ABSTRACT

For computer vision systems to effectively perform diagnoses, identification, tracking, monitoring and surveillance, image data must be devoid of noise. Various types of noises such as Salt-and-pepper or Impulse, Gaussian, Shot, Quantization, Anisotropic, and Periodic noises corrupt images making it difficult to extract relevant information from them. This has led to a lot of proposed algorithms to help fix the problem. Among the proposed algorithms, the median filter has been successful in handling salt-and-pepper noise and preserving edges in images. However, its moderate to high running time and poor performance when images are corrupted with high densities of noise, has led to various proposed modifications of the median filter. The challenge observed with all these modifications is the trade-off between efficient running time and quality of denoised images. This paper proposes an algorithm that delivers quality denoised images in low running time. Two state-of-the-art algorithms are combined into one and a technique called Mid-Value-Decision-Median introduced into the proposed algorithm to deliver high quality denoised images in real-time. The proposed algorithm, High-Performance Modified Decision Based Median Filter (HPMDBMF) runs about 200 times faster than the state-of-the-art Modified Decision Based Median Filter (MDBMF) and still generate equivalent output.
1. INTRODUCTION

Digital images may be corrupted due to glitches in image sensing components of image capturing devices or transmission channel which makes extraction of relevant information from them difficult [1,2]. The need to, therefore, deal with noise in images is crucial for any system that relies on images as input data [3]. Various types of noises such as Salt-and-pepper or Impulse, Gaussian, Shot, Quantization, Anisotropic, and Periodic noises corrupts images making them unusable at some point. This has led to a lot of proposed denoising algorithms to help denoise images for further processing or manipulations, making digital image denoising an active research domain [2]. Among the lot, the median based denoising algorithms are prominent in the domain of Image Processing (IP) and Computer Vision (CV) applications due to their simplicity and effectiveness to handle and preserve edges in images. Specifically, the median based denoising algorithms can handle salt-and-pepper noise, which is a type of noise that randomly alters a certain amount of pixels into two extremes, either 0 or 255, for an 8-bit image. This contributes to certain amounts of the pixels in an image becoming white or black (salt or black) hence the name of the noise.

The Standard Median Filter (SMF) is believed to be a trivial statistical filter [1] but produces acceptable results for denoising Salt-and-pepper noise with a relatively better time. The filter’s strength is its ability to preserve edges [4]. The median filter, however, has other challenges such as its inability to deal effectively with all kinds of noise introduced into an image, the poor computational time for denoising, and does not use rigorous statistical operations to effectively denoise an image resulting in an alteration of uncorrupted pixel [5]. The moderate to high computational cost associated with the standard median filter [6-8] has led to several modifications of the filter over the last four (4) decades after it was proposed by Tukey [9]. These modifications normally tackle one issue or the other related to the median filter. However, almost all the modifications of the median filter use the rank-order information of pixel’s intensity values within a kernel. The filter primarily uses robust non-linear filtering approach to help perform the denoising task [10]. Arce and Paredes [11] proposed a Weighted Median Filter (WMF) which did not use direct rank-order information of pixels’ intensities within the filtering kernel, but generated outputs with similar characteristics as the median filter.

The two main challenges of the SMF (moderate to high computational time and relatively poor output when an image is corrupted with a high density of impulse noise) have led to two main underlining quests to modify the SMF. The Recursive Median Filter (RMF), Adaptive Median Filter (AMF), Decision Based Median Filter (DBMF), Iterative Median Filter (IMF) proposed by [12], Modified Decision Based Median Filter, Weighted median filter (WMF) proposed by [13], Centre Weighted Median Filter (CWMF), Directional Median Filter (DMF), Centred Median Filter (CMF) are various modifications of the Standard Median Filter which target improving the quality of the denoised image. [14-24] proposed various modifications of the SMF which target reducing the running time of the SMF by either using the histogram based approach for the estimation of median values or approximation techniques to generate median values for the denoising task. A Fast Median Filter Approximation Proposed by [25] tackles the running time of the SMF to achieve denoising of impulse noise in real-time, while the Modified Decision Based Median Filter (MDBMF) proposed by [26] tackles the poor quality denoised images that are generated by the SMF. These denoising algorithms exhibit an interesting scenario of a trade-off between the quality of restored images and computational time or running time. Effective denoising algorithms may suffer from high computational time complexities, while those efficient with running time may also be ineffective in generating quality deroised images.

