Educational Administration: Theory and Practice
2024,30 (6), 4479-4494
ISSN:2148-2403
https://kuey.net/
Research Article

MRI Image Denoising Influenced By Iterative Hint Pixel Searching Through Multi-Order Neighbours And WSDHM Based Dimensionality Reduction

S. Prathiba[1*] ,B. Sivagami[2]

[1*]Research Scholar, Department of Computer Science & Research Centre, S.T. Hindu College, Nagercoil – 629002, Affiliated to Manonmaniam Sundaranar University, Abishekapatti, Tirunelveli – 627012, Tamil Nadu, India.
[2]Associate Professor, Department of Computer Science & Application, S.T. Hindu College, Nagercoil – 629002, Affiliated to Manonmaniam Sundaranar University, Abishekapatti, Tirunelveli – 627012, Tamil Nadu, India.

***Corresponding author**: S. Prathiba
*(prathibasuyambu26@gmail.com).

Citation: S. Prathiba ,et.al (2024), MRI Image Denoising Influenced By Iterative Hint Pixel Searching Through Multi-Order Neighbours And WSDHM Based Dimensionality Reduction, Educational Administration: Theory and Practice, 30(6), 4479-4494
Doi: 10.53555/kuey.v30i6.7442

ARTICLE INFO **ABSTRACT**

An essential tool in medical diagnostics is Magnetic Resonance Imaging (MRI), but its effectiveness is often compromised by noise, particularly impulse noise (or salt-and-pepper noise), which can obscure vital image details. This research proposes a denoising method entitled 'MRI image denoising using Iterative Hint Pixel Searching through Multi-Order Neighbors and WSDHM-based Dimensionality Reduction (IHSMW)', which uniquely combines a new hint pixel searching algorithm and a new hint-data dimensionality reduction algorithm. A novel algorithm for hint pixels searching is designed which is named 'Four Corners based Multi-order supported Iterative Hint Pixel searching algorithm (FCMIHP)'. A new data dimensionality reduction technique known as Weighted Shortest Distance Hierarchical Mapping (WSDHM) that significantly improves the denoising process is also designed. The proposed IHSMW filter excels in preserving essential structural details in grayscale brain MRI images, thus enhancing the diagnostic potential, especially in brain cancer detection and segmentation. The proposed method is validated against traditional denoising techniques and demonstrates superior performance in preservation of crucial image components while noise is significantly reduced. The experimental findings reveal that the proposed IHSMW method using the PRNV-DB database gives a better PSNR value of 31.55% and the lowest MSE value of 45.507%, which means it gives a better result than existing methods.

Keywords: Magnetic Resonance Imaging (MRI), Denoising, Iterative Hint Pixel Searching, Noise Reduction, Weighted Shortest Distance Hierarchical Mapping (WSDHM).

1 INTRODUCTION

In general, diagnosing brain cancer usually begins with Magnetic Resonance Imaging (MRI) (Brain MRI 2023). Computer vision algorithms can be applied to medical imaging data, such as brain MRI scans, to automatically detect and segment brain cancer (Shinde et al. 2022). These algorithms can identify regions of interest, quantify cancer characteristics (e.g., size, shape), and assist radiologists in making accurate diagnoses. These medical diagnosis processes meet the challenge when the MRI image is corrupted by impulse noise, which affects the accuracy of medical diagnosis results. Salt and Pepper noise is a type of noise that appeared on images. This type of noise is also known as impulse noise. This type of noise is generated by the image errors that occur due to the contamination of digital images at the time of acquisition. Several noise types are available, and they are Gaussian noise, Random noise, Poisson noise, Speckle noise, etc. Among all of these noises, the salt and pepper noise is the most dangerous one, because it is the dominant noise type which corrupts the image contents severely through the sources such as Wireless communication, Device ageing, and Device malpractice. The existing methods of noise reduction are suffered by lack of Peak Signal to Noise Ratio, blurring and high complexity. Therefore, in this research, a denoising method for removal of impulse noise in MRI images is proposed.

The proposed denoising method is named IHSMW filter. The input for this method is the MRI brain grayscale image. This method is constituted based on the following two contributions:

- A new hint pixel searching algorithm namely 'Four Corners based Multi-order supported Iterative Hint Pixel searching algorithm (FCMIHP)'
- A new Neighbour data dimensionality reduction algorithm namely 'Data dimensionality reduction mechanism using Weighted Shortest Distance and Histogram peak based Majority (WSDHM)'.

The novel IHSMW filter enhances the brain MRI images by eliminating the 'impulse noise' or 'salt and pepper noise' from the corrupted MRI. This method is enriched by the FCMIHP algorithm and the WSDHM algorithm. The FCMIHP algorithm effectively gathers the neighbour hint pixels (i.e., non-noisy neighbour pixel) using four corners based searching along with multi iterations and orders. The WSDHM algorithm mitigates the length of the linear neighbour hint vector using shortest distance and histogram peak based majority process. This section explains the proposed IHSMW denoising method deeply.

2 LITERATURE SURVEY

In recent days, several frameworks were introduced by the researchers, primarily to increase the accuracy of MRI image denoising. In this section, a few of them frameworks are briefly examined.

In 2019 Enginoglu, et al., Depending on the source or the transmission channel of the image, it can be affected by different types of noises or combinations. Among the noise types, additive noise, multiplicative noise, and impulse noise are the foregrounds. Impulsive noise is a general term that refers to a variation in brightness in some pixels of an image. Salt-and-pepper noise is a typical type of impulse noise (Li et al. 2020). Noise corruption mainly occurs during the process of acquisition and transmission (Zhang, et al. 2018). Images usually corrupt pixels due to faulty memory locations in hardware, malfunctioning pixels in the camera sensor, scanning machine sensor, or transmission in a noisy channel (Kimiaei, et al. 2019). In this type of noise, some pixels of a digital image have a maximum (i.e., intensity value of 255) or minimum value (i.e., intensity value of 0) (Bai 2014, Chen 2018).

Removing noise while preserving image details and textures is one of the most important and fundamental issues in image processing. As an essential pre-processing step, image denoising algorithms are also widely used in computer vision, pattern recognition, and medical image analysis fields (Fu et al. 2019). A medical image diagnosis helps in preplanning the surgery or treatment of the patient. MRI images are frequently affected by salt and pepper noise, which disturbs the post-processing of segmentation and classification that finally yields a low-quality output (Gupta et al. 2017).

Magnetic Resonance Images are corrupted by salt and pepper noise, mainly due to sensor faults in image acquisition devices and sudden disturbances in the image signal, which in turn degrade the image quality by the formation of artifacts and blurring in MR images (Ali 2016, Alrabai 2021, Ebrahimnejad et al. 2021). Salt and pepper noise usually brings more obvious visual interference (black or white pixels) (Fu et al. 2019); hence, it should be removed to enhance the visual quality of medical images. The removal of impulse noise includes spatial domain approaches such as median (Gonzalez et al. 2002) and Adaptive Median Filtering (AMF) (Hwang et al. 1995), transform domain methods like wavelet denoising (Sree et al. 2013), fuzzy based approaches like Adeli et al. (2012), and neural-network based approaches like Xing et al. (2019) and Yi et al. (2020).

Vargas et al. (2018) put forth a denoising scheme based on the nonlinear filters, which consists of a re-descending M-estimator based on Hampel's three-part re-descending influence function to control the contribution of each pixel in the calculation of the best estimation of a noiseless pixel. In order to suppress high-density fixed-value impulse noise in grayscale images of larger size, this method is implemented on a heterogeneous CPU-GPU architecture. The advantage of this method is that it not only achieves better noise suppression but also has better image detail preservation. The demerit of this method is that it demands a high computational cost if it has to be used in real time. Singh et al. (2020) developed a spatially adaptive image denoising method through an enhanced noise detection method (SAID-END) for grayscale and color images. This method is implemented by means of two algorithms, which are Enhanced adaptive noise detection and non-corrupted pixel sensitive adaptive image restoration. The benefit of this method is that it uses an adaptive window mechanism with non-corrupted pixel ratio criteria, which provides the ability to process a huge noise level. The pitfall of this method is that the preservation of image details is not achieved properly.

Guanyu et al. (2021) achieved the restoration of noisy images by Distribution Transformed Network (DTN), which used the Convolutional Neural Network (CNN) to study the pixel-distribution features from noisy images. The goodness of using this method is that in the light-weight structure, the DTN performs better than the existing methods. The disadvantage of this method is that it shows better results only for images of smaller size, and its efficiency diminishes while applying it for larger size images. Hien et al. (2022) exposed a noise removal method for salt and pepper noise (SPN) using thresholding and regularization techniques. The core concepts set in this method are Total Variation (TV) based regularization and characteristics of SPN, in addition with Nesterov optimal method. Standard grayscale images and color images are used as sources for testing purposes. The better structural similarity is the advantage of this method. The demerit is that the performance quality of denoising is affected by block-artifacts when it is applied at huge noise level. The denoising of salt and pepper noise improves the quality of MRI images, leading to better cancer segmentation and reducing the false-segmentation ratio. Also, the existing denoising algorithms suffer from high time consumption, blur issue, and lower Peak Signal to Noise Ratio (PSNR). Hence, this research concentrates on removing salt and pepper from brain MRI images.

