. Wavelet chứa thông tin cả về dạng phân tích và kích thước cửa sổ. Hình 3.8 trình bày ví dụ về một Wavelet, được biết đến với tên Wavelet Morlet. Bên cạnh đó nhiều dạng của hàm Wavelet được phát triển để có thể trích xuất đặc trưng tùy theo dạng tín hiệu ngõ vào.



Hình 3.8. *Wavelet họ Morlet*

Một hàm phân tích $\psi(t)$ được phân loại là một Wavelet nếu thỏa mãn các điều kiện toán học sau:

i. Một sóng Wavelet phải có năng lượng hữu hạn

$$E = \int_{-\infty}^{\infty} |\psi(t)|^2 dt < \infty \tag{3.28}$$

Mức năng lượng E bằng với tích phân của bình phương độ lớn của hàm phân tích $\psi(t)$ và phải có giá trị hữu hạn.

ii. Nếu $\Psi(f)$ là biến đổi Fourier của sóng Wavelet $\psi(t)$ thì

$$C_\psi = \int_{-\infty}^{\infty} \frac{\left|\hat{\psi}(f)\right|^2 dt}{f} < \infty \tag{3.29}$$

https://tieulun.hopto.org

Điều kiện này chỉ ra rằng sóng Wavelet không có thành phần tần số bằng không ($\Psi(0) = 0$), nghĩa là trung bình của các sóng Wavelet phải bằng không. Điều kiện này được biết đến như là hằng số chấp nhận (admissibility constant). Giá trị C_ψ phụ thuộc vào hàm Wavelet được chọn.

iii. Biến đổi Fourier của các Wavelet phức tạp phải là số thực và triệt tiêu các tần số âm.

3.2.2.3. Biến đổi Wavelet liên tục

Hàm biến đổi Wavelet liên tục được định nghĩa:

$$X_{WT}(\tau, s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t)\psi^*\left(\frac{t-\tau}{s}\right) dt \tag{3.30}$$

Tín hiệu đã được biến đổi $X_{WT(\tau,s)}$ là hàm với hai biến: độ dịch chuyển τ và tham số tỷ lệ s. Hàm Wavelet mẹ được ký hiệu ψ, ký hiệu * ám chỉ toán tử liên hợp phức được dùng trong trường hợp của Wavelet phức tạp. Năng lượng của tín hiệu được chuẩn hóa tại từng tỷ lệ bằng cách chia nhỏ các hệ số Wavelet cho $1/\sqrt{|s|}$. Điều này bảo đảm các sóng Wavelet đều có cùng mức năng lượng tại cùng một hệ số tỷ lệ.

Hàm Wavelet mẹ có thể co hay giãn bằng cách thay đổi hệ số tỷ lệ. Biến tỷ lệ s không những làm thay đổi tần số trung tâm của sóng Wavelet f_c mà còn tác động lên độ rộng của cửa sổ. Do đó, hệ số tỷ lệ dùng thay cho tần số trong việc biểu diễn các kết quả phân tích Wavelet. Thông số dịch chuyển τ đánh dấu vị trí của sóng Wavelet trong miền thời gian và bằng cách thay đổi τ sóng Wavelet có thể được dịch theo tín hiệu. Với hệ số tỷ lệ s cố định và thay đổi biến τ thì các hàng trong không gian thời gian – tỷ lệ được tính, trong khi thay đổi hệ số tỷ lệ và giữ nguyên hằng số dịch chuyển τ thì các cột trong không gian được tính toán. Các phần tử $X_{WT}(\tau, s)$ được gọi là các hệ số Wavelet.

Biến đổi Wavelet cũng có biến đổi ngược giống như trong trường hợp của biến đổi Fourier và STFT. Biến đổi ngược Wavelet liên tục (ICWT) được xác định như sau:

$$x(t) = \frac{1}{C_\psi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} X_{WT}(\tau, s) \frac{1}{s^2} \psi\left(\frac{t-\tau}{s}\right) d\tau ds \tag{3.31}$$

Lưu ý: hằng số trong biểu thức (3.31) phải thỏa mãn điều kiện thứ hai như đã nói ở trên (biểu thức (3.30)).

Hàm Wavelet có tần số trung tâm f_c tại từng giá trị hệ số tỷ lệ và nó tỷ lệ nghịch với tần số. Hệ số tỷ lệ lớn tương ứng với tần số thấp, cung

https://tieulun.hopto.org

cấp thông tin của toàn bộ tín hiệu, trong khi đó, các giá trị tỷ lệ nhỏ ứng với tần số cao sẽ cung cấp các thông tin chi tiết của tín hiệu.

Với biến đổi Wavelet, bất đẳng thức Heisenberg vẫn còn ý nghĩa, tích tần số - thời gian $\Delta t \Delta f$ vẫn luôn là hằng số. Giảm hệ số tỷ lệ s, ứng với kích thước cửa sổ nhỏ, sẽ làm tăng độ phân giải thời gian Δt và giảm độ phân giải tần số Δf. Điều này cũng có nghĩa là độ phân giải tần số tỷ lệ với tần số f, cụ thể, phân tích Wavelet có độ phân giải tần số gần như là cố định. Wavelet Morlet, trong hình 3.8, đạt được bằng cách sử dụng cửa sổ Gaussian, với f_c là tần số trung tâm và f_b là thông số băng thông.

$$\psi(t) = g(t)e^{-j2\pi f_c t}, g(t) = \sqrt{\pi f_b}e^{t^2/f_b} \tag{3.32}$$

Tần số trung tâm f_c và độ rộng băng thông f_b của Wavelet là các thông số có thể được điều chỉnh. Chẳng hạn, họ Wavelet Morlet, thông số tỷ lệ và tần số có mối quan hệ như sau:

$$f = \frac{f_c}{s} \tag{3.33}$$

Việc tính toán biến đổi Wavelet liên tục CWT thường được thực hiện bằng cách lấy các giá trị rời rạc ứng với các hệ số tỷ lệ s và độ dịch chuyển τ. Các hệ số Wavelet được gọi là chuỗi Wavelet. Ứng với từng mục đích phân tích, quá trình rời rạc hóa có thể được thực hiện bất kỳ, tuy nhiên, nếu việc tái cấu trúc tín hiệu được yêu cầu, các điều kiện của sóng Wavelet trở nên rất quan trọng. Chuỗi Wavelet có thể được tính như sau:

$$\begin{aligned} X_{WT_{m,n}} &= \int_{-\infty}^{\infty} x(t)\psi_{m,n}(t)dt \\ \psi_{m,n} &= s_0^{-m/2}\psi(s_0^{-m}t - n\tau_0) \end{aligned} \tag{3.34}$$

Số nguyên m và n điều chỉnh độ dịch chuyển và độ giản nở của sóng Wavelet. Ứng với lược đồ nhị nguyên (dyadic grid), $s_0 = 0$ và $\tau_0 = 1$. Các sóng Wavelet này được lựa chọn sao cho trực chuẩn, nghĩa là, chúng trực giao với nhau và được chuẩn hóa để có mức năng lượng đơn vị. Việc lựa chọn này cho phép xây dựng lại tín hiệu gốc thông qua biểu thức sau:

$$x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} X_{WT_{m,n}} \psi_{m,n}(t) \tag{3.35}$$

Với tín hiệu như trong hình 3.9(a), thực hiện biến đổi Wavelet liên tục bằng hàm phân tích Morlet, sau đó được rời rạc và biểu diễn như

https://tieulun.hopto.org

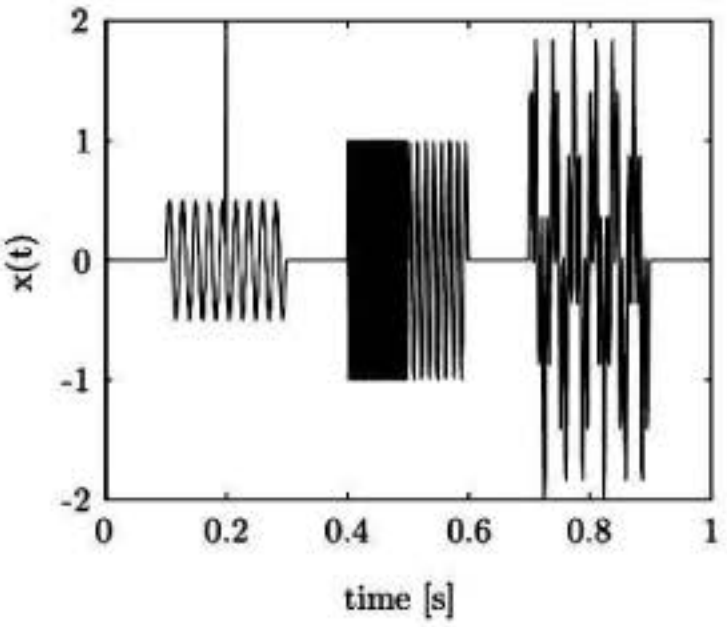
trong hình 3.9(b) và (c) dưới dạng lược đồ không gian tỷ lệ - thời gian và dưới dạng đường viền (contour plot). Trong cả hai lược đồ, hệ số tỷ lệ lớn tương ứng với các tần số thấp và ngược lại, trong khi đó, có được độ phân giải tần số tốt với những thành phần tần số cao (hệ số tỷ lệ nhỏ) và độ phân giải thời gian tốt với các thành phần tần số thấp (hệ số tỷ lệ lớn). Các tần số khác biệt được phát hiện tại các thời điểm chính xác (tại thời điểm 0.2 s như trong hình 3.9(c)).

Hình 3.9. *Biến đổi Wavelet liên tục*

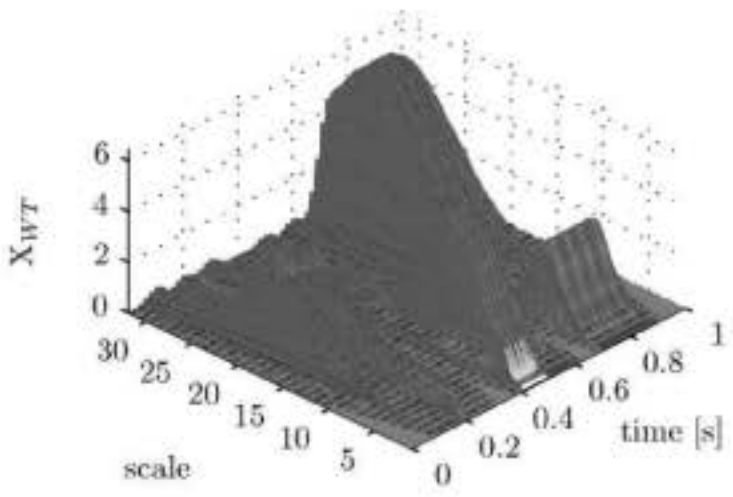
(a) Tín hiệu $x(t)$

(b) Surface plot

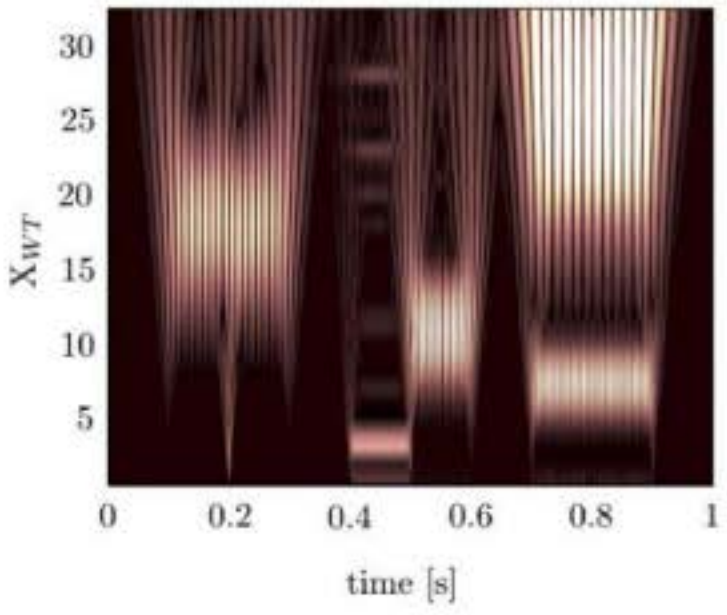
(c) Contour plot



(a)



(b)



(c)

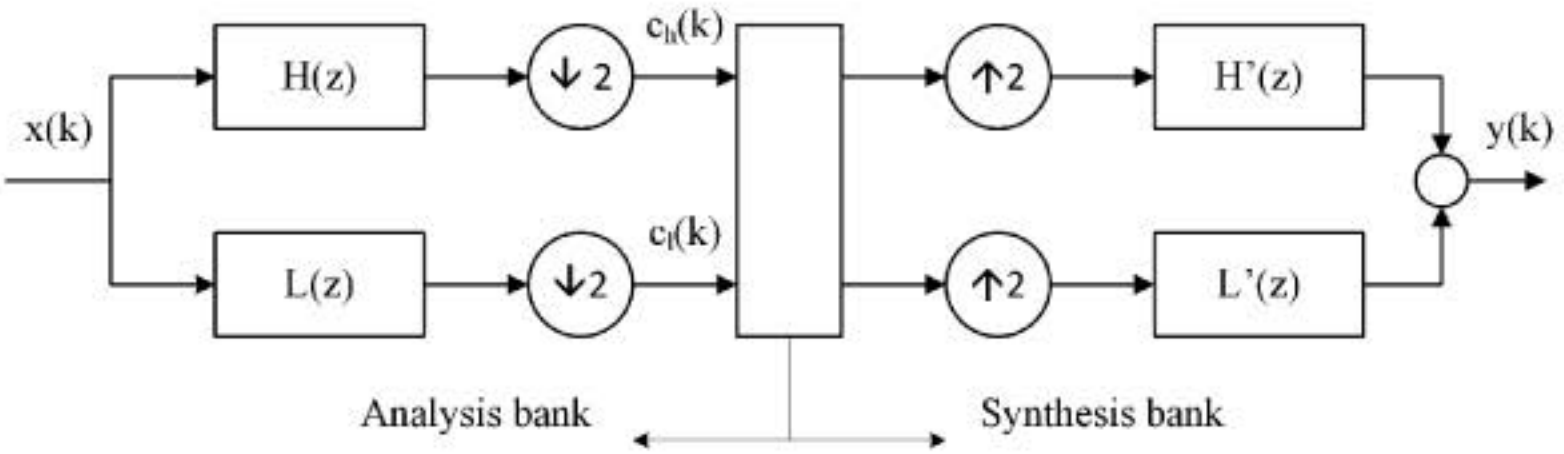
3.2.2.4. Biến đổi Wavelet rời rạc

3.2.2.4.1. Giàn lọc

Một giàn lọc (filter bank) bao gồm các bộ lọc con giúp phân chia tín hiệu thành các băng con với nhiều tần số. Một ví dụ về giàn lọc hai kênh như trong hình 3.10. Tín hiệu rời rạc theo thời gian $x(k)$ đưa vào phân tích và được lọc bằng các bộ lọc $L(z)$ và $H(z)$, chúng phân chia nội dung tần số của tín hiệu thành các dải tần số có độ rộng tương đương nhau. Các bộ lọc $L(z)$ và $H(z)$ tương ứng với bộ lọc thông thấp và thông cao. Ngõ ra của chúng là thông tin về tần số của tín hiệu với nửa trên và

https://tieulun.hopto.org

nửa dưới tần số cắt, nhưng có số lượng mẫu ở mỗi ngõ ra đều bằng với tín hiệu gốc. Qua đây có thể thấy rằng, ngõ ra chứa lượng thông tin về mặt tần số là tương đương với tín hiệu gốc, nhưng lại gấp đôi số lượng mẫu. Do đó, kỹ thuật giảm mẫu xuống một nửa, ký hiệu ↓2, được áp dụng cho các ngõ ra của bộ lọc trong quá trình phân tích.



Hình 3.10. *Giàn lọc hai kênh*

Việc tái cấu trúc tín hiệu gốc cũng được thực hiện bằng cách sử dụng các bộ lọc, gọi là giàn lọc tổng hợp, khi đó, chúng cần được tăng mẫu (↑2) trước khi được đưa qua các bộ lọc là $L'(z)$ và $H'(z)$. Các bộ lọc này dựa trên các bộ lọc trong tập phân tích. Ngõ ra của các bộ lọc sẽ được kết hợp và cấu thành nên tín hiệu ban đầu là $y(k)$.

Các tín hiệu ngõ ra khác nhau của giàn lọc phân tích được gọi là các băng con và kỹ thuật dựa trên việc sử dụng giàn lọc cũng được gọi là mã hóa băng con.

3.2.2.4.2. Kỹ thuật tăng và giảm mẫu

Các bộ lọc thông thấp $L(z)$ và $H(z)$ phân chia nội dung tần số của tín hiệu làm hai phần bằng nhau, và cần được giảm mẫu để tránh hiện tượng dư thừa. Nếu một nửa số mẫu của tín hiệu đã được lọc là $c_l(k)$ và $c_h(k)$ được cắt giảm, nó vẫn có thể được dùng để xây dựng lại tín hiệu gốc như ban đầu là $x(k)$. Ký hiệu giảm mẫu (↓2) chỉ ra những mẫu chẵn sẽ được loại bỏ. Trong miền tần số, ảnh hưởng của việc loại bỏ thông tin có thể dẫn đến hiện tượng méo tín hiệu. Nếu tuân theo định lý lấy mẫu Shannon sẽ không có thông tin nào bị mất trong toàn bộ quá trình lấy mẫu. Định lý Shannon phát biểu rằng giảm mẫu của tín hiệu đã được lấy mẫu với hệ số M sẽ tạo ra tín hiệu mà toàn bộ phổ của nó vẫn có thể được tính toán bằng cách chia nhỏ phổ của tín hiệu thành M băng tần tương đương và sau đó lấy tổng của chúng.

Trong giàn lọc tổng hợp, các tín hiệu sẽ được tăng mẫu trước khi lọc. Việc tăng số lượng mẫu lên gấp đôi (↑2) được thực hiện bằng cách

https://tieulun.hopto.org

thêm vào các mẫu giá trị bằng không giữa các mẫu của tín hiệu gốc. Lưu ý, việc giảm mẫu tín hiệu và sau đó tăng trở lại sẽ không trả về tín hiệu như ban đầu. Chỉ có thể thu được tín hiệu bằng cách tăng mẫu khi chèn các mẫu giá trị không và sau đó giảm mẫu một lần nữa như trong biểu thức (3.36).

$$x = \begin{bmatrix} \cdot \\ x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ \cdot \end{bmatrix} (\downarrow 2) = \begin{bmatrix} \cdot \\ x(0) \\ x(2) \\ x(4) \\ \cdot \end{bmatrix} (\uparrow 2)(\downarrow 2) = \begin{bmatrix} \cdot \\ x(0) \\ 0 \\ x(2) \\ 0 \\ x(4) \\ \cdot \end{bmatrix} \tag{3.36}$$

3.2.2.4.3. Tái cấu trúc tín hiệu

Để xây dựng lại tín hiệu ban đầu, giàn lọc phải trực giao với nhau. Do đó, một vài tiêu chuẩn thiết kế dành cho các giàn lọc trong quá trình phân tích và tổng hợp cần phải được đưa ra để hạn chế gây méo và suy hao tín hiệu cũng như bảo đảm việc khôi phục tín hiệu được tối ưu.

Trong giàn lọc hai kênh như trong hình 3.10, bộ lọc $L(z)$ và $H(z)$ chia tín hiệu thành hai băng con, nghĩa là, các bộ lọc này tương ứng với lọc thông thấp và thông cao. Nếu các bộ lọc có dốc đứng tốt (brick-wall) thì quá trình giảm mẫu không mất nhiều thông tin. Tuy nhiên, các bộ lọc lý tưởng không thể thực hiện được trong thực tế, vì vậy mà luôn tồn tại băng tần chuyển tiếp. Bên cạnh méo, điều này cũng dẫn đến sự suy hao biên độ và pha tại mỗi kênh.

Ứng với giàn lọc hai kênh trong hình 3.10, méo có thể được khắc phục bằng giàn lọc tổng hợp được cho như sau:

$$L'(z) = H(-z) \tag{3.37}$$

$$H'(z) = -L(-z) \tag{3.38}$$

Để hạn chế suy hao, bộ lọc nhân $P_0(z) = L'(z)L(z)$ được đề xuất. Suy hao có thể tránh nếu:

$$P_0(z) - P_0(-z) = 2z^{-N} \tag{3.39}$$

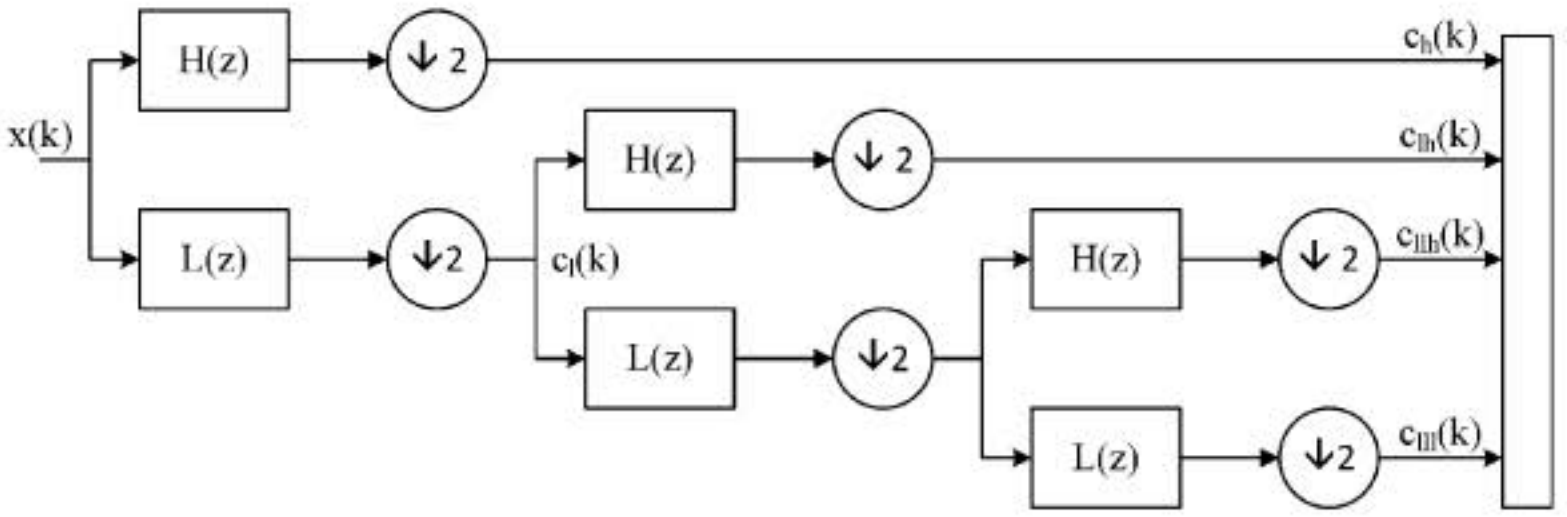
Trong đó N là khoảng trễ của toàn bộ giàn lọc. Tổng quát, bộ lọc bậc N^{th} sẽ tạo ra độ trễ N mẫu. Giàn lọc dùng cho việc xây dựng lại tín hiệu cần được thiết kế theo hai bước:

1. Thiết kế một bộ lọc thông thấp P_0 thỏa mãn biểu thức (3.39)
2. Thế P_0 vào trong $L'(z)L(z)$và dùng biểu thức (3.37) và (3.38) để tính $H(z)$ và $H'(z)$.

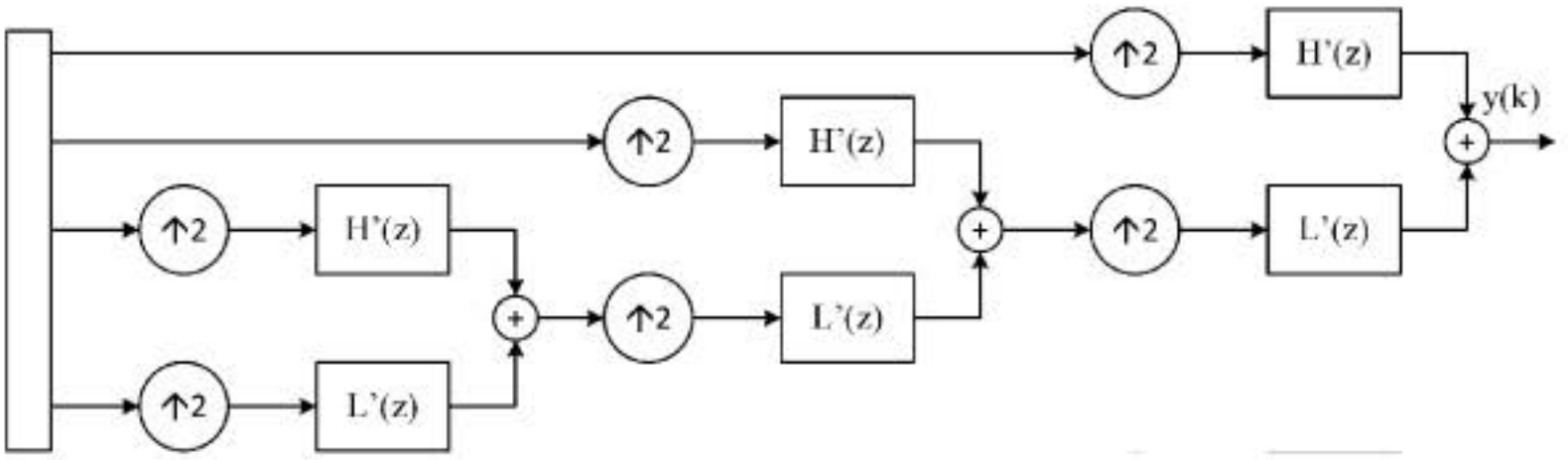
https://tieulun.hopto.org

Lưu ý, quá trình thiết kế bộ lọc P_0 với hệ số ở bước một và tìm thừa số trong bước hai có thể thực hiện bằng nhiều cách khác.

3.2.2.4.4. Các giàn lọc đa phân giải



(a) Giàn lọc dùng trong quá trình phân tích



(b) Giàn lọc dùng trong quá trình tổng hợp

Hình 3.11. *Giàn lọc với ba mức phân tích*

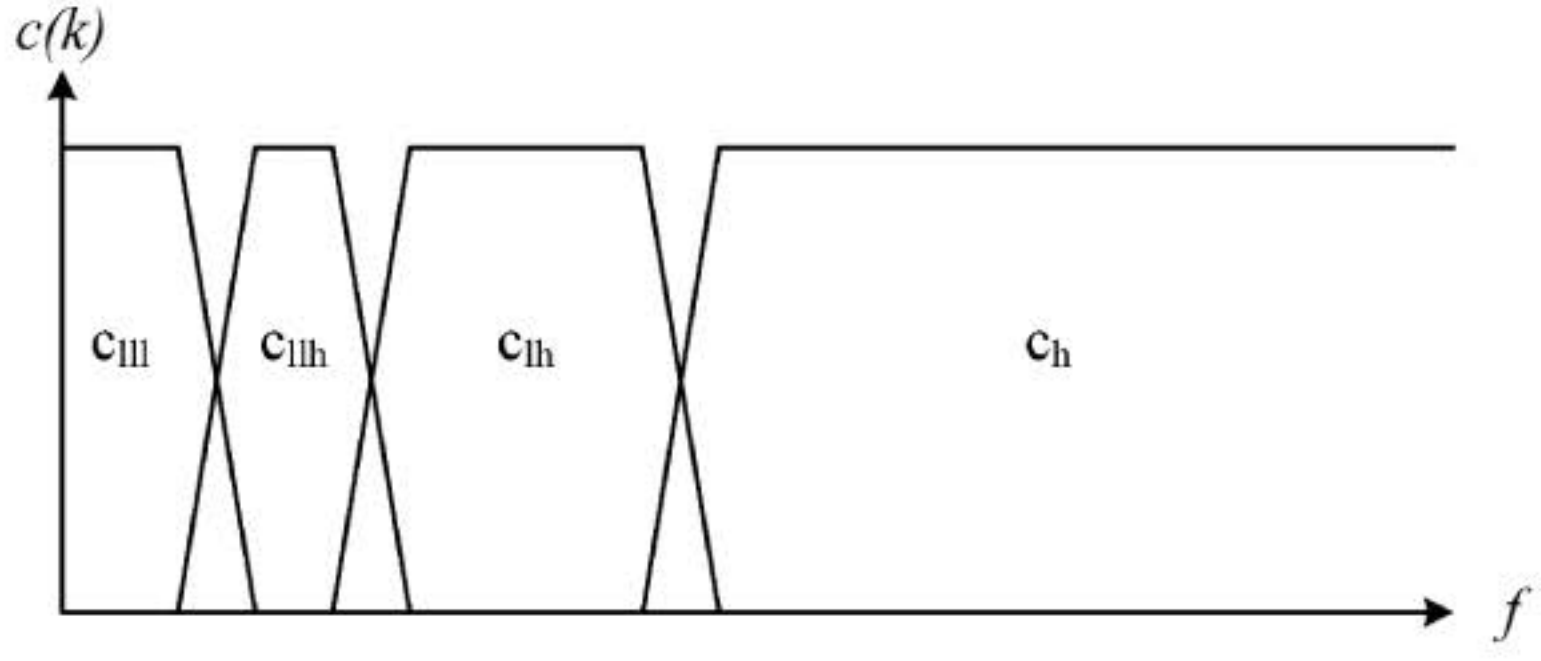
Biến đổi Wavelet liên tục CWT thực hiện phân tích đa phân giải cho một tín hiệu tại nhiều tần số khác nhau. Phân tích đa phân giải có thể đạt được bằng cách sử dụng các giàn lọc với biến đổi Wavelet rời rạc DWT. Lưu ý là việc rời rạc hóa biến đổi CWT không đồng nghĩa với biến đổi Wavelet rời rạc. Biến đổi DWT dùng tập các bộ lọc chuyên biệt, trong khi rời rạc hóa CWT sử dụng dạng được rời rạc hóa của trục tỷ lệ.

Các bộ lọc thông thấp và thông cao sẽ trích xuất thành phần xấp xỉ và chi tiết của tín hiệu $x(k)$. Hình 3.11 trình bày giàn lọc 3 mức dùng trong quá trình phân tích và tổng hợp tín hiệu. Giàn lọc có thể được mở rộng với số mức phân tích bất kỳ, tùy thuộc vào độ phân giải mong muốn. Các hệ số $c_l(k)$ trong hình 3.11(a) biểu diễn mức nửa dưới dải tần của tín hiệu $x(k)$. Độ phân giải thời gian được giảm đi một nửa, nghĩa là, chỉ một nửa số mẫu được biểu diễn trong thành phần $c_l(k)$.

Tại mức phân tích thứ hai, ngõ ra của $L(z)$ và $H(z)$ gấp đôi độ phân giải thời gian và giảm đi nội dung thông tin tần số, nghĩa là, độ rộng của cửa sổ được gấp đôi. Sau mỗi mức, ngõ ra của bộ lọc thông cao chứa

https://tieulun.hopto.org

nội dung về tần số của nửa trên của ngõ ra bộ lọc thông thấp ở mức trước đó. Khi đó, ta xem bộ lọc thông cao này như là một bộ lọc thông dải bình thường. Độ phân giải thời gian của giàn lọc dùng trong phân tích trong hình 3.11(a) thì tương tự như độ phân giải trình bày trong hình 3.7 trước đó. Với tập đặc biệt các bộ lọc $L(z)$ và $H(z)$, cấu trúc thực hiện này được gọi là DWT, trong khi các bộ lọc gọi là bộ lọc Wavelet. Sự phân chia dải tần có thể được kham thảo trong hình 3.12 ứng với từng băng con.

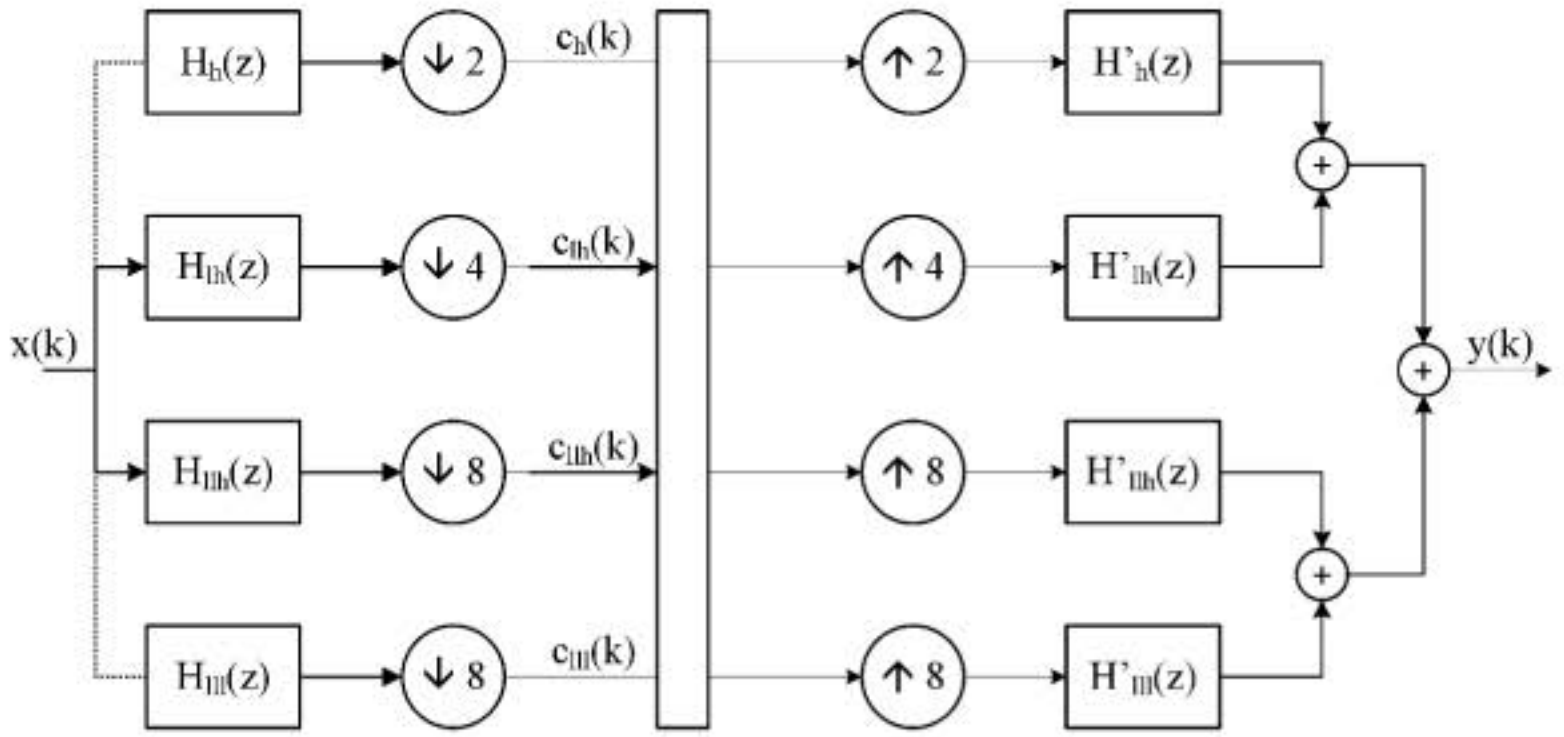


Hình 3.12. *Biểu diễn sự phân chia tần của các băng con trong hình 3.11*

Mối quan hệ giữa biến đổi Wavelet liên tục và rời rạc là không thật sự rõ ràng. Các sóng Wavelet trong CWT có một tần số trung tâm và giữ vai trò như là một bộ lọc thông dải trong phép tích chập giữa hàm Wavelet và tín hiệu $x(t)$. Một chuỗi các bộ lọc thông thấp, bộ tăng và giảm mẫu cũng được xem như bộ lọc thông dải.

Để dễ dàng so sánh giữa DWT và CWT, tập bộ lọc trong hình 3.11 được viết lại như trong hình 3.13. Các bộ lọc có thể được hiểu như là các hàm Wavelet tại nhiều mức tỷ lệ khác nhau. Tuy nhiên, chúng không phải lúc nào cũng chính xác, nếu số lượng mức phân tích tăng và đáp ứng xung của các bộ lọc tương đương hội tụ đến một dạng sóng ổn định thì các bộ lọc $L(z)$ và $H(z)$ chính là các bộ lọc Wavelet. Các chuỗi bộ lọc sau đó trở thành dạng đã được lấy tỷ lệ của nhau. Các bộ lọc Wavelet này biểu diễn nội dung tần số của một hàm Wavelet tại một tỷ lệ nhất định. Chúng có thể được sắp xếp và phân loại thành hai dạng, trực giao và song trực giao.

https://tieulun.hopto.org



Hình 3.13. *Giàn lọc tương đương hình 3.11*

Các hàm Wavelet giới hạn, nghĩa là, các sóng ổn định, có thể được xây dựng dễ dàng từ các giàn lọc phân tích. Với nhánh tần số thấp hơn trong hình 3.11(b), chỉ bao gồm các bộ lọc thông thấp và quá trình tăng số mẫu, đáp ứng xung hội tụ đến hàm sau cùng $l(n)$ ứng với biểu thức sai lệch sau đây:

$$\phi(t) = \sum_{n=0}^{N} l(n)\phi(2t - n) \tag{3.40}$$

Hàm này được biết đến như là hàm tỷ lệ của Wavelet. Ứng với chuỗi thông dải, các đáp ứng xung hội tụ đến biểu thức sai lệch vốn được tính theo:

$$\psi(t) = \sum_{n=0}^{N} h(n)\psi(2t - n) \tag{3.41}$$

Biểu thức sau cùng của chuỗi thông dải $h(n)$ là làm Wavelet $\psi(t)$.

Các băng con với các hệ số Wavelet c_{lll} được gọi là băng con xấp xỉ cA và chứa các tần số thấp. Các băng con khác được gọi là băng con chi tiết cD chứa các thông tin chi tiết của tín hiệu và các hệ số Wavelet biểu diễn nội dung của tín hiệu với nhiều tần số khác nhau.

Ứng với mức phân tích p, tần số cao nhất được quan sát nằm trong các hệ số Wavelet băng xấp xỉ c_{lll} có thể được tính như là hàm của tần số mẫu f_s:

$$f_l = \frac{f_s}{2^{p+1}} \tag{3.42}$$

https://tieulun.hopto.org

Nội dung tần số của băng con xấp xỉ và băng con chi tiết có thể được tính như sau:

$$f_{cA} = [0, 2^{-p-1} f_s] \tag{3.43}$$

$$f_{cD_p} = [2^{-p-1} f_s, 2^{-p} f_s] \tag{3.44}$$

Kết quả phân tích Wavelet phụ thuộc nhiều vào việc lựa chọn bộ lọc Wavelet và đặc trưng của tín hiệu xem xét.

Với một tín hiệu một chiều dạng rời rạc $f[n]$, DWT được tính theo công thức sau:

$$\begin{aligned} W_\phi(j_0, k) &= \frac{1}{\sqrt{M}} \sum_n f(n) \phi_{j_0,k}(n) \\ W_\psi(j, k) &= \frac{1}{\sqrt{M}} \sum_n f(n) \psi_{j,k}(n) \quad j \geq j_0 \end{aligned} \tag{3.45}$$

Trong đó, $\phi_{j_0,k}(n)$ và $\phi_{j_0,k}(n)$ là các hàm tỷ lệ và hàm Wavelet dạng rời rạc được xác định trong khoảng $[0, M-1]$ và trực giao với nhau và j_0 là tỷ lệ khởi tạo bất kỳ. $W_\phi(j_0, k)$ biểu diễn các hệ số xấp xỉ (ứng với các tần số thấp), trong khi, $W_\psi(j, k)$ đại diện cho các hệ số chi tiết (ứng với tần số cao). Ta hoàn toàn có thể khôi phục được tín hiệu rời rạc gần đúng theo công thức sau đây:

$$\begin{aligned} f(n) = &\frac{1}{\sqrt{M}} \sum_k W_\phi(j_0, k) \phi_{j_0,k}(n) \\ &+ \frac{1}{\sqrt{M}} \sum_{j=j_0}^{\infty} \sum_k W_\psi(j, k) \psi_{j,k}(n) \end{aligned} \tag{3.46}$$

Nếu hàm Wavelet và hàm tỷ lệ được cho biết trước, các hệ số của nó sẽ được xác định theo công thức (3.45). Tuy nhiên có một phương pháp khác cũng dùng để tính các hệ số trong khi vẫn tiết kiệm được thời gian tính toán. Từ hàm tỷ lệ:

$$\phi(x) = \sum_n h_\phi(n) \sqrt{2} \phi(2x - n) \tag{3.47}$$

Bằng cách lấy tỷ lệ 2^j và dịch chuyển đi k với biến x, và đặc $m = 2k + n$, từ (3.47), ta có

https://tieulun.hopto.org

$$\phi(2^j x - k) = \sum_n h_\phi(n)\sqrt{2}\phi(2(2^j x - k) - n)$$
$$= \sum_m h_\phi(m - 2k)\sqrt{2}\phi(2^{j+1}x - m) \quad (3.48)$$

Tương tự với hàm Wavelet:

$$\psi(2^j x - k) = \sum_m h_\psi(m - 2k)\sqrt{2}\phi(2^{j+1}x - m) \quad (3.49)$$

Thế biểu thức (3.48) và (3.49) vào các biểu thức tính hệ số Wavelet rời rạc là (3.45), ta có:

$$\begin{aligned} W_\phi(j,k) &= \frac{1}{\sqrt{M}}\sum_n f(n)\phi_{j,k}(n) \\ &= \frac{1}{\sqrt{M}}\sum_n f(n)2^{j/2}\phi(2^j n - k) \\ &= \frac{1}{\sqrt{M}}\sum_n f(n)2^{j/2}\left[\sum_m h_\phi(m-2k)\sqrt{2}\phi(2^{j+1}x - m)\right] \\ &= \sum_m h_\phi(m - 2k)\left[\frac{1}{\sqrt{M}}\sum_n f(n)2^{(j+1)/2}\phi(2^{j+1}x - m)\right] \\ &= \sum_m h_\phi(m-2k)\, W_\phi(j+1,k) \end{aligned} \quad (3.50)$$

$$W_\phi(j,k) = h_\phi(-n) \star W_\phi(j+1,n)\big|_{n=2k,k\geq 0}$$

Tương tự với hàm Wavelet (ứng với các hệ số chi tiết):

$$W_\psi(j,k) = h_\psi(-n) \star W_\psi(j+1,n)\big|_{n=2k,k\geq 0} \quad (3.51)$$

Thông thường tín hiệu được dùng thuộc dạng rời rạc, chẳng hạn như ảnh số, tín hiệu gốc có thể được xem như là các hệ số xấp xỉ với bậc cho trước là J. Khi đó, $f(n) = W_\phi(J,n)$, bằng cách sử dụng công thức tính (3.49) và (3.50) cho các hệ số xấp xỉ và chi tiết ở mức phân tích kế tiếp. Thuật toán này gọi là Fast Wavelet Transform do có thể tìm được các hệ số theo từng level trực tiếp nhanh hơn so với dùng công thức (3.45).

https://tieulun.hopto.org

3.2.2.4.5. Một số hàm Wavelet thông dụng

Daubechies

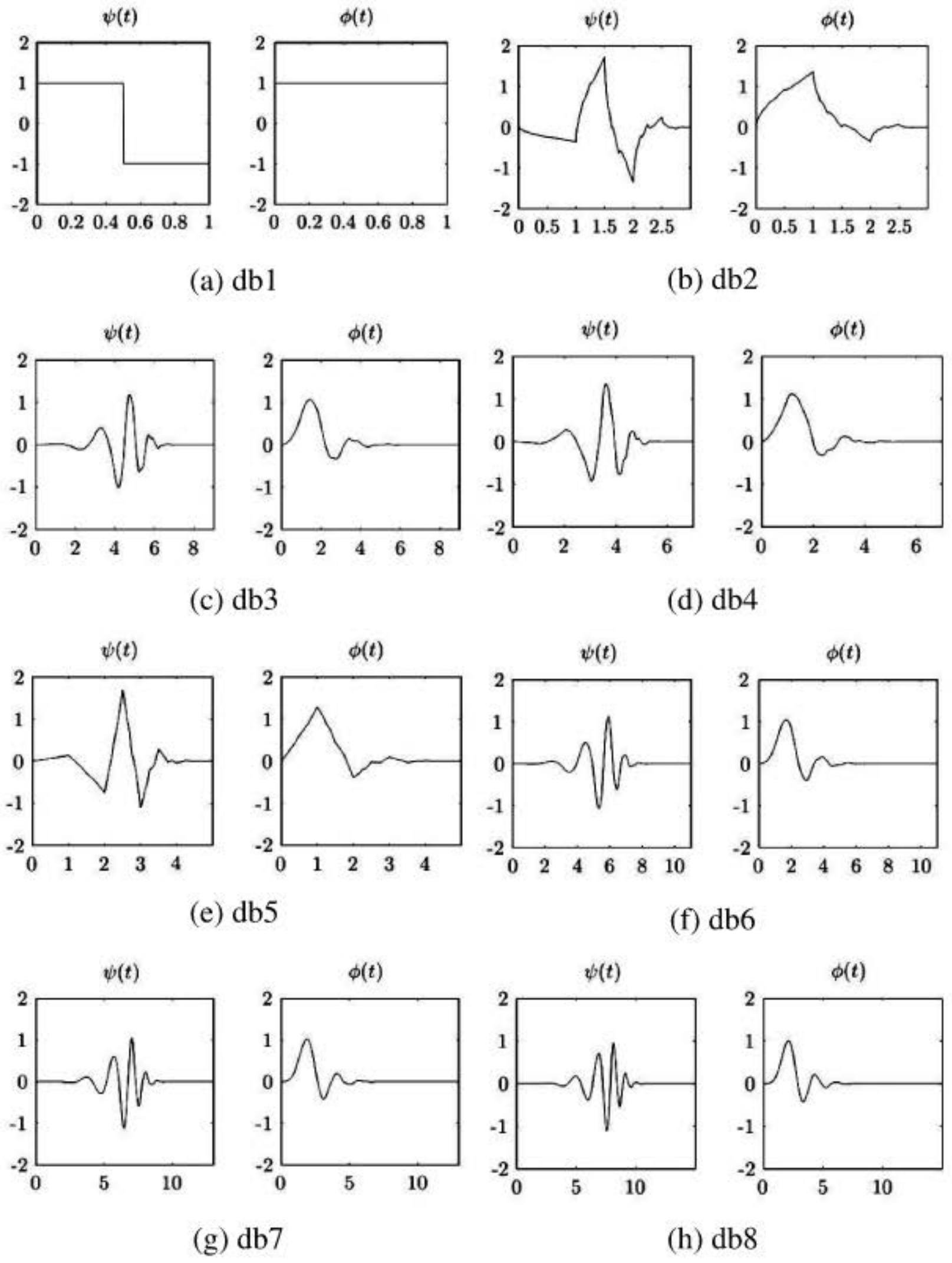
Hàm Wavelet Daubechies bậc nhất được biết đến như là hàm Wavelet Haar, vốn là hàm Wavelet đơn giản nhất. Hàm Wavelet và hàm tỷ lệ của Wavelet Daubechies có bậc từ 1 đến 8 được trình bày trong hình 3.14. Hàm Wavelet Haar hay db1 có thể được viết như sau:

$$\psi(t) = \begin{cases} 1 \ nếu \ x \in [0 \quad 0.5] \\ -1 nếu \ x \in [0.5 \quad 1] \\ 0 \quad nếu \ x \notin [0 \quad 1] \end{cases} \tag{3.52}$$

$$\phi(t) = \begin{cases} 1 \ nếu \ x \in [0 \quad 1] \\ 0 \ nếu \ x \notin [0 \quad 1] \end{cases} \tag{3.53}$$

Các hàm Daubechies bậc cao hơn không thật sự dễ để mô tả bằng biểu thức toán học. Bậc của Wavelet Daubechies đánh dấu số lượng moment triệt tiêu, hay số lượng của moment không (zero moment) của hàm Wavelet. Nó có liên quan đến số lượng dao động của hàm Wavelet. Số lượng moment triệt tiêu càng lớn thì sự định vị tần số của trong quá trình phân tích càng tốt. Sự độc lập giữa các hệ số Wavelet với các giá trị tỷ lệ khác nhau giảm khi tăng số bậc của Wavelet. Số bậc này được so sánh với bậc của bộ lọc tuyến tính. Các Wavelet Daubechies được xem là các Wavelet trực giao, trong khi các bộ lọc tỷ lệ là các bộ lọc có pha nhỏ nhất.

https://tieulun.hopto.org

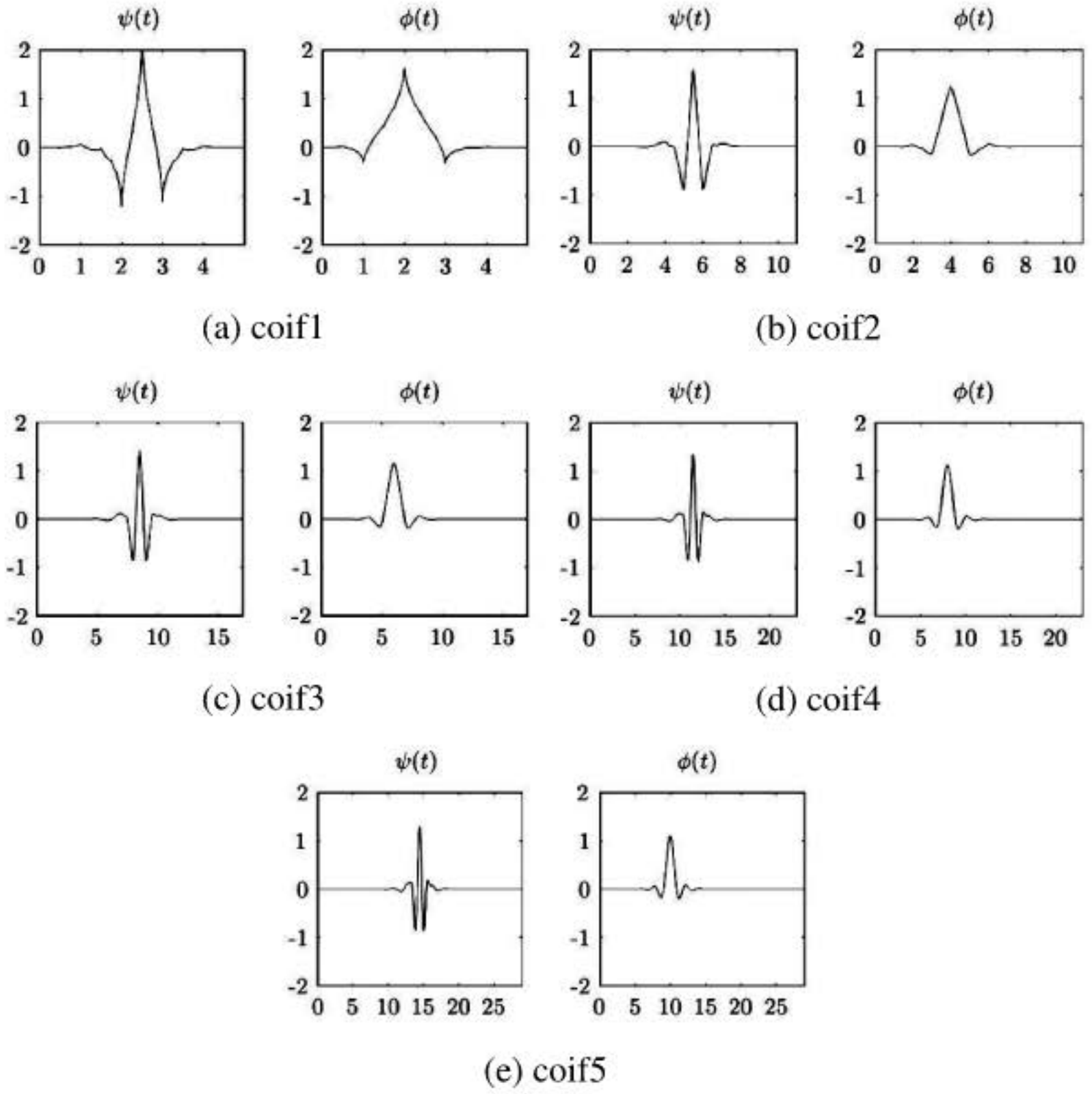


Hình 3.14. *Hàm Wavelet và hàm tỷ lệ của họ Daubechies*

https://tieulun.hopto.org

Coiflets

Coiflets cũng là hàm Wavelet trực giao với số lượng moment triệt tiêu lớn nhất với cả hàm Wavelet và hàm tỷ lệ có độ rộng cho trước. Các sóng Wavelet Coiflet đối xứng hơn và có nhiều moment triệt tiêu hơn so với Daubechies. Coiflet và các hàm tỷ lệ với bậc từ 1 đến 5 được trình bày trong hình 3.15.

