A TWO STAGE IMAGE DENOISING APPROACH BASED ON SUPERPIXEL ALGORITHM

A Project report submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

T . Yaswanth Ganesh(319126512120) P. L. Madhumitha(319126512108)

E. Manikanta Sai(319126512080) T. Jagadeeswar(319126512119)

Under the guidance of

Dr.J.Bhaskara Rao

Associate Professor

Department of ECE

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(*Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC*)

Sangivalasa, bheemili mandal, visakhapatnam Dist.(A.P)

2022-2023

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA&NAAC)

Sangivalasa, Bheemili mandal, Visakhapatnam Dist.(A.P)

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled **"A TWO STAGE IMAGE DENOISING APPROACH BASED ON SUPERPIXEL ALGORITHM"** submitted by **T . Yaswanth Ganesh (319126512120) , P. L. Madhumitha (319126512108), E .Manikanta Sai (319126512080) , T. Jagadeeswar (319126512119)** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Electronics & Communication Engineering of Anil Neerukonda Institute of technology and Sciences(A), Visakhapatnam is a record of bonafide work carried out under my guidance and supervision.

Project Guide

Dr.J.Bhaskara Rao

Associate Professor

Department of E.C.E

ANITS

Associate Professor
Department of E.C.E.
Anil Neerukonda
Institute of Technology & Sciences
Sangivalasa, Visakhapatnam-531 162

Head of the Department

Dr. B.Jagadeesh

Professor and HOD

Department of E.C.E

ANITS

Head of the Department
Department of E C E
Anil Neerukonda Institute of Technology & Sciences
Sangivalasa - 531

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our project guide **Dr. J. Bhaskara Rao, Associate Professor,** Department of Electronics and Communication Engineering, ANITS, for his guidance with unsurpassed knowledge and immense encouragement. We are grateful to **Dr. B. Jagadeesh**, Head of the Department, Electronics and Communication Engineering, for providing us with the required facilities for the completion of the project work.

We are very much thankful to the **Principal and Management, ANITS, Sangivalasa,** for their encouragement and cooperation to carry out this work.

We express our thanks to all **Teaching faculty** of Department of ECE, whose suggestions during reviews helped us in accomplishment of our project. We would like to thank **all Non-Teaching staff** of the Department of ECE, ANITS for providing great assistance in accomplishment of our project.

We would like to thank our parents, friends, and classmates for their encouragement throughout our project period. At last but not the least, we thank everyone for supporting us directly or indirectly in completing this project successfully.

PROJECT STUDENTS

T . Yaswanth Ganesh (319126512120)

P. L. Madhumitha (319126512108)

E.Manikanta Sai (319126512080)

T . Jagadeeswar(319126512119)

ABSTRACT

Most denoising algorithms presently lack the capability to differentiate between pixels that contain noise and those that do not. However, these algorithms still utilize uniform rules when processing all pixels. When used on photographs with tiny details or low contrast between the subject and the backdrop, denoising techniques frequently result in the loss of original image information. The research suggests a two-stage noise localization-based picture denoising technique to address the aforementioned issues. Thresholds T'1 and T'2 are created in the first step using the distribution of grey values in the picture. To achieve denoising, the method involves two stages. Initially, edge extraction is applied to preserve the edge information, followed by the use of singular value decomposition to obtain the singular value matrix from the resulting edgeless grayscale image. Finally, the singular value matrix is compressed using a percentage threshold η. To enhance image denoising, the proposed algorithm involves a two-stage approach. In the first stage, thresholds T'1 and T'2 are derived by analyzing the distribution of gray values in the image. The second stage involves creating an edgeless grayscale image by employing edge extraction to remove and save the image edge information. The edgeless picture is then subjected to singular value decomposition to produce the singular value matrix, which is then compressed using a percentage threshold to achieve efficient denoising. To perform coarse noise filtering, the proposed approach employs inverse matrix decomposition. Moreover, the algorithm utilizes adaptive thresholds T1 and T2, which are obtained from the image histogram. The image is categorized into three regions, namely "Dark Area," "Gray Area," and "Light Area" based on these thresholds. The final step in the proposed approach involves merging the denoised image with the image edges to produce the output. A comparative analysis of the proposed algorithm is performed by examining the peak signal-to-noise ratio (PSNR) and processing time against other state-of-the-art denoising algorithms. The results demonstrate that the proposed algorithm is more effective in denoising various images

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

Symbol	Title
T1	Adaptive threshold Value of range [129,255]
T2	Adaptive threshold Value of range [1,128]
η	Percentage threshold in SVD
α	Threshold between [24, 36] in final filtering

LIST OF ABBREVATIONS

CNN	convolutional neural networks
MSE	Mean Squared Error
SVD	Singular value decomposition
SLIC	Simple Linear Clustering algorithm
PSNR	Peak signal-to-noise ratio
SSIM	Structural Similarity Index
IEF	Image Enhancement Factor

CHAPTER 1

INTRODUCTION

CHAPTER 1
INTRODUCTION

1.1 Project Objective

Modern image processing methods, such as deep cascade neural networks, face hallucination techniques, multi-gait recognition, etc., all heavily rely on picture denoising. Image denoising is the process of taking away noise from a picture so that the original image can be seen. Denoised photos may unavoidably lose some information since noise, edge, and texture are high frequency components that are difficult to differentiate throughout the denoising process. The filter denoising method, matrix sparse denoising algorithm, and threshold denoising algorithm are all examples of image denoising techniques. The wavelet filtering denoising method, the mean filter denoising algorithm, the median filter denoising algorithm, and the Wiener filtering denoising algorithm are the primary components of the filter denoising algorithm.

Different image processing and computer vision issues frequently include the relevant topic of picture denoising. The key to a successful picture denoising technique is that it must preserve edges while removing noise as completely as feasible. None of the denoising algorithms can precisely pinpoint where the picture noise is located, and a significant amount of the original image information is lost. The filter denoising method is pre-processed to acquire the contour in the picture, which eliminates the drawback of employing it to make images fuzzy. Multi-stage picture denoising is recommended in order to address the aforementioned issues. Here, the texture and edge are safeguarded. Using the super-pixel technique and singular value decomposition, we introduced the two-stage picture denoising.

Fig 1.1 Noisy image to denoised image

1.2 LITERATURE SURVEY

Yongxia Zhang, Xuemei Li, Xifeng Gao, Caiming Zhang has proposed "A Simple Algorithm of Superpixel Segmentation With Boundary Constraint":

Superpixel has been utilised frequently as supporting regions for primitives to minimise computations in a variety of computer vision tasks since it is one of the most popular picture over segmentations. The approach used in this research to segment superpixels is based on a distance function that aims to balance the features of the resulting superpixels' boundary adherence, intensity uniformity, and compactness (COM). This approach starts by initialising the superpixel seed positions to obtain the initial labels of pixels, given an expected number of superpixels. Then, using the specified distance measurement to iteratively optimise the superpixels, update the positions and intensities of the superpixel seeds using the three-sigma rule. The experimental results show that our technique achieves a comparable balance between superpixel COM and adherence to object boundaries and is more efficient and accurate than prior superpixel methods.

Amarjit R, Hussain LR has proposed "Fuzzy SVM based fuzzy adaptive filter for denoising impulse noise from color images":

This paper proposes a fuzzy adaptive filter based on fuzzy SVM. Calculate the vector distance between the centre feature vector and the other feature vectors in the cluster using this approach. Create a new cluster out of the features that are closer together than the threshold distance, and then record the matching fuzzy membership degree for each feature vector. For each cluster, repeat this. When removing impulse noise from colour photographs, noisy and non-noisy pixels are classified here. This approach can be used to remove impulse noise from photos that have both fixed valued and random valued impulse noise.

Oleh Misko, Valerii Veseliak has proposed "Singular Value Decomposition in image noise filtering":

Modern systems utilise a huge number of photos. Noise frequently muddles these photos. To keep as much information while removing as much noise as feasible is a crucial issue for many fields. The essential point is that one of the best filters, k-SVD filtering, effectively removes noise from an image while maintaining good quality and recognition. By comparing many methods, k-SVD emerged as one of the most effective and suitable to various image types. The suggested method can be applied to a variety of fields and paired with one of the examined filters to improve performance.

Li B, Liu QS, Xu JW et al has proposed "A new method for removing mixed noises":

The trilateral filter now performs substantially better at removing mixed noises. It removes Gaussian noises as effectively as the non-local means filter and random impulse noises as effectively as the trilateral filter. They started by introducing the similarity principle, which provides a straightforward mathematical explanation for the non-local means filter modified to exclude Gaussian disturbances. A novel filter termed the mixed noise filter (MNF), which is based on this idea and employs an impulse-controlled weighted norm, is then suggested to eliminate the mixing of Gaussian and random impulse noise.

Umam AK, Yunus M has propose "Quaternion wavelet transform for image denoising":

Using a magnitude and three angle phases, QWT can be shown in polar form. The magnitude and angle phases of a quaternion can be condensed into a polar form. QWT effectively reduces image noise. The quaternion wavelet transform (QWT) uses the same base as the quaternion Fourier transform (QFT), but it focuses on scaling and translation. To lower detail coefficient magnitudes, use the hard thresholding approach with VisuShrink. Use a mean filter to approximate the magnitude of the coefficient as well.By using the quaternion wavelet transform in reverse, reconstruct the denoised (IQWT).

Luo L, Zhao ZQ, Li XP et al has proposed "A stochastic image denoising method based on adaptive patch-size":

An adaptive patch-size-based stochastic picture denoising technique. By selecting the best nonlocal comparable patch size for each individual image site, the quality of the restored image is enhanced. The process consists of two steps. In the initial stage, related patches are found using an adaptive patch size. Using the similar picture patches found in the first phase, the denoising algorithm is designed in the second phase

1.3 PROJECT OUTLINE

We cover image denoising pre-processing in chapter 2 because it's necessary to get a better denoising impact. In chapter 3, the noisy image is initially decomposed by SVD before being coarsely filtered using the percentage threshold. Later, to achieve noise fine filtering and regarding PSNR in chapter 4, the superpixel-like technique is utilised. Compared to the other denoising technique, which is based on Peak Signal to Noise Ratio, in chapter 5, (PSNR).

CHAPTER 2

IMAGE DENOISING

Chapter 2
Image Denoising

2.1 Introduction

Image denoising is a process of removing noise from an image to restore its original content. In modern image processing systems, it plays an important role in improving the quality of images for various applications such as deep cascade of neural networks, face hallucination techniques, and multi-gait identification, among others. Noise in images can be introduced by various sources such as poor lighting conditions, electronic sensors, or compression artifacts. The goal of image denoising is to remove this noise while preserving the true image content, including edges, textures, and other important features. However, this is a challenging task as noise, edges, and textures are high frequency components and can be difficult to distinguish during the denoising process. To tackle this challenge, various image denoising methods have been proposed, including traditional methods such as median filtering, wavelet thresholding, and nonlocal means filtering, as well as deep learning-based methods such as convolutional neural networks (CNNs). Deep learning-based methods have shown remarkable performance in image denoising by leveraging the power of large training datasets and complex models. Despite the progress in image denoising, there are still some challenges that need to be addressed, such as preserving the fine details in the images and dealing with complex noise patterns. In addition, trade-offs between computational efficiency and denoising performance must also be considered, especially for real-time applications. Nevertheless, image denoising remains an active area of research and is expected to continue to advance in the coming years.

2.2 Applications of Image Denoising in Modern Image Processing System

2.2.1 Deep Cascade of Neural Networks

Deep cascade of neural networks is a powerful approach in computer vision and machine learning for processing images. The idea behind this approach is to use multiple neural networks that are stacked and trained one after the other in a cascading manner. By doing so, each subsequent network in the cascade can refine the output of the previous network, leading to the extraction of more complex features from images and improved accuracy compared to single-layer neural networks. The deep cascade of neural networks approach consists of several stages. In the first stage, the input image is passed through a network that is trained to extract simple features such as edges and shapes. This network provides the first level of abstraction for the image, breaking it down into simpler components. The output of this network is then passed to the next network in the cascade, which is trained to extract more complex features, such as textures and patterns. This process continues, with each subsequent network in the cascade being trained to extract increasingly complex features from the image. The final output of the last network in the cascade is the final result of the image processing task. This approach allows for the extraction of complex features from images that may not be easily extractable with a single-layer network.

One of the key advantages of the deep cascade of neural networks approach is its ability to handle large amounts of data. This is because the deep structure of the cascade allows for the progressive abstraction of features, breaking down complex images into simpler components and allowing the network to focus on specific aspects of the image. Another advantage is that the deep cascade of neural networks approach is highly scalable, as new networks can be added to the cascade to handle more complex features or to improve accuracy. This also makes the approach highly adaptable to new data and new tasks, as the cascade can be easily modified to handle new types of images and new processing requirements. In addition to image processing, the deep cascade of neural networks approach has also been applied to other computer vision tasks, such as object detection, image segmentation, and image classification. These tasks benefit from the same advantages as image processing, including improved accuracy and scalability, making the deep cascade of neural networks a versatile and powerful approach for a wide range of computer vision applications.

In conclusion, the deep cascade of neural networks approach is a powerful method for processing images in computer vision and machine learning. By using multiple neural networks that are trained one after the other in a cascading manner, this approach allows for the extraction of complex features from images, leading to improved accuracy compared to single-layer neural networks. With its scalability and adaptability, the deep cascade of neural networks approach is a promising approach for a wide range of computer vision tasks and applications.

2.2.2 The Face Hallucination Technique

A computer vision approach known as "facial hallucination" is used to create high-resolution images of faces from low-resolution inputs. This method is especially helpful in applications like face identification and recognition, where only low-resolution face photographs are accessible, despite the desire for high-resolution images. The face hallucination technique aims to produce high-resolution face images that are as close to the real high-resolution photos as is practical. Deep convolutional neural networks (DCNNs), a class of neural networks that have shown promise in a number of image processing tasks, form the foundation of the face hallucination technique. The DCNN utilised in face hallucination learns the mapping between low-resolution and high-resolution face images through training on a sizable dataset of faces. Then, using this mapping, low-resolution inputs are used to create high-resolution facial images.When the DCNN is being trained, low-resolution face images are fed into the network, and the output is compared to the matching high-resolution face images. The DCNN modifies its settings to make the output of the network as comparable to the high-resolution facial photos as is humanly practical. The training procedure is repeatedly carried out, and the DCNN's capacity to produce high-resolution facial images from low-resolution inputs keeps getting better. The trade-off between accuracy and processing complexity is one of the main issues in face hallucination. On the one hand, producing high-resolution photographs that are as identical to the real high-resolution photos as feasible is ideal. Nevertheless, producing high-resolution photographs necessitates a lot of computer power and might take a long time. Researchers have proposed a number of methods to overcome this difficulty by lowering the computational complexity of the face hallucination process while keeping a high level of accuracy.

