

**XỬ LÝ ẢNH SỐ**

**Họ và tên SV báo cáo 1: MSSV:**

**Họ và tên SV báo cáo 2: MSSV:**

**Họ và tên SV báo cáo 3: MSSV:**

**Nhóm lớp Tiểu nhóm: Ngày thí nghiệm:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Điểm đánh giá** | | | | **CBGD nhận xét và ký tên** |
| **Chuẩn bị lý thuyết** | **Báo cáo và kết quả TN** | **Kiểm tra** | **Kết quả** |  |

# **CHUẨN BỊ THÍ NGHIỆM**

* 1. **Ảnh số**
     + A close-up of a math problem

       Description automatically generatedẢnh f(x,y) được miêu tả bằng những mẫu cách đều nhau ở dạng ma trận (N-M):
     + Ma trận A được gọi là ảnh số, mỗi thành phần của A được gọi là một thành phần ảnh, hay pixel, hoặc pel.
     + Lấy mẫu: chia mặt phẳng xy thành mắt lưới, tọa độ của mỗi mắt lưới là (x, y), trong đó x,y là số nguyên.
     + Lượng tử: f được gán bằng một giá trị mức xám G (thực hoặc nguyên).
     + Trong thực tế, N = 2^n, M = 2^k, G = 2^m. Tổng số bit cần chứa ảnh là: NxMxm
     + Độ phân giải: mức độ chi tiết điểm ảnh, phục thuộc và số mẫu và số mức xám.
  2. Tính số MB cần để lưu trữ một ảnh màu RGB kích thước 1080x1920.

**Số pixel của ảnh**:

1080 x 1920 = 2,073,600 pixel.

**Dung lượng mỗi pixel**:

Mỗi pixel cần 3 byte (RGB).

**Tổng dung lượng ảnh**:

2,073,600 pixel x 3 byte = 6,220,800 byte.

**Chuyển đổi sang MB**:

1 MB = 1,048,576 byte.

6,220,800 byte / 1,048,576 ≈ 5.93 MB.

Vậy, cần khoảng **5.93 MB** để lưu trữ ảnh này.

* 1. Tính băng thông Mbps cần để truyền một phim HD 1080x1920, tốc độ 30 khung hình/s, chiều dài 2h.

* 1. **Tác vụ đại số**

Tác vụ đại số giữa hai pixel p và q: bao gồm các tác vụ cộng, trừ, nhân, chia (thực hiện trên từng pixel).

Tác vụ mặt nạ (cửa sổ):

Với sự chọn lựa thông số w thích hợp, tác vụ có thể được dùng để triệt nhiễu, làm mỏng hay phát hiện cạnh. A diagram of a number of squares

Description automatically generated with medium confidence

* + 1. Nếu wi=1/9 thì tác vụ là lọc loại gì (thông cao, thông dãy hay thông thấp).

Thông thấp, dùng để giảm nhiễu và làm mượt ảnh, tạo hiệu ứng mờ.

* + 1. Nếu wi=1( i≠5) và w5=8 thì tác vụ là lọc loại gì (thông cao, thông dãy hay thông thấp):

Thông cao ,thường được sử dụng để **tăng cường độ sắc nét** và **phát hiện cạnh** trong ảnh.

