A Scheme for Salt and Pepper Noise Reduction and Its Application for OCR Systems

NUCHAREE PREMCHAISWADI[1], SUKANYA YIMGNAGM[2], WICHIAN PREMCHAISWADI[3]
[1]Faculty of Information Technology
Dhurakij Pundit University, Bangkok 10210, Thailand
E-mail: nucharee@dpu.ac.th
[2]Faculty of Resource and Environment,
Kasetsart University, Bangkok, Thailand
E-mail: yimngam@yahoo.com
[3]Graduate School of Information Technology,
Siam University, Bangkok 10163, Thailand
E-mail: wichian@siam.edu

Abstract: - This paper presents an algorithm for Salt and Pepper noise reduction which can be applied to binary, gray scale, and color formatted documents. The scheme combines the characteristics of the Applied kFill Algorithm and Median Filter Algorithm by using window sizes of 3x3 and 5x5, depending on the size of the Salt and Pepper noise. The goal of this technique is to increase the PSNR of picture images and improve the quality for scanning documents when using an optical character recognition (OCR) system. The experimental results show that the proposed scheme can remove Salt and Pepper noise better than the Applied kFill Algorithm and Median Filter Algorithm and can significantly improve the recognition accuracy of an optical character recognition (OCR) system.

Key-Words: - Applied kFill, kFill Algorithm, Image Processing, Median Filter, Noise Reduction, OCR systems, Salt and Pepper Noise.

1 Introduction

In document processing, scanning is the first step used to convert a paper document into an image document. The scanned images might be contaminated by additive noise and these low quality images will affect the next step of document processing. Therefore, a pre-processing step is required to improve the quality of images before sending them to subsequent stages of document processing [1-3]. The completeness of the input images also has an affect on the accuracy of a character recognition system [4-8].

There are many kinds of noise in images. One additive noise called "Salt and Pepper Noise", the black points and white points sprinkled all over an image, typically looks like salt and pepper, which can be found in almost all documents. A document usually uses a light background color to highlight text. The digitized result of these documents will generate salt and pepper noise in the background. Upon a closer inspection of many document images, salt and pepper noise components are found in binary, gray scale and color images.

Noise reduction is usually performed at a pre-processing stage in an image analysis process to improve the quality of the images [9]. Many methods have been proposed for removing salt and pepper noise from images such as the kFill Algorithm, Applied kFill Algorithm and Median Filter [10-12]. The kFill and Applied kFill Algorithms are capable of simultaneously removing both salt noise and pepper noise. However, these methods can only be used on binary images. The Median Filter method can be used to remove salt and pepper noise in binary, gray scale and color image. However, this method requires a long computation time. This paper presents a scheme which is a combination of the Applied kFill Algorithm and the Median Filter Algorithm to remove this type of noise in binary, gray scale and color images, with considerably less blurring than the Median Filter Algorithm while preserving useful detail in the image.

2 Related Works

2.1 kFill Algorithm [10-12]

The kFill filter is a scheme designed to reduce salt and pepper noise in images. For text images, in which the information is binary, salt and pepper noise is almost always prevalent. This noise appears as isolated pixels or pixel regions of ON noise in OFF backgrounds or OFF noise (holes) within characters and other foreground ON regions. In this algorithm, a black pixel is called ON while a white pixel is called OFF. The process of removing this noise called “Filling”.

In this algorithm, a window size of *k x k* pixels is moved over an image in the raster-scan direction. Inside the window, there are (k-2) x (k-2) regions, called the core, and 4(k-1) pixels on the window perimeter, called the neighborhood. The filling operation entails setting all values of the core to ON or OFF, depending on pixel values in the neighborhood. The decision of whether or not to fill with ON(OFF) requires that all core values must be OFF(ON), depending on two variables, determined from the neighborhood. For a fill-value equal to ON(OFF), the n variable is the number of ON-OFF pixels in the neighborhood, and c is the number of connected groups of ON-pixels in the neighborhood [10-11].

Filling occurred only when n is greater than a n and c are equal to 1. The value of n is set as a function of window size, n = 3k-4, to retain the text features described above. The stipulation that c=1 ensures that filling does not change connectivity (that is, does not join two letters together or separate two parts of the same connected letter). This method is performed iteratively on the image until no filling occurs and is only used in binary images. Fig. 5-8 shows the results of applying the kFill algorithm on a gray scale document image.

Fig. 5. An original image

Fig.6. The result of using the kFill algorithm on the image in Fig. 5.

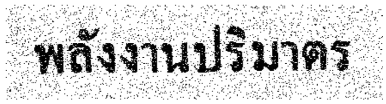


Fig.7. An original image after adding salt and pepper noise of 15%.

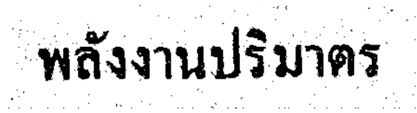


Fig.8. The result of using the kFill algorithm on the image in Fig. 7.

