**Chương 1**

# 1.1/1.2

**% Đọc ảnh**

**I = imread('cell.tif');**

**% Hiển thị ảnh**

**figure;**

**imshow(I);**

**title('Ảnh gốc');**

**% Truy xuất giá trị tại vị trí pixel (100, 20)**

**i = 100;**

**j = 20;**

**original\_value = I(i, j); % Lấy giá trị pixel ban đầu**

**% Thực hiện phép toán: cộng và trừ 25**

**I\_add = I; % Tạo bản sao để thực hiện phép cộng**

**I\_sub = I; % Tạo bản sao để thực hiện phép trừ**

**I\_add(i, j) = I\_add(i, j) + 25; % Cộng 25 vào pixel (100, 20)**

**I\_sub(i, j) = I\_sub(i, j) - 25; % Trừ 25 từ pixel (100, 20)**

**% Đảm bảo giá trị pixel nằm trong khoảng [0, 255] cho ảnh 8-bit**

**I\_add = uint8(max(min(I\_add, 255), 0));**

**I\_sub = uint8(max(min(I\_sub, 255), 0));**

**% Hiển thị ảnh sau khi thay đổi pixel**

**figure;**

**subplot(1, 3, 1); imshow(I); title('Ảnh gốc');**

**subplot(1, 3, 2); imshow(I\_add); title('Cộng 25 tại (100, 20)');**

**subplot(1, 3, 3); imshow(I\_sub); title('Trừ 25 tại (100, 20)');**

**% Khởi tạo hai ảnh mới**

**[rows, cols] = size(I); % Lấy kích thước ảnh**

**I1 = zeros(rows, cols, 'uint8'); % Ảnh I1: Cộng 25**

**I2 = zeros(rows, cols, 'uint8'); % Ảnh I2: Trừ 25**

**% Duyệt qua từng pixel của ảnh và thực hiện phép tính**

**for i = 1:rows**

**for j = 1:cols**

**% Cộng 25 và đảm bảo giá trị không vượt quá 255**

**I1(i, j) = min(I(i, j) + 50, 255);**

**% Trừ 25 và đảm bảo giá trị không nhỏ hơn 0**

**I2(i, j) = max(I(i, j) - 50, 0);**

**end**

**end**

**% Hiển thị kết quả**

**figure;**

**subplot(1, 3, 1); imshow(I); title('Ảnh gốc');**

**subplot(1, 3, 2); imshow(I1); title('Ảnh sau khi cộng 25');**

**subplot(1, 3, 3); imshow(I2); title('Ảnh sau khi trừ 25');**

**Chương 1**

# 1.3

**% Đọc ảnh gốc "cameraman.tif"**

**I = imread('cameraman.tif');**

**% Ghi ảnh dưới định dạng JPEG và PNG**

**imwrite(I, 'Ijpg.jpg', 'jpg'); % Lưu ảnh JPEG**

**imwrite(I, 'Ipng.png', 'png'); % Lưu ảnh PNG**

**% Đọc lại hai ảnh đã lưu**

**Ijpg = imread('Ijpg.jpg'); % Đọc ảnh JPEG**

**Ipng = imread('Ipng.png'); % Đọc ảnh PNG**

**% So sánh hai ảnh với ảnh gốc**

**diff\_jpg = I - Ijpg; % Chênh lệch với ảnh JPEG**

**diff\_png = I - Ipng; % Chênh lệch với ảnh PNG**

**% Hiển thị kết quả**

**figure;**

**subplot(2, 3, 1); imshow(I); title('Ảnh gốc (cameraman.tif)');**

**subplot(2, 3, 2); imshow(Ijpg); title('Ảnh JPEG (Ijpg.jpg)');**

**subplot(2, 3, 3); imshow(Ipng); title('Ảnh PNG (Ipng.png)');**

**subplot(2, 3, 4); imshow(I); title('Ảnh gốc (cameraman.tif)');**

**subplot(2, 3, 5); imshow(diff\_jpg, []); title('Chênh lệch với JPEG');**

**subplot(2, 3, 6); imshow(diff\_png, []); title('Chênh lệch với PNG');**

**Phân tích vấn đề và lý do không giống hoàn toàn:**

Nén mất thông tin (Lossy Compression - JPEG): Định dạng JPEG sử dụng phương pháp nén mất thông tin, có nghĩa là khi ảnh được lưu dưới định dạng JPEG, một số dữ liệu ảnh sẽ bị loại bỏ để giảm kích thước file. Các chi tiết không quá quan trọng hoặc các pixel có sự thay đổi nhỏ có thể bị làm mờ hoặc mất đi hoàn toàn. Vì vậy, khi bạn so sánh ảnh gốc và ảnh JPEG, sẽ có sự khác biệt về pixel, dẫn đến chênh lệch không bằng 0.