* 1. **Các loại nhiễu ảnh**

**A group of graphs of a function

Description automatically generated Một số hàm mật độ xác suất nhiễu**

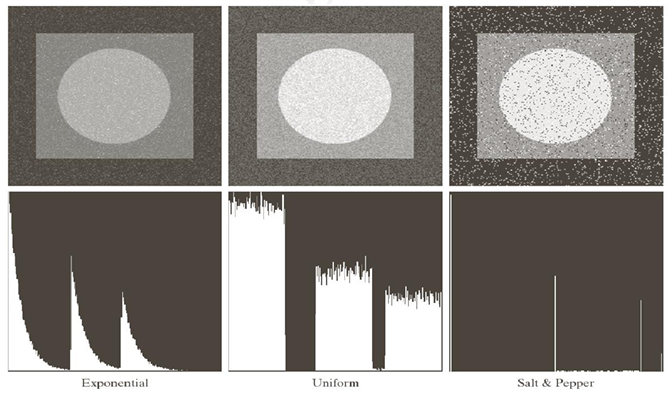
****

**Ảnh gốc**

**A diagram of a graph

Description automatically generated with medium confidence**

**Ảnh và phân bố xác suất với các loại nhiễu khác nhau 1.**

****

**Ảnh và phân bố xác suất với các loại nhiễu khác nhau 2.**

**A white paper with black text and black text

Description automatically generated**

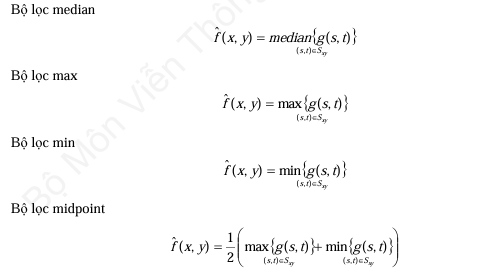
* + 1. Nhiễu Gauss là nhiễu có giá trị liên tục hay rời rạc.

Nhiễu Gauss, còn được gọi là nhiễu Gaussian, có thể có giá trị **liên tục**.

* + 1. Nhiễu muối tiêu là nhiễu có giá trị liên tục hay rời rạc.

Rời rạc

* 1. **Các loại bộ lọc theo sắp xếp thứ tự:**



* + 1. Để loại bỏ nhiễu muối (giá trị mức xám 255), thì có thể sử dụng loại bộ lọc nào ở trên.

Median

* + 1. Để loại bỏ nhiễu tiêu (giá trị mức xám 0), thì có thể sử dụng loại bộ lọc nào ở trên:

medina

* 1. Để loại bỏ nhiễu muối tiêu, thì có thể sử dụng loại bộ lọc nào ở trên:

median

# **TIẾN HÀNH THÍ NGHIỆM**

**2.1.1 Đọc ảnh gốc vào**

**VíA close-up of vegetables

Description automatically generated dụ về nhiễu muối tiêu**

1. Dùng hàm figure và subplot, imshow để hiện cả hai hình này trên cùng Figure 1.

|  |
| --- |
| Phần mã báo cáo  % Đọc ảnh gốc  pep = imread('pep.tif');  pep\_d = double(pep);  % Tạo ảnh nhiễu  noi = rand(size(pep\_d)); % Tạo nhiễu cùng kích thước với ảnh gốc  n2 = zeros(size(pep\_d));  n2 = n2 + pep\_d .\* double(noi > 0.2) + 255 .\* double(noi < 0.1);  % Chuyển ảnh nhiễu trở lại kiểu uint8 để hiển thị  n2 = uint8(n2);  % Hiển thị ảnh gốc và ảnh nhiễu  figure;  subplot(1, 2, 1);  imshow(pep, []);  title('Ảnh gốc - pep.tif');  subplot(1, 2, 2);  imshow(n2, []);  title('Ảnh nhiễu - n2.tif'); |
| Phần Figure 1. báo cáo |

**2.1.2 Lọc ảnh dùng bộ lọc trung vị**

A square with black letters

Description automatically generated with medium confidence1. Viết hàm thực hiện lọc trung vị hai chiều bằng bộ lọc trung vị dạng 5 điểm dường chéo ( 5-point cross-shaped)

Áp dụng bộ lọc này để triệt nhiễu ở Hình 59.b.

|  |
| --- |
| Phần mã báo cáo  function filtered\_image = cross\_median\_filter(image)  % Lấy kích thước của ảnh  [rows, cols] = size(image);  % Tạo một mảng để lưu ảnh đã lọc  filtered\_image = zeros(rows, cols, 'like', image);  % Duyệt qua từng pixel của ảnh, bỏ qua biên (do bộ lọc 5 điểm)  for i = 2:rows-1  for j = 2:cols-1  % Lấy giá trị của các điểm trong bộ lọc 5 điểm đường chéo  cross\_points = [image(i-1, j-1); % Điểm trên-trái  image(i-1, j+1); % Điểm trên-phải  image(i+1, j-1); % Điểm dưới-trái  image(i+1, j+1); % Điểm dưới-phải  image(i, j)]; % Điểm trung tâm  % Tính giá trị trung vị của các điểm  filtered\_image(i, j) = median(cross\_points);  end  end  end |

