Bộ lọc LOWPASS

**Nguyên lý hoạt động của bộ lọc Lowpass trong xử lý ảnh**

Bộ lọc Lowpass (lọc thông thấp) là một công cụ trong xử lý tín hiệu, bao gồm xử lý ảnh, nhằm loại bỏ các thành phần tần số cao khỏi tín hiệu, chỉ giữ lại các thành phần tần số thấp.

* **Trong xử lý ảnh:**
  + Ảnh số có thể được coi như một tín hiệu 2D với các tần số không gian.
  + Các tần số thấp thường đại diện cho thông tin tổng quát hoặc vùng mịn (như màu sắc hoặc các vùng sáng tối đồng nhất).
  + Các tần số cao biểu thị các chi tiết nhỏ hoặc biên (như cạnh sắc hoặc nhiễu).

Khi áp dụng bộ lọc Lowpass, ảnh đầu ra sẽ mất đi các chi tiết sắc nét và biên, trở nên mượt hơn. Bộ lọc này hoạt động bằng cách làm mờ (blur) ảnh thông qua việc áp dụng các kỹ thuật như:

* **Trung bình (Mean Filter):** Tính trung bình các giá trị pixel trong một vùng lân cận.
* **Gaussian Filter:** Sử dụng phân bố Gaussian để làm mờ, ưu tiên trọng số các điểm gần trung tâm hơn.

**Khi nào nên sử dụng bộ lọc Lowpass?**

* **Khi cần giảm nhiễu (Noise Reduction):**
  + Loại bỏ nhiễu ngẫu nhiên hoặc các chi tiết không mong muốn.
  + Ví dụ: Nhiễu muối tiêu hoặc nhiễu Gaussian trong ảnh.
* **Khi làm mờ ảnh (Blurring):**
  + Chuẩn bị ảnh cho các bước xử lý khác, chẳng hạn:
    - Tăng hiệu quả của việc phát hiện biên.
    - Làm trơn ảnh trước khi phân đoạn (segmentation).
* **Khi cần giảm bớt thông tin chi tiết:**
  + Làm ảnh trông mềm mại hơn (softening effect).
  + Tạo hiệu ứng làm mờ nền trong nhiếp ảnh.

**Bộ lọc Lowpass được sử dụng để làm gì?**

* **Làm mờ ảnh (Blurring):**
  + Tạo hiệu ứng làm mềm hình ảnh trong xử lý đồ họa.
* **Giảm nhiễu (Denoising):**
  + Loại bỏ các nhiễu có tần số cao, chẳng hạn như nhiễu ngẫu nhiên.
* **Tiền xử lý trước các thuật toán khác:**
  + Giảm chi tiết không cần thiết trước khi áp dụng các phương pháp phức tạp như phát hiện cạnh (Edge Detection) hoặc phân vùng (Segmentation).
* **Hậu xử lý trong đồ họa:**
  + Tạo hiệu ứng mờ để làm nổi bật các chi tiết quan trọng hơn hoặc làm cho ảnh trông tự nhiên hơn.

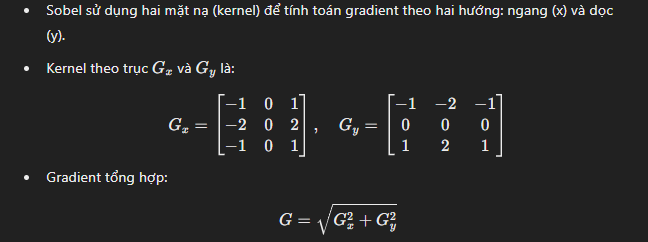
Tuy nhiên, việc sử dụng bộ lọc Lowpass cần lưu ý rằng nó có thể làm mất đi các chi tiết quan trọng trong ảnh nếu không được áp dụng đúng cách.

mặt nạ Sobel, Prewitt, và Robert

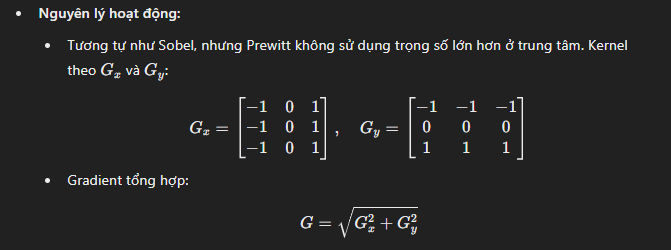
**Nguyên lý hoạt động của mặt nạ Sobel, Prewitt, và Robert trong xử lý ảnh**

Cả Sobel, Prewitt và Robert đều là các bộ lọc không gian (spatial filters) được sử dụng để phát hiện biên (Edge Detection) trong xử lý ảnh. Chúng hoạt động dựa trên việc tính toán đạo hàm bậc nhất của cường độ sáng trong ảnh, từ đó xác định các vùng chuyển đổi nhanh chóng về cường độ (biên).

