Project 2 Report

Liu Yifei
2024134022
arnoliu@shanghaitech.edu.cn

1 Methods

1.1 Mean filter

The mean filter is a simple yet effective spatial filtering technique used for image denoising. It works by replacing each pixel value with the average of the neighboring pixels within a defined window (kernel). This process reduces high-frequency noise while maintaining the overall structure of the image. However, it may introduce blurring effects, especially around edges.

The mean filter is applied using a moving window of size $k \times k$, centered on each pixel of the image. The output value at each pixel location (i, j) is computed as:

$$I'(i,j) = \frac{1}{k^2} \sum_{m=-\frac{k-1}{2}}^{\frac{k-1}{2}} \sum_{n=-\frac{k-1}{2}}^{\frac{k-1}{2}} I(i+m, j+n) \tag{1}$$

where $I(i, j)$ represents the original image pixel values, and $I'(i, j)$ denotes the filtered output.

The following pseudocode outlines the implementation of the mean filter:

```
Algorithm MeanFilter(image, kernel_size):
    Input: image (2D array), kernel_size (odd integer)
    Output: filtered_image (2D array)

    1. Get image dimensions: (height, width) ← size of image
    2. Create an empty array of same size as image: filtered_image
    3. Compute padding size: pad_size ← (kernel_size - 1) / 2
    4. Pad the image using zero or edge replication
    5. For each pixel (i, j) in the image:
        a. Extract the neighborhood of size kernel_size × kernel_size centered at (i, j)
        b. Compute the mean of the neighborhood
        c. Assign the mean value to filtered_image(i, j)
    6. Return filtered_image
```

1.2 Wavelet Transform

Wavelet Transform (WT)[1] is a powerful technique for image denoising that provides multi-resolution analysis. It allows for the separation of noise from significant image structures by decomposing the image into different frequency components. Unlike traditional filtering methods, the Wavelet Transform operates at various scales, enabling selective noise removal while preserving important features like edges and textures.

The wavelet denoising algorithm can be summarized in the following steps:

1. **Wavelet Decomposition:** Apply the discrete wavelet transform (DWT) to decompose the noisy image into multiple frequency subbands.
2. **Thresholding:** Apply a thresholding method (either soft or hard) to the high-frequency coefficients to suppress noise.
3. **Wavelet Reconstruction:** Perform the inverse discrete wavelet transform (IDWT) using the thresholded coefficients to reconstruct the denoised image.

The wavelet transform of an image $I(x, y)$ at scale j and position k is expressed as:

$$W_{j,k} = \int\int I(x, y)\psi_{j,k}(x, y)\, dx\, dy$$

where $\psi_{j,k}(x, y)$ represents the wavelet basis function at scale j and position k.

For denoising, thresholding is applied to the high-frequency coefficients, resulting in:

$$W'_{j,k} = \begin{cases} 0, & \text{if } |W_{j,k}| < T \quad \text{(Hard Thresholding)} \\ \text{sign}(W_{j,k})\left(|W_{j,k}| - T\right), & \text{if } |W_{j,k}| \geq T \quad \text{(Soft Thresholding)} \end{cases}$$

where T is the threshold value, typically chosen based on methods like universal thresholding or Bayesian shrinkage.

The following pseudocode describes the process of applying the Wavelet Transform for image denoising:

```
Algorithm WaveletDenoise(image, wavelet, level, threshold_type):
    Input: image (2D array), wavelet type, decomposition level, threshold type (hard/soft)
    Output: denoised_image (2D array)

    1. Perform wavelet decomposition:
        a. Apply discrete wavelet transform (DWT) to decompose the image into subbands.
    2. Apply thresholding to high-frequency coefficients:
        a. If threshold_type is "hard":
            - Set coefficients below threshold to zero.
        b. If threshold_type is "soft":
            - Reduce coefficients' magnitude by the threshold.
    3. Reconstruct image using inverse wavelet transform (IDWT).
    4. Return the denoised image.
```

1.3 BM3D

BM3D (Block-Matching and 3D Filtering)[2] is a state-of-the-art denoising algorithm that exploits both the spatial and transform domain characteristics of images. It leverages a non-linear, collaborative filtering process based on grouping similar image patches and applying 3D transformations to suppress noise while preserving image structures such as edges and textures.

The BM3D algorithm consists of the following major steps:

1. **Block Matching:** Identify similar patches (blocks) within the noisy image. For each patch, a search window is defined where similar patches are found using a block matching criterion.
2. **3D Transform:** Stack the matching blocks into a 3D array and apply a transform (typically, a 3D discrete cosine transform (DCT) or wavelet transform) to these blocks.
3. **Thresholding and Filtering:** Apply a soft thresholding operation to the 3D transformed blocks to suppress noise. This is followed by a Wiener filter or similar non-linear filtering.