Nén không mất thông tin (Lossless Compression - PNG): Định dạng PNG sử dụng phương pháp nén không mất thông tin, có nghĩa là ảnh PNG giữ nguyên tất cả dữ liệu của ảnh gốc mà không bị mất thông tin. Tuy nhiên, do các phương pháp nén khác nhau, mặc dù PNG không làm mất dữ liệu, ảnh PNG có thể có sự thay đổi nhỏ về cách lưu trữ và mã hóa ảnh, nhưng sự khác biệt này thường ít hơn so với JPEG.

**Giải thích về việc so sánh ảnh:**

Việc trừ từng pixel của hai ảnh để so sánh trực tiếp là hợp lý trong trường hợp bạn muốn kiểm tra sự khác biệt pixel-by-pixel. Tuy nhiên, vì JPEG có nén mất thông tin, sự khác biệt giữa ảnh gốc và ảnh JPEG sẽ là một số giá trị không bằng 0, do sự thay đổi trong quá trình nén.

Ảnh PNG có thể có sự khác biệt nhỏ khi so sánh, nhưng khác biệt này thường ít hơn so với JPEG vì PNG là định dạng nén không mất thông tin. Tuy nhiên, điều này không có nghĩa là ảnh PNG và ảnh gốc là hoàn toàn giống nhau, mà là sự khác biệt này rất nhỏ và không dễ nhận thấy bằng mắt thường.

**Kết luận:**

Chúng ta không mong đợi các ảnh giống nhau hoàn toàn khi so sánh ảnh gốc với ảnh JPEG và PNG.

Sự khác biệt không phải là lỗi mà là do định dạng nén (lossy vs. lossless) tạo ra sự thay đổi trong cách dữ liệu ảnh được lưu trữ.

Đoạn mã bạn cung cấp sẽ giúp bạn thấy rõ ràng sự khác biệt giữa các định dạng nén, đặc biệt là giữa JPEG (nén mất thông tin) và PNG (nén không mất thông tin).