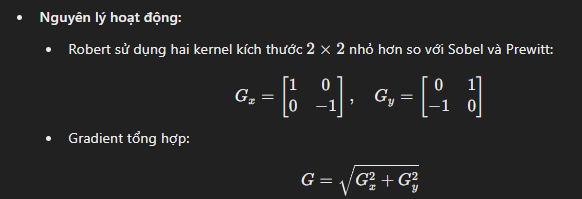
**1. Mặt nạ Sobel**

* **Nguyên lý hoạt động:**
  + Sobel sử dụng hai mặt nạ (kernel) để tính toán gradient theo hai hướng: ngang (x) và dọc (y).
  + Kernel theo trục GxG\_xGx​ và GyG\_yGy​ là:
* **Khi nào nên sử dụng?**
  + Khi cần phát hiện biên một cách mượt hơn, ít nhạy cảm với nhiễu.
  + Thường dùng trong các ứng dụng yêu cầu độ chi tiết cao.
* **Ứng dụng:**
  + Tìm cạnh trong ảnh (ví dụ: biên vật thể, đường viền khuôn mặt).
  + Là bước tiền xử lý cho nhận diện đối tượng.

**2. Mặt nạ Prewitt**

* **Nguyên lý hoạt động:**
  + Tương tự như Sobel, nhưng Prewitt không sử dụng trọng số lớn hơn ở trung tâm. **Khi nào nên sử dụng?**
  + Khi cần phát hiện biên nhanh, hiệu quả và ít nhạy cảm với nhiễu hơn Sobel.
* **Ứng dụng:**
  + Phát hiện biên trong ảnh có độ tương phản cao.
  + Thích hợp cho các bài toán không yêu cầu độ chính xác cao.

**3. Mặt nạ Robert**

* **Nguyên lý hoạt động:**
  + Robert sử dụng hai kernel kích thước 2×22 \times 22×2 nhỏ hơn so với Sobel và Prewitt:
* **Khi nào nên sử dụng?**
  + Khi cần phát hiện biên nhanh với độ chính xác cao ở ảnh có kích thước nhỏ.
  + Ứng dụng trên ảnh có độ nhiễu thấp.
* **Ứng dụng:**
  + Phân đoạn ảnh.
  + Phân tích biên trong ảnh y tế (như X-quang).

**So sánh Sobel, Prewitt, và Robert**

| **Phương pháp** | **Ưu điểm** | **Nhược điểm** |
| --- | --- | --- |
| Sobel | Nhạy với biên, mượt hơn, chống nhiễu tốt | Tốn thời gian xử lý hơn |
| Prewitt | Đơn giản, tính toán nhanh | Không hiệu quả trong xử lý nhiễu |
| Robert | Phát hiện biên tốt trên ảnh nhỏ | Không hiệu quả với ảnh lớn, dễ bị ảnh hưởng bởi nhiễu |

**Tóm lại, khi nào dùng?**

* **Sobel:** Khi cần phát hiện biên với độ chi tiết cao và khả năng chống nhiễu tốt.
* **Prewitt:** Khi cần hiệu quả và tốc độ nhanh cho ảnh có độ tương phản cao.
* **Robert:** Khi xử lý ảnh nhỏ hoặc yêu cầu tốc độ rất nhanh với độ chính xác cơ bản.

### ****1. Sự phân bố mức xám của ảnh****

#### **Nguyên lý hoạt động:**

* **Phân bố mức xám** thể hiện số lượng pixel tương ứng với mỗi mức xám (0-255 đối với ảnh 8-bit).
* Được biểu diễn bằng **histogram**, trục x là giá trị mức xám, trục y là số lượng pixel.

#### **Khi nào sử dụng?**

* Khi cần:
  + Phân tích cường độ sáng của ảnh.
  + Phát hiện vùng tối, sáng, hoặc tương phản.
  + Xử lý cân bằng sáng (histogram equalization).