Kunsoth & Biswas proposed Modified Decision Based Median Filter (MDBMF) for impulse noise removal which generates excellent denoised images or outputs [26]. The concept of using decisions to evaluate a pixel’s value before operations are performed on them has been applied extensively in the domain of filtering impulse noise. For example, the Weighted Median Filter (WMF), Directional Weighted Median Filter (DWMF) [27], Centered Weighted Median Filter (CWMF) [28], Decision Based Median Filter (DBMF) have employed some decisions in their implementations and have

Keywords: Image denoising; median filter; decision based median filters; real-time image processing; real-time computer vision.
effectively improved the standard median filter. The Adaptive Median Filter (AMF) and its family generally use such a concept of evaluating the output of the median operation and deciding whether values fit for denoising or not. Most Adaptive algorithms will expand the size of the window used for filtering when the generated pixel values are not fit for the denoised image. MDBMF performs extremely well-suppressing impulse noise, but its running time is high due to the high number of comparison operations that are done before selecting an ideal value to replace a corrupted one. This makes the algorithm inefficient for real-time IP or CV.

To achieve salt-and-pepper denoising in real-time, Marcus & Ward proposed a new non-discrete algorithm (DP) that quickly approximates a median filter [25]. The algorithm is a hybrid of principles proposed by [16] and [21]. The strength of DP is based on a technique that prevents the re-sorting of overlapping columns of adjacent sliding windows making the estimation of the median value completing below O(n) running time complexity. It outperforms most state-of-the-art algorithms in terms of running time [16,18,21,27]. However, DP also performs poorly when the impulse noise density in an image is high.

In this paper, we present an improved version of the MDBMF algorithm. The proposed version uses a median of medians approximation techniques to reduce the computational time required by MDBMF algorithm. The approximation technique is also improved by the introduction of a technique called Mid-Value-Decision-Median to deliver an effective algorithm for real-time image processing or computer vision task.

The rest of the paper is structured such that Section 2 presents the methodology. Under section 2, section 2.1 presents Modified Decision Based Median Filter (MDBMF), section 2.2 presents the Fast Median Filter Approximation. Section 2.3 describes the proposed algorithm. Section 3.0 presents and discusses the experimental results of this study. Finally, the last section concludes this research.

2. METHODOLOGY

The proposed algorithm is a hybrid of MDBMF and DP: Fast Median Approximation Algorithms. The two algorithms are therefore presented in section 2.1 and 2.2 respectively to help explain the background of the proposed algorithm. Section 2.3 presents the proposed High-Performance Modified Decision Based Median Filter and the underlying concept of the algorithm.

2.1 Modified Decision Based Median Filter (MDBMF)

Algorithm 1 presents the pseudocode of MDBMF, while Fig. 1 presents the flowchart of the same. The algorithm is just like the standard median except for two key concepts that contribute to its impressive output. The algorithm avoids altering pixels perceived to be non-corrupted, and in deciding the ideal pixels for replacing corrupted ones, only non-corrupted pixels are used to select a median value for replacement. It expands the window for filtering when 3 x 3 window’s pixels are all corrupted.

2.2 DP: A Fast Median Filter Approximation

The 3 x 3 kernel implementation of DP partitions a window into 3 columns and the median for each column is calculated. The median of the three (3) medians is also finally estimated. Fig. 2 illustrates how the filtering kernel or window is manipulated. In DP, the overlapping columns of adjacent sliding windows are not re-sorted as seen in a lot of median based filtering algorithms. This makes the algorithm complete faster than the traditional approximated median filtering algorithms. DP is presented in Algorithm 2.

2.3 High-Performance Modified Decision Based Median Filter (HPMDBF)

The proposed HPMDBF is similar to Algorithm 1. The median estimation phase of MDBMF is replaced with the approximation technique (median of medians) introduced in Algorithm 3. The Mid-Value-Decision-Median (MVDM) is used to improve the output of the median of medians technique. The HPMDBF algorithm is presented next.