2.2 Applied kFill Algorithm [10-12]

In many document images, salt and pepper noise components are frequently larger than one pixel. In such cases, a window size larger than 3x3 pixels should be used, and the kFill Algorithm will never fill the noise components smaller than the core size, The kFill Algorithm will not fill the core region with neither ON or with OFF, because all core pixels are not the same value. This algorithm fills the core with OFF when the majority of pixels are ON or when the majority of pixels are OFF. This method is fast and effective. It can remove noise of different sizes and shapes while maintaining the sharpness of the text and graphical components. However, the algorithm can only be used on binary images. The results of the Applied kFill algorithm are shown in Figure 9-12.

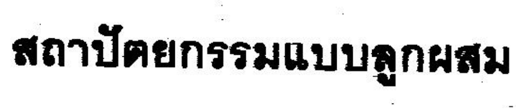


Fig. 9. An original image

Fig.10. The result of using the Applied kFill algorithm on the image in Fig. 9.

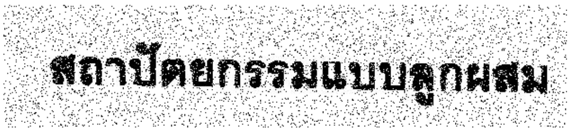


Fig.11. The original image in Fig.9 after adding salt and pepper noise of 15%.

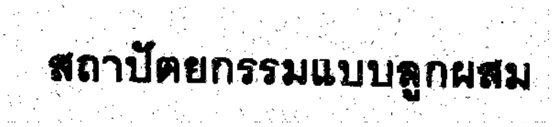
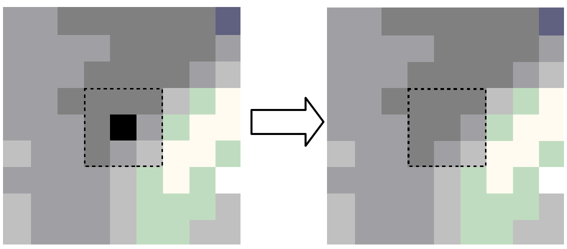


Fig.12. The result of using the Applied kFill algorithm on the image in Fig. 11.

2.3 Median Filter [9][13]

The median filter Algorithm is a non-linear digital filtering technique, often used to remove noise from images or other types of signals. The basic idea of this algorithm is to examine a sample of the input and decide if it is representative of the signal. This is performed using a window consisting of an odd number of samples. The values in the window are sorted into numerical order. The median value, the sample in the center of the window, is selected as the output. The oldest sample is then discarded and a new sample is acquired, and the calculation is repeated. Median filtering is a common step used in image processing. It is particularly useful for reducing speckle noise and salt and pepper noise. Its edge-preserving nature makes it useful in cases where edge blurring is undesirable. This is one of those techniques in which each pixel and its neighbors must be processed in turn. So select an example black pixel and a few of its neighbors as shown in Fig. 13 and the value of each pixel, which are presented in Table 1.



Before After

Fig. 13. An example of the Median Filter

Table 1 – The value of each pixel

129	113	132
126	0	166
139	168	202

The steps of the Median Filter procedure are described as:

1. Sort all the pixels in the neighborhood according to their strength :
   Sorted list: 0, 113, 126, 129,*132*, 139, 166, 168, 202
2. Take the pixel that is exactly in the middle of the 'sorted list' and replace the old 'object' pixel with the new 'middle' pixel. In other words, the old black pixel is replaced with the value of *132* (If the neighborhood under consideration contains an even number of pixels, the average of the two middle pixel values is used) Figure 14-16 illustrates a result.

Fig. 14 Original image

Fig. 15 Added Salt and Pepper Noise 15%

Fig. 16 After Median Filter

3 The Proposed Algorithm

The proposed algorithm is an extension of the author's work [7][8] to remove salt and pepper noise in binary, gray scale and color images. In this algorithm, a black pixel is defined as ON, a white pixel as OFF and k is the window size, similar to that used in the kFill algorithm. The steps of the algorithm are as the following:

- Count ON or OFF pixels of the core ((k-2)*(k-2)).
- If the numbers of ON or OFF pixels in the core are more than half of all the pixels in the core $(((k-2)^2/2)+1)$, then a decision is made to fill the core with the median pixel value from the core and the surrounding neighborhood.
- If the numbers of ON or OFF pixels in the core are less than one half of all pixels in the core $(((k-2)^2/2)+1)$, then a decision is made to fill the core with the median pixel value from the core.
- If all pixels in the core are not ON or OFF then fill the core with the original pixels values.

A flowchart of proposed algorithm is shown in the Fig. 17.