3 PROPOSED METHODOLOGY

The MRI brain cancer image is processed to identify the noise locations and hint pixels where hint pixels are searched from neighbor regions using concepts like multi-iteration, multi orders, and soft thresholding, The dimensionality of hint pixels is reduced using weighted schemes, and the predicted data is used to denoise the noisy pixel. Figure 1 shows the overall diagram of the proposed IHSMW filter.

3.1 Noise detection

An input grayscale brain MRI image in the size of 512 x 512 is processed to detect the noisy location corrupted by salt and pepper noise.

A noisy location is indicated by the extreme intensity value 255 and the minimum intensity value 0. The numeric 255 points out the salt noise (or white noise), while the numeric 0 points out the pepper noise (or black noise). The noisy status image is generated to localize the noisy positions in the input image using Equation (1).

$$I_{NS}^{i,j} = \begin{cases} 0, & if\ I_{IP}^{i,j} = 0 \mid I_{NS}^{i,j} = 255 \\ 1, & else \end{cases} \quad (1)$$

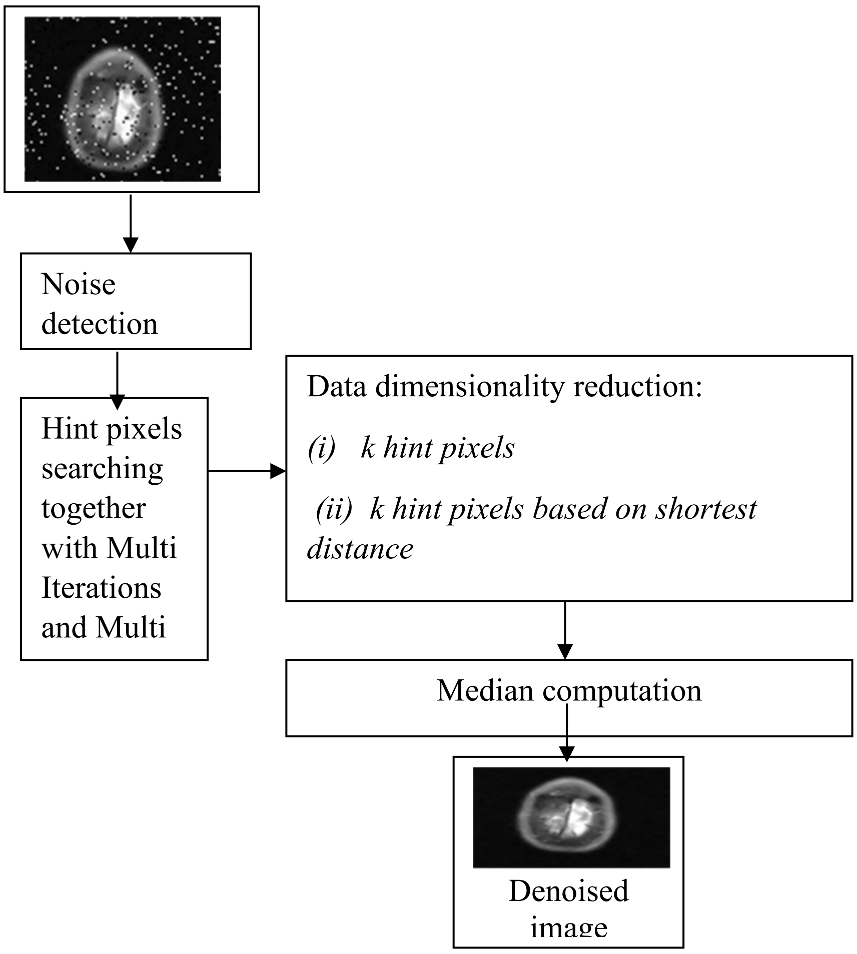
Noise detection
Hint pixels searching together with Multi Iterations and Multi
Data dimensionality reduction:
(i) k hint pixels
(ii) k hint pixels based on shortest distance
Median computation
Denoised image

Fig.1: Overall diagram of the proposed IHSMW filter.

Herein, the term I_{IP} refers to the Input image, and I_{NS} refers to Noisy Status image. In Equation (1), the noisy status image indicates the numeric value 0 for noisy locations where as it indicates 1 for non-noisy locations.

3.2 Hint pixels searching based on FCMIHP algorithm

This section removes the noise from the input image I_{IP} using the proposed IHSMW filter, and produces a noise-free image I_{NF}. The working process of the IHSMW filter is explained in Figure 2.

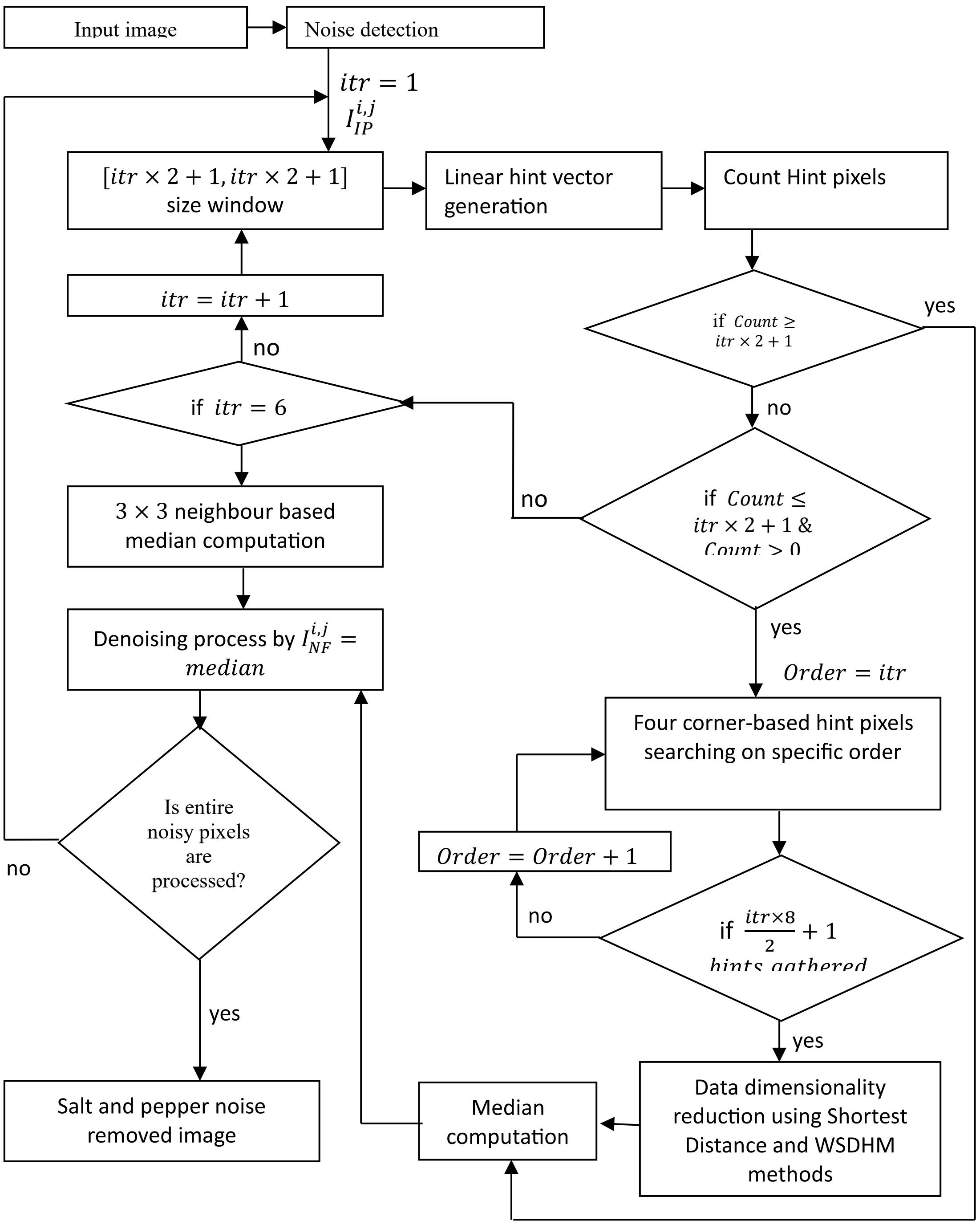
Input image
Noise detection
$itr = 1$
$I_{IP}^{i,j}$
$[itr \times 2 + 1, itr \times 2 + 1]$ size window
Linear hint vector generation
Count Hint pixels
if $Count \geq itr \times 2 + 1$
yes
no
$itr = itr + 1$
no
if $itr = 6$
no
if $Count \leq itr \times 2 + 1$ & $Count > 0$
3 × 3 neighbour based median computation
Denoising process by $I_{NF}^{i,j} = median$
yes
$Order = itr$
Four corner-based hint pixels searching on specific order
Is entire noisy pixels are processed?
no
$Order = Order + 1$
no
if $\frac{itr \times 8}{2} + 1$ hints gathered
yes
yes
Salt and pepper noise removed image
Median computation
Data dimensionality reduction using Shortest Distance and WSDHM methods

Fig.2: Work flow diagram of the proposed IHSMW filter.