The median of medians employed in the proposed algorithm partitions a window into 3 columns and the median of each column is estimated. The final median selected for the denoising stage is the median of the 3 medians. A window of size 3 x 3 is initially used when processing a pixel, but may be expanded to 5 x 5 due to the nature of impulse noise distribution in
the 3 x 3 window. To improve the output generated by the median of medians method, the Mid-Value-Decision-Median technique is used to select median values for the denoised image.

**Algorithm 1: MDBMF Algorithm – Pseudocode**

Read Image
For all pixels in the image
   If a pixel value $P_{ij}$ is between 0 and 255, use $P_{ij}$ for the denoised image
   Else
      Select a 3x3 window with the pixel ($P_{ij}$) at the centre.
      Store all pixel values in the 3x3 window that are neither 0 nor 255 into Array $M$
      If $M$ is empty, select a 5x5 window with the $P_{ij}$ at the centre.
      Store all pixel values in the window that are neither 0 nor 255 into Array $M$
      If $M$ is not empty, estimate the median of values in $M$ and use it for denoising image.
      Else: use $P_{ij-1}$ of denoise image for $P_{ij}$

Fig. 1. MDBMF algorithm
Source: Kunsoth and Biswas [26]
Algorithm 2: DP – Fast Median Filter Approximation

Let $I$ represent the image to be filtered
Let $H$ be the HEIGHT of the Image
Let $W$ be the WIDTH of the Image

for $i = 2$ to $H - 1$
  $\text{col1} \leftarrow \text{median}(I(i-1, 1), I(i, 1), I(i+1, 1))$
  $\text{col2} \leftarrow \text{median}(I(i-1, 2), I(i, 2), I(i+1, 2))$
  for $j = 3$ to $W - 1$
    $\text{col3} \leftarrow \text{median}(I(i-1, j), I(i, j), I(i+1, j))$
    $I'(i, j) \leftarrow \text{median}(\text{col1}; \text{col2}; \text{col3})$
    'Save the median of first overlapping column
  end for
end for

Algorithm 3: High-Performance Modified Decision Based Median Filter Algorithm (HPDBMF)

//Impulse noise pixels are pixels with values as 0 or 255.
Read Image

For all pixels in the image
  If a pixel value $P_{ij}$ is between 0 and 255 use $P_{ij}$ for the denoised image
  Else
    Select a 3x3 window with the pixel at the centre.
    Partition the window into three (3) columns ($\text{Col1}$, $\text{Col2}$, $\text{Col3}$). Each Col will have 3 elements to be identified as $\text{ColX1}$, $\text{ColX2}$, and $\text{ColX3}$.
    $M1 \leftarrow \text{MVDM} (\text{Col11}, \text{Col12}, \text{Col13})$
    $M2 \leftarrow \text{MVDM} (\text{Col21}, \text{Col22}, \text{Col23})$
    $M3 \leftarrow \text{MVDM} (\text{Col31}, \text{Col32}, \text{Col33})$
    $\text{Median\_Value} \leftarrow \text{MVDM} (M1, M2, M3)$
    If $\text{Median\_Value}$ is corrupted, traverse 5 x 5 window size and select the first encounter of a non-impulse pixel as median value.
    If $\text{Median\_Value}$ is still corrupted, select the last processed window’s central pixel as the median for the current window.
    Use $\text{Median\_Value}$ for the denoised image
Function
Function MVDM (X, Y, Z) // Mid-Value-Decision Median
Sort X, Y, Z in ascending order and assign them to P₁, P₂ and P₃ respectively
If P₂ is 0 then select P₃ as MVDM
else if P₂ is 255 then select P₁ as MVDM
else select P₂ as MVDM.
Return MVDM
End Function

If a partition P is sorted in ascending order, the arrangement can be defined as P = \{P₁ ≤ P₂ ≤ P₃\}. The value to be selected for the replacement of the central value of a window will depend on the current value of P₂. The technique is presented in Eq. 1.

