

國立交通大學

電控工程研究所

碩士論文

應用廣義加權平均集成運算於影像壞點偵測
Applying Weighted Generalized Mean Aggregation to
Noise Detection of Images

研究生：陳冠霖

指導教授：張志永

中華民國一百零二年七月

應用廣義加權平均集成運算於影像壞點偵測
Applying Weighted Generalized Mean Aggregation to
Noise Detection of Images

學 生：陳冠霖 Student : Kuan-Lin Chen

指導教授：張志永 Advisor : Jyh-Yeong Chang

國立交通大學

電機工程學系

碩士論文



Submitted to Department of Electrical Engineering

College of Electrical and Computer Engineering

National Chiao-Tung University

in Partial Fulfillment of the Requirements

for the Degree of Master in

Electrical Control Engineering

July 2013

Hsinchu, Taiwan, Republic of China

中華民國 一 百 零 二 年 七 月

應用廣義加權平均集成運算於影像壞點偵測

學生：陳冠霖

指導教授：張志永博士

國立交通大學電控工程研究所

摘要

本論文應用廣義加權平均建立的區間值模糊關係進行灰階影像壞點偵測。首先，我們使用兩個加權參數，對整張影像以 3×3 視窗內中心像素和其八鄰域像素進行加權平均集成運算。接著，為了避免單一組大的差值掩蓋其它組差值呈現，我們應用飽和門檻值轉換函數(Saturation threshold transfer function)在像素差值上。最後，經由門檻值(Threshold)作用後，可獲得影像壞點較合理的估測。

為了降低壞點誤判的個數，我們藉著離散型梯度演算法的概念進行加權平均參數的學習。另外，為了修正影像更高的訊雜比，相對於好點誤判，我們加重壞點誤判的權重約 20-100 倍。除此之外，我們更提出兩個不同的訓練階段，以維持影像的銳利度。從四張添加脈衝雜訊的灰階合成影像訓練結果顯示，整合區間值模糊關係與加權平均差值集成演算法，能產生更為強健的壞點偵測。

Applying Weighted Generalized Mean Aggregation to Noise Detection of Images

STUDENT: Kuan-Lin Chen

ADVISOR: Dr. Jyh-Yeong Chang

Institute of Electrical Control Engineering
National Chiao-Tung University

ABSTRACT

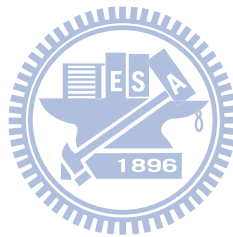
In this thesis, we apply weighted generalized mean to construct interval-valued fuzzy relations for grayscale image noise detection. First, we employ two weighting parameters and perform the weighted mean aggregation for the central pixel and its eight neighbor pixels in a 3×3 sliding window across the image. Then, in order to counter the over-weighting of a big difference term, we apply a saturation threshold transfer function to pixel difference values. Finally, the image noise map is obtained through a threshold operation.

In order to decrease the noise detection error, weighting parameters of the mean can be learned by the gradient method cast in discrete formulation. Moreover, to get higher PSNR in the corrected image, we have, in the training, put multiple weight ranging from 20 to 100, on erroneous noisy than that on the erroneous non-noise pixel. Besides, we also propose two training stages for the purpose of maintaining image sharpness and correction. By the training results of four grayscale natural images with adding impulse noises, we have shown that the integration of interval-valued fuzzy relations with the weighted mean aggregation algorithm can effectively detect the image noise and do the correction hereafter.

ACKNOWLEDGEMENTS

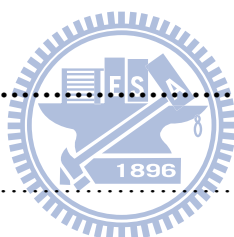
I would like to express my sincere gratitude to my advisor, Dr. Jyh-Yeong Chang for valuable suggestions, guidance, support and inspiration he provided. Without his advice, it is impossible to accomplish this research. Thanks are also given to all of my lab members for their suggestion and discussion.

Finally, I would like to express my deepest gratitude to my family for their concern, supports and encouragements.

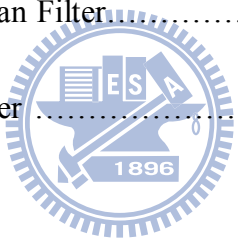


Contents

摘要	i
ABSTRACT	ii
ACKNOWLEDGEMENTS	iii
Contents	iv
List of Figures	vii
List of Tables	x
Chapter 1 Introduction	1
1.1 Motivation	1
1.2 Noisy Pixel Detection and Correction	2
1.2.1 Noise Model	2
1.2.2 Noisy Pixel Correction	3
1.3 Research Method	5
1.4 Thesis Outline	6
Chapter 2 Apply Interval-Valued Fuzzy Relation to Image Noise Detection	7
2.1 Fuzzy Relation	9



2.2	Lower and Upper Constructor	10
2.3	Construction of Interval-Valued Fuzzy Relation	13
2.4	W-Fuzzy Relation	13
Chapter 3 Some Noisy Pixel Detection and Correction Methods.....		14
3.1	Peer Group Filter	14
3.2	Adaptive Peer Group Filter	16
3.3	Fast Similarity-Based Impulsive Noise Removal Vector Filter	19
3.4	Center Weighted Median Filter.....	21
3.5	Switching Median Filter	24
Chapter 4 Parameter Learning of Weighted Mean Based Noise Pixel		
Detection.....		28
4.1	Weighted Mean Based Interval-Valued Fuzzy Relation	29
4.2	The Weighted Mean to Calculate the Difference of Neighbor Pixels of Images	30
4.3	Operating Parameter Learning Mechanism	33
4.3.1	Learning Rule of Mean Weighting Parameters	34
4.3.2	Proposed Training Method	40



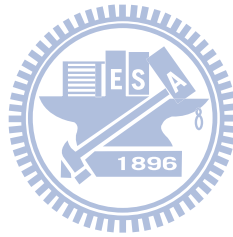
Chapter 5 Experimental Results42

5.1 Results of Salt and Pepper Noise Correction44

5.2 Results of Random-Valued Impulse Noise Correction.....55

Chapter 6 Conclusions66

References67



List of Figures

Fig. 2.1.	The flowchart to obtain noise map by using interval-valued fuzzy relation	8
Fig. 2.2.	Example of upper constructor	12
Fig. 3.1.	Example for illustrating FPGF process.....	16
Fig. 3.2.	The working window size of $n \times n$, ($n=3, 5, 7, \dots$).....	17
Fig. 3.3.	Block diagram of the Adaptive Peer Group Filter.....	18
Fig. 3.4.	(a) First the cumulative similarity value M_0 between the central pixel F_0 and its neighbors is calculated. (b) Then pixel F_0 is rejected from the filter window and the cumulative similarity values M_k , $k = 1, \dots, n$, of the pixels F_1, \dots, F_n are determined [21].	21
Fig. 3.5.	Block diagram of the Center-Weighted Median Filter.....	24
Fig. 3.6.	Four directional convolutionary kernels.....	26
Fig. 4.1.	Four 10% impulse noise corrupted gray-scale natural images of size 128×128	29
Fig. 4.2.	Noise ground truths of four 10% impulse noise corrupted gray-scale natural images.....	29
Fig. 4.3.	The diagram of a 3×3 window centered at pixel $I(m, n)$ and its eight neighborhood pixels.	31
Fig. 4.4.	Sigmoid function for saturation thresholding	32
Fig. 5.1.	Four 10% salt and pepper noise training images with size of 128×128 . .	44
Fig. 5.2.	Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 20% salt and pepper noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive	

Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method. 46

Fig. 5.3. Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 40% salt and pepper noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method. 48

Fig. 5.4. Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 60% salt and pepper noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method. 50

Fig. 5.5. Four 10% random-valued impulse noise training images with size of 128×128 55

Fig. 5.6. Noisy pixel correction results of Pepper image filtered by different filters. (a) Original image. (b) Corrupted image with 20% random-valued impulse noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage.

(e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method..... 57

Fig. 5.7. Noisy pixel correction results of Pepper image filtered by different filters.

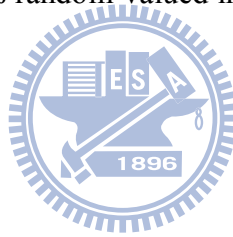
(a) Original image. (b) Corrupted image with 40% random-valued impulse noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method..... 59

Fig. 5.8. Noisy pixel correction results of Boat image filtered by different filters. (a)

Original image. (b) Corrupted image with 60% random-valued impulse noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method..... 61

List of Tables

TABLE I	THE NOISE REMOVAL RESULTS BY DIFFERENT FILTERS: (a) Corrupted Boat image with 10% salt and pepper noise, (b) Corrupted Boat image with 20% salt and pepper noise, (c) Corrupted Boat image with 40% salt and pepper noise, (d) Corrupted Boat image with 60% salt and pepper noise.....	51 – 54
TABLE II	THE NOISE REMOVAL RESULTS BY DIFFERENT FILTERS: (a) Corrupted Pepper image with 10% random-valued impulse noise, (b) Corrupted Pepper image with 20% random-valued impulse noise, (c) Corrupted Pepper image with 40% random-valued impulse noise, (d) Corrupted Boat image with 60% random-valued impulse noise.....	62 – 65



Chapter 1 Introduction

1.1 Motivation

Noise detection plays an important role in image processing and computer vision. During the image acquisition and transmission process, the quality of digital images may be affected by impulse noise, such as malfunctioning pixels in camera sensors, transmission in a noisy channel or faulty memory locations in hardware. Two common types of impulse noise are the salt-and-pepper noise and random-valued impulse noise. For images corrupted by salt-and-pepper noise, the noisy pixels can take only the maximum or minimum values, if the images are corrupted by random-valued impulse noise, the noisy pixels can take any random value.