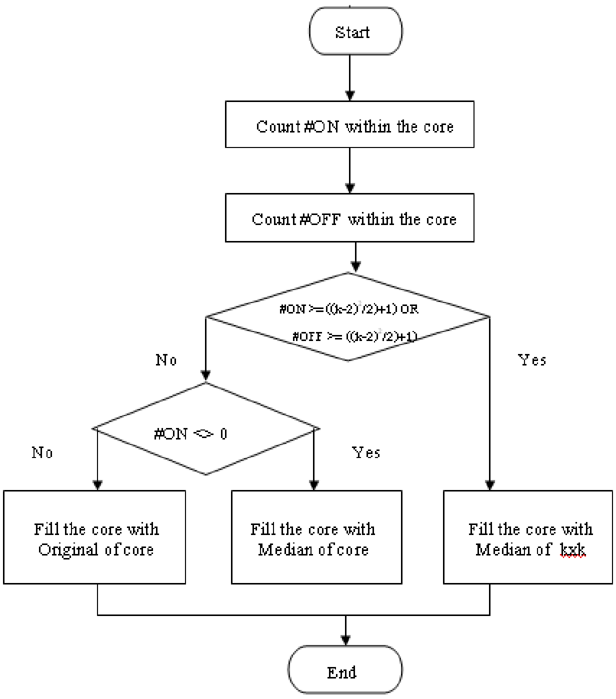


Fig. 17 Flowchart of the proposed algorithm

4 Peak signal-to-noise ratio (PSNR) [14]

The PSNR is most commonly used method to measure the quality of reconstructed images. The calculation used by this method is based on the mean squared error (MSE) which is defined as:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}||I(i,j) - K(i,j)||^2$$

Where $I(i,j)$ refers to the original image, $K(i,j)$ represents the approximated version and m, n are the dimensions of image. The PSNR is defined as:

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20 \cdot \log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

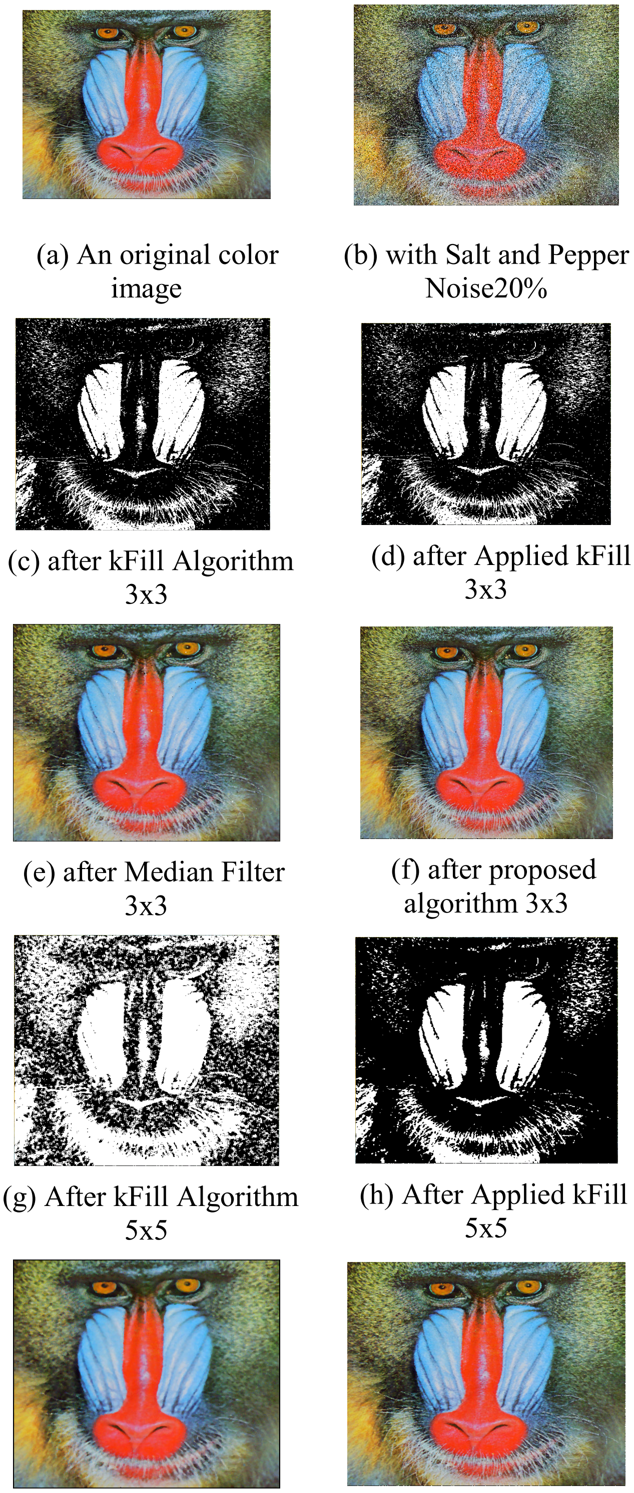
Here, MAX_i is the maximum possible pixel value in the image. For color images with three RGB values per pixel, the definition of PSNR is the same except the MSE is the sum over all squared value differences divided by image size and then divided by three. Typical values for the PSNR in a lossy image are between 30 and 50 dB, where a higher value is better.

5 Experimental results

The proposed scheme was tested with color and gray scale images and also tested with OCR systems.