The current processing pixel of $[i,j]^{th}$ location undergoes the noise removal process using iteration *iteration-1* with a 3x3 window. The non-noise pixels or hint pixels in the 3x3 window are collected, and the count of them is found. If the hint pixel count C is adequate to do the direct median computation, then the median value is computed, and it is used to remove the noise. If the hint pixel count C is inadequate to do the direct median computation, then the additional hint pixel information is gathered from multi-order neighbours by implementing the four-corner-based hint pixel searching algorithm to satisfy the soft threshold, which is parameterized by the iteration level.

The collected hint pixels undergo the data dimensionality reduction process to reduce the length of the hint pixel array. The length-reduced hint pixel array is used to predict the noisy location-oriented data via median computation. If the hint pixel count C is equal to 0, it means the unavailability of hint pixels. Hence, the hint pixel searching is directed to iteration-2. The same process is done on iteration-2, and if needed, the iteration-3 is initiated. This process is progressed until the predicted median value is obtained or completing the iteration-6. This phenomenon is applied to the entire number of noisy pixels in the I_{IP} image.

3.2.1 Iteration 1

The Input image I_{IP} is fed as input to this module. The image height is noted as h and the image width is noted as w. The $[i,j]^{th}$ noisy pixel is considered as the source to the process of noise removal. The variable itr is assigned as $itr = 1$. This iteration tries to obtain the predicted noise-free pixel using the window size of $[itr \times 2 + 1, itr \times 2 + 1]$ that is equivalent to (3x3), which is extracted by considering the $[i,j]^{th}$ noisy pixel as the center pixel. This window extraction process is presented using Equation (2).

$$W_{(itr\times2+1)\times(itr\times2+1)}^{m+itr,n+itr} = I_{IP}^{i+m,j+n} \quad (2)$$

$m \in [-itr, +itr], n \in [-itr, +itr]$

Equation (2) can be simplified in Equation (3)

$$W_{3\times3}^{m+1,n+1} = I_{IP}^{i+m,j+n} \quad (3)$$

$$m \in [-1, +1], n \in [-1, +1]$$

where

W - Window that is extracted in 3x3 sizes

The noisy elements of the 3x3 window is trimmed and the remaining non-noisy elements are collected together into a linear array namely Hint array. It can be represented using Equation (4)

$$HA = Func\_TrimmedLinearConversion\ (W_{3X3}) \quad (4)$$

where

HA - Linear hint array

$Funct\_Trimmed\ Linear\ Conversion()$ - Function to do trimming as well as linear conversion

In Equation (4), the trimmed output is converted into linear array form and stored in HA. The length of hint array is stored in l. Hence, it contains the valid range of hint data from $HA^{0}\ to\ HA^{l-1}$.

In the current situation, there are three possibilities for predicting the noise-free pixel. They are:

Case 1:

There may be sufficient pixels in the neighbour window, meaning that, the HA contains adequate pixels to compute the median computation. For example, a 3x3 window may possess a maximum of eight neighbours which are apt for median computation. But the partial availability of neighbours can yield fewer pixels than the maximum-available-eight pixels. There is a minimum hint pixel requirement so that the median computation is meaningful, and it can be decided as 3 for a 3x3 window. This threshold thr is fixed as $thr = 3$ and it can be written in general form as $thr = itr \times 2 + 1$. Hence, if the hint pixel count l is greater than or equal to thr (i.e., $l \geq thr$), then the median computation is done using the HA and the predicted median value is used to replace the noisy pixel. Afterwards, the iterative process is terminated to progress the denoising of the next-coming noisy pixel.

Case 2:

There may be a possibility of more linear pixels than the thr (i.e., $l < thr$), which is not an optimum for the median computation. Therefore, the quality of the hint pixels is leveraged by finding hint pixels to obtain a sufficient range of hint pixels based on the context of l upto 5 for a 3x3 window. It can be generalized by l up-to $\left(\frac{itr \times 8}{2}\right) + 1$, which can be expressed by $\left(\frac{1 \times 8}{2}\right) + 1 = 5$ for the iteration $itr = 1$. When the resultant count of hint pixels after the hint pixel searching process is equal to 5, then the progressively hint-searching process is terminated; otherwise, the search process progresses with next-order neighbour data. Herein, the search process includes elements of multi-order to have a set of hint pixels. The searched hint pixels are gathered, and they undergo a data dimensionality reduction process to reduce the size of the hint pixels. After getting the specified length of hint pixels, the median computation is performed, and that median value is used to replace the noisy pixel.

Case 3:

Sometimes, there may be an absence of non-noisy pixels in the entire portion of the surrounded 3x3 window, and in that situation, there are no hint pixels (or zero hint pixels), which cannot aid in the computation of the median. So, there is a need to recollect the hint pixels in another possible way, which is by searching for hint pixels using multi-iterations. It is possible to search hint pixels up to six-orders according to the center $[i,j]^{th}$ noisy pixel through *iteration 2* (generally $itr = itr + 1$).

The implementation model of these three cases is shown in the following algorithm.

IF $l \geq (itr \times 2 + 1)$ THEN

```
            $I_{INF}^{i,j} = Func\_Medain\ (HA)$    (5)
ELSE IF $l < (itr \times 2 + 1)\ \&\ l = 0$ THEN
$HA = Func\_CollectHintPixels\ (I_{IP}, itr, i, j)$    (6)
$I_{NF}^{i,j} = Func\_Medain\ (HA)$    (7)
ELSE IF $l = 0$
DO Iteration using $itr = itr + 1$
END
```

In the above algorithm, the term $Func\_Medain()$ refers to a function to compute the median. The term $Func\_CollectHintPixels()$ refers to a function to collect the hint pixels using multi-orders, four corner-based searching and data dimensionalities reduction via WSDHM method.
If the current pixel $[i, j]$ is processed by iteration-1 by having $l >= 3$, then the linear vector HA undergoes the median computation process. The steps of median computation are described in Equation (8) and Equation (9).

$$SHA = Func\_AscSort(HA) \qquad (8)$$

$$M = SHA^{fix\left(\frac{l}{2}\right)} \qquad (9)$$

where
SHA - Sorted Hint Array
$Func\_AscSort()$ - Function to sort the hint array in ascending order
M - Median value
Equation (8) makes the hint array in ascending order. Herein, the median value is computed by fixing the middle value such as $fix\left(\frac{l}{2}\right)$. Suppose $l = 8$, it provides the center location by 4 based on Equation (9). The median value is used to replace the noisy pixel using Equation (10).

$$I_{INF}^{i,j} = M \qquad (10)$$

If the current pixel $[i, j]$ is processed by iteration-1 by having the unsaturated quantity of hint pixels, then the hint pixel searching process progresses from 1st order elements to 6th order elements until the saturated level of hint pixels is gained. Herein, a new algorithm, namely the 'Four corner-based hint pixel searching method through multi order neighbours' is introduced to effectively find the saturated range of hint pixels. For 3x3 size windows, the available count of hint pixels is 1 to 2; hence they should be increased to reach the saturated range of hint pixels up to $\left(\frac{itr \times 8}{2}\right) + 1 = 5$ through the searching of additional hint pixels through multi orders. Herein, the term $\left(\frac{itr \times 8}{2}\right) + 1$ yields the soft threshold to compute the saturated level of hint pixels. The 1st order to 6th order elements are expressed in Figure 3.
In Figure 3, the white data shows the current $[i, j]^{th}$ pixel, and the green-colored positions show the 1st order elements. Like that, the red-colored position shows the 2nd order elements. The 4th, 5th, and 6th orders are also indicated in Figure 3 based on the blue, purple, yellow, and maroon colors, respectively. The hint pixel searching algorithm can be generalized by the statement that it uses each neighbour of the current window as a source to search for hint pixels in their own surrounding neighbour, and to concatenate them in the hint pixel array where no hint pixel is added twice or more. This concept consumes a lot of time and has a lot of complexity. Shortly speaking, in a 3x3 window, each neighbour makes their own searches in their 3x3 region and stores the hint pixels in the common array, namely the *hint array*. The searching process covers the second order element also. This tedious process can be simplified using the new method called 'Four corner-based hint pixel searching'. It can be illustrated using Figure 4.

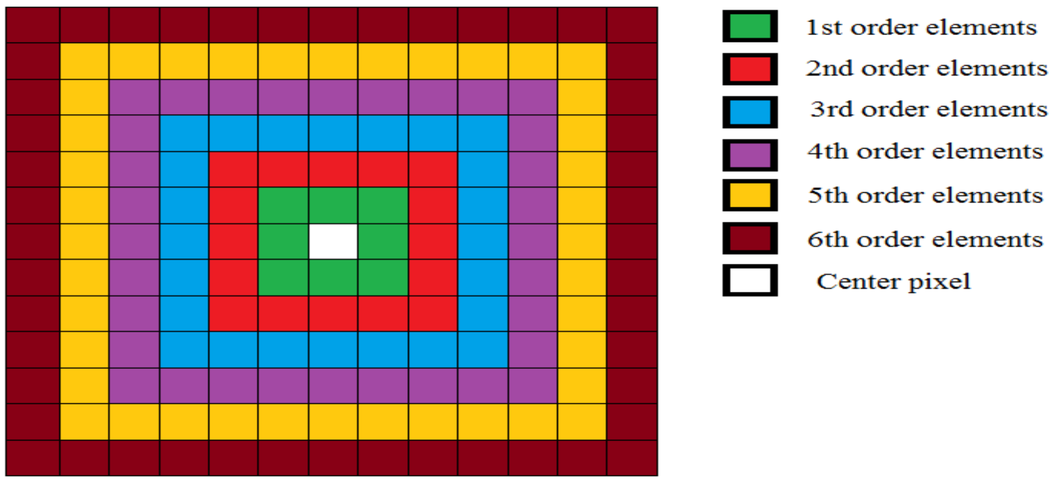

Fig.3: Presentation of position of multi order elements related to center pixel.