Efficient detection of noise from image data is of key importance in most image processing applications, because the performances of subsequent image processing tasks are dependent on the success of the noise detection theory. There are many works on the restoration of images corrupted by impulse noise, the standard median filter was the most popular nonlinear filter because of its good denoising power and computational efficiency. However, when the noise level is high in an image, their main drawback is that the noisy pixels are replaced by the median value without considering local features such as the presence of edges, some details and edges of the original image are smeared by the filter.

Different remedies of the median filter have been proposed, e.g., the Adaptive Median Filter [1], Weighted Median Filter [2] and the Center Weighted Median Filter [3], these so-called “decision-based” or “switching” filters first detect possible noisy pixels and then replace them by using the median filter.

In this thesis, a new impulse noise detector is proposed. We apply weighted mean

to construct interval-valued fuzzy relations for grayscale image noise detection. More precisely, we first employ two weighting parameters, and perform the weighted mean aggregation for the central pixel and its eight neighbor pixels in a 3×3 sliding window across the image. Then, in order to counter the over-weighting of a big difference term, we apply a saturation threshold transfer function to confine pixel difference values. Finally, the image noise map is obtained through a threshold operation. Moreover, we have derived the training formula of these two weighting parameters and threshold of the mean aggregation, which can be learned iteratively. Results indicate that the integration of interval-valued fuzzy relations with the weighted mean aggregation algorithm will produce a more robust response in detecting the image noise.



1.2 Noisy Pixel Detection and Correction

1.2.1 Noise Model

In this thesis, we use two common types of impulse noise are the salt-and-pepper noise and the random-valued impulse noise. In the first case, the noisy pixels can take only the maximum and the minimum values. Let X_{ij} be the grayscale of the noisy image X at pixel (i, j) , Y_{ij} be the grayscale of a true image Y at pixel (i, j) and $[n_{\min}, n_{\max}]$ be the dynamic range of Y , the first noise model is given as Eq.

(1.1)

$$X_{ij} = \begin{cases} R_{ij} ; & \text{with probability } p \\ Y_{ij} ; & \text{with probability } 1-p \end{cases} \quad (1.1)$$

where $R_{ij} \in [n_{\min}; n_{\max}]$ and p is the noise ratio. In the second case, we set up five categories of impulse noisy pixel, pixels are randomly corrupted from any one of these categories, the second noise model is given as Eq.(1.2).

$$X_{ij} = \begin{cases} [0 ; 50] & \text{with probability } p_1 p \\ [0 ; 100] & \text{with probability } p_2 p \\ [0 ; 150] & \text{with probability } p_3 p \\ [0 ; 200] & \text{with probability } p_4 p \\ [0 ; 255] & \text{with probability } p_5 p \\ Y_{ij} & \text{with probability } 1-p \end{cases} \quad (1.2)$$

where p denotes the degree of random-valued impulse noise distortion, $\sum_i p_i = 1, i = 1, \dots, 5$. X_{ij} be the grayscale of the noisy image X at pixel (i, j) and Y_{ij} be the grayscale of a true image Y at pixel (i, j) .

1.2.2 Noisy Pixel Correction

In many practical situations, during image formation, storage, acquisition and transmission, many types of distortions degrade the quality of the digital images, images are corrupted by the impulsive noise of short duration and high energy, and common sources of impulsive noise include industrial machines, lightening, car starters, faulty or dusty insulation of high-voltage power lines and various unprotected electric switches [4-6].

Detection and correction of impulse noise from digital images have been of high research interest in the last years because the presence of noise in an image may be a drawback in any subsequent image processing tasks, such as pattern recognition, image segmentation or edge detection. Filtering the image to reduce the noise without degrading its quality, preserving edges, corners and other details is a major step in imaging systems such as image content retrieval, medical image processing, industrial visual inspection [7].

In order to recover the original pixel values, there are many works on the restoration of images corrupted by impulse noise. Among them, the most well-known to remove impulse noise is standard median filter. Median filter method sorts pixels in the working window, then picks the median value as the recovery pixel. It is the most popular nonlinear filter because of its high computational speed and good detection power. Over the last two decades, there is a significant improvement in the development of median filters. Weighted Median Filter (WMF) [2], Recursive Weighted Median Filter (RWMF) [8] are examples. The above methods are too much smoothing, which some edges and details of the original image are smeared by the filter. This undesired property is caused by unnecessary filtering of the noise-free pixel since the pixel that classifying as noise-free pixel should be left unchanged. To overcome this drawback, a switching mechanism has been introduced into the structure of the robust smoothing filters, [9, 10]. Such a switching filter first detects the pixel under consideration is affected by the noise process or not. If it is found to be contaminated, the noisy pixel will be replaced by the output of some robust filter. Otherwise, it remains unchanged. For example, Center Weighted Median Filter (CWMF) [3], Adaptive Median Filter (AMF) [1], and Switching Median Filter (SMF) [11] are popular switching filters in recent years. Results indicate that the switch-type methods have usually achieved good performance when the noise ratio is low.

1.3 Research Method

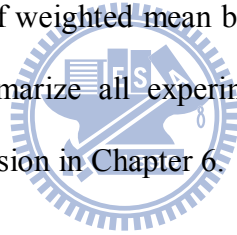
In this thesis, adopting the method proposed by Barrenechea et al. [12], the new noise detection method we proposed utilize generalized weighted mean aggregation algorithm to construct Interval-Valued Fuzzy Relations (IVFR). We can obtain two fuzzy relations by calculating the weighted mean difference of the central pixel and its 8-neighborhood pixels in a 3×3 sliding window across the image. To counter the over-weighting of a big difference term, we introduce a saturation threshold transfer function to limit the pixel difference values. If the intensity difference is smaller than the saturation threshold, its output value will be identical to the input. On the other hand, if the intensity difference is larger than the saturation threshold, then its output value will become gradually saturated. Finally, the image noise map is obtained through a thresholding operation.

Moreover, we have derived the iterative learning mechanism of these two weighting parameters of the mean aggregation and the threshold as well. In the parameters learning phase, we first use four typical natural 128×128 gray images with the same percentage of impulse noise rate by the computer program as the training images for the parametric learning. Then, we update the weighting parameters to decrease total error number of wrongly classified noise pixels and wrongly noise free pixels according to the four noisy training images by the steepest decent method [13, 14] casted in discrete formulation, namely in a spirit similar to perceptron learning. However, there could be no solution for perfect noise map. Therefore, we will exploit the pocket algorithm to our learning algorithm. First, we will record the parameter set in the pocket during the course of learning, and then pop out the best parameter set in our pocket as our best weighting parameters when we finished this

long enough learning epochs.

1.4 Thesis Outline

The thesis is organized as follows. The motivation of this study and the basic concept of image noise are introduced in Chapter 1. We introduce the noise detection method for grayscale images by applying the concept of interval-valued fuzzy relation proposed by Barrenechea et al [12] in Chapter 2. The basic concepts and technique concerning the noisy pixel detection and correction methods are described in Chapter 3. Then the parameter learning of weighted mean based noise detection are described in Chapter 4. Finally, we summarize all experimental results in Chapter 5, and conclude this thesis with a discussion in Chapter 6.



Chapter 2 Apply Interval-Valued Fuzzy Relation to Image Noise Detection

Noise detection from image data is of key importance in most image processing applications. In this chapter, we will introduce to construct interval-valued Fuzzy Relation (IVFR) [12] of an image by applying t -norm and t -conorm (also called s -norm) in the fuzzy theory for noisy pixel detection. For each 3×3 sliding window, we will first calculate the intensity differences between the central pixel and its eight neighbor pixels. To counter the over-weighting of a big difference term, we introduce a saturation threshold transfer function for pixel difference values. If the intensity difference is smaller than the saturation threshold, its output value will be identical to the input. On the other hand, if the intensity difference is larger than the saturation threshold, then its output value will become gradually saturated. By the above procedures, we can compute the upper and lower bound differences corresponding to each pixel of the image, which lead to interval-valued fuzzy relation of the image.