\[
\text{Mid-Value-Decision-Median} = \begin{cases} 
  P₁ & \text{if } P₂ = 255 \\
  P₂ & \text{if } 0 < P₂ < 255 \\
  P₃ & \text{if } P₂ = 0
\end{cases}
\]  

2.3.1 Proof of mid-value-decision-median

Each column for a 3 x 3 window has three (3) elements. Let use the variables A, B and C to represent the three values in the sorted column. Assuming that:

A stores zero (0): A = 0
B stores any value greater than zero (0) and less than 255: 0 < B < 255
C stores 255: C = 255

Let A and C represent impulse noise pixel values in a given column or partition. Given that a sorted column or partition is made up of the three (3) values, then the ascending order of column x can be presented as \(x₁ ≤ x₂ ≤ x₃\). Table 1. presents all the possible list of three (3) number derived from the variables A, B and C in ascending order, selected values for the classical median from the list and Mid-Value-Decision-Median values from the list.

From Table 1, the number of times the classical median selects an impulse value (A or C) as the median is 5 times (seq. 1, 2, 5, 8 and 9), while that of the Mid-Value-Decision-Median is 3 times (seq. 1, 5, and 9). This demonstrates that the proposed concept (Mid-Value-Decision-Median), when integrated into the median of medians approach, can effectively generate pixel values better than using the traditional median selection in the median of medians based filter. In sequence 1(A, A, A), 5 (A, C, C) and 9 (C, C, C) where the Mid-Value-Decision-Median selected impulse noise values, all the elements were noisy values (A or C). The estimated values generated by the Mid-Value-Decision-Median is impulse noise values only if all the values in the sliding window happen to be corrupted.

| Seq. | Ordered List (Ascending) using values A, B and C | Classical Median Value Selected | Mid-Value-Decision Median |
|------|-----------------------------------------------|--------------------------------|---------------------------|
| 1    | A, A, A                                       | A                              | A                         |
| 2    | A, A, B                                       | A                              | B                         |
| 3    | A, B, B                                       | B                              | B                         |
| 4    | A, B, C                                       | B                              | B                         |
| 5    | A, C, C                                       | C                              | A                         |
| 6    | B, B, B                                       | B                              | B                         |
| 7    | B, B, C                                       | B                              | B                         |
| 8    | B, C, C                                       | C                              | B                         |
| 9    | C, C, C                                       | C                              | C                         |
3. RESULTS AND DISCUSSION

3.1 Testing of High-performance Modified Decision Based Median Filter

The algorithms were implemented in MATLAB and the Matlab's inbuilt functions, tic and toc, used to estimate the running times of the algorithms (Scripts). The Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Metric (SSIM) were used to statistically evaluate the performances of the algorithms. These metrics measure how much noise remains in an image after denoising or restoration. Higher PSNR and SSIM values suggest excellent denoising or restoration by an algorithm. The Peppers, Lenna, and Mandrill images in Fig. 3 were used to evaluate the performances of the proposed algorithm. The images were corrupted with impulse noise of various densities. The proposed algorithm together with SMF, DBMF and MDBMF were made to denoise the corrupted images.

Images in Fig. 3 were selected as a result of the fact that they have become the benchmark images for evaluating the performances of salt-and-pepper denoising algorithms.

![Peppers, Lenna, and Mandrill images](image)

Fig. 3. (a) Peppers (b) Lenna and (c) Mandrill images

| Noise density | SMF  | DBMF | MDBMF | Proposed |
|---------------|------|------|-------|----------|
| 0.10          | 1.28 | 0.97 | 9.16  | 0.02     |
| 0.20          | 1.31 | 1.19 | 10.68 | 0.03     |
| 0.30          | 1.33 | 1.44 | 12.30 | 0.06     |
| 0.40          | 1.32 | 1.69 | 13.99 | 0.06     |
| 0.50          | 1.27 | 1.94 | 15.71 | 0.08     |
| 0.60          | 1.29 | 2.11 | 17.55 | 0.08     |
| 0.70          | 1.25 | 2.36 | 20.50 | 0.11     |
| 0.80          | 1.18 | 2.50 | 25.86 | 0.16     |
| Average Time  | 1.28 | 1.77 | 15.72 | 0.08     |