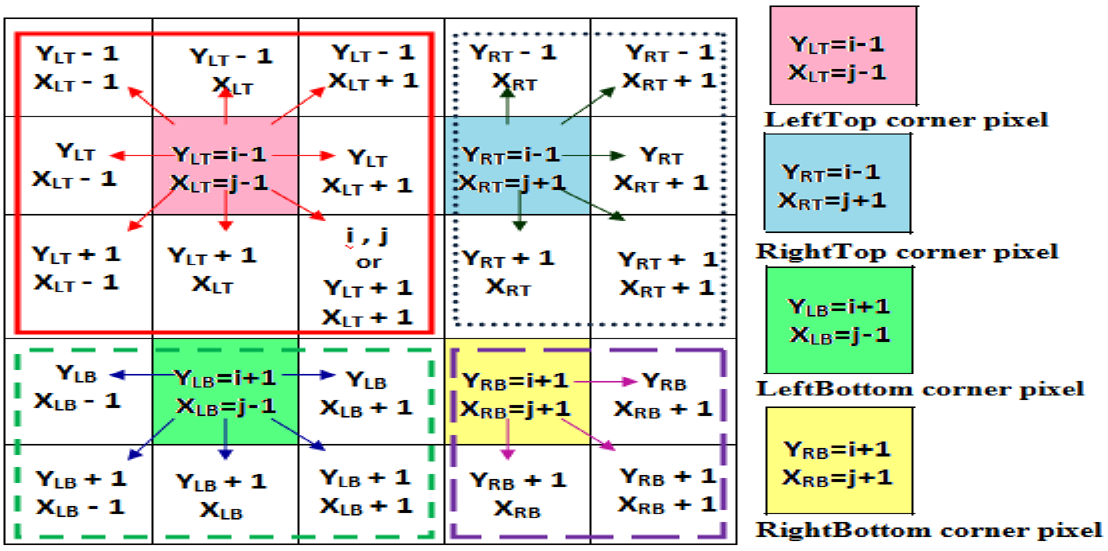

Fig.4: Illustration of four corner-based hint pixel searching.

According to Figure 4, it can be noticed that instead of searching for hint pixels using each element of the 3x3 window of $[i,j]^{th}$ pixel, it is enough to search through only the four corner elements of the 3x3 window of $[i,j]^{th}$ pixel. Those four corners are marked as:
- Left-top corner specified by $Y_{LT} = i - 1$ and $X_{LT} = j - 1$
- Right-top corner specified by $Y_{RT} = i - 1$ and $X_{RT} = j + 1$
- Left-Bottom corner specified by $Y_{LB} = i + 1$ and $X_{LB} = j - 1$
- Right-Bottom corner specified by $Y_{RB} = i + 1$ and $X_{RB} = j + 1$.

Shortly speaking, these four corners achieve the same result which can be done using the entire neighbours. This minimized searching model ensures the same result that is done by the complex searching model, to reduce the time complexity and memory requirement. The traditional entire neighbour-based searching model works based on the recursive procedure that eats more memory and it is unsuitable for big size images. But the proposed Four corner-based searching method just make searches through their neighbours of the four Corners. Figure 4 shows the pink-colored box as Left Top corner element. In Figure 4, the green-colored box indicates the left bottom corner pixel and the yellow-colored box indicates the Right Bottom corner pixel. The center pixel is noted by $[i,j]^{th}$ location. If the four corner pixels make their own searching through their 3x3 neighbours, then there may be a rising of overlapping issue, because they share some locations for searching at their borders. To avoid that overlapping, a new design is described which fixes the boundary of searching of each corner pixels. The left top corner makes searching inside the boundary marked by red color in Figure 4. The right top corner limits its searching boundary within the blue dotted lines in Figure 4. The left bottom corner pixel designs its searching boundary within the green dotted lines which is shown in Figure 4. The right bottom corner pixel shrinks its boundary range inside the purple colored dotted box.
Equation (11) shows the searching of Left Top corner based on 3x3 window where $Order = itr$ (i.e., herein $Order = 1$).

$$[HP_{LT}, HC_{LT}] = Func\_LT\_Search(i, j, I_{IP}, I_{NS}, Order) \qquad (11)$$

where
HP_{LT} - Hint pixels gathered by Left Top corner pixel
$Func\_LT\_Search()$ - Function to extract hint pixels using Left top corner
HC_{LT} - Hint pixels count based on Left Top corner based search
The function $Func\_LT\_Search()$ can be designed and defined based on the following statements:

```
$[HP_{LT}, HC_{LT}] = Func\_LT\_Search(i, j, I_{IP}, I_{NS}, Order)$

$CenterY = i - Order$
$CenterX = j - order$
$Count = 0$
$For\ m = CenterY - Order\ to\ CenterY + Order$
$For\ n = CenterX - Order\ to\ CenterX + Order$
If $(I_{NS}^{m,n} = 1)$
$HP_{LT}^{count} = I_{IP}^{m,n}$
$Count = Count + 1$
$End$
```

```
$End$
$End$
$HC_{LT} = Count$
return $HP_{LT}, HC_{LT}$
End
```

Equation (12), shows the searching of Right Top Corner based in a 3x3 window, where $Order = itr$ (i.e., herein Order = 1)

$[HP_{RT}, HC_{RT}] = Func\_RT\_Search(i, j, I_{IP}, I_{NS}, Order)$ (12)
where
HP_{RT} - Hint pixels gathered by Right Top corner pixel
$Func\_RT\_Search()$ - Function to extract the hint pixels based on Right top corner
HC_{RT} - Hint pixels count based on Right Top corner based search
The function $Func\_RT\_Search()$ can be designed and generalized based on the following statements:

```
$[HP_{RT}, HC_{RT}] = Func\_RT\_Search(i, j, I_{IP}, I_{NS}, Order)$
$Center\ Y = i - Order$
$Center\ X = j + Order$
$Count = 0$
$For\ m = CenterY - Order\ to\ CenterY + Order$
$For\ n = CenterX\ to\ CenterX + Order$
If ($I_{NS}^{m,n} = 1$)
$HP_{RT}^{count} = I_{IP}^{m,n}$
$Count = Count + 1$
$End$
$End$
$End$
$HC_{RT} = Count$

return $HP_{RT}, HC_{RT}$
$End$
```

Equation (13) shows the searching of Left bottom corner based on 3x3 window where $Order = itr$ (i.e., herein Order = 1).

$[HP_{LB}, HC_{LB}] = Func\_LB\_Search(i, j, I_{IP}, I_{NS}, Order)$ (13)
where
HP_{LB} - Hint pixels searched by Left Bottom corner
$Func\_LB\_Search()$ - Function to search the hint pixels using Left Bottom corner
HC_{LB} - Count of hint pixels according to Left Bottom Corner
The function $Func\_LB\_Search()$ is designed in the generalized form using the following statements:

```
$[HP_{LB}, HC_{LB}] = Func\_LB\_Search(i, j, I_{IP}, I_{NS}, Order)$
$Center\ Y = i + Order$
$Center\ X = j - Order$
$Count = 0$
$For\ m = CenterY\ to\ CenterY + Order$
$For\ n = CenterX - Order\ to\ CenterX + Order$
$If\ (I_{NS}^{m,n} = 1)$
$HP_{LB}^{count} = I_{ip}^{m,n}$
$Count = Count + 1$
$End$
$End$
$End$
$HC_{LB} = Count$
return $HP_{LB}, HC_{LB}$
$End$
```

Equation (14) describes the searching of Right Bottom Corner based on 3x3 window where order =1
$[HP_{RB}, HC_{RB}] = Func\_RB\_Search(i, j, I_{IP}, I_{NS}, Order)$ (14)
where
HP_{RB} - Right Bottom based hint pixels searching oriented array
$Func\_RB\_Search()$ - Function to search the hint pixels using Right bottom corner
HC_{RB} - Quantity of hint pixels related to Right Bottom search

The function $Func\_RB\_Search()$ is designed and generalized based on the following statements:

```
$[HP_{RB}, HC_{RB}] = Func\_RB\_Search(i, j, I_{IP}, I_{NS}, Order)$
$Center\ Y = i + Order$
$Center\ X = j + Order$
$Count = 0$
$For\ m = CenterY\ to\ CenterY + Order$
$For\ n = CenterX\ to\ CenterX + Order$
$If\ (I_{NS}^{m,n} = 1)$
$HP_{RB}^{count} = I_{IP}^{m,n}$
$Count = Count + 1$
$End$
$End$
$End$
$HC_{RB} = Count$
return $HP_{RB}, HC_{RB}$
End
```