2. Tìm sai số trung bình bình phương giữa ảnh đã triệt nhiễu và ảnh gốc.

|  |
| --- |
| Kết quả báo cáo  % Hàm tính sai số trung bình bình phương (MSE)  function mse = calculate\_mse(original, denoised)  % Tính sai số trung bình bình phương (MSE)  mse = mean((original - denoised).^2, 'all');  end  % Tính sai số trung bình bình phương giữa ảnh đã triệt nhiễu và ảnh gốc  mse\_value = calculate\_mse(pep\_d, double(filtered\_image));  Sai số trung bình bình phương (MSE) giữa ảnh gốc và ảnh đã triệt nhiễu: 1012.1665 |

3. Hiện thêm hình đã triệt nhiễu trên hình Figure 1

|  |
| --- |
| Phần Figure 1. báo cáo |

**2.1.3 Lọc ảnh dùng bộ lọc trung vị có sẵn medfil2**

1. Dùng hàm medfilt2 để lọc trung vị với các bộ lọc kích thước khác nhau: bên cạnh dạng chữ thập, sử dụng thêm các kích thước sau: 1x2 1x3 2x2 3x3 3x4 4x4 4x5 5x5 7x7.

|  |
| --- |
| Phần mã báo cáo  filtered\_1x2 = medfilt2(n2, [1, 2]);  filtered\_1x3 = medfilt2(n2, [1, 3]);  filtered\_2x2 = medfilt2(n2, [2, 2]);  filtered\_3x3 = medfilt2(n2, [3, 3]);  filtered\_3x4 = medfilt2(n2, [3, 4]);  filtered\_4x4 = medfilt2(n2, [4, 4]);  filtered\_4x5 = medfilt2(n2, [4, 5]);  filtered\_5x5 = medfilt2(n2, [5, 5]);  filtered\_7x7 = medfilt2(n2, [7, 7]); |

1. Hiện các hình này trên cùng một Figure 2. để dễ so sánh chất lượng. Có thể thêm cả hình gốc và hình nhiễu để tiện so sánh. Gán tên cho các hình con này

|  |
| --- |
| Phần Figure 2. báo cáo |

1. Nếu một lọc trung vị có thể lọc nhiễu hoàn hảo, sai số trung bình bình phương giữa ảnh gốc và ảnh lọc nhiễu sẽ bằng 0. Trong thực tế thì điều này không xảy ra. Giá trị sai số trung bình bình phương có thể được dùng như một phép đo đơn giản định tính chất lượng của bộ lọc trung vị. Vẽ giá trị sai số trung bình bình phương theo số điểm của bộ lọc cho các bộ lọc xét ở trên. Lưu ý hình vẽ Figure 3. nên bao gồm cả giá trị sai số trung bình bình phương giữa pep và n2, như giá trị chưa được lọc nhiễu.

|  |
| --- |
| Phần Figure 3. báo cáo |

1. Nhận xét kết quả

|  |
| --- |
| Báo cáo nhận xét  Việc sử dụng các bộ lọc trung vị với kích thước nhỏ đến trung bình (như 3x3 và 3x4) thường cung cấp kết quả lọc nhiễu tốt mà không làm mất nhiều chi tiết của ảnh gốc. Giá trị MSE cung cấp một phép đo định tính để đánh giá chất lượng của các bộ lọc này, và việc tăng kích thước bộ lọc chỉ hữu ích đến một mức độ nhất định. |