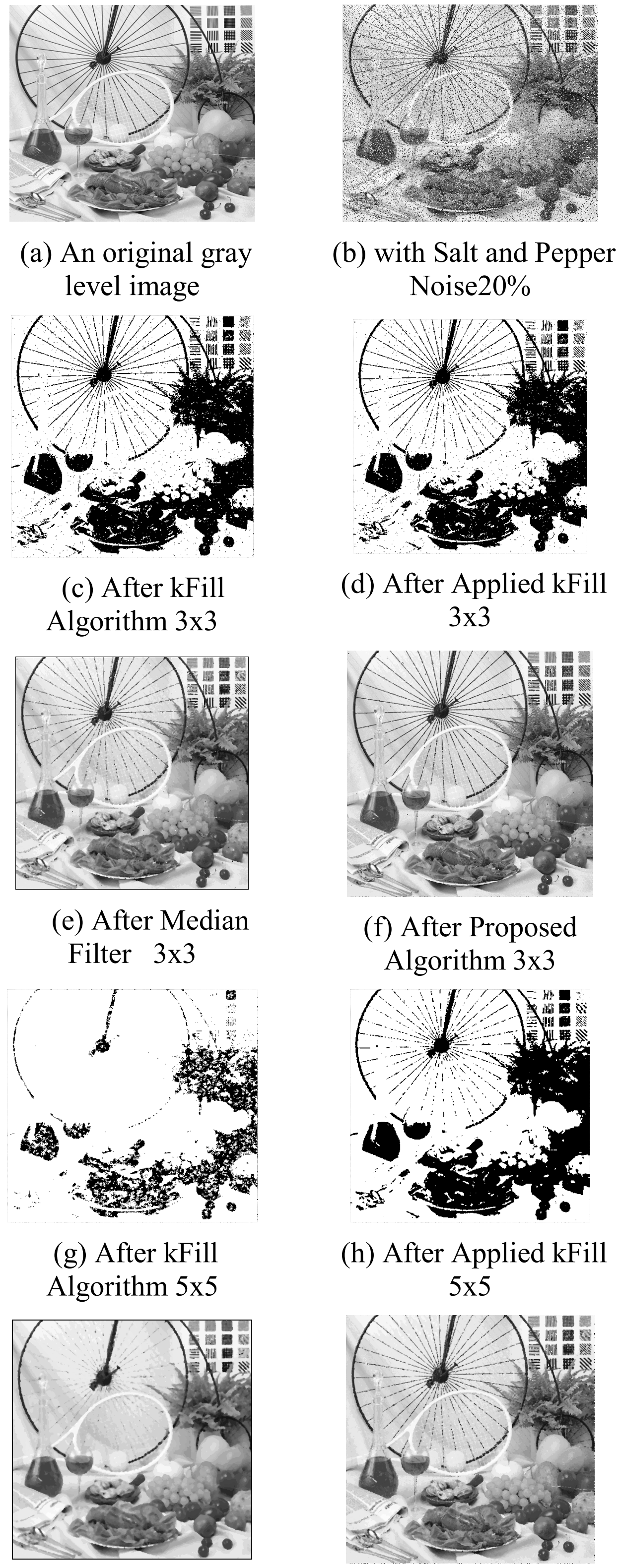
5.1 Color and Gray scale Images

The proposed algorithm was implemented and tested with 10 color images and 10 gray scale images which were downloaded from an image database website. In the experiment, each image had salt and pepper noise added with probabilities ranging from 5% to 30%. Then, each image was tested using window sizes of 3 x 3 and 5 x 5. The experimental results of the proposed algorithm were then compared with the kFill Algorithm, Applied kFill Algorithm, and the Median Filter algorithm. Some of the experimental results of the tested color and gray scale images are shown in Fig. 18 and Fig. 19, respectively.



(a) An original color image
(b) with Salt and Pepper Noise20%
(c) after kFill Algorithm 3x3
(d) after Applied kFill 3x3
(e) after Median Filter 3x3
(f) after proposed algorithm 3x3
(g) After kFill Algorithm 5x5
(h) After Applied kFill 5x5
(i) After Median Filter 5x5
(j) After Proposed Algorithm 5x5

Fig. 18: The results of a gray scale image with a window size of 3x3 and 5x5



(a) An original gray level image
(b) with Salt and Pepper Noise20%
(c) After kFill Algorithm 3x3
(d) After Applied kFill 3x3
(e) After Median Filter 3x3
(f) After Proposed Algorithm 3x3
(g) After kFill Algorithm 5x5
(h) After Applied kFill 5x5
(i) After Median Filter 5x5
(j) After Proposed Algorithm 5x5

Fig. 19: The results of a gray scale image with a window size of 3x3 and 5x5

Comparison of the PSNR for color images of the proposed scheme and other methods with a window size of 3x3 and 5x5 are shown in Table 2 and Table 3, respectively.