The fused pixels set HP is generated using Equation (15), which integrates the four corner-based hint pixels such as $HP_{LT}, HP_{RT}, HP_{LB}, HP_{RB}$.

$$HP = \{ HP_{LT}, HP_{RT}, HP_{LB}, HP_{RB} \} \quad (15)$$

The length HC of the fused hint pixels is found using Equation (16).

$$HC = HC_{LT} + HP_{RT} + HP_{LB} + HP_{RB} \quad (16)$$

If the value of HC is greater than or equal to $\frac{itr \times 8}{2} + 1$, then the hint pixel searching process is terminated, otherwise, the fresh hint pixel searching process is progressed based on the next-order using the criterion $Order = Order + 1$. This process is explained in Figure 2.
The data optimization is described in Figure 5. It ensures that if $itr = 1$ then there is a need for 5 hint pixels according to the criterion $\frac{itr \times 8}{2} + 1$, because 5 pixels can satisfy the median computation process. Afterwards, the median is computed, and it is used to remove the $[i, j]^{th}$ noisy pixel using $I_{INF}^{i,j} = median$.
Suppose the condition $Order = 1$ cannot yield the saturated level hint pixels, then the hint pixels searching process is progressed based on next order, i.e., Order=2. The equations such as Equation (14), Equation (15), Equation (16), and Equation (17) extract the hint pixels based on the 2^{nd} Order search.

$$[HP_{LT}, HC_{LT}] = Func\_LT\_Search(i, j, I_{IP}, I_{NS}, Order = 2) \quad (17)$$
$$[HP_{RT}, HC_{RT}] = Func\_RT\_Search(i, j, I_{IP}, I_{NS}, Order = 2) \quad (18)$$
$$[HP_{LB}, HC_{LB}] = Func\_LB\_Search(i, j, I_{IP}, I_{NS}, Order = 2) \quad (19)$$
$$[HP_{RB}, HC_{RB}] = Func\_RB\_Search(i, j, I_{IP}, I_{NS}, Order = 2) \quad (20)$$

The fused hint pixels HP are generated using Equation (15), and the fused hint pixel count HC is computed using Equation (16). If $HC \geq 5$, then the hint pixel searching process is terminated; otherwise, the next searching process is forwarded with Order=3. If saturated HC is obtained, then the median-based noise removal is proceeded, and the iterative process is terminated.
Suppose the condition $Order = 2$ cannot yield the saturated hint pixels, then the next search is started using $Order = 3$. The equations such as Equation (17), Equation (18), Equation (19), and Equation (20) are executed by modifying the input parameter O$rder$ by $Order = 3$. The fused hint pixels HP are generated using Equation (15), and the fused hint pixels count HC is evaluated using Equation (16). If $HC > 5$, then median- based noise removal is performed and also the entire iterations regarding to $[i, j]^{th}$ noisy pixel are terminated. Suppose the condition $Order = 3$ cannot gather the saturated hint pixels, then the order=4 oriented searching progresses along with noise reduction. A similar process is forwarded to the $Order = 5$ criterion, and if it yields the saturated hint pixels, then denoising is done based on the median. If the first-five orders based searching is not apt to give the hint pixels, finally $Order = 6$ oriented searching is processed, and the maximum obtained hint pixels are considered. If the value of HC is more than 5, the first 5 elements are used to compute the median; otherwise, the possibly obtained elements are used for the median computation. Finally, the noisy pixel $[i, j]$ is resolved using median value.
Suppose hint pixel length l is equal to 0 then the next iteration $itr = 2$ is progressed to compute the predicted noisy pixel.

3.2.2 Iteration 2

The variable itr is assigned as $itr = 2$. This iteration tries to obtain the predicted noise-free pixel using the window size of $[itr \times 2 + 1, itr \times 2 + 1]$, which is equivalent to 5×5. This 5×5 size window centered by the $[i, j]^{th}$ noisy pixel is extracted using Equation (2).

The trimmed non-noisy elements are collected together into the linear array namely Hint array using Equation (4).

The 'case 1' context is tested on the 5×5 window. The threshold $thr = itr \times 2 + 1$ is computed. If $l \geq thr$ then the median computation is performed and the noisy pixel is removed. Otherwise, the 'case 2' context is tested on the 5×5 window. If $l > 0$ and $l < ((itr \times 8)/2) + 1$ is true then case 2 process involves the hint pixel searching which started with $Order = itr$ and ended up with $order = 6$, depending on the availability of saturated hint pixels.

Herein, the considered multi orders are 2 to 6. When the saturated hint pixels count HC reaches the condition that $HC > [(itr \times 8)/2] + 1$, (here it is $HC > 9$), the multi order based hint pixel searching is terminated according to Figure 2, and the denoising is performed using the first 9 hint pixels according to the data dimension reduction depicted in Figure 5. If $l = 0$ then the next iteration is forwarded to denoise the noisy pixel.

3.2.3 Iteration 3

Iteration 3 is activated by assigning the variable itr as $itr = 3$. A window of size $[itr \times 2 + 1, itr \times 2 + 1]$ which is equivalent to 7×7, is extracted using Equation (2). The trimmed, non-noisy elements are stored in the hint array. If the count of the hint array meets the condition $l \geq thr$, then the median based operation is used to denoise the $[i, j]^{th}$ noisy pixel.

If the condition $l > 0$ and $l < \left(\frac{itr \times 8}{2}\right) + 1$, is true the case 2 process is invoked with hint pixel searching based on $Order = 3$ to $Order = 6$ until reaching the saturated hint pixels. When it reaches the saturated level of hints, i.e., $\left(\frac{itr \times 8}{2}\right) + 1 = 13$, it undergoes the data dimensionality process described in Figure 5, because according to median computation, 13 pixels count is a little more excess. Among the 13 hints, 10 hint pixels are selected based on the Shortest Distance (SD) technique, which incorporates the shorter distance by providing 10 hint pixels. The distance between two pixels (i.e., the center pixel $[i, j]$ and any hint pixel $[p, q]$) can be expressed using Equation (21).

$$d = \sqrt{(i - p)^2 + (j - q)^2} \qquad (21)$$

The distance produced by the hint pixels according to the center pixel $[i, j]$ is stored in the distance array DA. This distance array is stored in ascending order, so those shorter distances producing hint pixels are ordered in the mode of minimum to maximum. Among them, the first 10 elements are chosen as the optimum hint pixels based on Figure 5. The median process is adapted to denoise the $[i, j]^{th}$ pixel. If the condition $l = 0$ is true, then *iteration 4* has progressed.

3.2.4 Iteration 4

Iteration 4 is forwarded using $itr = 4$. A window of 9×9 is extracted, and both the hint pixels array and its count l are computed. If $l > thr$ then the median computation based denoising is performed. If l is set between 1 and $\left(\frac{itr \times 8}{2}\right) + 1$, then the four-corner based hint pixel searching is enabled, and both the hint pixels array HP and count HC are computed.

The count HC undergoes the data dimensionality reduction process using the shortest distance method to pick the first 11 elements, according to Figure 5. The median based denoising replaces the noisy pixel using the 11 elements.

3.2.5 Iteration 5

Iteration 5 is initiated by assigning $itr = 5$. A window of size 11×11 is used to collect the hint pixels. If $l \geq thr$, then the direct median computation process is used to replace the noisy pixel. If l is less than the saturation level of hint pixels, then four corner based hint pixel searching is performed to gather the 21 hint pixels in the multi-order positions.

The numeric 21 is determined using $\left(\frac{itr \times 8}{2}\right) + 1 = 21$. The WSDHM based data dimensionality process reduces these 21 hint pixels to 12 hint pixels, which are selected based on privileges such as shortest distance and majority occurrence.

This research proposes the novel data dimensionality reduction method WSDHM to balance the two parameters, such as shortest distance and majority occurrence.

Some pixels may have closeness to the center-pixel but not come under the majority case, and in some other cases they serve with the majority occurrence but are not set to a short distance.

To balance these two criteria, the WSDHM method uses individual weight scores of SD and majority property. Finally, they are integrated to reveal the real weight score of the hint pixels. The high-scoring hint pixels are chosen as the optimum 12 hint pixels. This phenomenon is expressed in Figure 5.