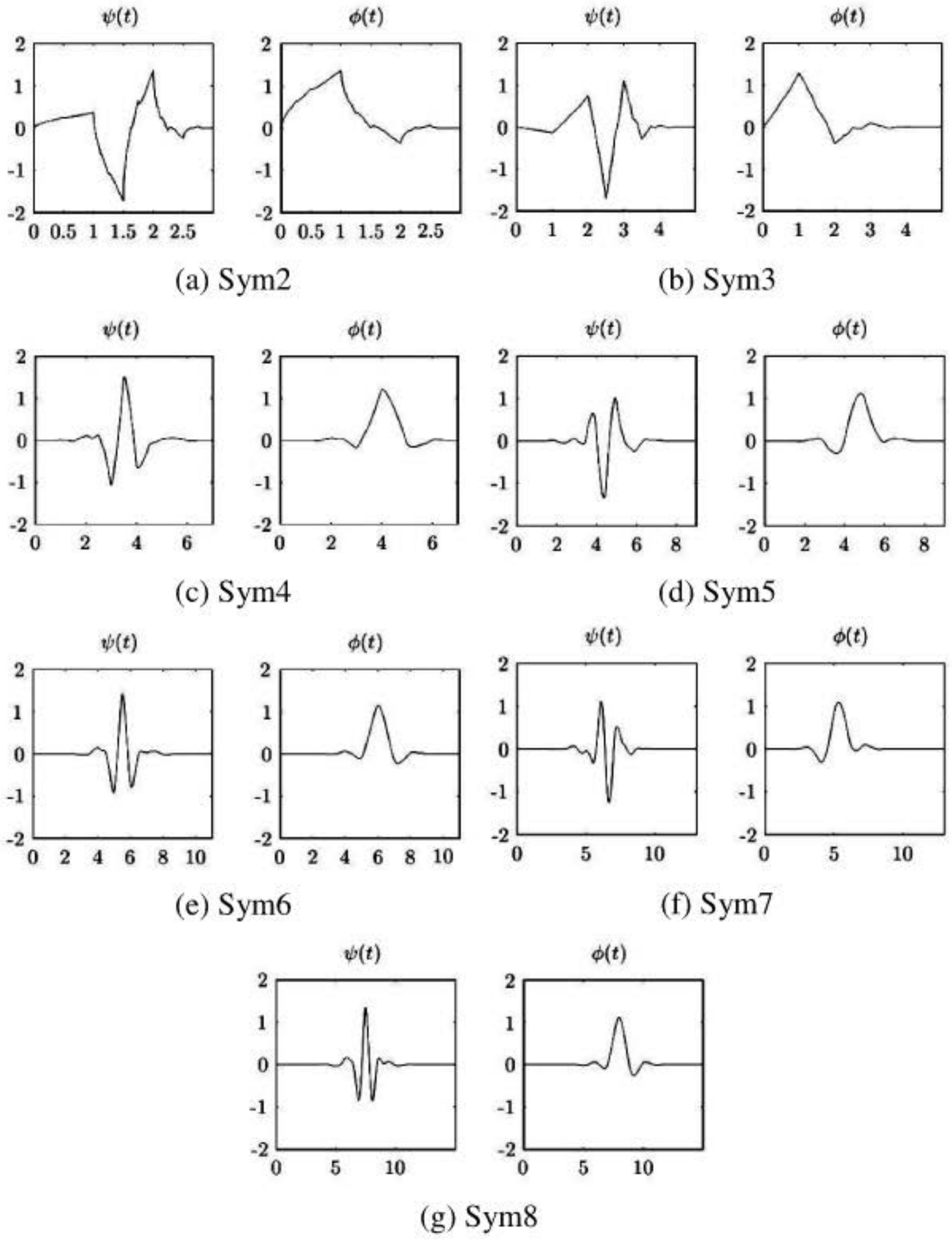


Hình 3.15. *Hàm Wavelet và hàm tỷ lệ của họ Coiflet*

https://tieulun.hopto.org

Symlets

Symlets cũng là một trong số những họ Wavelet trực giao. Symlets gần như đối xứng và có chút không đối xứng. Các bộ lọc tỷ lệ được cấp phát là các bộ lọc có pha tuyến tính. Đặc trưng của symlets gần giống với họ Wavelet Daubechies. Trong hình 3.16 là các hàm Wavelet và hàm tỷ lệ của symlets từ bậc 2 đến 8.



Hình 3.16. *Hàm Wavelet và hàm tỷ lệ của họ Symlets*

https://tieulun.hopto.org

Biorthogonal

Họ Wavelet biorthogonal (song trực giao) có biến đổi dạng thuận nghịch nhưng không nhất thiết phải trực giao và việc xây dựng có thể thực hiện bằng cách sử dụng bộ lọc FIR, điều này vốn dĩ là không thể đối với các bộ lọc trực giao. Sự đối xứng có nghĩa là các bộ lọc phải có pha tuyến tính.

Họ Wavelet này sử dụng các Wavelet riêng biệt và các hàm tỷ lệ cho quá trình phân tích và tổng hợp tín hiệu. Ứng với việc phân tích, hàm Wavelet được sử dụng, các hệ số Wavelet tương ứng có thể được tính như sau:

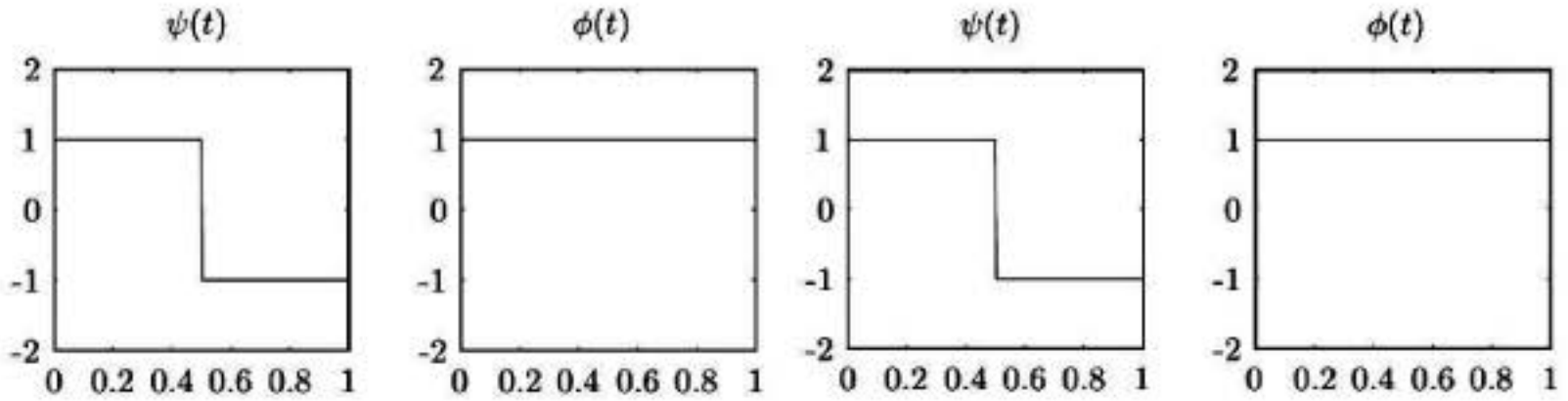
$$\tilde{c}_{j,k} = \int x(t)\tilde{\psi}_{j,k}(t)dt \tag{3.54}$$

Quá trình tổng hợp sử dụng Wavelet

$$x(t) = \sum_{j,k} \tilde{c}_{j,k}\psi_{j,k} \tag{3.55}$$

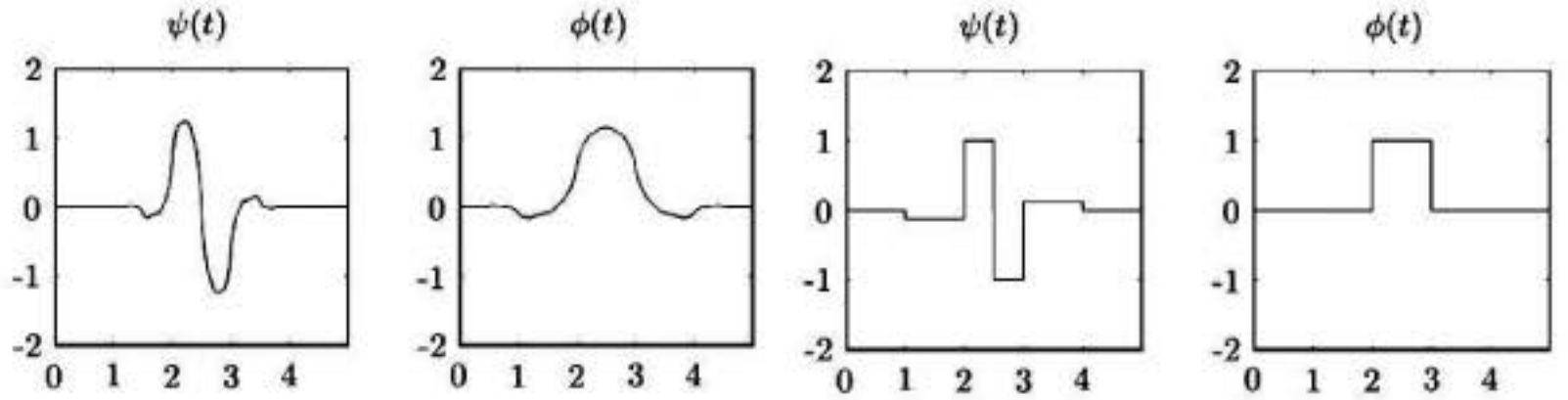
Hình 3.17 trình bày một số ví dụ về hàm phân tích và tổng hợp song trực giao. Bậc của bộ lọc dùng trong phân tích và tổng hợp được đánh dấu trong tên của họ Wavelet.

https://tieulun.hopto.org



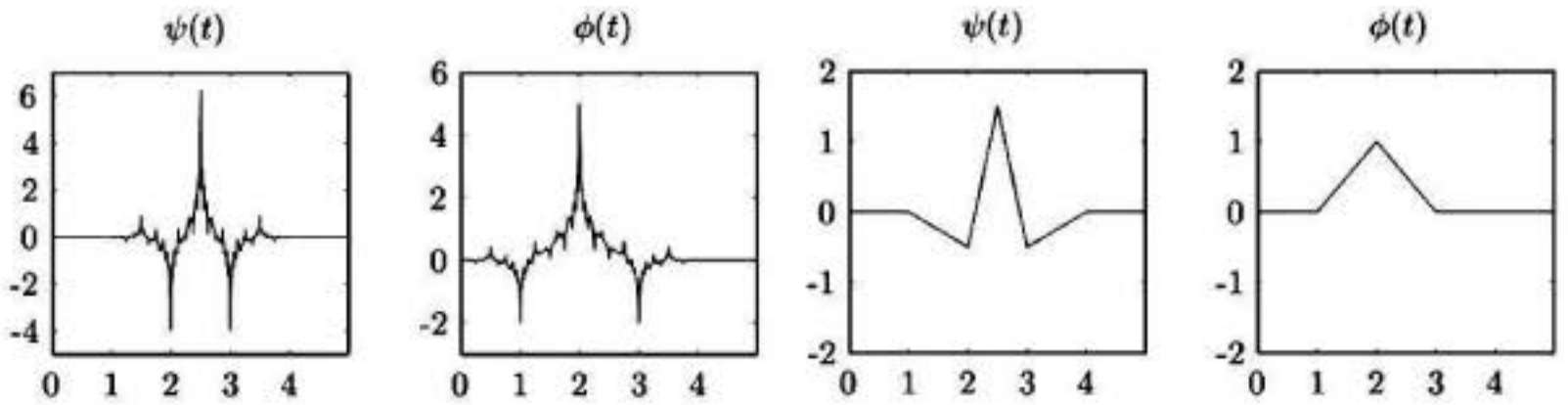
(a) Analysis (b) Synthesis

Bior1.1



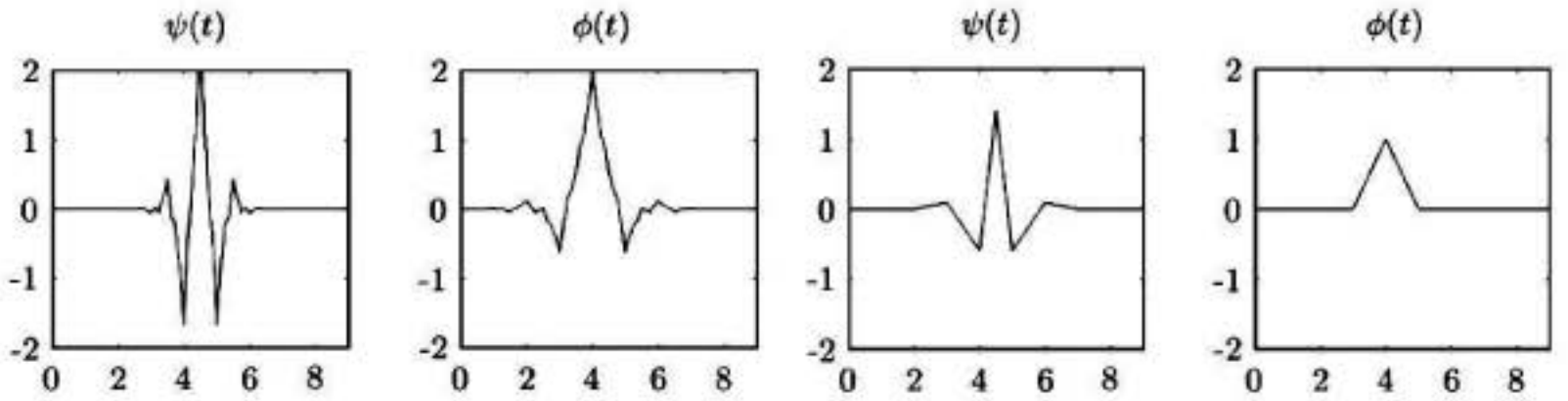
(c) Analysis (d) Synthesis

Bior1.3



(e) Analysis (f) Synthesis

Bior2.2



(g) Analysis (h) Synthesis

Bior2.4

Hình 3.17. *Hàm Wavelet và hàm tỷ lệ ứng với phép phân tích và tổng hợp của họ Biorthogonal*

https://tieulun.hopto.org

3.2.2.5. Biến đổi Wavelet trong ảnh số

Bằng việc mở rộng số chiều trong biến đổi Wavelet, cụ thể là với hàm tỷ lệ $\phi(x,y)$ và hàm Wavelet $\psi(x,y)$, ta có thể áp dụng biến đổi Wavelet cho dữ liệu hai chiều, chẳng hạn như ảnh số. Trong trường hợp này, hàm tỷ lệ và dịch chuyển được định nghĩa như sau:

$$\begin{aligned} &\phi_{j,m,n}(x,y) = 2^{j/2}\phi\left(2^j x - m, 2^j y - n\right), \\ &\psi^i_{j,m,n}(x,y) = 2^{j/2}\psi^i\left(2^j x - m, 2^j y - n\right), i = \{H,V,D\} \end{aligned} \tag{3.56}$$

Sẽ có ba hàm Wavelet khác nhau, $\psi^H(x,y)$, $\psi^V(x,y)$ và $\psi^D(x,y)$. Theo như định nghĩa trước đó, hàm tỷ lệ luôn là thành phần tần số thấp của hàm tỷ lệ trước đó trong không gian hai chiều. Do đó, sẽ chỉ có một hàm tỷ lệ hai chiều. Tuy nhiên, hàm Wavelet liên quan đến bậc dùng trong các bộ lọc. Nếu hàm Wavelet được tách biệt rõ ràng, nghĩa là, $f(x,y) = f_1(x)f_2(y)$. Các hàm này được viết rõ ra như sau:

$$\begin{aligned} \phi(x,y) &= \phi(x)\phi(y), \\ \psi^H(x,y) &= \psi(x)\phi(y), \\ \psi^V(x,y) &= \phi(x)\psi(y), \\ \psi^D(x,y) &= \psi(x)\psi(y), \end{aligned} \tag{3.57}$$

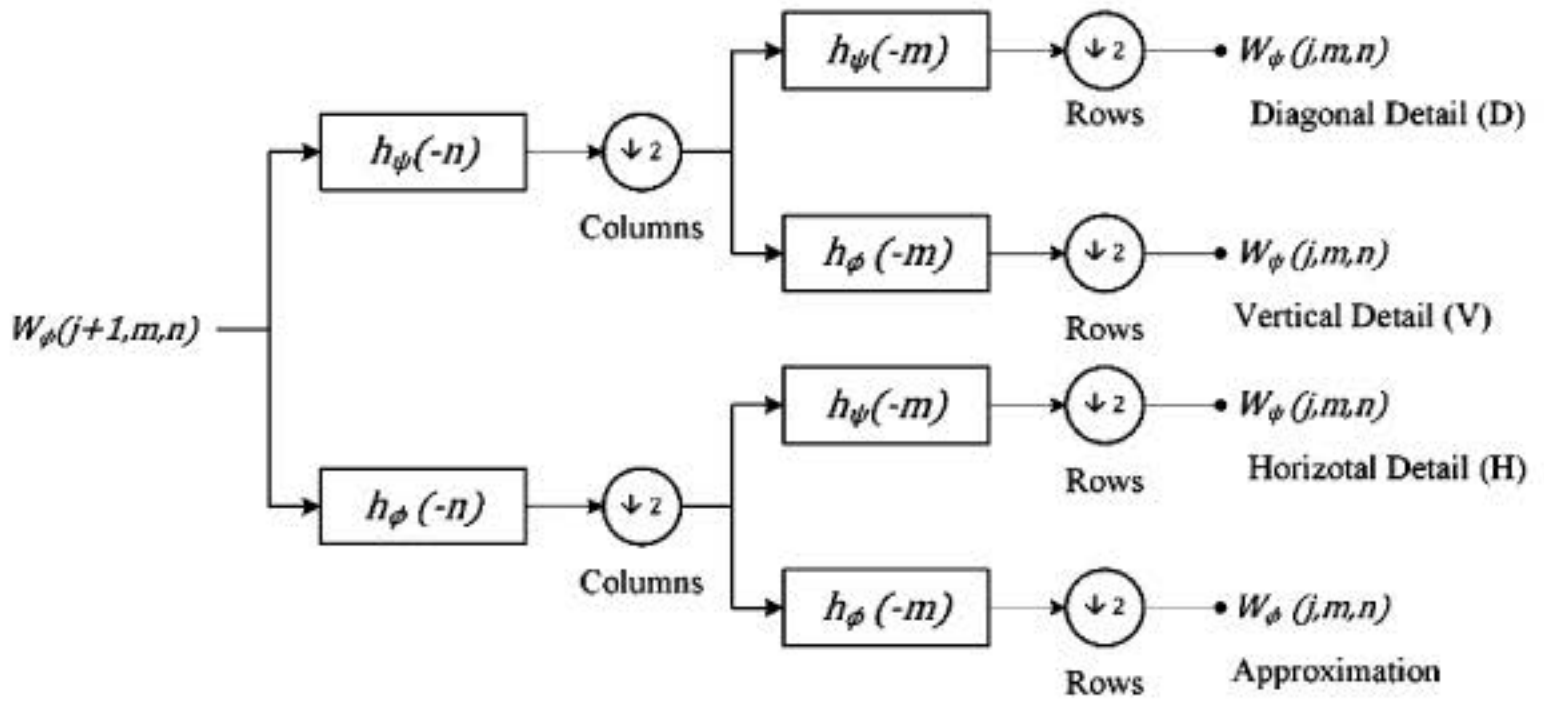
Nếu xem đây là các hàm tách biệt thì sẽ rất dễ dàng để phân tích các hàm hai chiều này và do đó ta chỉ nên tập trung vào việc thiết kế các hàm một chiều trước. Biểu thức hàm phân tích và tổng hợp được sửa đổi lại dưới dạng hai chiều như sau:

$$\begin{aligned} W_\phi(j_0,m,n) &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\,\phi_{j_0,m,n}(x,y) \\ W_\psi(j,m,n) &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\,\phi^i_{j,m,n}(x,y), i = \{H,V,D\} \\ f(x,y) &= \frac{1}{\sqrt{MN}}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} W_\phi\left(j_{0,m,n}\right)\phi_{j_0,m,n}(x,y) \\ &\quad + \frac{1}{\sqrt{MN}}\sum_{i=H,V,D}\sum_{j=j_0}^{\infty}\sum_{m}\sum_{n} W^i_\psi(j,m,n)\,\phi^i_{j,m,n}(x,y) \end{aligned} \tag{3.58}$$

Đây là dạng tổng quát của biến đổi Wavelet hai chiều. Nếu các hàm tỷ lệ và hàm Wavelet là tách biệt, sự tổng hợp có thể được chia thành hai bước: tính theo trục chiều học và chiều ngang; bên cạnh đó, có thể sử

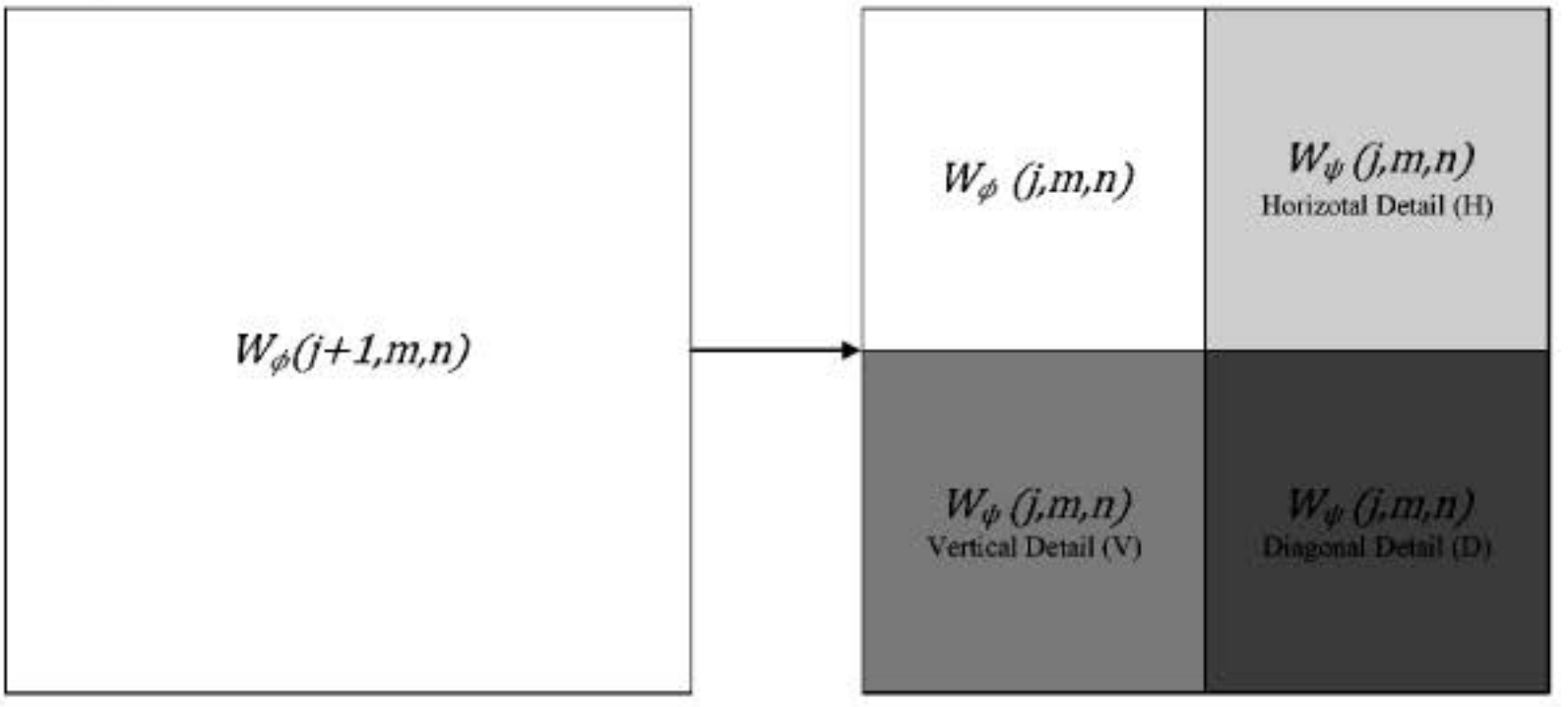
https://tieulun.hopto.org

dụng biến đổi Wavelet nhanh để tiết kiệm thời gian. Hình 3.18 biểu diễn quá trình phân tích dữ liệu hai chiều thành bốn thành phần riêng biệt với một mức phân tích. Ở đây các bộ lọc thông thấp và thông cao được hiểu như là các hàm tỷ lệ và hàm Wavelet.



Hình 3.18. *Sơ đồ thực hiện phép biến đổi Wavelet hai chiều*

Với một tín hiệu hai chiều như ảnh số, dữ liệu sẽ được phân chia thành bốn băng con: LL (left-top) biểu diễn các hệ số có tần số thấp nhất chứa các thành phần xấp xỉ, HL (right-top) chứa các thành phần chi tiết theo chiều ngang, LH (left-bottom) chứa các thành phần chi tiết theo chiều dọc và HH (right-bottom) chứa các chi tiết theo đường chéo như trong hình 3.19.



Hình 3.19. *Biểu diễn các thành phần băng con khi phân tích Wavelet cho ảnh số*

https://tieulun.hopto.org

3.2.2.6. Biến đổi Wavelet dùng Wavelet Toolbox

Biến đổi Wavelet được hỗ trợ rất nhiều trong Matlab với riêng một Toolbox. Đầu tiên là việc cung cấp các dạng hàm Wavelet khác nhau thông qua hàm `wfilters` với cú pháp cơ bản như sau:

```
[Lo_D,Hi_D,Lo_R,Hi_R] = wfilters('wname');
```

Trong đó, Lo_D và Hi_D là các bộ lọc thông thấp và thông cao dùng trong quá trình phân tích Wavelet, trong khi đó, Lo_R và Hi_R là các bộ lọc dùng trong quá trình tổng hợp, và tất nhiên các bộ lọc này phải trực giao với nhau. Ngoài ra, người đọc có thể tham khảo thêm hàm tạo các bộ lọc tương đương với hàm `wfilters`:

```
[F1,F2] = wfilters('wname','type')
```

Còn để trả về dạng số xấp xỉ của hàm Wavelet hay hàm tỷ lệ, hàm `wavefun` trong Matlab cho phép thực hiện điều đó:

```
[phi, psi, xval] = wavefun(vname,iter)
```

Kết quả trả về các vector xấp xỉ `phi` và `psi`, trong khi đó `xval` là vector ước lượng. Số nguyên dương `iter` xác định độ chính xác của việc xấp xỉ bằng cách điều chỉnh số lượng bước lặp được dùng trong tính toán. Với các biến đổi trực giao, hàm phù hợp được cho như sau:

```
[phi1, psi1, phi2, psi2, xval] = wavefun(wname, iter)
```

Trong đó `phi1` và `psi1` là các hàm phân giải, trong khi `phi2` và `psi2` là các hàm tái cấu trúc.

Ví dụ 3.3: Biểu diễn các hàm bộ lọc, hàm tỷ lệ và hàm Wavelet của họ Wavelet Haar ở hình 3.20

```
clear all;
[Lo_D,Hi_D,Lo_R,Hi_R] = wfilters('Haar')
waveinfo('Haar')
[phi,psi,xval]=wavefun('Haar',10)
xaxis=zeros(size(xval));
subplot(121);
plot(xval,phi,'k',xval,xaxis,'--k');
axis([0 1 -1.5 1.5]);
axis square;
title('Haar Scaling Function')
subplot(122);
```


https://tieulun.hopto.org

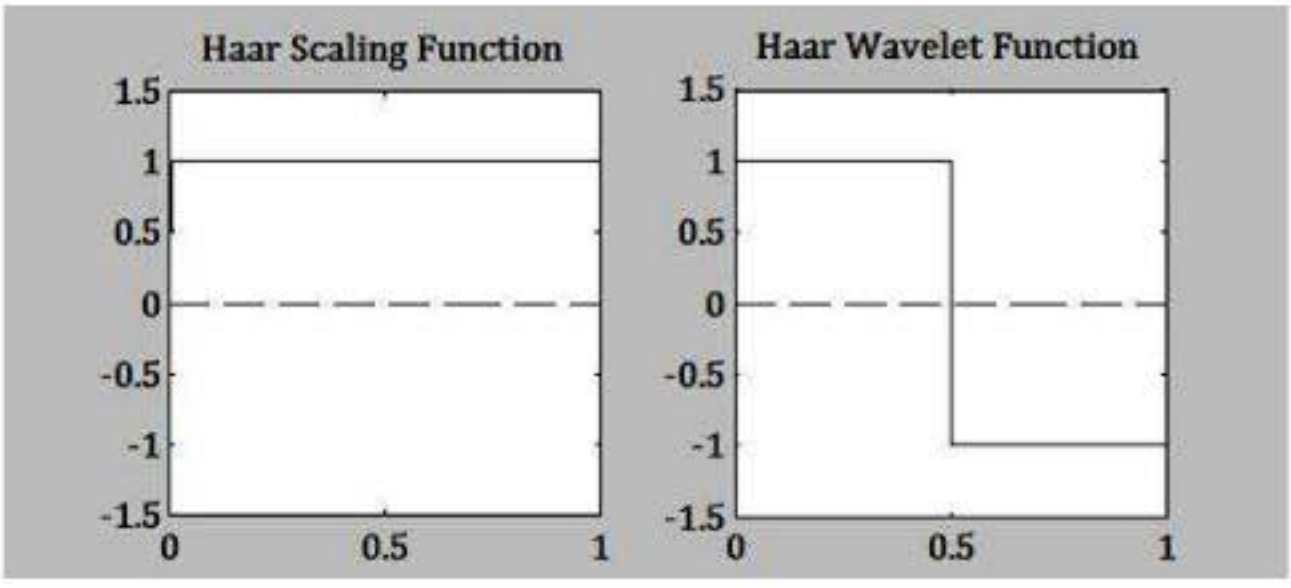
```
plot(xval,psi,'k',xval,xaxis,'--k');
axis([0 1 -1.5 1.5]);
axis square;
title('Haar Wavelet Function');
```

Kết quả thực thi chương trình

```
>>
Lo_D =
    0.7071    0.7071
Hi_D =
   -0.7071    0.7071
Lo_R =
    0.7071    0.7071
Hi_R =
0.7071   -0.7071
Information on Haar Wavelet.
    Haar Wavelet
    General characteristics: Compactly supported
    Wavelet, the oldest and the simplest Wavelet.
    scaling function phi = 1 on [0 1] and 0 otherwise.
    Wavelet function psi = 1 on [0 0.5[, = -1 on [0.5 1]
and 0 otherwise.
    Family                  Haar
    Short name              haar
    Examples                haar is the same as db1
    Orthogonal              yes
    Biorthogonal            yes
    Compact support         yes
    DWT                     possible
    CWT                     possible

    Support width           1
    Filters length          2
```


https://tieulun.hopto.org

```
Regularity                haar is not continuous
Symmetry                  yes
Number of vanishing
moments for psi           1

Reference: I. Daubechies,
Ten lectures on Wavelet,
CBMS, SIAM, 61, 1994, 194-202.
```



Hình 3.20. *Hàm tỷ lệ và hàm Wavelet của họ Haar*

Một trong những cách đơn giản để tính toán việc biến đổi Wavelet dành cho các tín hiệu hai chiều được hỗ trợ bởi Toolbox Wavelet là hàm `wavedec2` với cú pháp như sau:

```
[C, S] = wavedec2(X, N, Lo_D, Hi_D)
```

trong đó, `X` là ma trận hai chiều hay ảnh số, `N` là mức phân tích, `Lo_D` và `Hi_D` là các hàm bộ lọc dùng trong phân tích. Cú pháp đơn giản hơn của hàm này như sau:

```
[C,S] = wavedec2(X,N,'wname')
```

Với cú pháp này không cần thiết phải chỉ rõ hàm bộ lọc, trong khi chỉ cần khai báo tên họ Wavelet dùng để phân tích. Ngõ ra bao gồm vector hàng `C` (kiểu double) chứa các hệ số Wavelet đã được biến đổi. Ma trận `S` (cũng thuộc kiểu double) xác định bố cục của các hệ số trong `C`. Mối quan hệ giữa `C` và `S` được khảo sát trong ví dụ và được mô tả trong phần tiếp theo.

Ví dụ 3.4: Xem xét biến đổi Wavelet dùng hàm `wavedec2` dùng họ Haar

```
clear all;
f=magic(4);
```


https://tieulun.hopto.org

```
[C,S]=wavedec2(f,1,'Haar');
```

Hàm `magic(m)` trả về ma trận vuông có kích thước m^2 với các phần tử có giá trị từ 1 đến m^2.

```
f =

    16     2     3    13

     5    11    10     8

     9     7     6    12

     4    14    15     1

C =

  Columns 1 through 10

   17.0000    17.0000    17.0000    17.0000     1.0000    -
1.0000   -1.0000    1.0000    4.0000   -4.0000

  Columns 11 through 16

   -4.0000     4.0000    10.0000     6.0000    -6.0000   -
10.0000

S =

     2     2

     2     2

     4     4
```

Lưu ý ma trận `S` thông thường có kích thước $(N + 2) \times 2$. Với ví dụ như trên, hàng đầu tiên chỉ ra kích thước của ma trận hệ số xấp xỉ, hàng thứ hai ứng với kích thước của các ma trận hệ số chi tiết và hàng cuối là kích thước của ngõ vào. Dựa trên kích thước được chỉ rõ trong ma trận `S`, ta có thể xác định được các hệ số trong vector `C`. Do chỉ phân tích với một mức duy nhất nên ma trận hệ số của các băng con xấp xỉ và chi tiết có kích thước bằng phân nửa so với dữ liệu vào. Cụ thể, `C(1:4)` là hệ số xấp xỉ và `C(5:8)`, `C(9:12)`, `C(13:16)` tương ứng là hệ số chi tiết theo chiều ngang, chiều dọc và đường chéo.

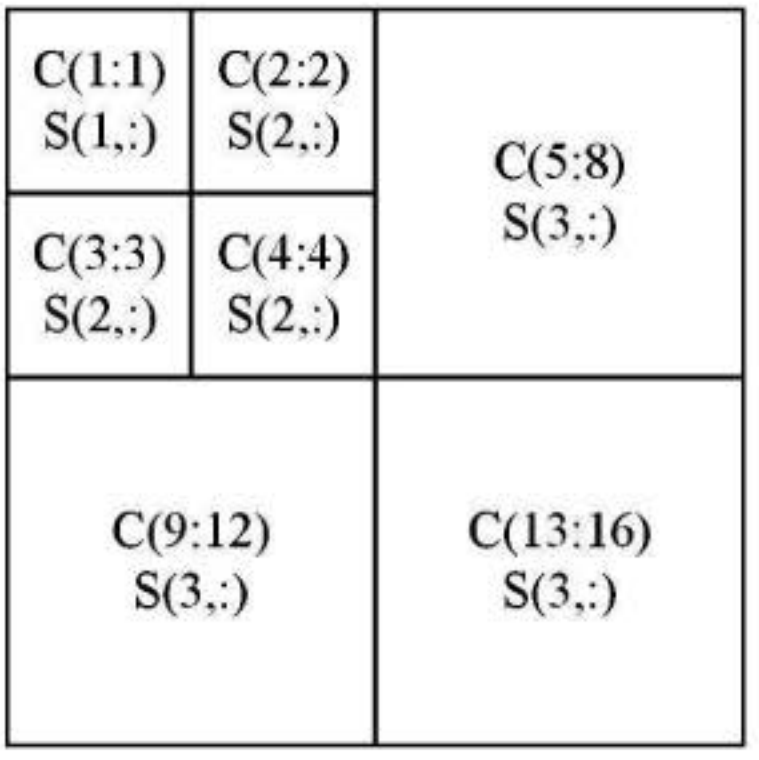
Để hiểu rõ hơn về cách thức trình bày ngõ ra của hàm `wavedec2`, vẫn với cách thực hiện như ví dụ 3.4 nhưng sử dụng hai mức phân tích.

```
[C,S]=wavedec2(f,2,'Haar');


C =

  Columns 1 through 10

   34.0000         0         0    0.0000    1.0000   -
1.0000   -1.0000    1.0000    4.0000   -4.0000
```


https://tieulun.hopto.org

```
  Columns 11 through 16

   -4.0000      4.0000    10.0000      6.0000    -6.0000   -
10.0000


S =

     1     1

     1     1

     2     2

     4     4
```

Có thể thấy `S(1,:)` chỉ kích thước của ma trận hệ số xấp xỉ tại mức phân tích thứ 2. `S(2,:)` chỉ kích thước của các ma trận hệ số chi tiết tại mức phân tích thứ 2. `S(3,:)` kích thước của các ma trận hệ số chi tiết tại mức phân tích thứ nhất và `S(4,:)` là kích thước của ma trận dữ liệu ngõ vào. Hình 3.21 trình bày sự phân bố các băng con dưới dạng ma trận.



Hình 3.21. *Sơ đồ bố trí của các băng con khi dùng hàm wavedec2*

Ngoài ra, Wavelet Toolbox còn hỗ trợ cho phép trích xuất chính xác các hệ số Wavelet của từng băng con tại từng mức phân tích. Tham khảo ví dụ sau đây cũng dựa trên kết quả thực hiện trước đó.

Ví dụ 3.5: Trích xuất các hệ số Wavelet của từng băng con tại các mức phân tích xác định

```
clear all;
f=magic(4);
[C,S]=wavedec2(f,2,'Haar');
A=appcoef2(C,S,'Haar',1)
```


https://tieulun.hopto.org

```
V=detcoef2('v',C,S,'Haar',2)
```

Trong ví dụ trên, hàm `appcoef2` sẽ trả về ma trận chứa các hệ số Wavelet của băng con xấp xỉ tại bất kỳ mức phân tích nào, lưu ý, mức phân tích khai báo phải nhỏ hơn giá trị khai báo trong hàm `wavedec2` trước đó. Cú pháp của hàm `appcoef2` được cho như sau:

```
A = appcoef2(C,S,'wname',N)
```

Hay

```
A = appcoef2(C,S,Lo_R,Hi_R,N)
```

Tương tự với hàm `detcoef2` cũng trả về ma trận hệ số Wavelet của băng con chi tiết, nhưng có thêm một tham số cần khai báo, đó là băng con cụ thể. Trong trường hợp của ví dụ 3.5 thì băng con trích xuất là vertical - băng con chi tiết theo chiều dọc. Cú pháp của hàm `detcoef2` được trình bày:

```
D = detcoef2(O,C,S,N)
```

Tham số `O` chỉ rõ băng con chi tiết cần khai báo, chẳng hạn `'h' - horizontal`, `'v' - vertical`, `'d' - diagonal`.

Ví dụ 3.6: Thực hiện biến đổi DWT cho ma trận 2 chiều dùng Wavelet Haar.

Với kết quả thu được từ ví dụ 3.3 là các hàm bộ lọc thông thấp và thông cao dùng trong quá trình phân tích thì hoàn toàn có thể thực hiện việc biến đổi DWT với ma trận được cho từ hàm `magic(4)`.

$$f = \begin{bmatrix} 16 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 15 & 1 \end{bmatrix}$$

Với hai bộ lọc thông thấp $Lo_D = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$ và $Hi_D = \begin{bmatrix} \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$

Như đã đề cập, thành phần xấp xỉ chứa các hệ số có tần số thấp nhất, do đó có thể thực hiện theo như sơ đồ trong hình 3.18. Cụ thể theo các bước sau đây (chỉ với thành phần xấp xỉ):

- Tính tích chập giữa f và Lo_D theo từng hàng. Gọi O_1 là ngõ ra, O_1 có kích thước trong ví dụ này là 5×4. Tích chập được tính như sau:

https://tieulun.hopto.org

$$\begin{array}{cccc} 2^{-1/2} & & & \\ 2^{-1/2} & 16 & 2 & 3 & 13 \\ & 5 & 11 & 10 & 8 \\ & 9 & 7 & 6 & 12 \\ & 4 & 14 & 15 & 1 \end{array}$$

(a)

$$\begin{array}{cccc} 2^{-1/2} & 16 & 2 & 3 & 13 \\ 2^{-1/2} & 5 & 11 & 10 & 8 \\ & 9 & 7 & 6 & 12 \\ & 4 & 14 & 15 & 1 \end{array}$$

(b)

$$\begin{array}{cccc} & 16 & 2 & 3 & 13 \\ 2^{-1/2} & 5 & 11 & 10 & 8 \\ 2^{-1/2} & 9 & 7 & 6 & 12 \\ & 4 & 14 & 15 & 1 \end{array}$$

(c)

$$\begin{array}{cccc} & 16 & 2 & 3 & 13 \\ & 5 & 11 & 10 & 8 \\ 2^{-1/2} & 9 & 7 & 6 & 12 \\ 2^{-1/2} & 4 & 14 & 15 & 1 \end{array}$$

(d)

$$\begin{array}{cccc} & 16 & 2 & 3 & 13 \\ & 5 & 11 & 10 & 8 \\ & 9 & 7 & 6 & 12 \\ 2^{-1/2} & 4 & 14 & 15 & 1 \\ 2^{-1/2} & & & & \end{array}$$

(e)

Bước (a) xác định được các hệ số như sau:

$$O_1(1,1) = 1/\sqrt{2} \times 16$$

$$O_1(1,2) = 1/\sqrt{2} \times 2$$

$$O_1(1,3) = 1/\sqrt{2} \times 3$$

$$O_1(1,4) = 1/\sqrt{2} \times 13$$

Tương tự bước (b) xác định được:

$$O_1(2,1) = 1/\sqrt{2} \times (16 + 5)$$

$$O_1(2,2) = 1/\sqrt{2} \times (2 + 11)$$

$$O_1(2,3) = 1/\sqrt{2} \times (3 + 10)$$

$$O_1(2,4) = 1/\sqrt{2} \times (13 + 8)$$

Ngõ ra O_1 được xác định:

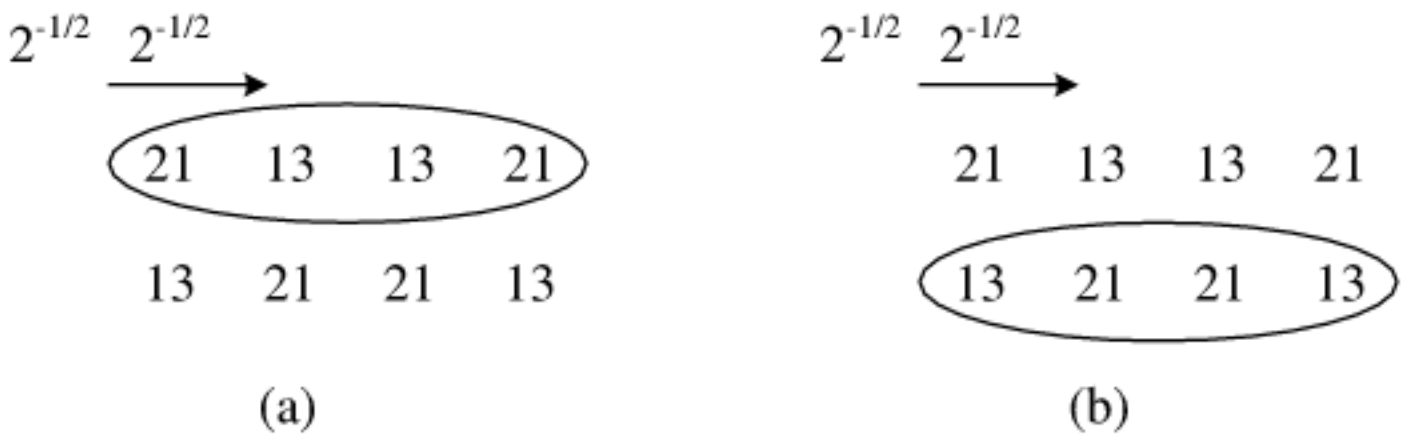
$$O_1 = 1/\sqrt{2} \times \begin{bmatrix} 16 & 2 & 3 & 13 \\ 21 & 13 & 13 & 21 \\ 14 & 18 & 16 & 20 \\ 13 & 21 & 21 & 13 \\ 4 & 14 & 15 & 1 \end{bmatrix}$$

https://tieulun.hopto.org

- Thực hiện giảm mẫu ở những mẫu lẻ theo hàng (loại bỏ hàng lẻ)

$$O_1' = 1/\sqrt{2} \times \begin{bmatrix} 21 & 13 & 13 & 21 \\ 13 & 21 & 21 & 13 \end{bmatrix}$$

- Thực hiện tích chập một lần nữa cũng với hàm bộ lọc thông thấp. Đặt ngõ ra là O_2



Ngõ ra sau khi tính toán:

$$O_2 = \left(1/\sqrt{2}\right)^2 \times \begin{bmatrix} 21 & 34 & 26 & 34 & 21 \\ 13 & 34 & 42 & 34 & 13 \end{bmatrix}$$

- Sau đó thực hiện giảm mẫu theo cột với những cột lẻ. Kết quả thu được là các hệ số tần số thấp hay còn gọi là thành phần xấp xỉ.

$$A = \left(1/\sqrt{2}\right)^2 \times \begin{bmatrix} 34 & 34 \\ 34 & 34 \end{bmatrix} = \begin{bmatrix} 17 & 17 \\ 17 & 17 \end{bmatrix}$$

Các thành phần chi tiết có thể thực hiện theo cách như trên với sự hỗ trợ từ các bộ lọc thông thấp và thông cao. Để kiểm tra kết quả, dùng hàm `appcoef2` hay `detcoef2` để trích xuất và so sánh.

Bên cạnh hàm `wavedec2` cho phép phân tích Wavelet đa mức, MATLAB cũng hỗ trợ hàm phân tích Wavelet hai chiều từng mức với ngõ ra tách biệt từng băng con. Hàm `dwt2` cho phép trả về bốn ma trận riêng biệt, gồm: thành phần xấp xỉ (cA), thành phần chi tiết theo chiều ngang (cH), chi tiết theo chiều dọc (cV) và chi tiết theo đường chéo (cD).

