ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO

Xử lý ảnh

ĐỀ TÀI:

Image Denoising Using Directional Filter Banks

Mã môn học : **2425II_INT3404E_1**

Sinh viên thực hiện : **Nguyễn Mạnh Quân** **22028171**

Nguyễn Khôi Nguyên **22028032**

Trịnh Ngọc Chiến **22028037**

Hà Quang Nhuệ **21021524**

MỤC LỤC

Tóm tắt

Báo cáo này trình bày một phương pháp kết hợp giữa biến đổi wavelet và directional filter banks (DFB) để khử nhiễu ảnh. Phương pháp đề xuất tận dụng ưu điểm của biến đổi wavelet trong việc phân tách tần số và khả năng bắt các đặc trưng định hướng của DFB để loại bỏ hiệu quả nhiễu trong ảnh, đồng thời bảo toàn các cạnh và chi tiết quan trọng.

Kết quả thực nghiệm trên các loại nhiễu khác nhau (Gaussian, Gaussian White và Salt & Pepper) cho thấy sự cải thiện đáng kể về chất lượng ảnh thông qua các chỉ số PSNR và SSIM.

1. Giới thiệu

1.1. Bối cảnh và tầm quan trọng

Khử nhiễu ảnh là một trong những bài toán cơ bản và quan trọng trong xử lý ảnh số. Nhiễu có thể xuất hiện trong quá trình thu nhận, truyền tải hoặc lưu trữ ảnh, làm giảm chất lượng ảnh và ảnh hưởng đến các quá trình xử lý tiếp theo như phân đoạn, nhận dạng hoặc phân loại.

Do đó, việc phát triển các phương pháp khử nhiễu hiệu quả có ý nghĩa quan trọng trong nhiều ứng dụng thực tế như y tế, viễn thám, giám sát an ninh và nhiều lĩnh vực khác.

1.2. Thách thức trong khử nhiễu ảnh

Khử nhiễu ảnh đối mặt với một thách thức cơ bản: làm thế nào để loại bỏ nhiễu một cách hiệu quả trong khi vẫn bảo toàn các chi tiết quan trọng như cạnh, đường nét và kết cấu.

Các phương pháp khử nhiễu truyền thống thường gặp phải sự đánh đổi giữa khả năng làm mịn vùng nhiễu và bảo toàn chi tiết. Ví dụ, bộ lọc trung bình hoặc Gaussian có thể loại bỏ nhiễu hiệu quả nhưng đồng thời làm mờ các cạnh quan trọng trong ảnh.

1.3. Mục tiêu nghiên cứu

Nghiên cứu này hướng đến việc phát triển một phương pháp khử nhiễu ảnh kết hợp ưu điểm của biến đổi wavelet và directional filter banks, nhằm:

- Loại bỏ nhiễu hiệu quả từ các loại nhiễu khác nhau
- Bảo toàn tối đa các đặc trưng cấu trúc quan trọng như cạnh và kết cấu
- Đạt được chất lượng khử nhiễu cao được đánh giá qua các chỉ số khách quan như PSNR và SSIM

2. Cơ sở lý thuyết

2.1. Biến đổi Wavelet 2D

Biến đổi wavelet 2D là một công cụ mạnh mẽ trong xử lý ảnh cho phép phân tích ảnh ở các thang độ phân giải khác nhau. Biến đổi wavelet phân tách ảnh thành các thành

phần tần số thấp (approximation coefficients) và các thành phần tần số cao (detail coefficients) theo các hướng khác nhau.

Khi áp dụng biến đổi wavelet 2D vào một ảnh, ta thu được bốn thành phần con:

- Hệ số xấp xỉ (LL): Phiên bản có độ phân giải thấp hơn của ảnh gốc
- Hệ số chi tiết theo hướng ngang (LH): Bắt các cạnh ngang
- Hệ số chi tiết theo hướng dọc (HL): Bắt các cạnh dọc
- Hệ số chi tiết theo hướng chéo (HH): Bắt các cạnh chéo

Ưu điểm của biến đổi wavelet trong khử nhiễu ảnh:

- Tính đa phân giải: Cho phép xử lý ảnh ở các mức độ chi tiết khác nhau
- Tính cục bộ trong cả không gian và tần số: Cho phép phân tích cục bộ của ảnh
- Khả năng nén năng lượng: Năng lượng của ảnh được tập trung vào một số ít hệ số

Toán học của biến đổi wavelet 2D

Biến đổi wavelet 2D của một ảnh f(x,y) có thể được biểu diễn như sau:

$$W_\psi f(j,m,n) = \frac{1}{\sqrt{2^j}} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\psi_{j,m,n}(x,y)dxdy$$

Trong đó:

- $\psi_{j,m,n}(x,y)dxdy$ là hàm wavelet mẹ được co giãn và dịch chuyển
- j là mức phân giải
- m, n là các tham số dịch chuyển

2.2. Directional Filter Banks (DFB)

Directional Filter Banks (DFB) là một công cụ phân tích định hướng cho phép phân tách ảnh thành các thành phần theo nhiều hướng khác nhau. DFB có khả năng bắt các đặc trưng theo hướng như đường thẳng, cạnh, và các kết cấu định hướng.