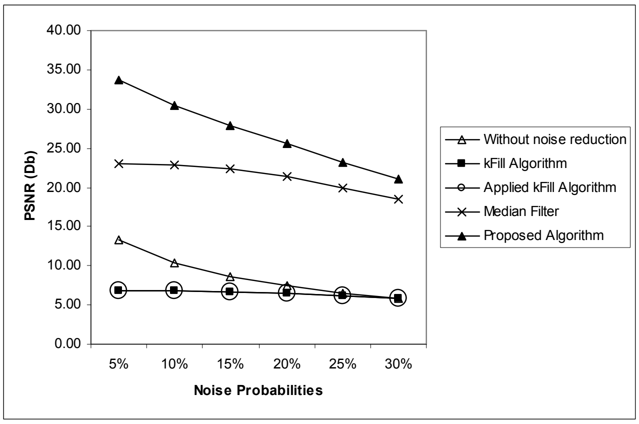
Table 2 - The comparison of PSNR of color images of window size of 3x3

Algorithms	PSNR Window size of 3x3 Noise Probability (%)					
	5%	**10%**	**15%**	**20%**	**25%**	**30%**
Original image with noise	13.23	10.31	8.59	7.42	6.52	5.82
kFill Algorithm	6.84	6.77	6.63	6.42	6.14	5.83
Applied kFill Algorithm	6.84	6.77	6.63	6.42	6.14	5.83
Median Filter	23.05	22.80	22.29	21.36	19.90	18.50
Proposed Algorithm	33.69	30.50	27.91	25.57	23.12	21.10

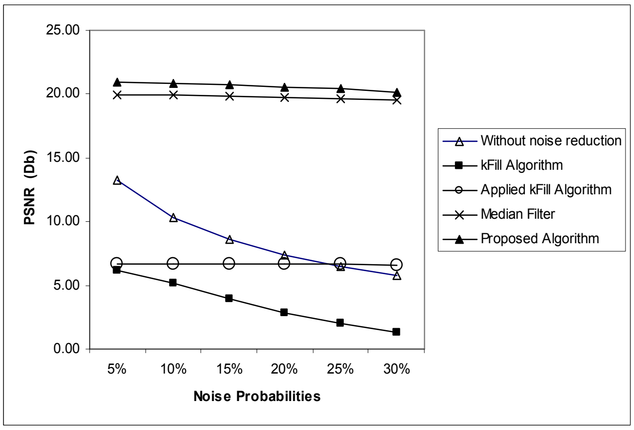
Table 3 - The comparison of PSNR of color images of window size of 5x5

Algorithms	PSNR Window size of 5x5 Noise Probability (%)					
	5%	**10%**	**15%**	**20%**	**25%**	**30%**
Original image with noise	13.23	10.31	8.59	7.42	6.52	5.82
kFill Algorithm	6.16	5.14	3.96	2.87	2.00	1.35
Applied kFill Algorithm	6.72	6.71	6.69	6.67	6.64	6.58
Median Filter	19.99	19.92	19.84	19.76	19.68	19.55
Proposed Algorithm	20.92	20.87	20.74	20.58	20.40	20.12

Comparison graph of the PSNR for color images of the proposed scheme and other methods using a window size of 3x3 and 5x5 are also shown in Fig.20.



a) Window size of 3x3



b) Window size of 5x5

Fig. 20 The graph of PSNR of color images

Table 4 - The comparison of PSNR of gray level images of window size of 3x3

Algorithms	**PSNR** **Window size of 3x3** **Noise Probability (%)**					
	5%	**10%**	**15%**	**20%**	**25%**	**30%**
Original image with noise	11.86	8.73	7.21	6.05	5.13	4.42
kFill Algorithm	5.81	5.69	5.57	5.33	5.01	4.66
Applied kFill Algorithm	5.81	5.69	5.57	5.33	5.01	4.66
Median Filter	21.19	20.78	20.40	19.55	18.14	16.61
Proposed Algorithm	30.30	27.52	25.02	23.10	21.15	19.19

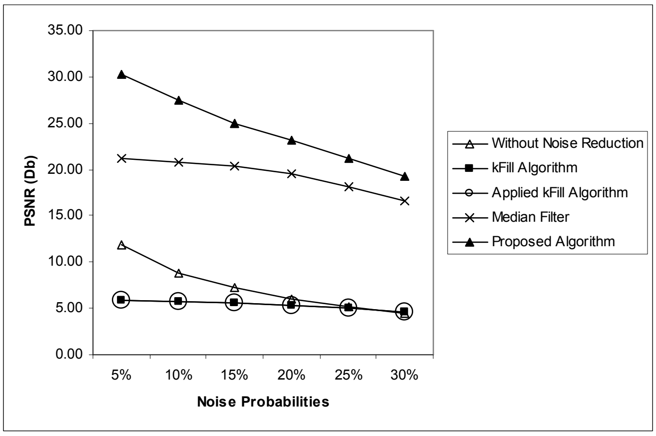
Table 5 - The comparison of PSNR of gray level images of window size of 5x5