**Chương 2**

# 2.1/abcd

**% Đọc ảnh vào biến r**

**r = imread('Fig0304(a)(breast\_digital\_Xray).tif');**

**% Kiểm tra nếu ảnh không phải là ảnh xám, chuyển đổi nó**

**if size(r, 3) == 3**

**r = rgb2gray(r); % Chuyển đổi ảnh màu sang ảnh xám**

**end**

**% Cấp mức xám của ảnh (L = 256 cho ảnh 8-bit)**

**L = 256;**

**% Thực hiện biến đổi âm bản: sa = L - 1 - r**

**sa = L - 1 - r;**

**% Tính histogram của ảnh gốc và ảnh âm bản**

**[counts\_r, grayLevels\_r] = imhist(r); % Histogram ảnh gốc**

**[counts\_sa, grayLevels\_sa] = imhist(sa); % Histogram ảnh âm bản**

**% Hiển thị histogram của ảnh gốc và ảnh âm bản**

**figure;**

**subplot(1, 2, 1);**

**bar(grayLevels\_r, counts\_r);**

**xlabel('Mức xám');**

**ylabel('Số lượng pixel');**

**title('Histogram ảnh gốc');**

**subplot(1, 2, 2);**

**bar(grayLevels\_sa, counts\_sa);**

**xlabel('Mức xám');**

**ylabel('Số lượng pixel');**

**title('Histogram ảnh âm bản');**

**% Chọn ngưỡng t = 127**

**t = 127;**

**% Tạo ảnh nhị phân từ ảnh xám**

**B = r >= t; % Mỗi pixel >= ngưỡng t sẽ thành 1 (trắng), còn lại thành 0 (đen)**

**% Chuyển ảnh nhị phân về kiểu uint8 để hiển thị**

**B = uint8(B) \* 255; % 1 trở thành 255, 0 giữ nguyên**

**% Hiển thị ảnh gốc và ảnh nhị phân trên một cửa sổ**

**figure;**

**subplot(2, 2, 1);**

**imshow(r);**

**title('Ảnh xám gốc');**

**subplot(2, 2, 2);**

**imshow(sa);**

**title('Ảnh âm bản');**

**subplot(2, 2, 3);**

**imshow(B);**

**title('Ảnh nhị phân với ngưỡng t = 127');**

**Phân tích sự phân bố mức xám của ảnh:**

**a. Phân tích sự phân bố mức xám của ảnh:**

**Histogram của ảnh gốc:**

Sự phân bố mức xám của ảnh gốc được thể hiện qua histogram đầu tiên. Histogram này cho biết số lượng pixel ở mỗi mức xám từ 0 đến 255. Nếu histogram có một đỉnh cao ở mức xám thấp (0-50) và đỉnh thấp ở các mức xám cao, điều này cho thấy ảnh chủ yếu có các pixel tối, hoặc ngược lại nếu đỉnh nằm ở mức xám cao.

**Histogram của ảnh âm bản:**

Histogram của ảnh âm bản có thể khác biệt rõ rệt so với ảnh gốc. Do ảnh âm bản chuyển các pixel sáng thành tối và ngược lại, histogram của ảnh âm bản sẽ có phân bố ngược lại so với histogram của ảnh gốc.

**Đánh giá phân tích:**

Nếu histogram của ảnh gốc có một đỉnh rõ ràng ở các mức xám trung bình hoặc tối, điều này có thể chỉ ra rằng ảnh có các vùng tối chiếm ưu thế.

Nếu histogram của ảnh gốc là phân bố đều, điều đó có thể chỉ ra rằng ảnh có sự phân bố mức xám khá đều và không có vùng tối hoặc sáng chiếm ưu thế.

Sự phân bố mức xám của ảnh gốc có thể phản ánh độ sáng tổng thể của ảnh, trong khi sự phân bố của ảnh âm bản phản ánh sự tương phản ngược lại.

**Ảnh đã phân tích tốt chưa?**

Ảnh đã được phân tích tốt vì bạn đã tính toán và hiển thị histogram của ảnh gốc và ảnh âm bản, giúp bạn thấy được sự phân bố mức xám của ảnh.

**Đánh giá sự phân bố:**

Cần kiểm tra xem histogram của ảnh gốc có sự phân bố rõ ràng không, liệu ảnh có độ sáng hoặc tối chiếm ưu thế hay không.

Nếu histogram của ảnh gốc là đều, tức là các mức xám phân bố đều từ 0 đến 255, thì phân tích có thể xem là tốt, thể hiện ảnh có sự phân bổ mức xám đồng đều.

**Tóm lại:**

Mã của bạn thực hiện đúng yêu cầu và giúp phân tích sự phân bố mức xám của ảnh. Việc phân tích mức xám qua histogram là một phương pháp chính xác để hiểu về độ sáng, độ tương phản và đặc điểm của ảnh.