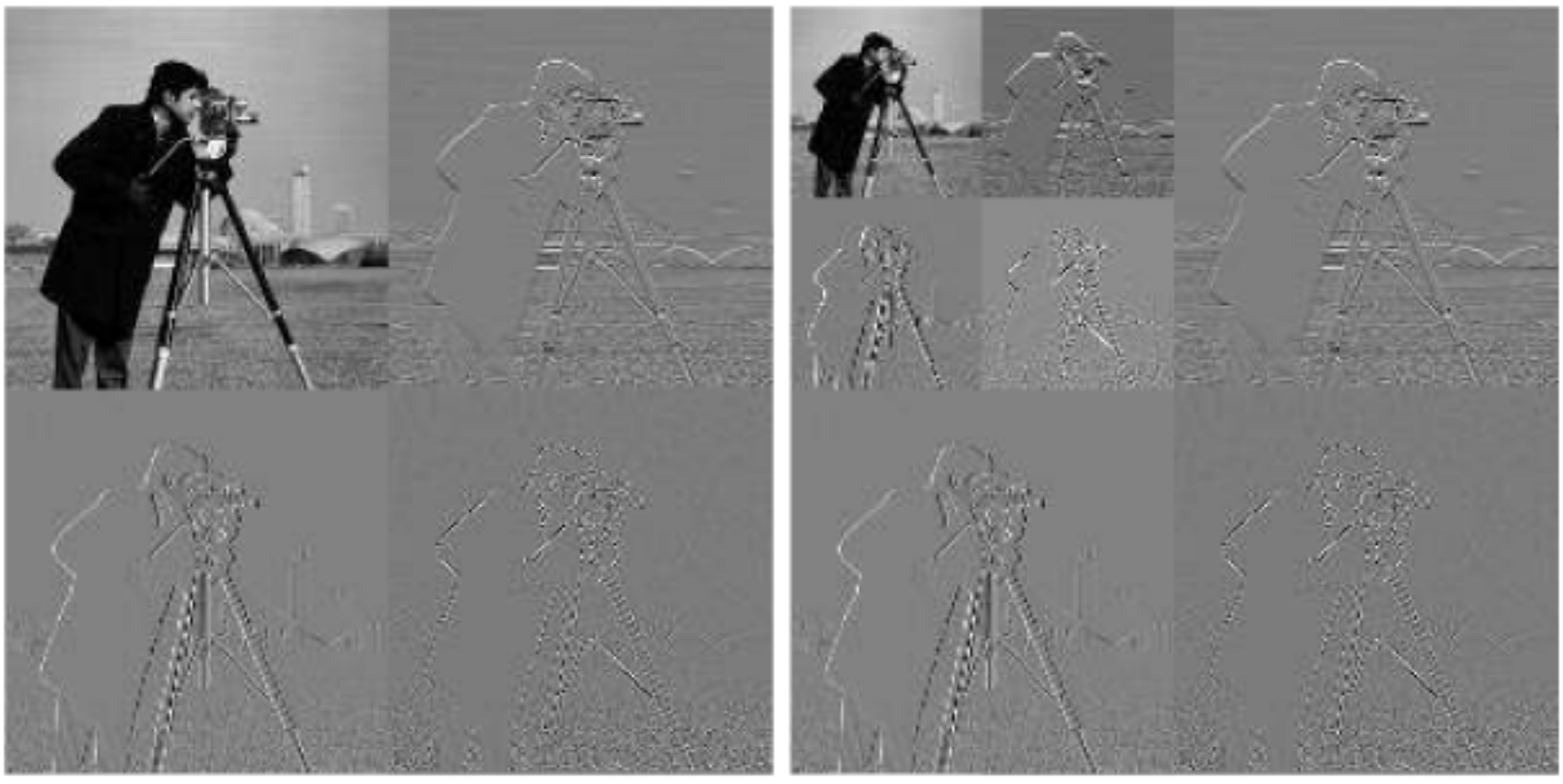
Ví dụ 3.7: Phân tích và biểu diễn các thành phần băng con với một ảnh cho trước

```
clear all;
f=imread('cameraman.bmp');
[cA1,cH1,cV1,cD1]=dwt2(f,'Haar');
f1=[mat2gray(cA1) mat2gray(cH1);mat2gray(cV1) mat2gray(cD1)];
[cA2,cH2,cV2,cD2]=dwt2(cA1,'Haar');
f22=[mat2gray(cA2) mat2gray(cH2);mat2gray(cV2) mat2gray(cD2)];
```


https://tieulun.hopto.org

```
f2=[f22 mat2gray(cH1);mat2gray(cV1) mat2gray(cD1)];
imshow(f1);
figure;imshow(f2);
```

Lưu ý, hàm `mat2gray` cho phép chuyển đổi một ma trận bất kỳ thành một ảnh xám. Trong ví dụ trên, các hệ số của các thành phần xấp xỉ hay chi tiết đều có dạng thập phân và kiểu double nên cần được chuẩn hóa. Việc chuẩn hóa có thể được thực hiện thủ công hay ngắn gọn là chỉ việc chuyển nó thành một ảnh dùng để biểu diễn như hình 3.22 với: (a) biểu diễn các thành phần với mức ứng với một mức phân tích và (b) ứng với hai mức phân tích. Với mỗi mức phân tích thì kích thước của ma trận giảm đi phân nửa so với trước đó.



(a) (b)

Hình 3.22.*Biểu diễn các thành phần tần số sau khi dùng DWT cho ảnh xám: (a) Một mức phân tích; (b) Hai mức phân tích*

3.2.2.7. Một số ứng dụng của phép biến đổi Wavelet

Biến đổi Wavelet được dùng nhiều trong các ứng dụng xử lý ảnh do tính hiệu quả so với biến đổi Fourier trong việc biểu diễn các đặc trưng tần số theo thời gian. Do đó, ứng dụng xử lý ảnh trên miền Wavelet thông thường là các thuật toán chỉnh sửa, thay đổi giá trị của các hệ số trong các băng con *xấp xỉ* hay *chi tiết*. Công việc này có thể được tóm gọn trong ba bước cơ bản như sau:

1) Biến đổi ảnh từ miền không gian sang miền Wavelet (thực hiện biến đổi Wavelet rời rạc)

2) Thay đổi các hệ số Wavelet

https://tieulun.hopto.org

3) Biến đổi ngược các hệ số sau khi chỉnh sửa từ miền Wavelet trở lại miền không gian.

Để hiểu rõ hơn ứng dụng Wavelet, một số ví dụ sẽ được trình bày bên dưới, nhưng cơ bản sẽ bao gồm các ứng dụng như: triệt nhiễu tần số cao, nén ảnh, tách biên…

Ví dụ 3.8: Thực hiện thuật toán giảm nhiễu dùng biến đổi Wavelet

```
clear all
f=imread('cameraman.bmp');
fn=imnoise(f,'gaussian',0.01);
[cA,cH,cV,cD]=dwt2(fn,'Haar');
[THR_H,SORH_H,KEEPAPP_H]=ddencmp('den','wv',cH);
cH=wthresh(cH,SORH_H,THR_H);
[THR_V,SORH_V,KEEPAPP_V]=ddencmp('den','wv',cV);
cV=wthresh(cH,SORH_V,THR_V);
[THR_D,SORH_D,KEEPAPP_D]=ddencmp('den','wv',cD);
cD=wthresh(cH,SORH_D,THR_D);
fdn=uint8(idwt2(cA,cH,cV,cD,'Haar'));
```

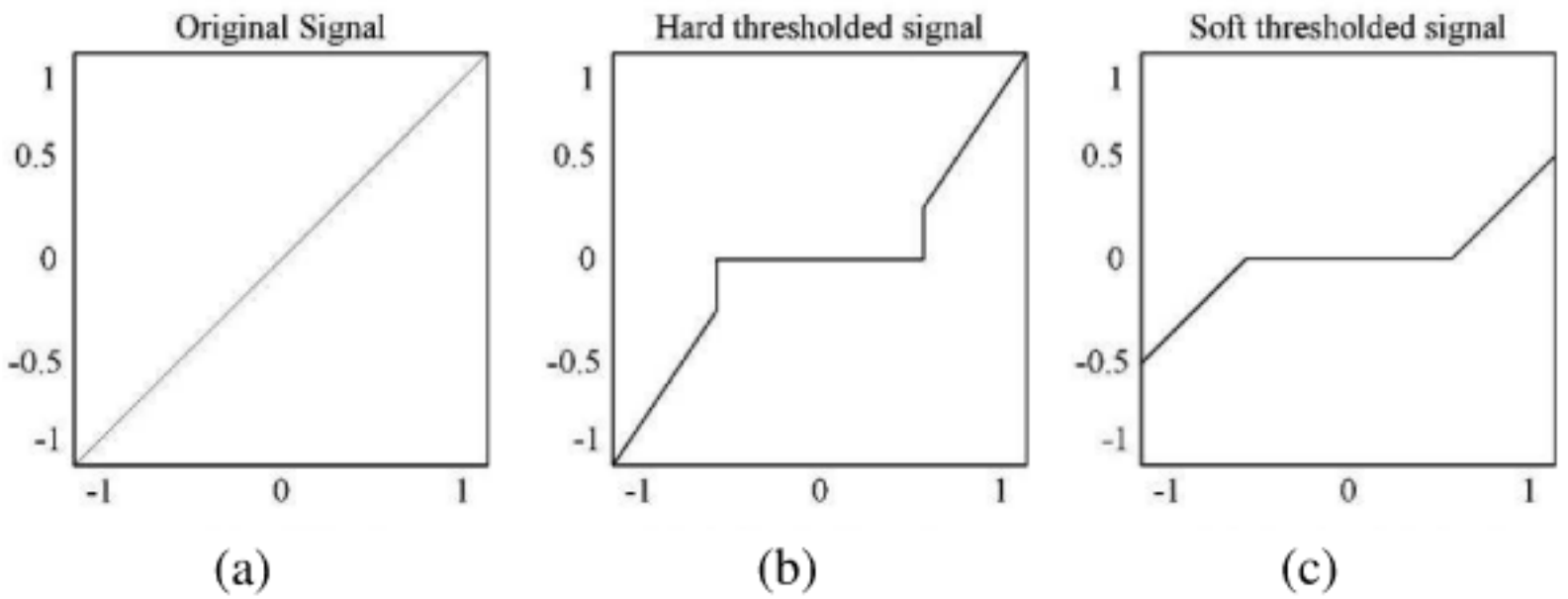
Trong ví dụ trên, hàm `idwt2` thực hiện quá trình biến đổi Wavelet ngược với cú pháp tương tự như hàm `dwt2`, chỉ có điều ngõ ra của hàm `dwt2` bây giờ trở thành ngõ vào của hàm `idwt2`. Cú pháp hoàn chỉnh được cho như sau:

```
X = idwt2(cA,cH,cV,cD,'wname')
```

Ảnh sau khi được thêm nhiễu muối tiêu với mật độ cho trước. Mục tiêu là giảm nhiễu và nâng cao chất lượng của ảnh được đánh giá qua thông số PSNR. Do đây là dạng nhiễu tần số cao làm ảnh hưởng đến các chi tiết của ảnh nên thuật toán triệt nhiễu chỉ được áp dụng cho các băng tần chi tiết (băng tần tần số cao), cụ thể là thành phần cH, cV và cD. Thuật toán trong ví dụ trên chỉ đơn giản là lấy ngưỡng các hệ số Wavelet trong các băng tần chi tiết. Có hai kiểu lấy ngưỡng: lấy ngưỡng cứng (hard thresholding) và lấy ngưỡng mềm (soft thresholding) như trong hình 3.23. Các thông số như giá trị ngưỡng và lựa chọn kiểu lấy ngưỡng được xác định thông qua việc sử dụng hàm `ddencmp`, đây là hàm sẽ trả về các thông số mặc định cho việc lấy ngưỡng sao cho phù hợp với từng dữ liệu ngõ vào. Cú pháp của hàm `ddencmp` được mô tả:

```
[THR,SORH,KEEPAPP] = ddencmp('den','wv',X)
```


https://tieulun.hopto.org



Hình 3.23. *Mô tả các kiểu lấy ngưỡng*

Thông qua việc so sánh chất lượng của ảnh nhiễu và sau khi giảm nhiễu để thấy được hiệu quả của thuật toán triệt nhiễu bằng Wavelet hay có thể quan sát trực quan ảnh trước và sau khi thực hiện thuật toán như hình 3.24.

```
[MSE1,PSNR1]=psnr(f,fn)
[MSE2,PSNR2]=psnr(f,fdn)

MSE1 =
   2.0508e+03
PSNR1 =
   15.0115
MSE2 =
  726.5902
PSNR2 =
   19.5179
```


https://tieulun.hopto.org

(a)

(b)

(c)

Hình 3.24. *Kết quả thực hiện giảm nhiễu sử dụng phân tích Wavelet*
(a) Ảnh gốc
(b) Ảnh bị nhiễu (PSNR=15.01dB)
(c) Ảnh sau khi triệt nhiễu (PSNR=19.51dB)

BÀI TẬP CHƯƠNG 3

3.1. So sánh biến đổi Fourier, biến đổi Fourier thời gian ngắn và biến đổi Wavelet.

3.2. Giải thích vì sao biến đổi Wavelet phù hợp trong việc phân tích ảnh số thay vì dùng biến đổi Fourier.

3.3. Cho ma trận được tạo từ hàm `f=magic(4)`. Dùng hàm `dwt2` để phân tích Wavelet, sau đó thực hiện việc biến đổi Wavelet ngược để khôi phục tín hiệu ban đầu.

3.4. Một ảnh có kích thước $N \times N$. Ảnh cho có thể được phân tích Wavelet tối đa bao nhiêu mức.

3.5. Viết chương trình triệt nhiễu muối tiêu dùng các hàm Wavelet họ Daubechies khác nhau. So sánh kết quả, từ đó rút ra kết luận.

https://tieulun.hopto.org

Chương 4
LỌC ẢNH

Ảnh trước khi được xử lý trong hệ thống cần được tiền xử lý. Lọc là một trong những bước này. Chương này trình bày các bộ lọc không gian và tần số để áp dụng lên ảnh.

4.1. LỌC TRONG MIỀN KHÔNG GIAN

Một bộ lọc không gian bao gồm một tập các điểm ảnh và toán tử xác định được áp dụng lên điểm ảnh đang xem xét và các điểm ảnh lân cận của nó. Bộ lọc (toán tử lọc ảnh) được di chuyển đến từng điểm ảnh để tạo ra ảnh. Nếu toán tử được thực hiện trên ảnh là tuyến tính thì ta gọi đó là bộ lọc không gian tuyến tính. Ngược lại ta có bộ lọc phi tuyến.

Hình 4.1 mô tả cơ chế của phép lọc không gian với cửa sổ 3×3. Tại bất kỳ điểm ảnh nào (x, y), điểm ảnh tương ứng $g(x, y)$ của bộ lọc là tổng các tích của các hệ số của bộ lọc với các giá trị điểm ảnh tại vị trí tương ứng:

$$\begin{aligned} g(x,y) &= w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) \\ &+ \ldots + w(0,0)f(x,y) + \ldots + w(1,1)f(x+1,y+1) \end{aligned} \tag{4.1}$$

Lưu ý là hệ số trung tâm của bộ lọc, $w(0,0)$, tương ứng với pixel đang xét là (x, y). Với cửa sổ lọc có kích thước $m \times n$, chúng ta giả sử rằng $m = 2a + 1$ và $n = 2b + 1$, với a và b là các số nguyên dương. Nói cách khác trong phần này, chúng ta chỉ tập trung vào các bộ lọc có kích thước lẻ (nhỏ nhất là 3×3). Tổng quát, lọc không gian tuyến tính cho một ảnh có kích thước $M \times N$ với cửa sổ lọc là $m \times n$:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \tag{4.2}$$

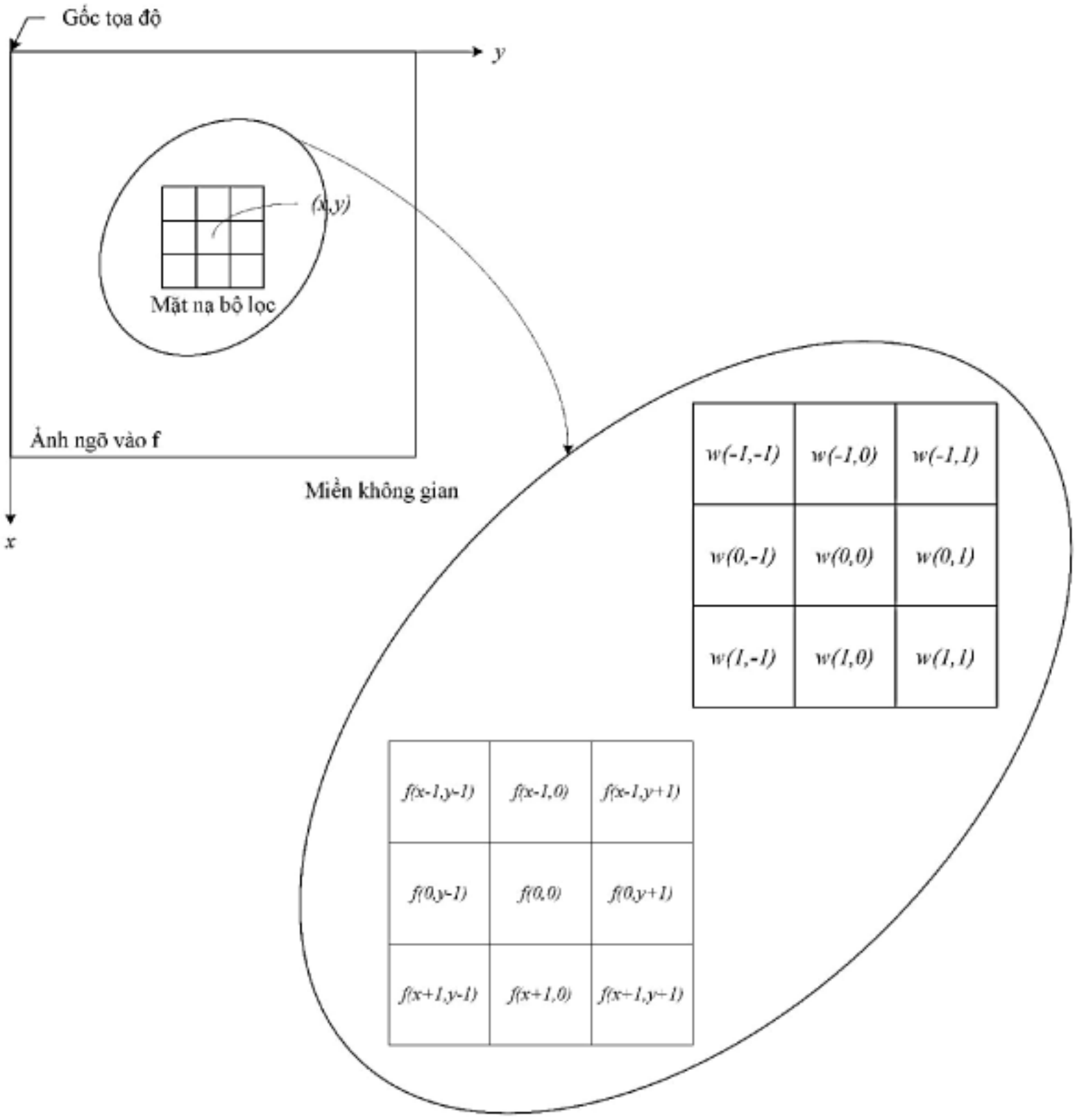
Với x và y được thay đổi sao cho mỗi điểm ảnh trong w đi qua từng điểm ảnh trong f.

Tương quan và tích chập không gian

Có hai khái niệm cần được hiểu rõ trước khi thực hiện lọc không gian: một là tính tương quan và hai là tích chập. Tương quan trong xử lý ảnh được thực hiện bằng cách di chuyển cửa sổ lọc trên toàn bộ ảnh và tính tổng của tích tại từng vị trí. Thực hiện tương tự khi tính tích chập,

https://tieulun.hopto.org

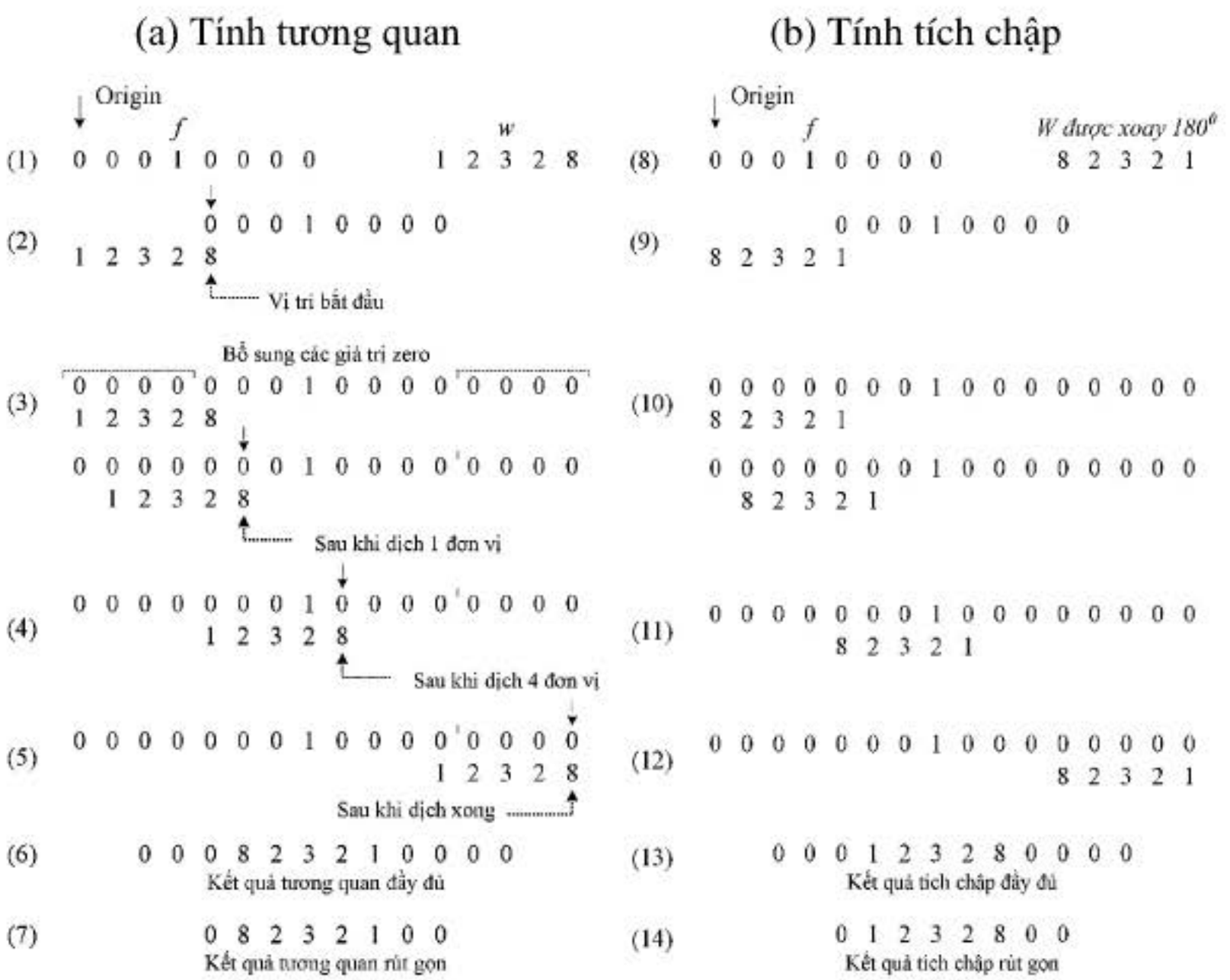
ngoại trừ bộ lọc được xoay 180^0. Tham khảo ví dụ bên dưới trong hình 4.2 để hiểu rõ hơn về cách thực hiện các phép tính này.



Hình 4.1. *Cơ chế hoạt động của bộ lọc không gian tuyến tính sử dụng cửa sổ lọc* $\mathbf{3 \times 3}$

Có hai điểm cần lưu ý trong phần này: thứ nhất, tương quan được xem như một hàm di chuyển của bộ lọc như trong hình 4.2(a); thứ hai, tính tương quan của bộ lọc với hàm được tính toán tương tự như tương quan nhưng giá trị bộ lọc được xoay đi 180^0 làm kết quả cũng được xoay 180^0 như trong hình 4.2(b).

https://tieulun.hopto.org



Hình 4.2. *Mô tả phương pháp tính tương quan và tích chập 1 chiều với xung đơn vị dạng rời rạc. Lưu ý là tương quan hay tích chập đều là các hàm dạng chuyển vị.*

Một cách tổng quát, chúng ta có tương quan của một bộ lọc $w(x,y)$ có kích thước $m \times n$ với ảnh $f(x,y)$ được cho, gọi là $w(x,y) \star f(x,y)$, được cho bởi công thức dưới đây:

$$w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \tag{4.3}$$

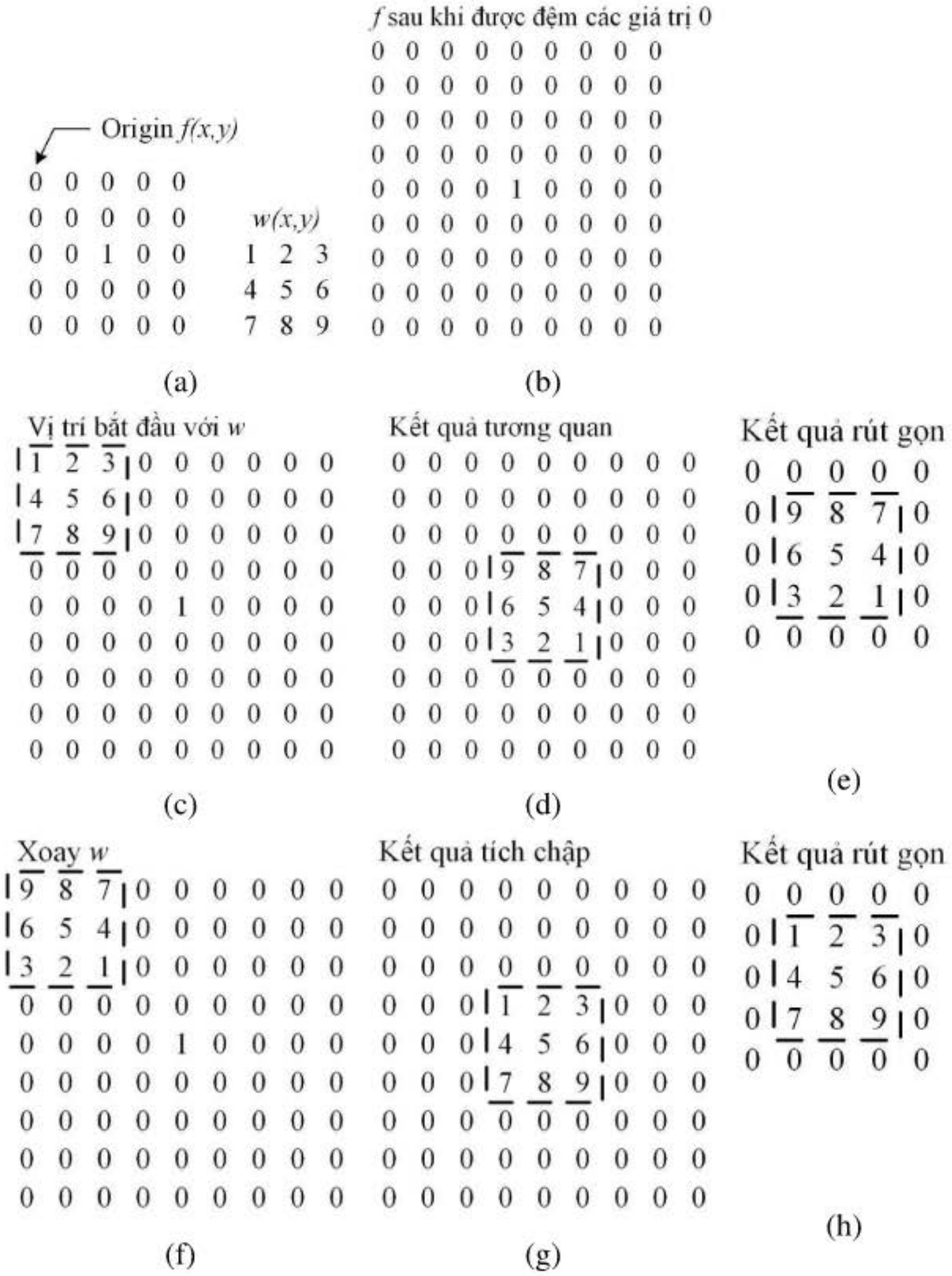
Biểu thức trên được thực hiện cho tất cả các giá trị của các biến x và y để đảm bảo rằng tất cả các phần tử trong w được di chuyển đến từng điểm ảnh trong f. Như đã đề cập trước đó, $a = (m-1)/2, b = (n-1)/2$ được giả định mang giá trị lẻ.

Với phương thức tương tự, tích chập của $w(x,y)$ và $f(x,y)$, được gọi là $w(x,y) \star f(x,y)$, được cho bởi công thức sau:

$$w(x,y) \star f(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x-s, y-t) \tag{4.4}$$

https://tieulun.hopto.org

Dấu trừ trong biểu thức của f biểu diễn cho việc xoay kết quả đi 180^0. Lật và xoay f thay vì w được thực hiện để cho việc tính toán được đơn giản và thuận tiện. Tham khảo thêm hình 4.3 mô tả việc thực hiện tính tương quan và tích chập dành cho bộ lọc 3×3.



Hình 4.3. *Tính tương quan (hàm giữa) và tích chập (hàng cuối) với bộ lọc hai chiều có kích thước* $\mathbf{3 \times 3}$.

https://tieulun.hopto.org

Trong MATLAB, hàm `imfilter` thực hiện phép lọc ảnh với số chiều tương ứng. Hàm `ones` cho phép tạo ra ma trận với các hệ số có giá trị bằng 1. Cú pháp hoàn chỉnh của hàm `imfilter` như sau:

```
g=imfilter(f,h,'Boundary Options','OutputSize','Correlation
and Convolution Options')
```

với `f` là ảnh ngõ vào, `g` là ảnh ngõ ra, `h` là cửa sổ lọc, `'Boundary Options'` cho phép lựa chọn đường bao của ảnh ngõ ra, `'Output Size'` cho phép lựa chọn kích thước ảnh ngõ ra và cơ chế thực hiện bộ lọc được thay đổi dựa vào thông số `'Correlation and Convolution Options'`. Các thông số trên có thể tham khảo trong bảng 4.1

Bảng 4.1. *Các thông số hàm* `imfilter`

Tùy chọn	Mô tả
Chế độ lọc	
`'corr'`	Sử dụng phép toán tương quan cho quá trình lọc. Đây là chế độ mặc định
`'conv'`	Sử dụng phép toán tích chập cho quá trình lọc
Đường bao	
`P`	Các đường bao của ảnh được mở rộng bằng cách gia tăng thêm điểm ảnh có giá trị P. Mặc định P bằng 0.
`'replicate'`	Kích thước của ảnh được mở rộng bằng cách tạo một bản sao ngay đường biên chính nó
`'symmetric'`	Kích thước của ảnh được mở rộng bằng cách ánh xạ các giá trị mức xám thông qua đường biên
`'circular'`	Kích thước của ảnh được mở rộng bằng cách dàn trải ảnh như một hàm có chu kỳ 2-D
Kích thước ảnh	
`'full'`	Ảnh ngõ ra có cùng kích thước với ảnh đã được mở rộng
`'same'`	Ảnh ngõ ra giữ nguyên kích thước. Đây là tùy chọn mặc định

MATLAB hỗ trợ việc tạo ra một số bộ lọc tuyến tính 2-D bằng hàm `fspecial` có cú pháp như sau:

```
w=fspecial('type',parameters);
```


https://tieulun.hopto.org

trong đó `'type'` đánh dấu loại bộ lọc và `parameters` định nghĩa một số đặc trưng của bộ lọc. Các bộ lọc không gian này được hỗ trợ bởi hàm `fspecial` được tổng kết trong bảng 4.2, bao gồm các thông số đặc trưng của từng loại.

Bảng 4.2.

Type	**Cú pháp và các thông số**
`'average'`	`fspecial('average',[r c])`. Bộ lọc trung bình có kích thước $r \times c$. Mặc định là 3×3.
`'disk'`	`fspecial('disk',r)`. Bộ lọc trung bình dạng hình tròn có bán kính `r`. Mặc định bán kính có giá trị là 5.
`'gaussian'`	`fspecial('gaussian',[r c],sig)`. Bộ lọc thông thấp Gaussian có kích thước $r \times c$ và độ lệch chuẩn sig (dương). Mặc định không khai báo thì $r \times c$ là 3×3 và $sig = 0.5$.
`'laplacian'`	`fspecial('laplacian',alpha)`. Bộ lọc Laplacian với hình dạng được đặc trưng bởi *alpha* nằm trong khoảng [0,1]. Mặc định `alpha` = 0.5
`'log'`	`fspecial('log',[r c],sig)`. Bộ lọc Laplacian của hàm Gauss có kích thước $r \times c$ và độ lệch chuẩn sig (dương). Mặc định không khai báo thì $r \times c$ là 3×3 và `sig = 0.5`.
`'motion'`	`fspecial('motion',len,theta)`. Đây là bộ lọc giả định sự dịch chuyển của camera với `len` là số điểm ảnh bị dịch chuyển và `theta` là hướng dịch chuyển. Mặc định thì `len` và `theta` có giá trị lần lượt là 9 và 0.
`'prewitt'`	`fspecial('prewitt')`. Bộ lọc Prewitt 3×3 giúp tăng cường góc theo chiều ngang bằng phương pháp xấp xỉ các gradient theo chiều dọc. Chuyển vị bộ lọc để thu được kết quả tăng cường góc theo chiều dọc.
`'sobel'`	`fspecial('sobel')`. Bộ lọc Sobel 3×3 giúp tăng cường góc theo chiều ngang dùng hiệu ứng làm trơn bằng phương pháp xấp xỉ các gradient theo chiều dọc. Chuyển vị bộ lọc để thu được kết quả tăng cường góc theo chiều dọc.

https://tieulun.hopto.org

Để hiểu rõ hơn các thành phần cấu tạo, một số bộ lọc được biểu diễn dưới dạng toán học:

- Với bộ lọc Gaussian

$$h_g(n_1, n_2) = e^{\frac{-(n_1^2+n_2^2)}{2\sigma^2}} \tag{4.5}$$

$$h(n_1, n_2) = \frac{h_g(n_1, n_2)}{\sum n_1 \sum n_2 h_g} \tag{4.6}$$

- Với bộ lọc Laplacian

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \tag{4.7}$$

$$\begin{bmatrix} \frac{\alpha}{1+\alpha} & \frac{1-\alpha}{1+\alpha} & \frac{\alpha}{1+\alpha} \\ \frac{1-\alpha}{1+\alpha} & \frac{-4}{1+\alpha} & \frac{1-\alpha}{1+\alpha} \\ \frac{\alpha}{1+\alpha} & \frac{1-\alpha}{1+\alpha} & \frac{\alpha}{1+\alpha} \end{bmatrix} \tag{4.8}$$

- Với bộ lọc Log (Laplacian of Gaussian)

$$h(n_1, n_2) = \frac{(n_1^2 + n_2^2 - 2\sigma^2) h_g(n_1, n_2)}{2\pi\sigma^6 \sum n_1 \sum n_2 h_g} \tag{4.9}$$

- Mặt nạ của bộ lọc Prewitt

$$h = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \tag{4.10}$$

- Mặt nạ của bộ lọc Sobel

$$h = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{4.11}$$

Ví dụ 4.1: Thực hiện lọc tuyến tính với các bộ lọc có sẵn trong MATLAB

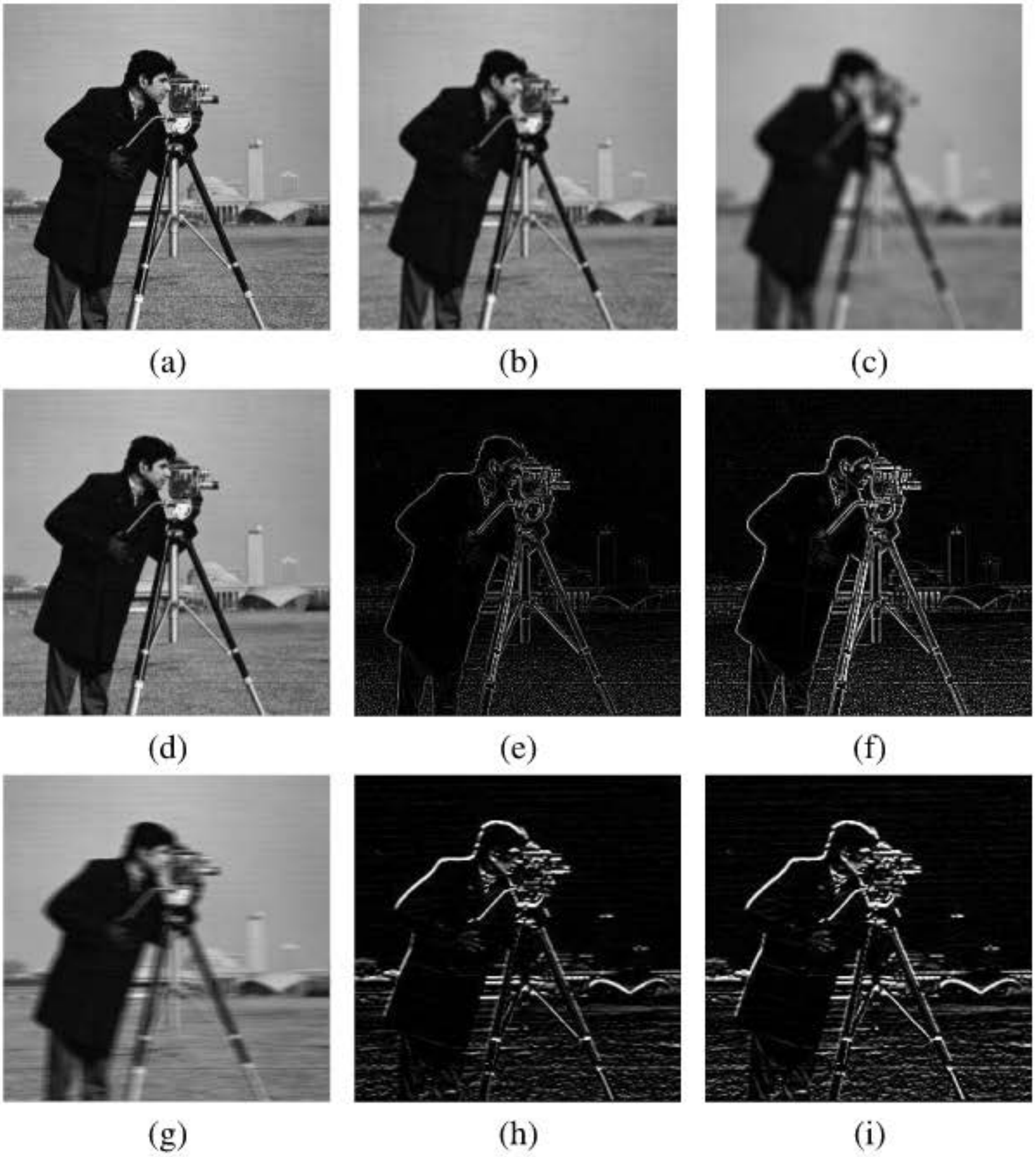
```
clear all;
f=imread('cameraman.bmp');
```


https://tieulun.hopto.org

```
ha=fspecial('average');
ga=imfilter(f,ha,'replicate');
hd=fspecial('disk');
gd=imfilter(f,hd,'replicate');
hg=fspecial('gaussian');
gg=imfilter(f,hg,'replicate');
hlap=fspecial('laplacian');
glap=imfilter(f,hlap,'replicate');
hlog=fspecial('log');
glog=imfilter(f,hlog,'replicate');
hm=fspecial('motion');
gm=imfilter(f,hm,'replicate');
hp=fspecial('prewitt');
gp=imfilter(f,hp,'replicate');
hs=fspecial('sobel');
gs=imfilter(f,hs,'replicate');
```

Trong thực tế, các bộ lọc này thường được sử dụng trong một chuỗi các tiến trình xử lý với mục đích cải thiện chất lượng ảnh, triệt nhiễu, tách biên và những ứng dụng khác. Do đó, rất ít khi thấy được ứng dụng độc lập được xây dựng từ các bộ lọc này.

https://tieulun.hopto.org

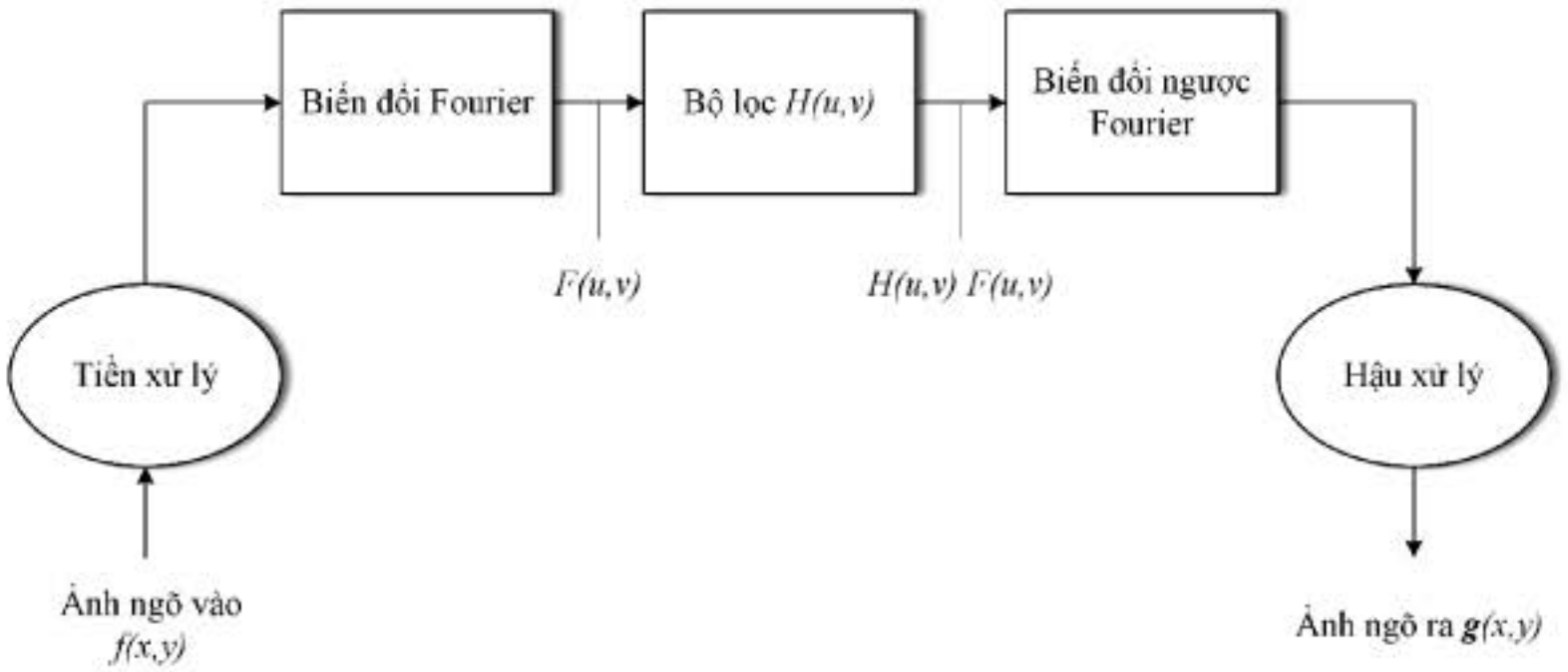


Hình 4.4. *Thực hiện lọc ảnh với nhiều bộ lọc khác nhau*
(a) Ảnh gốc; (b) Average; (c) Disk; (d) Gaussian; (e) Laplacian
(f) Log; (g) Motion; (h) Prewitt; (i) Sobel

4.2. LỌC TRONG MIỀN TẦN SỐ

Trong nhiều ứng dụng đòi hỏi cần phải xử lý trên miền tần số, biến đổi Fourier luôn là lựa chọn ưu tiên như một phép biến đổi phổ biến và dễ sử dụng. Một trong số nhiều ứng dụng phổ biến nhất là thực hiện lọc trong miền tần số, khi đó, ảnh được xem như tín hiệu hai chiều, cũng tồn tại các thành phần tần số khác nhau. Các bước thực hiện lọc trong miền tần số cơ bản được trình bày trong hình 4.5.

https://tieulun.hopto.org



Hình 4.5. *Các bước cơ bản khi thực hiện phép lọc trong miền tần số*

Dựa trên các bước được trình bày trong hình 4.5, quá trình lọc trên miền tần số có thể được viết dưới dạng một hàm riêng trong MATLAB với ảnh ngõ vào và hàm lọc cho trước. Một điều cần lưu ý, hàm bộ lọc được sử dụng cũng phải được biến đổi về miền tần số.

```
function g=dftfilt(f,H)
F=fft2(f,size(H,1),size(H,2));
G=H.*F;
g=ifft2(G);
f=real(g);
g=g(1:size(f,1),1:size(f,2))
end
```

Biến `H` ở đây là đáp ứng xung bộ lọc trên miền tần số được giả sử có kích thước gấp 4 lần ảnh ngõ vào. Trên miền không gian, quá trình lọc được xem xét như phép tính tích chập thì trên miền tần số là phép nhân trực tiếp giữa dữ liệu ngõ vào và giá trị bộ lọc. Sau khi thực hiện, ảnh ngõ ra thu được cần phải biến đổi ngược Fourier và lấy về đúng kích thước ban đầu.

Xây dựng bộ lọc tần số từ các bộ lọc không gian đã biết trước được thực hiện nhờ vào sự hỗ trợ của MATLAB với hàm `freqz2` với ngõ ra là các đáp ứng xung hữu hạn, vì thế, các bộ lọc được tạo ra từ hàm này còn được gọi là bộ lọc đáp ứng xung hữu hạn FIR (Finite Impulse Response). Cú pháp của hàm `freqz2` được mô tả như sau:

```
H=freqz2(h,R,C)
```

Với `h` là bộ lọc không gian hai chiều, `H` là bộ lọc đáp ứng tần số hai chiều. `R` và `C` là số hàng và cột mà ta mong muốn có ở `H`.

https://tieulun.hopto.org

***Ví dụ 4.2**:* Thực hiện việc lọc ảnh trong miền không gian dùng bộ lọc Sobel và lọc trong miền tần số dùng DFT

```
clear all;
f=imread('cameraman.tif');
h=fspecial('sobel');
sizeH=size(f)*2;
H=freqz2(h,sizeH);
H1=ifftshift(H);
gd=dftfilt(f,H1);
gs=imfilter(double(f),h);
d=abs(gd-gs);
gd=uint8(gd);imshow(gd);
gd=uint8(gd);imshow(gs);
```

Để kiểm tra sự tương đồng giữa ảnh sau khi lọc bằng hai phương pháp khác nhau, chúng ta tính độ sai lệch theo biểu thức sau:

$$d = |gs - gd|$$

Trong đó gs là ảnh được lọc trực tiếp trên miền không gian, gd là ảnh được lọc trên miền tần số. Kết quả của ví dụ 4.2 được trình bày trong hình 4.6.

(a)

(b)

(c)

***Hình 4.6.** Lọc ảnh bằng bộ lọc Sobel*

(a). Trên miền tần số

(b) Trên miền không gian

(c) Sai lệch giữa hai phương pháp

$\max D = 3.4106 \times 10^{-13}$

https://tieulun.hopto.org

Do việc tính FFT trong MATLAB giả định gốc tọa độ của hàm truyền tại vị trí góc trái phía trên của hình chữ nhật tần số, vì vậy, việc tính toán khoảng cách cần phải chú ý. Trong trường hợp này, gốc tọa độ cần phải được sắp xếp lại sao cho góc nằm ngay tại trung tâm bằng cách sử dụng hàm `fftshift` như đã trình bày trước đó. Để cung cấp một sơ đồ lưới dùng tính toán khoảng cách và cho các ứng dụng tương tự, hàm `dftuv` được viết như sau:

```
function [U,V]=dftuv(m,n)
u=0:(m-1);
v=0:(n-1);
idx=find(u>m/2);
u(idx)=u(idx)-m;
idy=find(v>n/2);
v(idy)=v(idy)-n;
[V,U]=meshgrid(v,u);
end
```

Để minh họa, ta tính bình phương khoảng cách từ mỗi điểm trong hình chữ nhật có kích thước 8 × 5 đến gốc:

```
[U,V]=dftuv(8,5);
D=U.^2+V.^2
D =
     0     1     4     4     1
     1     2     5     5     2
     4     5     8     8     5
     9    10    13    13    10
    16    17    20    20    17
     9    10    13    13    10
     4     5     8     8     5
     1     2     5     5     2
```

Giá trị khoảng cách bằng không tại góc trên bên trái và các giá trị lớn nhất tại trung tâm của hình chữ nhật được giải thích như ở hình 3.2(a). Hiển nhiên, ta có thể dùng hàm `fftshift` để đạt được các giá trị khoảng cách với mối liên hệ đến trung tâm của hình chữ nhật dùng biểu diễn tần số (frequency retangle).

```
fftshift(D)
```


https://tieulun.hopto.org

```
ans =

    20    17    16    17    20
    13    10     9    10    13
     8     5     4     5     8
     5     2     1     2     5
     4     1     0     1     4
     5     2     1     2     5
     8     5     4     5     8
    13    10     9    10    13
```

4.2.1. Lọc thông thấp

Bộ lọc thông thấp lý tưởng (Ideal LowPass Filter - ILPF) có hàm truyền như sau:

$$H(u,v) = \begin{cases} 1 & D(u,v) \le D_0 \\ 0 & D(u,v) > D_0 \end{cases} \tag{4.12}$$

Với D_0 được chỉ định là một giá trị không âm, $D(u,v)$ là khoảng cách từ điểm (u,v) đến trung tâm của bộ lọc. Cần phải nhớ rằng quỹ đạo của các điểm $D(u,v) = D_0$ là hình tròn. Mặc dù đây không phải là bộ lọc thực tiễn do những yếu tố khách quan nhưng nó vẫn có thể được mô phỏng trên MATLAB bằng các hàm cho trước.

Bộ lọc thông thấp Butterworth (Butterworth LowPass Filter - BLPF) bậc n với tần số cắt tại giá trị khoảng cách D_0, có hàm truyền được cho như sau:

$$H(u,v) = \frac{1}{1 + [D(u,v)/D_0]^{2n}} \tag{4.13}$$

Không giống như hàm ILPF, hàm BLPF không có được góc cắt tuyệt đối ngay tại D_0. Để có được bộ lọc, thông thường người ta định nghĩa tần số cắt đặt tại các điểm mà $H(u,v) = 0.5$ (giảm 50% so với trị lớn nhất là 1) khi $D(u,v) = D_0$.