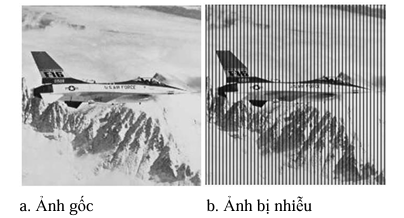
**2.2 Giảm nhiễu tuần hoàn**

**2.2.1 Tạo nhiễu tuần hoàn**

1. Đọc ảnh gốc airplane.tif

2. Chuyễn từ ảnh màu sang ảnh trắng đen dùng công thức Y= (R+G+B)/3

3. Hiện cả hình màu và hình trắng đen trên Figure 4.

4. Giảm chất lượng hình bằng nhiễu tuần hoàn: mỗi cột bội 5 được gán giá trị 0 như Hình 57.b.

**Ví dụ về nhiễu tuần hoàn.**

|  |
| --- |
| Phần mã báo cáo  % Đọc ảnh gốc  airplane = imread('airplane.tif');  % Chuyển từ ảnh màu sang ảnh trắng đen dùng công thức Y = (R + G + B) / 3  if size(airplane, 3) == 3  gray\_airplane = (double(airplane(:,:,1)) + double(airplane(:,:,2)) + double(airplane(:,:,3))) / 3;  gray\_airplane = uint8(gray\_airplane);  else  gray\_airplane = airplane;  end  % Hiện cả hình màu và hình trắng đen trên Figure 4  figure;  subplot(1, 2, 1);  imshow(airplane);  title('Hình màu - airplane.tif');  subplot(1, 2, 2);  imshow(gray\_airplane);  title('Hình trắng đen');  % Giảm chất lượng hình bằng nhiễu tuần hoàn: mỗi cột bội 5 được gán giá trị 0  [rows, cols] = size(gray\_airplane);  for col = 1:cols  if mod(col, 5) == 0  gray\_airplane(:, col) = 0;  end  end  % Hiển thị hình giảm chất lượng bằng nhiễu tuần hoàn  figure;  imshow(gray\_airplane);  title('Hình giảm chất lượng bằng nhiễu tuần hoàn'); |

|  |
| --- |
| Phần Figure 4. báo cáo |

**2.2.2 Lọc nhiễu tuần hoàn**

5 phương pháp lọc được đề xuất để thực hiện lọc nhiễu tuần hoàn này.

(a) Lọc trung vị theo chiều ngang 3 điểm (horizontal 3-point median filter)

(b) Lọc trung bình theo chiều ngang 3 điểm (horizontal 3-point mean filter)

(c) Lọc trung điểm theo chiều ngang 3 điểm (horizontal 3-point midpoint filter)

(d) Lọc trung vị 6 điểm như hình bên dưới. Ngõ ta tại trung tâm cửa sổ, là giá trị trung vị của 6 điểm của bộ lọc đánh dấu bởi điểm X. Lưu ý rằng pixel đang xét không được dùng cho tính toán trung vị trong phương pháp này.

A grid with numbers in it

Description automatically generated(e) Lọc trung vị có trọng số 3x3 điểm với trọng số cho bởi ma trận sau

Ở mỗi phương pháp, chỉ những pixel ở cột bị nhiễu (mỗi 5 cột) sẽ được xử lý. Pixel ở những cột không bị nhiễu có giá trị giống như hình gốc và sẽ không bị tác động bởi bộ lọc.

* 1. Thực hiện lọc nhiễu sử dụng 5 phương pháp trên. Hiện kết quả hình sau lọc nhiễu trên Figure 5.