**Chương 2**

# 2.1/abcd Không dùng hàm có sẵn

**% Đọc ảnh vào biến r**

**r = imread('Fig0304(a)(breast\_digital\_Xray).tif');**

**%Câu 2.1ab=============================================**

**% Cấp mức xám của ảnh (L = 256 cho ảnh 8-bit)**

**L = 256;**

**% Thực hiện biến đổi âm bản: sa = L - 1 - r**

**sa = L - 1 - r;**

**% Tính histogram của ảnh gốc và ảnh âm bản**

**[counts\_r, grayLevels\_r] = imhist(r); % Histogram ảnh gốc**

**[counts\_sa, grayLevels\_sa] = imhist(sa); % Histogram ảnh âm bản**

**% Hiển thị histogram của ảnh gốc và ảnh âm bản**

**figure;**

**subplot(1, 2, 1);**

**bar(grayLevels\_r, counts\_r);**

**xlabel('Mức xám');**

**ylabel('Số lượng pixel');**

**title('Histogram ảnh gốc');**

**subplot(1, 2, 2);**

**bar(grayLevels\_sa, counts\_sa);**

**xlabel('Mức xám');**

**ylabel('Số lượng pixel');**

**title('Histogram ảnh âm bản');**

**%Câu 2.1cd==============================================**

**% Chọn ngưỡng t = 127**

**t = 127;**

**% Tạo ảnh nhị phân từ ảnh xám**

**B = r >= t; % Mỗi pixel >= ngưỡng t sẽ thành 1 (trắng), còn lại thành 0 (đen)**

**% Chuyển ảnh nhị phân về kiểu uint8 để hiển thị**

**% Duyệt từng pixel để tạo ảnh nhị phân**

**for i = 1:size(r, 1) % Duyệt qua từng hàng**

**for j = 1:size(r, 2) % Duyệt qua từng cột**

**if r(i, j) >= t**

**B(i, j) = 255; % Pixel >= ngưỡng t chuyển thành 255 (trắng)**

**else**

**B(i, j) = 0; % Pixel < ngưỡng t chuyển thành 0 (đen)**

**end**

**end**

**end**

**% Hiển thị ảnh gốc và ảnh nhị phân trên một cửa sổ**

**figure;**

**subplot(2, 2, 1);**

**imshow(r);**

**title('Ảnh xám gốc');**

**subplot(2, 2, 2);**

**imshow(sa);**

**title('Ảnh âm bản');**

**subplot(2, 2, 3);**

**imshow(B);**

**title('Ảnh nhị phân với ngưỡng t = 127');**

**Chương 2**

# 2.2/abc

**% Đọc ảnh vào biến I**

**I = imread('Fig0122(a)(fractal-iris).tif');**

**%2.2a=================================================**

**% Tạo ảnh i3 từ bit plane thứ 3 của ảnh I**

**bit\_plane\_3 = 3;**

**I3 = bitget(I, bit\_plane\_3);**

**%2.2b=================================================**

**% Tạo ảnh i6 từ bit plane thứ 6 của ảnh I**

**bit\_plane\_6 = 6;**

**I6 = bitget(I, bit\_plane\_6);**

**% Truy xuất bit plane thứ 7 và thứ 8**

**bit\_plane\_7 = bitget(I, 7); % Lấy bit thứ 7**

**bit\_plane\_8 = bitget(I, 8); % Lấy bit thứ 8**

**% Tạo ảnh i78 bằng cách kết hợp bit thứ 7 và 8**

**i78 = bitset(zeros(size(I)), 7, bit\_plane\_7); % Đặt bit thứ 7 của ảnh mới**

**i78 = bitset(i78, 8, bit\_plane\_8); % Đặt bit thứ 8 của ảnh mới**

**% Hiển thị ảnh gốc và ảnh từ bit plane thứ 3**

**figure;**

**subplot(2, 2, 1); imshow(I);**

**title('Ảnh gốc');**

**subplot(2, 2, 2); imshow(I3, []); % Hiển thị ảnh i3 (bit plane thứ 3)**

**title('Bit plane thứ 3');**

**% Hiển thị ảnh từ bit plane thứ 6**

**subplot(2, 2, 3); imshow(I6, []); % Hiển thị ảnh i6 (bit plane thứ 6)**

**title('Bit plane thứ 6');**

**% Hiển thị ảnh từ bit plane thứ 7 và 8**

**subplot(2, 2, 4); imshow(i78, []); % Hiển thị ảnh i78 (bit plane thứ 7 và 8)**

**title('Ảnh i78 (Bit plane thứ 7 và 8)');**

**Chương 2**

# 2.2/abc Không dùng hàm có sẵn

**% Đọc ảnh vào biến I**

**I = imread('Fig0122(a)(fractal-iris).tif');**

**% Kích thước ảnh**

**[rows, cols] = size(I);**

**% 2.2a: Tạo ảnh i3 từ bit plane thứ 3 của ảnh I**

**bit\_plane\_3 = 3; % Bit thứ 3**

**I3 = zeros(rows, cols); % Khởi tạo ma trận ảnh I3**

**for i = 1:rows**

**for j = 1:cols**

**% Lấy giá trị bit thứ 3 của mỗi