#### **Ứng dụng:**

* Cải thiện chất lượng ảnh (tăng tương phản).
* Tìm ngưỡng phân đoạn (thresholding) trong ảnh nhị phân.
* Phân tích vùng ảnh quan trọng trong xử lý ảnh y tế.

### ****2. Biến đổi âm bản của ảnh****

#### **Nguyên lý hoạt động:**

* Biến đổi âm bản (Negative Transformation) thực hiện đảo ngược giá trị mức xám: s=L−1−rs = L - 1 - rs=L−1−r
  + LLL: số mức xám (256 cho ảnh 8-bit).
  + rrr: giá trị mức xám gốc.
  + sss: giá trị mức xám mới (âm bản).

#### **Khi nào sử dụng?**

* Khi cần:
  + Làm nổi bật các chi tiết ở vùng sáng.
  + Xử lý ảnh có độ sáng cao, như X-quang hoặc ảnh y tế.

#### **Ứng dụng:**

* Phân tích ảnh y tế (chẩn đoán từ X-quang hoặc CT scan).
* Tạo hiệu ứng ngược màu trong thiết kế đồ họa.
* Làm ảnh dễ đọc hơn trong các vùng sáng chói.

### ****3. Tạo ảnh nhị phân****

#### **Nguyên lý hoạt động:**

* Ảnh nhị phân chỉ có hai mức giá trị pixel: 000 (đen) và 111 (trắng).
* Quá trình tạo ảnh nhị phân dựa trên ngưỡng (threshold): g(x,y)={1neˆˊu f(x,y)>T0neˆˊu f(x,y)≤Tg(x, y) = \begin{cases} 1 & \text{nếu } f(x, y) > T \\ 0 & \text{nếu } f(x, y) \leq T \end{cases}g(x,y)={10​neˆˊu f(x,y)>Tneˆˊu f(x,y)≤T​
  + TTT: ngưỡng phân đoạn (threshold).

#### **Khi nào sử dụng?**

* Khi cần:
  + Phân đoạn (segmentation) các vùng quan tâm trong ảnh.
  + Đơn giản hóa ảnh để giảm chi tiết không cần thiết.

#### **Ứng dụng:**

* Xử lý ký tự (OCR - nhận diện văn bản).
* Nhận dạng đối tượng (Object Detection).
* Xử lý ảnh vệ tinh và ảnh y tế.
* Xác định vùng ROI (Region of Interest).

### ****Làm mỏng mặt phẳng bit (Bit Plane Slicing)****

#### **Nguyên lý hoạt động:**

* **Bit Plane Slicing** chia ảnh thành các **mặt phẳng bit** (bit planes), mỗi mặt phẳng tương ứng với một vị trí bit (từ bit thấp nhất LSB đến bit cao nhất MSB) trong biểu diễn nhị phân của giá trị mức xám.

Ví dụ: Với ảnh 8-bit (giá trị mức xám từ 0-255), mỗi pixel được biểu diễn dưới dạng 8 bit:

Pixel value: P=b7b6b5b4b3b2b1b0\text{Pixel value: } P = b\_7b\_6b\_5b\_4b\_3b\_2b\_1b\_0Pixel value: P=b7​b6​b5​b4​b3​b2​b1​b0​

* **Mặt phẳng bit thứ kkk:** Bit Plane k:bk=bit thứ k của moˆ˜i pixel.\text{Bit Plane } k: b\_k = \text{bit thứ } k \text{ của mỗi pixel.}Bit Plane k:bk​=bit thứ k của moˆ˜i pixel.

#### **Khi nào sử dụng?**

* Khi cần:
  + Phân tích thông tin trong ảnh ở mức độ chi tiết cao.
  + Tách và xử lý thông tin từ các mặt phẳng quan trọng (thường là mặt phẳng bit cao như b7b\_7b7​, b6b\_6b6​).
  + Loại bỏ nhiễu hoặc giảm dữ liệu (nén ảnh).

#### **Ứng dụng:**

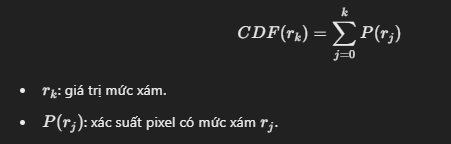
1. **Xử lý ảnh:**
   * Giữ lại các mặt phẳng bit cao để tái tạo ảnh rõ nét hơn (ảnh chỉ chứa thông tin quan trọng).
   * Phân tích chi tiết các vùng cụ thể trong ảnh.
2. **Mật mã ảnh (Steganography):**
   * Giấu thông tin trong các mặt phẳng bit thấp (LSB).
3. **Giảm nhiễu:**
   * Loại bỏ thông tin ở các mặt phẳng bit thấp vì chúng thường chứa nhiễu hơn.
4. **Nén ảnh:**
   * Chỉ giữ lại một số mặt phẳng bit để giảm dung lượng lưu trữ.