Hàm chuyển đổi của bộ lọc thông thấp Gaussian (GLPF) được cho như sau:

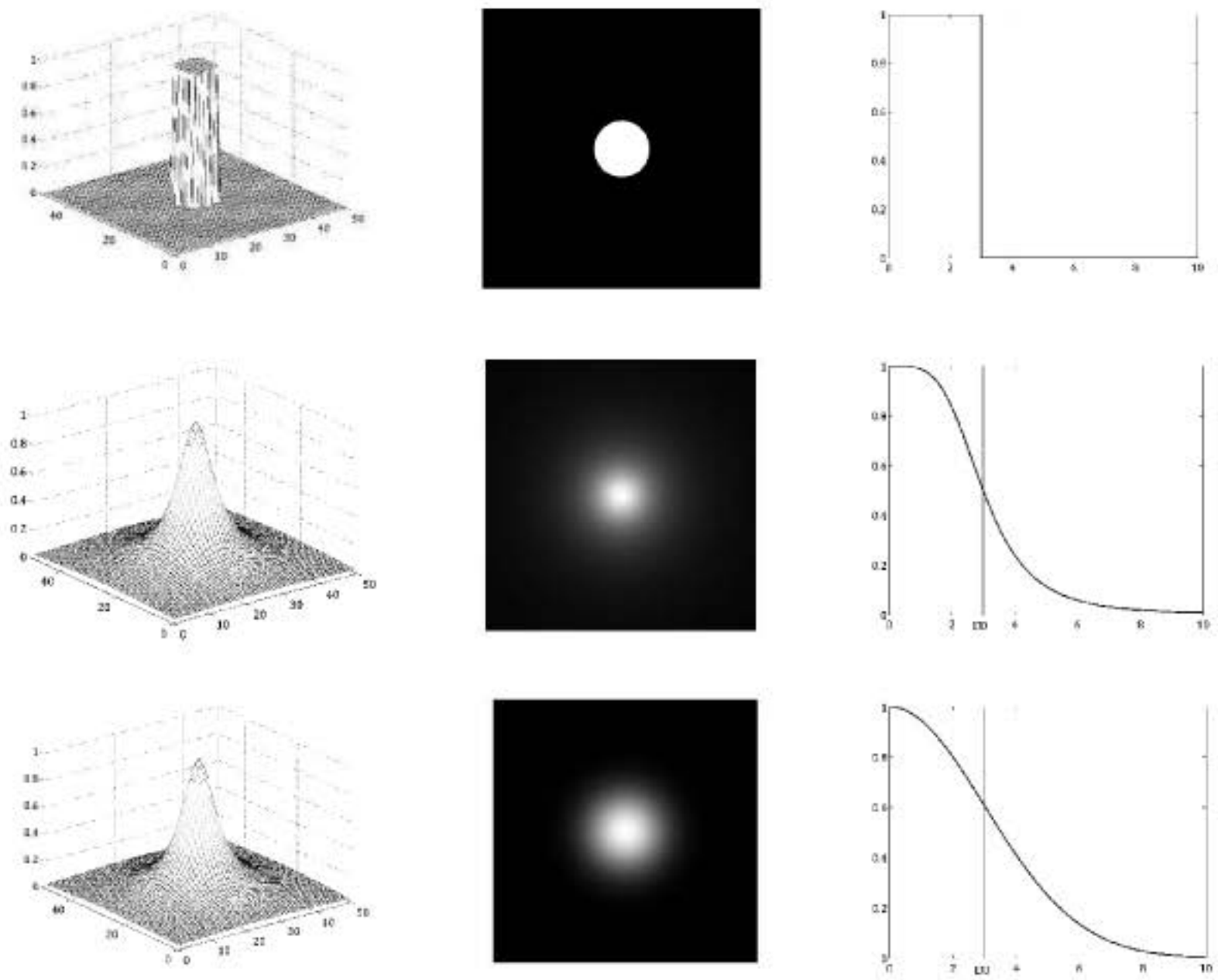
$$H(u,v) = e^{-D^2(u,v)/2\sigma^2} \tag{4.14}$$

Với σ là độ lệch chuẩn. Bằng cách cho $\sigma = D_0$, chúng ta có được dạng triển khai như sau:

$$H(u,v) = e^{-D^2(u,v)/2D_0^2} \tag{4.15}$$

https://tieulun.hopto.org

Khi $D(u,v) = D_0$, giá trị ngay tại tần số cắt là 0.0607. Hình 4.7 mô tả dạng 3-D cũng như biểu đồ đáp ứng tần số của từng bộ lọc.



Hình 4.7. *Cột trái: minh họa 3-D của đáp ứng cửa sổ lọc. Cột giữa: bộ lọc được biểu diễn như một ảnh. Cột phải: đáp ứng tần số với tần số cắt D_0. Hàng đầu: lọc thông thấp lý tưởng. Hàng giữa: lọc thông thấp Butterworth. Hàng cuối: lọc thông thấp Gaussian.*

Từ đây, ta có thể tự xây dựng một hàm chuyên dành để thực hiện lọc thông thấp `lpfilter` với ba bộ lọc cơ bản cho trên:

```
function [H,D]=lpfilter(type,M,N,DO,n)
[U,V]=dftuv(M,N);
D=sqrt(U.^2+V.^2);
switch type
    case 'ideal'
        H=double(D<=DO);
    case 'btw'
        if nargin==4
            n=1;
```


https://tieulun.hopto.org

```
        end
        H=1./(1+(D./DO).^(2*n));
    case 'gaussian'
        H=exp(-(D.^2)./(2*(DO^2)));
    otherwise
        error('Unknown filter type');
end
```

Một trong những phương pháp hữu hiệu để biểu diễn bộ lọc trong không gian ba chiều là sử dụng hàm `mesh` với cú pháp cho sau:

```
mesh(H)
```

Hàm này sẽ cho ra một biểu đồ với `x=1:M` và `y=1:N`, với `[M,N]=size(H)`. Biểu đồ này về cơ bản là không chấp nhận mức độ dày đặc nếu M và N đều lớn, trong đó, chúng ta vẽ từng điểm bằng cách dùng hàm

```
mesh(H(1:k:end,1:k:end))
```

Mặc định thì biểu đồ dạng hàm `mesh` sẽ có màu, nhưng để đơn giản, chúng ta thay đổi sang dạng trắng đen bằng cách thiết lập

```
colormap([0 0 0])
```

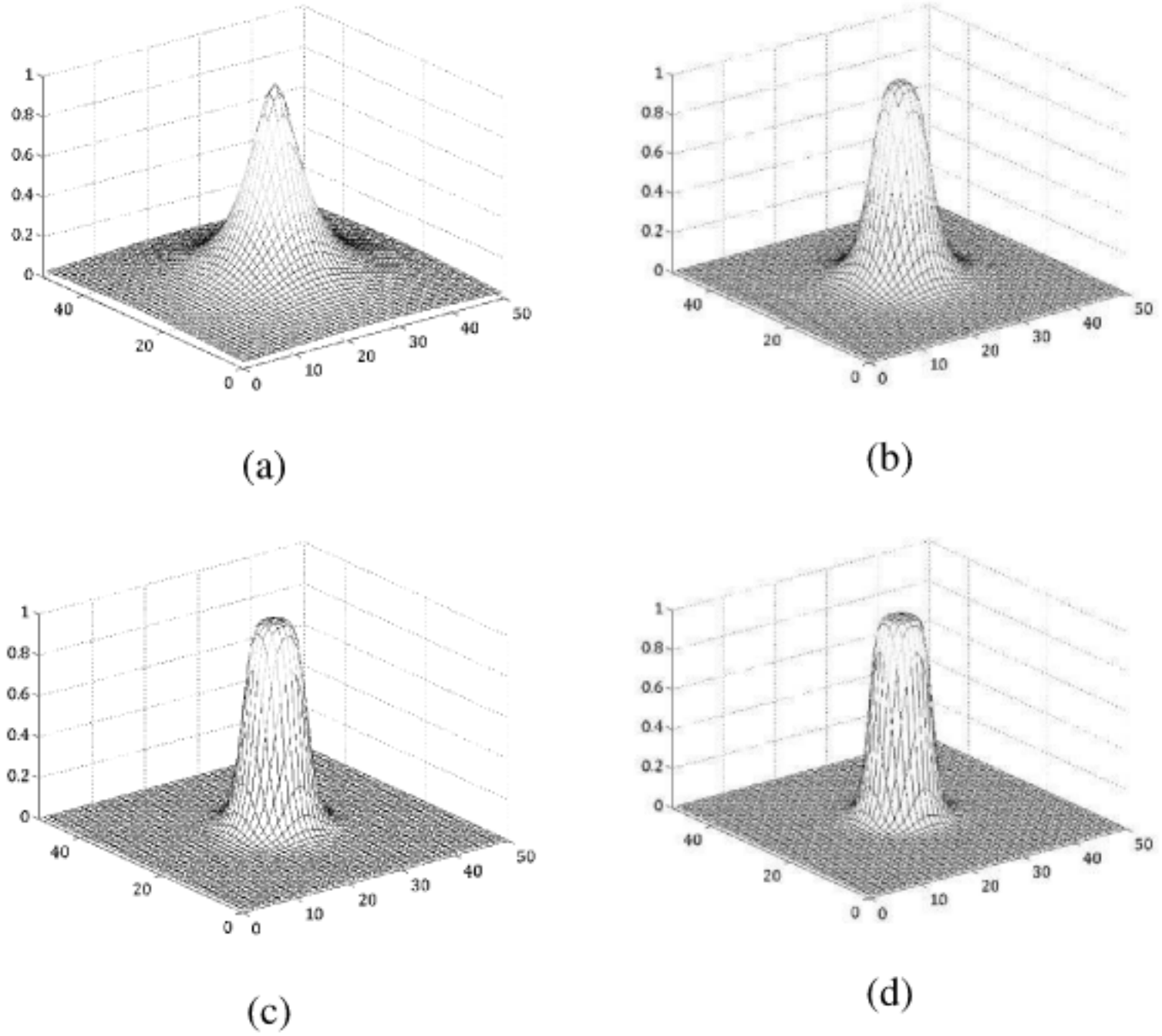
Một số hàm thường sử dụng trong vẽ biểu đồ như: `grid on` (kích hoạt dạng lưới), `axis` (cho phép hiển thị hay không giá trị các trục).

***Ví dụ 4.3**:* Vẽ biểu đồ dạng mesh cho hàm lọc thông thấp Butterworth với các giá trị bậc khác nhau.

```
clear all;
BLPF1=fftshift(lpfilter('btw',500,500,50,1));
mesh(BLPF1(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);
saveas(gcf, 'Hinh3.7a.tif', 'tif');

BLPF2=fftshift(lpfilter('btw',500,500,50,2));
figure;mesh(BLPF2(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);
saveas(gcf, 'Hinh3.7b.tif', 'tif');
```


https://tieulun.hopto.org

```
BLPF3=fftshift(lpfilter('btw',500,500,50,3));
figure;mesh(BLPF3(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);
saveas(gcf, 'Hinh3.7c.tif', 'tif');

BLPF4=fftshift(lpfilter('btw',500,500,50,4));
figure;mesh(BLPF4(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);
saveas(gcf, 'Hinh3.7d.tif', 'tif');
```



Hình 4.8. *Biểu đồ dạng mesh của các bộ lọc Butterworth với bậc khác nhau*

(a) Bậc một *(b) Bậc hai* *(c) Bậc ba* *(d) Bậc bốn*

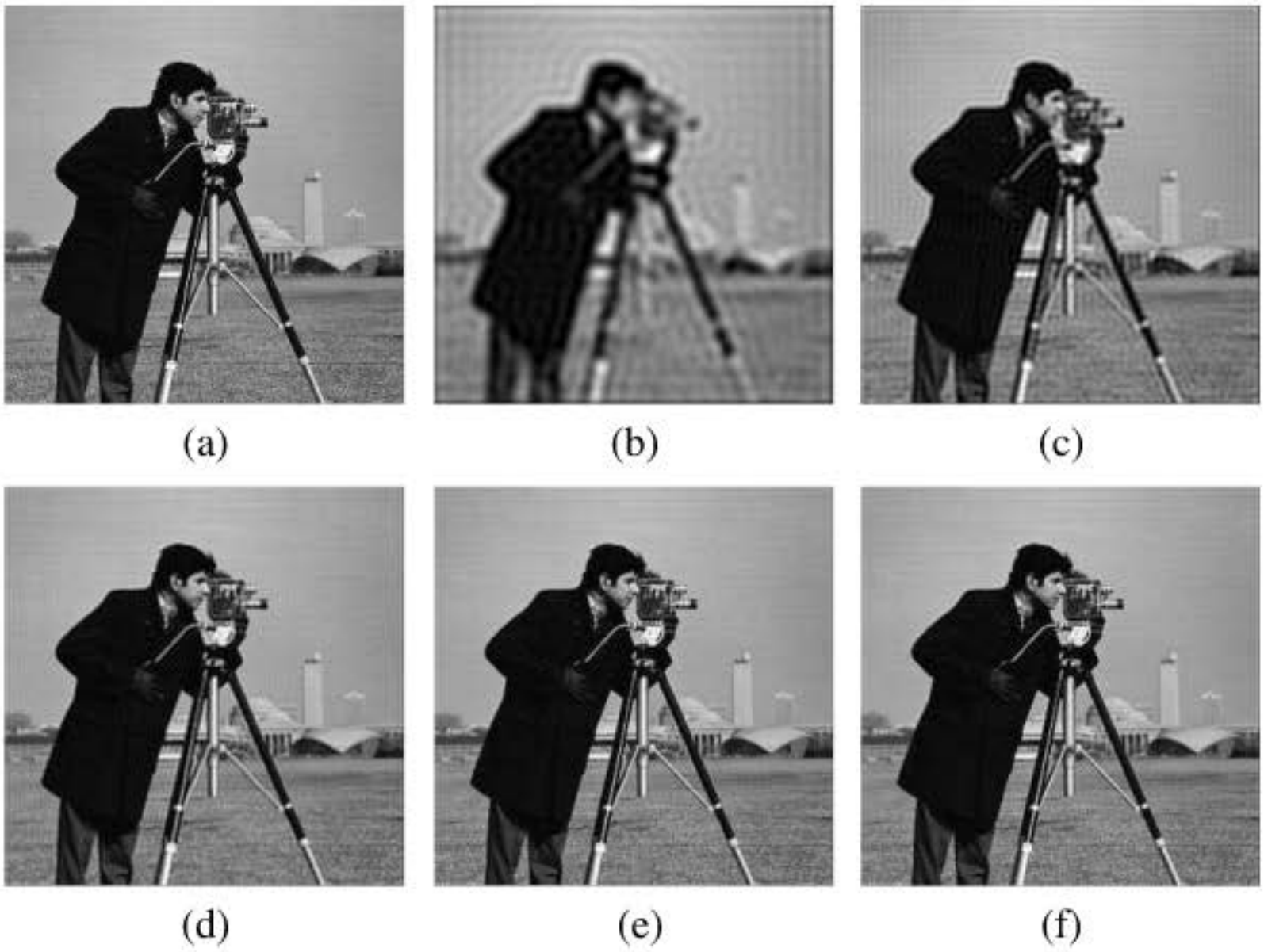
Ví dụ 4.4*:* Khảo sát bộ lọc thông thấp với nhiều tần số cắt khác nhau. So sánh kết quả.

```
clear all;
f=imread('cameraman.bmp');
```


https://tieulun.hopto.org

```
D01=1/5*size(f,1);
[M,N]=size(f);
[H,D]=lpfilter('ideal',2*M,2*N,D01);
g=dftfilt(f,H);

D02=2/5*size(f,1);
[M,N]=size(f);
[H,D]=lpfilter('ideal',2*M,2*N,D02);
g=dftfilt(f,H);

D03=3/5*size(f,1);
[M,N]=size(f);
[H,D]=lpfilter('ideal',2*M,2*N,D03);
g=dftfilt(f,H);

D04=4/5*size(f,1);
[M,N]=size(f);
[H,D]=lpfilter('ideal',2*M,2*N,D04);
g=dftfilt(f,H);

D05=size(f,1);
[M,N]=size(f);
[H,D]=lpfilter('ideal',2*M,2*N,D04);
g=dftfilt(f,H);
```


https://tieulun.hopto.org



Hình 4.9. *Ảnh cameraman* $\boldsymbol{M \times N}$ *được lọc thông thấp dùng bộ lọc lý tưởng với tần số cắt khác nhau*

(a) Ảnh gốc *(b)* $D_0 = \frac{M}{5}$ *(c)* $D_0 = \frac{2M}{5}$

(d) $D_0 = \frac{3M}{5}$ *(e)* $D_0 = \frac{4M}{5}$ *(f)* $D_0 = M$

4.2.2. Lọc thông cao

Không như bộ lọc thông thấp làm mờ ảnh, bộ lọc thông cao thường dùng trong các ứng dụng cần làm ảnh trở nên sắc nét hơn bằng cách loại bỏ thành phần tần số thấp thông qua biến đổi Fourier. Cho bộ lọc thông thấp có hàm truyền $H_{lp}(u, v)$, chúng ta dễ dàng có được bộ lọc thông cao bằng cách đơn giản sau:

$$H_{hp}(u, v) = 1 - H_{lp}(u, v) \tag{4.16}$$

Do đó, với hàm `lpfilter` được xây dựng trước đó, ta hoàn toàn có thể sử dụng lại để xây dựng hàm dành riêng cho lọc thông cao

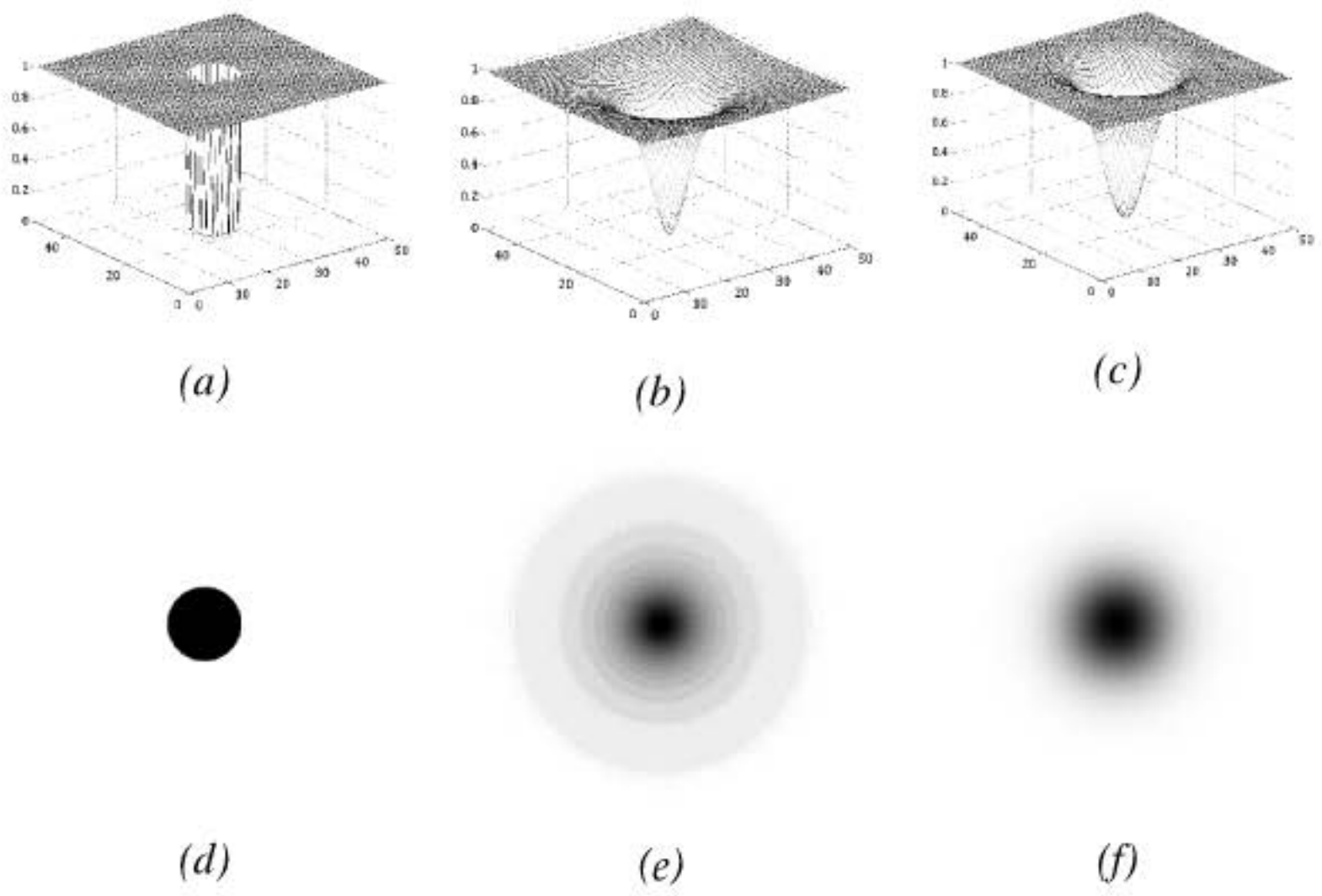
```
function H=hpfilter(type,M,N,D0,n)
if nargin==4
    n=1;
end
```


https://tieulun.hopto.org

```
Hlp=lpfilter(type,M,N,D0,n);
H=1-Hlp;
end
```

Ví dụ 4.5: Biểu diễn các bộ lọc thông cao dưới dạng mesh và ảnh.

```
clear all;close all;clc
ILPF=fftshift(hpfilter('ideal',500,500,50));
mesh(ILPF(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);

BLPF=fftshift(hpfilter('btw',500,500,50));
mesh(BLPF(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);

GLPF=fftshift(hpfilter('gaussian',500,500,50));
mesh(GLPF(1:10:500,1:10:500));
axis([0 50 0 50 0 1]);
```



Hình 4.10. *Biểu diễn các bộ lọc thông cao. Hàng trên: biểu diễn dạng sơ đồ mesh. Hàng dưới: biểu diễn dưới dạng ảnh. Cột trái: bộ lọc thông cao lý tưởng. Cột giữa: bộ lọc thông cao Butterworth. Cột phải: bộ lọc thông cao Gaussian*

https://tieulun.hopto.org

Các bộ lọc thông cao chuẩn hóa về không các thành phần một chiều dc, do đó, vô tình làm giảm trị trung bình của ảnh về không. Để giải quyết vấn đề này chúng ta thêm vào thành phần offset một chiều vào bộ lọc thông cao. Khi thành phần offset này kết hợp cùng phép nhân giữa bộ lọc và hằng số lớn hơn 1, phương pháp này gọi là lọc thông cao tăng cường (high frequency emphasis filtering) do hằng số nhân làm nổi bật các tần số cao. Hàm biểu diễn cho quá trình lọc thông cao tăng cường được trình bày như sau:

$$H_{hfe}(u,v) = a + bH_{hp}(u,v) \tag{4.17}$$

Với a là thành phần offset, b là hằng số nhân và $H_{hp}(u,v)$ là hàm truyền của bộ lọc thông cao.

Ví dụ 4.6: Thực hiện lọc ảnh dùng bộ lọc thông cao tăng cường.

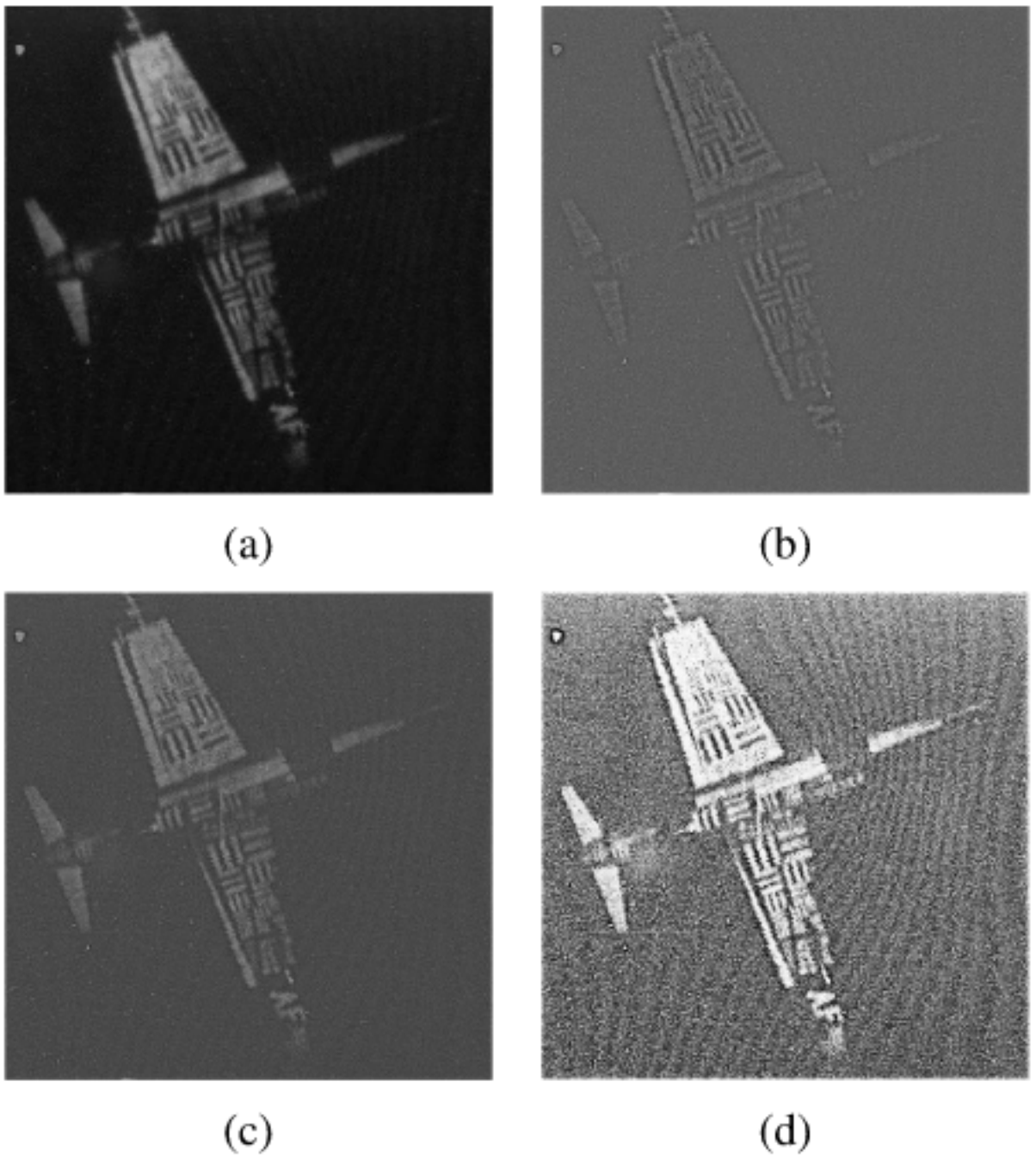
```
clear all;
f=imread('satellite.tiff');
[M N]=size(f);
D0=0.1*size(f,1);
HBW=hpfilter('btw',2*M,2*N,D0);
H=0.5+2*HBW;
gbw=dftfilt(f,HBW);
gbw=gscale(gbw);
ghf=dftfilt(f,H);
ghf=gscale(ghf);
ghe=histeq(ghf,256);
```

Trong ví dụ 4.6, hàm `gscale` thực hiện việc chuẩn hóa và đổi kiểu dữ liệu chứa cường độ mức xám của ảnh. Hàm `gscale` này có thể tham khảo như dưới dây:

```
function g=gscale(f)
fmin=min(f(:));
fmax=max(f(:));
for i=1:size(f,1)
    for j=1:size(f,2)
g(i,j)=255*(f(i,j)-fmin)/(fmax-fmin);
    end
end
g=uint8(g);
end
```


https://tieulun.hopto.org

Hình 4.11 trình bày kết quả trong ví dụ 4.6 với ảnh gốc satellite khá mờ. Hình 4.11(b) trình bày ảnh sau khi lọc thông cao dùng bộ lọc Butterworth, trong khi đó, ảnh sau khi dùng bộ lọc thông cao tăng cường được trình bày trong hình 4.11(c). Ưu điểm của bộ lọc này là một số thành phần tần số thấp vẫn còn được giữ lại, và chúng được làm rõ hơn bằng thuật toán cân bằng histogram như trong hình 4.11(d).



Hình 4.11. *Lọc thông cao ảnh Satellite*

(a) Ảnh gốc

(b) Lọc thông cao Butterworth

(c) Lọc thông cao tăng cường

(d) Cân bằng histogram ảnh (c)

BÀI TẬP CHƯƠNG 4

4.1. Trình bày ma trận mặt nạ khác nhau có được từ hàm `fspecial`. Vẽ đáp ứng tần số của các bộ lọc này và cho nhận xét.

4.2. Thực hiện lọc thông thấp Butterworth và Gaussian với nhiều tần số cắt khác nhau. Từ đó rút ra kết luận.

4.3. Ảnh được cho bị ảnh hưởng bởi nhiễu Gaussian. Thực hiện lọc thông thấp lý tưởng cho ảnh. Khảo sát chất lượng ảnh sau khi lọc dùng thông số PSNR.

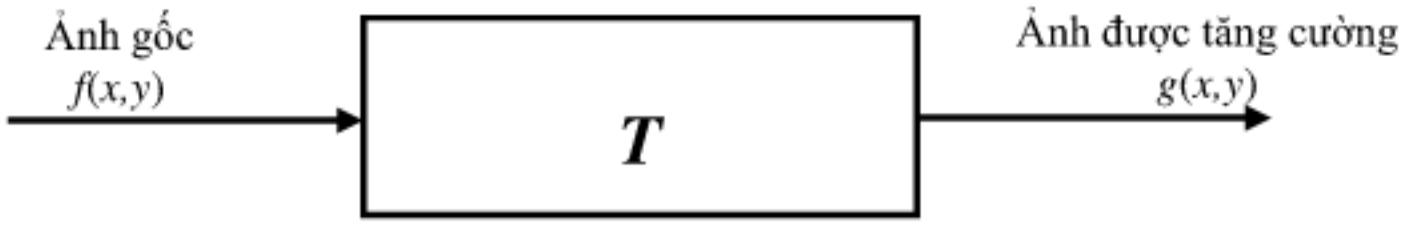
4.4. Vẽ đáp ứng tần số của bộ lọc thông thấp và thông cao với cùng tần số cắt. Cho nhận xét.

https://tieulun.hopto.org

Chương 5

TĂNG CƯỜNG ẢNH

Tăng cường ảnh là làm cho ảnh có chất lượng tốt hơn theo ý đồ sử dụng. Ảnh thu được thường bao gồm nhiễu cần phải loại bỏ hoặc ảnh không sắc nét, bị mờ hoặc cần làm rõ các đặc trưng của ảnh như: thay đổi độ tương phản, giảm nhiễu, làm trơn biên, khuếch đại ảnh, điều khiển mức xám... Những kỹ thuật tăng cường và tái tạo được áp dụng cho một ảnh ngõ vào thông qua những thuật toán và cho ra ảnh tăng cường hay tái tạo ở ngõ ra, như trong hình 5.1.



Hình 5.1. *Sơ đồ của kỹ thuật tăng cường ảnh*

Những kỹ thuật tăng cường và tái tạo ảnh có thể được thực hiện trong miền điểm, miền không gian hay trong miền tần số. Những kỹ thuật xử lý điểm được thực hiện cho một điểm ảnh (pixel) trong ảnh gốc tại tọa độ (x,y) để cho ra một điểm ảnh tương ứng tại tọa độ (x,y) trong ảnh tăng cường. Điều này có nghĩa chỉ pixel trong ảnh gốc giữ vai trò trong việc tính toán giá trị pixel tương ứng trong ảnh tăng cường và những pixel này được xác định ở cùng một vị trí tọa độ trong ảnh gốc. Kỹ thuật tăng cường ảnh trong không gian hay xử lý mặt nạ được phân nhóm theo công dụng: làm trơn nhiễu, nổi biên. Để làm trơn ảnh hay tách nhiễu, người ta sử dụng các bộ lọc tuyến tính (lọc trung bình, thông thấp) hay lọc phi tuyến (trung vị, giả trung vị, lọc đồng hình). Để làm nổi biên (ứng với tần số cao), người ta dùng các bộ lọc thông cao, bộ lọc Laplace. Một số phép biến đổi có tính toán phức tạp được chuyển sang miền tần số để thực hiện, kết quả cuối cùng được chuyển trở lại miền không gian nhờ các biến đổi ngược.

5.1. XỬ LÝ ĐIỂM

Xử lý điểm ảnh là biến đổi giá trị một điểm ảnh dựa vào giá trị của chính nó mà không hề dựa vào các điểm ảnh khác. Sự lựa chọn chính xác phép biến đổi nào phụ thuộc vào mục đích của vấn đề xử lý điểm. Xử lý điểm thường thực hiện trên một ảnh để cải thiện chất lượng của nó bằng việc đảm bảo dãy mức xám (gray-level) của nó.

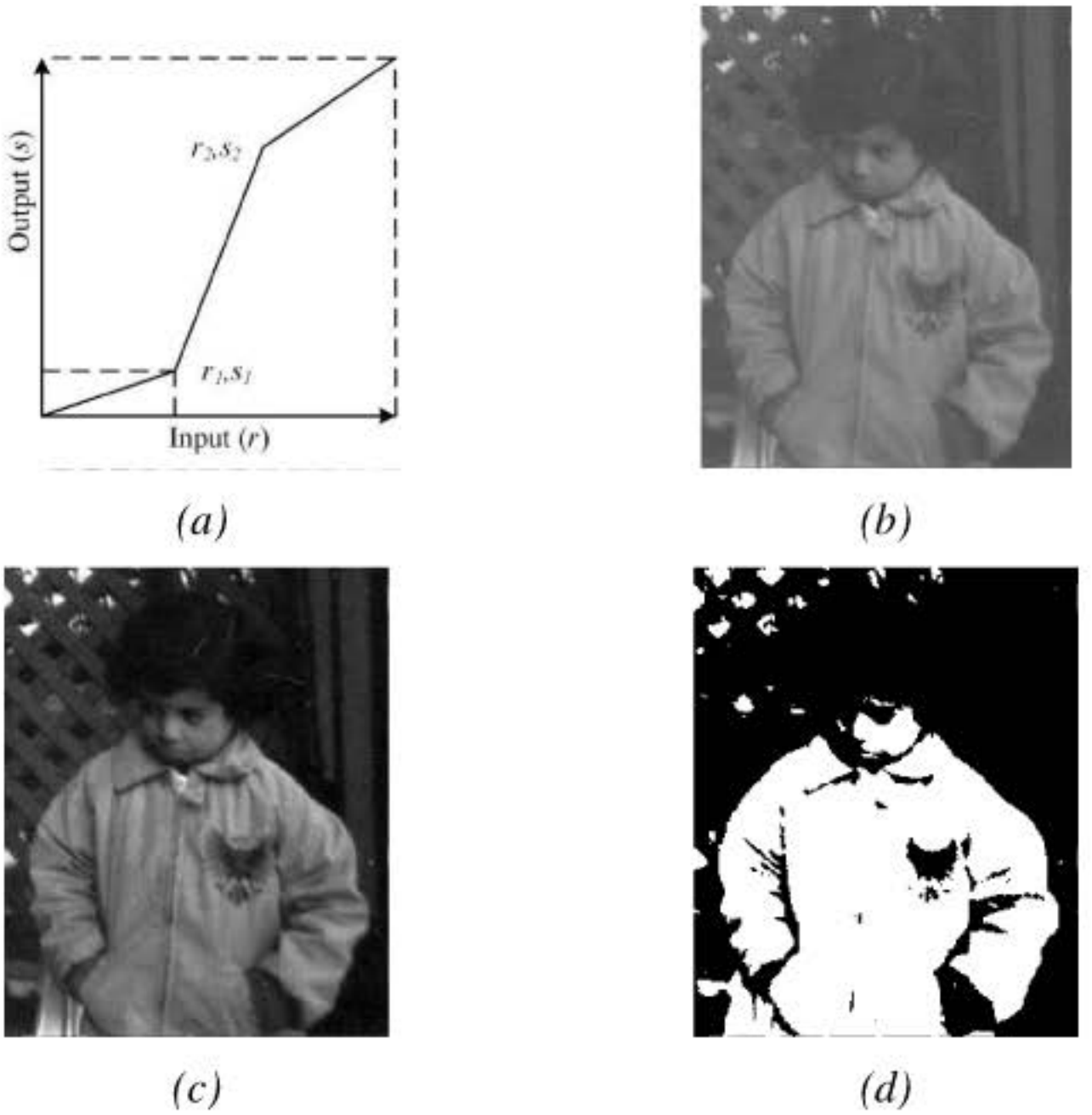
https://tieulun.hopto.org

5.1.1. Mở rộng độ tương phản

Một trong những hàm tuyến tính từng đoạn đơn giản nhất là biến đổi dạng mở rộng độ tương phản. Những ảnh có độ tương phản thấp là kết quả của quá trình chụp trong môi trường thiếu sáng hay có thể do chất lượng của cảm biến hình ảnh, hay thậm chí do lỗi của người sử dụng trong quá trình thiết lập thông số. Mở rộng độ tương phản là tiến trình làm tăng khoảng giá trị mức xám trong ảnh giúp ảnh hiển thị tốt hơn trên các thiết bị hiển thị trong môi trường ánh sáng mạnh.

Hình 5.2(a) biểu diễn ánh xạ cường độ mức xám của dạng mở rộng độ tương phản, trong đó có hai điểm (r_1, s_1) và (r_2, s_2) cho phép điều chỉnh hình dạng của hàm biến đổi. Nếu trường hợp $r_1 = s_1$ và $r_2 = s_2$ thì không có bất kỳ sự thay đổi nào tại ảnh ngõ ra. Còn nếu $r_1 = r_2$, $s_1 = 0$ và $s_2 = L-1 = 255$ thì ta có hàm lấy ngưỡng. Việc thay đổi các giá trị của hai điểm nêu trên cho phép tạo ra ảnh có độ tương phản thay đổi khác nhau. Trong hình 5.2(b) trình bày ảnh có độ tương phản thấp và tương ứng là ảnh sau khi được mở rộng trong hình 5.2(c) với thông số $(r_1, s_1) = (r_{min}, 0)$ và $(r_2, s_2) = (r_{max}, L-1)$ với r_{min} và r_{max} đánh dấu giá trị mức xám nhỏ nhất và lớn nhất của ảnh ngõ ra. Do đó, hàm này sẽ mở rộng khoảng giá trị mức xám của ảnh trên một khoảng lớn nhất có thể [0, *L*-1]. Cuối cùng là ảnh trong hình 5.2(d) là kết quả khi dùng hàm lấy ngưỡng với $(r_1, s_1) = (m, 0)$ và $(r_2, s_2) = (m, L-1)$ với *m* là giá trị mức xám trung bình của ảnh ngõ vào.

https://tieulun.hopto.org



Hình 5.2. *Phép mở rộng độ tương phản*

(a) Hàm biến đổi; (b) Ảnh gốc; (c) Ảnh sau khi dùng phép mở rộng độ tương phản; (d) Ảnh sau khi dùng hàm lấy ngưỡng

Ví dụ5.1: Thực hiện phép mở rộng độ tương phản trong hình 5.2

```
clear all;
f=imread('pout.tif');
r1=min(min(f));
r2=max(max(f));
for i=1:size(f,1)
    for j=1:size(f,2)
if f(i,j)==r1
            g1(i,j)=0;
        elseif f(i,j)==r2
            g1(i,j)=255;
else
            g1(i,j)=255/(r2-r1)*(f(i,j)-r1);
```


https://tieulun.hopto.org

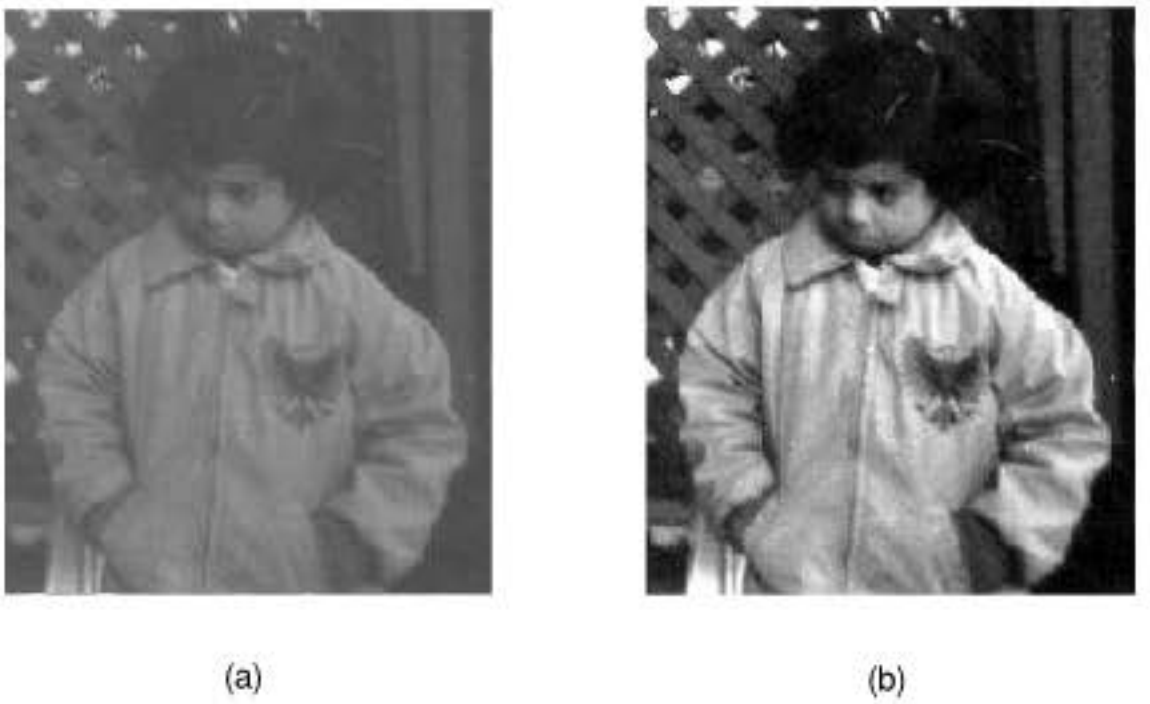
```
end
    end
end
g1=uint8(g1);
m=mean2(f);
for i=1:size(f,1)
    for j=1:size(f,2)
        if f(i,j) <= m
g2(i,j)=0;
        else
            g2(i,j)=255;
end
    end
end
g2=uint8(g2);
```

Trong MATLAB, hàm `stretchlim` cho phép tìm các giới hạn (cận trên và cận dưới giá trị mức xám) dùng trong phép mở rộng độ tương phản cùng với hàm `imadjust`.

Ví dụ 5.2: Mở rộng độ tương phản dùng hàm `imadjust`

```
clear all;
f=imread('pout.tif');
g=imadjust(f,stretchlim(f));
figure;
subplot(1,2,1)
imshow(f)
xlabel('(a)')
subplot(1,2,2)
imshow(g)
xlabel('(b)')
```


https://tieulun.hopto.org



(a) (b)

Hình 5.3. *Tăng cường ảnh kết hợp hàm stretchlim và imadjust: (a) Ảnh gốc; (b) Ảnh đã mở rộng độ tương phản*

5.1.2. Xử lý lược đồ (Histogram)

Histogram của một ảnh xám có L mức xám khác nhau, là một hàm rời rạc, có biểu thức:

$$h(r_k) = n_k \tag{5.1}$$

trong đó r_k là giá trị mức xám thứ k trong đoạn [0, L-1] và n_k là số pixels có giá trị mức xám là r_k. Ví dụ với đoạn [0, 255], $r_0 = 0$, $r_1 = 1$…

Histogram thường được chuẩn hóa. Với n là tổng số pixels của ảnh, histogram chuẩn hóa được tính bằng công thức:

$$p(r_k) = \frac{h(r_k)}{n} = \frac{n_k}{n} \tag{5.2}$$

Có thể xem $p(r_k)$ là hàm mật độ xác suất của r_k, cho biết khả năng xuất hiện tương ứng của từng giá trị mức xám. Sử dụng hàm `imhist` để thực hiện histogram của ảnh:

```
h = imhist(f, b)
```

Trong đó, `f` là ảnh cần thực hiện histogram, `b` là số đoạn biểu diễn (mặc định là 256). Nếu `b` nhỏ, giả sử `b` = 2, thì thang cường độ sẽ chia làm 2 khoảng: 0 đến 127, 128 đến 255, trong đó `h(1)` là số pixels có giá trị trong đoạn [0, 127] và `h(2)` là số pixels có giá trị trong đoạn [127, 255]. Nếu không có thông số đầu ra thì hàm `imhist` biểu diễn cho histogram của ảnh.

Ngoài ra có thể có được hàm $p(r_k)$ qua dòng lệnh:

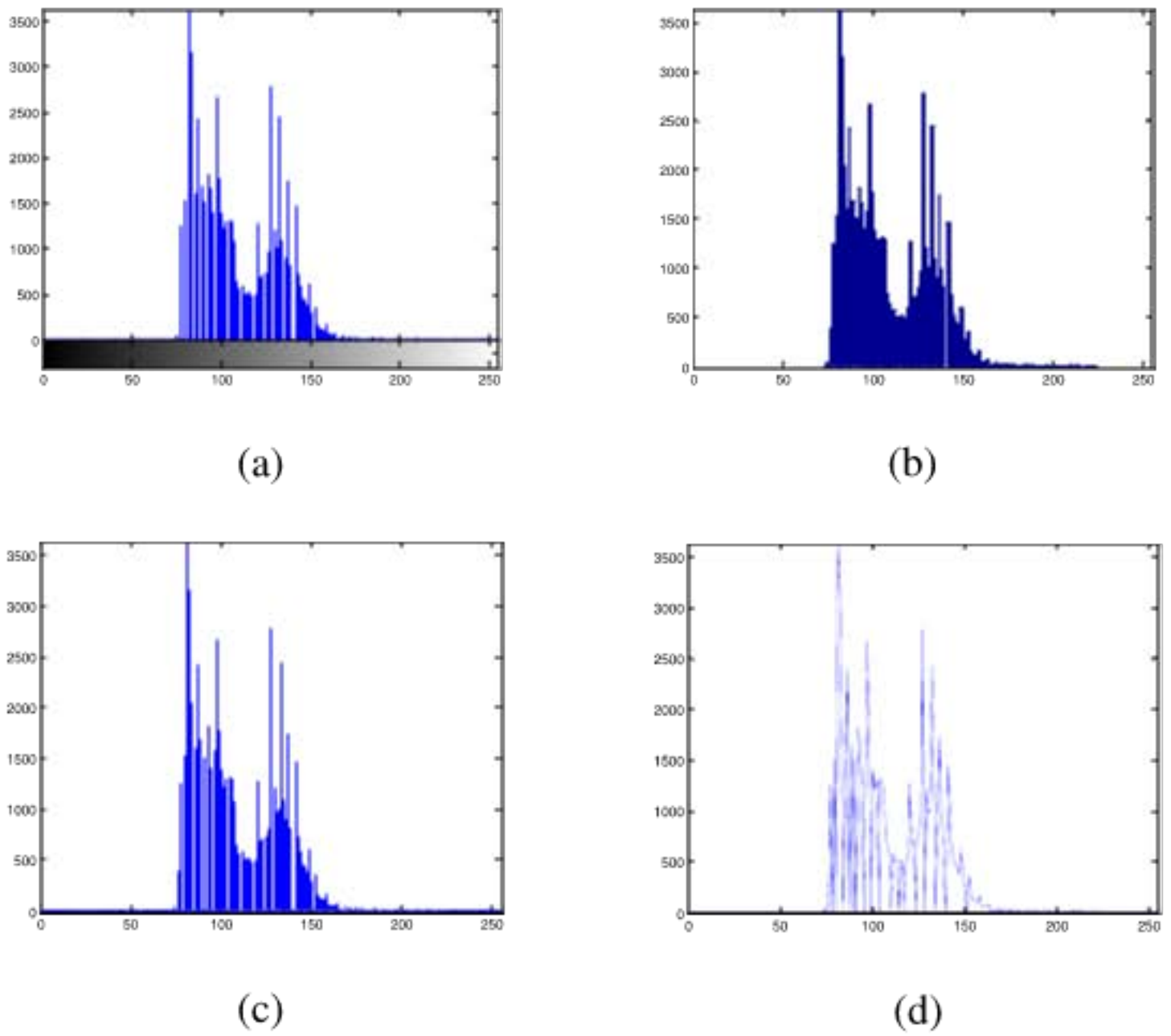
https://tieulun.hopto.org

```
p = imhist(f,b)/numel(f);
```

Với `numel` là tổng số pixels có trong ảnh `f`.