pixel**

**I3(i, j) = bitand(I(i, j), 2^(bit\_plane\_3 - 1)) > 0;**

**end**

**end**

**% 2.2b: Tạo ảnh i6 từ bit plane thứ 6 của ảnh I**

**bit\_plane\_6 = 6; % Bit thứ 6**

**I6 = zeros(rows, cols); % Khởi tạo ma trận ảnh I6**

**for i = 1:rows**

**for j = 1:cols**

**% Lấy giá trị bit thứ 6 của mỗi pixel**

**I6(i, j) = bitand(I(i, j), 2^(bit\_plane\_6 - 1)) > 0;**

**end**

**end**

**% Truy xuất bit plane thứ 7 và thứ 8**

**bit\_plane\_7 = 7;**

**bit\_plane\_8 = 8;**

**bit7 = zeros(rows, cols); % Ảnh chứa bit thứ 7**

**bit8 = zeros(rows, cols); % Ảnh chứa bit thứ 8**

**for i = 1:rows**

**for j = 1:cols**

**% Lấy giá trị bit thứ 7 và 8**

**bit7(i, j) = bitand(I(i, j), 2^(bit\_plane\_7 - 1)) > 0;**

**bit8(i, j) = bitand(I(i, j), 2^(bit\_plane\_8 - 1)) > 0;**

**end**

**end**

**% Tạo ảnh i78 bằng cách kết hợp bit thứ 7 và 8**

**i78 = zeros(rows, cols, 'uint8'); % Khởi tạo ảnh i78**

**for i = 1:rows**

**for j = 1:cols**

**% Đặt bit thứ 7 và thứ 8**

**i78(i, j) = bit7(i, j) \* 2^(bit\_plane\_7 - 1) + bit8(i, j) \* 2^(bit\_plane\_8 - 1);**

**end**

**end**

**% Hiển thị ảnh**

**figure;**

**subplot(2, 2, 1); imshow(I);**

**title('Ảnh gốc');**

**subplot(2, 2, 2); imshow(I3, []);**

**title('Bit plane thứ 3');**

**subplot(2, 2, 3); imshow(I6, []);**

**title('Bit plane thứ 6');**

**subplot(2, 2, 4); imshow(i78, []);**

**title('Ảnh i78 (Bit plane thứ 7 và 8)');**

**Chương 2**

# 2.3/abc

**% Đọc ảnh**

**I = imread('Fig0316(4)(bottom\_left).tif');**

**% Hiển thị ảnh gốc**

**figure;**

**imshow(I);**

**title('Ảnh gốc');**

**% Tính toán histogram (lược đồ xám)**

**[counts, grayLevels] = imhist(I);**

**% Hiển thị histogram**

**figure;**

**bar(grayLevels, counts);**

**xlabel('Mức xám');**

**ylabel('Số lượng pixel');**

**title('Lược đồ xám của ảnh gốc');**

**% Cân bằng lược đồ xám của ảnh**

**I\_eq = histeq(I);**

**% Hiển thị ảnh gốc và ảnh đã cân bằng**

**figure;**

**subplot(1, 2, 1);**

**imshow(I);**

**title('Ảnh gốc');**

**subplot(1, 2, 2);**

**imshow(I\_eq);**

**title('Ảnh sau khi cân bằng mức xám');**

**% Hiển thị histogram của ảnh gốc và ảnh đã cân bằng**

**figure;**

**subplot(1, 2, 1);**

**imhist(I);**

**title('Histogram của ảnh gốc');**

**subplot(1, 2, 2);**

**imhist(I\_eq);**

**title('Histogram của ảnh sau khi cân bằng');**

**Chương 2**

# 2.3/abc Không dùng hàm có sẵn

**% Đọc ảnh xám**

**i = imread('Fig0316(4)(bottom\_left).tif');**

**% Kiểm tra nếu ảnh là RGB, chuyển sang ảnh xám**

**if size(i, 3) == 3**

**i = rgb2gray(i);**

**end**

**% Kích thước ảnh**

**[N, M] = size(i); %N số dòng, M số cột**

**L = 256; % Số mức xám**

**% Bước 1: Tính histogram gốc**

**hist\_orig = zeros(1, L); %Số lần xuất hiện của 1 mức xám cụ thể**

**for x = 1:N**

**for y = 1:M**

**hist\_orig(i(x, y) + 1) = hist\_orig(i(x, y) + 1) + 1;**

**end**

**end**

**% Bước 2: Tính hàm tích lũy xác suất (CDF)**

**cdf = zeros(1, L); %Tổng mức xám tích lũy từ 0 đến k**

**cdf(1) = hist\_orig(1);**

**for k = 2:L**

**cdf(k) = cdf(k-1) + hist\_orig(k);**

**end**

**cdf = cdf / (N \* M); % Chuẩn hóa CDF, sau khi chuẩn hóa ta có tỷ lệ xuất hiện của 1 mức xám cụ thể**

**% Bước 3: Ánh xạ mức xám mới**

**s = round(cdf \* (L - 1)); % Ánh xạ CDF vào khoảng 0–255, Hàm round được sử dụng để làm tròn kết quả về số nguyên**

**% Bước 4: Tạo ảnh sau cân bằng**

**i\_equalized = zeros(N, M, 'uint8');**

**for x = 1:N**

**for y = 1:M**

**i\_equalized(x, y) = s(i(x, y) + 1);**

**end**

**end**