We refer to a method proposed by Barrenechea et al [12] of an image by applying t -norm and s -norm in the fuzzy theory to the central pixel and its eight neighbor pixels in a sliding window across the image. It can indicate that the noisy pixel should make it clear that the adjacent pixels having a big enough variation to this central pixel intensity. To measure this variation between the intensity of a pixel and the intensities of the neighboring pixels, we construct, by means of lower and upper constructors, the interval-valued fuzzy relation and its associated W-fuzzy relation. Fig. 2.1 demonstrates the steps of the application of interval-valued fuzzy relations in noise detection of images.

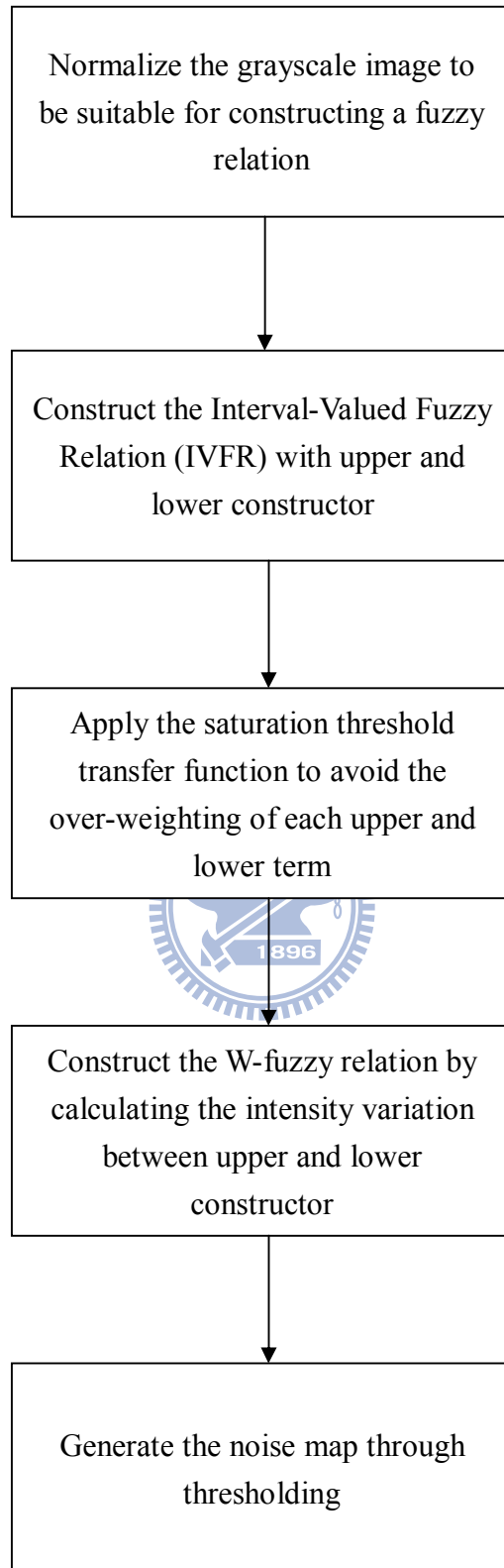


Fig. 2.1. The flowchart to obtain noise map by using interval-valued fuzzy relation.

2.1 Fuzzy Relation

Based on the fuzzy relation, we can define two fuzzy relations of an image, one is “upper” and the other is “lower” constructor to constitute the interval-valued fuzzy relation with the same dimensions of the images [12]. We must consider carefully that the gray-scale intensity of each pixel and its neighbor pixels contained in a fixed range of the testing image in the procedure. First, we focused on a grayscale image with size of $M \times N$, execute the normalization step as follows. We divide the grayscale maximal intensity “255,” so that the grayscale value of each pixel will be between interval of $[0, 1]$.

Next, we consider two finite universes $X = \{0, 1, \dots, M-1\}$ and $Y = \{0, 1, \dots, N-1\}$. Then, $R = \{(x, y), R(x, y) \mid (x, y) \in X \times Y\}$ is called a fuzzy relation from X to Y . Fuzzy relations are defined as an $M \times N$ matrices in the following expression (2.1) :

$$R = \begin{pmatrix} R(0,0) & \cdots & R(0,N-1) \\ R(1,0) & \cdots & R(1,N-1) \\ \vdots & \ddots & \vdots \\ R(M-1,0) & \cdots & R(M-1,N-1) \end{pmatrix}. \quad (2.1)$$

In addition, $F(X \times Y)$ represents the set of all fuzzy relations from X to Y [15], [16].

2.2 Lower and Upper Constructor

In this section, we define the expressions of the lower and upper bound respectively for a k -tuple pair (x_1, x_2, \dots, x_k) , where $x_1, x_2, \dots, x_k \in [0, 1]$

$$\underset{i=1}{\overset{k}{T}} x_i = T(\underset{i=1}{\overset{k}{T}} x_i, x_k) = T(x_1, x_2, \dots, x_k) \quad (2.2)$$

$$\underset{i=1}{\overset{k}{S}} x_i = S(\underset{i=1}{\overset{k}{S}} x_i, x_k) = S(x_1, x_2, \dots, x_k) \quad (2.3)$$

The above equations “ T ” and “ S ” stand for one kind of t -norm and s -norm operators introduced in fuzzy theory; we know that t -norm and s -norm are both an associative, commutative, increasing function and they have four basic types respectively. Here, we take the common “min” and “max” operators for examples. The operations of them are expressed as the following Eq. (2.4) and Eq. (2.5):

$$T_M(x, y) = \min(x, y) \quad x, y \in [0, 1] \quad (2.4)$$

$$S_M(x, y) = \max(x, y) \quad x, y \in [0, 1] \quad (2.5)$$

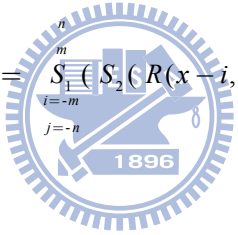
Let $R \in F(X \times Y)$ be a Fuzzy Relation (FR) from X to Y . Consider two t -norms T_1 and T_2 , and two values $m, n \in \mathbb{N}$ so that $m \leq \frac{M-1}{2}$, and $n \leq \frac{N-1}{2}$. The lower constructor associated with T_1 , T_2 , m , and n is defined in the following way:

$$L_{T_1, T_2}^{m, n} [R]: F(X \times Y) \rightarrow F(X \times Y) \quad (2.6)$$

$$L_{T_1, T_2}^{m, n} [R](x, y) = \underset{j=-n}{\overset{n}{T_1}} \left(\underset{i=-m}{T_2} (R(x-i, y-j), R(x, y)) \right) \quad (2.7)$$

In Eq. (2.7), all of $(x, y) \in X \times Y$, and the indices i, j take values such that $0 \leq x-i \leq M-1$ and $0 \leq y-j \leq N-1$. The values of m and n represent that the sliding window is a matrix of $(2m+1) \times (2n+1)$ dimension, which is centered at (x, y) . Similarly, we can define the upper bound of an interval by considering two s-norms S_1 and S_2 . Eq. (2.8) and Eq. (2.9) are the expression of upper constructor.

$$U_{S_1, S_2}^{m, n} [R]: F(X \times Y) \rightarrow F(X \times Y) \quad (2.8)$$

$$U_{S_1, S_2}^{m, n} [R](x, y) = \underset{j=-n}{\overset{n}{S_1}} \left(\underset{i=-m}{S_2} (R(x-i, y-j), R(x, y)) \right) \quad (2.9)$$


In Fig. 2.2, we illustrate how the upper constructor operation works with $m=n=1$, and $S_1=S_2=S_M$. For the element $(0,0)$ and $(x_1, y_1) \in X \times Y$ in the fuzzy relation, we have:

$$U_{S_M, S_M}^1 [R](0, 0) = \max(\max(0.74, 0.71), \max(0.69, 0.71), \max(0.72, 0.71), \max(0.71, 0.71,)) = 0.74$$

$$U_{S_M, S_M}^1 [R](x_1, y_1) = \max(\max(0.56, 0.49), \max(0.56, 0.49), \max(0.53, 0.49), \max(0.49, 0.49), \max(0.49, 0.49), \max(0.58, 0.49), \max(0.59, 0.49), \max(0.77, 0.49), \max(0.78, 0.49)) = 0.78$$

(0,0) 0.71	(0,1) 0.72			
(1,0) 0.69	(1,1) 0.74			
		$(x_1 - 1, y_1 - 1)$ 0.78	$(x_1 - 1, y_1)$ 0.77	$(x_1 - 1, y_1 + 1)$ 0.59
		$(x_1, y_1 - 1)$ 0.58	(x_1, y_1) 0.49	$(x_1, y_1 + 1)$ 0.49
		$(x_1 + 1, y_1 - 1)$ 0.53	$(x_1 + 1, y_1)$ 0.56	$(x_1 + 1, y_1 + 1)$ 0.56

(0,0) 0.74				
			(x_1, y_1) 0.78	

Fig. 2.2. Example of upper constructor.