Algorithms	PSNR					
	Window size of 5x5 Noise Probability (%)					
	5%	10%	15%	20%	25%	30%
Original image with noise	11.86	8.73	7.21	6.05	5.13	4.42
kFill Algorithm	4.43	3.52	2.41	1.38	0.10	-0.66
Applied kFill Algorithm	5.48	5.47	5.46	5.44	5.40	5.33
Median Filter	18.05	17.98	17.93	17.91	17.81	17.69
Proposed Algorithm	18.00	18.04	17.98	17.91	17.74	17.56

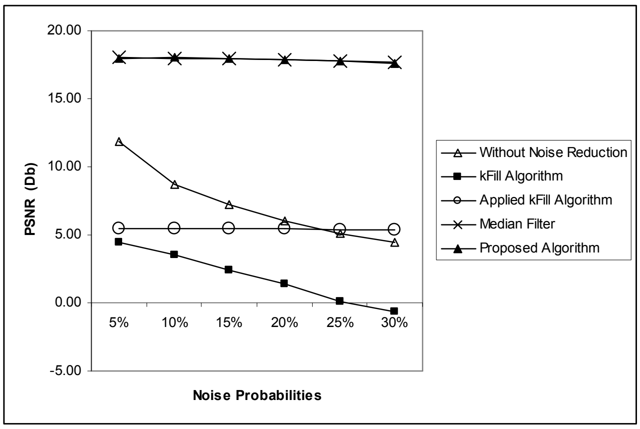
Comparison of the PSNR for gray scale images of the proposed scheme and other methods with a window size of 3x3 and 5x5 are shown in Table 4 and Table 5, respectively.
A comparison graph of the PSNR for gray scale images of the proposed scheme and other methods with the window size of 3x3 and 5x5 are also shown in Fig.21.

5.2 Its Application on OCR systems

The proposed scheme was also tested with 24 documents. These documents were scanned with a resolution of 300 x 300 dpi. In the experiments, salt and pepper noise was added to these image documents with a probability ranging from 5% to 30%. The results of the experiment were used to compare the proposed algorithm using a window size of 3 x 3 and 5 x 5 depending on the size of noise in the image with the Median Filter Algorithm, the kFill Algorithm, and the Applied kFill Algorithm. The experimental scanned results were used as the input to two commercials OCR software systems in the Thai language namely: ThaiOCR version 1.5 and ArnThai version 1.0. Both using a window sizeof 3x3 and 5x5. The results after filtering are shown in Fig. 22 and Fig. 23.

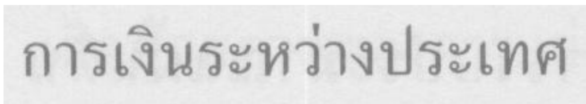


a)Window size of 3x3

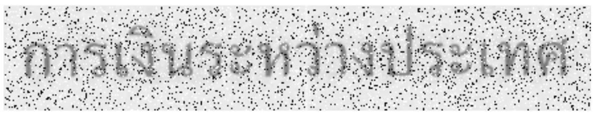


b) Window size of 5x5

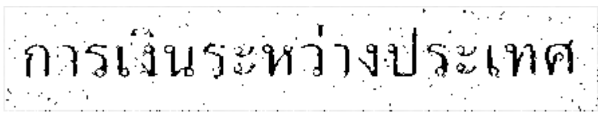
Fig. 21 The graph of PSNR for gray scale images



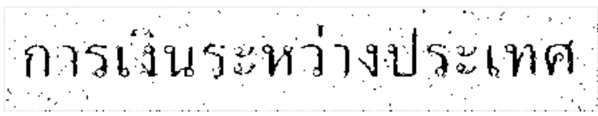
a) original gray scale image



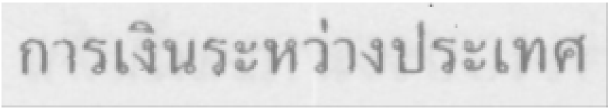
b) original image with Salt and Pepper Noise 15%



c) after the kFill Algorithm 3x3



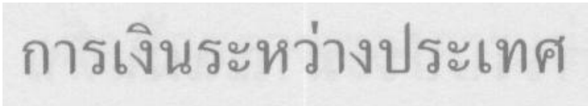
d) after the Applied kFill Algorithm 3x3



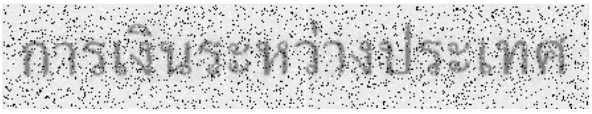
e) after the Median Filter 3x3

f) after the proposed algorithm 3x3

Fig. 22. Using a window size of 3x3



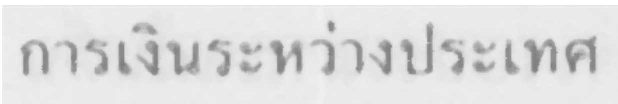
a) original gray scale image



b) original image with Salt and Pepper Noise 15%

c) after the kFill Algorithm 5x5

d) after the Applied kFill Algorithm 5x5



e) after the Median Filter 5x5

f) after the proposed algorithm 5x5

Fig. 23. The used of a window size of 5x5

The result of applying the proposed scheme to documents and then processing them using an OCR system is shown in Table 6-9.