2.2.3 Multi-Gait Identification Technique

Multi-gait identification is a process in which an individual is identified based on the way they walk. This approach uses computer vision techniques to analyze video or depth sensor data and extract features related to gait. Multi-gait identification is a promising area of research for applications such as surveillance, human-computer interaction, and biometric authentication. Gait is a highly individualistic trait and can be used as a biometric characteristic for identification purposes. It is a dynamic and behavioral characteristic that is unique to each individual, and it can be captured through video or depth sensor data. The use of gait as a biometric characteristic has several advantages over other biometric traits such as fingerprint recognition or facial recognition. For example, gait can be captured from a distance, it is less intrusive than other biometric traits, and it is less affected by changes in appearance, such as a change in clothing or hairstyle.

In multi-gait identification, the first step is to capture gait data through video or depth sensor data. The captured gait data is then processed to extract features related to gait. These features may include gait speed, stride length, gait symmetry, and other gait-related characteristics. Once the features have been extracted, they are then used to identify individuals. The process of multi-gait identification involves training a machine learning model on a large dataset of gait data. This dataset is used to train the model to learn the relationship between the gait features and the identity of the individual. Once the model has been trained, it can then be used to identify individuals based on their gait features. The machine learning model used for multi-gait identification may be a support vector machine, a decision tree, or a deep neural network. One of the key challenges in multi-gait identification is to extract robust and reliable gait features from the gait data. Gait features are often affected by factors such as clothing, footwear, and surface type, and it is important to account for these factors when extracting gait features. To address this challenge, researchers have proposed various techniques for normalizing the gait data and removing the effects of extraneous factors.

In conclusion, multi-gait identification is a promising area of research for applications such as surveillance, human-computer interaction, and biometric authentication. By using computer vision techniques to analyze video or depth sensor data and extract gait features, multi-gait identification is able to identify individuals based on their gait, even when they are partially occluded or have changed their appearance. With the continued development of computer vision and machine learning techniques, multi-gait identification has the potential to become an important tool in a variety of applications.

2.3 Grey Scale Imaging

A method known as "grey scale imaging" uses no colour information and instead portrays a picture in shades of grey ranging from white to black. the reason for using grey scale Imaging is the process of simplifying and making an image easier to process. In recent years, there has been a rise in interest in the use of grey scale imaging for a variety of purposes, including remote sensing, computer vision, and medical imaging. Nevertheless, noise frequently affects grey scale photographs, leading to blurring, the loss of details, and inaccurate interpretations of the image's content. Many denoising methods have been suggested and described in the literature as solutions to these problems. The goal of denoising algorithms is to eliminate or significantly reduce the noise in a grayscale image while maintaining the image's original quality. Popular denoising methods like wavelets, sparsity, and total variation are based on mathematical models and try to keep the details of the image while removing the noise. Other machine learning-based methods, including deep learning and neural networks, require a lot of training data to figure out how to remove noise from photos. Denoising algorithms have improved, however there is always room for development in terms of their functionality**.** The kind and severity of the noise, the content of the image, and the intended level of detail preservation all affect how well a denoising algorithm works. Certain denoising methods might work well with some kinds of photos but not with others. Moreover, some algorithms could be excessively complicated or demand a lot of computer power to implement in real-time applications.

In summary, the field of grey scale image denoising is still in its infancy, and there is still a sizable window for achieving the requisite level of application. To efficiently remove noise from grey scale photographs while keeping their quality and details, new algorithms and procedures must be developed.

2.4 Image Denoising Algorithms

Image denoising is a fundamental task in image processing that involves removing noise from images. There are various algorithms that can be used to achieve this goal, but three of the most popular methods are filter denoising, matrix sparse denoising, and threshold denoising. In this article, we will delve into these algorithms to understand their workings.

Image denoising algorithms can be implemented using various techniques. These include filter denoising, matrix sparse denoising, and threshold denoising algorithms. Each algorithm has its advantages and disadvantages and is suitable for different types of noise in images. Therefore, it is essential to choose the appropriate algorithm based on the image's characteristics to achieve the best results.

2.4.1 Filter Denoising Algorithm

The filter denoising algorithm includes various filters, including the median filter, mean filter, wavelet filtering, and Wiener filtering.

1. **Median filter denoising algorithm**: The median filter is a nonlinear filter that substitutes the neighbourhood median value for the pixel values. It works well for eliminating impulsive noise, such as salt and pepper noise. By moving a window over the image and calculating the median value of the pixel values in the window, the filter is applied. The median value is then used to replace the centre pixel. The median filter keeps the image's edges, textures, and minute features.
2. **Mean filter denoising algorithm**: The mean filter is a linear filter that substitutes the average value of the neighbourhood for the pixel values. It is a powerful technique for taking out Gaussian noise. By moving a window over the image and calculating the average value of the pixel values in the window, the filter is applied. The average value is then used to replace the centre pixel. The image is smoothed and has less detail thanks to the mean filter.

3. **Wavelet filtering denoising algorithm**: The wavelet filtering denoising algorithm uses the wavelet transform to divide the image into various frequency components, and then independently removes noise from each component. The removal of noise while maintaining image details and edges is accomplished by wavelet filtering. In order to remove the noise, the programme thresholds the wavelet coefficients. The noise is often found in the high-frequency scales after the wavelet transform divides the image into various scales. The noise is eliminated by setting the high-frequency coefficients to zero and using a thresholding function.
4. **Wiener filtering denoising algorithm**: The Wiener filter is a linear filter that applies a filter to reduce noise by estimating the power spectra of the signal and noise. It is a good way to get rid of Gaussian noise while keeping image features. The least mean square error theory serves as the foundation for the Wiener filter. The algorithm computes the Wiener filter and estimates the power spectra of the noise and the signal. The image is then processed to eliminate the noise.

A fuzzy SVM-based fuzzy adaptive filter is suggested in the study by Amarjit R and Hussain LR to remove impulsive noise from colour images. The suggested approach seeks to overcome the shortcomings of conventional denoising filters, particularly when impulsive noise is present. The authors first apply a fuzzy adaptive filter to eliminate the noise after using a fuzzy SVM to locate the noisy pixels. A series of colour photos were used to evaluate the suggested approach, and the denoising results were encouraging.2019 saw the publication of the piece in the journal Multimedia Tools and Applications.

A similar denoising technique is used by both the mean filter denoising method and the median filter denoising algorithm. The mean value of the window's surrounding pixels replaces the window's central pixel in the mean filter method. The image is then totally covered by this window after being moved over it.. However, similar to the median filter, it can be difficult to determine the appropriate denoising extent for the mean filter algorithm. This is because both methods smooth the image, which can result in a loss of image details, and may not remove all the noise. The denoising extent needs to be carefully selected to balance noise removal and preservation of image features. It is clearly described in the article by Konieczka, Balcerek, and Dąbrowski who proposes an iterative average filtering method for image denoising. The proposed method applies an average filter to the noisy image, and then iteratively updates the filtered image by subtracting a fraction of the difference between the original image and the filtered image. This process is repeated until a stopping criterion is met. The authors evaluate the proposed method on a set of test images and compare it to other denoising methods. The experimental results demonstrate that the proposed iterative average filtering method achieves competitive denoising performance in terms of both visual quality and quantitative metrics. The article was published in the book "Signal Processing: Algorithms. Architectures, Arrangements, and Applications" in 2013.

In their article, Kumar, Panigrahy, and Sahu suggested employing empirical mode decomposition (EMD) and a non-local mean (NLM) technique to denoise electrocardiogram (ECG) signals. The suggested approach entails employing EMD to break down the noisy ECG signal into intrinsic mode functions (IMFs), which are then each subjected to the NLM methodology to eliminate noise. The suggested method is assessed by the authors using a collection of fake and real ECG signals, and it is contrasted with various denoising techniques. The experimental findings show that, in terms of denoising performance and preservation of the ECG signal characteristics, the proposed EMDNLM approach works better than previous methods. 2018 saw the publication of the piece in the journal Biocybernetics and Biomedical Engineering. A fresh technique was put out by Li, Liu, Xu, and others to eliminate mixed noise from signals. To isolate the various types of noise in a signal, the suggested method combines sparse representation and non-negative matrix factorization. The suggested method is assessed by the authors using a set of synthetic and real-world signals, and it is contrasted with previous denoising methods. The experimental results demonstrate that the proposed method achieves competitive denoising performance and outperforms other methods in terms of both visual quality and quantitative metrics. The article was published in the journal Science China (Information Sciences) in 2011.

In the wavelet denoising algorithm, the noise signal features are initially extracted, followed by the removal of noise information through low-pass filtering, and ultimately, the signal is reconstructed to produce the denoised image. The article by Umam and Yunus proposes a denoising technique for images using quaternion wavelet transform. The proposed method involves decomposing the image into real and imaginary parts using quaternion wavelet transform, followed by applying a thresholding technique to remove the noise. The authors evaluate the proposed method on a set of synthetic and real-world images and compare it to other denoising methods. The experimental results demonstrate that the proposed quaternion wavelet transform method outperforms other methods in terms of both visual quality and quantitative metrics. The article was published in the Journal of Physics: Conference Series in 2018.The BM3D algorithm is a denoising technique that employs Wiener filtering. It is regarded as one of the best denoising algorithms due to its high-quality results. The algorithm involves two main steps. The image is first estimated simply, and using block matching, it is then grouped into three dimensions. In this stage, a hard threshold calculation is used to estimate the value of each pixel point, and the corresponding position of each image block is returned. By grouping the image blocks in the second stage, the final denoising is completed. Using the noise image similar block matrix and the basic estimated image similar block matrix, two matrices are obtained in this step. These matrices are then subjected to Wiener filtering. The image chunks are reaggregated to create the final denoising effect. In order to reduce the mean square error (MSE) between the true image and the estimated image, Hasan M used a wiener filter that emphasises maximising structural similarity (SSIM). The outcomes of their tests show that, regardless of the kind of image being processed, this method exceeds BM3D in terms of denoising performance. Ali Abdullah Yahya et al. proposed an adaptive filtering technique to replace the traditional hardthresholding method in the BM3D denoising algorithm. They also introduced an Adaptive Weight Function (AWF) based on spatial distance, PSNR, and SSIM indices to maintain significant image information while filtering out noise. Despite its good denoising ability, BM3D often causes severe edge distortion due to its three-dimensional transformation process, which can result in the loss of some image details.

2.4.2 The Matrix Sparse Algorithm

Matrix sparse denoising algorithms are used to remove noise from a signal that can be represented in a matrix form, such as an image or a video. These algorithms are based on the assumption that the signal can be expressed as a linear combination of a few dominant features or patterns, where the coefficients of the combination are sparse (i.e., most coefficients are zero). The algorithm aims to estimate the sparse coefficients that best represent the signal by exploiting the sparsity property. These coefficients are then used to reconstruct a denoised version of the signal that preserves the significant features while filtering out the noise. There are various matrix sparse denoising algorithms, such as the BM3D algorithm, the K-SVD algorithm, and the nonlocal means algorithm. These algorithms use different approaches to estimate the sparse coefficients, such as dictionary learning, non-local similarity, or collaborative filtering. The denoising performance of these algorithms depends on several factors, such as the choice of the sparse representation, the noise level, and the regularization parameters. The effectiveness of the algorithm can be evaluated using metrics such as peak signalto-noise ratio (PSNR) and structural similarity index (SSIM). The paper by Zhang et al. proposes a global sparse gradient-based coupled system for image denoising. The proposed method combines the sparse gradient and coupled partial differential equations (PDEs) to preserve the image edges while removing the noise. The algorithm consists of three main steps. First, an initial denoised image is obtained using the sparse gradient approach, which exploits the sparse representation of the image gradients in a transform domain. The resulting image is then used as the input for the second step, which involves solving a coupled system of PDEs that encourage edge preservation and smoothing of the image. The system of PDEs is formulated using the joint total variation (JTV) regularization term, which couples the gradient of the image and the gradient of the PDE solution.

Finally, the third step involves refining the denoised image by re-applying the sparse gradient approach. The proposed method is evaluated on several benchmark datasets, and the results show that it outperforms several state-of-the-art denoising methods in terms of peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM). Overall, the proposed method presents a promising approach to image denoising, which integrates sparse gradient and coupled PDEs to achieve a balance between preserving image edges and smoothing the image.

2.4.3 Threshold Denoising Algorithm

Thresholding algorithms are a type of image or signal processing technique that involves setting all values below a certain threshold to zero, and possibly scaling the remaining values. The basic principle of thresholding algorithms is to remove noise or other unwanted components from a signal or image by setting their values to zero. This is often done in the transform domain, such as the wavelet or Fourier domain, where the energy of the signal is more compact and easier to manipulate. The threshold denoising algorithm are of three types. They include hard threshold algorithm, soft threshold and adaptive threshold.

The Hard Thresholding (HT) algorithm is based on comparing each signal or image coefficient with a threshold value, and setting all coefficients below the threshold to zero. This principle is often used in conjunction with wavelet-based image processing techniques to remove noise from an image. In contrast, the Soft Thresholding (ST) algorithm modifies coefficients larger than the threshold by subtracting the threshold value. This allows for a more gradual reduction of coefficients and can lead to better preservation of image details and textures. While the Hard Thresholding algorithm is mainly used in particle swarm optimization, where it is used to filter out noise information, it can also be used for other types of signal processing tasks. Overall, the Hard Thresholding and Soft Thresholding algorithms are simple and computationally efficient methods for denoising signals and images, but they can result in over-smoothing or loss of detail in complex images. More advanced thresholding algorithms, such as adaptive thresholding, have been developed to address these limitations.

In 2003, Fodor and Kamath proposed several thresholding methods for wavelet-based denoising, including soft thresholding (ST), semi-soft thresholding, and Garrotte thresholding. These methods aim to remove noise from an image by applying a thresholding function to the wavelet coefficients, where coefficients above a certain threshold are kept and those below the threshold are modified or eliminated. Semi-soft thresholding is similar to soft thresholding, but with a "semi-soft" modification that introduces a second threshold to control the degree of thresholding. Garrotte thresholding, on the other hand, is a variant of soft thresholding that uses a weighted threshold to minimize a combination of the mean-squared error and the number of nonzero coefficients.