Table 2. CPU running times in sec of SMF, DBMF, MDBMF and HPDBMF on Peppers

| Noise density | SMF  | DBMF | MDBMF | Proposed MDBMF |
|---------------|------|------|-------|----------------|
| 0.10          | 1.76 | 1.26 | 11.97 | 0.03           |
| 0.20          | 1.74 | 1.61 | 14.06 | 0.04           |
| 0.30          | 1.74 | 1.89 | 15.97 | 0.06           |
| 0.40          | 1.70 | 2.20 | 18.33 | 0.07           |
| 0.50          | 1.72 | 2.62 | 20.71 | 0.09           |
| 0.60          | 1.66 | 2.73 | 23.54 | 0.11           |
| 0.70          | 1.66 | 3.05 | 27.60 | 0.14           |
| 0.80          | 1.61 | 3.29 | 34.81 | 0.23           |
| Average Time  | 1.70 | 2.33 | 20.87 | 0.10           |

Table 3. CPU running times in sec of SMF, DBMF, MDBMF and HPDBMF on Lenna
3.2 CPU Running Times of Algorithms

Tables 2, 3 and 4 present relative CPU running times for denoising Peppers, Lenna and Mandrill images respectively. The average running times for MDBMF and the Proposed Filter on Peppers are 15.72s and 0.08s, Lenna is 20.87s and 0.10s and Mandrill is 19.68s and 0.09s respectively. The averages for the three images are 18.76s for MDBMF and 0.09s for the proposed algorithm.

Table 4. CPU running times in sec of SMF, DBMF, MDBMF and HPDBMF on Mandrill

| Noise density | SMF | DBMF | MDBMF | Proposed MDBMF |
|---------------|-----|------|-------|----------|
| 0.10          | 1.65| 1.16 | 11.44 | 0.03     |
| 0.20          | 1.63| 1.56 | 13.24 | 0.04     |
| 0.30          | 1.61| 1.84 | 15.28 | 0.05     |
| 0.40          | 1.62| 2.11 | 17.52 | 0.07     |
| 0.50          | 1.58| 2.53 | 19.62 | 0.08     |
| 0.60          | 1.57| 2.64 | 22.11 | 0.11     |
| 0.70          | 1.54| 2.89 | 25.69 | 0.13     |
| 0.80          | 1.51| 3.10 | 32.53 | 0.21     |
| Average Time | 1.59| 2.23 | 19.68 | 0.09     |

Fig. 4. Relative CPU running times of SMF, DBMF, MDBMF and HPDBMF on peppers image

Fig. 5. Relative CPU running times of SMF, DBMF, MDBMF and HPDBMF on Lenna
Figs 4, 5 and 6 illustrate the running times of the various algorithms on the images used for the experiment. From the figures, the proposed algorithm performs best among all the algorithms. The Modified Decision Based Median Filter proposed by [25] performed worst among the algorithms. From the results presented in Tables 2, 3 and 4 the average running time of MDBMF is 18.76s, while that of HPMBMF is 0.09s. The ratio of MDBMF’s running time to HPMBMF’s running time can be estimated as \(208.4:1\). The running time of the proposed algorithms is therefore about 200 times faster than the MDBMF algorithm.

3.3 Psnr Values for Algorithms

Figs. 7, 8 and 9 illustrate the relative PSNR values recorded by the various algorithms used in the experiment. The results show that the proposed algorithm and the MDBMF generate outputs that are equivalent to all the noise densities that were used for the experiment. The SMF and DBMF have their performances dwindling significantly with increasing noise density while that of the HPMBMF and MDBMF performances were relatively stable with increasing noise density in the image.
Fig. 8. PSNR performance of various decision based median filters on Lenna

Fig. 9. Relative PSNR performance of various decision based median filters on Mandrill

3.4 SSIM Values for Algorithms

Figs. 10, 11 and 12 illustrate the relative SSIM values recorded by the various algorithms used in the experiment. Again the results for all the images show that the proposed algorithm and the MDBMF generate output images that are equivalent for all the noise densities that were used for the experiment. The SMF and DBMF performances for the SSIM also dwindle significantly with increasing noise density while that of the HPMDBMF and MDBMF are relatively stable with increasing noise density in the image.
Fig. 10. SSIM performances of various decision based median filters on Peppers