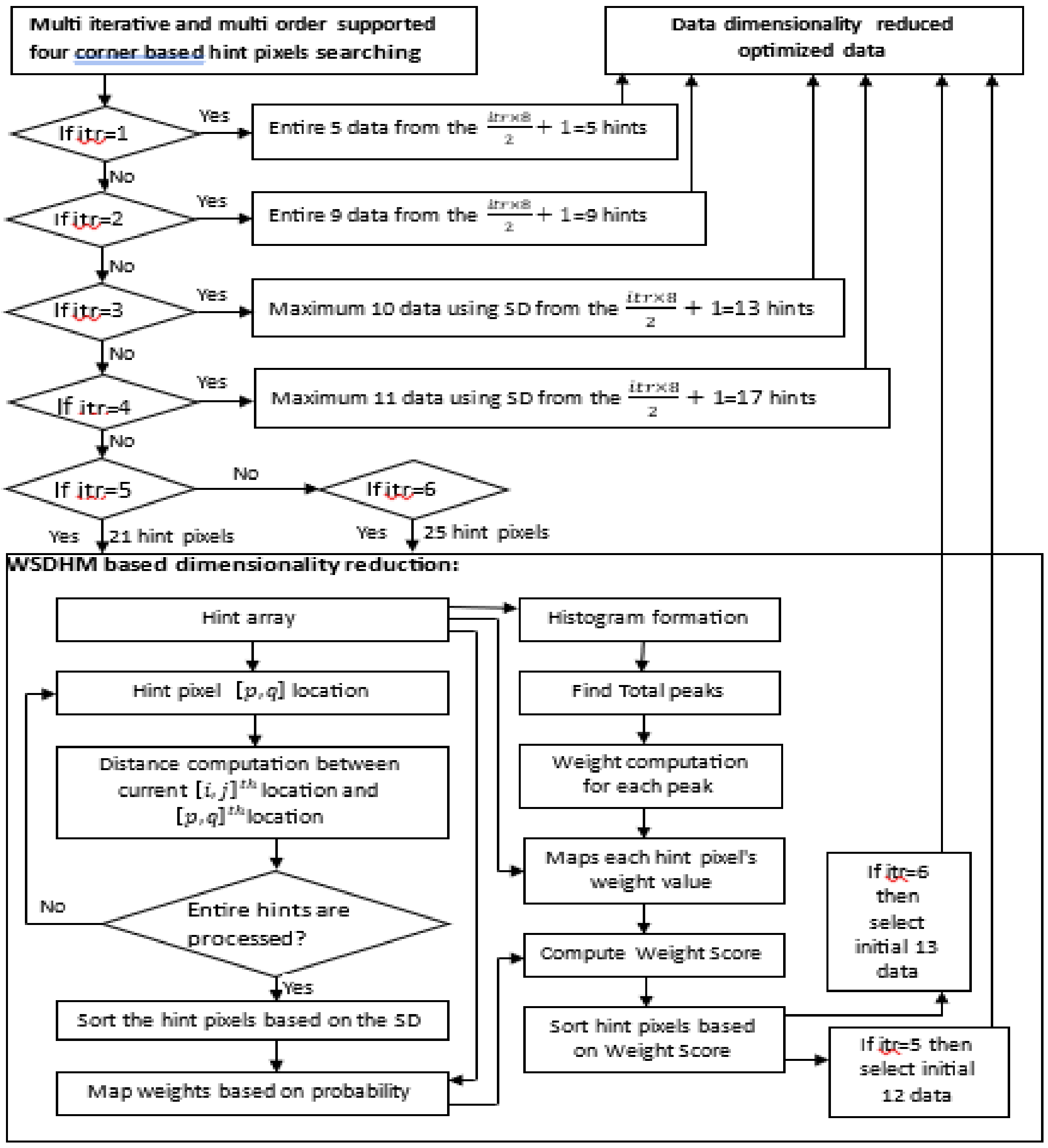
Multi iterative and multi order supported four corner based hint pixels searching
Data dimensionality reduced optimized data
If itr=1
Yes
Entire 5 data from the $\frac{itr\times 8}{2}+1$=5 hints
No
If itr=2
Entire 9 data from the $\frac{itr\times 8}{2}+1$=9 hints
If itr=3
Maximum 10 data using SD from the $\frac{itr\times 8}{2}+1$=13 hints
If itr=4
Maximum 11 data using SD from the $\frac{itr\times 8}{2}+1$=17 hints
If itr=5
If itr=6
21 hint pixels
25 hint pixels
WSDHM based dimensionality reduction:
Hint array
Histogram formation
Hint pixel [p,q] location
Find Total peaks
Distance computation between current $[i,j]^{th}$ location and $[p,q]^{th}$ location
Weight computation for each peak
Maps each hint pixel's weight value
Entire hints are processed?
Compute Weight Score
If itr=6 then select initial 13 data
Sort the hint pixels based on the SD
Sort hint pixels based on Weight Score
If itr=5 then select initial 12 data
Map weights based on probability

Fig.5: Block diagram of the proposed data dimensionality reduction method.

The WSDHM algorithm first computes the shortest distance between the hint pixels and the center pixel $[i,j]$, and then the shortest distance based ascending sort is performed in hint pixels. The SD-oriented weight value is computed using Equation (22).

$$\alpha = 1 - \left(k \times \left(\frac{1}{HC}\right)\right) \qquad (22)$$

where
α - Shortest distance based weight value

k - Index of sorted position
The weight value α is set to high values for the starting indexes in the sorted array. It is set to low values for the end-range indexes in the sorted array. Shortly speaking, the lowest distance providing hint pixel reaches the highest weight value, while the highest distance providing hint pixel receives the least weight value, according to Equation (22).
The next process is the computation of the majority-based weight value. First, a histogram is formulated. It shows the frequency of intensity values related to hint pixels. Then, the quantity of all peaks is calculated. Hint pixels are ordered based on the high-peak information, meaning that, hints belonging to higher peaks are arranged in an array SA. The total peaks count is computed and noted as TP. The majority-oriented weight value computation is done using Equation (23).

$$\beta = 1 - \left(k \times \left(\frac{1}{TP}\right)\right) \tag{23}$$

where

β - Majority occurrence based weight value

k - Index of high peak based sorted array

In Equation (23), the highest peak providing hint pixels, receives the highest weight value, while the lowest peak provider of hint pixels achieves the least weight value. Herein, histogram analysis is used because histograms easily explore the majority occurrence of hint pixels.

The weight score is computed using Equation (24) by adding α and β weights.

$$\gamma = \alpha + \beta \tag{24}$$

where

γ - Weight score from both SD and majority oriented weights

The hint pixels array is mapped against their weight score value γ. Then, the hint pixels array HP is descendingly sorted based on the γ mapping. The high scoring hint pixels are seated at initial levels in the sorted array. Finally, 12 elements from the starting positions of the sorted array are chosen as the optimum hint pixels, and they undergo the median computation to replace the noisy pixel. This is the working procedure of the proposed WSDHM data dimensionality reduction method. If $l = 0$, then the next iteration, namely *iteration 6,* will be continued.

3.2.6 Iteration 6

Iteration 6 is invoked by assigning $itr = 6$. A window of 13x13 is used to gather the hint pixels. If $l \geq thr$, then a direct median process is applied to remove the noise. If there are hint pixels less than the saturated level, then the four corner pixels based hint pixels searching module is activated on the 6th order only. The collected hint pixels (i.e., up to 25 elements) undergo the WSDHM based dimensionality reduction process to extract the optimum 13 hint pixels according to Figure 5. These optimum hint pixels are used to compute the median value to replace the noisy pixel. Suppose, *iteration 6* produces zero hint pixels, then the immediate 3x3 neighbours of $[i,j]^{th}$ noisy pixel are extracted without trimming process, and they are used to compute the median value, which can be the solution to denoise the noisy pixel. This process is repeated for the entire noisy pixels, and the output is stored in the noise-free image I_{NF}.

Thus, the proposed IHSMW method effectively denoise the salt and pepper noises from brain MRI images using the four corner pixels searching algorithm, WSDHM based data dimensionality reduction method, and multi-order position handling with multi-iterations.

4 RESULT AND DISCUSSION

In this work, there are four different datasets are used in evaluating denoising methods. This work focuses on the measurements like MSE, PSNR, TT, IEF, and MSSIM.

4.1 Dataset description

The four different datasets used in this analysis are MMHRC-DB, PRNV-DB, NSC-DB, and SSL-DB. Figure 6 expresses the sample images from MMHRC-DB database while Figure 7 shows the sample images from PRNV-DB database. Figure 8 depicts the sample images from NSC-DB database and Figure 9 describes the sample images from SSL-DB database.

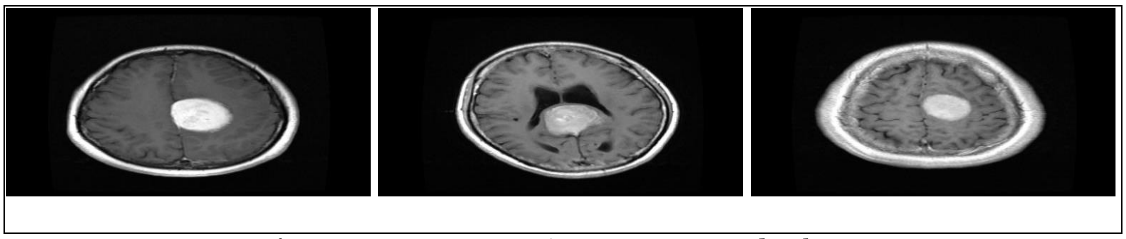

Fig.6: Images representing MMHRC-DB database.