BayesShrink is a well-known image denoising algorithm proposed by Chang et al. in 2000. The algorithm is based on Bayesian estimation theory, which provides a mathematical framework for selecting the optimal threshold value for each sub-band of a wavelet transform. BayesShrink is an extension of the traditional wavelet thresholding approach, which uses a fixed threshold value for all sub-bands. BayesShrink applies a soft threshold to the wavelet coefficients in each sub-band, where the threshold value is estimated using a Bayesian approach. The algorithm first models the distribution of the wavelet coefficients in each sub-band as a generalized Gaussian distribution. Then, it estimates the parameters of the distribution using the maximum a posteriori (MAP) method and uses the estimated parameters to compute the optimal threshold value. The advantage of BayesShrink over traditional wavelet thresholding is that it adapts the threshold to the local statistics of each sub-band, resulting in better denoising performance. The algorithm has been shown to perform well in a wide range of applications and has been extended to other types of transforms, such as the contourlet transform and the nonsubsampled shearlet transform.

The suggested technique uses a two-part denoising model after pre-processing the image to extract its contour in order to get beyond the drawbacks of conventional denoising algorithms. The Matrix Sparse Denoising and Threshold Denoising algorithms are combined in the model's initial stage to accomplish rough denoising. The second phase involves fine denoising of the image using the concept of super-pixel image segmentation. To improve the ability to denoise the images and prevent edge blurring, the pre-processed and two-segmented images are then superimposed. Experiments are used to validate the algorithm and show its viability and efficacy.

The remaining chapters of this book are arranged as follows. The pre-processing techniques, such as edge extraction and adaptive threshold extraction, will be covered in chapter 2. We will go through the SVD decomposition principle and its importance in noise coarse filtering in chapter 3. The superpixel algorithm is used to create the superpixel-like method in chapter 4, which also introduces the idea of fine noise filtering. Chapter 5 will analyse the suggested denoising algorithm, and Chapter 6 will provide a performance report.

CHAPTER 3

IMAGE PRE-PROCESSING

Chapter 3
Image Pre-processing

3.1 Edge Detection:

Edge detection is the process of identifying edges in an image, which are the locations where the intensity or colour values change abruptly. There are many different methods for detecting edges, but most involve some form of filtering or gradient computation. In this response, I will provide a detailed overview of the steps involved in edge detection and how they lead to the edge image and edge extraction image. The first step in edge detection is to convert the image to grayscale, which simplifies the edge detection process by reducing the image to a single intensity channel. Colour information can still be used in some methods, but converting to grayscale is a common pre-processing step. Once the image has been converted to grayscale, the next step is to apply a filter to the image to smooth out noise and reduce the impact of small variations in intensity.

This method involves first extracting the edge of the image from a noisy image. This can be achieved using edge detection techniques such as the Canny edge detector or the Sobel operator. The resulting edge image is then modified so that the edge is changed to white, with a gray value of 255. This ensures that the upper limit of the gray value is reached, and that any colour superimposed on white will still result in white in a gray image.

Finally, the white edge image is superimposed on the original noisy image, resulting in a new noisy image with white edges. This noise image with white edges can be useful for a variety of applications, including object recognition and segmentation, where the edges are used to define the boundaries of objects in the image. The logic of edge perfection is a simple but effective method for creating an image with white edges, and it can be used in conjunction with other image processing techniques to achieve a wide range of tasks. By extracting the edges of an image and changing them to white, we can obtain an image with clear and distinct edges that can be used for a variety of image processing applications.

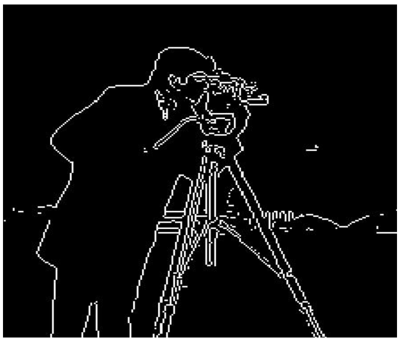


Fig 3.1 Edge image

3.2 Types of Edge Detection:

3.2.1 Sobel Method

The Sobel method is a popular edge detection algorithm used in image processing. It was named after its inventor, Irwin Sobel. The Sobel operator is a gradient-based method that calculates the gradient magnitude of an image by convolving it with a small filter kernel. The Sobel kernel has two components: one for horizontal changes and one for vertical changes in intensity. To use the Sobel method to detect edges in an image, the following steps are typically performed. Convert the image to grayscale, if it is not already in grayscale. Convolve the image with the Sobel kernel in both the horizontal and vertical directions to obtain the gradient magnitude at each pixel. Threshold the gradient magnitude image to obtain a binary edge map. The resulting binary edge map will have white pixels at the locations of edges detected by the Sobel operator, and black pixels elsewhere. The Sobel method is a simple and effective method for detecting edges in images, but it can be sensitive to noise in the image. There are other edge detection methods, such as the Canny edge detector, that are more robust to noise and produce more accurate edge maps.

3.2.2 Canny Edge Detection

Canny edge detection is a popular and widely used edge detection algorithm in image processing. It was developed by John F. Canny in 1986 and is known for its high accuracy and low error rate.

The Canny edge detection algorithm has several steps:

- Gaussian blur: The image is smoothed using a Gaussian filter to reduce noise and small details.

- Gradient calculation: The gradient magnitude and direction are calculated using the Sobel operator or a similar filter.
- Non-maximum suppression: This step removes all the pixels that are not considered to be part of the edges. It is done by only keeping the pixels with the highest gradient magnitude in the direction of the gradient.
- Double thresholding: This step separates the remaining pixels into two groups: strong edges and weak edges. A high threshold is used to identify strong edges and a low threshold is used to identify weak edges.
- Edge tracking: The weak edges are only kept if they are connected to strong edges. This is done by tracing the edges in the image using a connectivity algorithm, such as the Hysteresis thresholding algorithm.

The result of the Canny edge detection algorithm is a binary edge map, where the edges are represented by white pixels and the non-edges are represented by black pixels. The algorithm is highly effective at identifying edges in an image while minimizing false positives and false negatives. The Canny edge detection algorithm is widely used in computer vision and image processing applications, such as object detection, image segmentation, and feature extraction. However, it can be computationally expensive, especially for large images, and may not perform well in the presence of noise or in images with complex textures.

3.2.3 Prewitt Edge Detection:

Prewitt edge detection is another commonly used edge detection algorithm in image processing. It is a gradient-based method, similar to the sobel operator, that calculates the gradient magnitude of an image by convolving it with a small filter kernel. To use the Prewitt method to detect edges in an image, the following steps are typically performed. Convert the image to grayscale, if it is not already in grayscale. Convolve the image with the Prewitt kernel in both the horizontal and vertical directions to obtain the gradient magnitude at each pixel. Combine the horizontal and vertical gradient images using the square root of the sum of squares method to obtain the overall gradient magnitude. Threshold the gradient magnitude image to obtain a binary edge map. The resulting binary edge map will have white pixels at the locations of edges detected by the Prewitt operator, and black pixels elsewhere.

The Prewitt method is a simple and effective method for detecting edges in images, but it may produce slightly thicker edges compared to the sobel operator. It is also sensitive to noise in the image. Like the Sobel method, there are other edge detection methods, such as the Canny edge detector, that are more robust to noise and produce more accurate edge maps.

3.2.4 Robert Edge Detection:

Robert edge detection is a simple and computationally efficient edge detection algorithm that works by calculating the gradient magnitude of an image using two small filter kernels. To use the Robert method to detect edges in an image, the following steps are typically performed. Convert the image to grayscale, if it is not already in grayscale. Convolve the image with the Robert kernel in both the horizontal and vertical directions to obtain the gradient magnitude at each pixel. Combine the horizontal and vertical gradient images using the square root of the sum of squares method to obtain the overall gradient magnitude. Threshold the gradient magnitude image to obtain a binary edge map. The resulting binary edge map will have white pixels at the locations of edges detected by the Robert operator, and black pixels elsewhere. Robert edge detection is a simple and fast method for detecting edges in images, but it is not as accurate as other edge detection algorithms, such as the Sobel and Canny operators. The Robert operator is also highly sensitive to noise in the image. As a result, it is typically used for quick and simple edge detection applications where speed is more important than accuracy, or as a pre-processing step for other more accurate edge detection algorithms.

3.3 Edge perfection

The logic of edge perfection is a simple but effective method for creating an image with white edges, and it can be used in conjunction with other image processing techniques to achieve a wide range of tasks. By extracting the edges of an image and changing them to white, we can obtain an image with clear and distinct edges that can be used for a variety of image processing applications.

In the proposed algorithm, the process of edge perfection is an important step to improve the quality of the final denoising image.

- The first step in this process is to extract the edge of the original noisy image. This is done through edge detection techniques such as the Canny edge detector, which is commonly used due to its accuracy and low error rate. The resulting edge map is a binary image where the edges are represented by white pixels and the background is black.
- In the next step, the edge map is changed to white. This is done by assigning a gray value of 255 to all white pixels, which is the upper limit of gray values in an 8-bit grayscale image. This step ensures that the edges are easily distinguishable from the background.
- The resulting white edge image is then superimposed on the noisy image to obtain a noise image with white edges. This new noise image is then processed through the two-stage denoising algorithm to obtain a denoised image with enhanced edge sharpness and reduced noise.

By perfecting the edges in the image, the proposed algorithm is able to preserve the important edge information while effectively removing the noise. This is important for applications such as medical imaging, where the preservation of edges is crucial for accurate diagnosis and analysis. The edge perfection process also improves the overall visual quality of the final denoised image, making it more appealing to the human eye.

Most denoising algorithms aim to reduce noise in images, but they often do not take into account the important edge information in the image. This can result in the denoising algorithm filtering out useful edge information, leading to edge distortion. To address this issue, a proposed algorithm extracts and saves the edge information, which is typically only one pixel wide.

Fig 3.3 Edge extraction

This edge information is then separated from the rest of the image and a new, edgeless noise image is used for the denoising algorithm. By doing so, the edge information is preserved, and the algorithm can effectively reduce noise without distorting the edges of the image.

$$I = J \cap \Omega \quad (1)$$

The proposed algorithm involves extracting and preserving the edge information Ω' of the noisy image J, and separately obtaining the edge information Ω of zero gray value. These two edge sets are then intersected to obtain a new noise image I with white edges. This edgeless noise image I is used as the input for the denoising algorithm to effectively remove noise while preserving the edges. In summary, the algorithm extracts and preserves the edges of the noisy image and uses the resulting edgeless noise image for the denoising process.

3.4 Adaptive threshold extraction

In digital image processing, image denoising is an essential pre-processing step for improving the quality of images used in various applications such as computer vision, medical imaging, and surveillance systems. Image denoising algorithms aim to remove unwanted noise and preserve the important image features. The main challenge in image denoising is to find a suitable threshold for filtering out the noise while preserving the edges and details of the image. In recent years, many thresholding techniques have been proposed, but most of them have limitations in terms of their accuracy and adaptability.

One of the major issues with traditional thresholding techniques is that they rely on fixed values that are determined based on empirical observations. These fixed thresholds may work well for certain types of images, but they are not adaptable to different image characteristics. Therefore, an adaptive threshold method is proposed in this paper, which is based on image gray value.

The gray value of an image represents the brightness or intensity of the image pixel. In grayscale images, the gray value ranges from 0 to 255, with 0 being black and 255 being white. In this method, the gray values are used to estimate the noise level in the image. The algorithm consists of two main steps: noise level estimation and thresholding.

Noise level estimation is done by calculating the variance of the gray values in the image. The variance is a measure of the spread of the gray values around their mean. The higher the variance, the higher the noise level in the image. Once the noise level is estimated, the threshold is calculated using a function that takes into account the noise level and the local image characteristics.

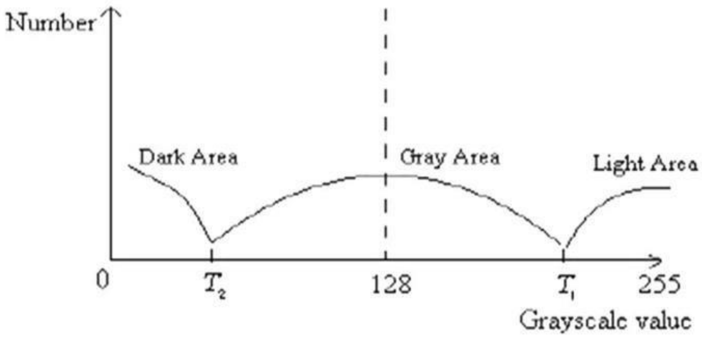


Fig 3.4 Schematic diagram of the adaptive threshold

The proposed method has several advantages over traditional thresholding techniques. First, it is adaptive to the noise level and the image characteristics, which makes it more accurate and robust. Second, it preserves the important features of the image, such as edges and details, while removing the noise. Third, it is computationally efficient and can be implemented in real-time applications.

In conclusion, the proposed adaptive thresholding method based on image gray value is a promising approach for image denoising. It overcomes the limitations of traditional thresholding techniques by adapting to the image characteristics and the noise level. The method can be applied to a wide range of image denoising applications and has the potential to improve the quality of the processed images

Step 1: To begin, a gray histogram of an image is obtained, which represents the distribution of pixel intensities in the image. Each bin in the histogram represents a range of intensity values, and the height of the bin indicates the number of pixels in the image that have intensities within that range.

Next, the median of the histogram is calculated, which in this case is an integer value of 128. This value is used to divide the image into two areas: E1 and E2.E1 represents the area of the image where the gray value of the pixels is greater than 128. This can be thought of as the "brighter" part of the image. E2, on the other hand, represents the area where the gray value of the pixels is less than 128, or the "darker" part of the image.

Step 2: The threshold values T01 and T02 are derived for E1 and E2 respectively. To calculate T01 for E1, we take the maximum value of the intensities in E1 and subtract 2. Similarly, to calculate T02 for E2, we take the maximum value of the intensities in E2.

$$\mathbf{T01=max(E1)} \quad \mathbf{(3.1)}$$

$$\mathbf{T02=max(E2)} \quad \mathbf{(3.2)}$$

Step 3: The image is divided into three blocks based on the threshold values calculated in step 2. A pixel with intensity greater than T0 1 belongs to block G1, which contains the brightest pixels in the image. A pixel with intensity smaller than T01 and larger than T0 2 belongs to block G2, which contains pixels with intermediate intensity values. A pixel with intensity smaller than T0 2 belongs to block G3, which contains the darkest pixels in the image.