2.3 Construction of Interval-Valued Fuzzy Relation

Suppose $R \in F(X \times Y)$, consider a lower constructor $L_{T_1, T_2}^{m, n}$ and a upper constructor $U_{S_1, S_2}^{m, n}$. Then the interval-valued fuzzy relation can be denoted by

$$R^{m, n}(x, y) = [L_{T_1, T_2}^{m, n}[R](x, y), U_{S_1, S_2}^{m, n}[R](x, y)] \in L([0, 1]), \quad (2.10)$$

where $(x, y) \in X \times Y$, and $L([0, 1])$ represents the set of all closed subintervals of $[0, 1]$. For simplicity, we denote $R^{m, n}$ as R^m if $m = n$.

2.4 W-Fuzzy Relations



After building interval-valued fuzzy relation from FR, we construct a new fuzzy relation $W[R^{m, n}]$, which is defined as follows:

$$W[R^{m, n}](x, y) = \overline{R^{m, n}}(x, y) - \underline{R^{m, n}}(x, y) = U_{S_1, S_2}^{m, n}[R](x, y) - L_{T_1, T_2}^{m, n}[R](x, y) \quad (2.11)$$

$$x \in X = \{0, 1, \dots, M-1\}, y \in Y = \{0, 1, \dots, N-1\}$$

When using upper and lower constructors, the length of the interval associated with a position indicates the intensity variation in its neighborhood. Then, in the construction of W-fuzzy relations, the length of an interval represents the membership degree of each element to the new FR [12].

Chapter 3 Some Noisy Pixel Detection and Correction Methods

During Image formation, storage or transmission, many types of distortions degrade the quality of digital images. In many practical situations, most of images are corrupted by the impulsive noise. Therefore, there are many noisy pixel detection and correction methods have been proposed, such as *center weighted median filters* (CWMF) [3], *peer group filter* (PGF) [17] or *switching median filter* (SMF) [11]. These efficient noise detector methods will be described in the following.

3.1 Peer Group Filter



The main target of the noise reduction algorithms of an image is to suppress noise while preserving image features like edges or texture. Vector Median Filter (VMF) is a popular filter but the main drawback is it fails to distinguish between the original uncorrupted pixels and pixels affected by the noise process. In order to alleviate the problems, the VMF-based filter such as *peer group filter* (PGF) [17] has been widely used. Essentially, the peer group of pixels in a given window represents the set of neighbor pixels that are similar enough to each pixel according to a particular measurement. In *peer group filter* (PGF) [17], the pixels are sorted in ascending order according to their distances to the central pixel. Then, the peer group center pixel $x_{(1)}$ is determined as the filtering window pixels that rank the lowest in the sorted sequence. If the distance between $x_{(1)}$ and the central pixel $x_{(c)}$ is exceeding

threshold d , then we will identify the central pixel as a noise candidate and replaced with the VMF output $x_{(1)}$. Otherwise, the central pixel is free of noise and remains unchanged. The range of threshold d is set to [40, 60]. Eq. (3.1) is the expression of PGF output.

$$\text{PGF}_{\text{out}} = \begin{cases} x_{(1)}, & \text{if } \|x_{(1)} - x_{(c)}\|_2 > d \\ x_{(c)}, & \text{if } \|x_{(1)} - x_{(c)}\|_2 \leq d \end{cases} \quad (3.1)$$

The following method is the improvement of PGF; it is a fast modification of the PGF where the central pixel is considered to be noise-free as soon as at least m pixels in the sliding window are decided to be similar enough, known as *fast peer group filter* (FPGF) [18]. In other words, if the central pixel $x_{(c)}$ has m neighbors, and the distance between $x_{(c)}$ and neighbors is not exceeding threshold d , then the central pixel $x_{(c)}$ is belong to the peer group $P(x_{(c)}, m, d)$. Equation is expressed as follows:

$$\text{FPGF}_{\text{out}} = \begin{cases} x_{(c)}, & \text{if } x_{(c)} \in P(x_{(c)}, m, d) \\ x_{(1)}, & \text{otherwise} \end{cases} \quad (3.2)$$

For the understanding of the algorithm mentioned above, a 3×3 windowed with the central pixel "0" is used as an example for illustrating FPGF process. Assume $d = 60$ and $m = 4$, as shown in Fig.3.1.

100	150	120
100	0	50
150	100	0

Fig. 3.1. Example for illustrating FPGF process.

Step1) Calculate the difference between the central pixel and adjacent eight neighboring pixels; i.e., $V = [100, 150, 120, 100, 50, 150, 100, 0]$.

Step2) Calculate the number of neighbors of central pixel, in this example, $m = 2$.

Step3) The number of neighbors of central pixel is smaller than the default value $m = 4$. Therefore, we regarded the central pixel as a noisy pixel.

3.2 Adaptive Peer Group Filter

When the concentration of the impulse noise is not high in an image, peer group filter with 3×3 window is good enough to detect the noise pixel. However, when the noise intensity is in high level or the noisy pixels are blobs, we may need to use

another mechanism to detect the noise because the number of noisy pixels in working window is too many, they may be misjudged as free of noise. Instead, we may use a larger size sliding window in order to detect noisy pixels in blobs or in high intensity of noise. There is a main drawback of using larger size of the sliding window to detect noisy pixel. However, the large size of working window can detect a group of noisy pixels well, but the boundary in the image is usually become blurred. On the contrary, small size of the sliding window cannot detect a group of noisy pixels, but the details can be preserved well. Fortunately, Adaptive Peer Group Filter [19] can prevent this drawback. First, we use PGF with 5×5 sliding window because 3×3 cannot detect noisy pixels in blobs well, when the center pixel is detected as a noisy pixel (see Fig. 3.2, Fig. 3.3), we will use the median of 3×3 sliding window to correct, this step can reduce image blurring. After correction, we will use 5×5 working window to detect the center pixel again, if it is still noisy pixel, we increase the size of the working window. Repeat the above steps until the central pixel is corrected.

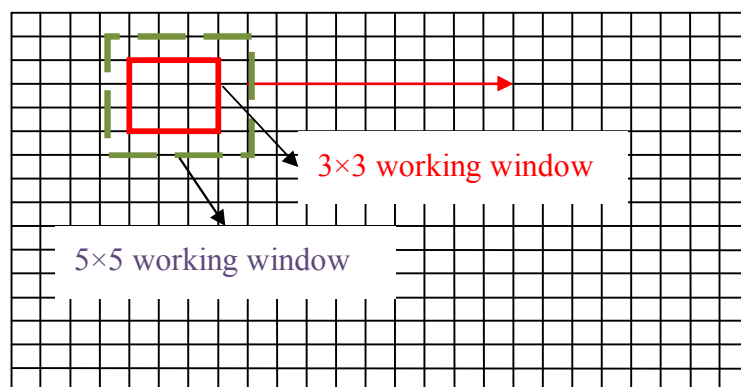


Fig. 3.2. The working window size of $n \times n$, ($n=3, 5, 7, \dots$).