### ****Cân bằng mức xám (Histogram Equalization)****

#### **Nguyên lý hoạt động:**

* **Cân bằng mức xám** (Histogram Equalization) là một kỹ thuật tăng cường độ tương phản của ảnh bằng cách phân bố lại giá trị mức xám dựa trên histogram của ảnh.
* Mục tiêu: Chuyển đổi histogram của ảnh gốc (thường tập trung ở một vùng) thành histogram cân đối hơn (trải đều các mức xám).

#### **Các bước thực hiện:**

1. **Tính histogram của ảnh**: Đếm số pixel tại mỗi mức xám.
2. **Tính xác suất tích lũy (CDF - Cumulative Distribution Function)**: 
3. **Chuyển đổi giá trị mức xám**:

#### **Khi nào sử dụng?**

* Khi cần:
  + Tăng cường độ tương phản cho ảnh.
  + Làm rõ các chi tiết bị mờ hoặc khó phân biệt trong ảnh.
  + Xử lý ảnh có độ sáng không đồng đều hoặc bị tối.

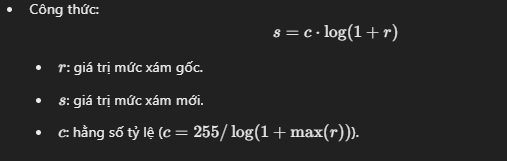
#### **Ứng dụng:**

1. **Xử lý ảnh y tế:**
   * Tăng cường chi tiết trong ảnh X-quang, CT scan, hoặc MRI.
2. **Ảnh vệ tinh:**
   * Làm rõ các chi tiết đất đai, thực vật, hoặc vùng nước.
3. **Nhận dạng mẫu:**
   * Chuẩn bị dữ liệu đầu vào cho các thuật toán nhận dạng khuôn mặt, vật thể.
4. **Thiết kế đồ họa:**
   * Cải thiện chất lượng ảnh trước khi in ấn hoặc xử lý hậu kỳ.

### ****1. Biến đổi Logarit****

#### **Nguyên lý hoạt động:**

* Biến đổi logarit (Log Transformation) thực hiện nén các giá trị mức xám cao và làm rõ các giá trị mức xám thấp trong ảnh.



#### **Khi nào sử dụng?**

* Khi cần:
  + Làm nổi bật chi tiết trong vùng tối hoặc sáng.
  + Xử lý ảnh có dải mức xám rộng (ảnh với độ sáng chênh lệch cao).

#### **Ứng dụng:**

1. **Xử lý ảnh y tế:**
   * Làm rõ các vùng tối trên X-quang hoặc ảnh siêu âm.
2. **Xử lý ảnh thiên văn:**
   * Nén độ sáng cao từ các ngôi sao hoặc nguồn sáng mạnh.
3. **Tăng cường ảnh:**
   * Làm rõ các chi tiết bị che khuất bởi độ sáng.

### ****2. Biến đổi Lũy thừa (Power-Law Transformation)****

#### **Nguyên lý hoạt động:**

* Biến đổi lũy thừa (Gamma Correction) điều chỉnh độ sáng hoặc tương phản của ảnh bằng cách nâng giá trị mức xám lên lũy thừa gamma (γ\gammaγ).

#### **Khi nào sử dụng?**

* Khi cần:
  + Tăng hoặc giảm độ sáng toàn cục của ảnh.
  + Điều chỉnh độ sáng trong các ứng dụng đặc biệt (hiển thị màn hình, ảnh kỹ thuật số).

#### **Ứng dụng:**

1. **Hiển thị ảnh:**
   * Điều chỉnh độ sáng phù hợp với màn hình hoặc thiết bị hiển thị.
2. **Xử lý ảnh y tế:**
   * Làm rõ các chi tiết bị ẩn trong vùng tối hoặc sáng.
3. **Tăng cường ảnh:**
   * Tăng hoặc giảm độ tương phản trong các vùng cụ thể.