Step 4: The average values a1, a2, and a3 of G1, G2, and G3 are calculated. This involves computing the mean intensity values of all the pixels in each block separately. The average value a1 represents the average intensity of the brightest pixels in the image, while a3 represents the average intensity of the darkest pixels. The average value a2 represents the average intensity of the pixels with intermediate intensity values.

Step 5: The adaptive thresholds T1 and T2 are computed, both equations serve the same purpose of determining appropriate threshold values for segmenting an image into the "Dark Area", "Gray Area", and "Light Area"**.**

$$\mathbf{T1 = (a1 + a2)/2} \quad \mathbf{(3.3)}$$

$$\mathbf{T2 = (a2 + a3)/2} \quad \mathbf{(3.4)}$$

Step 6: The adaptive thresholding technique involves dividing an image into small blocks or regions and then determining whether each block belongs to the "Dark Area", "Gray Area", or "Light Area" based on its average gray value. The image is physically divided into a × b small blocks {B1, B2…Bp × b}, where b < a < n. In each block, the average gray value is calculated, and these values are stored in the array e1; e2…ep × b. The average value of the i-th gray value is denoted by ei, and the corresponding image block is Bx.

Based on the average gray value ex in each block, the following conclusions can be drawn:

- When the average gray value ex is less than T2, where T2 is a threshold value, the corresponding block Bx is classified as belonging to the "Dark Area".
- When the average gray value ex is between T2 and T1, where T1 is another threshold value, the corresponding block Bx is classified as belonging to the "Gray Area".
- When the average gray value ex is greater than or equal to T1, the corresponding block Bx is classified as belonging to the "Light Area".

This approach can help to segment an image into regions with different levels of brightness, making it easier to apply appropriate image processing techniques to each region.

CHAPTER 4

BASIC FILTERING ALGORITHM OF NOISE

Chapter 4
Basic Filtering Algorithm of Noise

4.1 Singular Value Decomposition

Singular Value Decomposition (SVD) is a powerful mathematical technique used to decompose a matrix into three separate matrices, each of which contains important information about the original matrix. SVD is widely used in many applications, including image and signal processing, data analysis, and machine learning. In this response, we will elaborate on SVD in the context of noise filtering in image processing. Noise is an unwanted signal that can corrupt the original image and reduce its quality. One approach to removing noise from an image is to use a filtering algorithm that separates the noise from the signal. However, filtering algorithms can be complex, and some may even remove the essential features of the image.

To address this problem, researchers have developed an approach called noise coarse filtering, which combines SVD with filtering to separate the noise from the signal. In this approach, the noisy image is first prepared by applying an adaptive threshold to segment the image into three areas based on their brightness level: Dark Area, Gray Area, and Light Area. Then, SVD is applied to the image matrix, resulting in a matrix of singular values. The singular values represent the amount of information contained in the original matrix, with larger values corresponding to more significant information and smaller values corresponding to noise or less significant information. By applying a threshold to the singular value matrix, we can remove the smaller values that correspond to noise or less significant information, while retaining the larger values that represent the essential features of the image.

To apply the threshold, a percentage threshold η is defined, and the singular value matrix is reduced by the threshold. The thresholded singular value matrix is obtained by setting any singular value below the threshold to zero. Once the singular value matrix has been reduced by the threshold, the matrix is restored using inverse SVD. This involves multiplying the reduced singular value matrix with the transposes of the orthogonal matrices U and V to obtain the filtered matrix. The resulting matrix contains only the significant information, which corresponds to the true image, while the noise and less significant information have been filtered out. Finally, the filtered matrix is used to reconstruct the image, resulting in a denoised version of the original image.

One of the advantages of SVD is that it is a powerful tool for dimensionality reduction. In many applications, data is represented by high-dimensional matrices, which can be difficult to process or analyse. SVD can be used to transform high dimensional data into lower-dimensional representations while preserving the most important features of the data. This makes SVD an important tool for image and signal processing, as it allows us to extract the essential information from a noisy signal or image. Another advantage of SVD is that it is a computationally efficient technique. Unlike some other approaches to noise filtering, SVD can be computed quickly and efficiently, making it practical for use in real-world applications.

In conclusion, Singular Value Decomposition is a powerful mathematical technique that has many applications in image and signal processing. In the context of noise filtering, SVD is an essential tool for separating the noise from the signal and preserving the essential features of the image. By applying a threshold to the singular value matrix, we can remove the noise components while retaining the significant information. SVD is a computationally efficient technique that allows us to extract essential information from high-dimensional data, making it an important tool in many fields.

Singular value decomposition (SVD) is an important matrix decomposition technique in linear algebra. It is used to decompose an m x n matrix M, where all elements belong to the real or complex field, into the product of three matrices: U, S, and V* (the conjugate transpose of V).This factorization can be written as

$$\mathbf{M = USV^*} \qquad \mathbf{(4.1)}$$

where U is an m x m unitary matrix, S is an m x n diagonal matrix with non-negative real numbers on the diagonal, and V* is an n x n unitary matrix. The diagonal elements of S are called the singular values of M, and they represent the square roots of the eigenvalues of the matrix M*M, where * denotes the conjugate transpose. SVD has many applications in signal processing, image compression, and data analysis. It can be used to reduce the dimensionality of data, to remove noise from signals, and to solve linear equations. In addition, SVD is also used in other matrix factorizations, such as principal component analysis (PCA) and non-negative matrix factorization (NMF). Singular value decomposition (SVD) is a powerful matrix factorization technique that decomposes a matrix M into three matrices: U, S, and V*, where U and V* are unitary matrices and S is a diagonal matrix with singular values on the diagonal. In image denoising, SVD is often used to remove noise from images while preserving important features of the image.

The columns of U form a set of basis vectors for the orthogonal "input" or "analysis" of M, and the columns of V form a set of basis vectors for the orthogonal "output" of M. These basis vectors are eigenvectors of MM* and M*M, respectively. The diagonal elements of S are the singular values of M and can be thought of as scalar "expansion controls" between input and output. These singular values correspond to the column vectors of U and V, and they represent the importance of the basis vectors in the input and output spaces. In image denoising, the matrix M represents the noisy image, and the singular value matrix S is obtained by performing SVD on M.

The threshold filtering process is performed on the singular value matrix S to filter out the noise information in the image. This is done by setting all singular values less than a threshold value to zero. This effectively removes the high-frequency components of the image that correspond to noise. After the thresholding step, the matrix is restored by performing the inverse SVD. This results in a matrix that has been filtered to remove the noise. Finally, the denoised image is obtained by taking the product of the left singular matrix U, the reduced singular value matrix S, and the right singular matrix V*. The denoised image is a high-quality representation of the original image with most of the noise removed. One advantage of SVD for image denoising is that it is computationally efficient, especially for small to medium-sized images. In addition, SVD is a very flexible technique that can be adapted to different types of noise and different types of images. However, SVD also has some limitations. For example, it assumes that the noise is additive and uncorrelated, which may not always be the case. In addition, the threshold value used in the filtering process can have a significant impact on the quality of the denoised image. Therefore, it is important to carefully select the threshold value to achieve the best possible results.

Singular value matrix S is a diagonal matrix that is obtained through the singular value decomposition (SVD) of a given matrix M. The elements on the diagonal of S correspond to the singular values of M. These singular values are arranged in descending order, with the largest singular value at the top left corner and the smallest singular value at the bottom right corner. In most cases, the values of the singular values decrease rapidly from the largest to the smallest. This means that the information contained in the singular values is not uniformly distributed, and a few large singular values can represent the majority of the information in the matrix M. As a result, it is often possible to obtain an approximate representation of the original matrix by retaining only a small number of the largest singular values and setting the remaining singular values to zero.

The reduced singular value matrix that is obtained by setting the smaller singular values to zero can be used to reconstruct an approximate version of the original matrix. This approach is often referred to as "truncating" the SVD. The number of singular values to retain depends on the desired level of accuracy of the approximate matrix. In most cases, the sum of the singular values of the first 5% to 10% accounts for more than 95% of the sum of all singular values. Therefore, retaining only the top 5% to 10% of the largest singular values can provide a good approximation of the original matrix.

The truncated SVD can be used in various applications, such as image compression, image denoising, and data analysis. In image compression, the reduced singular value matrix can be used to represent the image in a compressed form. This approach can significantly reduce the size of the image without losing much of the important information. In image denoising, the reduced singular value matrix can be used to remove noise from the image while preserving the important features of the image. In data analysis, the reduced singular value matrix can be used to perform dimensionality reduction, which can help to identify the most important features of the data.

4.2 Singular Value Inverse Decomposition

Singular value inverse image decomposition is a technique used in image denoising to separate noise from the signal in an image. This technique is based on the singular value decomposition (SVD) of the image matrix, which is a mathematical technique used to decompose a matrix into three separate matrices, each containing important information about the original matrix. In image denoising, the first step is to prepare the image by applying an adaptive threshold to segment the image into three areas based on their brightness level: Dark Area, Gray Area, and Light Area. The image matrix is then decomposed using SVD, which results in a matrix of singular values.

The singular values represent the amount of information contained in the original matrix, with larger values corresponding to more significant information and smaller values corresponding to noise or less significant information. By applying a threshold to the singular value matrix, we can remove the smaller values that correspond to noise or less significant information while retaining the larger values that represent the essential features of the image. To apply the threshold, a percentage threshold is defined, and the singular value matrix is reduced by the threshold. The thresholded singular value matrix is obtained by setting any singular value below the threshold to zero. Once the singular value matrix has been reduced by the threshold, the matrix is restored using inverse SVD. The inverse SVD involves multiplying the reduced singular value matrix with the transposes of the orthogonal matrices U and V to obtain the filtered matrix. The resulting matrix contains only the significant information, which corresponds to the true image, while the noise and less significant information have been filtered out. Finally, the filtered matrix is used to reconstruct the image, resulting in a denoised version of the original image. The singular value inverse image decomposition technique is efficient and can be used for real-time image denoising applications.

In summary, singular value inverse image decomposition is a technique used in image denoising to remove noise from an image by reducing the singular value matrix and restoring the matrix using inverse SVD. The technique is efficient and practical for real-world image denoising applications.

Noise coarse filtering is a technique used to remove noise from digital images by filtering out high-frequency components of the image. High-frequency components in an image correspond to the areas of the image with rapid changes in intensity, such as edges and textures. These areas are typically most affected by noise, and removing them can significantly improve the image quality. The technique of noise coarse filtering involves several steps. The first step is to obtain a singular value matrix using singular value decomposition (SVD). SVD is a matrix factorization technique that breaks down a matrix into its constituent parts, namely, the left singular vectors, the singular values, and the right singular vectors. In the context of image processing, SVD is used to separate the underlying signal (i.e., the image) from the noise.

Once the singular value matrix has been obtained, the next step is to apply a percentage threshold, η, to the singular values. The threshold value is chosen such that only the largest singular values are retained, and the smaller ones are set to zero. The effect of this thresholding is to reduce the rank of the singular value matrix, effectively filtering out the high-frequency components of the image corresponding to the noise. After the thresholding step, the reduced singular value matrix is then restored by performing the inverse SVD. This step involves multiplying the left singular vectors with the retained singular values and the right singular vectors. The result is a filtered version of the original image, with the high-frequency components corresponding to the noise removed. One of the key advantages of the noise coarse filtering technique is that it is computationally efficient, making it well-suited for real-time applications. The SVD decomposition and inverse SVD can be performed using standard linear algebra routines, and the thresholding step is a simple element-wise operation.

However, there are also some limitations to the technique. One of the main limitations is that it is only effective for removing certain types of noise, such as Gaussian noise. Other types of noise, such as salt-and-pepper noise or speckle noise, may require different filtering techniques. Additionally, the threshold value η needs to be carefully chosen to ensure that the noise is adequately filtered out while preserving the important features of the image.

In summary, noise coarse filtering is a simple and effective technique for removing noise from digital images. By using SVD to decompose the image into its signal and noise components and applying a threshold to filter out the noise, it is possible to significantly improve the image quality. However, as with any image filtering technique, it is important to carefully choose the threshold value and consider the limitations of the technique in order to obtain the best results.

Image denoising is the process of removing noise or unwanted artifacts from an image while preserving the important features of the image. The goal is to obtain an image that is as close as possible to the original, noise-free image. One common approach to image denoising is to use a technique called singular value decomposition (SVD). SVD is a mathematical technique that decomposes a matrix into three matrices: a left singular matrix, a diagonal matrix of singular values, and a right singular matrix. By thresholding the singular values, we can remove the high-frequency components of the image that correspond to noise, while preserving the low-frequency components that correspond to the important features of the image.

The first step in the process of noise coarse filtering is to obtain a singular value matrix by performing SVD on the noisy image. Once the singular value matrix is obtained, a threshold value η is defined, typically as a percentage of the maximum singular value. The singular value matrix is then reduced by setting all singular values less than η to zero. This effectively removes the high-frequency components of the image that correspond to noise. After the thresholding step, the matrix is restored by performing the inverse SVD. This results in a matrix that has been filtered to remove the noise. Finally, the coarse noise filtered image is obtained by taking the product of the left singular matrix, the reduced singular value matrix, and the right singular matrix.

In summary, the singular value matrix S is a diagonal matrix that contains the singular values of a given matrix M. The values of the singular values decrease rapidly, and a few large singular values can represent the majority of the information in the matrix M. Truncating the SVD by retaining only the top few singular values can provide an approximate representation of the original matrix, which can be used in various applications such as image compression, image denoising, and data analysis.

The q-th order singular value matrix using a percentage threshold η. The singular values in the singular value matrix are arranged in descending order. The sum of the first k singular values is used to estimate the percentage of the total sum of singular values that they represent. By setting a percentage threshold η, a decision is made on the number of singular values required to represent the image accurately.

$$\lambda 1+\lambda 2 + \dots \lambda k \approx \eta\ (\lambda 1+\lambda 2 + \lambda 3 \dots \lambda q) \tag{4.2}$$

$$\eta \subseteq (0.8,\ 0.95) \tag{4.3}$$

The sum of the first k singular values λ1, λ2, ..., λk is approximated to η times the sum of all singular values λ1, λ2, λ3, ..., λq. The percentage threshold η is a value between 0.8 and 0.95, which means that the approximation should be accurate enough to preserve 80% to 95% of the image's energy. This approximation enables the reconstruction of the singular value matrix with a reduced number of singular values while still preserving most of the important information contained in the original image.