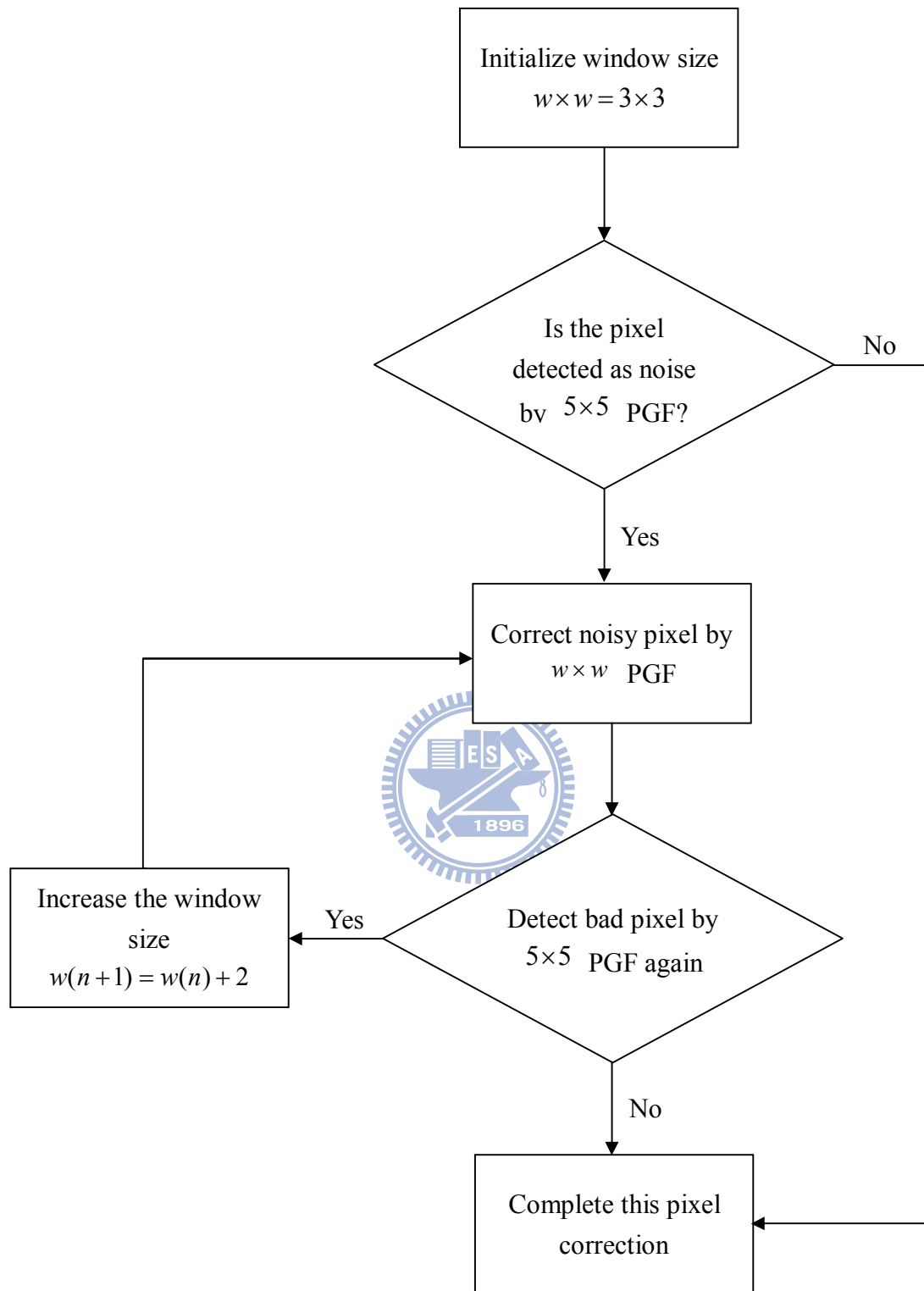


Fig. 3.3. Block diagram of the Adaptive Peer Group Filter.

3.3 Fast Similarity-Based Impulsive Noise Removal Vector Filter

In this section, we introduce a new filter that the computational complexity is lower than the *vector median filter* (VMF) and the simulation results indicate that the new filter outperforms the VMF. According to the filter introduced by Smolka et al. in [20–23], the *fast similarity-based impulsive noise removal vector filter* (FSVF) is described below.


We assume that W is a window of finite size $n+1$ (filter length), and the noisy pixels inside the window W will be denoted as $F_j, j = 0, 1, \dots, n$, where n is the number of neighbors of the center pixel F_0 . The distance between two pixels F_i, F_j is denoted as $\rho(F_i, F_j)$. Let us define a similarity function $\mu: [0; \infty) \rightarrow \mathbb{R}$ which is non-ascending and convex in $[0; \infty)$ and satisfies $\mu(0)=1$, and $\lim_{x \rightarrow \infty} \mu(x)=0$. The similarity between two pixels of the same gray level should be 1, and the similarity between pixels with maximal gray level “255” and minimal gray level “0” should be very close to 0. Here, we use the similarity function defined by the following equation

$$\mu\{F_i, F_j\} = \mu(\|F_i - F_j\|) \quad (3.3)$$

where $\|\cdot\|$ denotes the specific vector norm (typically the L_1 or L_2 vector norms can easily satisfy the required conditions). We additionally define the cumulated sum M of similarities between a given pixel and the adjacent pixels belonging to window W , where equation is expressed as follows:

$$M_0 = \sum_{j=1}^n \mu(F_0, F_j), \quad M_k = \sum_{\substack{j=1 \\ j \neq k}}^n \mu(F_k, F_j) \quad (3.4)$$

We introduce M_0 for the central pixel and M_k for the neighbors of F_0 (see Fig. 3.4). From the above equation, we found that the similarity between F_k and F_0 are not take into account, which privileges the central pixel. Hence, when it is really noisy, the reference pixel F_0 is replaced by one of its neighbors if $M_0 < M_k$, $k = 1, \dots, n$, preserving the details of original image. If this is the case then, F_0 is replaced by that F_{k^*} for which $k^* = \arg \max M_k$, $k = 1, \dots, n$. The central pixel output is expressed as follows:



$$output = \begin{cases} F_{k^*}, & M_0 < M_k \\ F_0, & M_0 \geq M_k \end{cases} \quad (3.5)$$

F_1	F_2	F_3
F_8	F_0	F_4
F_7	F_6	F_5

(a)

F_1	F_2	F_3
F_8		F_4
F_7	F_6	F_5

(b)

Fig. 3.4. (a) First the cumulative similarity value M_0 between the central pixel F_0 and its neighbors is calculated. (b) Then pixel F_0 is rejected from the filter window and the cumulative similarity values M_k , $k=1, \dots, n$, of the pixels F_1, \dots, F_n are determined [21].

There are several convex functions fulfilling the above conditions, but the best results were achieved for the simplest similarity function $\mu_7(x)$. Applying the linear similarity function μ_7 we obtain

$$\mu_7(F_i, F_j) = \begin{cases} 1 - \frac{\rho(F_i, F_j)}{h}, & \text{for } \rho(F_i, F_j) < h \\ 0, & \text{otherwise} \end{cases} \quad (3.6)$$

where $h \in (0, \infty)$, the parameter h influences the intensity of the filtration process. $\rho(F_i, F_j)$ represents the distance between two pixels F_i, F_j . This function allows constructing a fast noise reduction algorithm.

3.4 Center Weighted Median Filter

The median-based noise detection strategies work well for fixed-valued impulse noise; they have been recognized as a useful image enhancement technique. However, it requires some caution because median filtering tends to blur image details such as corners and lines while reducing noise. In response to these difficulties, several

variations of median filters have been introduced. Here, we focus on an extension of weighted median filters called *center weighted median filter* (CWMF) [3], this filter gives more weight to the central pixel.

First, we defined a filtering window W surrounding the current or original pixel $W = \{(s, t) | -h \leq s \leq h, -h \leq t \leq h\}$. The output of CWM filters is expressed as follows [3]:

$$Y_{ij}^w = \text{median}(X_{ij}^w) \quad (3.7)$$

where

$$X_{ij}^w = \{X_{i-s, j-t}, w \diamond X_{ij} \mid (s, t) \in W, (s, t) \neq (0, 0)\} \quad (3.8)$$

In the above equation, w denotes the center weight and the operator \diamond represents the repetition operation. For the current pixel X_{ij} , we first define differences

$$d_k = |Y_{ij}^w - X_{ij}| = |Y_{ij}^{2k+1} - X_{ij}| \quad (3.9)$$

where $k = 0, 1, \dots, L-1$. These differences provide the information about the likelihood of corruption for the central pixel, the decision making mechanism is achieved by using a set of thresholds T_k ($k = 0, 1, \dots, L-1$), where $T_{k-1} > T_k$ for $k = 0, 1, \dots, L-1$, which is shown as follows:

$$T_k = s \times MAD + \delta_k \quad (3.10)$$

where the median of the absolute deviations from the median is defined as

$$MAD = \text{median} \left\{ |X_{i-s, j-t} - Y_{ij}^1| \mid (s, t) \in W \right\} \quad (3.11)$$

from the simulations conducted on a broad variety of images, the selection satisfying $[\delta_0, \delta_1, \delta_2, \delta_3] = [40, 25, 10, 5]$ yields good results in filtering random-valued impulse noise, the setting of $[\delta_0, \delta_1, \delta_2, \delta_3] = [55, 40, 25, 15]$ performs well in removing fixed-valued impulse noise, it is also observed that good result could be obtained by using $0 \leq s \leq 0.6$ in both types of impulse noise. Specifically, if any of the inequalities $d_k > T_k$ are true, then the central pixel X_{ij} is regarded as an impulse noise. Otherwise, it assumes the current pixel as noise-free; the proposed filter can be defined as follows:

$$\hat{X}_{ij} = \begin{cases} Y_{ij}^1, & \text{if } \exists k, d_k > T_k \\ X_{ij}, & \text{otherwise} \end{cases} \quad (3.12)$$

where \hat{X}_{ij} represents the final estimate of current pixel X_{ij} . Fig. 3.5 shows the block diagram of center weighted median filter.