**% Tính histogram sau cân bằng**

**hist\_equalized = zeros(1, L);**

**for x = 1:N**

**for y = 1:M**

**hist\_equalized(i\_equalized(x, y) + 1) = hist\_equalized(i\_equalized(x, y) + 1) + 1;**

**end**

**end**

**% Hiển thị kết quả**

**figure;**

**subplot(2, 2, 1); imshow(i); title('Ảnh gốc');**

**subplot(2, 2, 2); bar(0:L-1, hist\_orig); title('Histogram gốc');**

**subplot(2, 2, 3); imshow(i\_equalized); title('Ảnh sau cân bằng');**

**subplot(2, 2, 4); bar(0:L-1, hist\_equalized); title('Histogram sau cân bằng');**

**Chương 2**

# 2.7

**% Đọc các ảnh đầu vào**

**pillsetc = imread('pillsetc.png');**

**tape = imread('tape.png');**

**coins = imread('coins.png');**

**eight = imread('eight.tif');**

**% Chuyển đổi ảnh sang ảnh nhị phân với ngưỡng cố định**

**pillsetc\_1 = im2bw(pillsetc, 0.1); % Tách ngưỡng tự động với ngưỡng 0.1**

**tape\_1 = im2bw(tape, 0.1); % Tách ngưỡng tự động với ngưỡng 0.1**

**coins\_1 = im2bw(coins, 0.1); % Tách ngưỡng tự động với ngưỡng 0.1**

**eight\_1 = im2bw(eight, 0.1); % Tách ngưỡng tự động với ngưỡng 0.1**

**% Hiển thị các ảnh gốc và ảnh đã tách ngưỡng**

**figure;**

**subplot(4, 2, 1), imshow(pillsetc);**

**title('Ảnh gốc: pillsetc');**

**subplot(4, 2, 2), imshow(pillsetc\_1);**

**title('Ảnh nhị phân: pillsetc');**

**subplot(4, 2, 3), imshow(tape);**

**title('Ảnh gốc: tape');**

**subplot(4, 2, 4), imshow(tape\_1);**

**title('Ảnh nhị phân: tape');**

**subplot(4, 2, 5), imshow(coins);**

**title('Ảnh gốc: coins');**

**subplot(4, 2, 6), imshow(coins\_1);**

**title('Ảnh nhị phân: coins');**

**subplot(4, 2, 7), imshow(eight);**

**title('Ảnh gốc: eight');**

**subplot(4, 2, 8), imshow(eight\_1);**

**title('Ảnh nhị phân: eight');**

**Chương 2**

# 2.7 Không dùng hàm có sẵn

**Chương 2**

# 2.8

**% Đọc ảnh f**

**f = imread('Fig0122(a)(fractal-iris).tif');**

**% a. Tạo ảnh h bằng cách đặt 4 plane bit thấp của ảnh f bằng 0**

**h = bitset(f, 1, 0); % Đặt bit 1 bằng 0**

**h = bitset(h, 2, 0); % Đặt bit 2 bằng 0**

**h = bitset(h, 3, 0); % Đặt bit 3 bằng 0**

**h = bitset(h, 4, 0); % Đặt bit 4 bằng 0**

**% b. Tạo ảnh g từ phép trừ f - h sử dụng vòng lặp**

**[M, N] = size(f); % Kích thước ảnh**

**g = zeros(M, N, 'uint8'); % Khởi tạo ảnh g với kích thước giống f và h**

**for x = 1:M**

**for y = 1:N**

**g(x, y) = f(x, y) - h(x, y); % Phép trừ từng pixel**

**end**

**end**

**% c. Tạo ảnh i từ việc cân bằng mức xám của ảnh g**

**i = histeq(g); % Cân bằng histogram của ảnh g**

**% Hiển thị các ảnh**

**figure;**

**subplot(2, 2, 1);**

**imshow(f);**

**title('Ảnh gốc f');**

**subplot(2, 2, 2);**

**imshow(h);**

**title('Ảnh h (4 plane bit thấp = 0)');**

**subplot(2, 2, 3);**

**imshow(g);**

**title('Ảnh g (f - h)');**

**subplot(2, 2, 4);**

**imshow(i);**

**title('Ảnh i (Cân bằng mức xám của g)');**

**Chương 2**

# 2.8 Không dùng hàm có sãn

====================================================================

**Chương 2**

# 2.9

====================================================================

**Chương 2**

# 2.9 Không dùng hàm có sẵn

====================================================================

**Chương 2**

# 2.10

====================================================================

**Chương 2**

# 2.10 Không dùng hàm có sẵn

====================================================================

**Chương 2**

# 2.11

====================================================================

**Chương 2**

# 2.11 Không dùng hàm có sẵn