After obtaining the threshold η, the singular value matrix is reduced by setting all singular values below the threshold to zero. This reduces the size of the singular value matrix and effectively filters out the noise in the image. The resulting matrix is then restored by applying the inverse singular value decomposition, which reconstructs the denoised image using only the significant singular values.

CHAPTER 5

FINAL FILTERING ALGORITHM OF NOSIE

Chapter 5
Final Filtering Algorithm of Nosie

5.1 Super Pixel Algorithm

Superpixels are a useful tool for image segmentation, which involves dividing an image into distinct regions or objects. Superpixels are groups of adjacent pixels that have similar properties, such as colour, texture, or intensity. They are typically used to simplify subsequent image processing tasks by reducing the amount of data to be processed and improving the accuracy of segmentation.

The Simple Linear Clustering (SLIC) algorithm is a popular method for generating superpixels. It is an efficient and fast algorithm that works by clustering pixels in a five-dimensional space, which includes the (L, a, b) color values and the (x, y) coordinates of each pixel. The algorithm aims to group together adjacent pixels that are similar in colour and location, resulting in compact and uniform superpixels.

The first step in the SLIC algorithm is to convert the input image from the RGB color space to the CIE-Lab color space. This is because the CIE-Lab color space is designed to be perceptually uniform, meaning that differences in color values correspond to differences in perceived color. This makes it easier to cluster pixels based on their color values. Next, the image is divided into a grid of square regions, with each region containing a fixed number of pixels. The size of the regions is determined by the desired size of the superpixels. For example, if the desired size of the superpixels is 100 pixels, and the input image is 500 x 500 pixels, then the image would be divided into a grid of 25 x 25 regions, each containing 100 pixels. For each region, the algorithm selects a "seed" pixel that is the pixel with the lowest gradient value in the region. The gradient is calculated using the Sobel operator, which measures the change in color or intensity between adjacent pixels. The seed pixel is used as the initial center of the superpixel. Next, the algorithm searches for neighbouring pixels that are within a certain distance of the seed pixel in the five-dimensional space. The distance metric used is the Euclidean distance, which measures the straight-line distance between two points. The distance between two pixels is calculated using their (L, a, b, x, y) values. Once a set of neighbouring pixels is identified, the algorithm calculates the average (L, a, b) color values and (x, y) coordinates of the pixels in the set. This becomes the new center of the superpixel. The process is repeated until the centers of all superpixels converge.

The final step of the SLIC algorithm is to enforce connectivity between superpixels. This is done by comparing the distance between the centers of neighbouring superpixels with a threshold value. If the distance between two superpixels is below the threshold, then they are merged into a single superpixel. The SLIC algorithm has several advantages over other superpixel algorithms. It is efficient, fast, and can generate uniform superpixels that are compact and well-defined. It also has a small number of parameters, which makes it easy to use and tune for different applications. The SLIC algorithm is a superpixel segmentation technique that generates a fixed number of superpixels by clustering pixels in a five-dimensional space. The algorithm starts by initializing a fixed number of seed points or cluster centers. The number of cluster centers is determined by a user-defined parameter Q, which specifies the desired number of superpixels in the output.

The initial seed points are evenly spaced on a regular grid with a distance of $p = \text{sqrt}(Y/Q)$ pixels between them, where Y is the total number of pixels in the input image. This distance ensures that each superpixel has approximately the same size and that there is a sufficient overlap between adjacent superpixels. Next, the algorithm performs a local search around each seed point to find the nearest pixels in the fivedimensional space. The search is limited to a rectangular region of size $2P \times 2P$ around the seed point, where P is the size of the superpixel in pixels. The search space is chosen to be sufficiently large to capture all the pixels that are potentially similar to the seed point while still being computationally efficient. Each pixel found in the search space is assigned to the nearest seed point based on the Euclidean distance in the fivedimensional space. This assigns each pixel to a superpixel, creating an initial segmentation of the input image.

Next, the algorithm calculates the average color and position of all pixels assigned to each superpixel. This average color and position become the new cluster center for that superpixel. This step helps to refine the initial segmentation by moving the cluster centers closer to the true centers of each superpixel. After updating the cluster centers, the algorithm repeats the local search step to find the nearest pixels around each new cluster center. The pixels are assigned to the closest superpixel based on the updated cluster centers. The algorithm then recalculates the average color and position of all pixels assigned to each superpixel and updates the cluster centers accordingly. The process of local search, pixel assignment, and cluster center update is repeated iteratively until convergence. Convergence is reached when the cluster centers stop moving or when a maximum number of iterations is reached. The final output of the algorithm is a set of Q superpixels, where each superpixel contains a set of pixels that are similar in color and position. The size of each superpixel is approximately equal, and the boundary between adjacent superpixels is well-defined.

As mentioned earlier, when the size of the image is large, the spatial distance between pixels (x, y) also plays an important role in determining the similarity of pixels. To incorporate this spatial distance into the superpixel generation process, the SLIC algorithm normalizes the (x, y) coordinates before computing the vector distance between pixels.

$$\mathbf{Dlab=sqrt}(lk - li)^{\wedge}2 + (ak - ai)^{\wedge}2 + (bk - bi)^{\wedge}2\ ) \qquad \mathbf{(5.1)}$$

Equation shows the calculation of the color distance between two pixels in the CIE-Lab color space. Here, l, a, and b represent the lightness, green-red, and blueyellow color values of the two pixels, respectively. The subscript k denotes the reference pixel, and denotes the pixel being compared to the reference pixel. The distance is calculated using the Euclidean distance formula in a 3D color space.

$$\mathbf{dxy=}\sqrt{}(xk - xi)^{\wedge}2 + (yk - yi)^{\wedge}2 \qquad \mathbf{(5.2)}$$

Equation calculates the spatial distance between two pixels using the Euclidean distance formula. The values xi, yi, xk, and yk represent the x and y coordinates of the two pixels.

$$\mathbf{Ds = dlab + (w/P) \times dxy} \qquad \mathbf{(5.3)}$$

To combine the color distance and spatial distance, the algorithm uses Equation 10 to calculate the final similarity measure between two pixels. The parameter w adjusts the weight given to the spatial distance compared to the color distance. A larger value of w gives more weight to the spatial distance, while a smaller value gives more weight to the color distance.

The term P in Equation represents the size of the superpixel in terms of the number of pixels. The larger the superpixel, the smaller the weight given to the spatial distance. This helps to ensure that larger superpixels are generated in areas where the color values are more homogeneous. By iteratively updating the cluster centers based on the average values of the pixels belonging to each superpixel, the SLIC algorithm generates superpixels that are both visually homogeneous and compact. The use of the CIE-Lab color space and the incorporation of spatial distance into the similarity measure allows the algorithm to generate superpixels that are more perceptually meaningful than those generated by other algorithms.

5.2 Final Filtering Algorithm of Noise

Denoising algorithms are commonly used to remove noise from images, which is a common problem in digital image processing. In many cases, the noise in an image is additive, meaning it is independent and identically distributed (i.i.d.) and is usually modeled as Gaussian noise. Traditional denoising algorithms aim to reduce the noise level in the image while preserving the underlying structure and details.

The algorithm determines whether a pixel is noise or not by comparing it to the surrounding pixels and analyzing the statistical properties of the noise. If the pixel is determined to be noise, the algorithm applies denoising measures based on the degree of noise and the importance of the pixel to the overall image quality. By selectively filtering the noisy pixels and retaining the important information, the fine noise filtering algorithm can achieve a better denoising effect compared to traditional denoising algorithms that treat all pixels equally. The algorithm can also be adapted to different types of noise and images, making it more versatile and effective.

From superpixel algorithm, the super-pixel algorithm is obtained. It can also be applied to image denoising. This algorithm works divides the image into superpixels and then analyzing the features of the pixels within each superpixel to determine whether they are noisy or not. Compared to traditional denoising algorithms that treat each pixel equally, the superpixel-like algorithm can selectively filter noisy pixels while preserving the important image features. In particular, the "image-pixel" level of the superpixel-like algorithm is more suitable for image denoising than the "image-area" level of the superpixel algorithm. At the image-pixel level, each pixel is treated as a separate superpixel, and its features, such as the Gray value, are extracted and compared to the features of neighbouring pixels. This approach is more fine-grained than the image-area level, where larger regions are considered, and is therefore better suited for denoising tasks where small-scale details are important. By analyzing the features of each pixel in the neighbourhood, the superpixel-like algorithm can effectively filter out noise while retaining the important image features. However, like other denoising algorithms, the effectiveness of the superpixel-like algorithm depends on the characteristics of the noise and the image. In some cases, the noise may be too strong or too complex to be effectively filtered, or the important image features may be too subtle to be preserved. Therefore, it is important to carefully evaluate the algorithm's performance and optimize its parameters for specific applications.

The denoising algorithm uses a template of size m x m, with a default value of m set to 3, to extract a region around the pixel being processed. The purpose of this step is to analyze the pixel intensities in the local neighbourhood of the target pixel.Once the template is defined, the algorithm calculates an average value of all the pixel intensities in the template and assigns it to a variable P. In addition, a threshold value α is defined between [24, 36]. This threshold is used to control the sensitivity of the algorithm to noise and plays a crucial role in determining which pixels are considered to be noisy and should be filtered out.

Step 1: The denoising algorithm that uses a template-based approach to identify and filter out noisy pixels. Here is the description of about the algorithm steps:

- A template of size m x m (default value of m is 3) is selected around the pixel being processed.
- A threshold value α between 24 and 36 is defined.
- The centre pixel of the template is denoted as P (xn, yn), and the average value of all pixels is calculated in the template .

The purpose of these steps is to define the region around the pixel being processed and to obtain an average value of the pixel intensities in this region. This information will be used in subsequent steps to determine whether the pixel is noisy or not. The threshold value α is used to control the sensitivity of the algorithm to noise. If the threshold value is set too high, the algorithm may not be able to detect noisy pixels, while if it is set too low, the algorithm may remove too much detail from the image. Therefore, an appropriate value of α needs to be chosen for the specific application.

Step 2: To further analyze the pixel intensities, the algorithm establishes a two dimensional coordinate system along the edge of the m x m template. The centre point of the template is defined as P(xn, yn), and there are eight neighbouring pixels in the 3 x 3 neighbourhood surrounding the centre point.

To determine whether a pixel is noisy or not, the algorithm calculates the absolute difference between the intensity of the centre pixel and the intensity of the pixel above it, as well as the pixel on the left side of it. Specifically, the absolute difference between P (xn-1, yn) and P(xn, yn) is calculated and stored as TU.

TU = | P(xn-1, yn) - P(xn, yn) |

The absolute difference between P(xn+1, yn) and P(xn, yn) is calculated and stored as TD.

TD = | P(xn+1, yn) - P(xn, yn) |

The absolute difference between P(xn, yn-1) and P(xn, yn) is calculated and stored as TL.

TL = | P(xn, yn-1) - P(xn, yn) |

P(xn-1, yn-1)	P(xn-1, yn)	P(xn-1, yn+1)
P(xn, yn-1)	P(xn, yn)	P(xn, yn+1)
P(xn+1, yn-1)	P(xn+1, yn)	P(xn+1, yn+1)

Fig 5.2 Templete of P

The absolute difference between P(xn, yn+1) and P(xn, yn) is calculated and stored as TR. Similarly calculate the TLU, TRU, TLD,TRD.

$$TR = | P(xn, yn+1) - P(xn, yn) |$$

The absolute difference between the pixels of an image helps in order to measure the amount of noise present in the image. The absolute difference between two pixels is simply the absolute value of the difference between their intensity values. This metric is commonly used in denoising algorithms because it measures the difference in intensity between neighbouring pixels. In a noisy image, neighbouring pixels may have significantly different intensities due to the presence of noise. By measuring the absolute difference between these pixels, we can identify areas of the image that are likely to be affected by noise.

By using this metric, we can develop denoising algorithms that can identify and remove noise from an image. The goal of image denoising is to remove noise while preserving as much of the original image as possible. By measuring the absolute difference between neighbouring pixels, denoising algorithms can identify areas of the image that are likely to be affected by noise and apply appropriate filters to reduce the noise while protecting the original image.

Step 3: In “Light Area” and “Dark Area”, the difference matrix M is calculated. Then determining whether each block belongs to the "Light Area", "Dark Area", or "Gray Area" based on its average gray value. In chapter 3, we already find the threshold values and find the average value of pixels in small block (m x m) with the help of these values find the region of the block belongs to the gray area, Dark area or Light area.

$$\mathbf{M} = \begin{pmatrix} \mathbf{TLU} & \mathbf{TU} \\ \mathbf{TRU} & \mathbf{TL} \\ \mathbf{TR} & \mathbf{TLD} \\ \mathbf{TD} & \mathbf{TRD} \end{pmatrix} \qquad \mathbf{(5.4)}$$

It describes a method for identifying and removing noise in an image using a template based approach. The method involves the various steps. Define a template of size MxM that will be used to analyze the image for noise. The center pixel of the template is not considered when analyzing for noise. Apply the template to every pixel in the image and calculate the difference between the center pixel of the templete and the pixel values. This will result in a matrix M, where each element represents the difference between the centre pixel of the template and the corresponding pixel value except the center pixel.

In this technique, a template is defined that will be used to analyze the image for noise. The template is of size MxM and is centered at each pixel in the image. However, the center pixel is not considered when analyzing for noise. Subtract a threshold value α from each element in the matrix M to obtain a new matrix M'. If an element in M' is greater than 0, it indicates that the corresponding pixel in the image is noisy. Identify all the pixels in the image that are determined to be noisy using the matrix M'. If there are N or more noisy pixels in the template (excluding the center pixel), the template is classified as a noise template. For all pixels within a noise template, their gray values are changed to a fixed value S1. Here, S1 is the average of all pixels in the template of matrix m x m where S is the template. By applying this method to the image, the noise can be identified and removed from the image. However, the effectiveness of the method may depend on the choice of template size and threshold value, which can affect the sensitivity and specificity of the noise "detection algorithm.

$$N = \left(\frac{m^2-1}{2} + 1\right) \tag{5.5}$$

$$M' = \begin{pmatrix} TLU' & TU' \\ TRU' & TL \\ TR' & TLD' \\ TD' & TRD' \end{pmatrix} \tag{5.6}$$

The identification and removal of noise from an image is a common problem in image processing, and many techniques have been developed to tackle. Calculating the difference between a template and the pixel values of an image, and then identifying the pixels that are determined to be noisy. Once the difference matrix M is obtained, the next step is to apply a threshold value α to create a new matrix M'. Elements in M' that are greater than 0 indicate the presence of noise in the image. The number of elements in M' that are greater than 0 is then counted and recorded as N′.