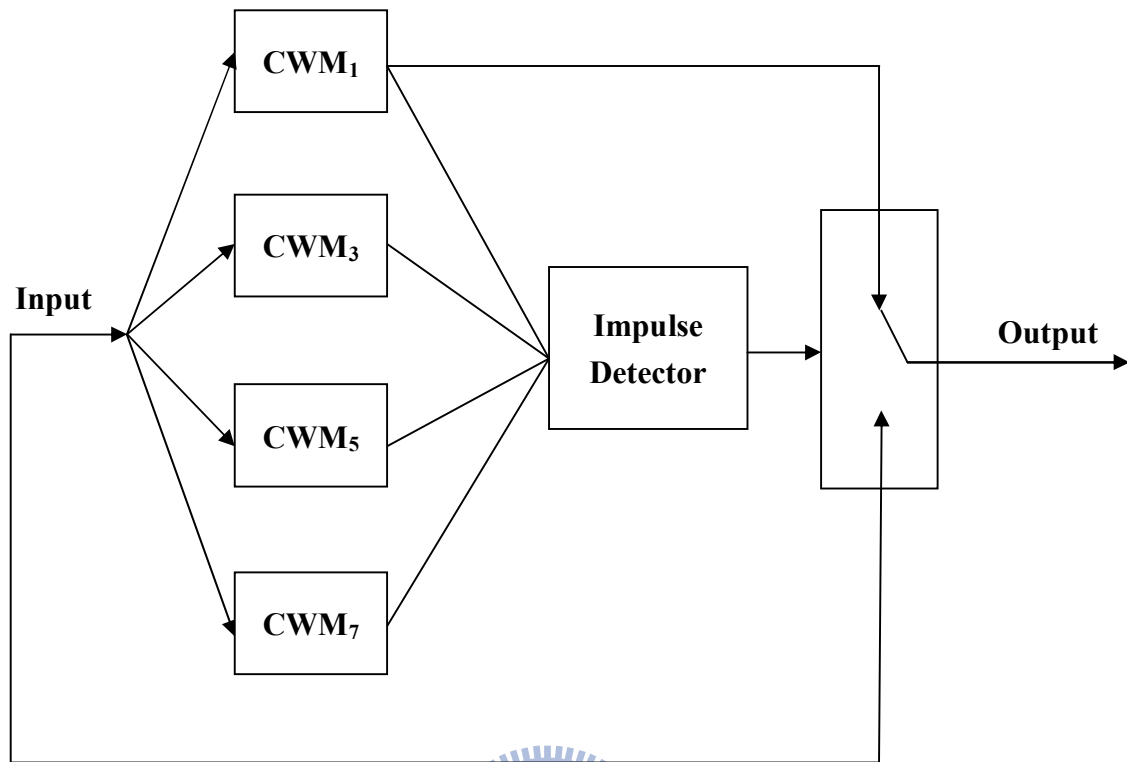


Fig. 3.5. Block diagram of the Center Weighted Median Filter.

3.5 Switching Median Filter

In this section, we introduce a new impulse noise detector based on *boundary discriminative noise detection* (BDND) algorithm [11]. Like BDND, it also consists two iterations and all pixels in the image are examined. In the first stage, we use the local histogram to determine the decision boundaries, the central pixel is examined from its 21×21 neighborhood, only when a pixel is judged as noise candidate, it will be piped into the second validation stage. If both stages are classified into “Noise” class, it will be considered to be a “true” impulse noise. The criterion can be summarized as follows:

$$class(x_{i,j}) = \begin{cases} \text{Pepper Noise,} & x_{i,j} \leq \min(b_1, m) \\ \text{Uncorrupted,} & \text{otherwise} \\ \text{Salt Noise,} & x_{i,j} > \max(b_2, 255 - m) \end{cases} \quad (3.13)$$

where b_1, b_2 are the two decision boundaries and $x_{i,j}$ is the intensity of the current pixel, parameter m represents the interval which lie in the two ends of the intensity range. The following steps show how to obtain the decision boundaries by using the BDND algorithm.

Step1) Impose a 21×21 window centers on $x_{i,j}$

Step2) Sort the pixels lie in the window region according to the ascending order v_0 and find the median med .

Step3) Compute the intensity difference of v_0 and obtain the difference vector v_D .

Step4) Find the maximum intensity difference in the intervals $[0, med]$ and $(med, 255]$, set these two pixels' intensities as the decision boundaries b_1 and b_2 respectively.

After obtaining two decision boundaries, the second validation stage of BDND reduces the window's dimension to 3×3 , it is obviously that the order of magnitude lacks statistical significance seriously. In order to make the decisions in both stages more robust and reliable, we modified the second stage, the basic philosophy is to find a way to accurately quantify the intensity differences between the central pixel and its neighbors in four directions (see Fig. 3.6).

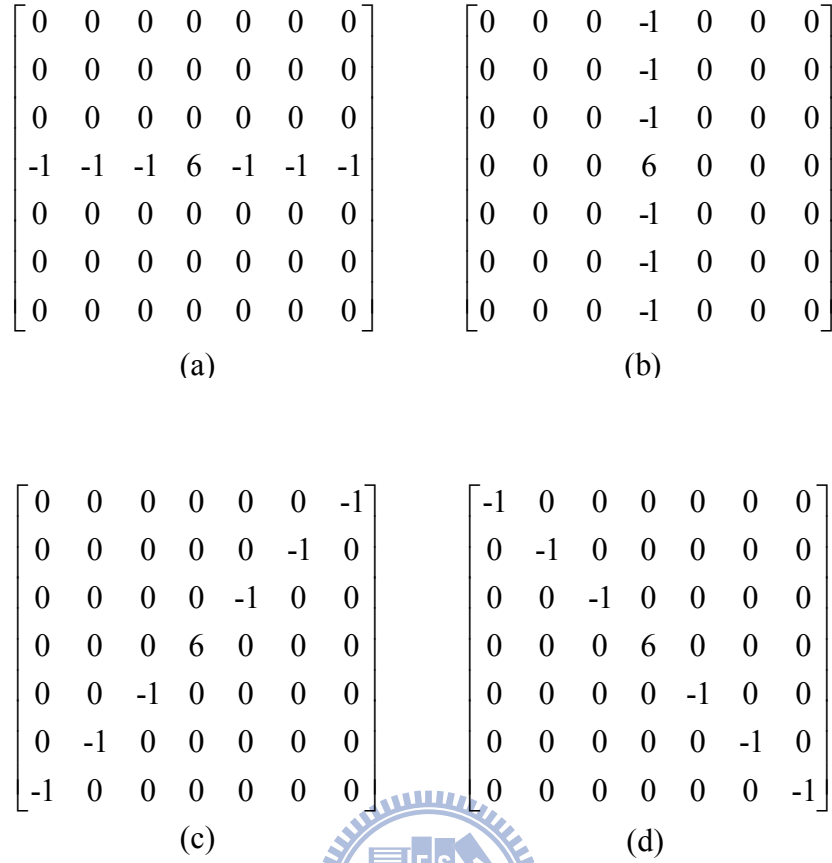


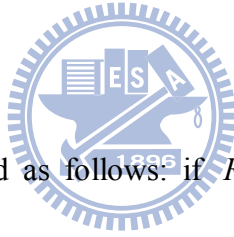
Fig. 3.6. Four directional convolutional kernels.

The dimension of the neighborhood of each pixel is 7×7 and the directions we care about are $0, \pi/4, \pi/2, 3\pi/4$. Then by convolving the group of kernels with the noise image, each pixel has four convolutional results. The computation strategy can be formulated as follows:

$$V_{ij}^{(k)} = \left| I \otimes \text{Ker}^{(k)} \right|_{ij}, \quad k = 1, \dots, 4 \quad (3.14)$$

where I is the noisy image, \otimes denotes the convolution operator, $\text{Ker}^{(k)}$ is the k^{th} kernel we designed and $V_{ij}^{(k)}$ is the k^{th} absolute convolutionary value for pixel I_{ij} . In [24], only the minimum of the four descriptors is used, the main disadvantage of this strategy is only comparing the minimum with a given threshold to decide whether the current pixel is corrupted. If the noise density is high, this rule will completely fail. Therefore, we utilize both the minimum and maximum of aforementioned four descriptors simultaneously. These two values can be obtained according to Eq.(3.15).

$$\begin{cases} R_{ij}^{\min} = \min \{V_{ij}^{(1)}, V_{ij}^{(2)}, V_{ij}^{(3)}, V_{ij}^{(4)}\} \\ R_{ij}^{\max} = \max \{V_{ij}^{(1)}, V_{ij}^{(2)}, V_{ij}^{(3)}, V_{ij}^{(4)}\} \end{cases} \quad (3.15)$$



The classification rule is defined as follows: if $R_{ij}^{\min} > T_1$ or $R_{ij}^{\max} - R_{ij}^{\min} > T_2$, then pixel I_{ij} is regarded as noisy pixel. Otherwise I_{ij} is not corrupted. In this criterion, the two thresholds T_1 and T_2 are respectively set to 5 and 1.

Chapter 4 Parameter Learning of Weighted Mean Based Noise Pixel Detection

In this chapter, we will propose a new impulse noise detection method through executing the weighted-mean aggregation to calculate the intensity difference between a central pixel and its eight neighborhood pixels in a 3×3 window across the image. Furthermore, to counter the over-weighting of a big difference term, we apply a saturation threshold transfer function to pixel difference values. Then, we obtain the noise map through a threshold operation .