Fig. 11. SSIM performance of various decision based median filters on Lenna

Fig. 12. SSIM performance of various decision based median filters on Mandrill
Table 5. Peppers corrupted with noise and outputs generated by SMF, DBMF, MDBMF and Proposed MDBMF

| (a) Peppers corrupted with 20% of impulse noise | (b) Peppers corrupted with 40% of impulse noise | (c) Peppers corrupted with 60% of impulse noise |
| SMF output of image (a) | SMF output of image (b) | SMF output of image (c) |
| DBMF output of image (a) | DBMF output of image (b) | DBMF output of image (c) |
| MDBMF output of image (a) | MDBMF output of image (b) | MDBMF output of image (c) |
| Proposed algorithm (HPMDBMF) output of image (a) | Proposed algorithm (HPMDBMF) output of image (b) | Proposed algorithm (HPMDBMF) output of image (c) |

Table 5 presents corrupted peppers image restored by the various algorithms. From the table, MDBMF and HPMDBMF algorithms can generate better output when the density of noise increases. Peppers with 60% impulse noise were effectively denoised by both the proposed method and the MDBMF. The output of the HPMDBMF and MDBMF are similar because the algorithms use a similar concept as explained in the method. The technique that avoids altering of non-corrupted pixels as well as reducing the probability of the approximation technique
selecting impulse pixel for the denoise image make it possible for the proposed algorithm to achieve such as results. Again, the number of elements in a column enable the implementation to be done with variables. Accessing variables are less computationally expensive as compared with arrays which are used to implement the MDBMF. These techniques make the algorithm complete with a relatively minimal amount of time, that is about 200 times faster than MDBMF, and achieves equivalent output.

4. CONCLUSION

In this paper, an effective algorithm is proposed for denoising salt-and-pepper noises in real-time. The proposed algorithm, HPMDBMF, uses a median of medians approximation technique, the mid-value-decision-median technique, first non-corrupted pixel value encountered for replacement during the expansion of the 3 x 3 to 5 x 5 and limited access of the image array during the sorting phase to deliver an effective median based algorithm. These approaches, especially the mid-value-decision-median technique help to minimize a lot of the decisions that are taken by the general decision based median filters (DBMF and MDBMF). The HPMDBMF generates outputs that are equivalent to the state-of-the-art MDBMF but completes with an average speed of about 200 times faster than the MDBMF, making it ideal for real-time image processing and computer vision applications.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Goyal B, Dogra A, Agrawal S, Sohi BS, Sharma A. Image denoising review: From classical to state-of-the-art approaches. Information Fusion; 2020. DOI:10.1016/j.inffus.2019.09.003
2. Buades A, Coll B, Morel JM. A review of image denoising algorithms, with a new one. Multiscale Modeling and Simulation; 2005. DOI:https://doi.org/10.1137/040616024
3. Oliva D, Abd-Elaziz M, Hinojosa S. Image Processing. Studies in Computational Intelligence. 2019;27–45. DOI:10.1007/978-3-030-12931-6_4
4. Szeliski R. Computer Vision: Algorithms and Applications. Texts in Computer Science. Springer, London. DOI:https://doi.org/10.1007/978-1-84882-935-0_1
5. Asano A, Itoh K, Ichikawa Y. Optimization of the weighted median filter by learning. Optics Express. 1991;16(3):168–170.
6. Huber PJ. Robust statistics. 1st Ed. New York: Wiley; 1981.
7. Hampel FR, Ronchetti EM, Rousseeuw PJ, Stahel WA. Robust Statistics: The Approach Based on Influence Functions. 1st ed. Wiley: New Jersey; 2005. DOI:https://doi.org/10.1002/9781118186435
8. Stewart CV. Robust parameter estimation in computer vision. SIAM Review. 1999; 41(3):513–537. DOI:https://doi.org/10.1137/s003614459845802
9. Tukey JW. Exploratory data analysis. 1st ed. Addison-Wesley. Reading, MA; 1977.
10. Pitas I, Venetsanopoulos AN. Order statistics in digital image processing. Proc. IEEE. 1992;80(12):1893–1921.
11. Arce GR, Paredes JL. Recursive weighted median filters admitting negative weights and their optimization. IEEE Transactions on Signal Processing. 2000;48(3):768–779.
12. Forouzan AR, Araabi BN. Iterative median filtering for restoration of images with impulsive noise. Proceedings of the IEEE International Conference on Electronics, Circuits, and Systems. 2003;1(l):232–235. DOI:https://doi.org/10.1109/ICECS.2003.1302019
13. Liu L, Chen CLP, Zhou Y, You X. A new weighted mean filter with a two-phase detector for removing impulse noise. Information Sciences. 2015;315:1–16.
14. Paeth AW. Median finding on a 3 x 3 grid. Graphics Gems; 1990. DOI:https://doi.org/10.1016/b9780080507053.85.500449
15. Singh S. An Alternate Algorithm for (3x3) Median Filtering of Digital Images. International Journal of Computers & Technology. 2011;1(1):7–9. DOI:https://doi.org/10.24297/ijict.v1i1.6732
16. Huang TS, Yang GJ, Tang GY. A Fast Two Dimensional Median Filtering Algorithm. IEEE Transactions on Acoustics, Speech, and Signal Processing. 1979;27(1):13–18. DOI:https://doi.org/10.1109/TASSP.1979.163188
17. Weiss B. Fast median and bilateral filtering. ACM Transactions on Graphics; 2006; DOI:https://doi.org/10.1145/1141911.1141918

