International Journal of Image Processing and Vision Science

Volume 1 | Issue 2 Article 8

October 2012

CONSTANT TIME SCANNING AND BETTER EDGE PRESERVATION FOR BETTER PERFORMING AND QUALITY OF MEDIAN FILTER

RITWIK SHARMA
School of Science and Technology, IGNOU-I2IT Center of Excellence, Pune, India, sharmaritwik@yahoo.co.in

SHUBHAM HARNAL
School of Science and Technology, IGNOU-I2IT Center of Excellence, Pune, India, shubham.harnal@live.com

Recommended Citation
SHARMA, RITWIK and HARNAL, SHUBHAM (2012) "CONSTANT TIME SCANNING AND BETTER EDGE PRESERVATION FOR BETTER PERFORMING AND QUALITY OF MEDIAN FILTER," *International Journal of Image Processing and Vision Science*: Vol. 1 : Iss. 2 , Article 8.
DOI: 10.47893/IJIPVS.2012.1020
Available at: https://www.interscience.in/ijipvs/vol1/iss2/8

CONSTANT TIME SCANNING AND BETTER EDGE PRESERVATION FOR BETTER PERFORMING AND QUALITY OF MEDIAN FILTER

RITWIK SHARMA & SHUBHAM HARNAL

School of Science and Technology, IGNOU-I2IT Center of Excellence, Pune, India
E-mail : sharmaritwik@yahoo.co.in, shubham.harnal@live.com

Abstract **–** The median filter is an important filter in many image processing algorithms and especially in removal of salt and pepper noise. Traditional median filters either focus on improving the performance or the quality of the median filtering. Generally, the methods which optimize performance do so at the cost of quality and vice-versa. In this paper a novel approach to median filtering is presented providing both better performance and quality without sacrificing either. The analysis is presented with respect to image processing and the results obtained are presented in tabular form.

Keywords*-Median filters, image processing, algorithms, complexity.*

I. INTRODUCTION

Images are continuously subjected to noise and interference from several sources, which includes sensor noise and channel noise. These noises are random in nature and can severely affect the quality of an image. Image noises that are introduced from sources like noisy sensors or interference in transmission channels usually appear as a discrete isolated pixel variation that are not spatially correlated to other pixels in its neighborhood. Noise of this kind has a higher-spatial-frequency spectrum than normal image components due to its spatial de-correlation [4]. Such pixels with noise appear visually different from their neighbors and are unpleasant to human eyes.

Most common type of impulse noise is “salt and pepper” noise. It got its name because of its appearance as white and black dots superimposed on an image. Impulse noise is removed using order-statistic filters. Order-statistic filters[3] are non-linear spatial filters which are based on ordering of the pixels in the image area under the filter and substituting the center pixel with value determined with the ordering result. The most popular filtering method in this category is the median-filtering. In median filtering the intensity value of center pixel is replaced with median of the intensity values of its neighboring pixels.

With advances in hardware processing capabilities, usefulness of median filtering has been hampered by its processing time requirements. It is a non-linear method with per-pixel complexity of $O(r^2 \log r)$(where ‘r’ is processing kernel radius)[1]. Further by keeping number of pixel constant and using efficient sorting algorithms this complexity can be reduced to $O(r^2)$, which is still not good performance achievement. Another issue with median filter is that its output quality is severely compromised in extremely noisy images [2] and it has a tendency to distort edges in its final output.

The paper proposes a method to overcome stated problems with median filtering, which includes achieving better edge preservation with effective noise reduction while least compromising with the processing performance.

II. CONTEMPORARY METHODS OF MEDIAN FILTERING

In regular median filtering a processing kernel is taken with N x N (N is odd number) dimensions and traversed over the image. It is populated with intensity values of the pixels in the neighborhood of center pixel in kernel and the median of those intensities is computed and is assigned to the center pixel in the processing kernel.

Traditionally median filtering has been a computationally intensive operation, requiring operations of the order of $O(r^2)$ (r is kernel radius) [1].