Moreover, we have derived the training formula of these weighting parameters of the mean aggregation, which can be learned iteratively. In the parameters learning procedure, we use four natural 128×128 gray images adding the same percentage of impulse noise rate as shown in Fig. 4.1 by the computer program as training input images for the parametric learning. Fig. 4.2 shows the noise ground truths of four 10% impulse noise corrupted gray-scale natural images. Then, by the steepest decent method [13, 14] caste in discrete formulation, we update the weighting parameters to decrease total error number of wrongly classified noise and noise free pixels according to the four noisy training images. Namely this learning formula is aggregation weights in a spirit similar to perceptron learning.



Fig. 4.1. Four 10% impulse noise corrupted gray-scale natural images of size 128×128 .

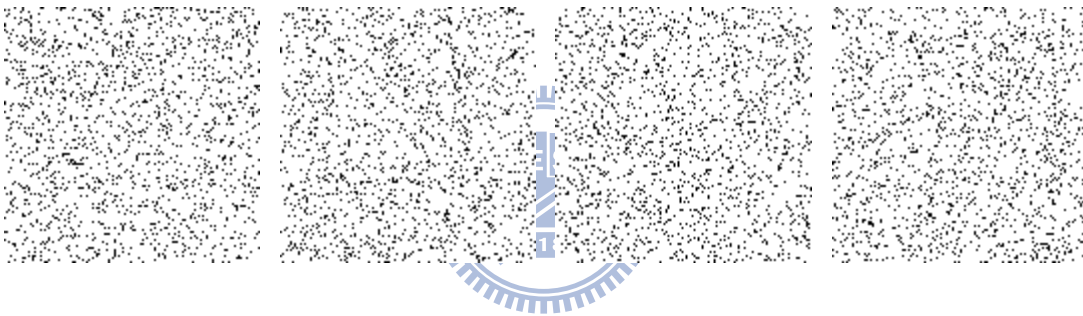


Fig. 4.2. Noise ground truths of four 10% impulse noise corrupted gray-scale natural images.

4.1 Weighted Mean Based Interval-Valued Fuzzy Relation

For the method proposed by Barrenechea et al [12], there are four selections for the s -norm and t -norm operators respectively to construct the interval-valued fuzzy relations. It is difficult to decide which combination is the most suitable to fit the image noise detection. Besides, they are nonlinear functions and hard to derive a

formula involving the “max” and “min” logical operators. In order to avoid these disadvantages, we will introduce the weighted-mean aggregation to generalize the s -norm and t -norm formula in a continuous setting. To this end, the s -type α_s and t -type α_t with $\alpha_s \in [0.5, 1]$ and $\alpha_t \in [0, 0.5]$, parameters in the weighted-mean aggregation are introduced to replace the s -norm and t -norm operators in constructing interval-valued fuzzy relations for noise detection.

4.2 *The Weighted Mean to Calculate the Difference of Neighbor Pixels of Images*



In this section, we first utilize the two operating parameters “ α_s ” and “ α_t ” to determine the intensity of s -type and t -type weighting aggregations and calculate the weighted difference for the pixel values of an image. Then, we apply a saturation threshold transfer function to pixel difference values in order to avoid the over-weighting of a big difference term. Finally, we can obtain the noise pixel map through a suitable threshold.

For an $M \times N$ image I , we first compute the s -type and t -type weighted difference value of the grayscale intensity between the central pixel $I(m, n)$ and its eight neighbor pixels in a 3×3 window as shown in Fig. 4.3.

1	2	3
8	$I(m, n)$	4
7	6	5

Fig. 4.3. The diagram of a 3×3 window centered at pixel $I(m, n)$ and its eight neighborhood pixels.

$$\begin{cases} y_{si}(I(m, n)) = \alpha_s a_i(I(m, n)) + (1 - \alpha_s) b_i(I(m, n)) \\ y_{ti}(I(m, n)) = \alpha_t a_i(I(m, n)) + (1 - \alpha_t) b_i(I(m, n)) \end{cases} \quad (4.1)$$

where $\alpha_s \in [0.5, 1]$; $\alpha_t \in [0, 0.5]$; $i = 1, 2, \dots, 8$; $m = 1, 2, \dots, M$; $n = 1, 2, \dots, N$. The subscript i represents the index of the eight neighbor pixels, $a_i(I(m, n))$ and $b_i(I(m, n))$ represent the larger and smaller grayscale value respectively between the central pixel $I(m, n)$ and one of its eight neighbor pixels. In order to counter the over-weighting of a big difference term, we apply a saturation threshold transfer function to pixel difference values. If the intensity difference is smaller than the saturation threshold (T_s), its output value will kept identical to the input as far as possible. On the other hand, if the intensity difference is larger than the saturation threshold, then its output value will become gradually saturated. To this end, the

saturation threshold transfer function could be defined as Sigmoid function as shown in Fig. 4.4.

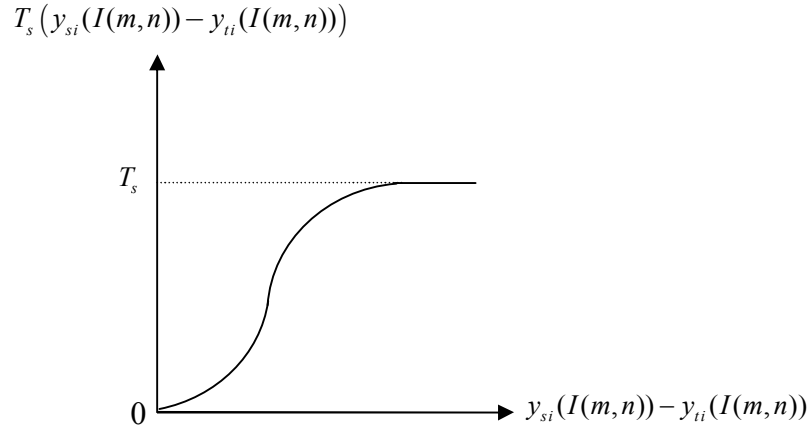


Fig. 4.4. Sigmoid function for saturation thresholding.

where $T_s(\cdot)$ represents the output value which pass through the saturation threshold (see Eq. (4.7)). Then, we compute the average of these output value as follows:

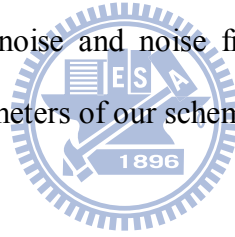
$$y_w(m,n) = \frac{\sum_{i=1}^8 T_s(y_{si}(I(m,n)) - y_{ti}(I(m,n)))}{8} \quad (4.2)$$

Finally, we determine whether the central pixel $I(m,n)$ belongs to noisy pixel or not according to the comparison of the $y_w(m,n)$ and the threshold T as given by the Eq. (4.3).

$$I(m, n) = \begin{cases} \text{noise,} & \text{if } y_w(m, n) \geq T \\ \text{noise free,} & \text{otherwise} \end{cases} \quad (4.3)$$

4.3 *Operating Parameter Learning Mechanism*

Before processing the parameter learning mechanism, we have to give a set of four initial operating parameters (α_s , α_t , T and T_s) which can be learned iteratively. One of the advantages of our learning mechanism is that these parameters are learned automatically that lead to beat noise and noise free pixel detection accuracy. The restrictions in setting initial parameters of our scheme are given by



- (1) $\alpha_s > \alpha_t$
- (2) $\alpha_s = [0.5, 1]$; $\alpha_t = [0, 0.5]$
- (3) $T_s \in \mathbb{R}^+$
- (4) $T \in \mathbb{R}^+$

4.3.1 Learning Rule of Mean Weighting Parameters

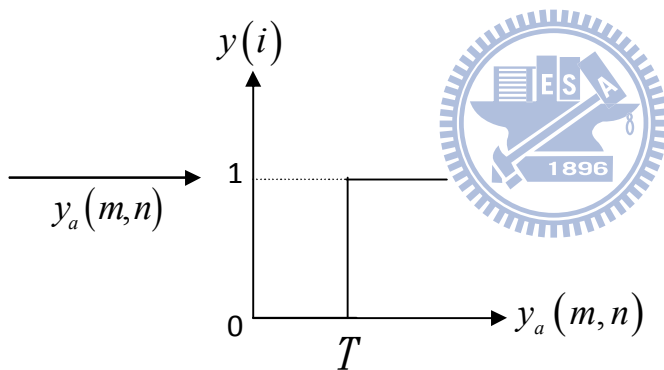
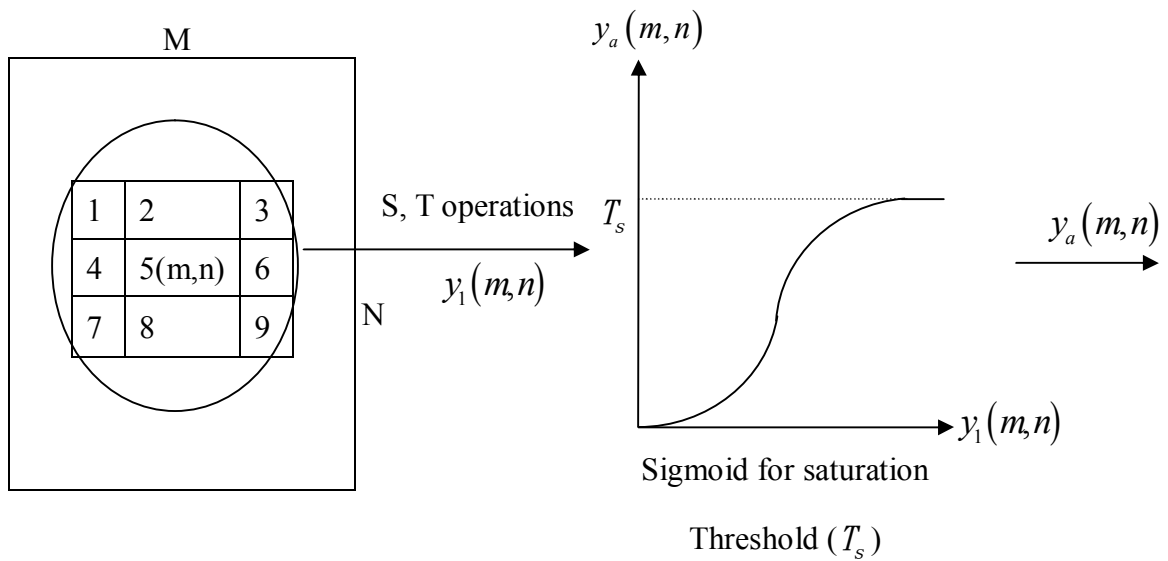