Ví dụ 5.3: Một số dạng đồ thị dùng để biểu diễn histogram

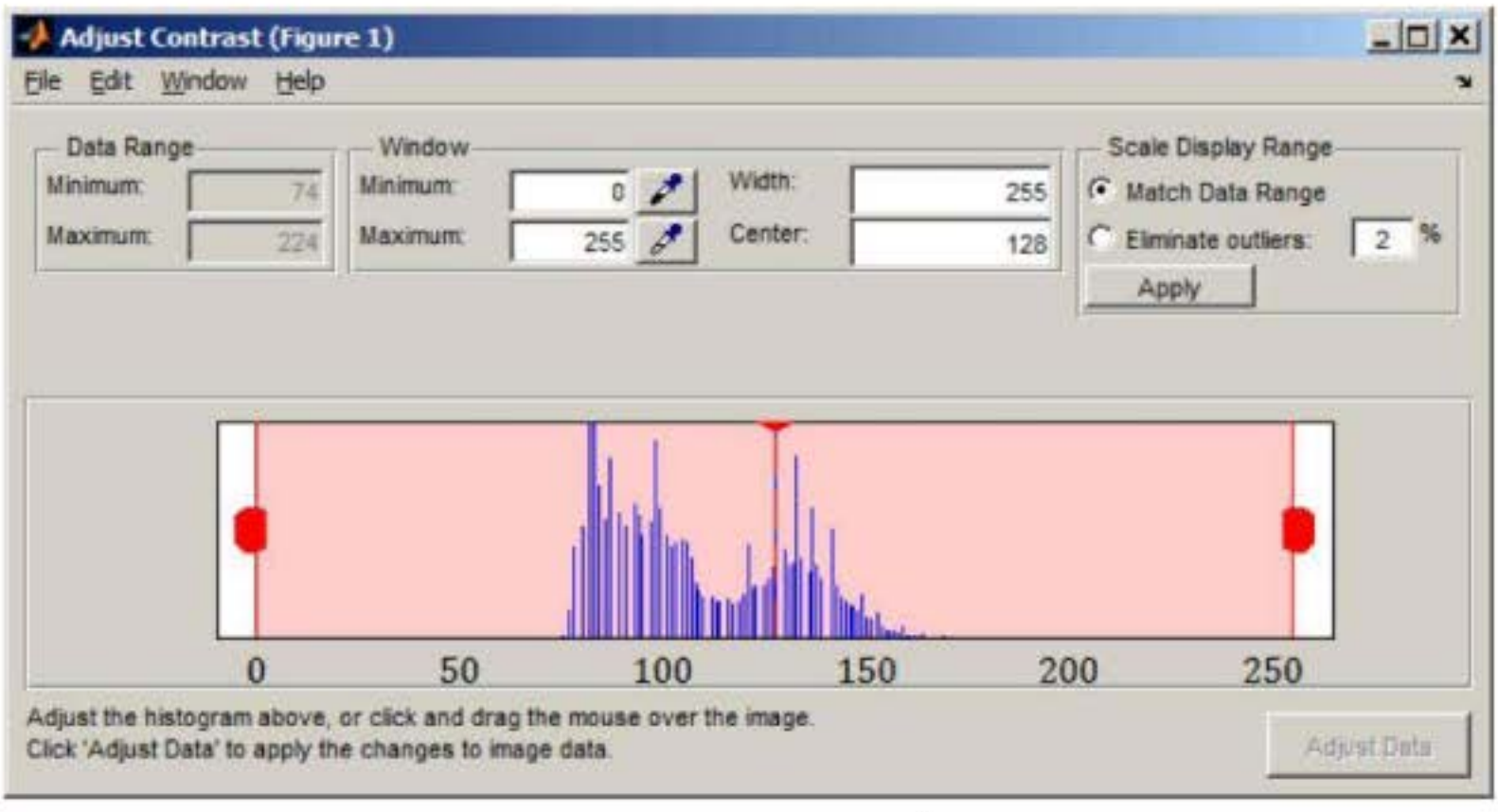
```
clear all;
f=imread('pout.tif');
h=imhist(f);
imhist(f);
axis([0 255 min(h) max(h)]);
figure;bar(0:255,h);
axis([0 255 min(h) max(h)]);
figure;stem(0:255,h,'marker','none');
axis([0 255 min(h) max(h)]);
figure;plot(0:255,h);
axis([0 255 min(h) max(h)]);
```



Hình 5.4. *Một số dạng đồ thị có thể được dùng để biểu diễn histogram(a) imhist (dạng mặc định); (b) bar; (c) stem; (d) plot*

https://tieulun.hopto.org

Một công cụ trực quan cho phép thay đổi độ tương phản thông qua việc thay đổi cận trên và cận dưới giá trị mức xám trong biểu đồ histogram được xây dựng sẵn là `imcontrast` với cú pháp: `imcontrast(f)`



Hình 5.5. *Mô tả giao diện cho phép thay đổi độ tương phản thông qua hàm* `imcontrast`

Lưu ý: Hàm `imcontrast` có thể dùng cho ảnh đang hiển thị trên hình thông qua cú pháp sau:

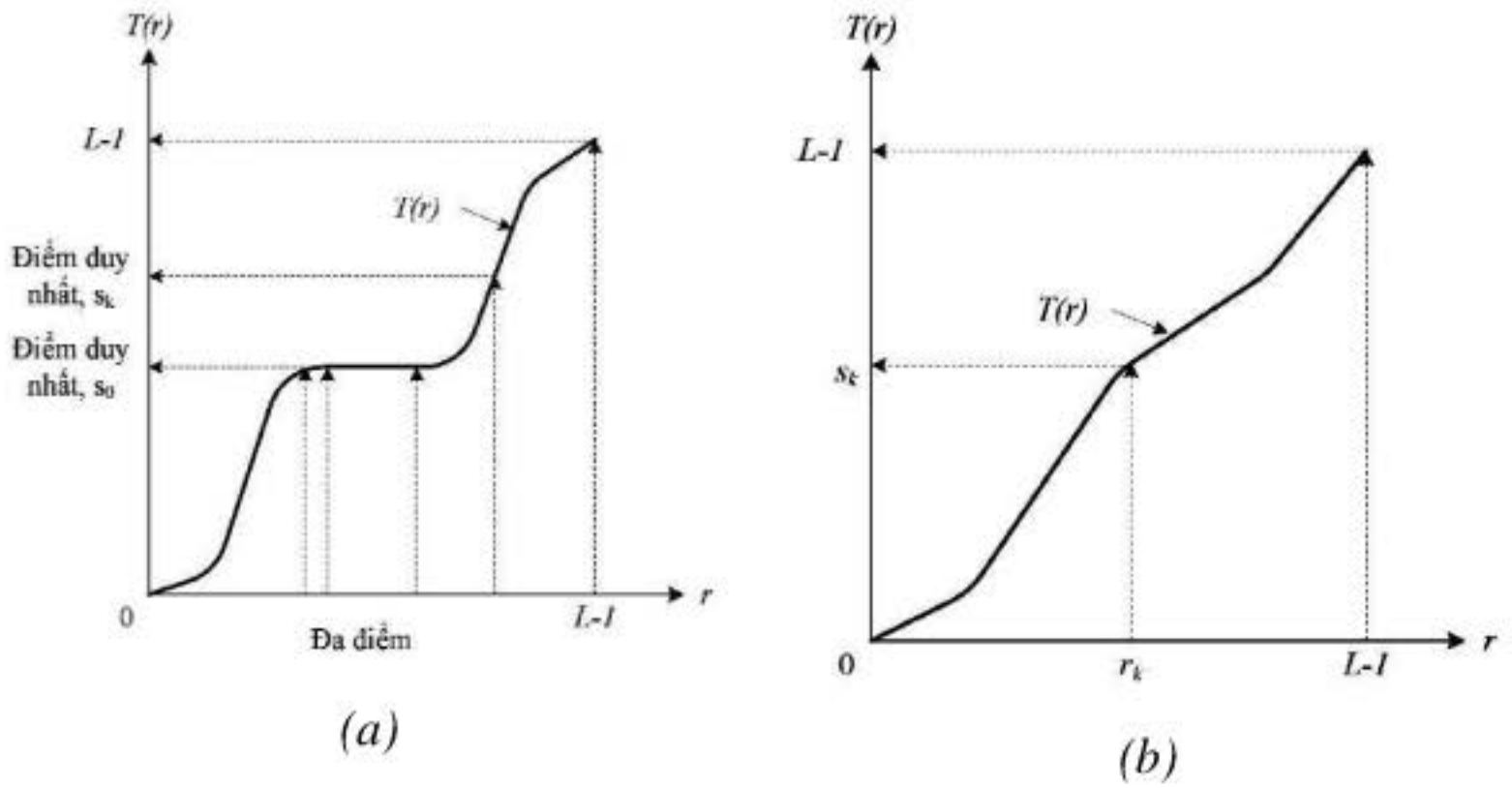
```
imshow(f)
imcontrast(gca)
```

5.1.2.1. Cân bằng Histogram

Như đã biết, hàm biến đổi mức xám có dạng tổng quát $s = T(r)$ với $0 \le r \le L-1$ sẽ luôn tạo ra giá trị mức xám ở ngõ ra là s ứng với mỗi giá trị mức xám ngõ vào là r. Chúng ta giả định rằng hàm $T(r)$ luôn tăng đơn điệu trong khoảng $0 \le r \le L-1$ và tất nhiên là $0 \le T(r) \le L-1$. Do đó, hoàn toàn có thể suy luận ngược lại rằng $r = T^{-1}(s)$ với $0 \le s \le L-1$.

Những điều kiện ràng buộc nêu trên với $T(r)$ luôn tăng để đảm bảo rằng giá trị mức xám ngõ ra không bao giờ nhỏ hơn ngõ vào tương ứng, điều này hạn chế các thành phần lạ xuất hiện trong ảnh ngõ ra, như trong hình 5.6(a). Ngoài ra, có thể nhận thấy rằng ảnh ngõ ra luôn có khoảng giá trị mức xám bằng với ngõ vào, và cũng như ánh xạ từ s đến r là dạng một – một ở hình 5.6(b).

https://tieulun.hopto.org



Hình 5.6. *(a) Hàm biến đổi tăng đơn điệu với hiện tượng nhiều giá trị mức xám có thể ánh xạ đến duy nhất một giá trị. (b) Hàm tăng đơn điệu hoàn toàn với ánh xạ một – một.*

Các giá trị mức xám trong ảnh có thể được biểu diễn như là các biến ngẫu nhiên trong khoảng [0, L-1] thông qua hàm mật độ xác suất PDF của chính nó. Với việc đánh dấu $p_r(r)$ và $p_s(s)$ là các hàm PDF của r và s tương ứng, nếu $p_r(r)$ và $T(r)$ được biết trước, và $T(r)$ là liên tục và khả vi trên khoảng đang xét thì PDF của biến s sau khi được ánh xạ có thể tìm được theo công thức sau:

$$p_s(s) = p_r(r)\left|\frac{ds}{dr}\right| \tag{5.3}$$

Hàm chuyển đổi dạng tổng quát cho trường hợp này được đưa ra có dạng:

$$s = T(r) = (L-1)\int_0^r p_r(w)dw \tag{5.4}$$

Với w là một biến hình thức của tích phân. Bên phải của biểu thức được xem như là hàm phân bố tích lũy (CDF) của các biến ngẫu nhiên r. Biểu thức (5.4) đã thỏa mãn tất cả các điều kiện nêu trên bao gồm hàm biến đổi là tăng đơn điệu và nằm trong khoảng cho phép [0, L-1].

Để tìm thành phần $p_s(s)$ tương ứng với hàm chuyển đổi đang được xem xét, áp dụng định luật Leibniz cho biểu thức (5.4)

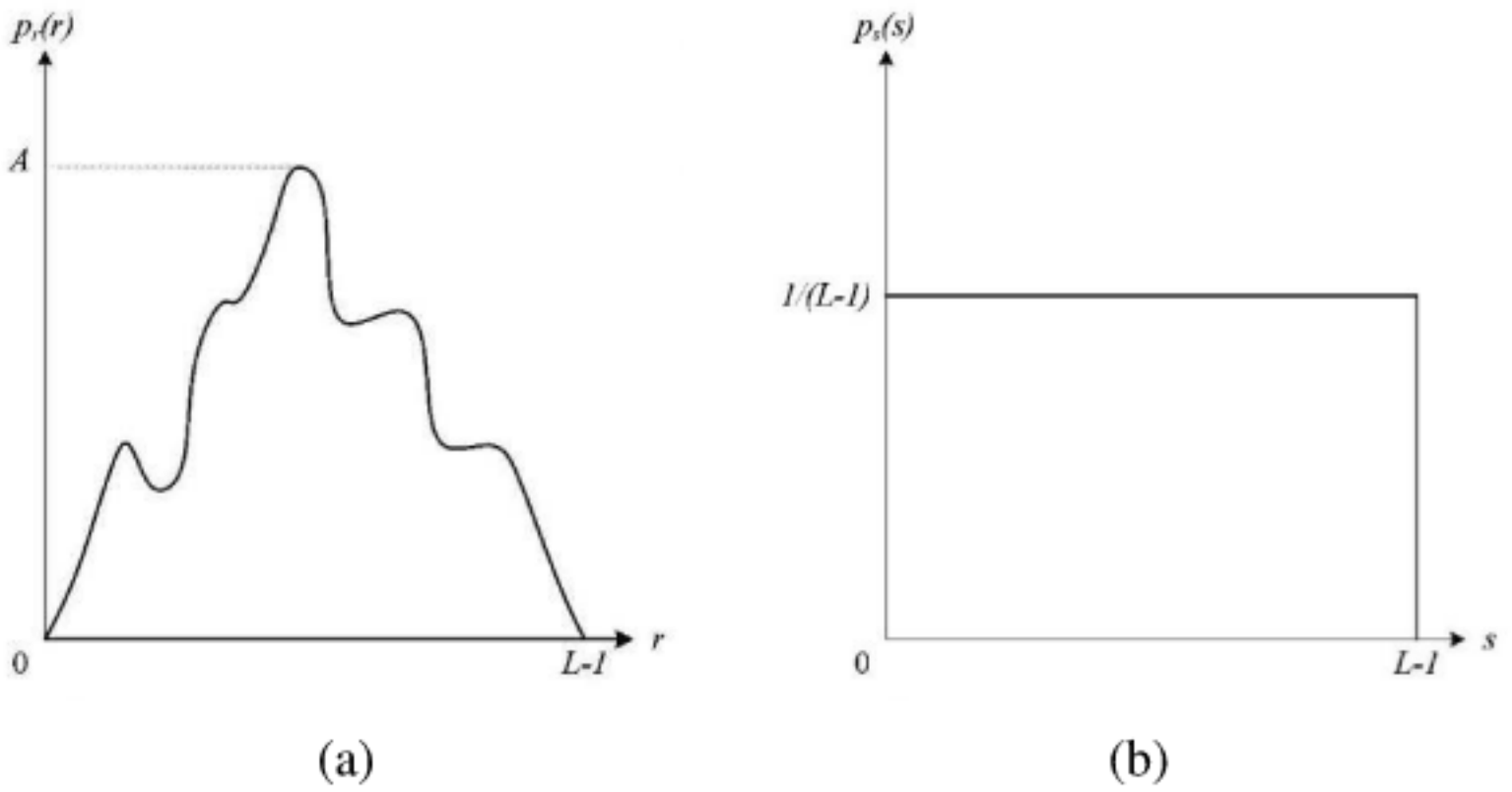
$$\frac{ds}{dr} = \frac{dT(r)}{dr} = (L-1)\frac{d}{dr}\left[\int_0^r p_r(w)dw\right] = (L-1)p_r(r) \tag{5.5}$$

https://tieulun.hopto.org

Thế kết quả này vào biểu thức (5.3):

$$p_s(s) = p_r(r)\left|\frac{ds}{dr}\right| = p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| = \frac{1}{L-1};0 \le s \le L-1 \qquad (5.6)$$

Với kết quả ở biểu thức (5.6), thành phần $p_s(s)$ có dạng hàm phân bố mật độ xác suất đều (uniform). Cần lưu ý là trong biểu thức (5.3), $T(r)$ phụ thuộc vào $p_r(r)$, nhưng trong biểu thức (5.6) thì $p_s(s)$ hoàn toàn độc lập với thành phần $p_r(r)$. Hình 5.7 làm rõ hơn về khái niệm này.



Hình 5.7. *(a) Hàm PDF được cho bất kỳ. (b) Kết quả của việc áp dụng của chuyển đổi trong biểu thức (5.6) cho tất cả các mức xám r. Kết quả ngõ ra $p_s(s)$ có dạng PDF đồng nhất trên tất cả các giá trị mức xám và hoàn toàn độc lập với PDF của r.*

Ví dụ 5.4: Cho hàm PDF:

$$p_r(r) = \frac{2r}{(L-1)^2};0 \le r \le L-1 \qquad (5.7)$$

Từ biểu thức (5.4), ta có

$$s = T(r) = (L-1)\int_0^r p_r(w)dw = \frac{2}{L-1}\int_0^r wdw = \frac{r^2}{L-1} \qquad (5.8)$$

Khi đó, giá trị mức xám ngõ ra bằng giá trị ngõ vào bình phương chia cho $L-1$. Hoàn toàn có thể kiểm chứng PDF của các giá trị mức xám trong ảnh mới là dạng đồng nhất bằng cách thế $p_r(r)$ vào biểu thức (5.6):

https://tieulun.hopto.org

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| = \frac{2r}{(L-1)^2}\left|\left[\frac{ds}{dr}\right]^{-1}\right| = \frac{2r}{(L-1)^2}\left|\left[\frac{d}{dr}\frac{r^2}{L-1}\right]^{-1}\right| \quad (5.9)$$

$$= \frac{2r}{(L-1)^2}\left|\frac{L-1}{2r}\right| = \frac{1}{L-1}$$

Với điều kiện là r không âm và $L > 1$.

Như đã được đề cập trước đó, xác suất xuất hiện của giá trị mức xám r_k trong ảnh được xấp xỉ theo:

$$p_r(r_k) = \frac{n_k}{MN}; k = 0,1,2,\ldots,L-1 \quad (5.10)$$

Với MN là tổng số điểm ảnh, n_k là số điểm ảnh có cùng giá trị r_k, và L là số lượng mức xám có thể biểu diễn (thường là 256, tương ứng với 8 bit). Hàm chuyển đổi dạng rời rạc trong biểu thức (5.4) có dạng như sau:

$$s_k = T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) = \frac{L-1}{MN}\sum_{j=0}^{k} n_j; k = 0,1,2,\ldots L-1 \quad (5.11)$$

Do đó, ảnh ngõ ra thu được bằng cách ánh xạ từng pixel trong ảnh ngõ vào với mức xám r_k thành điểm ảnh có mức xám s_k tương ứng ở ngõ ra theo như biểu thức (5.6), và ánh xạ $T(r_k)$ trong biểu thức này được gọi là cân bằng histogram.

***Ví dụ 5.5**:* Viết thuật toán cân bằng histogram

```
clear all;
f=imread('pout.tif');
h=histogram(f);
L=256;
for i=1:L
    s(i)=(L-1)*sum(h(1:i));
end
s=uint8(s);
for i=1:size(f,1)
    for j=1:size(f,2)
        g(i,j)=s(f(i,j)+1);
    end
```


https://tieulun.hopto.org

```
end
```

Hay có thể viết hàm `hist_equa` cho phép thực hiện nhanh thuật toán cân bằng histogram cho các ứng dụng về sau:

```
function g=hist_equa(f);
h=histogram(f);
L=256;
for i=1:L
    s(i)=(L-1)*sum(h(1:i));
end
s=uint8(s);
for i=1:size(f,1)
    for j=1:size(f,2)
        g(i,j)=s(f(i,j)+1);
    end
end
end
```

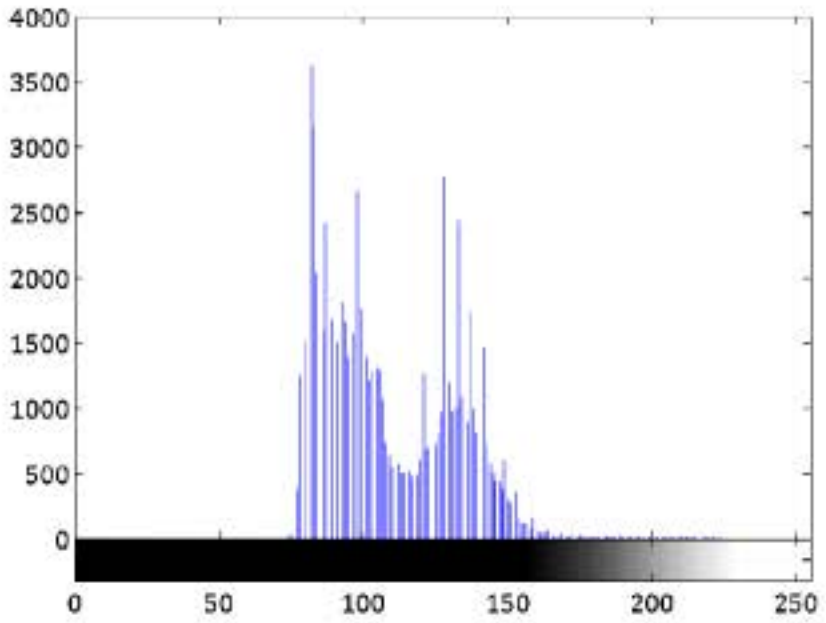
Trong MATLAB, hàm `histeq` có sẵn trong Toolbox cho phép thực hiện cân bằng histogram của ảnh với cú pháp sau:

```
J=histeq(I, hgram)
```

Lưu ý, hàm `histeq` vẫn có thể được dùng cho ảnh màu (xem thêm phần Help trong MATLAB).

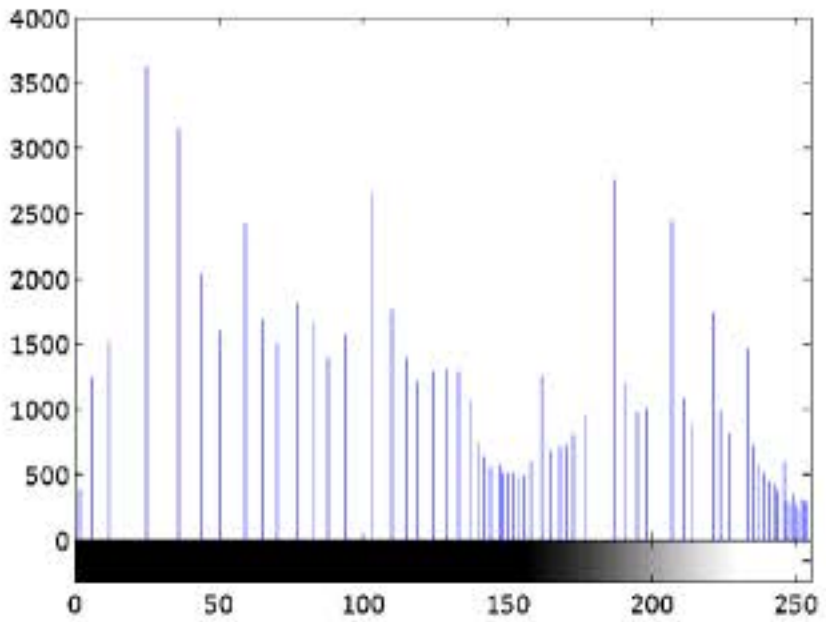
https://tieulun.hopto.org

(a)



(b)

(c)



(d)

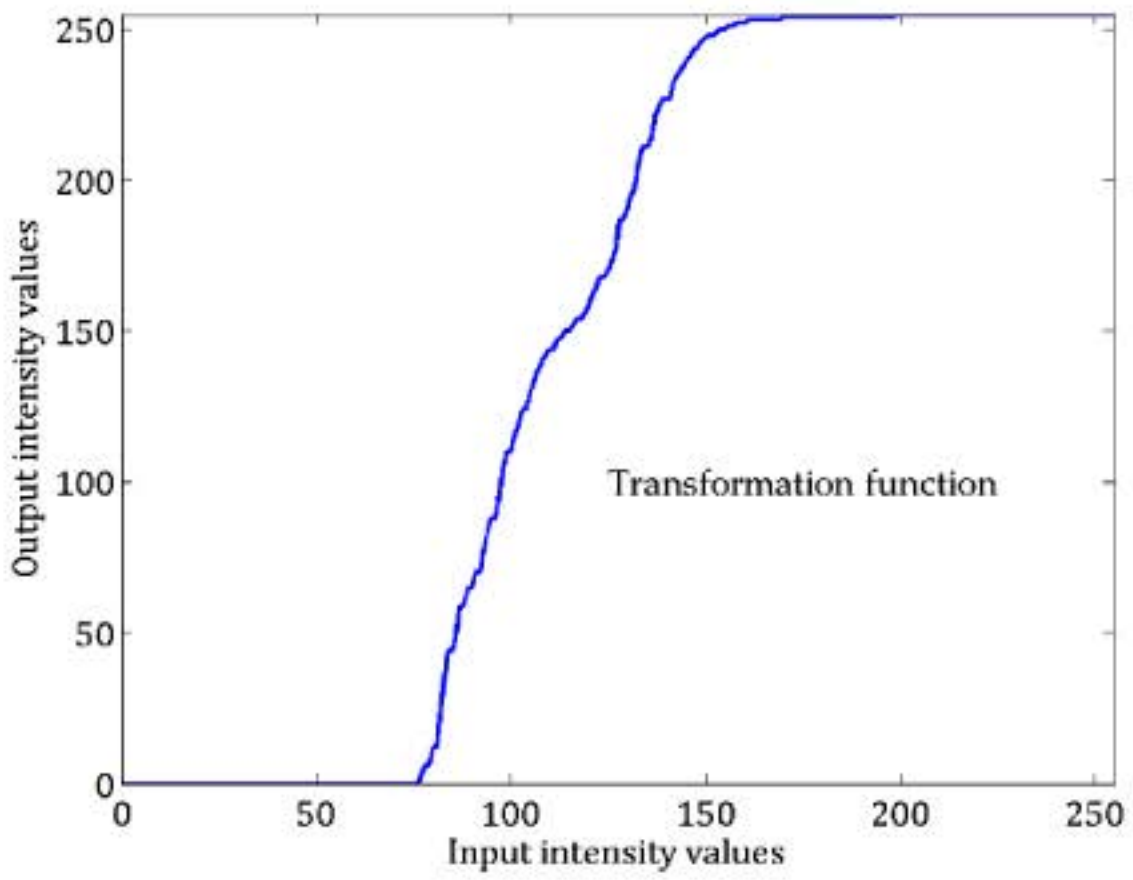
Hình 5.8. *Minh họa kết quả trong ví dụ 5.5 (a) – (b) Ảnh gốc và histogram (c) – (d) Ảnh sau khi cân bằng và histogram tương ứng.*

Ví dụ 5.7: Biểu diễn hàm chuyển đổi (ánh xạ) của thuật toán cân bằng histogram

```
clear all;
f=imread('pout.tif');
imhist(f);
h=histogram(f);
cdf=cumsum(h);
for i=1:256
    s(i)=255*cdf(i);
```


https://tieulun.hopto.org

```
end
s=uint8(s);
plot(0:255,s,'linewidth',2);
axis([0 255 0 255]);
xlabel('Input intensity values');
ylabel('Output intensity values');
text(125,100,'Transformation function');
```



Hình 5.9. *Hàm chuyển đổi được dùng để ánh xạ các giá trị mức xám từ ảnh ngõ vào trong hình 5.8(a) đến ảnh ngõ ra trong hình 5.8(c)*

Một điều cần lưu ý là những ảnh sau khi tăng cường bằng thuật toán cân bằng histogram có biểu đồ histogram như nhau. Đây cũng là một trong những hạn chế của thuật toán, đặc biệt là một ảnh được chụp trong các điều kiện sáng khác nhau sẽ luôn cho ra cùng một ảnh.

5.1.2.2. Kỹ thuật phối hợp lược đồ (Histogram matching)

Như đã biết, kỹ thuật cân bằng histogram sẽ tự động thực hiện chuyển đổi cường độ mức xám của ảnh để tạo ra ảnh mới có histogram đồng nhất. Khi việc tăng cường ảnh được yêu cầu, thì đây là kỹ thuật được áp dụng do có thể dự tính trước được kết quả cũng như thực hiện dễ dàng. Tuy nhiên, trong một vài ứng dụng thì kỹ thuật cân bằng dựa trên histogram đồng nhất không phải lúc nào cũng tốt nhất. Và kỹ thuật được dùng để tạo ra một ảnh với histogram đã được chỉ rõ gọi là histogram matching hay histogram specification.

https://tieulun.hopto.org

Nói một cách dễ hiểu, histogram matching là kỹ thuật điều chỉnh độ sáng, độ tương phản của hai ảnh bằng cách sử dụng histogram. Thông thường, nó được sử dụng như bộ cân bằng, được dùng để chuẩn hóa hai ảnh khi chúng được chụp trong cùng một môi trường ánh sáng cục bộ, cùng một địa điểm nhưng bởi các cảm biến, trong các điều kiện khí quyển hay ánh sáng toàn cục khác nhau.

Trong kỹ thuật này, hàm phân bố tích lũy CDF của ảnh cần chỉnh sửa sẽ được thay đổi sao cho càng giống với CDF của ảnh tham chiếu. Kỹ thuật histogram matching đầu tiên sẽ tính histogram của ảnh vào (ảnh sẽ được biến đổi) và của ảnh tham chiếu, sau đó CDF của từng ảnh được tính dựa trên hàm PDF của chúng. Ứng với mỗi giá trị mức xám ngõ vào sẽ được ánh xạ đến giá trị tương ứng tại hàm CDF của ảnh tham chiếu sao cho chúng có cùng giá trị mật độ tích lũy.

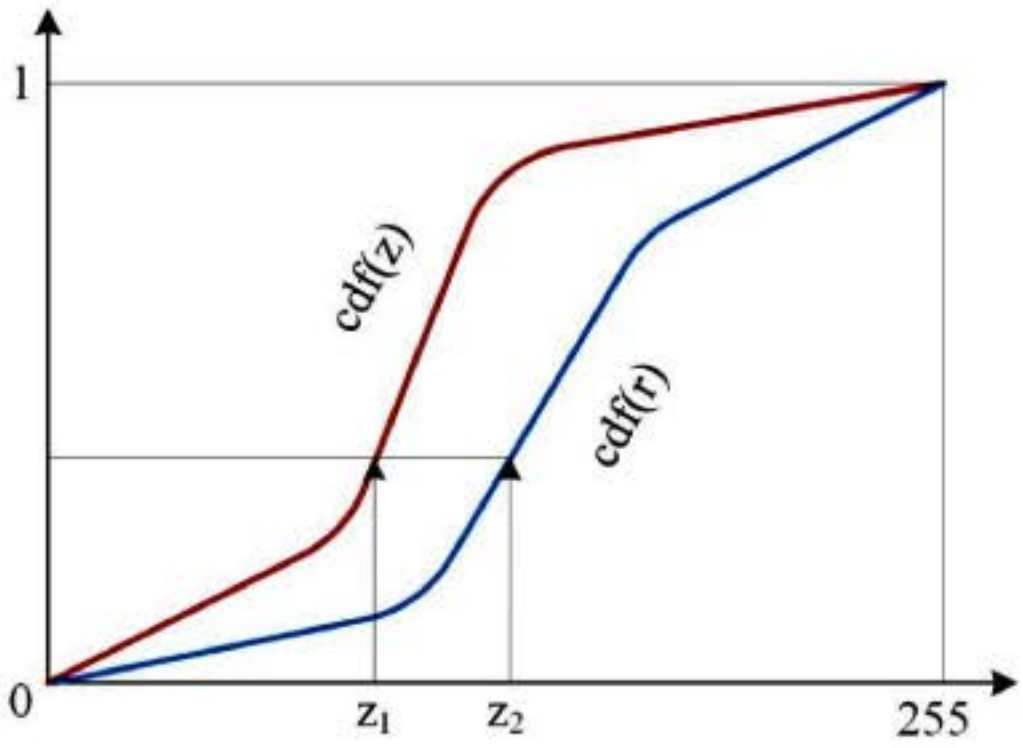
Gọi hàm phân bố xác suất của ảnh cần chỉnh sửa và ảnh tham chiếu lần lượt là $p_z(z)$ và $p_r(r)$. Hàm phân bố tích lũy được xác định.

$$cdf_z(z_k) = \sum_{i=0}^{k} p_z(z_k); \qquad cdf_r(r_k) = \sum_{i=0}^{k} p_r(r_k) \tag{5.12}$$

Ứng với mỗi giá trị mức xám $r_k \in [0,255]$ sẽ tìm thấy tương ứng một giá trị mức xám z_k sao cho

$$cdf_r(r_k) = cdf_z(z_k) \tag{5.13}$$

Từ đó, kết quả thu được là hàm ánh xạ giá trị mức xám từ ảnh tham chiếu sang. Hình 5.10 minh họa cho thuật toán histogram matching.



Hình 5.10. *Biểu đồ hàm cdf minh họa cho thuật toán histogram matching*

https://tieulun.hopto.org

Lưu ý:

- Kích thước của hai ảnh không cần phải giống nhau.
- Thuật toán có thể tạo ra những khoảng trống trong lược đồ histogram. Điều này có thể đoán được do histogram sẽ bị biến dạng. Điều này khá phổ biến với ảnh có mức giá trị nguyên.
- Kỹ thuật này không thích hợp với ảnh có chứa dữ liệu phân loại và ảnh màu RGB, trong trường hợp này, các kênh màu nên được phân tách riêng để xử lý.

Ví dụ 5.8: Thực hiện thuật toán histogram matching với hai ảnh: ảnh tham chiếu (ảnh gốc `pout`), ảnh cần được chỉnh sửa.

```
clear all;
f=imread('pout.tif');
g=imadjust(f,stretchlim(f));
z=imhistmatch(g,f,256);
```

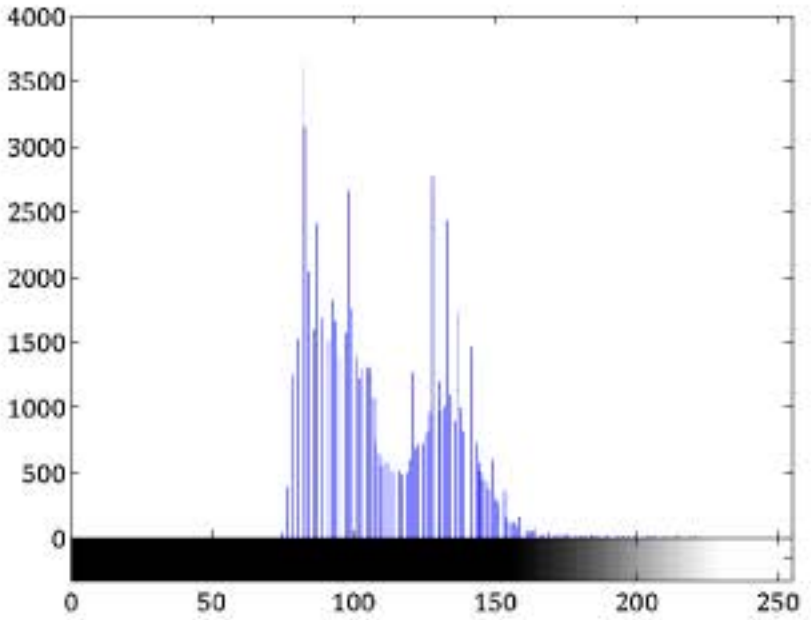
Kết quả ví dụ 5.8 được trình bày trong hình 5.11, ảnh cần được chỉnh sửa là ảnh đã bị biến đổi dùng hàm `imadjust` với sự hỗ trợ từ hàm `stretchlim` trong việc tìm cận trên và dưới của giá trị mức xám. Hàm thực hiện thuật toán histogram matching được xây dựng sẵn trong MATLAB là `imhistmatch` có cú pháp như sau:

```
B = imhistmatch(A,Ref,N)
```

Với `A` là ảnh sẽ được biến đổi, `Ref` là ảnh tham chiếu và `N` là số lượng giá trị mức xám được chuyển đổi. Mặc định của giá trị `N` là 64 nếu không được khai báo.

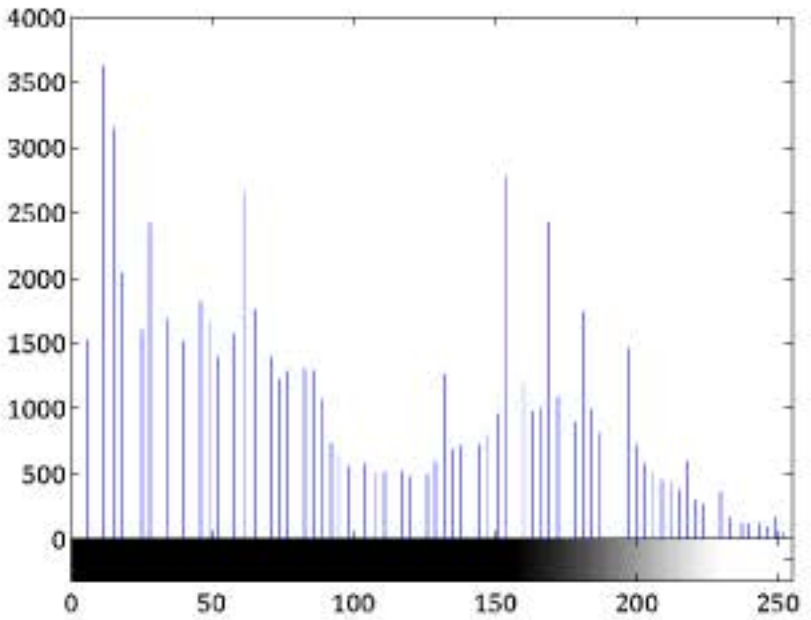
https://tieulun.hopto.org

(a)



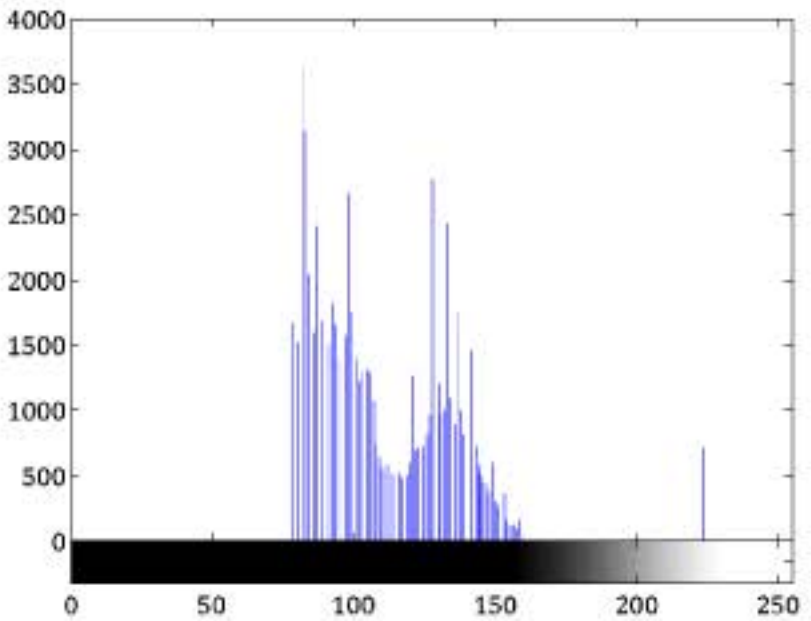
(b)

(c)



(d)

(e)



(f)

Hình 5.11. *Kết quả ví dụ 5.8: (a) Ảnh gốc; (c) Ảnh cần chỉnh sửa; (e) Ảnh sau khi thực hiện histogram matching và histogram tương ứng.*

https://tieulun.hopto.org

5.2. XỬ LÝ MẶT NẠ

Nhiễu do nhiều nguyên nhân khác nhau làm cho chất lượng ảnh bị giảm xuống. Để giảm nhiễu và nâng cao chất lượng ảnh ta sử dụng những phương pháp lọc với các mặt nạ khác nhau, phù hợp với mỗi loại nhiễu cụ thể. Trong phần này sẽ xét đến các bộ lọc ảnh trong không gian.

Lọc không gian (spatial filtering) cũng là một quá trình xử lý trên các điểm ảnh, dựa trên một phép toán với các điểm ảnh xung quanh. Phương pháp lọc ảnh bao gồm các bước:

- Xác định điểm ảnh trung tâm (x, y).
- Thực hiện các phép toán với các điểm xung quanh (x, y).
- Kết quả thu được đáp ứng của quá trình lọc tại (x, y).
- Lặp lại các bước trên với tất cả các điểm ảnh khác.

5.2.1. Các bộ lọc làm mịn ảnh

Các bộ lọc làm mịn ảnh được sử dụng để làm mờ và giảm nhiễu. Làm mờ ảnh được sử dụng trong quá trình tiền xử lý ảnh, nhằm mục đích loại bỏ các chi tiết nhỏ ra khỏi ảnh trước khi tiến hành tách các thành phần lớn hơn khỏi ảnh, làm mờ còn được sử dụng để làm liền lại những đứt quãng nhỏ của đường thẳng hoặc đường cong.

Như đã biết, lọc tuyến tính là một quá trình trong đó mỗi điểm ảnh có giá trị bằng trung bình của các điểm ảnh lân cận xác định bởi bộ lọc. Do đó, ảnh sau xử lý trở nên mượt hơn, giảm độ sắc nét so với ảnh gốc. Kết quả là các thành phần nhiễu ngẫu nhiên, thường có mức xám khác biệt với các vùng lân cận sẽ được loại bỏ. Tuy nhiên, một hạn chế dễ thấy khi tiến hành làm mượt ảnh là tại các vị trí biên (chi tiết được sử dụng nhiều trong xử lý ảnh), nơi có sự thay đổi nhanh chóng của các mức xám, lại bị làm mờ đi, ảnh hưởng đến các bước tiếp theo trong xử lý ảnh. Tuy nhiên, nếu sử dụng cửa sổ lọc thích hợp, có thể vừa giảm được nhiễu mà ít ảnh hưởng đến biên ảnh.

Một cửa sổ lọc chuẩn hóa thường thấy và đã được đề cập là cửa sổ có các hệ số giống nhau:

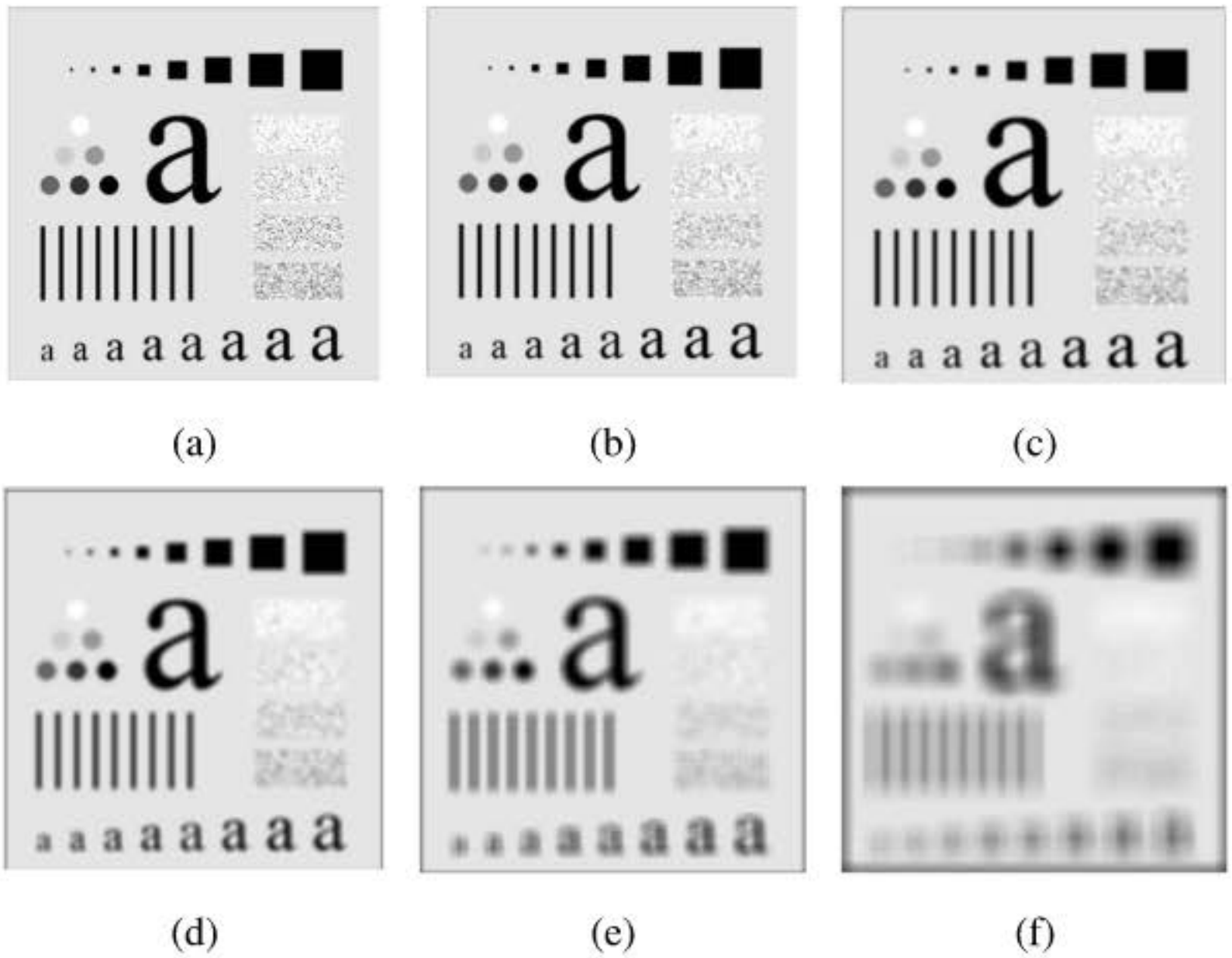
$$w = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{5.14}$$

Một loại cửa sổ lọc chuẩn hóa khác có các hệ số khác nhau:

https://tieulun.hopto.org

$$w = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{5.15}$$

Bộ lọc trên tiến hành lấy trung bình có trọng số đối với các điểm ảnh lân cận, tức là mỗi điểm ảnh được nhân với một hệ số khác nhau, hệ số càng lớn thì điểm ảnh đó có trọng số càng lớn. Ở bộ lọc trên, điểm ảnh trung tâm có vai trò quan trọng nhất trong phép toán tính trung bình; càng ra xa điểm ảnh trung tâm, trọng số của các điểm ảnh cũng giảm dần. Mục đích của việc làm này là hạn chế ảnh bị mờ khi tiến hành làm mượt. Tuy nhiên, ta rất khó thấy sự khác biệt giữa hai bộ lọc trên do cửa sổ lọc có kích thước nhỏ hơn nhiều so với kích thước ảnh.



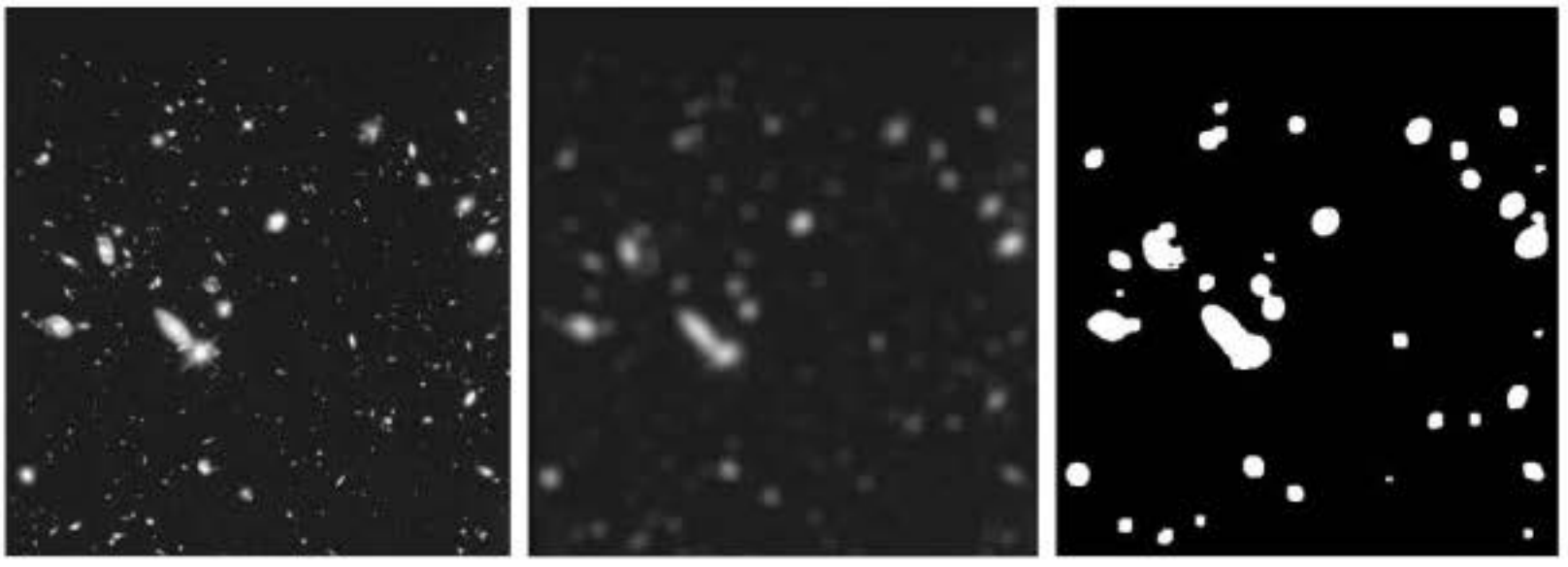
Hình 5.12. *Áp dụng cửa sổ lọc khác nhau vào ảnh cần làm mịn*

Xét ví dụ trên với việc sử dụng các bộ lọc trung bình có kích cỡ khác nhau, ở đây sử dụng padding là các giá trị 0. Hình 5.12(a) là ảnh gốc. Hình 5.12(b) sử dụng bộ lọc 3x3, hình bị mờ đi chút ít, với các chi tiết nhỏ như chữ a nhỏ và thành phần nhiễu bị mờ nhiều hơn so với các chi tiết khác.

https://tieulun.hopto.org

Kết quả tương tự cũng xảy ra với hình 5.12(c) sử dụng bộ lọc kích thước 5x5. Các chi tiết nhỏ như nhiễu đã giảm dần, các mép hình răng cưa cũng đã được làm mượt hơn, nhưng vẫn đảm bảo các thành phần kích thước lớn không bị ảnh hưởng nhiều.

Hình 5.12(d) dùng cửa sổ 9x9, ảnh mờ hơn, các chi tiết nhiễu đã được giảm khá nhiều. Hình 5.12(e) và hình 5.12(f) dùng các bộ lọc tương ứng 16x16 và 35x35, các chi tiết nhỏ gần như đã bị loại khỏi ảnh, do đó có thể dễ dàng lấy được các thành phần có kích thước lớn.



(a) (b) (c)

Hình 5.13. *Làm mịn ảnh sử dụng bộ lọc có kích thước 15x15*

Hình 5.13(a) là ảnh gốc. Hình 5.13(b) là ảnh sau lọc với bộ lọc kích thước 15x15, cho thấy các chi tiết nhỏ gần như bị loại bỏ. Để thu các thành phần kích thước lớn của ảnh, có thể biến đổi ảnh trong hình 5.13(b) thành ảnh nhị phân. Ở đây cho mức ngưỡng là 25% giá trị mức xám lớn nhất của ảnh 5.13(b). Kết quả được ảnh 5.13(c), tất cả các chi tiết nhỏ bị loại bỏ, chỉ còn các thành phần kích thước lớn.

Việc thực hiện làm mịn ảnh có thể được thực hiện nhờ các bộ lọc đã trình bày trong chương 4. Sau đây, ta sẽ xem xét việc ứng dụng các bộ lọc này trong tăng cường ảnh.

Ví dụ 5.9: Khảo sát các tùy chọn dùng lọc trung bình với cửa sổ lọc 15x15

```
clear all;
f=imread('check.tif');
h=ones(15)/(15^2);
g1=imfilter(f,h);
g2=imfilter(f,h,'replicate');
g3=imfilter(f,h,'symmetric');
g4=imfilter(f,h,'circular');
```


https://tieulun.hopto.org

Hình 5.14. *Kết quả dùng cửa sổ lọc 15x15 với các tùy chọn*

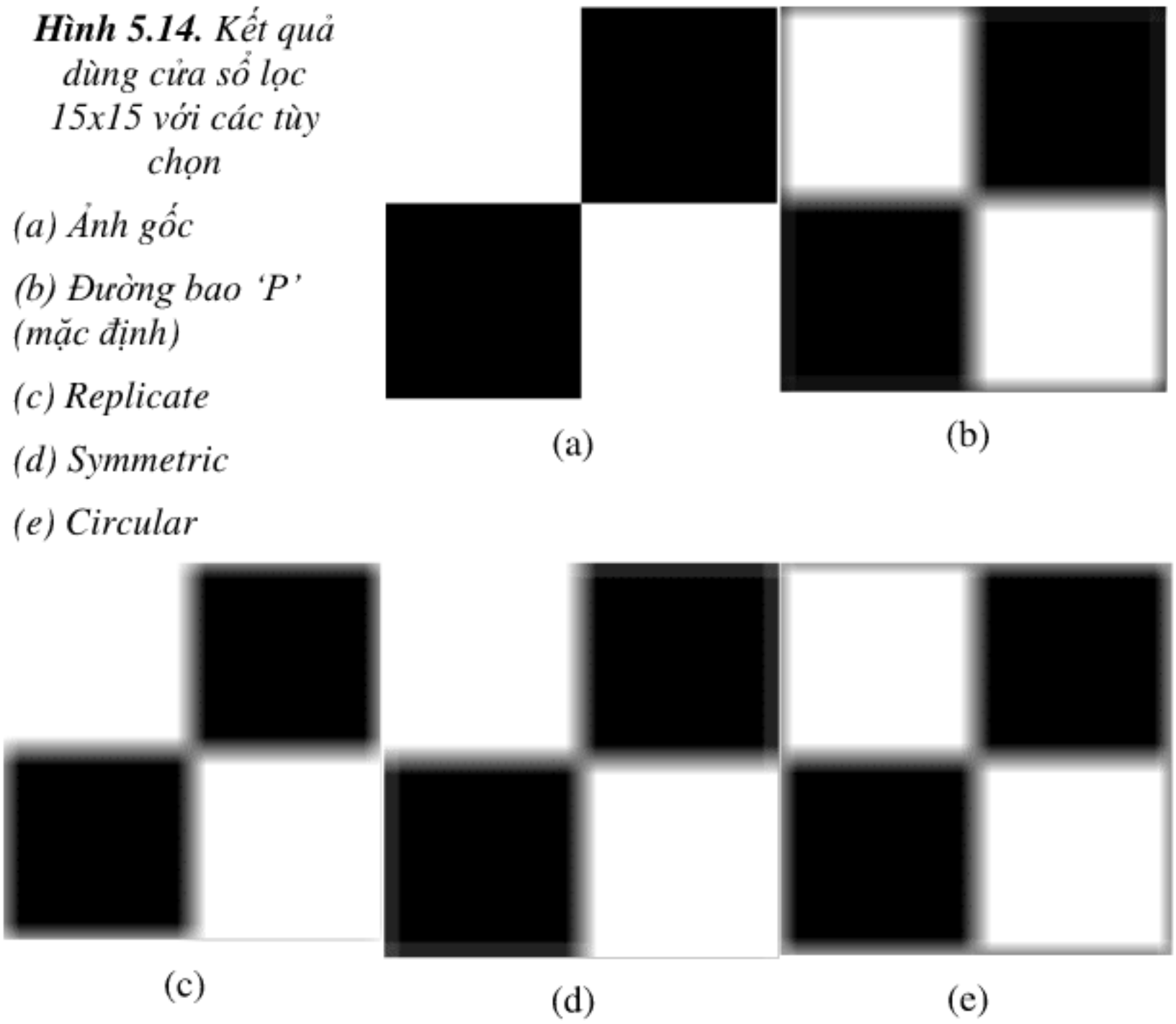
(a) Ảnh gốc

(b) Đường bao 'P' (mặc định)

(c) Replicate

(d) Symmetric

(e) Circular



Trong hình 5.14(b), do sử dụng tùy chọn mặc định với đường bao được mở rộng bằng cách thêm vào một điểm ảnh có giá trị là 0 nên ảnh ngõ ra sẽ có một đường viền màu đen. Để giải quyết việc xuất hiện đường bao không mong muốn này, tùy chọn `'replicate'` được sử dụng trong hình 5.14(c). Kết quả tương tự cũng đạt được với tùy chọn *`'symmetric'`*. Tuy nhiên ở tùy chọn cuối cùng là *`'circular'`*, kết quả ảnh ngõ ra gặp phải vấn đề tương tự như trong hình 5.14(b).