There have been various attempts to address the computationally intensive aspect of median filters. These attempts have individually addressed either the performance or the quality of the median filter.

An example of addressing the performance of median filtering, is [1] in which the order of the computation required for the median filtering is reduced to constant time $O(1)$. This approach utilizes the property of median values that they do not vary too much over a given data sample. As a result, instead of gathering values of all pixel values for a given kernel radius, in this approach a running median is calculated; this running total is calculated and updated by scanning only a subset of the image pixels in a columnar way; the median value is continuously updated as the pixels are scanned in this columnar fashion; each time one set of pixels is discarded and a newer set of pixels is added into the calculation.

This algorithm provides good performance (constant time) however the image obtained as output has severe blurring, especially of the edges Other approaches have focused more on the quality of the noise removal more than the performance.

In [2], effective de-noising and better edge preservation is obtained by highly accurate noise detection. By more accurate noise detection, the output is better in terms of clarity and retention of edge preservation but although the method achieves better output in terms of the median filter, the performance is again degraded due to the processing being of the order of $O(r^2)$.

Thus, in existing methods of median filtering, generally a tradeoff between time and quality exists; either the algorithm is good in performance (time) but the output is degraded considerably or the output is good but requires heavy performance.

In the current approach, the traversal is maintained to be of constant time yet better quality is obtained by highly accurate noise detection.

III. IMPLEMENTATION NOTES

A. Platform Details

The current algorithm was prototyped using the OpenCV library which provides C bindings. The C programming language[9] was used to interface with the OpenCV library. The only functionality of OpenCV[7] used was the ability to read, write and display images. All the other routines for scanning and filtering were implemented.

The hardware setup consisted of an Intel 32-bit machine, (Intel Core Due 2.0 GHz) with 4GB RAM.

The code was tested to be platform agnostic and was tested on both Windows (Vista) and also on various unix-based Linux distributions such as Ubuntu, OpenSUSE.

B. Steps

The proposed method has been divided mainly into two parts: first to achieve better processing performance and another to preserve edges and have better noise reduction.

The first step for achieving better performance concerns with how the image is traversed to populate the kernel.

The algorithm maintains M (in M x N image) arrays with size 'r'(r x r is kernel size). The processing kernel is composed of 'r' arrays. Initially for the first pass all elements of the M arrays are populated with the corresponding pixel values from first 'r' rows in the 8 bit gray scale single channel image. From second pass the rightmost array is shifted one pixel down and its value is updated as per new pixel value from the image. Then whole kernel is shifted to one array right. Values of the kernel are passed to second step of proposed algorithm which computes the new value for the center pixel in the kernel. This step is repeated until the whole image has been traversed. In this method instead of updating all the r x r kernel pixels only one pixel is updated and one array is subtracted and another is added to the kernel.

The second step to improve de-noising and to preserve edges:

The passes kernel values are sorted and populated in a vector array A_V in ascending order with E_{min} representing minimum pixel value, E_{max} representing maximum pixel value, E_{med} for median pixel value and E_{cnt} for kernel's center value.

TABLE I. PSNR Values: Regular Median v/s Modified Median Filter

NOISE(%)	PSNR (db)		
	With Original Noise	***With Regular Median Filter***	***With Modified Median Filter***
5	40.21	34.96	36.34
10	37.11	34.74	35.63
15	35.34	34.44	34.99
20	34.11	34.33	34.52
25	33.14	34.08	34.12
30	32.37	33.78	33.61
35	31.66	33.44	33.23
40	31.12	33.19	33.02
45	30.55	32.57	32.42
50	30.12	32.2	32.23
55	29.7	31.66	31.86
60	29.34	31.16	31.61
65	28.97	30.56	31.31
70	28.69	30.08	31.17
75	28.37	29.54	30.92