The value of N′ provides a measure of the amount of noise in the image. A higher value of N′ indicates more noise in the image, while a lower value indicates less noise. The threshold value α is an important parameter in this method and can affect the value of N′. A lower threshold value α will result in more elements in M' that are greater than 0, which in turn will result in a higher value of N′. Conversely, a higher threshold value α will result in fewer elements in M' that are greater than 0, which will result in a lower value of N′. Therefore, the choice of α should be based on the specific requirements of the application, and it should be carefully selected to achieve the desired level of noise reduction while preserving image quality.

After determining the value of N′, the next step is to identify the pixels in the image that are determined to be noisy. If a template contains N or more noisy pixels (excluding the center pixel), it is classified as a noise template. For all pixels within a noise template, their gray values are changed to a fixed value S, which effectively removes the noise from the image. The template size is another important parameter in this method. A larger template size will capture more information about the image, but it will also increase the computational cost of the algorithm. A smaller template size, on the other hand, will reduce the computational cost but may not capture enough information about the image to effectively reduce noise.

This method is a simple and effective technique for removing noise from images. However, it is not without its limitations. The method is only effective when the noise is relatively uniform and can be represented by a simple statistical model. It may not work well for images with complex noise patterns or images that contain noise in regions with high spatial frequency. The method described in the earlier paragraph is a template-based approach that involves calculating the difference between a template and the pixel values of an image, and then identifying and removing noisy pixels. The effectiveness of the method depends on several parameters, including the threshold value, template size, complexity of the noise pattern in the image. By carefully selecting these parameters, this method can be an effective technique for reducing noise in images.

$$P = \begin{cases} P1; & N' \geq N \\ P; & N' < N \end{cases} \tag{5.7}$$

Step 3: Determining whether each block belongs to the "Light Area", "Gray Area", or "Dark Area" based on its average gray value. we already find the threshold values and find the mean value of pixels in small block (m x m) with the help of these values find the region of the block belongs to the gray area. Then apply the template to every pixel in the image and calculate the difference between the template and the pixel values. This will create a matrix M where each element represents the difference between the center pixel and the corresponding neighbourhood pixel values except the center pixel. A threshold value α is then subtracted from each element in the matrix M to create a new matrix M'. If an element in M' is greater than 0, it indicates that the corresponding pixel in the image is noisy.

If $\forall T' \in M'$ then

$$\mathbf{P} = \begin{cases} \mathbf{P1;\ \ T' \geq 0} \\ \mathbf{P;\ \ T' < 0} \end{cases} \qquad \mathbf{(5.8)}$$

Step 4: The process of applying the noise filtering algorithm to an image involves moving the defined template smoothly over the entire image. The algorithm starts by selecting the first template from the image, which is done at time t0. The template is then used to calculate the matrix M and M′, and the pixels determined to be noise are replaced with the mean value S. This results in a partially filtered image. The algorithm then selects the next template from the image, which is done at time t1. The same process is repeated for this template, resulting in further noise reduction in the image. The algorithm continues to move the template smoothly over the image, selecting and filtering each template in turn, until the entire image has been covered. This completes the noise fine filtering process for the entire image.

The time t0, t1, t2, and so on, represent the time at which each template is selected and processed by the algorithm. The process of moving the template over the image and selecting the next template is done sequentially, which means that the time at which each template is processed is determined by the order in which they are encountered in the image. The smooth movement of the template over the image is an important aspect of the noise filtering algorithm, as it ensures that all parts of the image are covered and processed. The algorithm is designed to be efficient, and the templates are selected and processed in a systematic manner, which ensures that the filtering process is thorough and effective.

The noise fine filtering algorithm involves selecting a template from the image and processing it to determine which pixels are noise. The process is repeated for each template in the image, with the algorithm moving smoothly over the image to ensure that all parts of the image are covered. The time at which each template is processed is determined by the order in which they are encountered in the image. The algorithm is designed to be efficient and effective, and it can significantly reduce noise in digital images.

The process of ensuring that the template center is an adjacent pixel in each time is an important aspect of the noise filtering algorithm, as it ensures that every pixel in the image is processed. This means that the algorithm can effectively detect and remove noise from all parts of the image, resulting in a better denoising effect. In addition to the noise filtering algorithm itself, the final filtering algorithm of noise also includes the use of an adaptive threshold. This allows the algorithm to apply different denoising methods to different areas of the image, based on the level of noise present in each area. By using an adaptive threshold, the algorithm can more effectively preserve the effective information in the image, while still filtering out noise in the judged-noise areas.

The use of an adaptive threshold also improves on the limitations of traditional denoising algorithms, which can sometimes destroy the effective information in an image while attempting to remove noise. By using a more targeted approach to denoising, the algorithm is able to filter noise more thoroughly, resulting in a final image with a better denoising effect.

Overall, the noise filtering algorithm and final filtering algorithm of noise work together to effectively detect and remove noise from digital images. By using a combination of techniques, including template-based noise filtering and adaptive thresholding, the algorithm is able to preserve the effective information in an image while still achieving a high level of noise reduction. This results in a final image with a better quality and a more visually appealing appearance.

5.3 Peak signal-to-noise ratio

PSNR is a measure of image or video quality and is often used in image and video processing. It is defined as the ratio of the maximum potential of the signal to the power of the noise that affects the integrity of its representation. In short, it measures the difference between the original signal and the distorted or noisy version of the signal. PSNR is calculated as:

$$\mathbf{PSNR = 10\ log10\ (MAX\hat{\ }2\ /\ MSE)} \qquad \mathbf{(5.9)}$$

Where MAX is the maximum possible pixel value of the image and MSE is the average squared error between the original image and the corrupted version. The resulting PSNR is usually expressed in decibels (dB).

A higher PSNR indicates better picture or video quality as it means less distortion or noise compared to the original signal. However, PSNR is not a perfect measure of visual quality as it does not take into account human perception and sometimes gives inaccurate results. For this reason, it is often used in conjunction with other quality indicators, as and subjective evaluation.

PSNR is a general purpose measure used to measure the integrity of an image or video signal after it has been compressed, transmitted or otherwise processed. It is widely used in applications such as video encoding, video streaming, image and video editing, and image compression.

PSNR is a relative measure of quality as it compares the original signal with the randomness or noise of that signal. The higher the PSNR value, the closer the transition is to the original signal in terms of pixel-by-pixel similarity. PSNR is also on a logarithmic scale, meaning a 10 dB change means a tenfold improvement in image quality. However, there are some limitations to using PSNR as a quality measure. One limitation is that it only detects the frame error between the original signal and the degraded signal and does not take into account the difference in human perception of image or video quality.

Another limitation is that PSNR assumes that the first signal has an infinite value, which is not always the case. This can result in PSNR values that do not represent the actual image or video. In addition, PSNR does not take into account factors such as color space, contrast and surface detail that may affect the quality of the image or video.

PSNR is not only used for evaluating image compression, but it can also be used for other tasks such as denoising, super-resolution, and image restoration. PSNR is a measure of the distortion between the original and the reconstructed image, and it is often used in conjunction with other metrics such as Structural Similarity Index (SSIM) and Visual Information Fidelity (VIF) to obtain a more comprehensive evaluation of image quality. The range of PSNR values depends on the bit-depth of the image. For example, in an 8-bit grayscale image, the PSNR values range from 0 to 48 dB, whereas in a 16-bit grayscale image, the PSNR values can reach up to 96 dB.

PSNR can be affected by image content and structure. Images with simple and uniform content, such as synthetic images, can achieve higher PSNR values compared to complex and natural images. This is because complex images have more details and textures that are difficult to represent without introducing some level of distortion. PSNR is a pixel-based metric that compares the pixel values of the original and the reconstructed images. This means that PSNR may not capture the differences in the overall appearance of the images, such as changes in color or contrast. PSNR is sensitive to the scaling of the image. Example, if an image is scaled down and reconstructed, the PSNR value may be higher than if the original image is reconstructed directly. This is because scaling down the image reduces the amount of information that needs to be compressed, resulting in a lower distortion.

PSNR can be influenced by the choice of the reference image. Sometimes the image used may not be the original image, but a compressed or degraded version of it. This can affect the PSNR and interpretation of the results. There are some variations of PSNR that take into account the perceptual characteristics of the human visual system. For example, the Structural Similarity Index (SSIM), Visual Information Fidelity (VIF) index are metrics that aim to correlate more closely with human perception. Overall, PSNR is a useful for evaluating the quality of reconstructed images, but it has some limitations and should be used in conjunction with other metrics and with careful consideration of the image content and structure.

5.4 Structural Similarity Index (SSIM)

The Structural Similarity Index (SSIM) is a widely used metric for assessing the similarity between two images. It measures the quality of an image by comparing its structural information, such as texture and edges, with that of a reference image. In image denoising, SSIM can be used as a cost function for noise reduction techniques. The purpose of image noise reduction is to remove noise from the image while preserving its main features. SSIM can be used to quantify the degree of distortion caused by the denoising process, and the optimal denoising algorithm can be found by minimizing the SSIM metric.

SSIM relies on three factors: luminance, contrast, and structure. The luminance factor represents the overall brightness of the image, while the contrast factor measures the difference in brightness between different parts of the image. The structure factor represents the correlation between the image pixels.

To use SSIM in image denoising, the noisy image is first compared to the reference image using the SSIM metric. Then, a noise reduction algorithm is applied to the noisy image to create a noise-free image. The denoised image is compared with reference image using SSIM metric again, then the difference between the two SSIM values is used as a cost function to improve the denoising algorithm. Overall, SSIM is good to evaluate the denoised images. However, it is important to note that SSIM is not a perfect metric and should be used in conjunction with other evaluation methods to obtain a more accurate assessment of image quality.

5.5 Image Enhancement Factor

Image Enhancement Factor (IEF) is a measure of the amount of noise reduction achieved by a denoising algorithm when storing detailed images and sharpness. It is used to evaluate the effectiveness of an image denoising algorithm. The IEF is calculated as the ratio of the mean square error (MSE) between the original image and the noisy image to the MSE between the original image and the denoised image. The IEF value is from 1 to infinity, a large value indicates better performance in noise removal while preserving image features.

IEF is used to compare the performance of different denoising techniques, with a higher IEF value indicating a better algorithm. However, it is important to note that IEF does not include the optimum view of the noise-free image and it is possible for an algorithm to achieve a high IEF value but produce an image that looks unnatural or unrealistic. In summary, IEF is a useful quantitative measure for evaluating the performance of image denoising algorithms. It provides a measure of how much the denoising algorithm improves the image quality while preserving its details, but it should be used in conjunction with other evaluation methods to obtain a complete assessment of the algorithm's performance

CHAPTER 6

EXPERIMENTAL RESULTS AND ANALYSIS

Chapter 6
Experimental Results And Analysis

6.1 MATLAB

MATLAB2015b is a software package for numerical computing and data analysis. It provides a programming environment and a range of tools and functions for solving mathematical problems, creating visualizations, and working with large data sets. The "b" in MATLAB2015b refers to the second release of the software for the year 2015. This version includes new features and enhancements such as improved performance and reliability, new functions for working with data, and updates to the user interface. MATLAB2015b is widely used in fields such as engineering, finance, and scientific research for its versatility and ease of use. It is a powerful tool used for a variety of engineering and scientific applications. MATLAB is developed by MathWorks in the late 1970s. Since then, it has become a widely used tool for data analysis, modelling, and simulation. MATLAB 2015b is packed with new features and improvements. It has a new graphics system that makes it easier to create high-quality plots and visualizations. This new system includes a new user interface that makes it easier to work with graphics. The interface has been updated to include new graphics objects that allow for more flexibility in creating plots and visualization.

MATLAB R2022b is the latest version of MATLAB, and it introduces several new features, including new and updated toolboxes for machine learning and deep learning, enhancements to the MATLAB language, graphics system, and performance. R2022b also includes new capabilities for working with data, such as the ability to work with out-of-memory data more efficiently.

In conclusion, each version of MATLAB offers a unique set of features and enhancements, with newer versions generally including more advanced features and improved performance. Depending on the specific needs of the user, different versions of MATLAB may be more suitable for specific tasks or applications.

6.2 Comparison from different algorithms

The median filter works by smoothing the image while preserving edges and other important features. Unlike linear filters such as mean filters, median filters are less sensitive to outliers and can effectively remove impulse noise and other types of non-Gaussian noise. In practice, the median filter is implemented by sliding a window over the image and calculating the median value of the pixels within the window. The size of the window can be adjusted depending on the level of noise in the image and the desired amount of smoothing.

One advantage of median filtering is that it can be applied to both one-dimensional and two-dimensional signals, making it a versatile tool for a wide range of applications. It is commonly used in medical imaging, computer vision, and remote sensing applications, among others. While median filtering is a useful technique for removing noise, it is important to note that it can also introduce some level of blurring or smoothing to the image. Therefore, the size of the filter and the level of noise in the image must be carefully considered to achieve the desired level of smoothing without sacrificing important features or details in the image.

Median filtering is a non-linear filtering technique that replaces each pixel value with the median value of its neighbouring pixels. In this article, we will discuss the principle of median filtering and how it is used as a representative algorithm for filtering denoising.

Gaussian noise is a popular technique for denoising that follows a Gaussian distribution. It is a linear filtering technique that uses a Gaussian function to blur the image while preserving edges and other important features. The Gaussian filter works by convolving the image with a Gaussian kernel, which is a matrix of weights that reflect the shape of a Gaussian function. The values of the weights are highest at the center of the kernel and decrease with distance from the center, resulting in a smoothing effect that preserves edges and details in the image.

The size of the Gaussian kernel and the standard deviation of the Gaussian function can be adjusted to control the level of smoothing and the amount of noise reduction. A larger kernel and a higher standard deviation will result in more smoothing and noise reduction, while a smaller kernel and a lower standard deviation will preserve more of the original details in the image.