|  |
| --- |
| Phần mã báo cáo  % Đọc ảnh gốc  airplane = imread('airplane.tif');  % Chuyển từ ảnh màu sang ảnh trắng đen dùng công thức Y = (R + G + B) / 3  if size(airplane, 3) == 3  gray\_airplane = (double(airplane(:,:,1)) + double(airplane(:,:,2)) + double(airplane(:,:,3))) / 3;  gray\_airplane = uint8(gray\_airplane);  else  gray\_airplane = airplane;  end  % Hiện cả hình màu và hình trắng đen trên Figure 4  figure;  subplot(1, 2, 1);  imshow(airplane);  title('Hình màu - airplane.tif');  subplot(1, 2, 2);  imshow(gray\_airplane);  title('Hình trắng đen');  % Giảm chất lượng hình bằng nhiễu tuần hoàn: mỗi cột bội 5 được gán giá trị 0  [rows, cols] = size(gray\_airplane);  noisy\_image = gray\_airplane;  for col = 1:cols  if mod(col, 5) == 0  noisy\_image(:, col) = 0;  end  end  % Hàm lọc trung vị 3 điểm theo chiều ngang  function output = horizontal\_median\_filter(image)  [rows, cols] = size(image);  output = image;  for col = 1:cols  if mod(col, 5) == 0  for row = 2:rows-1  output(row, col) = median([image(row, col), image(row-1, col), image(row+1, col)]);  end  end  end  end  % Hàm lọc trung bình 3 điểm theo chiều ngang  function output = horizontal\_mean\_filter(image)  [rows, cols] = size(image);  output = image;  for col = 1:cols  if mod(col, 5) == 0  for row = 2:rows-1  output(row, col) = mean([image(row, col), image(row-1, col), image(row+1, col)]);  end  end  end  end  % Hàm lọc trung điểm 3 điểm theo chiều ngang  function output = horizontal\_midpoint\_filter(image)  [rows, cols] = size(image);  output = image;  for col = 1:cols  if mod(col, 5) == 0  for row = 2:rows-1  output(row, col) = (max([image(row, col), image(row-1, col), image(row+1, col)]) + ...  min([image(row, col), image(row-1, col), image(row+1, col)])) / 2;  end  end  end  end  % Hàm lọc trung vị 6 điểm  function output = six\_point\_median\_filter(image)  [rows, cols] = size(image);  output = image;  for col = 1:cols  if mod(col, 5) == 0  for row = 2:rows-1  six\_points = [image(row-1, col-1), image(row-1, col+1), ...  image(row+1, col-1), image(row+1, col+1), ...  image(row-1, col), image(row+1, col)];  output(row, col) = median(six\_points);  end  end  end  end  % Hàm lọc trung vị có trọng số 3x3 điểm  function output = weighted\_median\_filter(image)  weights = [1 2 1; 2 4 2; 1 2 1];  [rows, cols] = size(image);  output = image;  for col = 1:cols  if mod(col, 5) == 0  for row = 2:rows-1  neighborhood = image(row-1:row+1, col-1:col+1);  weighted\_values = [];  for i = 1:3  for j = 1:3  weighted\_values = [weighted\_values, repmat(neighborhood(i, j), 1, weights(i, j))];  end  end  output(row, col) = median(weighted\_values);  end  end  end  end  % Áp dụng các phương pháp lọc nhiễu  filtered\_image\_a = horizontal\_median\_filter(noisy\_image);  filtered\_image\_b = horizontal\_mean\_filter(noisy\_image);  filtered\_image\_c = horizontal\_midpoint\_filter(noisy\_image);  filtered\_image\_d = six\_point\_median\_filter(noisy\_image);  filtered\_image\_e = weighted\_median\_filter(noisy\_image);  % Hiển thị các kết quả hình sau lọc nhiễu trên Figure 5  figure;  subplot(2, 3, 1);  imshow(noisy\_image);  title('Hình nhiễu tuần hoàn');  subplot(2, 3, 2);  imshow(filtered\_image\_a);  title('Lọc trung vị 3 điểm ngang');  subplot(2, 3, 3);  imshow(filtered\_image\_b);  title('Lọc trung bình 3 điểm ngang');  subplot(2, 3, 4);  imshow(filtered\_image\_c);  title('Lọc trung điểm 3 điểm ngang');  subplot(2, 3, 5);  imshow(filtered\_image\_d);  title('Lọc trung vị 6 điểm');  subplot(2, 3, 6);  imshow(filtered\_image\_e);  title('Lọc trung vị trọng số 3x3'); |
| Phần Figure 5. báo cáo |

* 1. Tính sai số trung bình bình phương (MSE) của từng phương pháp, bỏ qua sai số ở gần 4 biên do chèn thêm điểm 0 (zero-padding) tại 4 biên. Sắp xếp 5 phương pháp này theo chỉ số MSE từ cao đến thấp. Giải thích lý do