Table 6 The recognition results of OCR software using a window size of 3x3

Text Type	**ThaiOCR 1.5** **Noise Probability (%)**					
	5%	10%	15%	20%	25%	30%
Original image with noise	60.9	30.2	18.9	10.1	6.2	3.7
kFill Algorithm	65.2	42.8	28.5	18.9	13.2	6.8
Applied kFill Algorithm	62.5	46.1	28.7	20.0	13.0	6.8
Median Filter	63.4	57.8	49.3	45.5	28.5	24.0
Proposed Algorithm	75.4	62.8	57.4	46.7	45.6	35.5

Table 7 The recognition results of OCR software using a window size of 3x3

Text Type	**ArnThai 1.0** Noise Probability (%)					
	5%	10%	15%	20%	25%	30%
Original image with noise	71.1	61.0	53.2	38.4	30.9	19.4
kFill Algorithm	77.9	70.2	67.9	54.1	50.2	46.5
Applied kFill Algorithm	77.9	68.2	67.9	54.1	49.8	46.5
Median Filter	68.2	59.7	56.0	50.3	47.1	43.6
Proposed Algorithm	75.1	73.7	72.0	63.1	58.7	59.6

Table 8 The recognition results of OCR software using a window size of 5x5

Text Type	ThaiOCR 1.5 Noise Probability (%)					
	5%	10%	15%	20%	25%	30%
Original image with noise	51.03	28.58	17.10	12.17	6.51	3.19
kFill Algorithm	7.60	2.32	2.44	1.25	1.30	1.01
Applied kFill Algorithm	57.02	44.23	36.62	29.36	14.44	15.04
Median Filter	16.78	19.63	13.14	8.12	8.98	7.56
Proposed Algorithm	53.09	37.47	40.50	29.01	27.43	19.54

Table 9 The recognition results of OCR software using a window size of 5x5

Text Type	ArnThai 1.0 Noise Probability (%)					
	5%	10%	15%	20%	25%	30%
Original image with noise	71.82	61.4	55.29	40.78	34.01	23.88
kFill Algorithm	16.03	10.40	12.16	6.87	7.90	6.82
Applied kFill Algorithm	61.89	58.41	54.87	43.81	38.85	36.98
Median Filter	48.61	44.10	36.38	34.84	28.38	27.83
Proposed Algorithm	68.99	63.37	61.67	54.34	53.20	48.38

A comparison of the effectiveness of the proposed scheme with other methods is also shown in Fig. 24-28.