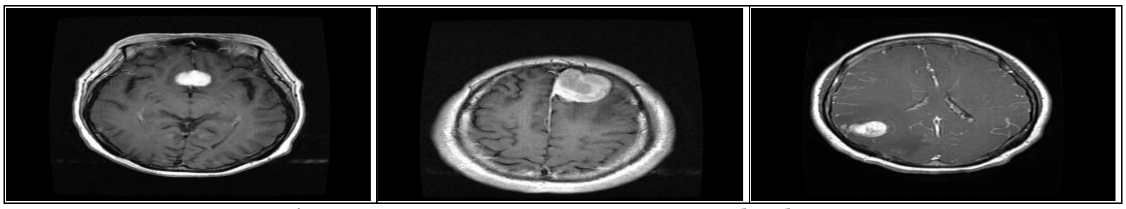

Fig.7: Images representing PRNV-DB database.

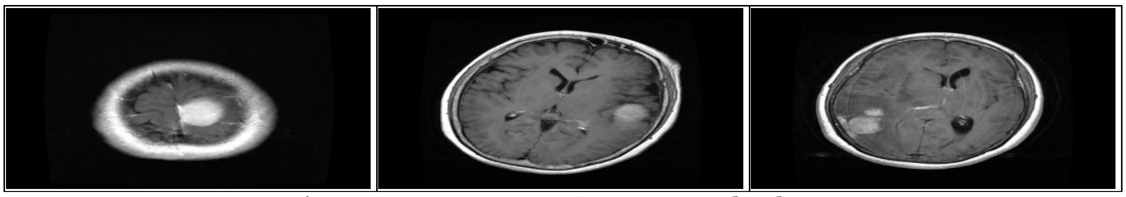

Fig.8: Images representing NSC-DB database.

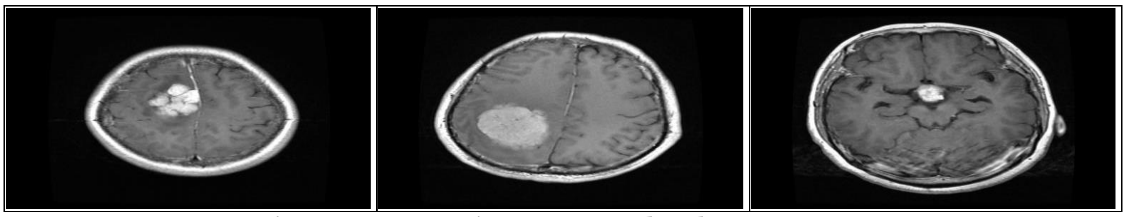

Fig.9: representing SSL-DB database.Images

4.2 Comparative Analysis

The comparative analysis section analyzes the performance of existing and proposed techniques. This section evaluates the performance of the various MRI image denoising techniques. Performance-based rank index analysis is done by comparing the outputs obtained by the five metrics, such as PSNR, MSE, IEF, TT, and MSSIM, regarding the five methods. This analysis section analyzes the proposed IHSMW filter against the four existing denoising methods to prove the performance quality of the IHSMW filter, and they are:

- Impulse Noise Reduction using Redescending m-estimator and Modified Nearest Neighbor filter (INR-RMNN) (Vargas et al. 2018)
- Impulse Noise Reduction using SAID-END method (INR-SE) (Singh et al. 2020)
- Impulse Noise Reduction using Distribution Transformed Network method (INR-DTN) (Guanyu et al. 2021)
- Impulse Noise Reduction using Thresholding and Regularization techniques (INR-TR) (Hien et al. 2022).

Table 1: MSE analysis on denoising for 60% noise corruption

Database	MSE				
	INR-RMNN method	**INR-SE method**	**INR-DTN method**	**INR-TR method**	**Proposed IHSMW method**
MMHRC-DB	248.93	238.83	208.49	148.28	**67.31**
PRNV-DB	229.13	220.33	197.28	138.06	**45.507**
NSC-DB	254.14	247.22	214.33	152,08	**71.463**
SSL-DB	233.39	227.03	202.81	143.91	**49.100**

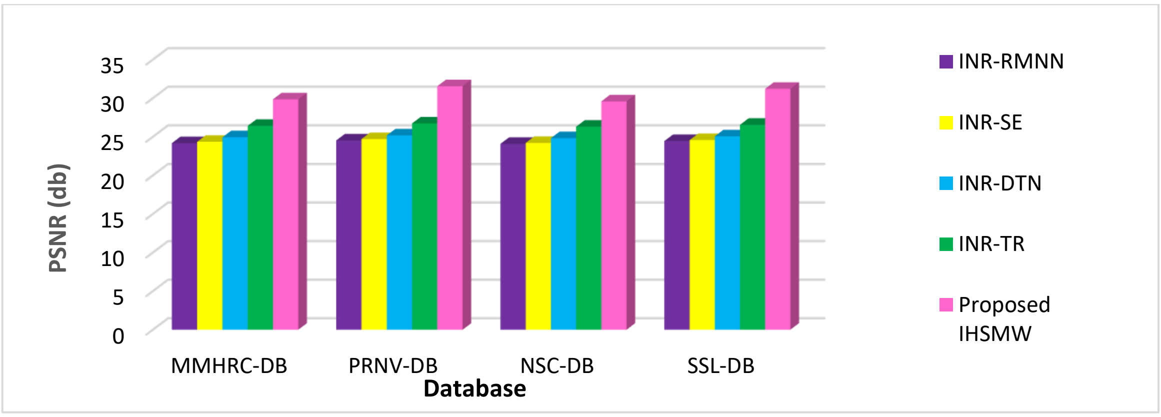

Fig.10: PSNR analysis chart on denoising for 60% noise corruption.

Table 1 reveals that the proposed IHSMW method using the PRNV-DB database gives a lower MSE value, which means it gives a better result in existing method. The average MSE value for INR-RMNN is 241.39, INR-SE is 233.35, INR-DTN is 205.72, INR-TR is 145.58, and the proposed IHSMW is 58.345. This analysis proves the proposed method's potential ability to remove impulsive noise from brain MRI images.

Figure 10 shows the PSNR analysis result. The PSNR value of the proposed IHSMW method for MMHRC-DB is 29.85, PRNV-DB is 31.55, NSC-DB is 29.59, and SSL-DB is 31.22. It is known that a higher PSNR value shows a higher quality of denoising. This analysis proven that the proposed IHSMW method shows a higher quality of denoising than the existing methods.

Table 2: Time Taken analysis on denoising for 60% noise corruption

Method	**Time Taken (in sec) for denoising**			
	Database			
	MMHRC-DB	**PRNV-DB**	**NSC-DB**	**SSL-DB**
INR-RMNN	7.469	7.134	7.630	7.228
INR-SE	9.675	9.467	9.880	9.608
INR-DTN	6.785	6.453	6.922	6.572
INR-TR	7.967	7.862	8.041	7.905
Proposed IHSMW	**4.753**	**4.430**	**4.805**	**4.613**

From the Table 2, it is proven that the proposed IHSMW method holds less time than the existing INR-RMNN, INR-SE, INR-DTN, and INR-TR methods against databases such as MMHRC-DB, PRNV-DB, NSC-DB, and SSL-DB. The mean duration of the proposed method considering the four databases is 4.650 sec, while the second-best INR-DTN method is 6.683, meaning that this analysis improves the time-efficiency by 30.42%.

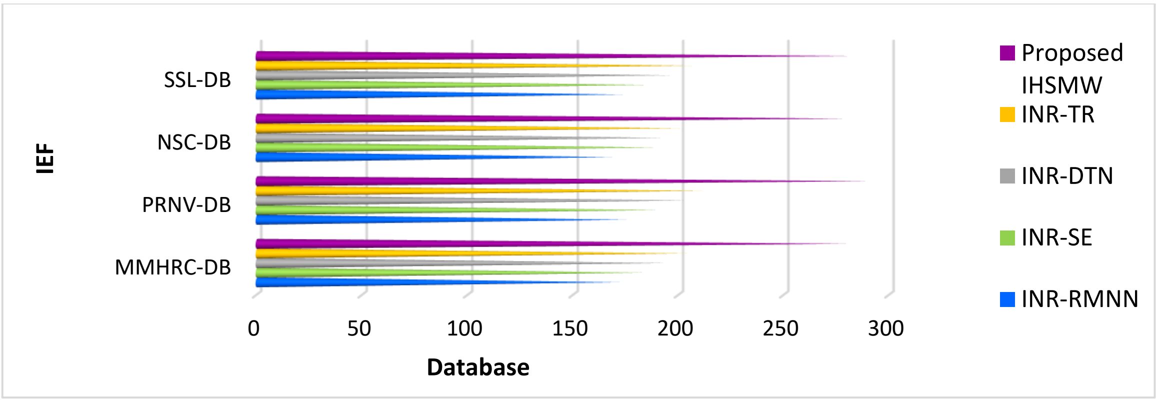

Fig.11: IEF analysis chart on image denoising for 90% noise corruption.

The IEF results for the proposed IHSMW method regarding the four databases such as MMHRC-DB, PRNV-DB, NSC-DB, and SSL-DB are 280.62, 290.81, 279.63, and 282.56, respectively. Also, the IEF values for the INR-TR method corresponding to the databases such as MMHRC-DB, PRNV-DB, NSC-DB, and SSL-DB are 203.73, 211.40, 200.74, and 206.73. From Figure 11, it is proven that the proposed IHSMW method gives the highest value of 290.81 related to the PRNV-DB database.