One advantage of Gaussian filtering is that it is a linear and separable filter, meaning that it can be applied efficiently using convolutional operations. It is also a well-studied filter with a strong theoretical foundation, making it a reliable tool for image processing and computer vision applications.

While Gaussian filtering is a powerful technique for smoothing and reducing noise in images, it can also introduce some level of blurring or smoothing to the image. Therefore, the size of the kernel and the standard deviation of the Gaussian function must be carefully considered to achieve the desired level of smoothing without sacrificing important features or details in the image.

Denoising is an important task in image processing, computer vision, where goal is to remove unwanted noise from an image. Low-rank matrix denoising is a representative algorithm of matrix sparse denoising that uses matrix dimension reduction to obtain a sparse matrix. This article will explain the principle behind low rank matrix denoising and how it is used as a representative algorithm for matrix sparse denoising.

The principle of low-rank matrix denoising is based on the assumption that the noise in the image is random and uncorrelated, and the signal is correlated. The signal can be modelled as a low-rank matrix with some sparse noise, and the task of the denoising algorithm is to separate the signal from the noise. Low-rank matrix denoising uses matrix dimension reduction to obtain a sparse matrix. The rank of a matrix refers to the number of non-zero singular values of the matrix, and a low-rank matrix has a small number of non-zero singular values. Matrix dimension reduction techniques, such as singular value decomposition (SVD), can be used to obtain a low-rank matrix representation of the image. Once the low-rank matrix is obtained, the next step is to threshold the sparse noise. Thresholding is a technique that sets small values of the sparse noise to zero, effectively removing the noise from the matrix. The threshold value can be determined using various methods, like adaptive thresholding, hard thresholding, or soft thresholding.

Low-rank matrix denoising can also handle large-scale images efficiently, making it suitable for applications that require real-time processing. In conclusion, lowrank matrix denoising is a representative algorithm of matrix sparse denoising that uses matrix dimension reduction and thresholding to obtain a sparse matrix representation of the image. The technique is effective in removing various types of noise and can handle large-scale images efficiently. While there are several matrices sparse denoising algorithms available, low-rank matrix denoising is a robust and stable solution that can find application in a wide range of image processing scenarios.

To ensure the result of the experiment, Gaussian noise was used with a mean zero and increasing the variance values as the noise signal. The effect diagrams depicting the performance of various denoising algorithms were derived by analyzing Figures. First and foremost, the median filter denoising algorithm is used for reducing noise in images. This algorithm works by replacing the central pixel in a window with the median value of the surrounding pixels. The window then moves across the entire image, and the same processing is applied to each pixel. While the median filter is effective in removing noise from an image, it can also filter out important image information along with the noise. This is because the median filter essentially averages the values of neighbouring pixels, which can result in fuzzy and loss of detail in the image. This loss of detail can be especially noticeable in areas of the image with high contrast or fine detail.

Additionally, the low-rank matrix denoising algorithm principle treats images as sparse processing, where some matrix pixels are changed to zero to obtain a low-rank representation. The low-rank matrix denoising algorithm is based on the principle of sparse processing of image matrices. This algorithm works by setting some pixels in the matrix to zero to get a lower rank representation. Sparse image matrix processing is performed using a specific threshold. However, this threshold is not adaptive and does not consider whether a pixel is noisy or not. Low-rank matrix denoising algorithms can be more efficient than median filters in preserving image information during matrix reconstruction, but can also introduce more image distortion as the center of the image extends to the edges. This is because not only the noise information but also a large amount of image information is filtered out, which may reduce the effect of noise reduction.

Our method combines coarse and fine denoising to effectively reduce noise in images. The algorithm works in two stages. In the first stage, the image edge information is extracted and saved. Next, the singular value matrix is obtained through singular value decomposition. The singular value matrix is then processed using a percent threshold η, which helps to filter out noise while preserving important image details. The first stage ends with singular value inverse decomposition, which results in a rough denoising of the image.

In the second stage, the superpixel-like algorithm is introduced to perform fine denoising on the image. Different processing methods are applied to different areas of the image based on their characteristics. This approach allows for more targeted denoising, resulting in a cleaner and more visually pleasing image. The two-stage image denoising algorithm based on noise localization is a highly effective method for reducing noise in images while preserving important details. One of the strengths of this algorithm is that it uses different processing methods based on the characteristics of different areas of the image. For example, in the "Dark Area" and "Light Area" of the image where there are fewer details, noise information is determined in units of templates. This allows the algorithm to effectively identify and filter out noise in these areas. In contrast, in the "Gray Area" of the image where there are more details, noise information is determined in units. This approach allows for more precise noise reduction, while still preserving the important image details. The fine denoising process is completed by deleting the noise information using superpixels. Superpixels are small regions of the image that are grouped together based on their similarity in colour and texture. By using superpixels, the algorithm is able to selectively remove noise while preserving the underlying structure and details of the image. This helps to ensure that the final denoised image is of high quality and retains as much important information as possible. In addition to the advantages mentioned earlier, the proposed two-stage image denoising algorithm based on noise localization also outperforms the artificial bee colony denoising algorithm in terms of denoising effectiveness. The artificial bee colony algorithm tends to retain too much noise information, which can be observed in the denoising effect diagrams in Figs. 6 to 9. On the other hand, the proposed algorithm effectively filters out most of the noise information while preserving the image information, resulting in a significant improvement in the denoising effect.

Furthermore, compared with the median filter and low rank matrix denoising algorithms, the proposed algorithm achieves a better balance between denoising effectiveness and image information preservation. The proposed algorithm's fine denoising step, which utilizes a superpixel-like algorithm, further contributes to improving the denoising effect by accurately identifying areas with different levels of detail and adapting the denoising process accordingly. From the below table, we can observe that the proposed algorithm is better than the many noise reduction techniques. By keeping the Gaussian mean constant and changing the variance we observe the PSNR value, IEF and SSIM for different noise reduction techniques. PSNR value of proposed algorithm is greater the median filtering, Low Rank Matrix denoising, Gaussian filtering. By changing variance, we observe that proposed algorithm gives the best PSNR value than compare to PSNR values of other denoising algorithms. SSIM is better for our proposed algorithm which gives 0.631. IEF values are generally increasing, which may indicate that some aspect of the image is being enhanced or improved as the parameter value increases.

Table 6.2 Comparison of PSNR, SSIM and IEF for different denoising algorithms of cameraman image

variance	Median filtering			Gaussian filtering			Low rank matrix denoising			Proposed algorithm		
	Psnr	SSIM	IEF	Psnr	SSIM	IEF	Psnr	SSIM	IEF	PSNR	SSIM	IEF
0.01	20.599	0.290	5.255	20.725	0.438	3.615	20.356	0.338	1.000	27.332	0.631	4.958
0.02	20.566	0.290	4.678	20.746	0.438	3.364	18.008	0.303	1.001	26.757	0.547	8.477
0.03	20.531	0.289	4.011	20.743	0.437	3.034	16.583	0.289	1.001	25.638	0.492	9.538
0.04	20.510	0.290	3.383	20.686	0.438	2.713	15.581	0.282	1.001	24.703	0.452	9.968
0.05	20.488	0.287	2.889	20.642	0.438	2.402	14.812	0.278	1.002	23.979	0.417	10.251
0.06	20.465	0.288	2.510	20.657	0.437	2.163	14.220	0.275	1.002	23.354	0.391	10.355
0.07	20.459	0.287	2.211	20.613	0.437	1.955	13.715	0.274	1.002	22.811	0.370	10.330
0.08	20.438	0.285	1.992	20.627	0.435	1.817	13.308	0.272	1.002	22.347	0.352	10.354
0.09	20.421	0.285	1.819	20.592	0.435	1.685	12.949	0.272	1.002	21.932	0.336	10.229
0.1	20.415	0.286	1.688	20.604	0.434	1.583	12.648	0.272	1.002	21.538	0.323	10.030

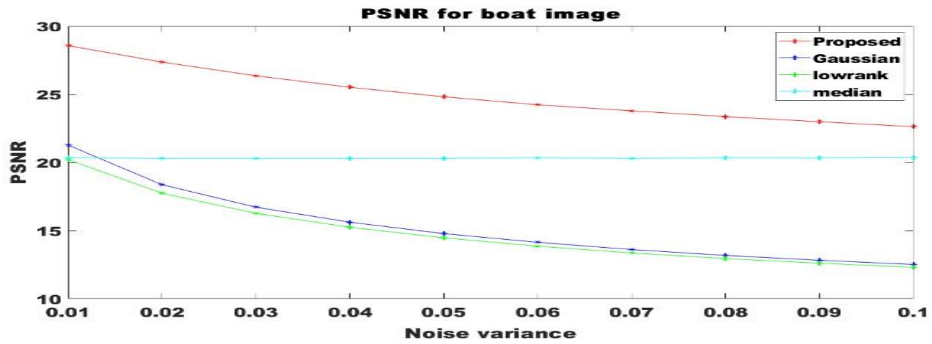


Fig 6.2.1 plot of PSNR and variance for boat image

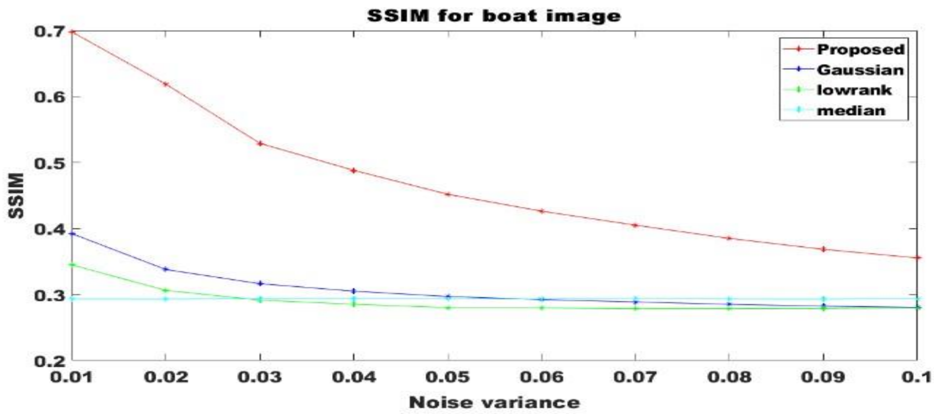


Fig 6.2.2 plot of SSIM and variance for boat image

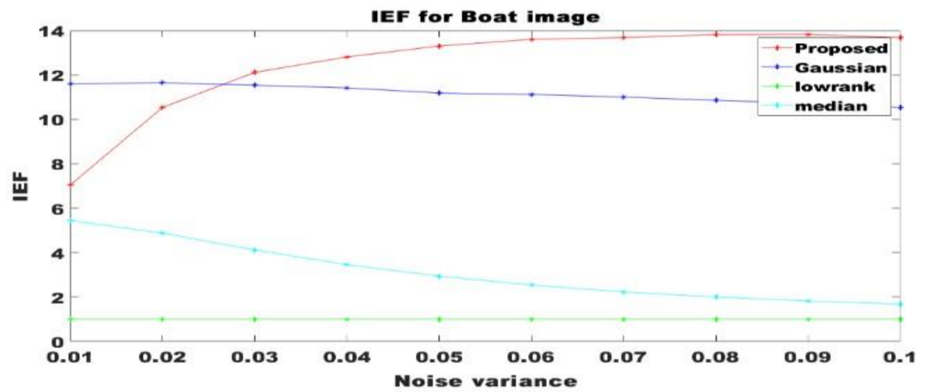


Fig 6.2.3 plot of IEF and variance for boat image

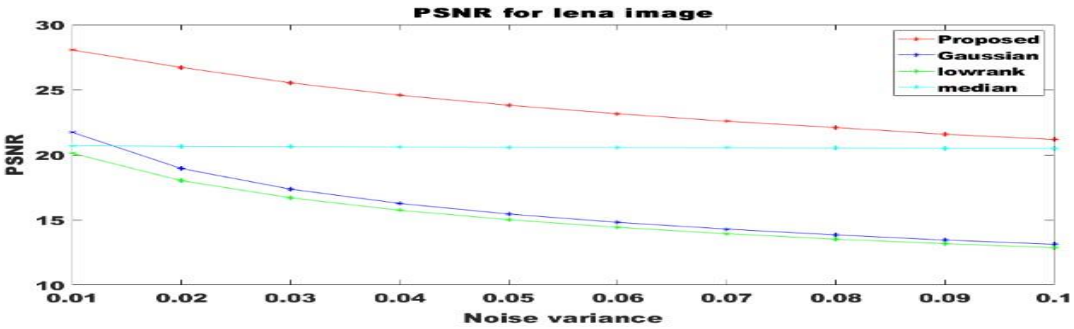


Fig 6.2.4 plot of PSNR and variance for lena image

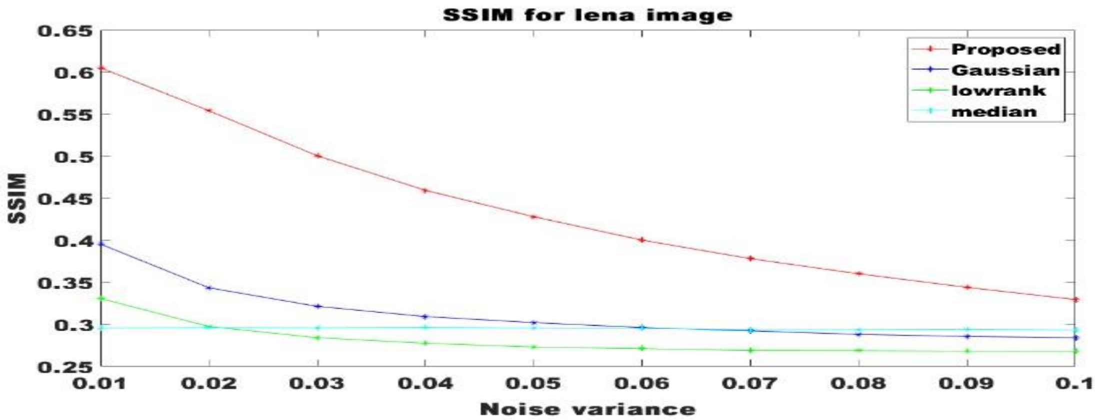


Fig 6.2.5 plot of SSIM and variance for lena image

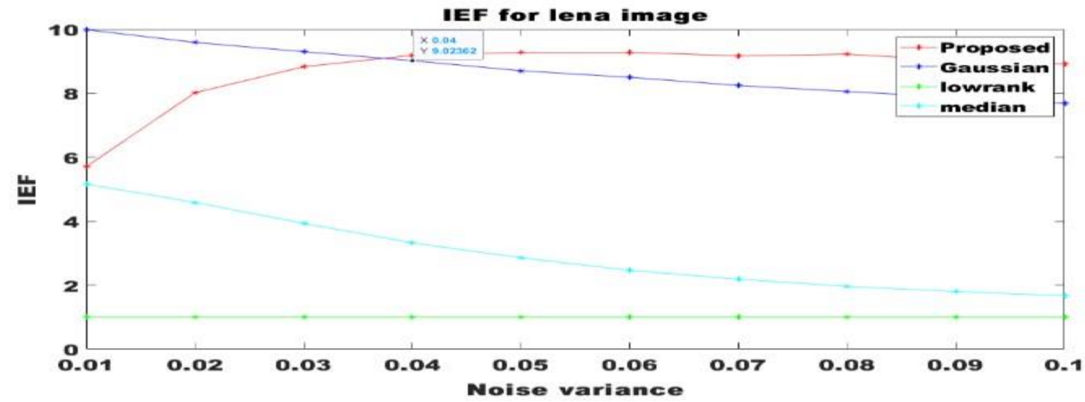


Fig 6.2.6 plot of IEF and variance for lena image

(a)