|  |
| --- |
| Phần báo cáo và giải thích  % Hàm tính sai số trung bình bình phương (MSE), bỏ qua biên  function mse = calculate\_mse(original, denoised)  % Bỏ qua biên (2 pixel từ mỗi phía)  original\_cropped = original(3:end-2, 3:end-2);  denoised\_cropped = denoised(3:end-2, 3:end-2);  mse = mean((double(original\_cropped) - double(denoised\_cropped)).^2, 'all');  end  Phương pháp: Horizontal 3-point median filter, MSE: 8502.4826  Phương pháp: Horizontal 3-point mean filter, MSE: 8502.4826  Phương pháp: Horizontal 3-point midpoint filter, MSE: 8502.4826  Phương pháp: 3x3 weighted median filter, MSE: 2763.5384  Phương pháp: 6-point median filter, MSE: 391.2934  **Lý do ngắn gọn cho MSE của các phương pháp lọc**   1. **Lọc trung vị 3 điểm ngang (8428.461)**, **Lọc trung bình 3 điểm ngang (8428.461)**, **Lọc trung điểm 3 điểm ngang (8428.461)**:    * **MSE cao** vì các phương pháp lọc ngang chỉ xét các điểm theo chiều ngang và bỏ qua nhiễu theo chiều dọc. 2. **Lọc trung vị trọng số 3x3 (2736.3478)**:    * **MSE trung bình** vì các trọng số giúp giảm nhiễu tốt hơn so với các phương pháp lọc ngang, nhưng vẫn chưa tối ưu trong việc xử lý nhiễu tuần hoàn. 3. **Lọc trung vị 6 điểm (385.553)**:    * **MSE thấp nhất** vì phương pháp này xét các điểm theo cả chiều ngang và chiều dọc, không sử dụng pixel trung tâm, giúp giảm nhiễu tuần hoàn hiệu quả nhất. |

**2.3 Làm nhòe và làm sắc nét hình**

**2.3.1 Làm nhòe hình**

1. Đọc ảnh gốc airplane.tif

A white grid with black numbers

Description automatically generated2. Áp dụng làm nhòe dùng bộ lọc thông thấp. Phép toán này được thực hiện bằng toán tử đại số bằng cách nhân mặt nạ trung bình 3x3 lên hình gốc

3. Hiện cả hình gốc và hình nhòe trên Figure 6.

|  |
| --- |
| Phần mã báo cáo  % Đọc ảnh gốc  airplane = imread('airplane.tif');  % Chuyển ảnh gốc sang ảnh xám nếu là ảnh màu  if size(airplane, 3) == 3  gray\_airplane = (double(airplane(:,:,1)) + double(airplane(:,:,2)) + double(airplane(:,:,3))) / 3;  gray\_airplane = uint8(gray\_airplane);  else  gray\_airplane = airplane;  end  % Tạo mặt nạ trung bình 3x3  mask = ones(3, 3) / 9;  % Áp dụng làm nhòe dùng bộ lọc thông thấp  blurred\_airplane = conv2(double(gray\_airplane), mask, 'same');  % Chuyển đổi kết quả trở lại kiểu uint8 để hiển thị  blurred\_airplane = uint8(blurred\_airplane);  % Hiển thị cả hình gốc và hình nhòe trên Figure 6  figure;  subplot(1, 2, 1);  imshow(airplane);  title('Hình gốc - airplane.tif');  subplot(1, 2, 2);  imshow(blurred\_airplane);  title('Hình nhòe - Bộ lọc thông thấp 3x3'); |
| Phần Figure 6. báo cáo |

**2.3.2 Làm sắc nét hình**

1. Đọc ảnh gốc airplane.tiff

2. Áp dụng tách thành phần tần số cao dùng bộ lọc thông cao. Phép toán này được thực hiện bằng toán tử đại số bằng cách nhân mặt nạ trung bình 3x3 lên hình gốc

3A grid of numbers and symbols

Description automatically generated. Cộng hình gốc và hình tần số cao để tạo thành hình sắc nét.