4. **Inverse 3D Transform:** Perform the inverse of the transform applied in step 2 to get back to the image domain.
5. **Aggregation:** Combine the filtered blocks back into the final denoised image, ensuring that overlapping regions are appropriately averaged to minimize boundary artifacts.

The algorithm can be mathematically formulated as follows:

Let $I_{\text{noisy}}(x, y)$ represent the noisy image and B_i denote the block of pixels that are being matched in the search window. For each block B_i, the matched set of blocks $\{B_i\}_j$ can be aggregated into a 3D data array:

$$\mathcal{B}_i = [B_i, B_j, \dots] \quad \text{(matching blocks stacked)}$$

Next, a transform is applied:

$$\mathcal{T} = \mathcal{F}(\mathcal{B}_i) \quad \text{(3D transform, e.g., DCT)}$$

The thresholding process is applied to the transformed coefficients $\mathcal{T}$:

$$\mathcal{T}' = \mathcal{T} - T \quad \text{(Soft thresholding of transform coefficients)}$$

Finally, after inverse transformation, the filtered patches are recombined to yield the denoised image I_{denoised}:

$$I_{\text{denoised}} = \text{Reconstruct}(\mathcal{T}', \mathcal{B}_i)$$

The pseudocode for BM3D denoising is as follows:

```
Algorithm BM3DDenoise(image, block_size, search_window, threshold_type):
    Input: image (2D array), block size, search window, threshold type (soft)
    Output: denoised_image (2D array)

    1. Perform block matching:
        a. For each block in the image, find similar patches within the search window.
    2. Stack the matching blocks into a 3D array.
    3. Apply 3D transform (DCT, wavelet) to the stacked blocks.
    4. Apply thresholding to the transformed coefficients:
        a. If threshold_type is "soft":
            - Reduce the magnitude of coefficients based on threshold value.
    5. Apply inverse transform to the filtered blocks.
    6. Reconstruct the denoised image from the filtered patches.
    7. Return the denoised image.
```

2 Barbara

Gaussion noise

(a) mean=0, std=0.01

(b) mean=0, std=0.05

Speckle noise

(a) std=0.01

(b) std=0.05

3 Boat

Gaussion noise

(a) mean=0, std=0.01

(b) mean=0, std=0.05

Speckle noise

(a) std=0.01

(b) std=0.05

4 Discussion

In this study, various image denoising techniques, namely the Mean Filter, Wavelet Transform (WT), and Block-Matching and 3D Filtering (BM3D), were evaluated. The performance of these methods was compared in terms of their ability to reduce noise while maintaining the crucial information of the image, such as edges and textures.

First, the Mean Filter was found to be the simplest yet the most basic denoising method. This filter works by averaging the pixel values within a local window. While effective at reducing high-frequency

noise, it also significantly blurs edges and fine details. This loss of important image features makes the Mean Filter less suitable for preserving the structural integrity of images.

Next, the Wavelet Transform filter performs better than the Mean Filter in terms of denoising performance. By decomposing the image into different frequency components, the WT filter allows for a more selective removal of noise. However, similar to the Mean Filter, it still suffers from the degradation of critical image details, especially for high-frequency components like edges. While it effectively reduces the noise, the soft or hard thresholding techniques can blur or suppress key features of the image, leading to a less clear and less sharp result.

Finally, BM3D, which is a state-of-the-art denoising algorithm, outperforms both the Mean Filter and Wavelet Transform in terms of both noise reduction and image clarity. The strength of BM3D lies in its ability to group similar image patches (blocks) and apply 3D filtering, which allows it to preserve more of the image's fine details while effectively removing noise. The process of transforming the blocks into the 3D domain and applying collaborative filtering before aggregating them back into the image domain results in a much cleaner image with minimal loss of important information. Additionally, the BM3D algorithm adaptively handles varying levels of noise across different parts of the image, offering a more refined denoising process. This adaptive filtering mechanism makes it more effective than traditional methods like the Mean Filter and Wavelet Transform.

The success of BM3D can be attributed to its advanced approach of using block matching and 3D filtering in the transform domain. This method preserves edges and texture much better than conventional spatial filters, and its ability to deal with high-frequency noise without compromising important image features is what places it at the forefront of denoising techniques. The use of thresholding techniques in the transform domain enables BM3D to suppress noise while retaining the critical structures that define the visual quality of an image.

In conclusion, while the Mean Filter and Wavelet Transform can serve as basic denoising tools, they fall short in terms of preserving image quality. BM3D, with its advanced block-matching and collaborative filtering techniques, provides superior performance, making it the state-of-the-art method for image denoising.

5 Bonus: KSVD

Image denoising can also be approached by using sparse and redundant representations over learned dictionaries. One such method is the K-SVD (K-Singular Value Decomposition) algorithm [3], which aims to represent image patches in a sparse form using a dictionary learned from the noisy data. K-SVD learns the dictionary iteratively by finding the best sparse representation of the image patches with respect to a dictionary and updating the dictionary atoms to improve the representation.