(b)

(c)

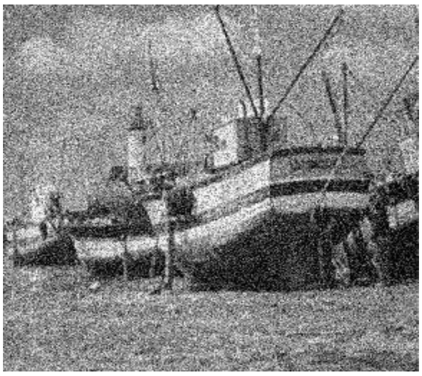
(d)

(e)

(f)

Fig 6.2.7 Comparison of cameraman image on different denoising algorithms
(a)Original image (b) noisy image (c) Median filtering (d) Gaussian filtering (e)Low rank matrix filtering (f) Proposed Algorithm

(a)



(b)

(c)

(d)

(e)

(f)

Fig 6.2.8 Comparison of boat image on different denoising algorithms

(a)Original image (b) noisy image (c) Median filtering (d) Gaussian filtering (e) Low rank matrix filtering (f) Proposed Algorithm

(a)

(b)

(c)

(d)

(e)

(f)

Fig 6.2.9Comparison of lena image on different denoising algorithms

(a)Original image (b) noisy image (c) Median filtering (d) Gaussian filtering (e) Low rank matrix filtering (f) Proposed Algorithm

CONCLUSION

In this method first divides the image into “Dark Area”, “Gray Area” and “Light Area” by adaptive threshold method, then extracts the image edge information to obtain the edgeless noise image. Dividing an image into "Dark Area", "Gray Area", and "Light Area" is a technique commonly used in image processing and computer vision. This technique is called image segmentation, where an image is partitioned into multiple segments or regions based on certain characteristics such as color, texture, and intensity. The segmentation is based on the brightness or intensity of the pixels. The dark, gray, and light areas represent regions of the image with different intensity values. The edges of an image can be detected by analyzing the discontinuities or changes in the intensity values between adjacent pixels.

The process of extracting the edge information from the image involves applying a filter, such as the Sobel filter, to identify the edges. The result is an edge map or image, which highlights the edges and boundaries in the original image while removing the noise and irrelevant details.The final result of this process, which involves segmenting the image into different areas based on intensity and then extracting the edge information, is an edgeless noise image. This image is useful for further processing or analysis, such as feature extraction or object recognition, as it contains only the relevant edges and features of the original image while removing the noise and irrelevant details.

Singular value decomposition (SVD) is a mathematical technique that decomposes a matrix into three components: the left singular vectors, the singular values, and the right singular vectors. In image processing, SVD is used to reduce the rank of the image matrix by keeping only the largest singular values and their corresponding singular vectors. This is based on the observation that noise usually has a small effect on the largest singular values of an image, while the smaller singular values are more affected by noise. To perform coarse noise filtering using SVD, the original image is first transformed into a zero matrix, except for the first k singular values, which are set to their original values. The remaining singular values are set to zero. This process is equivalent to removing the high-frequency components of the image, which are most affected by noise.

The resulting modified image is then transformed back to its original representation using the inverse SVD. This process effectively filters out the high frequency noise components while preserving the low-frequency components of the image. The threshold η mentioned is used to control the number of singular values retained in the transformed image. A higher value of η would result in fewer singular values being retained, which would result in a more aggressive noise filtering, but at the risk of losing some image details. Overall, coarse noise filtering using SVD is a useful technique for reducing noise in images, especially when combined with other denoising methods such as the one you described earlier. By removing high-frequency noise components and preserving low-frequency image details, SVD-based denoising can improve the visual quality of images while preserving their overall structure and content. The new image is changed to zero matrices except the first k by threshold η, then the coarse noise filtering is completed by singular value inverse decomposition.

Then apply the superpixel-like algorithm for locating the noisy pixels of the image. The image is divided into “Dark Area”, “Light Area” and “gray area”. In the “Gray Area”, compare threshold with zero then update the pixel value. In Dark area and Light area the pixel values are updated by comparing the numbering of noise pixels with calculated N value. Repeating the above process to the whole pixels in the image. Then denoised image is obtained. By observing the result, the proposed algorithm shows the better results than other denoising algorithms.

REFERENCES

[1]. Amarjit R, Hussain LR (2019) Fuzzy SVM based fuzzy adaptive filter for denoising impulse noise from color images. Multimed Tools Appl 78(2):

[2]. Chen GY, Bui TD, Krzyzak A (2008) Image denoising using neighbouring wavelet coefficients. Integrated Computer Aided Engineering 12(1):99–107

[3]. Donoho DL (1995) De-noising by soft-thresholding. IEEE Trans Inf Theory 41(3):613–627

[4]. Donoho DL, Johnstone JM (1994) Ideal spatial adaptation by wavelet shrinkage. Biometrika 81(3):425–455

[5]. Elad M, Aharon M (2006) Image denoising via sparse and redundant representations over learned dictionaries. IEEE Transactions on Image Processing 15(12):3736–3745

[6]. Fan L, Zhang F, Fan H, Zhang C (2019) Brief review of image denoising techniques. Visual Computing for Industry, Biomedicine, and Art 2(1):1–12

[7]. Fodor IK, Kamath C (2003) Denoising through wavelet shrinkage: an empirical study. Journal of Electronic Imaging 12(1):151–161

[8]. Goyal B, Dogra A, Agrawal S et al (2020a) Image denoising review: From classical to state-of-the-art approaches. Information Fusion 55:220–244

[9]. Goyal B, Dogra A, Agrawal S, Sohi BS, Sharma A (2020b) Image denoising review: From classical to stateof-the-art approaches. Information Fusion 55:220–244

[10]. Hasan M, El-Sakka M R (2018) Improved BM3D image denoising using SSIMoptimized Wiener filter. EURASIP Journal on Image and Video Processing 2018(1):1–12

[11]. Huang Q, Hao B, Chang S (2016) Adaptive digital ridgelet transform and its application in image denoising. Digital Signal Processing 52:45–54

[12]. Huang F, Lan B, Tao J et al (2017) A Parallel Nonlocal Means Algorithm for Remote Sensing Image Denoising on an Intel Xeon Phi Platform. IEEE Access 99:1–1

[13]. Karaboga D (2010) Artificial bee colony algorithm. Scholarpedia 5(3):6915

[14]. Konieczka A, Balcerek J, Dąbrowski A (2013) Iterative average filtering for image denoising. In: Signal Processing: Algorithms. Architectures, Arrangements, and Applications

[15]. Kumar S, Panigrahy D, Sahu PK (2018) Denoising of Electrocardiogram (ECG) signal by using empirical mode decomposition (EMD) with non-local mean (NLM) technique. Biocybernetics and Biomedical Engineering 38(2):297–312

[16]. Lebrun M (2012) An Analysis and Implementation of the BM3D Image Denoising Method. Image Processing on Line 2(25):175–213

[17]. Li H, Suen CY (2016) A novel Non-local means image denoising method based on grey theory. Pattern Recognition 49(1):237–248

[18]. Li B, Liu QS, Xu JW et al (2011) A new method for removing mixed noises. Science China (Information Sciences) 54(1):51–59

[19]. Liu G, Guo W (2010) Application of improved arithmetic of median filtering denoising. In: Computer Engineering & Applications

[20]. Liu R, Tan T (2002) An SVD-based watermarking scheme for protecting rightful ownership. IEEE Transactions on Multimedia 4(1):121–128

[21]. Lu CT, Chou TC (2012) Denoising of salt-and-pepper noise corrupted image using modified directionalweighted-median filter. Pattern Recognition Letters 33(10):1287–1295

[22]. Luo L, Zhao ZQ, Li XP et al (2019) A stochastic image denoising method based on adaptive patch-size. Multidimensional Systems and Signal Processing 30(2):705–725

[23]. Mafi M, Martin H, Cabrerizo M et al (2019) A comprehensive survey on impulse and Gaussian denoising filters for digital images. Signal Processing 157:236–260

[24]. Min L, Peng F, Zhu Y (2019) Identifying natural images and computer generated graphics based on binary similarity measures of PRNU. Multimed Tools Appl 78(1):489–506

[25]. Mohammad AA, Al-Haj A, Shaltaf S (2008) An improved SVD-based watermarking scheme for protecting rightful ownership. Signal Processing 88(9):2158–2180

[26]. Orchard J, Ebrahimi M, Wong A (2008) Efficient nonlocal-means denoising using the SVD. In: IEEE International Conference on Image Processing

[27]. Rohit U, Abdu RV, George SN (2017) A robust face hallucination technique based on adaptive learning method. Multimed Tools Appl 76(15):16809–16829

[28]. ShouDong H, Yong Z et al (2011) Gaussian super-pixel based fast image segmentation using graph cuts. Acta Automatica Sinica 37(1):11–20

[29]. Song-Tao L, Zhen-Xing L, Ning J (2018) Target segmentation of infrared image using fused saliency map and efficient subwindow search. Acta Automatica Sinica 12:2210–2221

[30]. Thanh DNH, Hai NH, Prasath S et al (2020) A two-stage filter for high density salt and pepper denoising. Multimed Tools Appl:1–23

[31]. Umam AK, Yunus M (2018) Quaternion wavelet transform for image denoising. Journal of Physics. Conference Series 974(1):12006

[32]. Wang XH, Istepanian RSH, Yong HS (2003) Microarray image enhancement by denoising using stationary wavelet transform. IEEE Transactions on Nanobioscience 2(4):184–189

[33]. Yahya AA, Tan J, Su B et al (2020) BM3D image denoising algorithm based on an adaptive filtering. Multimed Tools Appl 79(4):1–37

[34]. Yang H-Y, Wang X-Y, Niu P-P, Liu Y-C (2014) Image denoising using nonsubsampled shearlet transform and twin support vector machines. Neural Networks 57:152–165 14120 Multimedia Tools and Applications (2021) 80:14101–14122

[35]. Zhang R, Feng X, Yang L, Chang L, Zhu X (2019) A global sparse gradient based coupled system for image denoising. Computers and Mathematics with Applications 78(11):3692–3711

[36]. Zhao G, Liu J, Jiang J et al (2018) A deep cascade of neural networks for image inpainting, deblurring and denoising. Multimed Tools Appl 77(22):29589–29604

PAPER PUBLICATION DETAILS

ICRAC-23 (author)

New Submission | My Submissions | Conference | News | EasyChair

ICRAC-23 Submission 7049

If you want to **change any information** about your paper, use links in the upper right corner.

For all questions related to processing your submission you should contact the conference organizers. Click here to see information about this conference.

Submission 7049	
Title	A Two Stage Image Denoising Using Superpixel Algorithm
Paper:	(Apr 11, 02:08 GMT)
Author keywords	Edge extraction Singular value decomposition adaptive threshold percentage threshold
Abstract	The majority of denoising algorithms are currently unable to distinguish between pixels with noise and those witho pixels, these algorithms continue to use the same consistent principles. When used on photographs with tiny deta the subject and the backdrop, denoising techniques frequently result in the loss of original image information. The stage noise localization-based picture denoising technique to address the aforementioned issues. The distributior is used to determine thresholds T'1 and T'2 in the initial step. The technique includes two steps to achieve denoisi extracted in order to preserve it, and then singular value decomposition is used to recover the singular value matri edgeless grayscale picture. Lastly, a percentage threshold is used to compress the singular value matrix. The distr image is examined to determine thresholds T'1 and T'2 in the first stage. The second stage involves creating an ec employing edge extraction to remove and save the image edge information. The edgeless picture is then subjecte decomposition in order to get the singular value matrix, which is then compressed using a percentage threshold fc suggested method uses inverse matrix decomposition to carry out coarse noise filtering. Moreover, the system us thresholds that are gleaned from the picture histogram. Based on these criteria, the picture is divided into the "Dar "Light Area" sections. The result for the suggested method is created by merging the denoised picture with the ima the peak signal-to-noise ratio (PSNR), structural similarity index, and image enhancement factor of the suggested cutting-edge denoising algorithms, a comparative study of the proposed approach is conducted. The outcomes sh method is superior at denoising different pictures.
Submitted	Apr 11, 02:08 GMT
Last update	

Authors				
first name	last name	email	country	affiliation
Bhaskara Rao	Jana	janabhaskar@gmail.com	India	Anil Neerukonda Institute of Technology and Sciences
Yaswanth Ganesh	T	yaswanthganesh.2019.ece@anits.edu.in	India	Anil Neerukonda Institute of Technology and Sciences
Lakshmi Madhumitha	P	lakshmimadhumitha.2019.ece@anits.edu.in	India	Anil Neerukonda Institute of Technology and Sciences
Manikanta Sai	E	manikantasai.2019.ece@anits.edu.in	India	Anil Neerukonda Institute of Technology and Sciences
Jagadeeswar	T	jagadeeswar.2019.ece@anits.edu.in	India	Anil Neerukonda Institute of Technology and Sciences