### ****Phân tích Ngưỡng (Thresholding Analysis)****

#### **Nguyên lý hoạt động:**

* **Ngưỡng** là một kỹ thuật trong xử lý ảnh, dùng để phân chia ảnh thành hai vùng chính: **vùng foreground** (vùng đối tượng) và **vùng background** (vùng nền).
* Phân tích ngưỡng thực hiện bằng cách so sánh các giá trị mức xám của từng pixel trong ảnh với một ngưỡng TTT. Nếu giá trị pixel lớn hơn hoặc bằng ngưỡng TTT, pixel đó được gán giá trị tối đa (thường là 255); ngược lại, pixel đó được gán giá trị tối thiểu (thường là 0).

Công thức:

#### 

#### **Khi nào sử dụng?**

* Khi cần:
  + Phân tách các đối tượng trong ảnh (ví dụ: phân đoạn các đối tượng trên nền).
  + Tạo ảnh nhị phân từ ảnh xám.
  + Xử lý ảnh với các mục đích như nhận dạng, theo dõi chuyển động, hoặc phát hiện biên.

#### **Ứng dụng:**

1. **Nhận dạng đối tượng:**
   * Tách các đối tượng ra khỏi nền để nhận diện, phân loại.
2. **Phát hiện biên:**
   * Phát hiện các biên của đối tượng trong ảnh.
3. **Ứng dụng trong y tế:**
   * Phát hiện vùng tổn thương trong ảnh y tế (X-quang, MRI, siêu âm).
4. **Ảnh vệ tinh:**
   * Phân loại đất đai, khu vực nước, khu vực thực vật, v.v.

#### **Các loại phân ngưỡng:**

1. **Phân ngưỡng toàn cục (Global Thresholding):**
   * Sử dụng một ngưỡng cố định cho toàn bộ ảnh.
2. **Phân ngưỡng thích ứng (Adaptive Thresholding):**
   * Áp dụng ngưỡng khác nhau cho các vùng khác nhau của ảnh, giúp xử lý các ảnh có độ sáng không đồng đều.
3. **Phân ngưỡng Otsu:**
   * Tự động tìm ngưỡng tối ưu bằng cách tối đa hóa phương sai giữa các lớp (background và foreground).

### ****Kỹ Thuật Chọn Ngưỡng (Thresholding Selection Techniques)****

#### **Nguyên lý hoạt động:**

Kỹ thuật chọn ngưỡng nhằm xác định một giá trị ngưỡng tối ưu để phân chia ảnh thành các khu vực khác nhau (thường là giữa foreground và background). Việc chọn ngưỡng có thể được thực hiện thủ công hoặc tự động thông qua các phương pháp phân tích và tính toán, giúp tối ưu hóa việc phân đoạn ảnh.

Các phương pháp chọn ngưỡng phổ biến gồm:

1. **Phân ngưỡng toàn cục (Global Thresholding)**
2. **Phân ngưỡng thích ứng (Adaptive Thresholding)**
3. **Phương pháp Otsu**
4. **Phương pháp K-means**

### ****1. Phân Ngưỡng Toàn Cục (Global Thresholding)****

#### **Nguyên lý hoạt động:**

* Phân ngưỡng toàn cục là phương pháp sử dụng một ngưỡng cố định trên toàn bộ ảnh để phân chia ảnh thành hai vùng: một vùng đối tượng và một vùng nền.
* Ngưỡng này có thể được chọn thủ công hoặc tự động dựa trên phân tích histogram.

#### **Khi nào sử dụng?**

* Dùng trong các trường hợp mà độ sáng của ảnh là đồng nhất, không có sự thay đổi đáng kể giữa các vùng trong ảnh.

#### **Ứng dụng:**

* Phân loại ảnh có nền đồng nhất.
* Xử lý các ảnh có độ tương phản cao giữa foreground và background.

### ****2. Phân Ngưỡng Thích Ứng (Adaptive Thresholding)****

#### **Nguyên lý hoạt động:**

* Phân ngưỡng thích ứng tính toán ngưỡng riêng biệt cho từng pixel, thay vì sử dụng một ngưỡng duy nhất cho toàn bộ ảnh.
* Phương pháp này hữu ích khi độ sáng trong ảnh không đồng đều. Ngưỡng sẽ thay đổi tùy theo khu vực ảnh (local neighborhoods).

#### 

#### **Khi nào sử dụng?**

* Khi ảnh có độ sáng không đồng đều, ví dụ như ảnh chụp trong điều kiện ánh sáng không ổn định.