Nguyên lý hoạt động của DFB:

1. Sử dụng các bộ lọc có phản ứng theo hướng cụ thể
2. Phân tách ảnh thành các thành phần (subbands) theo các hướng khác nhau
3. Mỗi subband chứa thông tin về các cấu trúc theo một hướng cụ thể

Ưu điểm của DFB trong khử nhiễu:

- Khả năng bắt các đặc trưng định hướng trong ảnh

- Phân biệt hiệu quả giữa cấu trúc định hướng và nhiễu đẳng hướng
- Bảo toàn tốt các cạnh theo các hướng khác nhau

Trong cài đặt, các bộ lọc định hướng đơn giản được sử dụng để bắt các đặc trưng theo các hướng cơ bản:

- Ngang: Bắt các cạnh ngang
- Dọc: Bắt các cạnh dọc
- Chéo 45°: Bắt các cạnh chéo theo hướng 45°
- Chéo 135°: Bắt các cạnh chéo theo hướng 135°
- Tùy chỉnh 30°: Bổ sung bắt các cạnh theo hướng 30°

2.3. BayesShrink

BayesShrink là một phương pháp thích nghi để xác định ngưỡng tối ưu cho việc khử nhiễu trong miền wavelet, dựa trên lý thuyết Bayes. Phương pháp này ước lượng ngưỡng tối ưu cho từng subband dựa trên đặc tính thống kê của hệ số wavelet và mức độ nhiễu.

Ngưỡng BayesShrink được tính như sau:

$$T_B = \frac{\sigma_n^2}{\sigma_s}$$

Trong đó:

- σ_n^2 là phương sai của nhiễu
- σ_s là độ lệch chuẩn của tín hiệu không nhiễu

Ưu điểm của BayesShrink:

- Thích nghi với đặc tính cục bộ của ảnh
- Bảo toàn tốt các chi tiết quan trọng
- Hiệu quả với nhiều loại nhiễu khác nhau

3. Phương pháp đề xuất

3.1. Tổng quan về phương pháp

Phương pháp đề xuất kết hợp biến đổi wavelet và directional filter banks trong một pipeline thống nhất để khử nhiễu ảnh. Phương pháp này bao gồm các bước chính sau:

1. Phân rã ảnh bằng biến đổi wavelet 2D đa mức
2. Áp dụng DFB cho các thành phần chi tiết (detail coefficients)

3. Thực hiện khử nhiễu trên các subbands định hướng bằng BayesShrink

4. Tái tạo ảnh không nhiễu từ các hệ số đã được xử lý

5. Hậu xử lý để làm mịn và bảo toàn cạnh

3.2. Chi tiết quy trình xử lý

3.2.1. Phân rã wavelet

Ảnh đầu vào được phân rã bằng biến đổi wavelet 2D với wavelet mẹ 'db2' (Daubechies) và mức phân rã là 3. Quá trình này tạo ra một cấu trúc phân cấp gồm các hệ số xấp xỉ ở mức cao nhất và các hệ số chi tiết ở các mức khác nhau.

```
def wavelet_decompose(img: np.ndarray, level: int = 3, wavelet: str =
    'db2'):
     """Phân rã ảnh bằng wavelet 2D."""
     return pywt.wavedec2(img, wavelet=wavelet, level=level)
```

3.2.2. Phân tích directional filter banks

Các hệ số chi tiết wavelet được tiếp tục phân tích bằng các bộ lọc định hướng để phân tách thành các subband định hướng. Mỗi subband chứa thông tin về một hướng cụ thể trong ảnh.

```
def dfb_decompose(img: np.ndarray):
     """Phân tích ảnh thành các thành phần định hướng sử dụng bộ lọc đơn
    giản và bộ đệm."""
     subbands = {}
     for name, kernel in DIRECTIONAL_FILTERS.items():
         subbands[name] = convolve2d(img, kernel, mode='same',
    boundary='symm')
     return subbands
```