18. Perreault S, Hebert P. Median Filtering in Constant Time. IEEE Transactions on Image Processing. 2007;16(9):2389–2394. DOI:https://doi.org/10.1109/TIP.2007.902329

19. Zhu Y, Huang C. An Improved Median Filtering Algorithm for Image Noise Reduction. Physics Procedia. 2012;25:609–616. DOI:https://doi.org/10.1016/j.phpro.2012.03.133

20. Zhang Q, Xu L, Jia J. 100+ times faster weighted median filter (WMF). Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2014;2830–2837. DOI:https://doi.org/10.1109/CVPR.2014.362

21. Narendra PM. A separable median filter for image noise smoothing. Pattern Analysis and Machine Intelligence, IEEE Transactions on, PAMI. 1981;3(1):20–29.

22. Lu Y, Jiang L, Dai M, Li S. Sort optimization algorithm of median filtering based on FPGA. In 2010 International Conference on Machine Vision and Human Machine Interface, MVHI; 2010. DOI:https://doi.org/10.1109/MVHI.2010.145

23. Appiah O, Asante M, Hayfron-Acquah JB Adaptive approximated median filtering algorithm for impulse noise reduction. Asian Journal of Mathematics and Computer Research. 2016;12(2):134144.

24. Appiah O, Hayfron-Acquah BJ, Asante M. Approximation techniques for reducing running time of image pre-processing algorithms [Ph. D. Thesis]. Department of Computer Science, Kwame Nkrumah University of Science and Technology, unpublished.

25. Marcus RC, Ward WC. DP: A Fast Median Filter Approximation. Los Alamos National Lab.(LANL). Los Alamos-NM (United States) 2013;1–11. Available:https://permalink.lanl.gov/object/tt?what=info:lanl-repo/lareport/LA-UR-13-25331

26. Kunsoth R, Biswas M. Modified decision based median filter for impulse noise removal. 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai. 2016;1316–1319.

27. Dong Y, Xu S. A New Directional weighted median filter for removal of random-valued impulse noise. IEEE Signal Processing Letters, 2007;14(3):193–196. DOI:10.1109/lsp.2006.84014

28. Ko J, Lee YH. Center weighted median filters and their applications to image enhancement, IEEE Trans. Circuits Syst. 1991;38(9):984–993.

© 2020 Appiah et al.; This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Peer-review history:
The peer review history for this paper can be accessed here:
http://www.sdiarticle4.com/review-history/58045