#### **Ứng dụng:**

* Phân đoạn ảnh có ánh sáng không đồng đều.
* Xử lý ảnh trong điều kiện ánh sáng phức tạp (chẳng hạn như ảnh ngoài trời).

### ****3. Phương Pháp Otsu****

#### **Nguyên lý hoạt động:**

* Phương pháp Otsu tìm ngưỡng tối ưu bằng cách tối đa hóa phương sai giữa hai lớp (background và foreground).
* Otsu phân chia ảnh thành hai lớp sao cho tổng phương sai trong các lớp này là lớn nhất, giúp tách biệt foreground và background một cách hiệu quả.
* Phương pháp này không yêu cầu người dùng chọn ngưỡng thủ công và có thể hoạt động tốt ngay cả khi không biết rõ về phân bố mức xám của ảnh.

#### **Khi nào sử dụng?**

* Khi cần tìm ngưỡng tối ưu một cách tự động mà không cần thông tin về ảnh trước.
* Dùng trong các trường hợp phân đoạn ảnh có độ tương phản thấp hoặc không đồng đều.

#### **Ứng dụng:**

* Phân đoạn ảnh trong phân tích y tế, chẳng hạn như phân tách vùng tổn thương trong ảnh X-quang.
* Phân đoạn ảnh trong nhận diện đối tượng tự động.

### ****4. Phương Pháp K-means****

#### **Nguyên lý hoạt động:**

* K-means là một phương pháp phân cụm trong học máy, có thể áp dụng để chọn ngưỡng. K-means phân chia ảnh thành KKK nhóm, mỗi nhóm đại diện cho một phần của ảnh.
* Sau khi phân cụm, giá trị ngưỡng có thể được tính toán dựa trên các trung tâm cụm, giúp phân chia ảnh thành các vùng tương ứng.

#### **Khi nào sử dụng?**

* Khi ảnh có nhiều đối tượng và cần phân tách các vùng theo nhóm.
* Khi không chỉ có hai lớp (foreground và background) mà có thể có nhiều lớp khác nhau.

#### **Ứng dụng:**

* Phân đoạn ảnh trong nhận dạng đối tượng, ví dụ như phân đoạn các đối tượng trong ảnh nhiều lớp.
* Phân nhóm các pixel trong các ứng dụng phân tích ảnh phức tạp.

### ****Phép Trừ Ảnh (Image Subtraction)****

#### **Nguyên lý hoạt động:**

Phép trừ ảnh là quá trình trừ giá trị mức xám của mỗi pixel trong một ảnh từ giá trị mức xám của pixel tương ứng trong một ảnh khác. Kết quả là một ảnh mới, nơi mỗi pixel có giá trị là hiệu giữa các pixel của hai ảnh.

* Phép trừ ảnh có thể được áp dụng để làm nổi bật sự khác biệt giữa hai ảnh, ví dụ như phát hiện thay đổi trong ảnh động, nhận diện đối tượng di chuyển hoặc tách biệt các đối tượng.

#### **Khi nào sử dụng?**

* **Phát hiện thay đổi:** Khi bạn có hai ảnh chụp ở các thời điểm khác nhau và muốn phát hiện sự thay đổi giữa chúng (ví dụ: phát hiện sự chuyển động trong video).
* **Nhận diện đối tượng:** Trong một số trường hợp, phép trừ ảnh giúp loại bỏ phần nền để chỉ còn lại các đối tượng cần nhận diện.
* **Tách biệt đối tượng:** Phép trừ ảnh có thể giúp làm nổi bật các đối tượng trong ảnh bằng cách loại bỏ các vùng nền.

#### **Ứng dụng:**

* **Phát hiện chuyển động:** Ví dụ, trong hệ thống giám sát, bạn có thể so sánh ảnh chụp trong hai thời điểm khác nhau để phát hiện sự thay đổi giữa các khung hình.
* **Giảm nhiễu ảnh:** Trong trường hợp có nhiều ảnh nền giống nhau, phép trừ có thể loại bỏ nền và giúp làm sạch các đối tượng.
* **Nhận diện vật thể:** Tìm kiếm các đối tượng khác biệt giữa các ảnh.