- If the E_{cnt} lies within range $E_{min} < E_{cnt} < E_{max}$ and following conditions $E_{min} > 0$ and $E_{max} < 255$ are satisfied the E_{cnt} is considered uncorrupted and kept unchanged. But if any of these conditions fails E_{cnt} is considered as corrupted pixel.
- If E_{cnt} is corrupted and $E_{min} < E_{med} < E_{max}$ and $0 < E_{med} < 255$, E_{cnt} is replaced with E_{med}. If any condition in this step fails then next step is taken.
- Compute difference of adjacent pixels in vector array A_V and find the maximum difference between the adjacent pixels. Assign E_{cnt} with

corresponding pixel which had the maximum difference (the lower pixel value).

Repeat these steps until whole image is traversed.

IV. RESULTS

The main metric used to compare quality of median filtering is the PSNR value (Peak Signal to Noise Ratio. The higher the value of PSNR, the closer is the processed image to the original image.

Noise was added at various level to the original image over a range of 5% to 75% noise. PSNR values in Table I. were obtained after comparison of the original image with:

- Noisy images
- Images obtained after application of regular median filter
- Images obtained after applying the modified filter to the noisy image.

The PSNR values obtained by comparing modified median filter to those obtained by regular median filter, are consistently higher. This implies that the modified median filter is providing objectively better filtering as compared to the regular median filter.

As compared to complexity of regular median filtering ($O(r^2)$), the complexity of current filter is constant time in scanning so the performance is also better.

For a visual difference, refer to Figures 1 – 4.

As can be seen by comparing Figure 3 with Figure 4, the modified median filter is much effective in preserving edge effects and does not carry any residual noise as in the image with only the regular median filter applied. Thus the modified median filter has better output.

CONCLUSION

Two novel methods have been studied here to improve the performance and quality of median filtering for effective salt and pepper noise removal. For improving performance modified sliding window method has been employed to populate the kernel elements. The improvement has been achieved over normal population of kernel which has arithmetic complexity of $O(r^2)$ to arithmetic complexity of $O(1)$ as suggested in [1].

For quality improvement and effective removal of noise a modified median filtering method has been employed. Experiments suggest that the modified mean filter is very effective in removal of noise while preserving edges when the noise levels are moderate. For higher noise densities, the algorithm tends to introduce blurring effect in the images.

For further improvements, more efficient sorting methods for kernel elements can be utilized. Some other techniques can be used to preserve the blurring in highly corrupted images.

REFERENCES

[1] Simon Perreault* and Patrick H´ebert “Median Filtering in Constant Time”, IEEE Transactions on Image Processing, Vol 16, Issue-9.Sept. 2007.

[2] Gajanand Gupta “Algorithm for Image Processing Using Improved Median Filter and Comparision of Mean, Madian and Improved Medain Filter,” International Journal of Soft Computing and Engineering, Vol. 1,Issue-5. Nov 2011.

[3] Rafael C. Gonzalez and Richard E. Woods [2008]. Digital Image Processing Third Edition ,Pearson.”Order Static Filters” p.p. 156-157.

[4] William K. Pratt [2006]. Digital Image Processing PIKS Inside Third Edition, Wiley. “Nonlinear Noise Cleaning” p.p. 269-273.

[5] T. Huang, G. Yang, and G. Tang, “A Fast Two-Dimensional Median Filtering Algorithm,” IEEE Trans. Acoust., Speech, Signal Processing, vol. 27, no. 1, pp. 13–18, 1979.

[6] A.K. Jain, “Fundamentals of Digital Image Processing”, Prentice Hall of India, First Edition,1989.

[7] Gary Bradski & Adrian Kaehler, “Learning OpenCV”, OReiliy [2008].

[8] Wilhelm Burger and Mark J. Burge [2009]. “Principles of Digital Image Processing,” Springer. “Nonlinear Filters” p.p. 116-124.

[9] Herbert Schildt [2003]. The Complete Reference C++ Fourth Edition, Tata McGraw-Hill

◈◈◈