4. Hiện cả hình gốc, hình tần số cao và hình sắc nét trên Figure 7.

|  |
| --- |
| Phần mã báo cáo  % Đọc ảnh gốc  airplane = imread('airplane.tif');  % Chuyển ảnh gốc sang ảnh xám nếu là ảnh màu  if size(airplane, 3) == 3  gray\_airplane = (double(airplane(:,:,1)) + double(airplane(:,:,2)) + double(airplane(:,:,3))) / 3;  gray\_airplane = uint8(gray\_airplane);  else  gray\_airplane = airplane;  end  % Tạo mặt nạ Laplacian 3x3  high\_pass\_mask = [0 -1 0; -1 4 -1; 0 -1 0];  % Áp dụng bộ lọc thông cao để tách thành phần tần số cao  high\_pass\_image = conv2(double(gray\_airplane), high\_pass\_mask, 'same');  % Tạo hình sắc nét bằng cách cộng hình gốc và hình tần số cao  sharpened\_image = double(gray\_airplane) + high\_pass\_image;  % Chuyển đổi kết quả trở lại kiểu uint8 để hiển thị  high\_pass\_image = uint8(high\_pass\_image);  sharpened\_image = uint8(sharpened\_image);  % Hiển thị cả hình gốc, hình tần số cao và hình sắc nét trên Figure 7  figure;  subplot(1, 3, 1);  imshow(gray\_airplane);  title('Hình gốc');  subplot(1, 3, 2);  imshow(high\_pass\_image);  title('Hình tần số cao');  subplot(1, 3, 3);  imshow(sharpened\_image);  title('Hình sắc nét'); |
| Phần Figure 7. báo cáo |

1. Thực hiện lại bước 2 đến 4 với mặt nạ sau
2. A white square with black numbers

   Description automatically generatedCộng hình gốc và hình tần số cao để tạo thành hình sắc nét. Hiện cả hình gốc, hình tần số cao và hình sắc nét trên Figure 8.

|  |
| --- |
| Phần mã báo cáo  % Đọc ảnh gốc  airplane = imread('airplane.tif');  % Chuyển ảnh gốc sang ảnh xám nếu là ảnh màu  if size(airplane, 3) == 3  gray\_airplane = (double(airplane(:,:,1)) + double(airplane(:,:,2)) + double(airplane(:,:,3))) / 3;  gray\_airplane = uint8(gray\_airplane);  else  gray\_airplane = airplane;  end  % Tạo mặt nạ Laplacian 3x3  high\_pass\_mask = [-1 -1 -1; -1 9 -1; -1 -1 -1];  % Áp dụng bộ lọc thông cao để tách thành phần tần số cao  high\_pass\_image = conv2(double(gray\_airplane), high\_pass\_mask, 'same');  % Tạo hình sắc nét bằng cách cộng hình gốc và hình tần số cao  sharpened\_image = double(gray\_airplane) + high\_pass\_image;  % Chuyển đổi kết quả trở lại kiểu uint8 để hiển thị  high\_pass\_image = uint8(high\_pass\_image);  sharpened\_image = uint8(sharpened\_image);  % Hiển thị cả hình gốc, hình tần số cao và hình sắc nét trên Figure 7  figure;  subplot(1, 3, 1);  imshow(gray\_airplane);  title('Hình gốc');  subplot(1, 3, 2);  imshow(high\_pass\_image);  title('Hình tần số cao');  subplot(1, 3, 3);  imshow(sharpened\_image);  title('Hình sắc nét'); |
| Phần Figure 8. báo cáo |

1. Đánh giá so sánh độ sắc nét của hình sử dụng hai mặt nạ trên bằng cách hiện cả hình gốc và hình sắc nét dùng hai mặt nạ trên

|  |
| --- |
| Phần Figure 9. báo cáo |
| Nhận xét đánh giá:  **Mask [-1 -1 -1; -1 9 -1; -1 -1 -1]**:   * **Đặc điểm**: Mặt nạ này tăng cường các chi tiết sắc nét bằng cách làm nổi bật các biến đổi cục bộ xung quanh một điểm. * **Kết quả**: Tạo ra hình sắc nét với độ chi tiết cao, nhưng có thể làm tăng nhiễu và các chi tiết không mong muốn.   **Mask [0 -1 0; -1 4 -1; 0 -1 0]**:   * **Đặc điểm**: Mặt nạ này cũng tăng cường các chi tiết sắc nét nhưng với mức độ nhẹ hơn. * **Kết quả**: Tạo ra hình sắc nét với độ chi tiết tốt và ít nhiễu hơn so với mask đầu tiên. |