### ****Lọc Tuyến Tính (Linear Filtering)****

#### **Nguyên lý hoạt động:**

Lọc tuyến tính là quá trình áp dụng một bộ lọc (hoặc mặt nạ) lên mỗi pixel của ảnh, trong đó giá trị của pixel kết quả được tính toán dựa trên một phép toán tuyến tính từ các pixel xung quanh nó. Mỗi pixel trong ảnh đầu vào sẽ được thay thế bằng một giá trị tính toán từ các pixel trong một cửa sổ nhỏ (kernel) xung quanh nó.

Lọc tuyến tính có thể được sử dụng để làm mượt ảnh (lọc trung bình), làm sắc nét (lọc Laplace, Sobel), hoặc giảm nhiễu ảnh.

#### **Khi nào sử dụng?**

* **Lọc trung bình (Average Filtering):** Dùng để làm mượt ảnh, giảm nhiễu.
* **Lọc làm sắc nét (Sharpening):** Dùng để làm sắc nét các đối tượng trong ảnh.
* **Lọc Laplace:** Dùng để phát hiện biên (edges).
* **Giảm nhiễu:** Loại bỏ các nhiễu nhỏ trong ảnh.

#### **Ứng dụng:**

* **Xử lý ảnh y tế:** Làm mượt hoặc làm sắc nét ảnh X-quang.
* **Nhận dạng đối tượng:** Phát hiện biên của đối tượng trong ảnh.
* **Giảm nhiễu:** Xử lý ảnh vệ tinh hoặc ảnh có nhiễu.

### ****Các Hàm Con Liên Quan đến Lọc Tuyến Tính trong MATLAB****

#### **1. Lọc Trung Bình (Average Filtering)**

Hàm này áp dụng bộ lọc trung bình (mean filter), nơi mỗi pixel trong ảnh đầu ra là trung bình của các pixel trong một cửa sổ xung quanh nó.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Kích thước của kernel

kernel\_size = 3;

kernel = ones(kernel\_size) / (kernel\_size \* kernel\_size);

% Áp dụng lọc trung bình

img\_filtered = imfilter(img\_gray, kernel, 'same');

% Hiển thị kết quả

imshow(img\_filtered);

title('Ảnh sau lọc trung bình');

#### **2. Lọc Làm Sắc Nét (Sharpening Filter)**

Lọc làm sắc nét sử dụng một bộ lọc nâng cao để làm rõ các chi tiết và biên trong ảnh.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Tạo kernel làm sắc nét

kernel\_sharpen = [-1 -1 -1; -1 9 -1; -1 -1 -1];

% Áp dụng lọc làm sắc nét

img\_sharpened = imfilter(img\_gray, kernel\_sharpen, 'same');

% Hiển thị kết quả

imshow(img\_sharpened);

title('Ảnh sau lọc làm sắc nét');

#### **3. Lọc Sobel (Edge Detection Filter)**

Lọc Sobel là một phương pháp phổ biến để phát hiện biên (edge) trong ảnh.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Áp dụng bộ lọc Sobel

img\_sobel = edge(img\_gray, 'Sobel');

% Hiển thị kết quả

imshow(img\_sobel);

title('Ảnh sau lọc Sobel (Edge Detection)');

#### **4. Lọc Laplace (Laplacian Filter)**

Lọc Laplace được sử dụng để phát hiện các biên mạnh trong ảnh, đặc biệt là các chuyển đổi đột ngột giữa các mức xám.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Tạo kernel Laplacian

kernel\_laplace = [0 1 0; 1 -4 1; 0 1 0];

% Áp dụng lọc Laplace

img\_laplace = imfilter(img\_gray, kernel\_laplace, 'same');

% Hiển thị kết quả

imshow(img\_laplace);

title('Ảnh sau lọc Laplace');

#### **5. Lọc Gauss (Gaussian Filter)**

Lọc Gauss thường được sử dụng để làm mượt ảnh và giảm nhiễu.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Tạo bộ lọc Gaussian

sigma = 2; % Độ lệch chuẩn của Gaussian

kernel\_gauss = fspecial('gaussian', [5 5], sigma);

% Áp dụng lọc Gauss

img\_gaussian = imfilter(img\_gray, kernel\_gauss, 'same');

% Hiển thị kết quả

imshow(img\_gaussian);

title('Ảnh sau lọc Gaussian');

### ****Kết luận:****

* **Lọc tuyến tính** là một kỹ thuật cơ bản nhưng rất mạnh trong xử lý ảnh, được sử dụng để làm mượt ảnh, giảm nhiễu, làm sắc nét hoặc phát hiện biên.
* Các bộ lọc phổ biến như lọc trung bình, lọc Sobel, lọc Laplace và lọc Gauss đều có ứng dụng riêng trong các bài toán khác nhau về xử lý ảnh.