Table 3: MSSIM analysis for denoising on 90% noise

Database	**MSSIM analysis**				
	INR-RMNN method	**INR-SE method**	**INR-DTN method**	**INR-TR method**	**ProposedIHSMW method**
MMHRC-DB	0.652	0.690	0.721	0.781	**0.819**
PRNV-DB	0.669	0.715	0.742	0.796	**0.837**
NSC-DB	0.649	0.684	0.719	0.774	**0.811**
SSL-DB	0.659	0.703	0.734	0.792	**0.825**

Table 3 focuses on the MSSIM analysis. The method that has the highest MSSIM is considered to be the best. Hence, the proposed IHSMW method that has the highest MSSIM value of 0.837 is considered the best denoising method. The INR-RMNN method that has the least MSSIM value, hence, it is considered the method of least potential in denoising.

Table 4: Average of IEF, PSNR, MSSIM analysis on denoising for 90% noise corruption

Methods	**Average IEF**	**Average PSNR**	**Average MSSM**
INR-RMNN	172.26	11.944 db	0.658
INR-SE	187.29	13.452 db	0.698
INR-DTN	196.73	15.720 db	0.729

INR-TR	205.65	16.273 db	0.785
Proposed IHSMW	**283.405**	**26.310 db**	**0.823**

The average IEF, PSNR, MSSM value has been calculated for each method using the four databases, such as MMHRC-DB, PRNV-DB, NSC-DB, and SSL-DB, and has shown in Table 4. The proposed IHSMW method significantly outperforms all other methods in each metric, indicating the highest quality in image enhancement, noise reduction, and structural similarity. This analysis adds value to the proof that the proposed IHSMW method serves the best denoising for impulse noise in brain MRI images.

5 CONCLUSION

In this work, we introduces a novel MRI image denoising method namely IHSMW. This technique addresses the challenge of removing impulse noise while maintaining important anatomical characteristics, and removes noise such as salt and pepper from grayscale brain MRI images. The proposed method is empowered by the hint pixel searching through FCMIHP and data dimensionality reduction through WSDHM, hence, the IHSMW method demonstrates superior performance in enhancing image quality compared to traditional denoising methods. Extensive evaluations using various datasets and performance metrics, such as PSNR, SSIM, and IEF, indicate that our proposed method significantly improves the diagnostic potential of MRI scans. The experimental findings reveal that the proposed IHSMW method achieves a better average PSNR value of 26.310 db and an average IEF of 283.40 for 90% noise corruption, which means it gives a better result than the existing method. In the future, optimization of the IHSMW method and its application to other types of medical imaging should be considered.

REFERENCE

1. Vargas D.M., Rubio J., Kinani J.M.V., and Funes F.J.G., 2018, ‘An efficient nonlinear approach for removing fixed-value impulse noise from grayscale images’, Springer, Journal of real time image processing, vol. 14, pp. 617-633.
2. Singh A., Sethi G., and Kalra G.S., 2020, ‘Spatially adaptive image denoising via enhanced noise detection method for gray-scale and color images’, IEEE access, vol. 8, pp. 112985-113002.
3. Guanyu L., Fengqin Z., and Qiegen L., 2021, ‘Distribution-transformed network for impulse noise removal’, Springer, vol. 26, pp. 543-553.
4. Hien N.N., Thanh D.N.H., Erkan U., and Tavares J.M.R., 2022, ‘Image noise removal method based on thresholding and regularization techniques’, IEEE Access, vol. 10, pp. 71584-71597.
5. MSSIM, 2022, Accessed from <https://en.wikipedia.org/wiki/Structural_similarity>, Accessed on [19 July 2022].
6. MMHRC Database, 2021, Accessed from <https://www.mmhrc.in/>, Accessed on [5 Aug 2022].
7. PRNV Database, 2021, Accessed from <https://www.teleradiologyhub.com/diagnostics/details/36/927-pranav-diagnostics-centre>, Accessed on [5 Aug 2022].
8. NSC Database, 2021, Accessed from < https://nagercoilscancentre.business.site/>, Accessed on [5 Aug 2022].
9. SSL Database, 2021, Accessed from < https://www.saravanascans.com/>, Accessed on [5 Aug 2022].
10. Brain MRI, 2023, Accessed from <https://www.cancer.net/cancer-types/brain-tumor/diagnosis>, Accessed on [23-06-2023].
11. Shinde A.S., Mahendra B.M., Nejakar S., Herur S.M. and Bhat N., 2022. 'Performance analysis of machine learning algorithm of detection and classification of brain tumor using computer vision'. Advances in engineering software, vol. 173, Article ID: 103221.
12. Li G., Xu X., Zhang M., and Liu Q., 2020, 'Densely connected network for impulse noise removal', Pattern Analysis and Applications, vol. 23, pp. 1263-1275.
13. Enginoglu S., Erkan U., and Memis S., 2019, 'Pixel similarity-based adaptive Riesz mean filter for salt-and-pepper noise removal', Multimedia Tools and Applications, vol. 78, pp. 35401-35418.
14. Zhang Z., Han D., Dezert J., and Yang Y., 2018, 'A new adaptive switching median filter for impulse noise reduction with pre-detection based on evidential reasoning', Signal Process, vol. 147, pp. 173-189.
15. Bai T., Tan J., Hu M., and Wang Y., 2014, 'A novel algorithm for removal of salt and pepper noise using continued fractions interpolation', Signal Processing, vol. 102, pp. 247-255.
16. Chen J., Zhan Y., Cao H., and Wu X., 2018, 'Adaptive probability filter for removing salt and pepper noises', IET Image Processing, vol. 12, issue 6, pp. 863-871.
17. Fu B., Zhao X., Li Y., and Wang X., 2019, 'Patch-based contour prior image denoising for salt and pepper noise', Multimedia Tools and Applications, vol. 78, issue 21, pp. 30865-30875.
18. Kimiaei M. and Rahpeymaii F., 2019, 'Impulse noise removal by an adaptive trust-region method', *Soft Computing*, vol. *23*, pp.11901-11923.

19. Gupta S. and Sunkaria R.K., 2017, 'Real-time salt and pepper noise removal from medical images using a modified weighted average filtering', In 2017 fourth international conference on image information processing (ICIIP), pp. 1-6).
20. Ali H.M., 2016, 'A new method to remove salt & pepper noise in magnetic resonance images', In 2016 11th International Conference on Computer Engineering & Systems (ICCES), pp. 155-160.
21. Alrabai A.I.S., 2021, 'Denoising abnormal MRI images', Scientific journal of applied sciences of sabratha university, pp. 9-15.
22. Ebrahimnejad J. and Naghsh A., 2021, 'Adaptive removal of high-density salt-and-pepper noise (ARSPN) for robust ROI detection used in watermarking of MRI images of the brain', Computers in Biology and Medicine, vol. 137, issue C, Doi: https://doi.org/10.1016/j.compbiomed.2021.104831.
23. Gonzalez R.C., Richard E., 2002, 'Digital image processing, Prentice hall press, Upper Saddle River, NJ, USA, 2002.
24. Hwang H. and Haddad R., 1995, 'Adaptive median filters: new algorithms and results', IEEE transactions in image processing', vol. 4, pp. 499-502.
25. Sree P.S.J., Kumar P., Siddavatam R, and Verma R., 2013, 'Salt-and-pepper noise removal by adaptive median-based lifting filter using second-generation wavelets', Signal image video processing, vol. 7, pp. 111-118.
26. Adeli A., Tajeripoor F., Zomorodian M.J., and Neshat M., 2012, 'Comparison of the fuzzy-based wavelet shrinkage image denoising techniques', International journal of computer science, vol. 9, pp. 211-216.
27. Xing Y., Xu J., Tan J., Li D., and Zha W., 2019, 'Deep CNN for removal of salt and pepper noise', IET Image Processing, vol. 13, issue 9, pp. 1550-1560.
28. Yi H. and XiaoXuan M., 2020, 'August. image noise recognition algorithm based on BP neural network', In *2020 Chinese Control And Decision Conference (CCDC),* pp. 106-111.

S. PRATHIBA. She received the B.Sc. degree in Mathematics from Meenakshi College for Women (Autonomous), Chennai, in 2008 and the M.Sc. degree in Information Technology from Meenakshi College for Women (Autonomous), Chennai, in 2010. She received the M.Phil. degree in Computer Science from S.T. Hindu College, Nagercoil, in 2019. She is currently pursuing the Ph.D. degree in Computer Science at Manonmaniam Sundaranar University, Tirunelveli. Her research interest includes digital image processing and business intelligence.

Dr. B. SIVAGAMI. She received the B.Sc. degree in Computer Science from S.T. Hindu College, Nagercoil, in 1991 and the M.Sc. degree in Computer Science from Avinashilingam Deemed University, Coimbatore, in 2006. She received the M.Phil. degree in Computer Science from Azhagappa University, Karaikudi, in 2001 and the Ph.D. degree in Manonmaniam Sundaranar University, Tirunelveli, in 2014. She joined the Department of Computer Science & Application, S.T. Hindu College in 1993 as an Assistant Professor. Since 2019, she has been working as Associate Professor and Head of the Computer Science & Application department in S.T. Hindu College. Her research interest includes digital image processing.