Các mặt nạ lọc đã trình bày trong chương 4 có thể được áp dụng vào việc lọc ảnh. Tùy theo đặc điểm ảnh khác nhau mà loại mặt nạ cụ thể được sử dụng.

Ví dụ 5.10: Thực hiện lọc ảnh dùng mặt nạ Sobel

```
I=imread('peppers.png');
h=fspecial('average',[5 5]);
G=imfilter(I,h);
figure;
```


https://tieulun.hopto.org

```
subplot(1,2,1)
imshow(I)
xlabel('(a)')
subplot(1,2,2)
imshow(G)
xlabel('(b)')
```

(a) (b)

Hình 5.15. *Thực hiện mặt nạ lọc trung bình: (a) Ảnh gốc; (b) Ảnh đã lọc*

Một trong những bộ lọc cũng được sử dụng để loại nhiễu muối tiêu hay làm trơn ảnh là lọc trung vị. Bộ lọc này hiệu quả hơn các bộ chập khác nhờ việc đồng thời giảm nhiễu và bảo tồn biên. Trong MATLAB, bộ lọc này được khai báo:

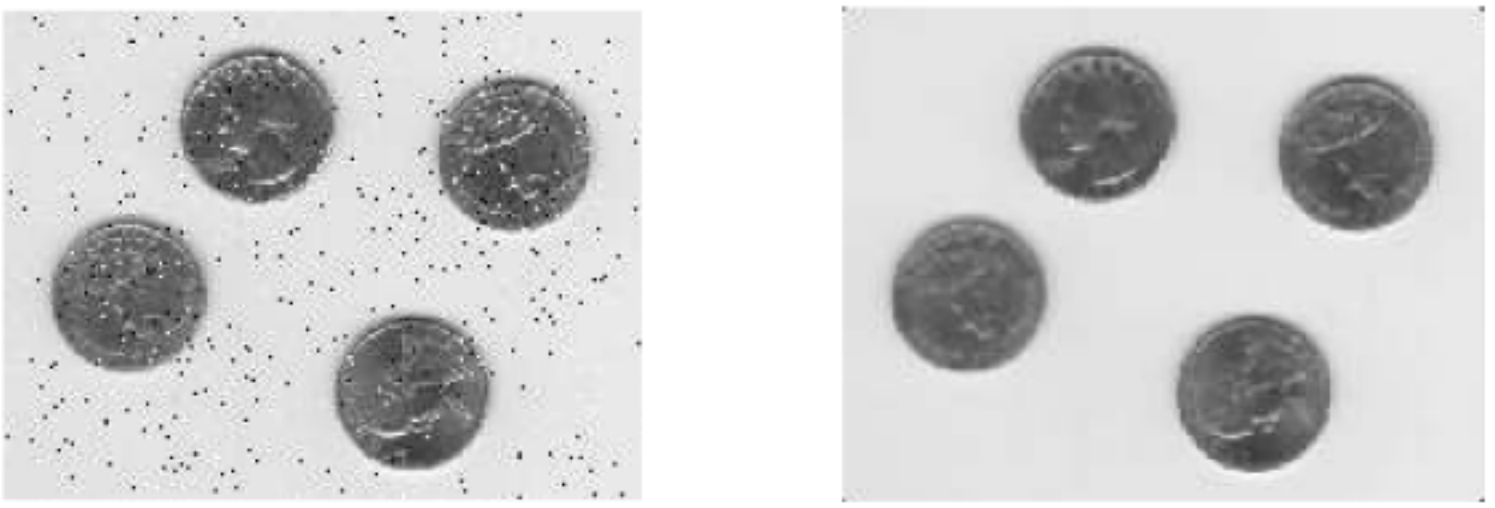
```
B=medfilt2(A, [m n])
```

Trong đó `A` là ảnh ngõ vào, mỗi điểm ảnh trong ảnh ngõ ra `B` là trung bình của `m×n` điểm ảnh lân cận.

Ví dụ 5.11: Thực hiện gây nhiễu muối tiêu lên ảnh và dùng `medfilt2` để lọc nhiễu này

```
I = imread('eight.tif');
J = imnoise(I,'salt & pepper',0.02);
K = medfilt2(J);
figure,
subplot(1,2,1)
imshow(J), xlabel('(a)')
subplot(1,2,2)
imshow(K), xlabel('(b)')
```


https://tieulun.hopto.org



(a) (b)

Hình 5.16. *Thực hiện bộ lọc trung vị: (a) Ảnh bị nhiễu muối tiêu; (b) Ảnh sau khi lọc trung vị*

5.2.2. Các bộ lọc làm sắc nét ảnh

Mục đích của việc làm sắc nét ảnh là làm nổi bật các chi tiết trong ảnh hoặc làm sắc nét các chi tiết bị mờ bởi quá trình làm mịn ảnh. Quá trình làm mịn ảnh là thực hiện phép lấy trung bình các giá trị lân cận điểm ảnh cần xử lý, tương tự như phép toán tích phân, trong khi đó quá trình làm sắc nét ảnh tập trung vào sự sai lệch giữa các chi tiết trong ảnh, giống như phép toán vi phân. Kết quả là biên ảnh và các chi tiết nhiễu, những nơi có sự khác biệt về mức xám với các điểm ảnh xung quanh, được làm nổi bật lên.

Trong phần này sẽ thực hiện làm sắc nét ảnh dựa trên đạo hàm bậc một và bậc hai của hàm rời rạc.

- Đạo hàm bậc một của hàm rời rạc một chiều:

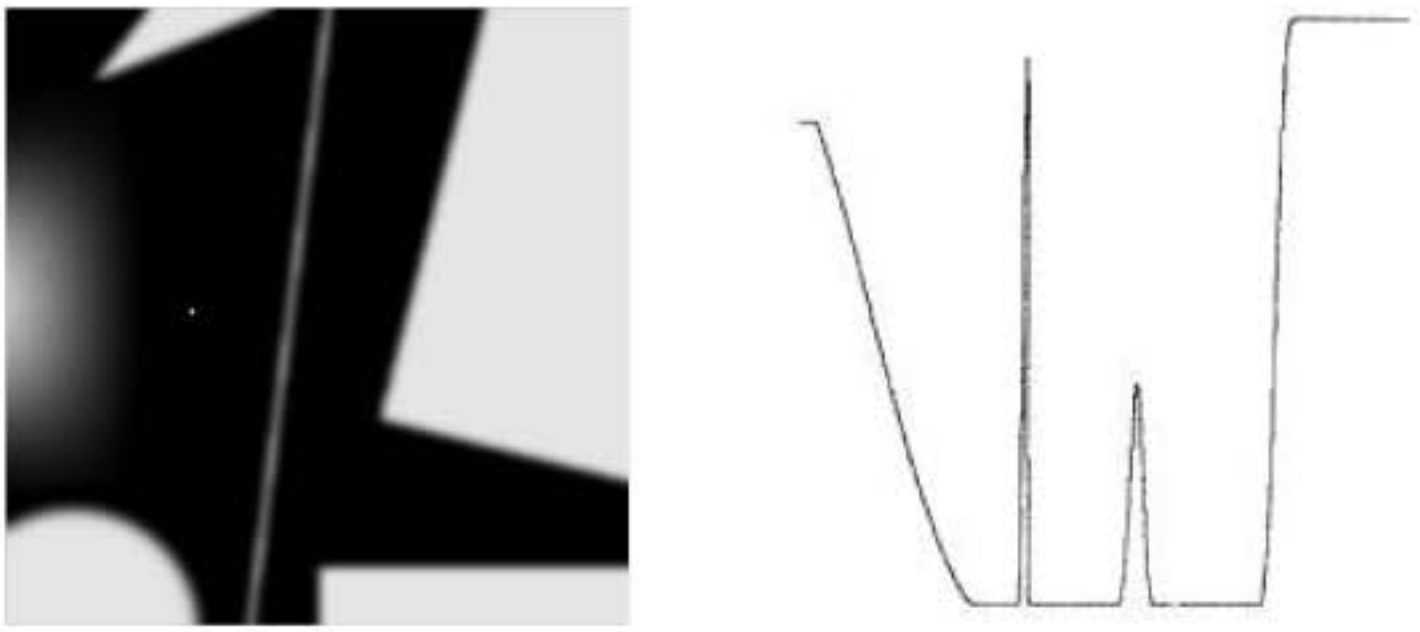
$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \tag{5.16}$$

- Đạo hàm bậc hai của hàm rời rạc một chiều:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) - f(x-1) - 2f(x) \tag{5.17}$$

Để thấy được sự giống cũng như khác nhau về cơ bản giữa hai phương pháp sử dụng đạo hàm bậc một và bậc hai, ta xét một ví dụ cụ thể sau:

https://tieulun.hopto.org



(a) (b)

Hình 5.17. *Biểu diễn mức xám của các pixel nằm trên đường ngang qua điểm giữa của ảnh*

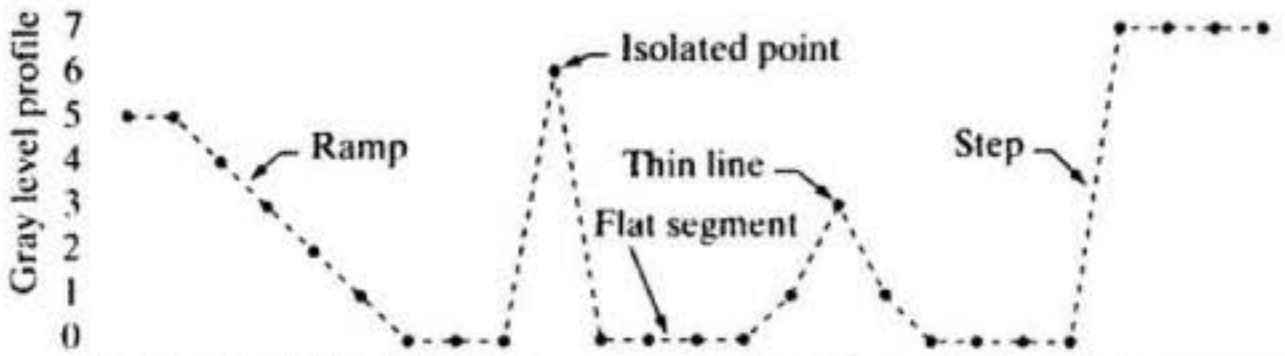


Image strip	5	5	4	3	2	1	0	0	0	6	0	0	0	0	1	3	1	0	0	0	0	7	7	7	7	•	•
First Derivative		-1	-1	-1	-1	-1	0	0	6	-6	0	0	0	1	2	-2	-1	0	0	0	7	0	0	0			
Second Derivative		-1	0	0	0	0	1	0	6	-12	6	0	0	1	1	-4	1	1	0	0	7	-7	0	0			

Hình 5.18. *Đơn giản hóa các giá trị mức xám của hình 5.16(b)*

Hình 5.17(a) có một đường chéo và một điểm nhiễu. Hình 5.17(b) biểu diễn mức xám của các pixel nằm trên đường ngang qua điểm giữa của ảnh, bao gồm cả điểm nhiễu. Hình 5.18 đơn giản hóa các giá trị mức xám của hình 5.17(b), chỉ gồm 8 mức xám khác nhau. Từ hình 5.18, có thể phân tích ảnh hưởng của phương pháp đạo hàm bậc một và bậc hai đối với điểm nhiễu, với đường chéo và cạnh biên giữa đối tượng và nền.

Các đoạn có mức xám không đổi thì đạo hàm bậc một và hai đều cho đáp ứng là 0. Với đoạn dốc thoải, đạo hàm bậc một cho các mức khác 0 trên toàn đoạn, đạo hàm bậc hai chỉ cho các giá trị khác 0 ở đầu và cuối đoạn, điều đó chứng tỏ với các đoạn chuyển tiếp thì đạo hàm bậc một tạo ra cạnh dày hơn và đạo hàm bậc hai tạo ra cạnh sắc hơn. Với điểm nhiễu, đáp ứng xung quanh và tại điểm nhiễu đạo hàm bậc hai lớn hơn so với đạo hàm bậc một, do đó đạo hàm bậc hai tạo ra chi tiết sắc hơn đối với nhiễu

https://tieulun.hopto.org

và các giá trị mức xám thay đổi nhanh. Đường chéo cũng tương tự với điểm nhiễu. Còn lại với bước nhảy thì đáp ứng của đạo hàm bậc một và hai là tương tự nhau. Do đó, có thể kết luận:

- Đạo hàm bậc một tạo ra các cạnh dày hơn so với đạo hàm bậc hai.
- Đạo hàm bậc hai ảnh hưởng nhiều hơn đến các chi tiết tinh.

Đạo hàm bậc hai được sử dụng nhiều hơn trong xử lý ảnh do nó làm nổi bật các chi tiết sắc, do đó chúng ta sẽ tập trung vào phương pháp này để làm sắc nét ảnh.

Do ảnh là một hàm rời rạc hai chiều nên ta cần có đạo hàm bậc hai của hàm hai chiều. Toán tử Laplace của hàm hai biến là:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{5.18}$$

Công thức tính đạo hàm bậc hai cho hàm rời rạc hai chiều thường sử dụng là:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \tag{5.19}$$

Và:

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y) \tag{5.20}$$

Toán tử Laplace trong không gian rời rạc hai chiều là:

$$\partial^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y) \tag{5.21}$$

Biểu thức trên có thể thực hiện đối với các phần tử của ảnh bằng cách nhân chập ảnh với ma trận:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \text{hoặc} \quad \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{5.22}$$

Một định nghĩa khác về đạo hàm bậc hai có sử dụng thêm các phần tử đường chéo:

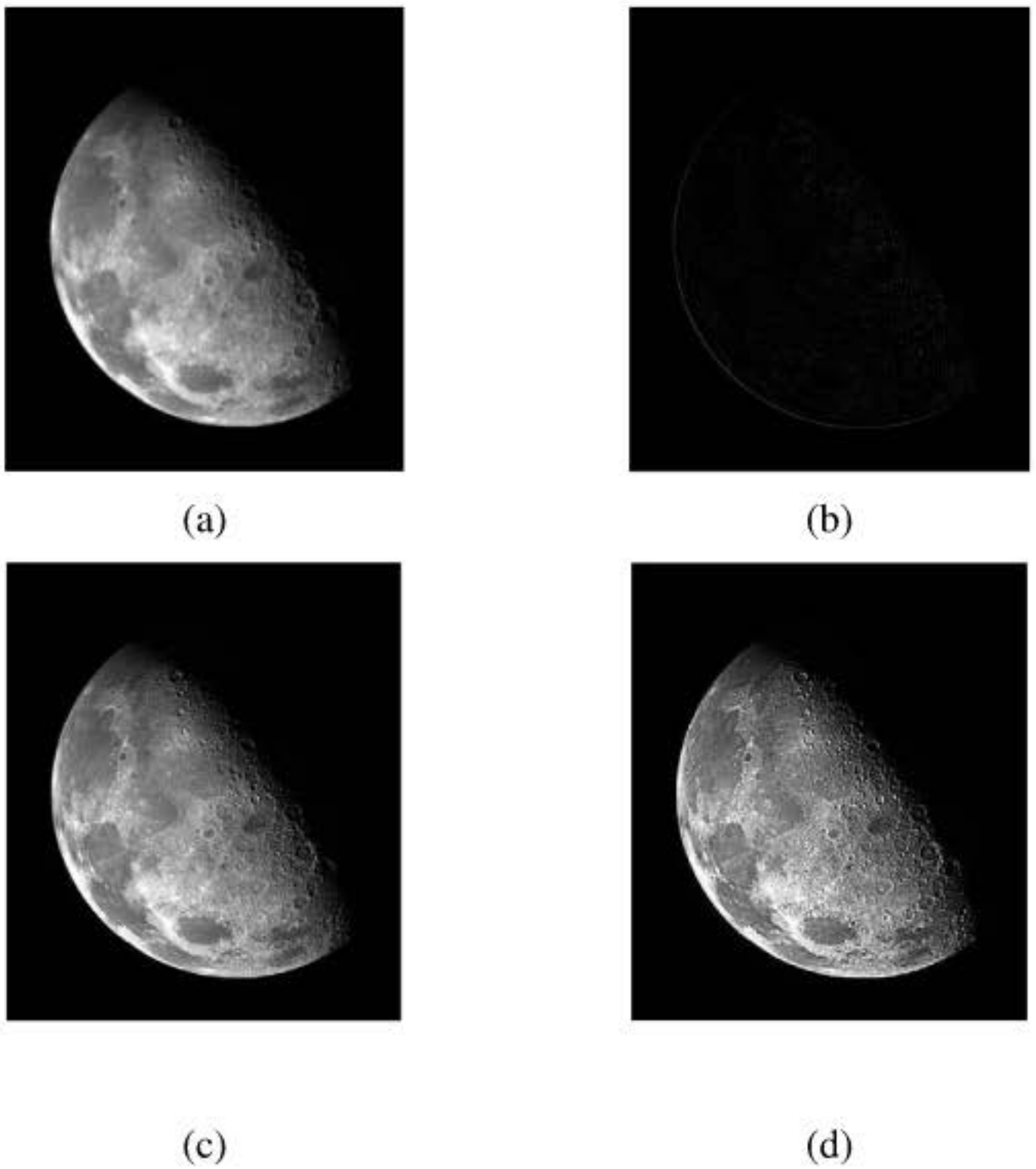
https://tieulun.hopto.org

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \text{hoặc} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{5.23}$$

Do Laplace là toán tử đạo hàm, nó làm sắc nét các chi tiết tinh hay các thành phần mức xám biến đổi nhanh, nhưng lại dẫn đến giá trị 0 cho những vùng có cùng một mức xám cũng như giảm giá trị của các thành phần mức xám ít biến đổi. Một cách đơn giản phục hồi lại các vùng này nhưng vẫn giữ cho các chi tiết sắc nét là cộng ảnh gốc với ảnh thực hiện phép lọc bằng toán tử Laplace.

$$g(x, y) = f(x, y) + c\left[\nabla^2 f(x, y)\right] \tag{5.24}$$

Trong đó $c = 1$ nếu hệ số trung tâm của mặt nạ lọc là dương, $c = -1$ nếu ngược lại.



Hình 5.19. *Xử lý ảnh dùng mặt nạ Laplace*

https://tieulun.hopto.org

Hình 5.19(a) là ảnh gốc. Hình 5.19(b) là ảnh sau xử lý dùng mặt nạ Laplace. Ở đây, vùng có thành phần mức xám biến đổi chậm sau khi lọc sẽ bị giảm giá trị. Trong khi đó phần cạnh biên và các chi tiết có mức xám biến đổi nhanh được thể hiện rõ nét hơn. Hình 5.18(c) là kết quả của việc cộng ảnh gốc với ảnh Laplace, phục hồi lại phần bị giảm mức xám nhưng vẫn giữ được sự sắc nét của các chi tiết. Hình 5.19(d) sử dụng mặt nạ có thêm các giá trị đường chéo, cho hình ảnh sắc nét hơn hình 5.19(c).

Mặt nạ Laplace có thể tạo ra trong Matlab nhờ hàm `fspecial`.

```
>> fspecial('laplacian', alpha)
```

Trong đó hệ số `alpha` cho phép điều chỉnh mức độ sắc nét của hình ảnh. Có thể dùng mặt nạ lọc trực tiếp như sau:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \text{hoặc} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} \qquad (5.25)$$

- ***Bộ lọc tăng cường (high-boost filter)***

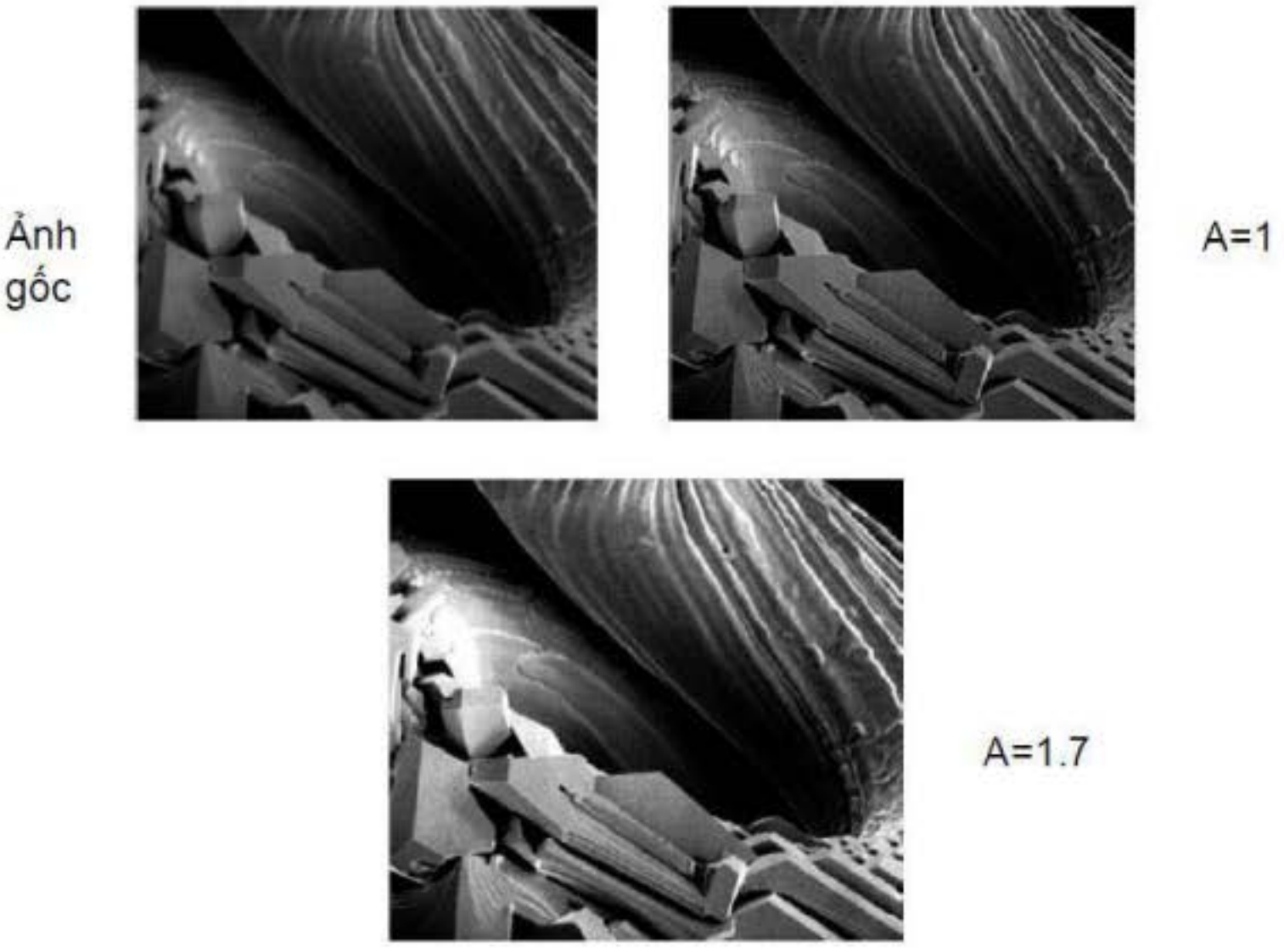
Bộ lọc tăng cường cũng là một ứng dụng của toán tử Laplace:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & A+4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \text{hoặc} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & A+8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad \text{với } A \geq 1 \qquad (5.26)$$

Với $A = 1$, bộ lọc tăng cường giống như mặt nạ lọc trực tiếp ở trên. $A > 1$ thì khả năng làm sắc nét ảnh giảm dần, nếu A đủ lớn thì ảnh sau xử lý gần như giống ảnh ban đầu nhân với một hằng số.

Một ứng dụng của bộ lọc tăng cường là làm sáng ảnh vì nó nâng mức xám trung bình của ảnh ban đầu mà vẫn giữ được tính chất làm nét ảnh.

https://tieulun.hopto.org



Hình 5.20. *Ứng dụng của bộ lọc tăng cường với hệ số A khác nhau*

BÀI TẬP CHƯƠNG 5

5.1. Trình bày cách tính histogram và các dạng đồ thị biểu diễn histogram.

5.2. Mở ảnh ‘trees.tif ’ và thực hiện cân bằng histogram cho ảnh này.

5.3. Mở ảnh ‘coins.png’. Tạo nhiễu muối tiêu lên ảnh này với các mức độ khác nhau. Dùng lọc trung vị để loại nhiễu với các mức độ khác nhau này.

5.4. Dùng mặt nạ Laplace để tăng độ nét cho ảnh 'ngc6543a.jpg'.

https://tieulun.hopto.org

Chương 6
PHÂN ĐOẠN VÀ TÁCH BIÊN

Sự thay đổi hay tính không liên tục trong biên độ của ảnh thường cho biết những đặc trưng quan trọng của một ảnh vì nó cung cấp thông tin về vật thể có trong ảnh. Tính không liên tục cục bộ trong một ảnh từ độ chói này sang độ chói khác được gọi là biên. Tính không liên tục toàn cục được gọi là đoạn đường bao. Chương này sẽ trình bày các kỹ thuật dùng để phát hiện biên và phân đoạn.

6.1. PHÂN ĐOẠN

Phân đoạn ảnh là công việc phân chia hay tách ảnh thành các vùng có thuộc tính giống nhau. Thuộc tính cơ bản nhất để phân đoạn là độ chói đối với ảnh đơn sắc và các thành phần màu đối với ảnh màu. Biên ảnh và kết cấu ảnh cũng là các thuộc tính quan trọng để phân đoạn.

6.1.1. Phương pháp phân đoạn biên độ chói

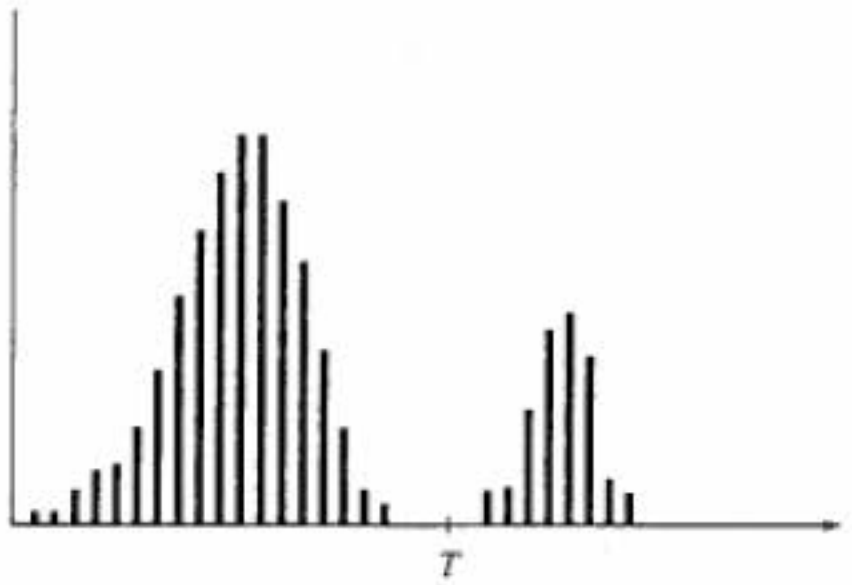
Phần này trình bày các phương pháp phân đoạn ảnh dựa trên ngưỡng của độ chói hay thành phần màu trong ảnh.

Giả sử một ảnh $f(x, y)$ có lược đồ xám như trong hình 6.1. Ảnh này gồm vật thể sáng trên nền tối vì thế vật thể và nền có độ chói của các điểm ảnh được chia thành hai nhóm. Một trong những cách để trích vật ra khỏi nền là chọn một ngưỡng T để chia hai nhóm này ra. Như vậy bất kỳ điểm (x, y) sao cho $f(x, y) \geq T$ được gọi là điểm vật thể ngược lại điểm ảnh được gọi là điểm nền. Nói cách khác, một ảnh đã phân ngưỡng $g(x, y)$ được định nghĩa

$$g(x, y) = \begin{cases} 1; & f(x, y) \geq T \\ 0; & f(x, y) < T \end{cases} \qquad (6.1)$$

Các điểm ảnh được đánh nhãn 1 tương ứng với vật thể trong khi đó nhãn 0 tương ứng với nền. Khi T là hằng số, phương pháp này gọi là đặt ngưỡng toàn cục.

https://tieulun.hopto.org



Hình 6.1. *Lược đồ xám của một ảnh có sự tách biệt về mức xám ở ngưỡng T*

Một trong những cách để chọn ngưỡng là dựa trên việc quan sát lược đồ xám. Lược đồ xám trong hình 6.1 có hai vùng rõ ràng vì thế T có thể được chọn một cách dễ dàng để tách chúng. Một phương pháp khác là chọn thử T và dùng phương pháp thử-sai, bằng cách chọn các giá trị ngưỡng khác nhau cho đến khi đạt được kết quả tốt nhất từ người quan sát. Để chọn ngưỡng một cách tự động, Gonzalez và Woods đã giới thiệu giải thuật sau

Bảng 6.1. *Các bước thực hiện chọn ngưỡng tự động*

1. Chọn một giá trị khởi tạo cho T (giá trị khởi tạo thường được chọn là trung bình của độ chói lớn nhất và nhỏ nhất trong ảnh).
2. Phân đoạn ảnh dùng ngưỡng T này. Điều này sẽ tạo ra hai nhóm điểm ảnh: G_1 chứa tất cả điểm ảnh có độ chói $\geq T$ và G_2 chứa các điểm ảnh có độ chói $<T$.
3. Tính độ chói trung bình μ_1 và μ_2 cho các điểm ảnh trong vùng G_1 và G_2.
4. Tính ngưỡng mới

$$T = \frac{1}{2}(\mu_1 + \mu_2)$$

5. Lặp lại từ bước 2 đến bước 4 cho đến khi sự sai biệt T nhỏ hơn một giá trị đã định trước T_0.

Hộp công cụ trong MATLAB cung cấp hàm tính ngưỡng dùng phương pháp Otsu. Hàm mật độ xác suất của lược đồ xám chuẩn hóa được cho bởi

https://tieulun.hopto.org

$$p_r(r_q) = \frac{n_q}{n} \qquad q = 0,1,2,\ldots L-1 \tag{6.2}$$

Trong đó n là tổng số lượng điểm ảnh trong ảnh, n_q là số lượng điểm ảnh có độ chói r_q và L là tổng số mức độ chói có thể có trong ảnh. Giả sử một ngưỡng k được chọn sao cho C_0 là tập các điểm ảnh với mức [0, 1, …, k-1] và C_1 là tập các điểm ảnh có độ chói [k, k+1, …, L-1]. Phương pháp Otsu tìm ngưỡng k sao cho cực đại hóa phương sai giữa hai lớp và được định nghĩa

$$\sigma_B^2 = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2 \tag{6.3}$$

Trong đó

$$\omega_0 = \sum_{q=0}^{k-1} p_q(r_q) \tag{6.3a}$$

$$\omega_1 = \sum_{q=k}^{L-1} p_q(r_q) \tag{6.3b}$$

$$\mu_0 = \sum_{q=0}^{k-1} q p_q(r_q) / \omega_0 \tag{6.3c}$$

$$\mu_1 = \sum_{q=k}^{L-1} q p_q(r_q) / \omega_1 \tag{6.3d}$$

$$\mu_T = \sum_{q=0}^{L-1} q p_q(r_q) \tag{6.3e}$$

Hàm `graythresh` thực hiện tìm ngưỡng này.

```
T=graythresh(f)
```

Ví dụ 6.1: Thực hiện phân đoạn với việc chọn ngưỡng từ phương pháp Otsu, ta được kết quả như trong hình 6.2.

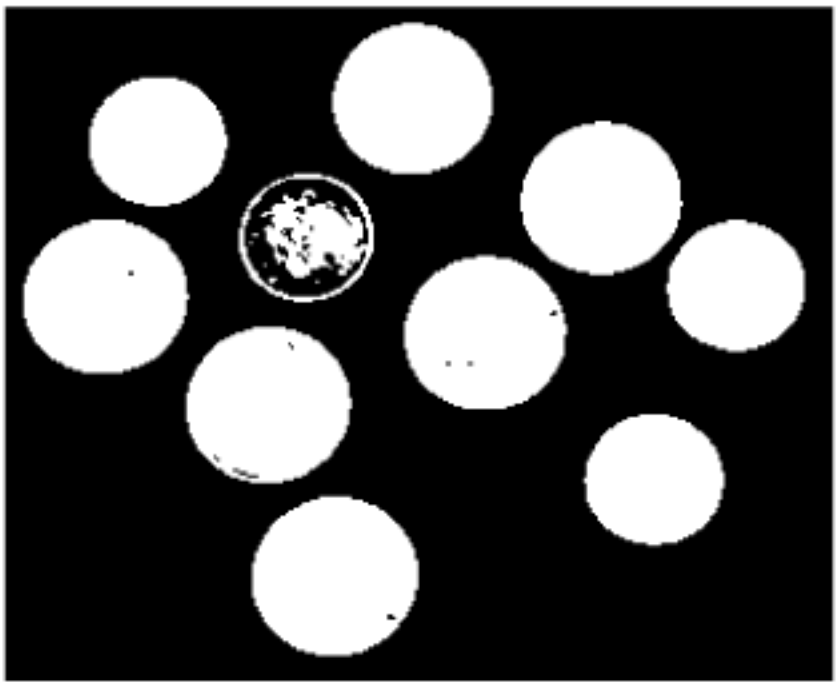
```
I=imread('cameraman.tif')
figure;
imshow(I)
t=255*graythresh(I);
f=I>=t;
figure;
imshow(f)
```


https://tieulun.hopto.org

Hình 6.2. *Kết quả phân đoạn dựa trên ngưỡng toàn cục*

Ví dụ 6.2: Thực hiện phân đoạn cho ảnh 'coins.png' và chuyển sang ảnh nhị phân.

```
I = imread('coins.png');
level = graythresh(I);
BW = im2bw(I,level);
figure, imshow(BW)
```



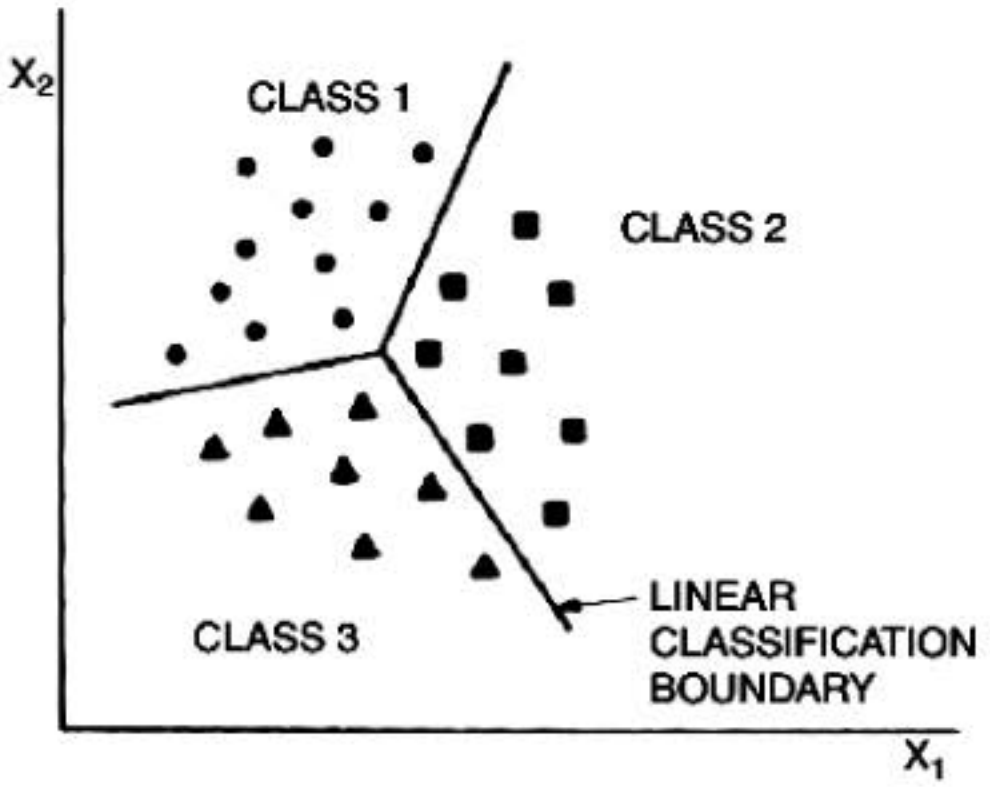
Hình 6.3. *Phân đoạn dùng hàm* `graythresh` *kết hợp nhị phân ảnh.*

6.1.2. Phương pháp phân đoạn nhóm

Giả sử một vector $\mathbf{x} = [x_1, x_2, \ldots, x_N]^T$ là giá trị đo được tại điểm ảnh tọa độ (j, k). Giá trị đo được có thể là giá trị phổ, thành phần màu hay các đặc trưng của điểm ảnh lân cận. Nếu các giá trị đo được từ các điểm ảnh khác nhau có điểm tương đồng thì ta có thể thực hiện phân đoạn. Điều này tức là việc phân đoạn trở thành việc chia không gian N chiều thành

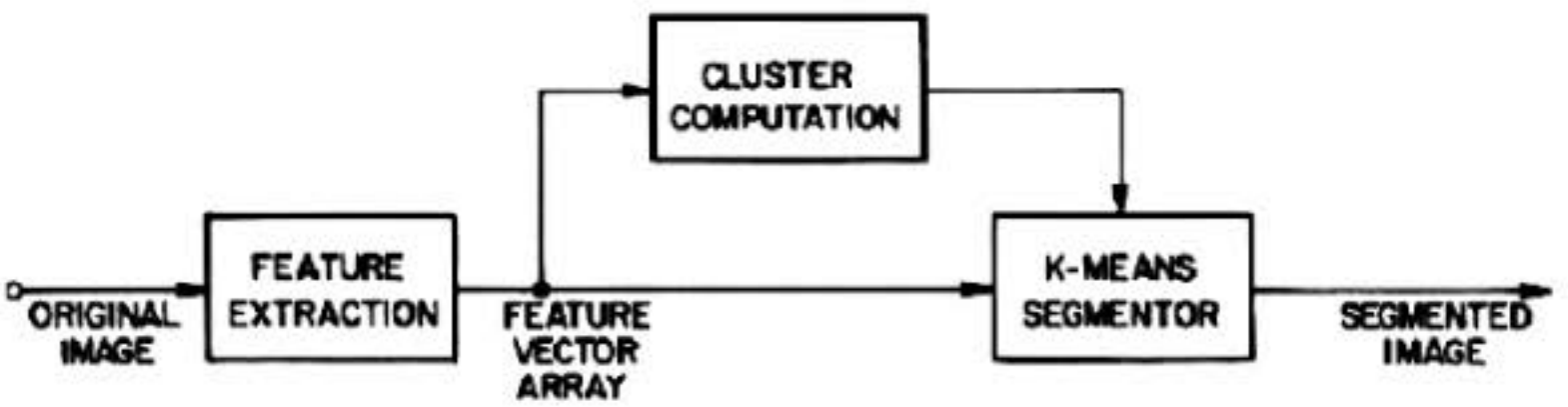
https://tieulun.hopto.org

các không gian rời rạc, mỗi không gian này bao lấy một nhóm cần phân đoạn. Hình 6.4 miêu tả khái niệm cho 3 đặc trưng.



Hình 6.4. *Phân chia đặc trưng*

Coleman và Andrews đã phát triển một giải thuật phân đoạn nhóm tin cậy. Hình 6.5 trình bày sơ đồ miêu tả thuật toán này theo dạng đơn giản để phân đoạn ảnh đơn. Bước đầu tiên của giải thuật là tính toán đặc trưng. Bước tiếp theo là phân nhóm.



Hình 6.5. *Sơ đồ phân đoạn theo nhóm*

Giải thuật phân nhóm bắt đầu bằng việc thiết lập hai trung tâm nhóm tạm thời. Tất cả các vector đặc trưng của một ảnh được gán vào trung tâm nhóm gần nhất. Tiếp theo, số lượng các trung tâm nhóm tăng lên (mỗi lần tăng 1 trung tâm nhóm), và một hệ số phẩm chất β được tính tại mỗi vòng lặp cho đến khi β đạt giá trị lớn nhất. Điều này thiết lập nên số nhóm tối ưu. Khi số nhóm tăng lên 1, trung tâm nhóm mới trở thành vector đặc trưng xa nhất tính từ trung tâm nhóm lân cận của nó. Hệ số β được định nghĩa

$$\beta = \mathrm{tr}\{\mathbf{S}_W\}\mathrm{tr}\{\mathbf{S}_B\} \tag{6.4}$$

https://tieulun.hopto.org

Trong đó S_W và S_B là ma trận tán xạ trong và giữa nhóm. Và tr{} là vết của ma trận. Ma trận tán xạ trong được tính

$$\mathbf{S}_W = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{M_k}\sum_{x_i \in S_k}(\mathbf{x}_i - \mathbf{u}_k)(\mathbf{x}_i - \mathbf{u}_k)^T \tag{6.5}$$

Trong đó K là số nhóm, M_k là số lượng phần tử vector trong nhóm thứ k, x_i là phần tử vector trong nhóm thứ k, u_k là trung bình của nhóm thứ k, và S_k là tập các phần tử trong nhóm thứ k. ma trận tán xạ giữa được định nghĩa

$$\mathbf{S}_B = \frac{1}{K}\sum_{k=1}^{K}(\mathbf{u}_k - \mathbf{u}_0)(\mathbf{u}_k - \mathbf{u}_0)^T \tag{6.6}$$

Trong đó $\mathbf{u}_0$ là trung bình của tất cả vector đặc trưng

$$\mathbf{u}_0 = \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i \tag{6.7}$$

Trong đó M là số lượng điểm ảnh cần được nhóm.

6.1.3 Phương pháp phân đoạn vùng

Các phần trước đã trình bày việc phân đoạn dựa vào ngưỡng dựa trên sự phân bố về tính chất của điểm ảnh chẳng hạn như độ chói. Trong phần này, kỹ thuật phân đoạn dựa trên việc tìm vùng một cách trực tiếp.

6.1.3.1. Công thức cơ bản

Đặt R đại diện cho vùng ảnh hiện tại. Ta có thể thấy việc phân đoạn ảnh là một quá trình chia R thành n vùng nhỏ, $R_1, R_2, \ldots, R_n$

Bảng 6.2. *Các tiền đề trong phân vùng*

a) $R = \bigcup_{i=1}^{n} R_i$
b) R_i là một vùng kết nối; i=1, 2, …, n.
c) $R_i \cap R_j =$ với mọi i và j sao cho $i \neq j$
d) $P(R_i)$= *TRUE* với i=1, 2, …, n.
e) $P(R_i \cup R_j) = FALSE$ cho bất kỳ vùng liền kề R_i và R_j.

Ở đây, $P(R_i)$ là tiền đề logic định nghĩa trên các điểm trong tập R_i. Điều kiện a) cho biết việc phân đoạn phải được hoàn tất, đó là các điểm ảnh phải ở trong một vùng. Điều kiện thứ 2 yêu cầu các điểm trong một

https://tieulun.hopto.org

vùng có tính kết nối dựa trên một số tiền đề. Điều kiện c) cho biết tính không kết nối giữa các vùng. Điều kiện d) liên quan đến tính chất phải được thỏa mãn trong vùng phân đoạn, tức là $P(R_i)= TRUE$ nếu tất cả điểm ảnh trong R_i có cùng mức xám. Cuối cùng, điều kiện e) cho biết các vùng liền kề là khác nhau theo tiền đề P.

6.1.3.2. Phân đoạn phát triển vùng

Phân đoạn dựa trên việc phát triển vùng là quá trình nhóm các điểm ảnh hay các nhóm phụ thành một vùng rộng hơn dựa trên các tiêu chuẩn đã định nghĩa trước. Việc phân đoạn được bắt đầu từ các điểm tìm kiếm và từ các điểm này các vùng được phát triển bằng cách thêm vào các điểm có tính chất tương đồng. Việc lựa chọn tiêu chuẩn tương đồng còn phụ thuộc vào định dạng dữ liệu của ảnh. Vấn đề khác trong phương pháp phát triển vùng là điều kiện để giải thuật ngừng lại. Một cách cơ bản, vùng sẽ ngừng phát triển khi không còn điểm ảnh nào thỏa tiêu chuẩn để thêm vào vùng.

Nguyên lý của phân đoạn vùng được thực thi dựa trên hàm `regiongrow` với cú pháp

```
[g, NR, SI, TI] = regiongrow(f, S, T)
```

Trong đó `f` là ảnh cần phân đoạn, thông số `S` có thể là mảng (cùng kích thước với `f`) hay số. Nếu `S` là một mảng, nó phải chứa giá trị 1 tại tất cả tọa độ có điểm tìm kiếm và 0 ở nơi khác. Một mảng như vậy có thể được tạo ra bằng việc kiểm tra hay một hàm tìm kiếm riêng. Nếu `S` là một số thì nó định nghĩa độ chói mà tất cả điểm ảnh trong `f` có giá trị này trở thành điểm tìm kiếm. Tương tự `T` cũng có thể là mảng (cùng kích thước với `f`) hoặc số. Nếu `T` là mảng, nó chứa giá trị ngưỡng cho mỗi vùng trong `f`. Nếu `T` là một số thì nó mang ý nghĩa là ngưỡng toàn cục.

Trong ngõ ra, `g` là ảnh đã phân đoạn với mỗi vùng được đánh nhãn theo số nguyên. Thông số `NR` là số lượng vùng khác nhau. `SI` là ảnh chứa các điểm tìm kiếm và `TI` là ảnh chứa các điểm ảnh trải qua quá trình kiểm tra ngưỡng trước khi được xử lý để liên kết. Cả `SI` và `TI` đều có cùng kích thước với `f`.

```
function [g, NR, SI, TI]=regiongrow(f, S, T)
f=double(f);
if numel(S)==1
    SI=f==S;
    S1=S;
else
```

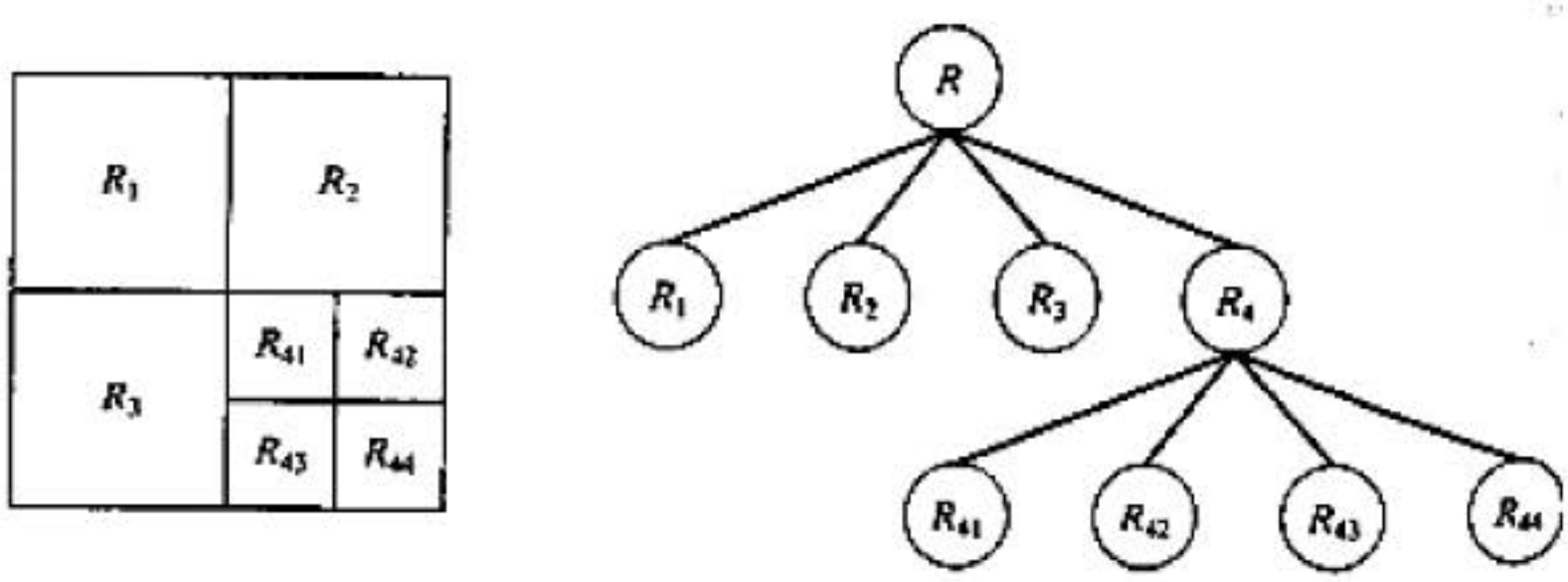

https://tieulun.hopto.org

```
    SI=bwmorph(S, 'shrink', Inf);
    J=find(SI);
    S1=f(J);
end
TI=false(size(f));
for K=1:length(S1)
    seedvalue=S1(K);
    S=abs(f-seedvalue)<=T;
    TI=TI | S;
end
[g, NR]= bwlabel(imreconstruct(SI, TI));
end
```

6.1.3.3. Tách vùng và ghép vùng

Phương pháp phát triển vùng dựa trên một tập các điểm tìm kiếm. Một phương pháp thay thế là chia ảnh thành các tập khởi tạo bất kỳ, ngắt các vùng và sau đó ghép hoặc/và tách các vùng để thỏa mãn các tính chất đã phát biểu trong bảng 6.3.