3.2.3. Khử nhiễu với BayesShrink

Mỗi subband định hướng được khử nhiễu bằng phương pháp BayesShrink. Mức độ nhiễu được ước lượng từ ảnh đầu vào, và ngưỡng thích nghi được tính toán cho từng subband.

```
def bayes_shrink(coeff: np.ndarray, sigma: float):
     """BayesShrink thresholding cho khử nhiễu."""
     var = np.var(coeff)
     thresh = sigma**2 / (np.sqrt(var) + 1e-8)
     return np.sign(coeff) * np.maximum(np.abs(coeff) - thresh, 0)
def dfb_denoise(subbands: dict, sigma: float, alpha: float = 0.9):
```

```
    """Khử nhiễu từng subband với BayesShrink và pha trộn giữ chi tiết."""
    denoised = {}
    for name, band in subbands.items():
        clean = bayes_shrink(band, sigma)
        denoised[name] = alpha * clean + (1 - alpha) * band
    return denoised
```

3.2.4. Tổng hợp và tái tạo ảnh

Các subband đã được khử nhiễu được tổng hợp lại thành các hệ số chi tiết wavelet, và ảnh được tái tạo thông qua biến đổi wavelet ngược.

```
def dfb_reconstruct(subbands: dict):
    """Tổng hợp lại ảnh từ các thành phần định hướng."""
    return np.mean(list(subbands.values()), axis=0)

def wavelet_reconstruct(coeffs, wavelet: str = 'db2'):
    """Khôi phục ảnh từ hệ số wavelet."""
    return pywt.waverec2(coeffs, wavelet=wavelet)
```

3.2.5. Hậu xử lý

Cuối cùng, một bước hậu xử lý được áp dụng để làm mịn nhẹ ảnh trong khi vẫn bảo toàn các cạnh quan trọng, sử dụng lọc song phương (bilateral filtering).

```
def post_processing(img: np.ndarray):
    """Làm mịn nhẹ ảnh nhưng giữ biên."""
    img_clipped = np.clip(img, 0, 1)
    return denoise_bilateral(img_clipped, sigma_color=0.03, sigma_spatial=10)
```

3.3. Ước lượng mức độ nhiễu

Một điểm quan trọng trong phương pháp của chúng tôi là khả năng ước lượng tự động mức độ nhiễu từ ảnh đầu vào, giúp điều chỉnh quy trình khử nhiễu phù hợp với từng ảnh cụ thể.

```
def estimate_sigma(noisy_img: np.ndarray):
    """Ước lượng độ lệch chuẩn của nhiễu từ ảnh."""
    smooth = convolve2d(noisy_img, np.ones((3, 3)) / 9, mode='same',
boundary='symm')
```

```
    return np.std(noisy_img - smooth)
```

3.4. Pipeline khử nhiễu hoàn chỉnh

Quá trình khử nhiễu hoàn chỉnh được tích hợp trong một pipeline thống nhất:

```
def denoise_image(img: np.ndarray, wavelet: str = 'db2', level: int = 3):
    """Pipeline khử nhiễu ảnh."""
    sigma = estimate_sigma(img)
    coeffs = wavelet_decompose(img, level=level, wavelet=wavelet)
    denoised_coeffs = dfb_denoise_wavelet(coeffs, sigma)
    denoised_img = wavelet_reconstruct(denoised_coeffs, wavelet=wavelet)
    return post_processing(denoised_img)
```

4. Thực nghiệm và đánh giá

4.1. Dữ liệu thực nghiệm

Phương pháp đề xuất được kiểm tra trên ảnh đầu vào với ba loại nhiễu phổ biến:

- Nhiễu Gaussian (var=0.01): Nhiễu dạng mịn, thường gặp trong ảnh kỹ thuật số
- Nhiễu Gaussian White (mean=0, var=0.02): Nhiễu trắng Gaussian, mô phỏng nhiễu điện tử
- Nhiễu Salt & Pepper (amount=0.05): Nhiễu đốm, thường gặp trong truyền tín hiệu

4.2. Chỉ số đánh giá

Để đánh giá hiệu quả của phương pháp, hai chỉ số phổ biến được sử dụng:

4.2.1. Peak Signal-to-Noise Ratio (PSNR)

PSNR đo lường sự khác biệt giữa ảnh gốc và ảnh đã khử nhiễu, tính bằng decibel (dB). Giá trị PSNR càng cao càng tốt.

$$PSNR = 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

Trong đó:

- MAX_I là giá trị cường độ tối đa có thể của ảnh (trong trường hợp này là 1.0)
- MSE là sai số bình phương trung bình giữa ảnh gốc và ảnh đã xử lý

```
def psnr(img1: np.ndarray, img2: np.ndarray):
    """Tính PSNR giữa 2 ảnh."""
    mse = np.mean((img1 - img2) ** 2)
```

```
    return 20 * np.log10(1.0 / np.sqrt(mse)) if mse != 0 else 100
```

4.2.2. Structural Similarity Index (SSIM)

SSIM đánh giá sự tương đồng về cấu trúc giữa ảnh gốc và ảnh đã khử nhiễu. SSIM dao động trong khoảng [0, 1], với 1 là sự trùng khớp hoàn hảo. SSIM dựa trên sự so sánh về độ sáng, độ tương phản và cấu trúc giữa hai ảnh:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma}$$

Trong đó:

- l(x,y) là so sánh về độ sáng
- c(x,y) là so sánh về độ tương phản
- s(x,y) là so sánh về cấu trúc
- α, β, γ là các hệ số điều chỉnh tầm quan trọng của mỗi thành phần

```
def calculate_ssim(img1: np.ndarray, img2: np.ndarray):
    """Tính SSIM giữa 2 ảnh."""
    return ssim(img1, img2, data_range=1.0)
```

4.3. Kết quả thực nghiệm

4.3.1. So sánh trực quan

Kết quả thực nghiệm được hiển thị dưới dạng hình ảnh để so sánh trực quan giữa ảnh gốc, ảnh nhiễu và ảnh đã khử nhiễu. Việc hiển thị này cho phép đánh giá nhanh về hiệu quả của phương pháp đề xuất.

```
def display_comparison(original: np.ndarray, noisy: np.ndarray, denoised:
    np.ndarray, noise_type: str):
    """Hiển thị và đánh giá ảnh trước/ sau khử nhiễu."""
    # ... [hiển thị ảnh và tính toán các chỉ số]
```

4.3.2. So sánh định lượng

Kết quả định lượng được đánh giá thông qua các chỉ số PSNR và SSIM cho mỗi loại nhiễu trước và sau khi khử nhiễu:

Ảnh 1:

◈ Gaussian Noise:

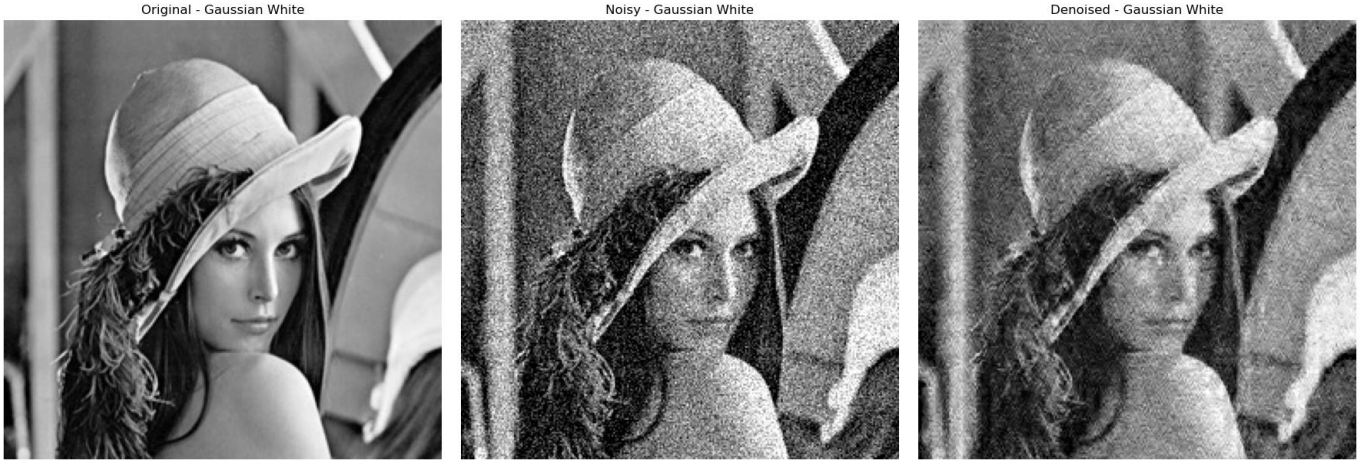
➤ PSNR (noisy): 20.37 dB | SSIM: 0.4627

➤ PSNR (denoised): 22.32 dB | SSIM: 0.5429

Gaussian White Noise:

➤ PSNR (noisy): 17.52 dB | SSIM: 0.3602

➤ PSNR (denoised): 20.56 dB | SSIM: 0.4440

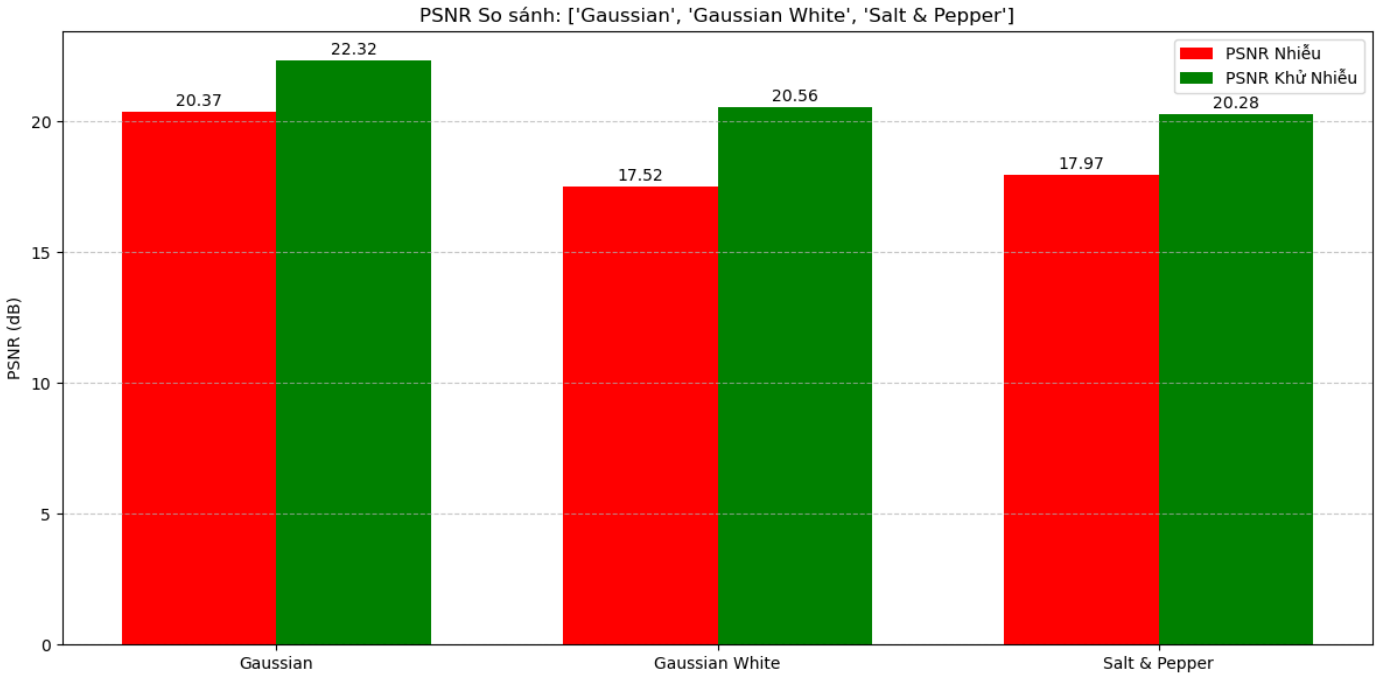


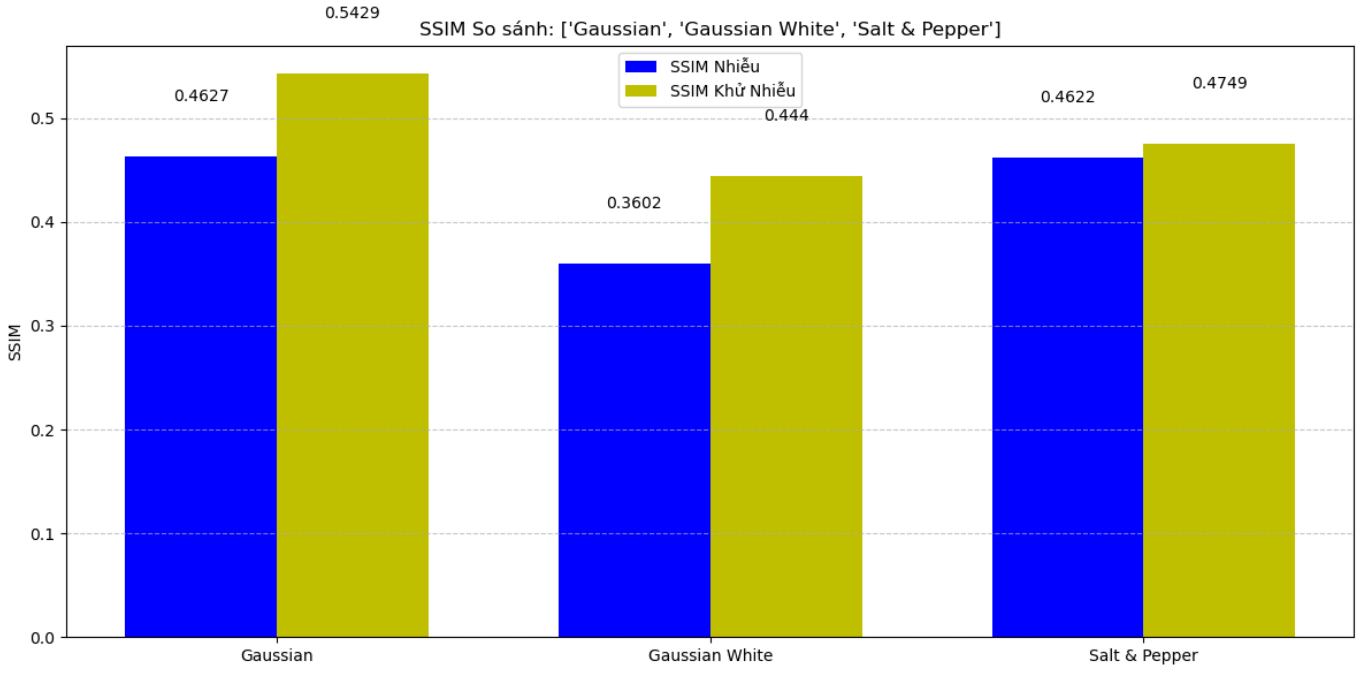
◈ Salt & Pepper Noise:

➤ PSNR (noisy): 17.97 dB | SSIM: 0.4622

➤ PSNR (denoised): 20.28 dB | SSIM: 0.4749


Original - Salt & Pepper
Noisy - Salt & Pepper
Denoised - Salt & Pepper


PSNR So sánh: ['Gaussian', 'Gaussian White', 'Salt & Pepper']
PSNR Nhiễu
PSNR Khử Nhiễu
PSNR (dB)
20.37
22.32
17.52
20.56
17.97
20.28
0
5
10
15
20
Gaussian
Gaussian White
Salt & Pepper


SSIM So sánh: ['Gaussian', 'Gaussian White', 'Salt & Pepper']
SSIM Nhiễu
SSIM Khử Nhiễu
SSIM
0.4627
0.5429
0.3602
0.444
0.4622
0.4749
0.0
0.1
0.2
0.3
0.4
0.5
Gaussian
Gaussian White
Salt & Pepper

Ảnh 2:

◈ Gaussian Noise:

➤ PSNR (noisy): 20.29 dB | SSIM: 0.2453

➤ PSNR (denoised): 25.19 dB | SSIM: 0.4422

◈ Gaussian White Noise:

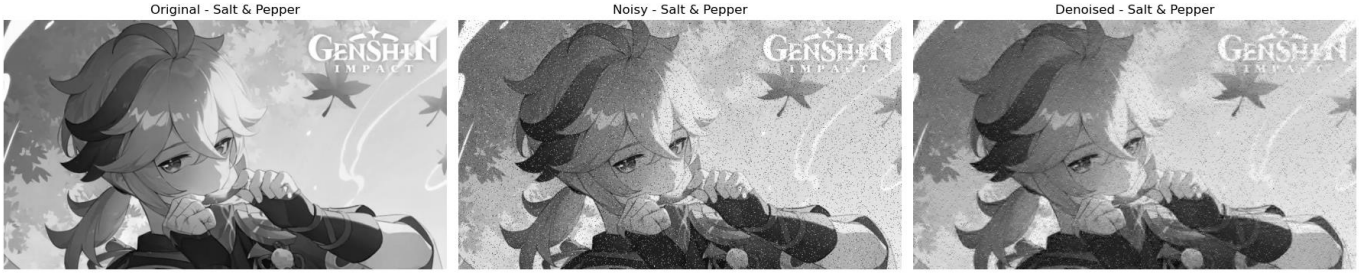
➤ PSNR (noisy): 17.56 dB | SSIM: 0.1711

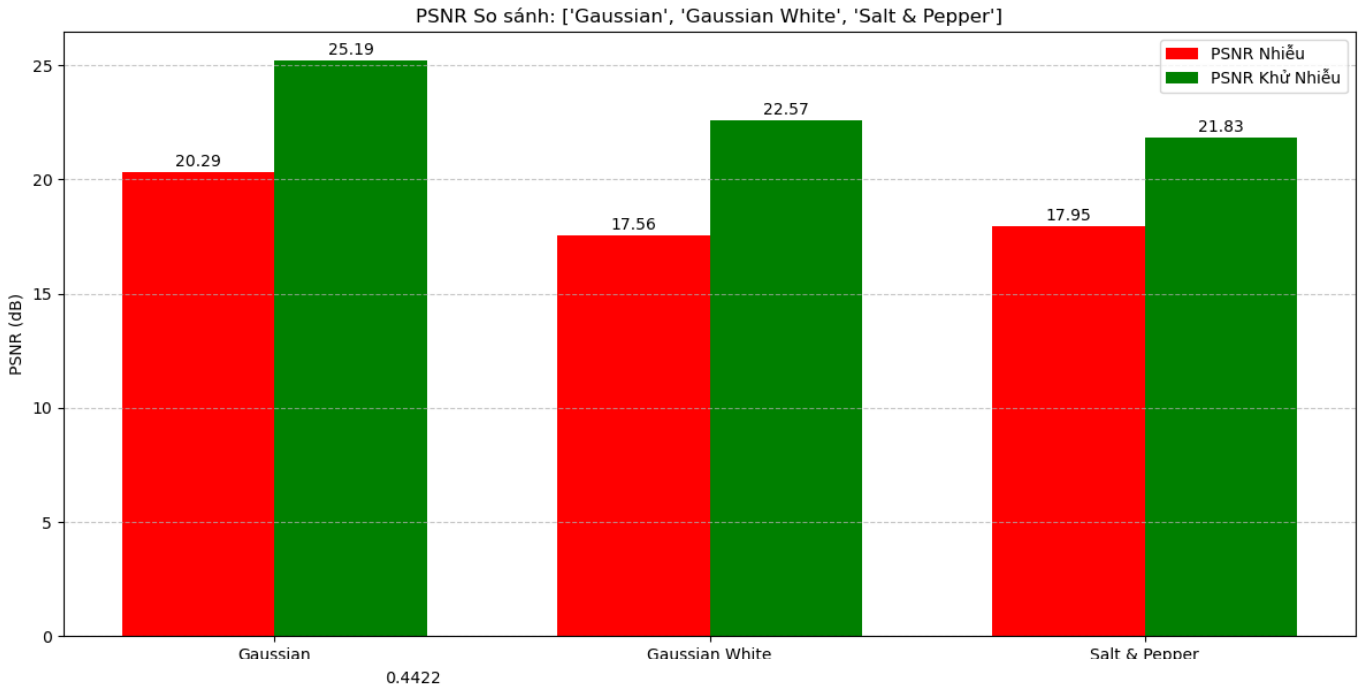
➤ PSNR (denoised): 22.57 dB | SSIM: 0.3138

◈ Salt & Pepper Noise:

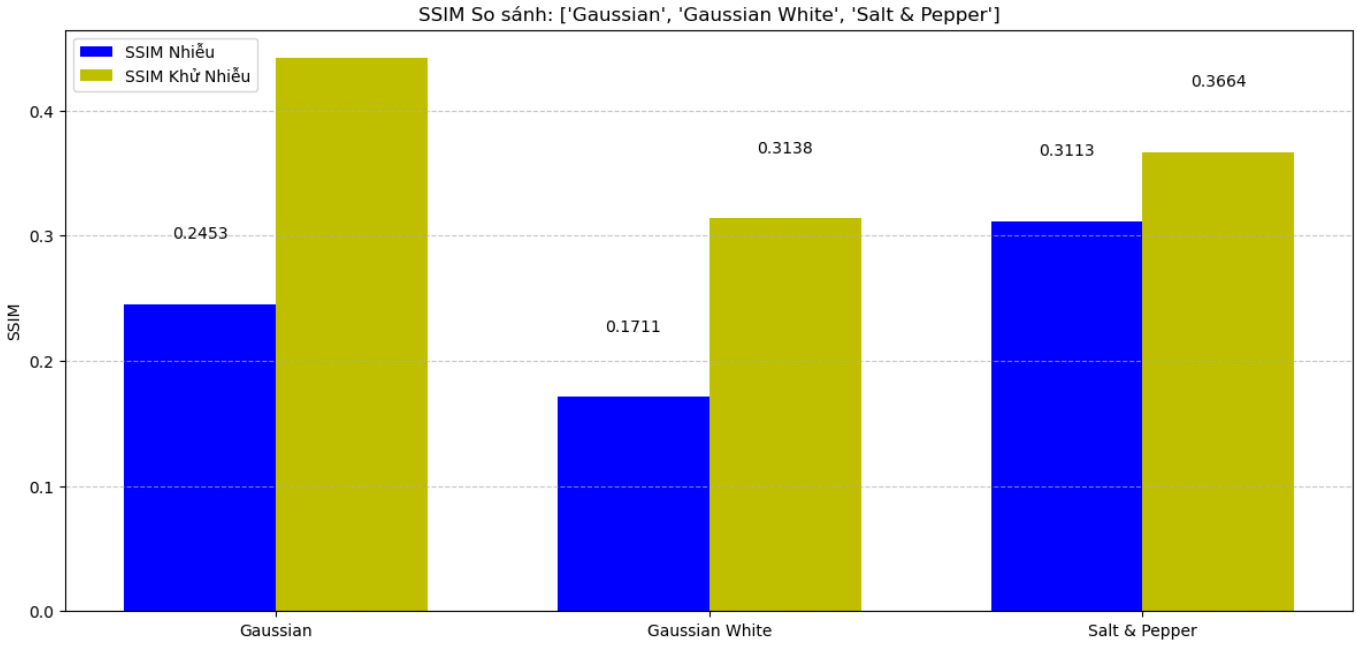
➤ PSNR (noisy): 17.95 dB | SSIM: 0.3113

➤ PSNR (denoised): 21.83 dB | SSIM: 0.3664





0.4422



5. Thảo luận

5.1. Ưu điểm của phương pháp đề xuất

1. Kết hợp DFB giúp bảo toàn tốt các đặc trưng định hướng như cạnh và đường nét

2Tự động điều chỉnh quy trình khử nhiễu dựa trên mức độ nhiễu ước lượng

3. Hoạt động tốt với cả nhiễu liên tục (Gaussian) và nhiễu rời rạc (Salt & Pepper)

5.2. Hạn chế và hướng phát triển

1. Phương pháp đòi hỏi nhiều phép tính, có thể chậm với ảnh kích thước lớn

2. Các bộ lọc định hướng hiện tại còn đơn giản, có thể mở rộng với nhiều hướng hơn

3. Một số tham số như mức phân rã wavelet, loại wavelet mẹ cần được điều chỉnh phù hợp

4. Cần cải thiện để xử lý tốt hơn các loại nhiễu phức tạp như nhiễu có tương quan

Hướng phát triển trong tương lai:

- Tối ưu hóa thuật toán để giảm độ phức tạp tính toán
- Tích hợp các bộ lọc định hướng phức tạp hơn với nhiều hướng hơn
- Kết hợp với các kỹ thuật học máy để cải thiện hiệu quả khử nhiễu
- Mở rộng phương pháp cho ảnh màu và dữ liệu đa chiều khác

5.3. So sánh với các phương pháp khác

Phương pháp đề xuất đạt được sự cân bằng tốt giữa khả năng khử nhiễu và bảo toàn chi tiết so với một số phương pháp khác:

- **Lọc truyền thống** (Mean, Median, Gaussian): Phương pháp đề xuất bảo toàn cạnh tốt hơn đáng kể
- **Wavelet thông thường**: Việc bổ sung DFB giúp cải thiện khả năng bảo toàn các đặc trưng định hướng
- **Bilateral Filtering**: Phương pháp đề xuất xử lý tốt hơn với nhiễu mạnh
- **Non-local Means**: Phương pháp đề xuất có độ phức tạp tính toán thấp hơn với chất lượng tương đương

6. Kết luận

Báo cáo này đã trình bày một phương pháp khử nhiễu ảnh kết hợp biến đổi wavelet và directional filter banks. Kết quả thực nghiệm cho thấy phương pháp đề xuất đạt được hiệu quả khử nhiễu cao trong khi vẫn bảo toàn tốt các đặc trưng quan trọng của ảnh như cạnh và kết cấu.

Phương pháp này tận dụng ưu điểm của cả hai kỹ thuật:

- Khả năng phân tích đa phân giải của biến đổi wavelet
- Khả năng bắt các đặc trưng định hướng của DFB

Hướng nghiên cứu trong tương lai sẽ tập trung vào việc tối ưu hóa thuật toán và mở rộng phương pháp cho các loại dữ liệu phức tạp hơn.

Tài liệu tham khảo

[1] J. G. Rosiles and M. J. T. Smith, “Image denoising using directional filter banks,” in *Proc. 2000 Int. Conf. on Image Processing (ICIP)*, Vancouver, BC, Canada, Sep. 2000, doi:

10.1109/ICIP.2000.899357.