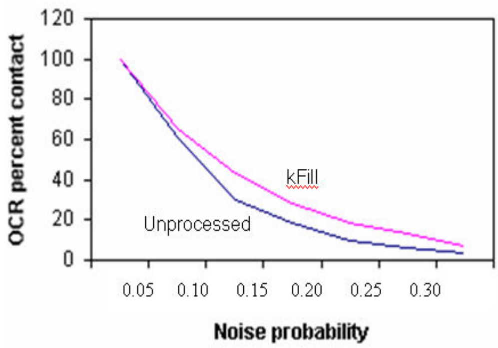


Fig. 24. A comparison of the proposed scheme with kFill

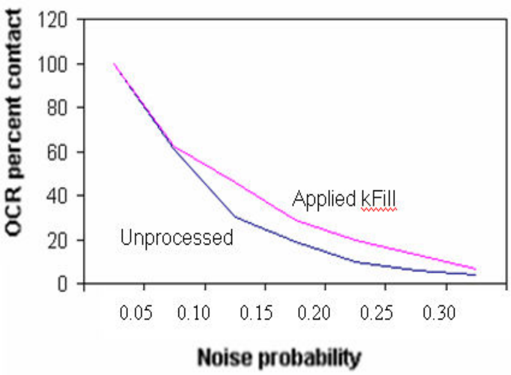


Fig. 25. A comparison of the proposed scheme with Applied kFill

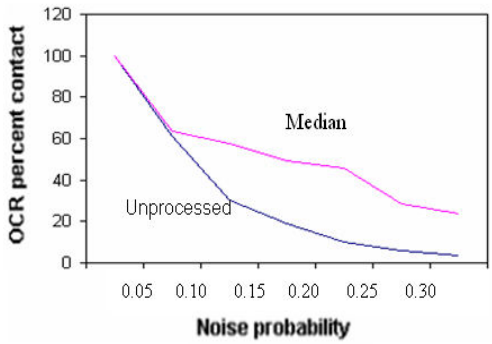


Fig. 26. A comparison of the proposed scheme with Median Filter

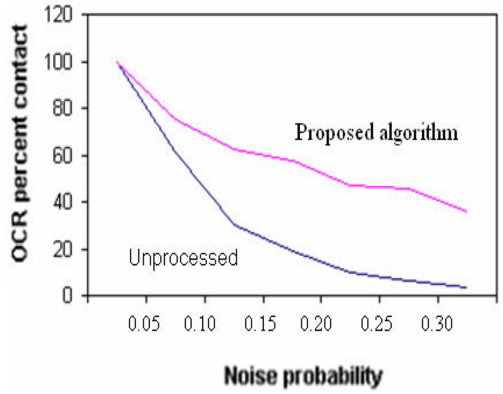


Fig. 27. A comparison of the proposed scheme with no noise reduction

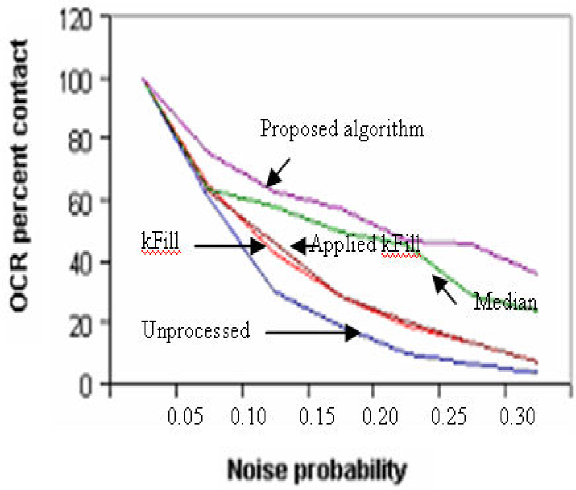


Fig. 28. A comparison of the proposed scheme with all other methods

6 Conclusion and discussion

This paper presents an algorithm for salt and pepper noise reduction in image documents. This algorithm is a combination of the Median Filter and Applied kFill Algorithms. The proposed algorithm can remove Salt and Pepper noise of any size that is smaller than the size of document objects. This method is fast and can be used effectively on binary, gray scale and color images, with considerably less blurring than other methods and at the same time preserving useful details in the image. The experimental results show that this proposed scheme can significantly increase the PSNR of color and gray scale images. It can be used to remove noise of difference sizes depending on the amount of noise. The experimental results also show that this proposed scheme can significantly improve the character recognition rate of commercially available OCR software.

References:

[1] R.C. Gonzalez and R.E. Woods (1992) Digital Image Processing, Addison-Wesley.

[2] National Institute of Standard and Technology (2003) Noise Reduction The Rank Filter [Online]. Available http://www.nist.gov/lispix/imlab/noise/shottc.html.

[3] Krisana Chinnasarn, Yuttapong Rangsanseri and Punya Thitimajshima (1998) Removing Salt-and-Pepper Noise in Text/Graphics Images. IEEE Computer Society Press

[4] G.A. Story, L.O'Gorman and D.Fox.(1992) The RightPages Image-Based Electronic Library for Alerting and Browsing, Computer, Vol. 25, No.9, Sept. 1992, pp.17-26.

[5] L.O'Gorman and R. Kasturi.(1995) Document Image Analysis, IEEE Computer Society Press

[6] Wikipedia, the free encyclopedia "Median Filter" [Online]. Available :http://en.wikipedia.org/wiki/ Median_filter

[7] Wikipedia, the free encyclopedia "Peak signal-to-noise ratio" [Online]. Available : http://en.wikipedia.org /wiki/Peak_signal-to-noise_ratio

[8] N. Premchaiswadi, S. Yimngam and W. Premchaiswadi (2007) An Input Improvement for an Optical Character Recognition System using Noise Reduction, Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2007*,* pp. 2136-2143

[9] N.Premchaiswadi, W. Premchaiswadi, U. Pachiyankul, S.Narita, (2003) Broken Characters Identification for Thai Character Recognition Systems, WSEAS Transactions on Computers, Issue 2, Volume 2, April 2003, pp.430-434

[10] N.Premchaiswadi, W. Premchaiswadi, S. Duangphasuk, S.Narita, (2003) SMILE: Printed Thai Character Recognition Systems, WSEAS Transactions on Computers, Issue 2, Volume 2, April 2003, pp.424-429

[11] W.Premchaiswadi, P. Sutheebanjard, N. Premchaiswadi, (2007) A Speed Enhancement Method for Document Page Segmentation Using Window and Optimum Image, WSEAS Transactions on Computers, Issue 3, Volume 6, March 2007, pp.500-506

[12] N. Premchaiswadi, S. Yimngam and W. Premchaiswadi (2009) A Scheme for Salt and Pepper Noise Reduction on Graylevel and Color Images Proceedings of the 9th WSEAS International Conference on SIGNAL PROCESSING, COMPUTATIONAL GEOMETRY and ARTIFICIAL VISION, pp.57-61

[13] L. Khriji, M. Merribout, M. Gabbouj, S.Akkari, (2003) Color Picture Enhancement using Rational Unsharp Masking-Based Approach, WSEAS Transactions on Computers, Issue 2, Volume 2, April 2003, pp.398-402

[14] W.Premchaiswadi, N. Premchaiswadi, S. Aphiwongsophon ans S. Narita (2003) A Scheme for Form Identification System, WSEAS Transactions on Computers, Issue 2, Volume 2, April 2003, pp.449-453