The general idea behind K-SVD-based denoising is to express the noisy image as a sparse linear combination of atoms from a learned dictionary. The noisy patches are first represented as sparse vectors, and the dictionary atoms are updated iteratively to minimize the reconstruction error. In this process, the noise tends to be discarded, leaving behind the important structures and features of the image. This method is particularly useful in situations where the image contains redundant structures or where natural image patches can be represented sparsely in a learned basis.

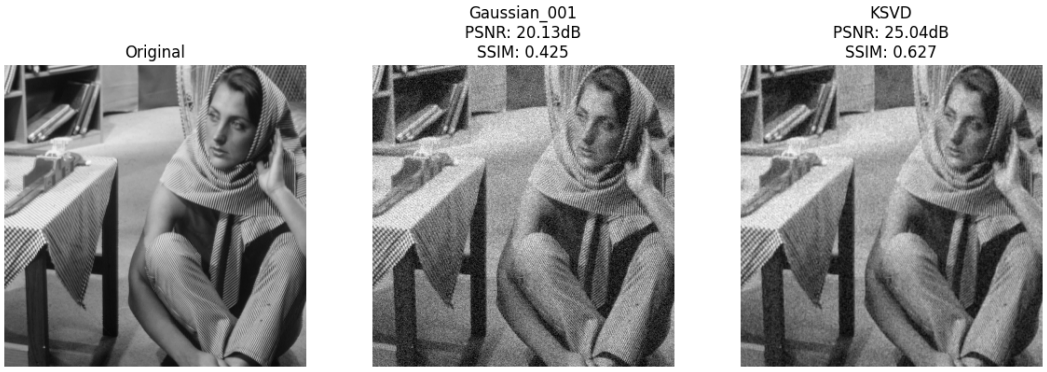
The key steps of the K-SVD algorithm for image denoising include:

1. **Patch Extraction:** Extract overlapping patches from the noisy image.
2. **Sparse Representation:** Represent each patch as a sparse linear combination of dictionary atoms.
3. **Dictionary Learning:** Learn the dictionary iteratively by updating its atoms to minimize the reconstruction error of the sparse representations.
4. **Denoising:** Reconstruct the denoised image by combining the sparsely represented patches.

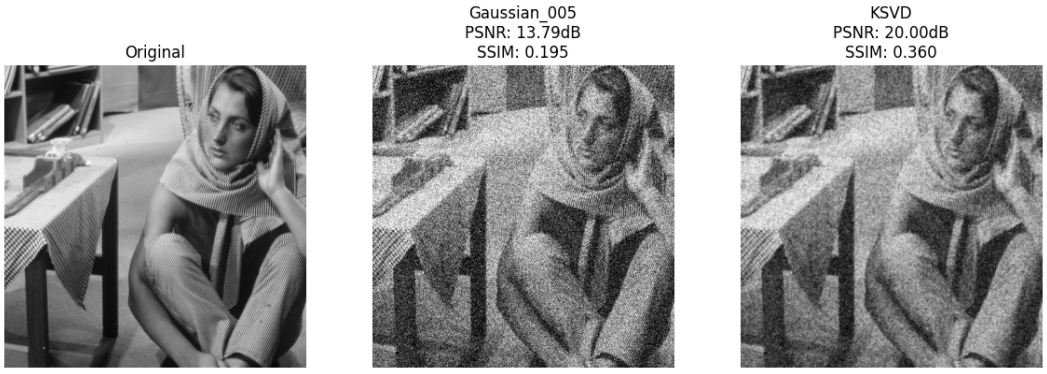
This method has the advantage of being able to learn an image-specific dictionary, making it highly effective in capturing the image's intrinsic structures. However, the computational complexity of dictionary learning can be high, especially for large images, as it requires solving optimization problems at each iteration. Despite these challenges, KSVD-based denoising is still widely used due to its ability to achieve excellent denoising results, particularly in images with significant structural redundancy.

5.1 Results of barbara

Gaussion noise



(a) mean=0, std=0.01



(b) mean=0, std=0.05

Speckle noise

(a) std=0.01

(b) std=0.05

5.2 Results of boat

Gaussion noise

(a) mean=0, std=0.01

(b) mean=0, std=0.05

Speckle noise

(a) std=0.01

(b) std=0.05

5.3 Results of boats

References

[1] S Grace Chang, Bin Yu, and Martin Vetterli. Adaptive wavelet thresholding for image denoising and compression. *IEEE transactions on image processing*, 9(9):1532–1546, 2000.

[2] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, 16(8):2080–2095, 2007.

[3] Michael Elad and Michal Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, 15(12):3736–3745, 2006.