Đặt R đại diện cho vùng ảnh cần phân đoạn và chọn một tiền đề P. Phân đoạn R tức là chia R thành 4 vùng nhỏ hơn sao cho với bất kỳ vùng R_i thì $P(R_i)= TRUE$. Nếu $P(R)= FALSE$, ta chia ảnh thành 4 phần. Nếu P sai ở bất kỳ phần tư nào thì ta lại chia nó thành 4 phần nhỏ hơn và tiếp tục như trong hình 6.6. Sau khi thực hiện chia ta được các vùng có thuộc tính nhất định. Hai vùng liền kề có thể được ghép lại với nhau nếu $P(R_j \cup R_k) = TRUE$.



Hình 6.6. *Sơ đồ phân vùng ảnh theo kiểu chia bốn*

https://tieulun.hopto.org

Quá trình này có thể được viết thành các bước như trong bảng 6.3.

Bảng 6.3. *Quá trình thực hiện tách vùng và ghép vùng*

1. Chia ảnh thành 4 vùng R_i bất kỳ sao cho $P(R_i) = FALSE$.
2. Khi không thể chia được nữa thì ghép các vùng liền kề R_j và R_k sao cho $P(R_j \cup R_k) = TRUE$.
3. Ngừng khi không thể ghép được nữa.

Hàm thực hiện phân giải trong hộp công cụ là `qtdecomp`. Cú pháp hàm như sau

```
S=qtdecomp(f, @split_test, parameters)
```

Trong đó `f` là ảnh ngõ vào, `S` là ma trận chứa cấu trúc cây.

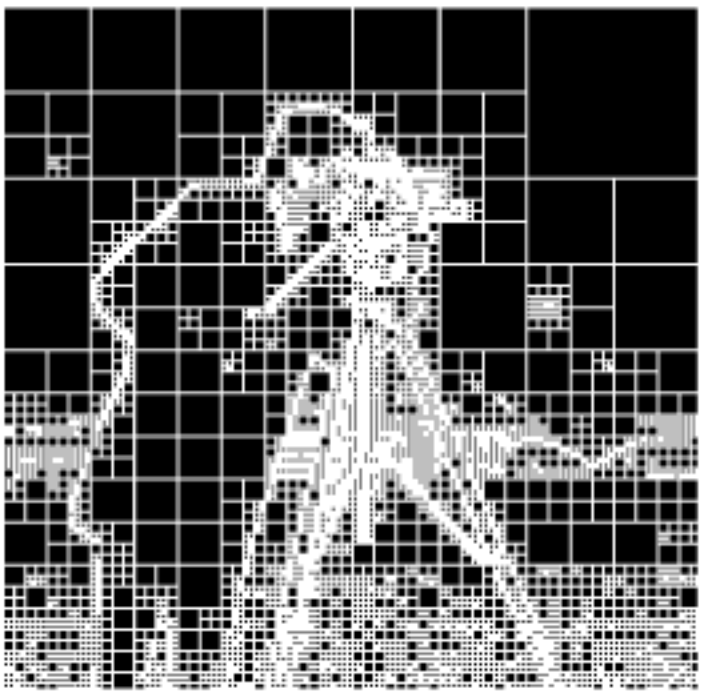
Để lấy được giá trị điểm ảnh trong cây phân giải, ta dùng hàm `qtgetblk` với cú pháp

```
[vals, r, c]= qtgetblk(f, S, m)
```

Trong đó `vals` là một mảng chứa các giá trị của các khối m×m trong cây phân giải của `f` và `S` là ma trận có được từ hàm `qtdecomp`.

Kết quả phân đoạn dựa trên phân giải cây được trình bày trong hình 6.7.

https://tieulun.hopto.org



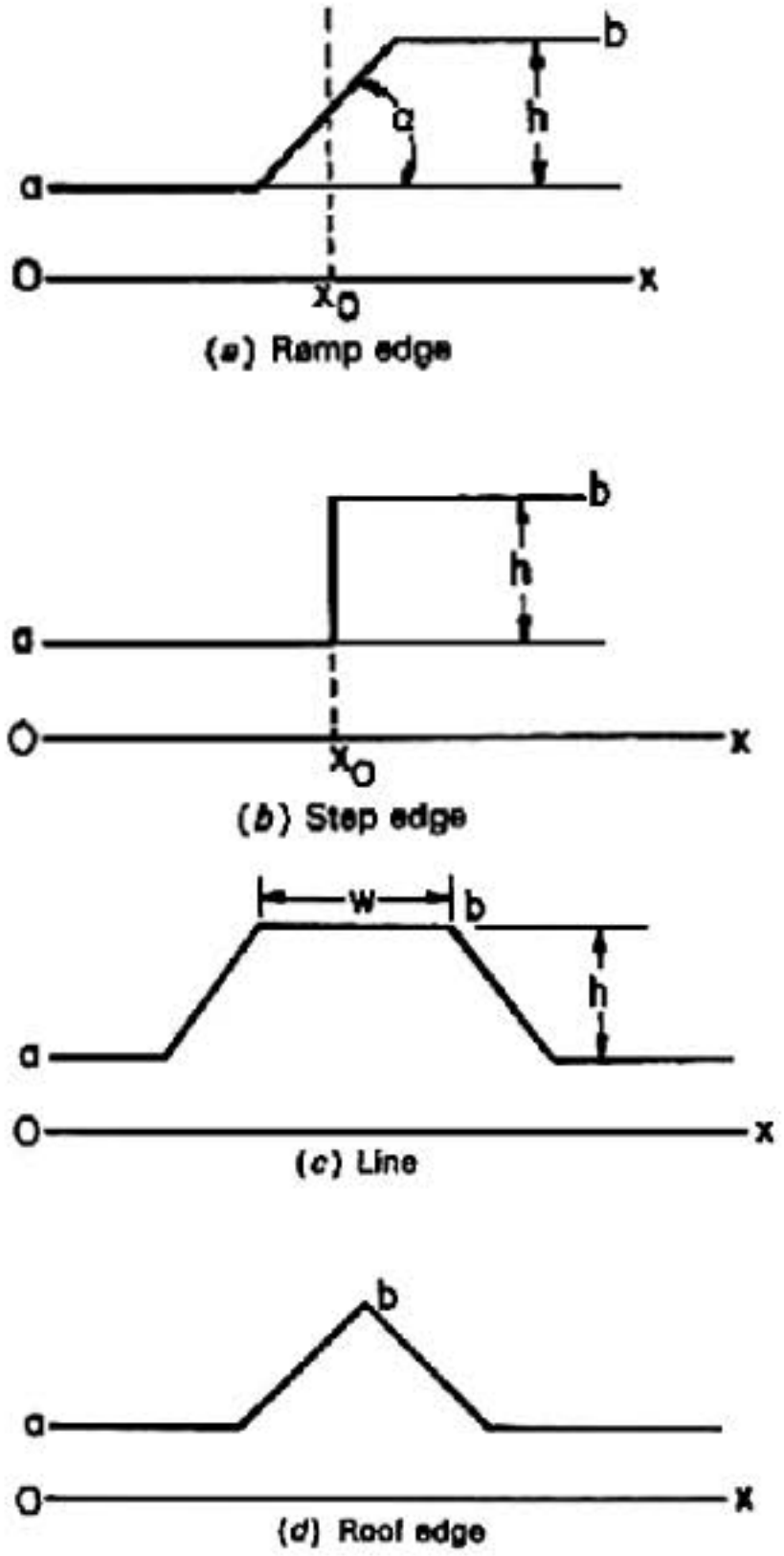
Hình 6.7. Kết quả phân đoạn theo phương pháp tách vùng - phân giải cây

```
I = imread('cameraman.tif');
S = qtdecomp(I,0.2);
blocks =
repmat(uint8(0),size(S));

for dim = [512 256 128 64 32
16 8 4 2 1];
  numblocks =
length(find(S==dim));
  if (numblocks > 0)
    values =
repmat(uint8(1),[dim dim
numblocks]);
    values(2:dim,2:dim,:) = 0;
    blocks =
qtsetblk(blocks,S,dim,values);
  end
end

blocks(end,1:end) = 1;
blocks(1:end,end) = 1;

imshow(I), figure,
imshow(blocks,[])
```

6.2. TÁCH BIÊN

Hình 6.8(a) được vẽ trong miền liên tục là mô hình biên dốc một chiều, trong đó dốc tăng biên độ từ thấp đến cao hoặc ngược lại. Biên được đặc trưng bởi độ cao, dốc, và gốc tọa độ theo chiều ngang ở điểm giữa của dốc. Một biên tồn tại khi chiều cao biên lớn hơn một giá trị đã định nghĩa. Một bộ tách biên lý tưởng cho biết vị trí biên là một pixel đơn tại điểm giữa của dốc. Nếu độ dốc trong hình 6.8(a) là 90° thì kết quả được gọi là biên bước, như trong hình 6.8(b). Trong các ảnh thực tế, ta thường không nhận được biên bước, tức là quá trình thay đổi độ chói không xảy ra đột ngột. Hình 6.8(c) biểu diễn một biên đường và khi độ rộng của đường bằng không, ta có biên nóc như hình 6.8(d).

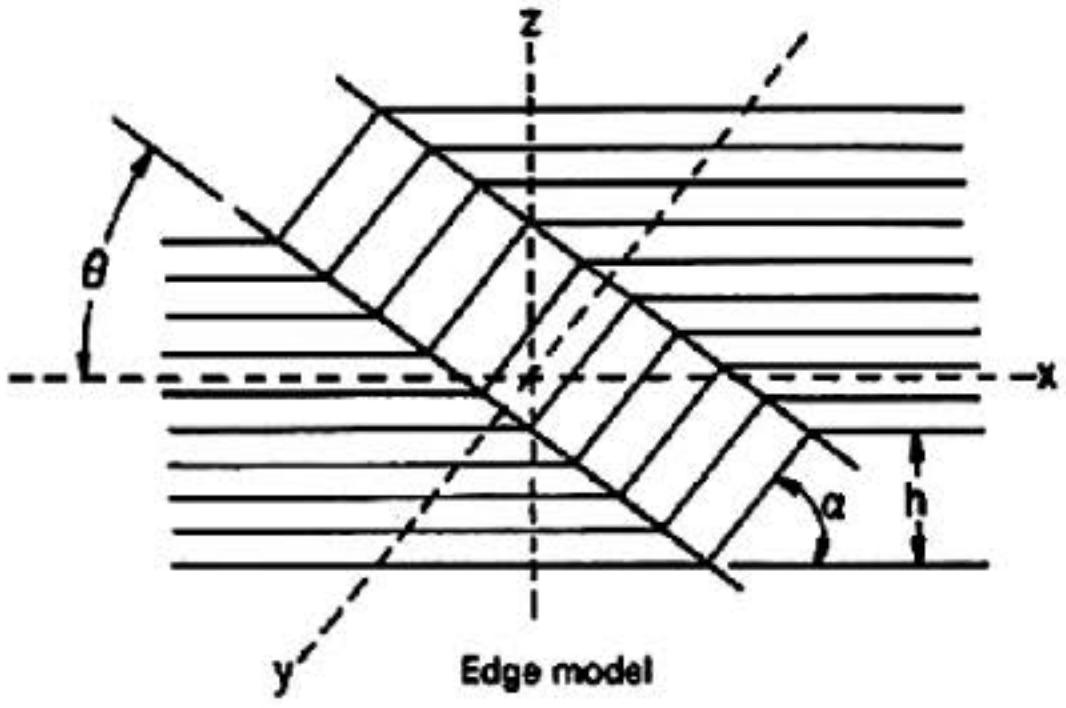
https://tieulun.hopto.org



Hình 6.8. *Các mô hình biên: a) biên dốc; b) biên bước; c) biên đường; d) biên nóc*

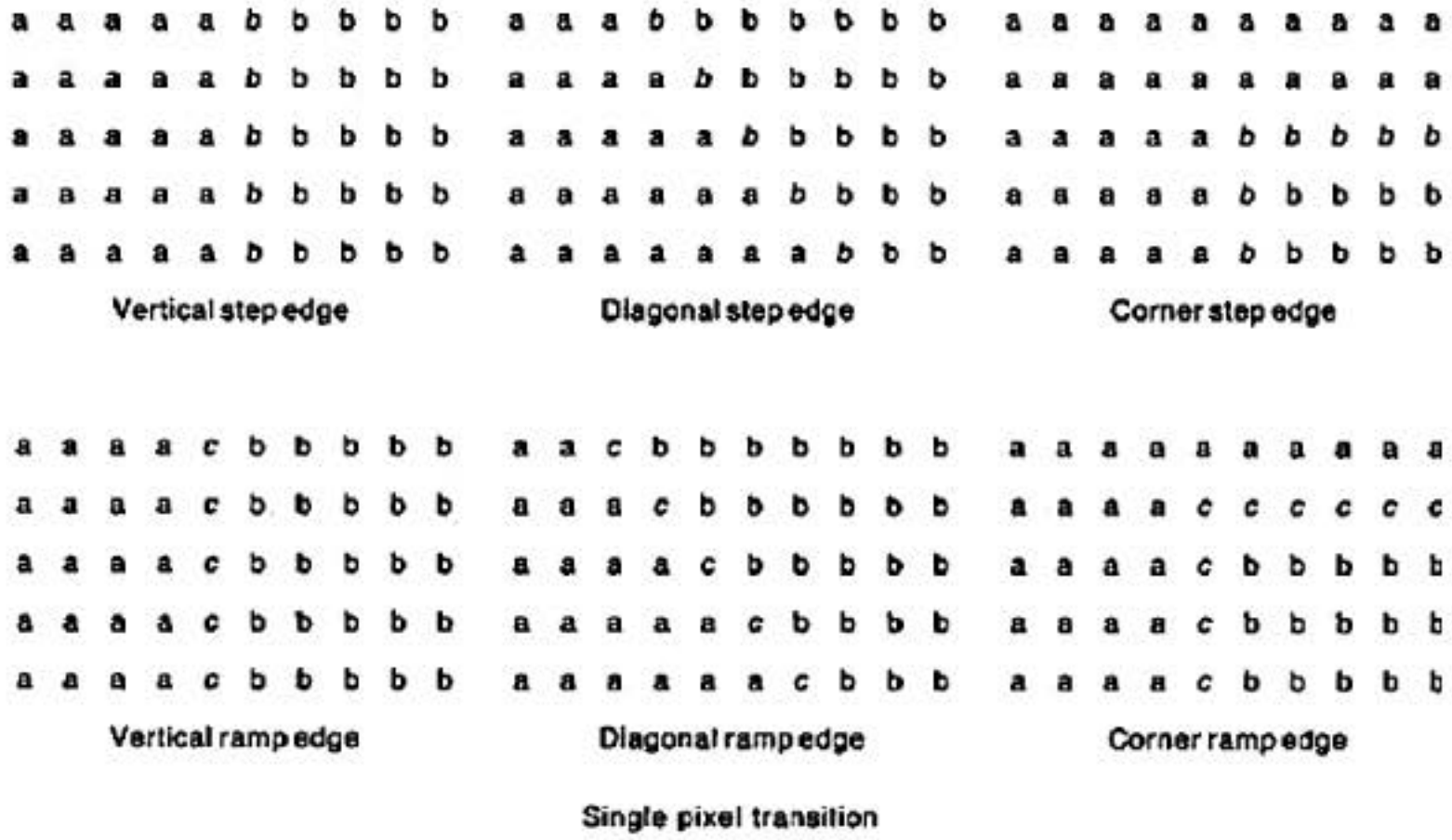
Trong miền hai chiều liên tục, mô hình biên và đường giả sử rằng tính không liên tục của biên độ duy trì trong một khoảng lân cận nhỏ so với biên và đường. Hình 6.9 trình bày một biên hai chiều. Khi xem xét thông số của biên hai chiều cần tham chiếu đến trục tọa độ.

https://tieulun.hopto.org



Hình 6.9. *Biên được xét theo hai chiều*

Hình 6.10 cho thấy mô hình biên bước và dốc trong trường hợp rời rạc. Mô hình biên dốc trong trường hợp này chứa một pixel đơn chuyển vị mà biên độ của chúng bằng trung bình của các pixel liền kề.



Hình 6.10. *Mô hình biên bước và biên dốc*

Mô hình chuyển vị phẳng có được bằng cách thực hiện cửa sổ trung bình dịch chuyển 2x2 trên mô hình biên đường.

https://tieulun.hopto.org

a	a	a	a	c	b	b	b	b	b	a	a	d	e	b	b	b	b	b	b	a	a	a	a	a	a	a	a	a	a
a	a	a	a	c	b	b	b	b	b	a	a	a	d	e	b	b	b	b	b	a	a	a	a	d	c	c	c	c	c
a	a	a	a	c	b	b	b	b	b	a	a	a	a	d	e	b	b	b	b	a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b	a	a	a	a	a	d	e	b	b	b	a	a	a	a	c	b	b	b	b	b
a	a	a	a	c	b	b	b	b	b	a	a	a	a	a	a	d	e	b	b	a	a	a	a	c	b	b	b	b	b

Vertical ramp edge — Diagonal ramp edge — Corner ramp edge

Smoothed transition

$$c = \frac{a+b}{2} \qquad d = \frac{3a+b}{4} \qquad e = \frac{a+3b}{4} \qquad b > a$$

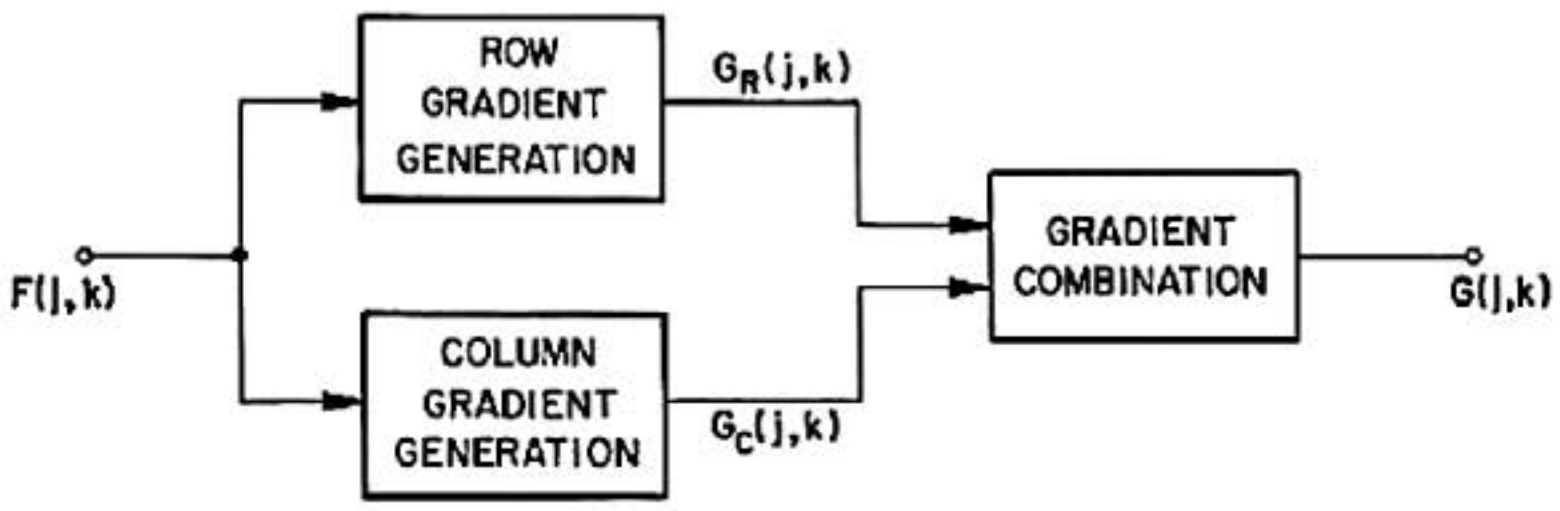
Hình 6.11. *Thực hiện chuyển vị phẳng với cửa sổ trung bình 2x2*

6.2.1. Tách biên theo đạo hàm bậc một

Có hai phương pháp cơ bản để tách biên theo đạo hàm bậc một. Phương pháp thứ nhất là tạo gradient của hai hướng trực giao trong ảnh. Phương pháp thứ hai là dùng một tập đạo hàm có hướng.

Gradient biên của một ảnh $G(j,k)$ được biểu diễn theo gradient hàng $G_R(j,k)$ và cột $G_C(j,k)$ như trong hình 6.12. Biên độ gradient trong không gian được cho bởi

$$G(j,k) = \left[[G_R(j,k)]^2 + [G_C(j,k)]^2\right]^{1/2} \tag{6.8}$$



Hình 6.12. *Gradient theo hàng và cột của ảnh*

Hướng theo không gian của gradient ứng với trục ngang là

$$\theta(j,k) = \arctan\left\{\frac{G_C(j,k)}{G_R(j,k)}\right\} \tag{6.9}$$

Phương pháp đơn giản nhất để tính gradient trong ảnh rời rạc là tìm sự khác nhau giữa các pixel theo hàng và cột. Gradient theo hàng là

$$G_R(j,k) = F(j,k) - F(j,k-1) \tag{6.10}$$

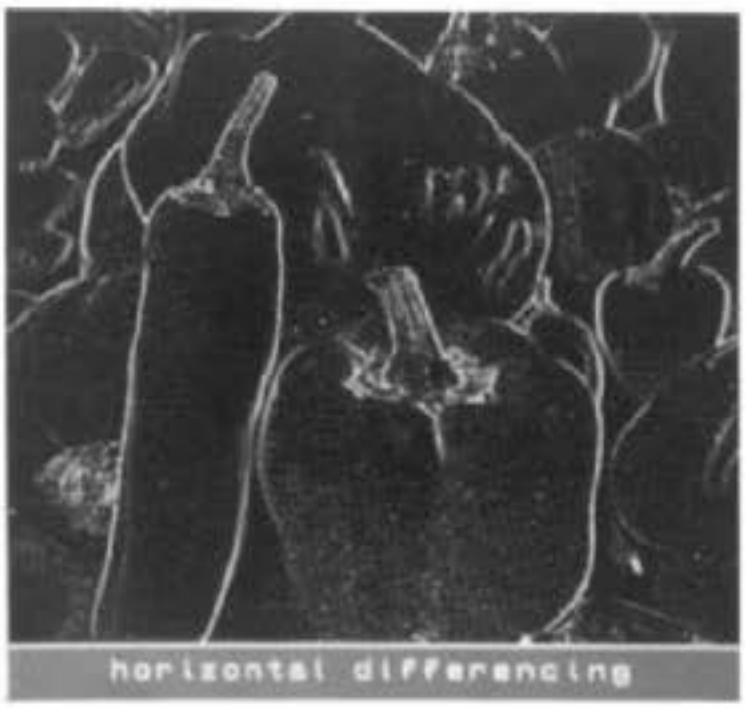
https://tieulun.hopto.org

Và theo cột là

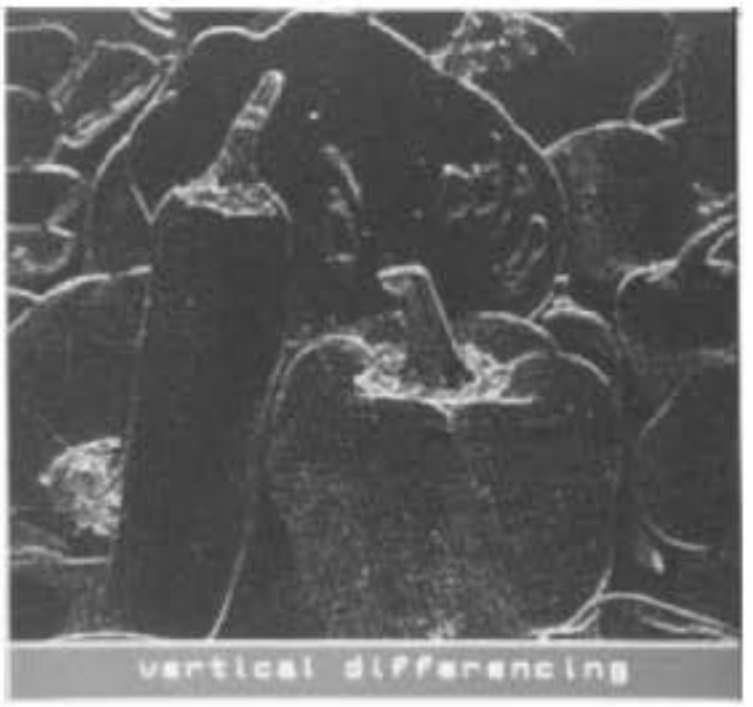
$$G_C(j,k) = F(j,k) - F(j+1,k) \tag{6.11}$$

Hình 6.13 là một ví dụ về gradient theo hàng và cột của ảnh đen trắng. Với phương pháp này, biên của vật thể không được thể hiện rõ.

(*a*) Original



(*b*) Horizontal magnitude



(*c*) Vertical magnitude

Hình 6.13. *Gradient theo hàng và cột*

Gradient đường chéo có thể tính bằng cách tìm sự khác nhau của các cặp theo đường chéo. Đây là phép toán cơ bản của sai phân chéo Roberts. Tính theo biên độ ta có

$$G(j,k) = |G_1(j,k)| + |G_2(j,k)| \tag{6.12}$$

Và theo căn bậc hai

$$G(j,k) = \left[[G_1(j,k)]^2 + [G_2(j,k)]^2\right]^{1/2} \tag{6.13}$$

https://tieulun.hopto.org

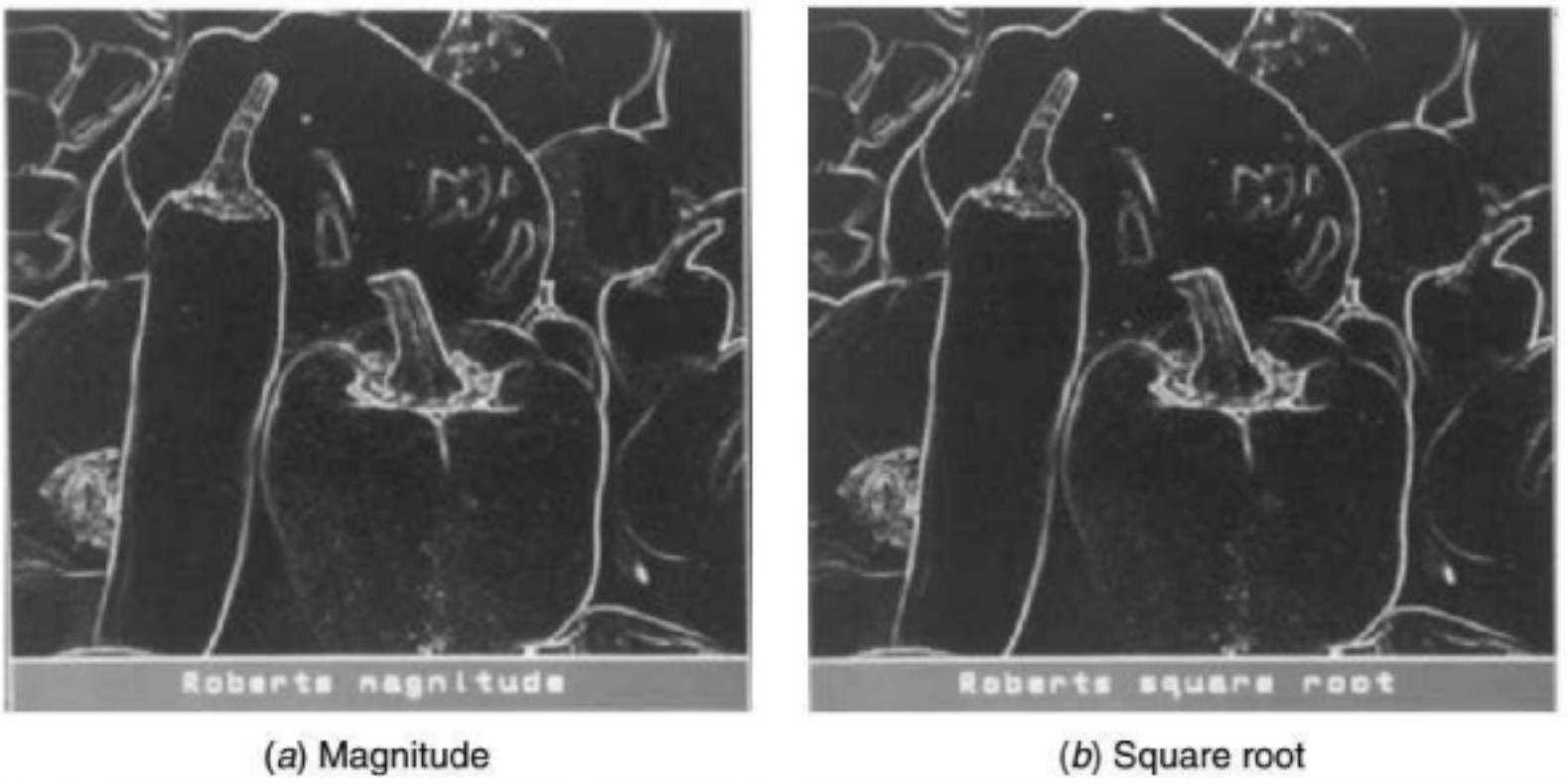
Trong đó

$$G_1(j,k) = F(j,k) - F(j+1,k+1) \tag{6.14}$$

$$G_2(j,k) = F(j,k+1) - F(j+1,k) \tag{6.15}$$

Hướng của biên ứng với trục ngang

$$\theta(j,k) = \frac{\pi}{4} + \arctan\left\{\frac{G_2(j,k)}{G_1(j,k)}\right\} \tag{6.16}$$



(*a*) Magnitude (*b*) Square root

Hình 6.14. *Kết quả gradient biên của ảnh dùng toán tử Roberts*

Mặc dù biên dốc xác định được đường biên của đối tượng nhưng phương pháp này cũng còn rất nhạy với sự thay đổi độ chói nhỏ trong ảnh. Vấn đề này có thể được khắc phục bằng cách dùng gradient hai chiều. Prewitt đã giới thiệu phép toán gradient 3x3 pixel như trong hình 6.15. Phép toán Prewitt căn bậc hai gradient được định nghĩa

$$G(j,k) = \left[[G_R(j,k)]^2 + [G_C(j,k)]^2\right]^{1/2} \tag{6.17}$$

Với

$$G_R(j,k) = \frac{1}{K+2}\left[(A_2 + KA_3 + A_4) - (A_0 + KA_7 + A_6)\right] \tag{6.18}$$

$$G_C(j,k) = \frac{1}{K+2}\left[(A_0 + KA_1 + A_2) - (A_6 + KA_5 + A_4)\right] \tag{6.19}$$

https://tieulun.hopto.org

A_0	A_1	A_2
A_7	$F(j,k)$	A_3
A_6	A_5	A_4

Hình 6.15. *Ma trận tách biên*

Trong đó K=1. Trong công thức này, gradient hàng và cột được chuẩn hóa để có độ lợi bằng ±1 ở vị trí phân biên. Phép toán Sobel khác với Prewitt là giá trị các pixel hướng bắc, nam, đông, tây được nhân đôi (tức là K=2).

Gradient theo hàng và cột của các phương pháp đã trình bày là sự kết hợp tuyến tính của các pixel với các điểm lân cận của nó. Ta có thể biểu diễn bằng quan hệ nhân chập

$$G_R(j,k) = F(j,k) * H_R(j,k) \tag{6.20}$$

$$G_C(j,k) = F(j,k) * H_C(j,k) \tag{6.21}$$

Trong đó $H_R(j,k)$ và $H_C(j,k)$ là các ma trận đáp ứng xung kích thước 3x3 như trong hình 6.16.

Operator	Row gradient	Column gradient
Pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Separated pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

Hình 6.16. *Ma trận các mặt nạ tách biên*

https://tieulun.hopto.org

Trong hộp công cụ xử lý ảnh, biên có thể được tìm dùng hàm `edge` với cú pháp

```
[g, t]= edge(f, 'method', parameters)
```

Sử dụng ảnh `'cameraman.tif'`, ta khảo sát dạng tách biên đã đề cập.

```
f=imread('cameraman.tif')
```

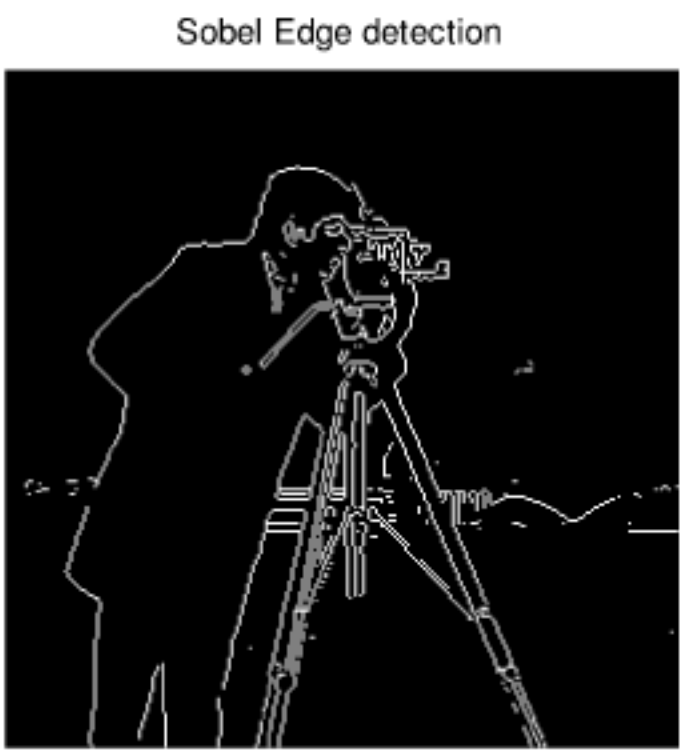
Hình 6.17. *Ảnh ban đầu*

Như vậy cụ thể với từng phương pháp, ta có

Tách biên Sobel

```
[g, t] = edge(f, 'sobel', T, dir)
```

Trong đó `f` là ảnh ngõ vào, `T` là ngưỡng định trước và `dir` là hướng tham khảo của biên cần phát hiện: `'horizontal'`, `'vertical'` hoặc `'both'`.



Hình 6.18. *Kết quả tách biên dùng mặt nạ Sobel*

https://tieulun.hopto.org

Tương tự hàm Prewitt và Roberts lần lượt được sử dụng cho tách biên là

```
[g, t] = edge(f, 'prewitt', T, dir)
```

Prewitt Edge detection

Hình 6.19. *Kết quả tách biên dùng mặt nạ Prewitt*

```
[g, t] = edge(f, 'roberts', T, dir)
```

Roberts Edge detection

Hình 6.20. *Kết quả tách biên dùng mặt nạ Roberts*

6.2.2. Phát hiện biên theo đạo hàm bậc hai

Phương pháp phát hiện biên sử dụng đạo hàm bậc hai được triển khai dựa trên một số dạng vi phân bậc hai để làm nổi biên. Một biên được đánh dấu khi có một sự thay đổi không gian theo đạo hàm bậc hai. Hai dạng của phương pháp đạo hàm bậc hai đã được nghiên cứu là: Laplace và đạo hàm bậc hai trực tiếp.

https://tieulun.hopto.org

6.2.2.1. Phương pháp Laplace

Biến đổi Laplace của một ảnh $F(x, y)$ trong miền liên tục được định nghĩa

$$G(x, y) = -\nabla^2\{F(x, y)\} \tag{6.22}$$

Trong đó, toán tử Laplace là

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \tag{6.23}$$

G(x, y) bằng không khi $F(x, y)$ là hằng số hoặc có biên độ thay đổi tuyến tính. Trong miền rời rạc, dạng xấp xỉ đơn giản nhất của biến đổi Laplace liên tục là tính sự sai biệt của độ dốc trên mỗi trục

$$\begin{aligned} G(j,k) &= [F(j,k) - F(j,k-1)] - [F(j,k+1) - F(j,k)] \\ &+ [F(j,k) - F(j+1,k)] - [F(j-1,k) - F(j,k)] \end{aligned} \tag{6.24}$$

Dạng Laplace 4 điểm này có thể được biểu diễn theo phép chập

$$G(j,k) = F(j,k) * H(j,k) \tag{6.25}$$

Với

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 2 & -1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & -1 & 0 \\ 0 & 2 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

Hay

$$\mathbf{H} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Phương pháp Laplace 4 điểm thường được chuẩn hóa để tạo độ lợi trung bình đơn vị của các điểm ảnh trọng số dương và âm trong ma trận 3x3 điểm ảnh liền kề. Đáp ứng xung chuẩn hóa độ lợi Laplace được định nghĩa

$$\mathbf{H} = \frac{1}{4}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Prewitt đã đề xuất một phương pháp Laplace 8 điểm được định nghĩa bởi ma trận đáp ứng xung

https://tieulun.hopto.org

$$\mathbf{H} = \frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Ma trận này không thể phân tách được thành tổng của 2 đạo hàm bậc hai như trong (6.25). Dạng Laplace 8 điểm phân tách được có thể biểu diễn theo dạng

$$\mathbf{H} = \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} + \begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$$

Trong đó sai biệt về độ dốc là trung bình trên 3 hàng và 3 cột. Dạng Laplace 8 điểm phân tách chuẩn hóa độ lợi được biểu diễn

$$\mathbf{H} = \frac{1}{8}\begin{bmatrix} -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \end{bmatrix}$$

6.2.2.2. Bộ tách biên LoG (Laplacian of Gaussian Detector)

Xét hàm Gauss

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}} \tag{6.26}$$

Trong đó $r^2 = x^2 + y^2$ và σ là độ lệch chuẩn. Đây là một hàm làm phẳng và nếu áp dụng trên ảnh nó sẽ làm mờ ảnh. Mức độ mờ được quyết định bởi độ lệch chuẩn. Laplace của hàm này là

$$\nabla^2 h(r) = -\left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}} \tag{6.27}$$

Việc thực hiên LoG tạo hai hiệu ứng: làm phẳng ảnh và tính Laplace. Cú pháp hàm

```
[g, t]= edge(f, 'log', T, sigma)
```

trong đó `sigma` là độ lệch chuẩn.

https://tieulun.hopto.org

Hình 6.21. *Kết quả tách biên dùng phương pháp LoG*

6.2.3. Bộ tách biên Canny

Phương pháp phát hiện biên Canny là một phương pháp rất tốt và thường được sử dụng. Để thực hiện tách biên Canny, ta thực hiện bốn bước như trong bảng 6.4.

Bảng 6.4. *Bộ tách biên Canny*

i. Ảnh được làm phẳng dùng bộ lọc Gauss.
ii. Gradient cục bộ của biên độ và hướng được tính.
iii. Tìm điểm ảnh có biên độ lớn nhất dùng kỹ thuật nonmaximal suppression. Các điểm ảnh đỉnh (tìm được từ bước ii) được chia thành hai ngưỡng T1 và T2, T1 < T2. Các điểm ảnh đỉnh có giá trị lớn hơn T2 được gọi là 'strong' và nằm trong khoảng T1 và T2 được gọi là 'weak'.
iv. Liên kết các điểm ảnh 'weak' có 8 kết nối đến điểm ảnh 'strong'

Cú pháp cho bộ tách biên Canny:

```
[g, t]= edge(f, 'canny', T, sigma)
```

Kết quả tách biên Canny cho ảnh 'cameraman.tif' được trình bày trong hình 6.22.

https://tieulun.hopto.org

Hình 6.22. *Kết quả thực hiện tách biên Canny*

BÀI TẬP CHƯƠNG 6

6.1. So sánh phương pháp tách biên theo đạo hàm bậc 1 và bậc 2.

6.2. Thực hiện tách biên theo phương pháp Canny cho ảnh 'coins.png'. Thay đổi các thông số ngõ vào của hàm `edge` dùng phương pháp này và rút ra nhận xét.

6.3. Thực hiện phân đoạn theo ngưỡng toàn phần cho ảnh 'coins.png' dùng/không dùng ngưỡng Otsu. So sánh hai kết quả.

6.4. Thực hiện phân đoạn dùng phương pháp phát triển vùng sử dụng hàm `qtdecomp` cho ảnh 'coins.png'.

https://tieulun.hopto.org

Chương 7
TRÍCH ĐẶC TRƯNG VÀ NHẬN DẠNG

Nhận dạng ảnh là khâu quan trọng của hệ thống xử lý ảnh. Nhận dạng là quá trình phân loại các đối tượng được biểu diễn theo một mô hình nào đó. Ảnh thu nhận về thông qua các bước tiền xử lý sẽ được trích đặc trưng. Với những đặc trưng thu được, ta thực hiện huấn luyện và nhận dạng. Chương này trình bày về phương pháp trích đặc trưng cơ bản là PCA. Sau đó, hai mạng nhận dạng được giới thiệu: ANN, SVM.

7.1. PHÂN TÍCH THÀNH PHẦN CHÍNH (PCA)

7.1.1. Giới thiệu

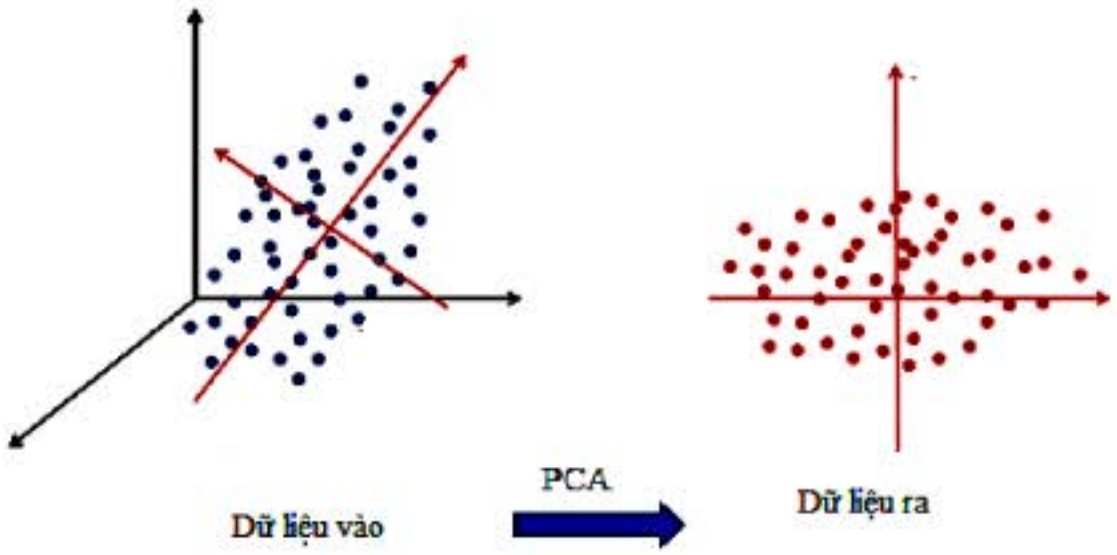
Mục đích của phương pháp PCA là loại bỏ đi một số hướng thành phần trong không gian dữ liệu và chỉ giữ lại các thành phần đặc trưng nhất.

PCA giúp làm giảm số chiều của dữ liệu. Hay nói theo cách khác, thay vì sử dụng các trục tọa độ của không gian cũ, PCA xây dựng một không gian mới có số chiều ít hơn, nhưng lại có thể biểu diễn dữ liệu tốt tương đương không gian cũ, nghĩa là đảm bảo độ biến thiên của dữ liệu trên mỗi chiều mới.

Hình 7.1. *Phép chiếu lên các trục tọa độ khác nhau có thể cho cách nhìn khác nhau về cùng một dữ liệu.*

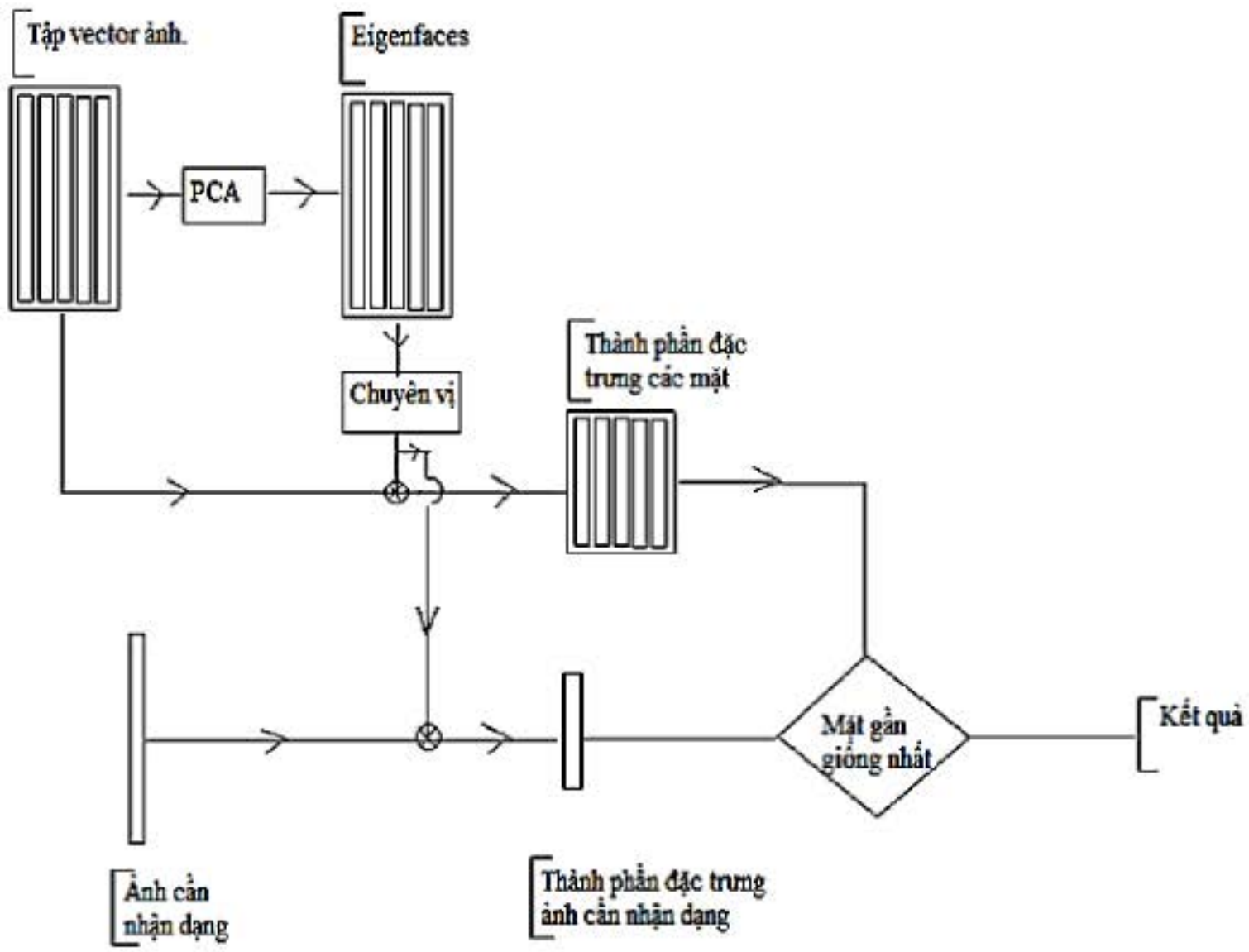
Hình 7.2 với một tập dữ liệu ban đầu (tập điểm màu xanh) được quan sát trong không gian ba chiều (trục màu đen), ta dễ dàng nhận thấy ba trục này không mô tả được dữ liệu một cách tốt nhất. PCA sẽ tìm một hệ trục tọa độ mới (là trục màu đỏ), với không gian mới, dữ liệu được mô tả rõ ràng hơn (hình bên phải màu đỏ). Rõ ràng hình bên phải chỉ cần hai trục tọa độ nhưng cách thể hiện dữ liệu tốt hơn so với hệ trục ban đầu.

https://tieulun.hopto.org



Hình 7.2. *Tìm các trục tọa độ mới sao cho dữ liệu có độ biến thiên cao nhất*

7.1.2. Thuật toán PCA



Hình 7.3. *Sơ đồ tiến trình nhận diện dùng PCA cho một khuôn mặt*

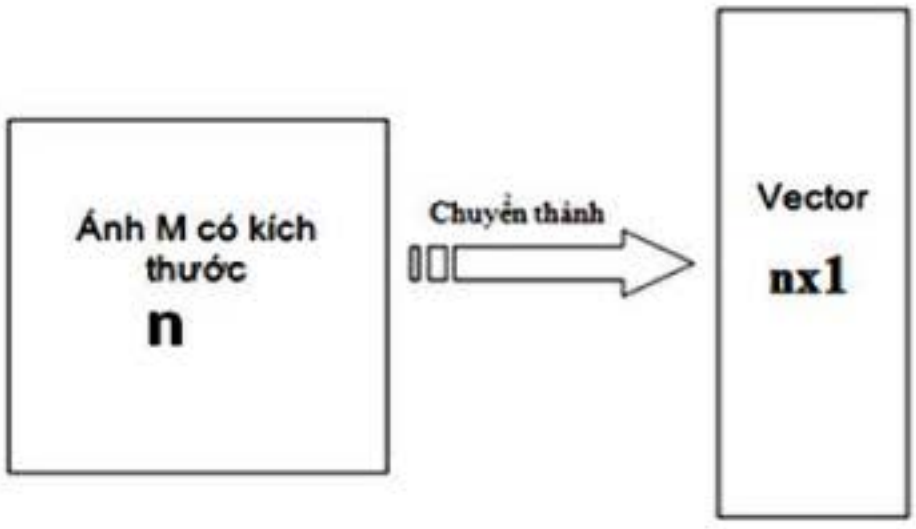
7.1.2.1. Chuyển đổi ảnh

Biểu diễn m ảnh trong không gian 2D thành 1D. Tạo vectơ có kích thước n (số hàng của ảnh × số cột của ảnh) như mô tả sau:

$$\vec{s_i} = [a_1 a_2 .....a_n]^T, \quad i = \overline{1,m} \tag{7.1}$$

với: a_i là giá trị pixel của ảnh, T là chuyển vị của ma trận S_i

https://tieulun.hopto.org



Hình 7.4. *Biểu diễn ảnh có kích thước hàng × cột =* ***n*** *thành vectơ* ***n***×*1*

Mỗi ảnh là một ma trận cột, ghép m ma trận cột ứng với m ảnh thành một ma trận có kích thước $n \times m$.

$$P_{n\times m} = \begin{pmatrix} a_{11}a_{12}a_{13}.....a_{1m} \\ a_{21}a_{22}a_{23}.....a_{2m} \\ a_{31}a_{32}a_{33}.....a_{3m} \\ ... \\ a_{n1}a_{n2}a_{n3}.....a_{nm} \end{pmatrix} \tag{7.2}$$

Chỉ số đầu là thành phần của vectơ, chỉ số sau là số thứ tự của ảnh.