### ****Lọc Trung Vị (Median Filtering) - Lọc Phi Tuyến****

#### **Nguyên lý hoạt động:**

Lọc trung vị là một phương pháp lọc phi tuyến được sử dụng để làm mượt ảnh, đặc biệt là để giảm nhiễu. Thay vì tính toán giá trị trung bình như trong lọc tuyến tính, lọc trung vị thay thế giá trị của mỗi pixel trong ảnh bằng giá trị trung vị trong cửa sổ (kernel) xung quanh nó.

* **Trung vị** của một tập hợp số là giá trị ở vị trí giữa khi các giá trị được sắp xếp theo thứ tự tăng dần.
* Lọc trung vị có khả năng giữ lại các biên (edges) trong ảnh trong khi loại bỏ nhiễu muối và tiêu (salt-and-pepper noise).

#### **Công thức:**

#### **Khi nào sử dụng?**

* **Giảm nhiễu muối và tiêu:** Lọc trung vị rất hiệu quả khi xử lý ảnh bị nhiễu muối và tiêu (salt-and-pepper noise).
* **Giữ lại biên:** Khác với lọc tuyến tính (như lọc trung bình), lọc trung vị có thể làm mượt ảnh mà không làm mờ đi các chi tiết biên (edges) quan trọng.
* **Ứng dụng trong ảnh y tế hoặc vệ tinh:** Phương pháp này thường được sử dụng để làm sạch ảnh mà không làm mất đi các thông tin quan trọng.

#### **Ứng dụng:**

* **Giảm nhiễu trong ảnh:** Lọc trung vị là một phương pháp phổ biến trong xử lý ảnh khi cần loại bỏ nhiễu mà không làm mờ các biên của đối tượng.
* **Chế độ ảnh y tế và vệ tinh:** Cải thiện chất lượng ảnh chụp trong các ứng dụng như MRI hoặc ảnh vệ tinh, nơi mà nhiễu muối và tiêu có thể xuất hiện do các sai sót trong quá trình chụp.
* **Ảnh trong nhận diện hình ảnh:** Dùng để làm sạch ảnh trước khi thực hiện các thao tác xử lý khác như nhận diện đối tượng.

### ****Các Hàm Con Liên Quan đến Lọc Trung Vị trong MATLAB****

#### **1. Lọc Trung Vị (Median Filter)**

Hàm medfilt2 trong MATLAB là một hàm rất dễ sử dụng để áp dụng lọc trung vị lên một ảnh.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Áp dụng lọc trung vị với kích thước cửa sổ 3x3

img\_filtered = medfilt2(img\_gray, [3 3]);

% Hiển thị kết quả

imshow(img\_filtered);

title('Ảnh sau lọc trung vị');

#### **2. Lọc Trung Vị với Cửa Sổ Lớn hơn**

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Áp dụng lọc trung vị với cửa sổ lớn hơn 5x5

img\_filtered = medfilt2(img\_gray, [5 5]);

% Hiển thị kết quả

imshow(img\_filtered);

title('Ảnh sau lọc trung vị (cửa sổ 5x5)');

#### **3. Lọc Trung Vị với Cửa Sổ Không Cân Đối**

Nếu bạn muốn sử dụng một cửa sổ không vuông, bạn có thể chọn một cửa sổ hình chữ nhật.

matlab

Sao chép mã

% Đọc ảnh xám

img = imread('image.png');

img\_gray = rgb2gray(img); % Chuyển sang ảnh xám nếu ảnh là màu

% Áp dụng lọc trung vị với cửa sổ hình chữ nhật 3x5

img\_filtered = medfilt2(img\_gray, [3 5]);

% Hiển thị kết quả

imshow(img\_filtered);

title('Ảnh sau lọc trung vị (cửa sổ 3x5)');

### ****Kết luận:****

* **Lọc trung vị** là một phương pháp phi tuyến mạnh mẽ trong xử lý ảnh, đặc biệt hiệu quả trong việc giảm nhiễu muối và tiêu mà vẫn giữ lại các biên quan trọng của ảnh.
* So với các bộ lọc tuyến tính như lọc trung bình, lọc trung vị thường tốt hơn trong việc bảo tồn các đặc trưng ảnh, đồng thời loại bỏ các nhiễu không mong muốn.