Ví dụ 7.1: Cho một ảnh 2D gồm có 2 cột và 6 dòng như sau:

$$A = \begin{pmatrix} 118 & 30 \\ 119 & 32 \\ 120 & 33 \\ 121 & 31 \\ 122 & 32 \\ 123 & 34 \end{pmatrix}$$

Biến đổi ảnh A thành vectơ 1D có kích thước 12 phần tử:

$$\vec{s} = (118 \quad 119 \quad 120 \quad 121 \quad 122 \quad 123 \quad 30 \quad 32 \quad 33 \quad 31 \quad 32 \quad 34)^T$$

Như vậy số chiều của ảnh đã được giảm đi, thuận tiện cho việc tính toán nhiều ảnh.

Trong Matlab, ta sử dụng các lệnh sau:

```
>>A=[118 30; 119 32; 120 33; 120 31; 121 32; 122 34]; % Tạo ma trận A
>>S = A(:); % tạo vectơ S từ ma trận A
```


https://tieulun.hopto.org

7.1.2.2. Ảnh trung bình

$$\overrightarrow{M} = \frac{1}{m}\sum_{i=0}^{m} x_i \tag{7.3}$$

Hay:

$$\overrightarrow{M} = \frac{1}{m}\begin{pmatrix} a_{11} + a_{12} + a_{13} + ... + a_{1m} \\ a_{21} + a_{22} + a_{23} + ... + a_{2m} \\ a_{31} + a_{32} + a_{33} + ... + a_{3m} \\ ..... \\ a_{n1} + a_{n2} + a_{n3} + ... + a_{nm} \end{pmatrix} = \begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ .... \\ m_n \end{pmatrix} \tag{7.4}$$

Ví dụ 7.2: Giả sử ta có 4 ảnh mẫu 2D được chuyển đổi thành 4 vectơ 1D như sau:

$$S_1 = \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix},\ \overrightarrow{S_2} = \begin{pmatrix} 4 \\ 2 \\ 13 \end{pmatrix},\ \overrightarrow{S_3} = \begin{pmatrix} 7 \\ 8 \\ 1 \end{pmatrix},\ \overrightarrow{S_4} = \begin{pmatrix} 8 \\ 4 \\ 5 \end{pmatrix}$$

Tiến hành ghép 4 vectơ tạo thành ma trận và tính ma trận trung bình:

$$\overrightarrow{M} = \frac{1}{4}\left(\begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} + \begin{pmatrix} 4 \\ 2 \\ 13 \end{pmatrix} + \begin{pmatrix} 7 \\ 8 \\ 1 \end{pmatrix} + \begin{pmatrix} 8 \\ 4 \\ 5 \end{pmatrix}\right) = \frac{1}{4}\begin{pmatrix} 20 \\ 16 \\ 20 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 5 \end{pmatrix}$$

Trong Matlab, dùng lệnh `mean(X,dim)` để tính giá trị trung bình của ma trận *X*, với `dim` là chiều lấy trung bình, nếu `dim` bằng 1 lấy trung bình theo cột, nếu `dim` bằng 2 lấy trung bình theo hàng. Không có tham số `dim` thì mặc định `dim` bằng 1.

7.1.2.3. Sai biệt ảnh trung bình

Nhằm mục đích tạo ra sự giãn tương đối giá trị pixel của các ảnh

$$\overrightarrow{n_{1m}} = \begin{pmatrix} a_{11} - m_1 \\ a_{21} - m_2 \\ a_{31} - m_3 \\ ... \\ a_{n1} - m_n \end{pmatrix},\ \overrightarrow{n_{2m}} = \begin{pmatrix} a_{12} - m_1 \\ a_{22} - m_2 \\ a_{32} - m_3 \\ ... \\ a_{n2} - m_n \end{pmatrix},.....,\ \overrightarrow{n_{1m}} = \begin{pmatrix} a_{1m} - m_1 \\ a_{2m} - m_2 \\ a_{3m} - m_3 \\ ... \\ a_{nm} - m_n \end{pmatrix} \tag{7.5}$$

https://tieulun.hopto.org

Xây dựng ma trận từ các $\overrightarrow{n_{im}}$ vừa tìm được

Đặt $A = \left(\overrightarrow{n_{1m}} \ \overrightarrow{n_{2m}} \ \overrightarrow{n_{3m}} \ ..... \ \overrightarrow{n_{Mm}}\right)$ sẽ được ma trận có kích thước $n \times m$.

Ví dụ 7.3: Từ ví dụ 7.2, ta tính được ma trận A như sau:

$$A = \begin{pmatrix} -4 & -1 & 2 & 3 \\ -2 & -2 & 4 & 0 \\ -4 & 8 & -4 & 0 \end{pmatrix}$$

7.1.2.4.Ma trận hiệp phương sai

Nhằm mục đích thể hiện sự tương quan của từng vectơ đối với các vectơ còn lại trong không gian.

$$\text{cov} = \frac{1}{n-1} AA^T \tag{7.6}$$

Trị riêng *(eigenvalue* λ_i*),* và vectơ riêng *(eigenvector* x_i*)* của ma trận hiệp phương sai này chính là đặc trưng thành phần thiết yếu của ảnh.

Nhưng thực tế, nếu ảnh có kích thước *200×230,* thì khi đó kích thước của ma trận *cov* là *46000×46000*. Kích thước khá lớn, do đó việc tính trị riêng, vectơ riêng là vấn đề trở ngại khi tính trực tiếp theo cách này. Vì vậy cần phải áp dụng lý thuyết đại số tuyến tính: trị riêng λ , và vectơ riêng có thể tính bằng cách giải quyết trị riêng, và vectơ riêng của ma trận A^TA *(kích thước m×m nhỏ hơn nhiều so với n×n).*

Đặt μ_i và d_i là các trị riêng, và vectơ riêng của ma trận $A^T A$. Kết quả như sau:

$$A^T A d_i = \mu_i d_i \tag{7.7}$$

Nhân mỗi vế của (7.6) cho A sẽ được:

$$AA^T (Ad_i) = \mu_i (Ad_i) \tag{7.8}$$

Với $X = Ad_i$

Điều này cho thấy m vectơ riêng x_i và, m trị riêng λ_i đầu tiên của AA^T tương ứng chính là tích A với vectơ riêng d_i của $A^T A$, và μ_i.

Các vectơ riêng là không gian đặc trưng các tập mẫu trong cơ sở dữ liệu ảnh ban đầu. Các vectơ riêng được sắp xếp theo thứ tự từ cao đến

https://tieulun.hopto.org

thấp theo trị riêng tương ứng. Vectơ riêng có trị riêng lớn nhất sẽ mang nhiều đặc trưng thiết yếu nhất của không gian các tập mẫu (tức nó quyết định nhiều nhất sự biến đổi trong ảnh). Ngược lại, vectơ riêng có trị riêng bé nhất sẽ mang thành phần ít đặc trưng nhất trong không gian đặc trưng các tập mẫu. Ở đây cho thấy chỉ với *m* hướng đặc trưng mang trị riêng lớn nhất trong $n \times n$ không gian đặc trưng.

Ví dụ 7.4: Ma trận hợp phương sai được tính từ ma trận *A* trong ví dụ 7.3 theo phương trình (7.6) như sau:

$$\text{cov} = \frac{1}{3}\begin{pmatrix} -4 & -1 & 2 & 3 \\ -2 & -2 & 4 & 0 \\ -4 & 8 & -4 & 0 \end{pmatrix}\begin{pmatrix} -4 & -2 & -4 \\ -1 & -2 & 8 \\ 2 & 4 & -4 \\ 3 & 0 & 0 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 30 & 18 & 0 \\ 18 & 24 & -24 \\ 0 & -24 & 96 \end{pmatrix} = \begin{pmatrix} 10 & 6 & 0 \\ 6 & 8 & -8 \\ 0 & -8 & 32 \end{pmatrix}$$

Ma trận *cov* có các trị riêng và vectơ riêng như sau:

$$\vec{\lambda} = (1.6057 \quad 13.8430 \quad 34.5513)^T$$

$$\overrightarrow{x_1} = (0.5686 \quad -0.7955 \quad -0.2094)^T$$

$$\overrightarrow{x_2} = (0.8193 \quad 0.5247 \quad 0.2312)^T$$

$$\overrightarrow{x_3} = (-0.0740 \quad -0.3030 \quad 0.9501)^T$$

Trong Matlab, dùng các hàm sau:

`E=eig(X)`: Trả về một vector chứa các giá trị riêng của ma trận vuông X.

`[V, D] = eig(X)`: tạo ra một ma trận đường chéo D của các giá trị riêng và một ma trận V có các cột tương ứng là các vector riêng, do đó: X * V = V * D

7.1.2.5. Phép chiếu

Chiếu lần lượt các ảnh trong cơ sở dữ liệu đến không gian đặc trưng *m* hướng này, để sinh ra các ảnh đặc trưng trong không gian ảnh mới.

$$\Omega_i = [x_1 x_2 .... x_n]^T . \overrightarrow{n_{im}} \quad i = \overline{1, m} \tag{7.9}$$

Với $[X]^T = \left[\overrightarrow{x_1} \ \overrightarrow{x_2} \ .... \ \overrightarrow{x_M}\right]^T$ là ma trận đặc trưng các ảnh mẫu đã rút trích ra được gọi là các eigensignal, $\overrightarrow{n_{im}}$ là vectơ ảnh thứ *i* trừ ảnh trung bình.

https://tieulun.hopto.org

7.1.2.6. Nhận dạng

Chuyển đổi ảnh cần nhận dạng thành vectơ 1 chiều:

$$\vec{r} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ \dots \\ r_N \end{pmatrix} \tag{7.10}$$

Tính sự sai số của ảnh cần nhận dạng với ảnh trung bình của các ảnh trong cơ sở dữ liệu.

$$\vec{r} = \begin{pmatrix} r_1 - m_1 \\ r_2 - m_2 \\ r_3 - m_3 \\ \dots \\ r_N - m_N \end{pmatrix} \tag{7.11}$$

Chiếu sai số lên không gian đặc trưng của các ảnh mẫu trong cơ sở dữ liệu.

$$\Omega_r = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_M \end{bmatrix}^T . \overrightarrow{r_m} \tag{7.12}$$

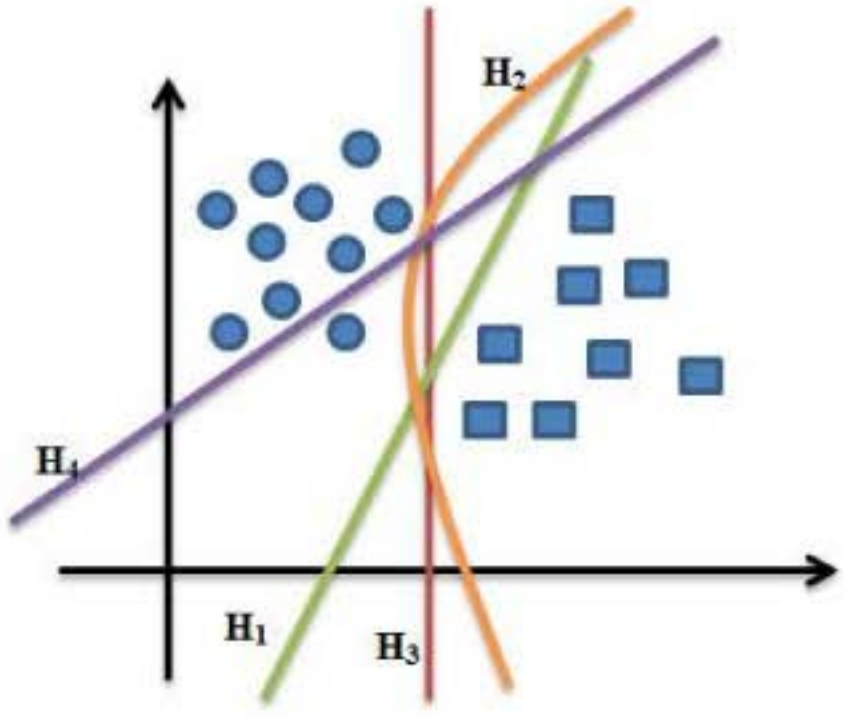
Từ đặc trưng đã trích từ phương pháp PCA, các hệ số biểu diễn ảnh có thể dùng để làm ngõ vào của một mạng phục vụ cho việc huấn luyện và phân loại.

7.2. MÁY VECTOR HỖ TRỢ (SVM)

7.2.1. Siêu phẳng - Hyperplane

Siêu phẳng là một yếu tố quan trọng trong thuật toán SVM. Giả sử ta có hai tập dữ liệu với các điểm như trên hình 7.5. Các điểm này thuộc về một trong hai tập. Như vậy vấn đề ở đây là nếu có một điểm dữ liệu mới thì ta sẽ xác định nó thuộc về tập nào.

https://tieulun.hopto.org

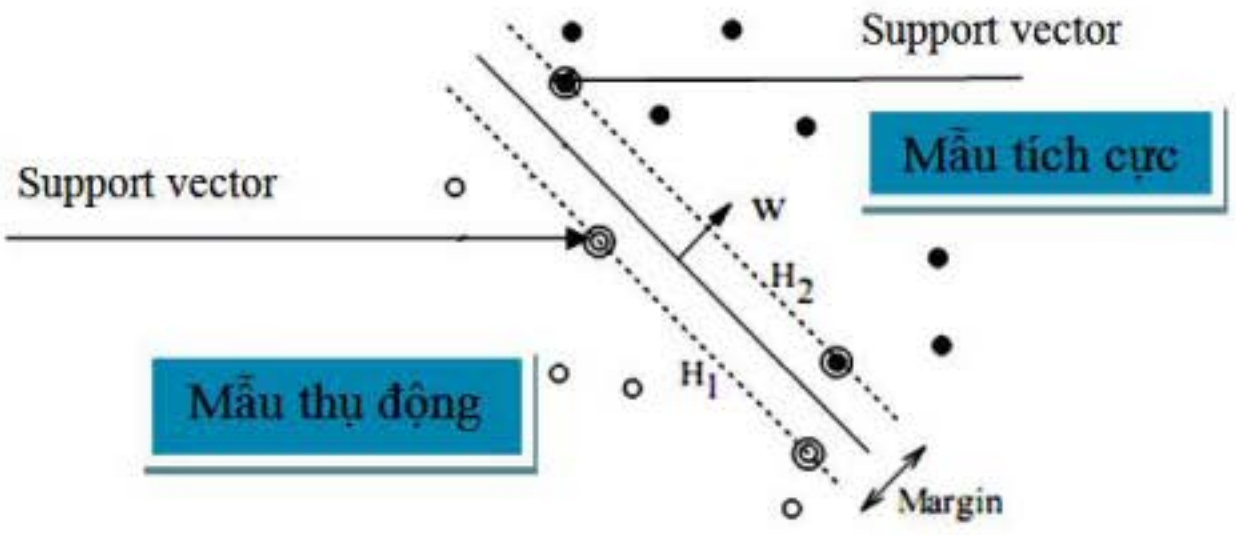


Hình 7.5. *Các đường phân chia hai tập dữ liệu mang đặc trưng khác nhau*

Một cách trực quan trên hình 7.5, ta thấy có thể có ba đường H_1, H_2, H_3 có thể chia hai tập dữ liệu thành hai miền khác nhau. Tuy nhiên, với đường phân chia H_4, đã có một điểm bị phân loại nhầm. Như vậy, việc tính toán các đường để chia các tập dữ liệu là hết sức quan trọng, và các đường này gọi là siêu phẳng. Ta cần tìm các siêu phẳng sao cho nó có độ phân cách lớn nhất hay lề (margin) giữa hai tập dữ liệu.

7.2.2. SVM tuyến tính

SVM tuyến tính thực hiện phân chia hai tập dữ liệu theo siêu phẳng tuyến tính. Giả sử ta có tập huấn luyện $\{\mathbf{x}_i, y_i\}$, $i = 1,\ldots,l$, $y_i \in \{-1,1\}$, $\mathbf{x}_i \in R^d$. Ta muốn phân chia các mẫu tích cực ra khỏi các mẫu thụ động như trong hình 7.6. Các mẫu tích cực nằm trong vùng có $y = 1$ hay vùng H_2, và các mẫu thụ động nằm trong vùng $y = -1$ tương ứng vùng H_1.



Hình 7.6. *Các siêu phẳng H_1 và H_2 phân chia các mẫu tích cực và thụ động thành hai lớp khác nhau*

Các điểm $\mathbf{x}_i$ nằm trên siêu phẳng thỏa mãn $\mathbf{w}.\mathbf{x} + b = 0$. Và $|b|/\|\mathbf{w}\|$ là khoảng cách từ siêu phẳng đến gốc tọa độ, trong đó $\|\mathbf{w}\|$ là độ lớn của

https://tieulun.hopto.org

w. Đặt d_+ (d_-) là khoảng cách ngắn nhất từ siêu phẳng đến mẫu tích cực (thụ động) gần nhất.

Lề của siêu phẳng là $d_+ + d_-$. Trong trường hợp tuyến tính này, vector hỗ trợ tìm các siêu phẳng phân cách với lề (margin) lớn nhất. Giả sử tất cả dữ liệu huấn luyện thỏa điều kiện:

$$\mathbf{x}_i.w + b \geq +1, \text{với } y_i = +1 \tag{7.13}$$

$$\mathbf{x}_i.w + b \leq -1, \text{với } y_i = -1 \tag{7.14}$$

Kết hợp (7.13) và (7.14)

$$y_i(\mathbf{x}_i.\mathbf{w} + b) - 1 \geq 0 \tag{7.15}$$

Khi dấu bằng trong (7.13) xảy ra, ta có các điểm nằm trên siêu phẳng $H_1 : \mathbf{x}_i.\mathbf{w} + b = 1$ và khoảng cách đến gốc tọa độ là $|1-b| / \| \mathbf{w} \|$. Tương tự, các điểm nằm trên siêu phẳng $H_2 : \mathbf{x}_i.\mathbf{w} + b = -1$ có khoảng cách đến gốc tọa độ là $|-1-b| / \| \mathbf{w} \|$. Vì thế, $d_+ = d_- = 1/\|\mathbf{w}\|$ và độ lớn lề là $2/\|\mathbf{w}\|$. Như vậy, ta có thể tìm cặp siêu phẳng sao cho độ lớn lề là lớn nhất theo phương trình (7.15).

Vấn đề ở đây là tìm **w** và *b*. Công việc tối ưu này được thực hiện bằng cách chuyển (7.15) sang vấn đề tương ứng với tiên đề Lagrange

$$L_p(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{l} \alpha_i y_i (\mathbf{x}_i.\mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i \tag{7.16}$$

Trong đó, $\alpha_i \geq 0$ là các nhân tử Lagrange.

Lấy đạo hàm L_p theo **w** và *b*, và đặt kết quả bằng 0, ta có

$$\begin{aligned} \frac{\partial L_p(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i = 0 \\ \frac{\partial L_p(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} &= \sum_{i=1}^{l} y_i \alpha_i = 0 \end{aligned} \tag{7.17}$$

Cuối cùng ta được

$$\mathbf{w} = \sum_{i=1}^{l} y_i \alpha_i \mathbf{x}_i \tag{7.18}$$

$$0 = \sum_{i=1}^{l} y_i \alpha_i \tag{7.19}$$

https://tieulun.hopto.org

Mỗi mẫu huấn luyện $\mathbf{x}_i$ tương ứng với một hệ số Lagrange α_i. Sau khi huấn luyện các mẫu có $\alpha_i \geq 0$ được gọi là vector hỗ trợ và nằm trên một trong hai siêu phẳng đã nói.

Khi đã có vector hỗ trợ được huấn luyện, ta có thể quyết định một cách đơn giản một mẫu thử **x** thuộc về vùng nào do các siêu phẳng H_1 và H_2 tạo ra bằng cách đặt lớp của **x** là hàm dấu

$$\mathrm{sgn}(\mathbf{w} \cdot \mathbf{x} + b) \tag{7.20}$$

7.2.3. Các điều kiện Karush-Kuhn-Tucker (KKT)

Các điều kiện KKT đóng một vai trò quan trọng trong việc giải quyết tối ưu các vấn đề lập trình phi tuyến trong thực tế. Điều kiện KKT phát biểu như sau:

$$\alpha_i = 0 \Leftrightarrow y_i(\mathbf{x}_i\mathbf{w} + b) \geq 1 \tag{7.21}$$

$$\alpha_i = C \Leftrightarrow y_i(\mathbf{x}_i\mathbf{w} + b) \leq 1 \tag{7.22}$$

$$0 < \alpha_i < C \Leftrightarrow y_i(\mathbf{x}_i\mathbf{w} + b) = 1 \tag{7.23}$$

Việc giải quyết các vấn đề liên quan đến SVM cũng tương tự như việc tìm đáp án cho các điều kiện KKT.

7.2.4. Giải thuật SMO

Để tối ưu thời gian, bộ nhớ trong việc tìm các nhân tử Lagrange, cũng như các trọng số **w** và support vector, có nhiều giải thuật đã được đề ra: Chunking, Osuna, SMO. Giải thuật tối ưu cực tiểu tuần tự - SMO (Sequential Minimal Optimisation) dựa trên ý tưởng của phương pháp phân giải và tối ưu tập cực tiểu chỉ có hai phân tử trong mỗi vòng lặp.

Với điều kiện $\sum_{i=1}^{l} \alpha_i y_i = 0$ thì số lượng nhân tử ít nhất cần phải có là 2, để đảm bảo khi một nhân tử được cập nhật thì nhân tử còn lại cũng phải được điều chỉnh để thỏa điều kiện trên. Như vậy, trong mỗi bước của giải thuật SMO có phần tử α_i và α_j được lựa chọn.

Giả sử ta đã có 2 phần tử được chọn là $\alpha_1$và α_2. Khi thay đổi giá trị này thì giá trị kia phải thay đổi theo để thỏa điều kiện ràng buộc, do đó

$$\alpha_1 y_1 + \alpha_2 y_2 = constant = \alpha_1^{old} y_1 + \alpha_2^{old} y_2 \tag{7.24}$$

Hay nói cách khác, các giá trị mới này nằm trên một đường thẳng trong không gian (α_1,α_2) và $0 \leq \alpha_1,\alpha_2 \leq C$, trong đó α_1^{old}, α_2^{old} là giá trị trước đó.

https://tieulun.hopto.org

Giải thuật sẽ tính giá trị mới α_2^{new}:

$$U \le \alpha_2^{new} \le V \tag{7.25}$$

Trong đó, nếu $y_1 \neq y_2$ thì

$$\begin{aligned} U &= \max(0, \alpha_2^{old} - \alpha_1^{old}) \\ V &= \min(C, C - \alpha_1^{old} + \alpha_2^{old}) \end{aligned} \tag{7.26}$$

Và nếu $y_1 = y_2$ thì

$$\begin{aligned} U &= \max(0, \alpha_1^{old} + \alpha_2^{old} - C) \\ V &= \min(C, \alpha_1^{old} + \alpha_2^{old}) \end{aligned} \tag{7.27}$$

Một cách rõ ràng hơn, đặt E_i là sai biệt giữa hàm ngõ ra và mục tiêu phân loại trên các mẫu huấn luyện $\mathbf{x}_1$ hoặc $\mathbf{x}_2$,

$$E_i = f(\mathbf{x}_i) - y_i = \left(\sum_{j=1}^{l} \alpha_j y_j K(x_j, x_i) + b \right) - y_i, \qquad i = 1,2 \tag{7.28}$$

Và đặt

$$\gamma = K(\mathbf{x}_1, \mathbf{x}_1) + K(\mathbf{x}_2, \mathbf{x}_2) - 2K(\mathbf{x}_1, \mathbf{x}_2) = \| \phi(\mathbf{x}_1) - \phi(\mathbf{x}_2) \|^2 \tag{7.29}$$

Trong đó $K(x_i,x_j)$ là hàm lõi (kernel function).

Tính toán một cách định lượng thì, giá trị mới của α_2 chưa bị giới hạn là

$$\alpha_2^{new,unc} = \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\gamma} \tag{7.30}$$

Để thỏa điều kiện $U \le \alpha_2^{new} \le V$ thì

$$\alpha_2^{new} = \begin{cases} V, khi & \alpha_2^{new,unc} > V \\ \alpha_2^{new,unc}, khi & U \le \alpha_2^{new,unc} \le V \\ U, khi & \alpha_2^{new,unc} < U \end{cases} \tag{7.31}$$

Và ta tính được giá trị α_1^{new}:

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2^{new}) \tag{7.32}$$

Tóm lại, giải thuật SMO thực hiện các bước:

https://tieulun.hopto.org

- Bước 1: Tìm một nhân tử Lagrange α_1 không thỏa điều kiện KKT.
- Bước 2: Tìm tiếp nhân tử α_2, tính các giá trị $\alpha_1^{new}, \alpha_2^{new}$.
- Bước 3: Lập lại bước 1, và 2 cho đến khi hội tụ.

Để tăng tốc độ hội tụ, giải thuật SMO sử dụng phỏng đoán heuristic để chọn 2 nhân tử Lagrange trong vấn đề tối ưu. Có 2 phỏng đoán heuristic độc lập nhau trong việc chọn nhân tử thứ nhất và nhân tử thứ hai:

- Phỏng đoán chọn lựa thứ nhất: Đây là vòng lặp ngoài của giải thuật SMO. Vòng lặp ngoài sẽ tìm trên tập huấn luyện các mẫu không thỏa điều kiện KKT. Nếu một mẫu không thỏa điều kiện KKT, nó được dùng cho việc tối ưu. Khi đã tìm xong trên cả tập, vòng lặp ngoài sẽ tìm các mẫu sao cho nhân tử Lagrange của chúng không bằng 0 hoặc không bằng C (các mẫu không biên). Một lần nữa, các mẫu này được kiểm tra và nếu không thỏa điều kiện KKT thì nó được dùng cho việc tối ưu. Vòng lặp ngoài sẽ lặp lại trên các mẫu không biên cho đến khi các điều kiện KKT thỏa sai số ε.
- Phỏng đoán chọn lựa thứ hai: Khi nhân tử Lagrange thứ nhất được chọn, SMO chọn nhân tử thứ 2 sao cho bước nhảy trong phương trình 7.29 là lớn nhất, tức là tối đa $|E_1 - E_2|$. Nếu E_1 dương, SMO chọn một mẫu sao cho E_2 là nhỏ nhất. Nếu E_1 âm, SMO chọn một mẫu với sai số E_2 lớn nhất.

Với việc cập nhật các nhân tử Lagrange, ngưỡng b sẽ được tính lại ở mỗi bước. ngưỡng b_1 được tính:

$$b_1 = E_1 + y_1(\alpha_1^{new} - \alpha_1)K(\mathbf{x}_1, \mathbf{x}_1) + y_1(\alpha_2^{new} - \alpha_2)K(\mathbf{x}_1, \mathbf{x}_2) + b \quad (7.33)$$

Và b_2:

$$b_2 = E_2 + y_1(\alpha_1^{new} - \alpha_1)K(\mathbf{x}_1, \mathbf{x}_2) + y_1(\alpha_2^{new} - \alpha_2)K(\mathbf{x}_2, \mathbf{x}_2) + b \quad (7.34)$$

Khi b_1 và b_2 hợp lệ chúng sẽ bằng nhau.

Vector trọng số $\mathbf{w}$ cũng được cập nhật:

$$\mathbf{w}^{new} = \mathbf{w} + y_1(\alpha_1^{new} - \alpha_1)\mathbf{x}_1 + y_2(\alpha_2^{new} - \alpha_2)\mathbf{x}_2 \quad (7.35)$$

Như vậy, giải thuật SMO được đề ra nhằm góp phần tăng tốc độ hội tụ cũng như thời gian xử lý trên máy tính trong vấn đề huấn luyện trên thuật toán SVM.

https://tieulun.hopto.org

Trong MATLAB, công cụ SVM được hỗ trợ thông qua `svmsmoset, svmtrain, svmclassify.`

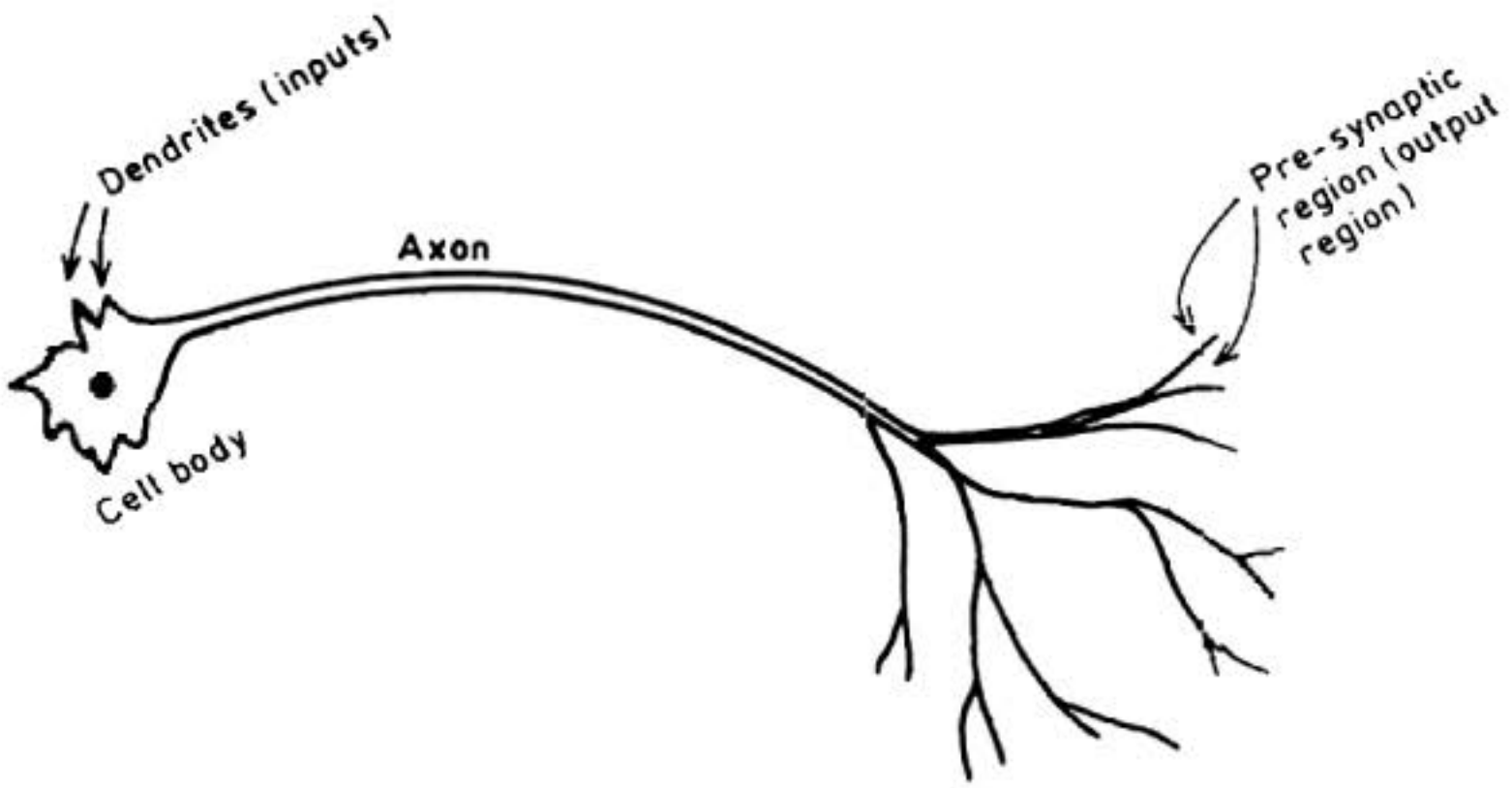
Ví dụ 7.5: Khai báo thông số, huấn luyện và nhận dạng dùng công cụ SVM.

```
SMO_OptsStruct = svmsmoset('TolKKT',0.0005,'MaxIter',10000);
svmStruct = svmtrain(dt,g,'SMO_Opts',SMO_OptsStruct);
classes = svmclassify(svmStruct,dtest);
```

`svmsmoset` cho người dùng khai báo các thông số về điều kiện KKT, số vòng lặp. Trong khi đó, `svmtrain` thực hiện huấn luyện ngõ vào `dt` theo ngõ ra mong muốn `g`. Cuối cùng, với `svmclassify` thực hiện phân loại trên cấu trúc SVM đã khai báo với ngõ vào cần phân loại.

7.3. MẠNG NƠ-RON NHÂN TẠO

Mạng nơ-ron nhân tạo (ANN - Artifical Neural Networks) là một trong những công cụ rất mạnh trong vấn đề phân loại và nhận dạng đối tượng. Dựa trên cấu trúc của nơ-ron sinh học như hình 7.7 trong hệ thống thần kinh người, nhiều dạng mạng nơ-ron nhân tạo đã được xây dựng và phát triển.

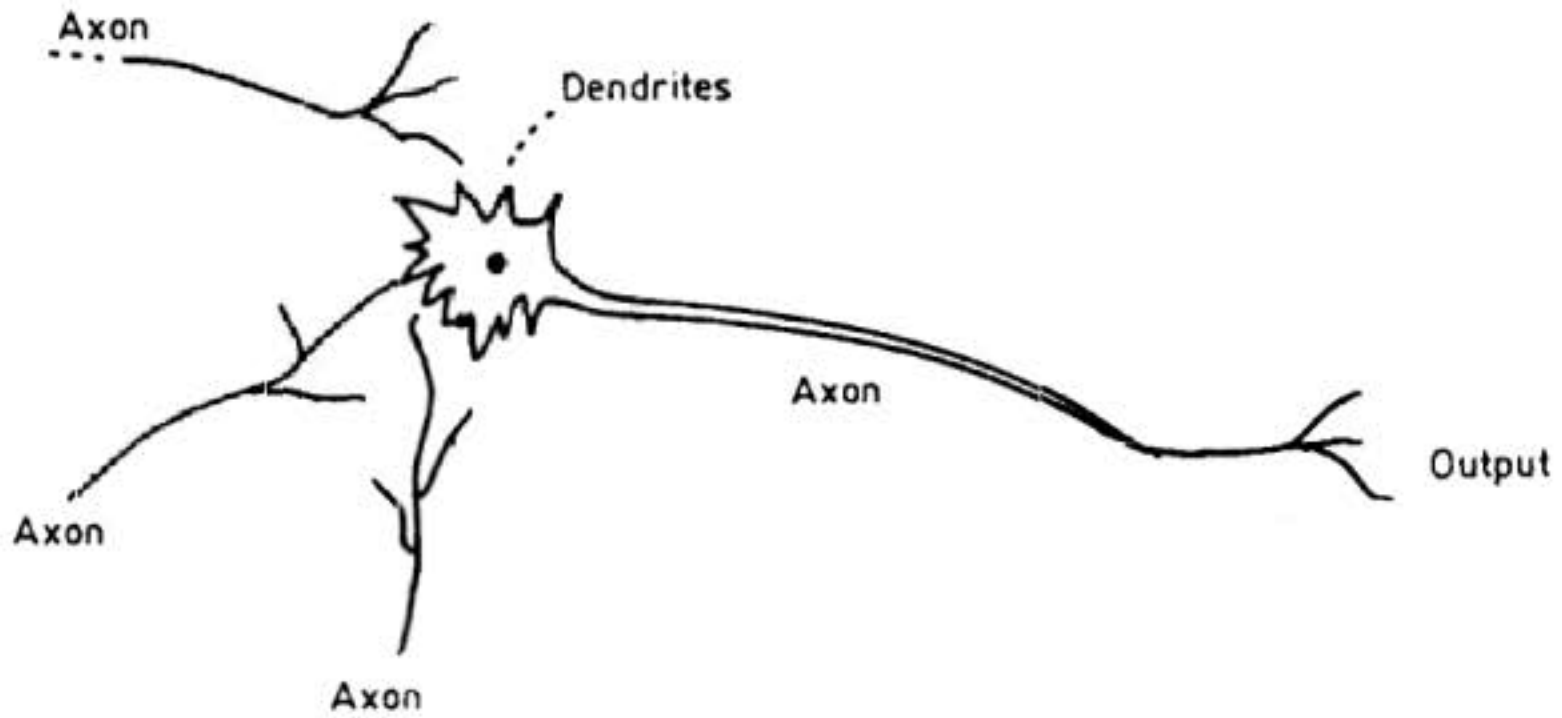


Hình 7.7. *Cấu trúc nơ-ron sinh học: cell body, axon, synaptic*

Mạng ANN đầu tiên được xây dựng đó là Perceptron (Frank Rosenblatt - 1958), tiếp theo đó là Artron, Adaline, Madaline. Bốn dạng cơ bản này, đặc biệt là Perceptron, là cơ sở để phát triển các mạng ANN. Trong đó, ba loại mạng cơ bản được phát triển tiếp theo là mạng lan truyền ngược (Back-propagation), Hopfield Network, Counter-Propagation Network.

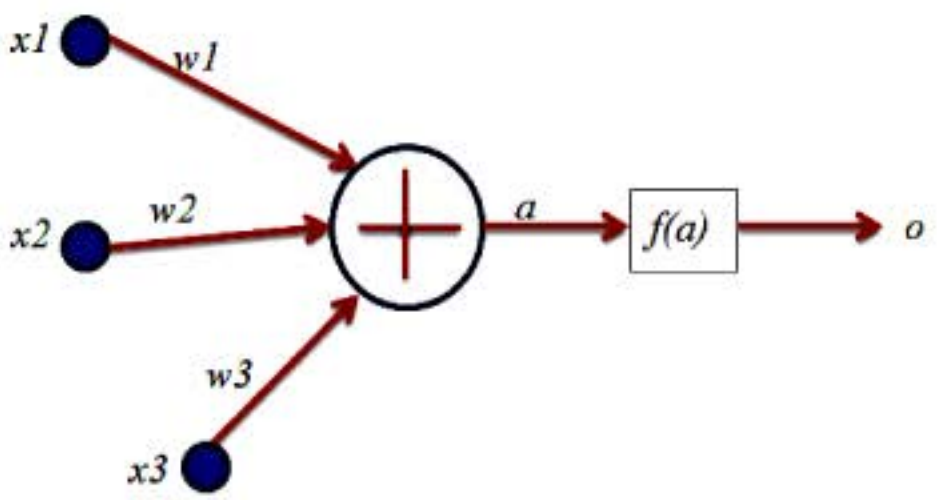
https://tieulun.hopto.org

Perceptron có cấu trúc cơ bản của một tế bào thần kinh được trình bày như trong hình 7.8, như ví dụ này thì có ba ngõ vào trọng số được tổng hợp lại và kết nối đến ngõ ra, ngõ ra này lại có thể là ngõ vào của các nơ-ron khác.



Hình 7.8. *Cấu trúc cơ bản của tế bào thần kinh*

Perceptron này được mô hình hóa theo dạng

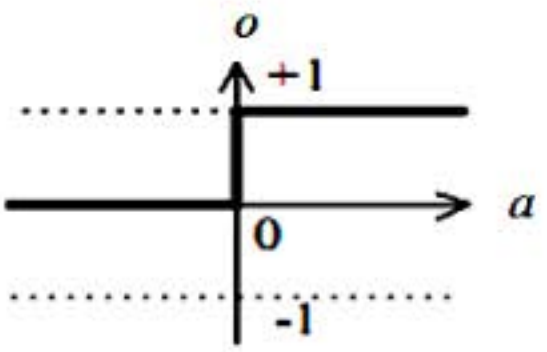


Hình 7.9. *Mô hình perceptron một nơ-ron*

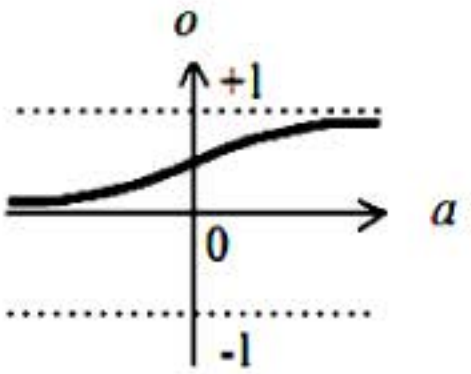
Trong đó xi, i =1, 2, 3 là các ngõ vào, wi là các trọng số với ngõ vào tương ứng, hàm hoạt hóa và ngõ ra $o = f(a)$, trong đó $a = x1.w1 + x2.w2 + x3.w3$.

Các hàm hoạt hóa thường gặp được tóm tắt trong hình 7.10, 7.11, 7.12:

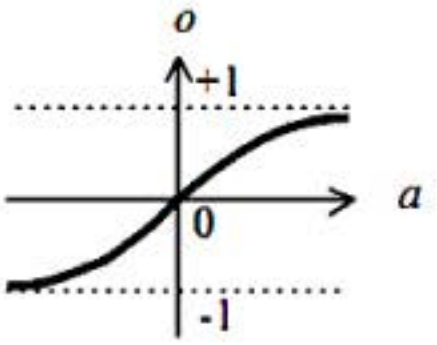
https://tieulun.hopto.org



Hình 7.10. *Hàm bước*



Hình 7.11. *Hàm sigmoid*



Hình 7.12. *Hàm double sigmoid*

Trong hình 7.10, hàm hoạt hóa là hàm bước

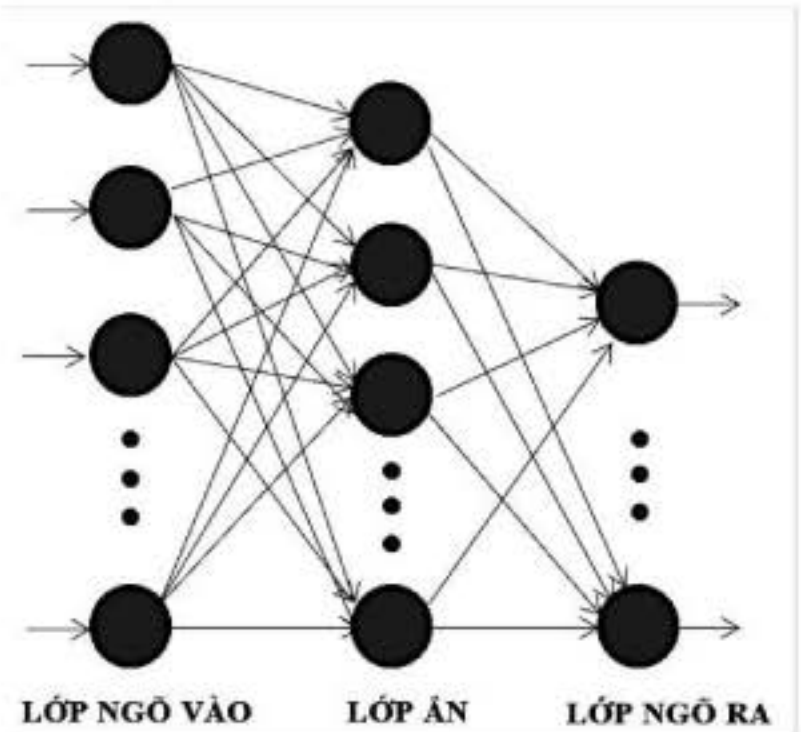
$$o = \begin{cases} 1, & a > 0 \\ 0, & a \le 0 \end{cases} \tag{7.36}$$

Hàm sigmoid và double sigmoid được miêu tả trong hình 7.12 và 7.13 tương ứng với công thức (7.37) và (7.38)

$$o = S(a) = \frac{1}{1+e^{-a}} \tag{7.37}$$

$$o = H(a) = 2S(a) - 1 \tag{7.38}$$

Mạng lan truyền ngược được xây dựng dựa trên cấu trúc Perceptron, trong đó có ít nhất hai lớp, và các hàm hoạt hóa phải là hàm có thể lấy đạo hàm (ví dụ ở đây là hàm sigmoid, double sigmoid). Hình 7.13 là mô hình mạng lan truyền ngược ba lớp (một lớp vào, một lớp ẩn, một lớp ra).



Hình 7.13. *Mạng lan truyền ngược với ba lớp*

Dựa trên giải thuật giảm gradient, mạng lan truyền ngược tìm các trọng số sao cho trung bình bình phương sai số giữa ngõ ra hiện tại và ngõ ra mong muốn đạt kết quả tốt nhất.

https://tieulun.hopto.org

$$E = \sum_{p=1}^{P} \left(o_p - d_p\right)^2 \qquad (7.39)$$

Trong đó: P là số lượng mẫu, o_p là ngõ ra của mạng, d_p là ngõ ra mong muốn.

Giả sử có I nút ở lớp ngõ vào, J nút ở lớp ẩn, K nút ở lớp ngõ ra. Trọng số $w_{j,i}^{(1,0)}$ là trọng số từ nút thứ i của lớp ngõ vào đến nút thứ j của lớp ngõ ẩn, và $w_{k,j}^{(2,1)}$ là trọng số từ nút thứ j của lớp ẩn đến nút thứ k lớp ngõ ra.

Giải thuật huấn luyện cho mạng ba lớp được tóm tắt trong bảng 7.1 như sau

Bảng 7.1. *Giải thuật cập nhật trọng số cho mạng lan truyền ngược ba lớp*

Chọn ngẫu nhiên giá trị các trọng số

Trong khi điều kiện MSE thỏa mãn hoặc chưa vượt số vòng lặp đặt trước,

Với mỗi ngõ vào x_p, $1 \le p \le P$, (*)

Tính các ngõ vào nút lớp ẩn $net_{p,j}^{(1)}$;

Tính các ngõ ra nút lớp ẩn $x_{p,j}^{(1)}$;

Tính các ngõ vào của nút lớp ngõ ra $net_{p,k}^{(2)}$;

Tính các ngõ ra của mạng $o_{p,k}$;

Điều chỉnh trọng số lớp ngõ ra

$$\Delta w_{k,j}^{(2,1)} = \eta (d_{p,k} - o_{p,k}) S'(net_{p,k}^{(2)}) x_{p,j}^{(1)}$$

Điều chỉnh các trọng số giữa nút ẩn và nút ngõ vào

$$\Delta w_{j,i}^{(1,0)} = \eta \sum_{k=1}^{K} \left((d_{p,k} - o_{p,k}) S'(net_{p,k}^{(2)}) w_{k,j}^{(2,1)} \right) S'(net_{p,j}^{(1)}) x_{p,i}$$

Kết thúc (*)

Kết thúc

Trong đó: $S()$ là hàm hoạt hóa, η là hệ số học hay tốc độ học.

https://tieulun.hopto.org

Trong MATLAB, ta có thể sử dụng hàm `newff` để khai báo một mạng truyền thẳng

```
net = newff(P, T, S);
```

trong đó `P`: ma trận các phần tử ngõ vào, `T`: ma trận các phần tử ngõ ra, `S`: kích thước các lớp ẩn.

Tiếp theo, mạng được huấn luyện dùng hàm `train`

```
[neto] = train(net,P,T);
```

Sau khi đã có mạng `neto`, dùng hàm `sim` để phân loại

```
res = sim(neto,dtest);
```

trong đó `dtest` là ngõ vào cần phân loại, `res` là kết quả phân loại.

BÀI TẬP CHƯƠNG 7

7.1. Thu nhận ảnh của 3 người. Thực hiện phân tích thành phần chính PCA để tìm vector đặc trưng.

7.2. Sử dụng mạng nơ-ron 3 lớp (lớp ngõ vào, lớp ẩn, lớp ngõ ra) thực hiện huấn luyện và nhận dạng từ kết quả bài 7.1.

7.3. Trình bày hướng phát triển để thực hiện phân loại SVM trong trường hợp nhiều hơn hai đặc trưng cần phân loại.

https://tieulun.hopto.org

TÀI LIỆU THAM KHẢO

1. Nguyễn Quang Hoan, *Xử lý ảnh,* lưu hành nội bộ, Học viện Công nghệ Bưu chính Viễn thông, 2006.
2. Đỗ Năng Tòan, Phạm Việt Bình, *Xử lý ảnh,* Giáo trình môn học, Đại học Thái Nguyên, Khoa Công nghệ thông tin, 2007.
3. Maria Petrou, Panagiota Bosdogianni, *Image Processing: The Fundamentals*, John Wiley & Sons Ltd, 1999.
4. William K. Pratt, *Digital Image Processing,* John Wiley & Sons, Inc., 2001.
5. Rafael C. Gonzalez, Richard E. Woods, *Digital Image Processing,* The Third Edition,Prentice Hall, 2008.
6. Rafael C. Gonzalez, Richard E. Woods, *Digital Image Processing Using Matlab,* Prentice Hall, 2004.
7. Kayvan N., Robert S., *Biomedical Signal and Image Processing*, Taylor and Francis Group, 2006.

https://tieulun.hopto.org

GIÁO TRÌNH
XỬ LÝ ẢNH
TS. NGUYỄN THANH HẢI

NHÀ XUẤT BẢN
ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
Khu Phố 6, Phường Linh Trung, Quận Thủ Đức, TPHCM
Số 3, Công trường Quốc tế, Quận 3, TP Hồ Chí Minh
ĐT: 38239171 – 38225227 - 38239172
Fax: 38239172 - Email: vnuhp@vnuhcm.edu.vn

*****

PHÒNG PHÁT HÀNH NHÀ XUẤT BẢN
ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
Số 3 Công trường Quốc tế - Quận 3 – TPHCM
ĐT: 38239170 – 0982920509 – 0913943466
Fax: 38239172 – Website: www.nxbdhqghcm.edu.vn

*****

Chịu trách nhiệm xuất bản:
NGUYỄN HOÀNG DŨNG

Chịu trách nhiệm nội dung:
NGUYỄN HOÀNG DŨNG

Tổ chức bản thảo và chịu trách nhiệm về tác quyền
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TPHCM

Biên tập:
PHẠM ANH TÚ

Sửa bản in:
THÙY DƯƠNG

Trình bày bìa
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TPHCM

Mã số ISBN: 978-604-73-2582-5

Số lượng 300 cuốn; khổ 16 x 24cm.
Số đăng ký kế hoạch xuất bản: 1007-2014/CXB/05-13/ĐHQGTPHCM.
Quyết định xuất bản số: 107 ngày 28/05/2014 của NXB ĐHQGTPHCM.
In tại Công ty TNHH In và Bao bì Hưng Phú.
Nộp lưu chiểu quý III năm 2014.

https://tieulun.hopto.org

ISBN: 978-604-73-2582-5

https://tieulun.hopto.org