Fig. 4.5. Block diagram of applying W-fuzzy relations to noise pixel detection.

Assume $a(i) < b(i)$, perform the weighted mean aggregation calculation between a 3×3 window central pixel and its eight neighbor pixels, we have

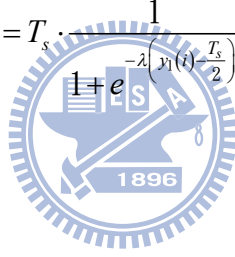
$$y_s(i) = \alpha_s b(i) + (1 - \alpha_s) a(i), \quad 0.5 \leq \alpha_s \leq 1. \quad (4.4)$$

$$y_t(i) = \alpha_t b(i) + (1 - \alpha_t) a(i), \quad 0 < \alpha_t \leq 0.5. \quad (4.5)$$

$$y_1(i) = y_s(i) - y_t(i) = (\alpha_s - \alpha_t)(b(i) - a(i)) \quad i = 1, 2, \dots, 8. \quad (4.6)$$

The central pixel must have a big enough variation in intensity near its neighbor if it is a noisy pixel. In order to counter the over-weighting of a big difference term, we construct the sigmoid activation function for the purpose of saturation thresholding.

$$y_a(i) = a(y_1(i)) = T_s \cdot \frac{1}{1 + e^{-\lambda \left(y_1(i) - \frac{T_s}{2} \right)}} \quad i = 1, 2, \dots, 8. \quad (4.7)$$



where $\lambda > 0$ in Eq. (4.7) determines the steepness of the continuous function $a(y_1(i))$ near $y_1(i) = 0$. Finally, we compute the average value of eight $y_a(i)$ which is the mean difference of eight neighboring pixel with respect to the central pixel. If this value exceeds the predefined threshold, we will regard the current pixel as a noise pixel. Otherwise, the central pixel is free of noise, as given by

$$y(i) = \begin{cases} 1 \text{ (noise),} & \text{if } \frac{\sum_{i=1}^8 y_a(i)}{8} - T \geq 0 \\ 0 \text{ (noise free),} & \text{if } \frac{\sum_{i=1}^8 y_a(i)}{8} - T < 0 \end{cases} \quad (4.8)$$

With the above formulation in mind, we are now ready to derive the training formula for the parameter set α_s , α_t , T_s and T . For a given $M \times N$ image, we first compute the total error $E(m, n)$ in order to obtain the smallest number of wrongly classified noise and noise-free pixels as follows.

$$E(m, n) = \frac{1}{2} \sum_{i=1}^M \sum_{i=1}^N (d(m, n) - y(m, n))^2, \quad (4.9)$$

where $d(m, n)$ and $y(m, n)$ indicates the ground truth noise map of the input image and the output noise map of the input image at (m, n) , respectively. Then, the derivative of $E(m, n)$ respect to α_s is given by

$$\frac{\partial E(m, n)}{\partial \alpha_s} = \frac{\partial E(m, n)}{\partial y(m, n)} \frac{\partial y(m, n)}{\partial y_a(m, n)} \frac{\partial y_a(m, n)}{\partial y_1(m, n)} \frac{\partial y_1(m, n)}{\partial \alpha_s} \quad (4.10)$$

$1 \leq m \leq M; 1 \leq n \leq N.$

Since,

$$\frac{\partial E(m, n)}{\partial y(m, n)} = -(d(m, n) - y(m, n)), \quad (4.11)$$

$$\frac{\partial y(m, n)}{\partial y_a(m, n)} = 1, \quad (4.12)$$

$$\frac{\partial y_a(m,n)}{\partial y_1(m,n)} = -T_s \cdot \frac{1}{\left(1 + e^{-\lambda\left(y_1(m,n) - \frac{T_s}{2}\right)}\right)^2} \cdot e^{-\lambda\left(y_1(m,n) - \frac{T_s}{2}\right)} \cdot (-\lambda), \quad (4.13)$$

$$\frac{\partial y_1(m,n)}{\partial \alpha_s} = \frac{\sum_{i=1}^8 (b(i) - a(i))}{8}. \quad (4.14)$$

Using the above equations, then a perceptron learning of the weighting parameter α_s iteratively can be given by

$$\alpha_{s_new} = \alpha_{s_old} - \eta_s \frac{\partial E(m,n)}{\partial \alpha_s} \quad (4.15)$$

where η_s is the learning constant of α_s . Similarly, a perceptron learning of the weighting parameters α_t iteratively can be given by

$$\frac{\partial E(m,n)}{\partial \alpha_t} = \frac{\partial E(m,n)}{\partial y(m,n)} \frac{\partial y(m,n)}{\partial y_a(m,n)} \frac{\partial y_a(m,n)}{\partial y_1(m,n)} \frac{\partial y_1(m,n)}{\partial \alpha_t} \quad (4.16)$$

$1 \leq m \leq M; 1 \leq n \leq N.$

Since,

$$\frac{\partial E(m,n)}{\partial y(m,n)} = -(d(m,n) - y(m,n)), \quad (4.17)$$

$$\frac{\partial y(m,n)}{\partial y_a(m,n)} = 1, \quad (4.18)$$

$$\frac{\partial y_a(m,n)}{\partial y_1(m,n)} = -T_s \cdot \frac{1}{\left(1 + e^{-\lambda\left(y_1(m,n) - \frac{T_s}{2}\right)}\right)^2} \cdot e^{-\lambda\left(y_1(m,n) - \frac{T_s}{2}\right)} \cdot (-\lambda), \quad (4.19)$$

$$\frac{\partial y_1(m,n)}{\partial \alpha_t} = -\frac{\sum_{i=1}^8 (b(i) - a(i))}{8}. \quad (4.20)$$

Then a perceptron learning of the weighting parameter α_t iteratively can be given by

$$\alpha_{t_new} = \alpha_{t_old} - \eta_t \frac{\partial E(m,n)}{\partial \alpha_t} \quad (4.21)$$

where η_t is the learning constant of α_t . Follow similar line of reasoning, the perceptron learning formula for parameters T_s and T , can be given by

$$\frac{\partial E(m,n)}{\partial T_s} = \frac{\partial E(m,n)}{\partial y(m,n)} \frac{\partial y(m,n)}{\partial y_a(m,n)} \frac{\partial y_a(m,n)}{\partial T_s} \quad (4.22)$$

$1 \leq m \leq M; 1 \leq n \leq N.$

Since,

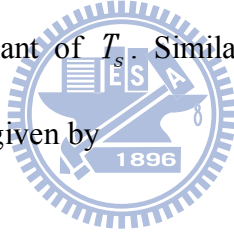
$$\frac{\partial E(m,n)}{\partial y(m,n)} = -(d(m,n) - y(m,n)), \quad (4.23)$$

$$\frac{\partial y(m, n)}{\partial y_a(m, n)} = 1, \quad (4.24)$$

$$\frac{\partial y_a(m, n)}{\partial T_s} = \frac{1}{1 + e^{-\lambda \left(y_1(m, n) - \frac{T_s}{2} \right)}} \cdot \left(T_s \cdot \frac{1}{\left(1 + e^{-\lambda \left(y_1(m, n) - \frac{T_s}{2} \right)} \right)^2} \right) \cdot e^{-\lambda \left(y_1(m, n) - \frac{T_s}{2} \right)} \cdot \left(\frac{\lambda}{2} \right). \quad (4.25)$$

$$T_{s_new} = T_{s_old} - \eta_{T_s} \frac{\partial E(m, n)}{\partial T_s} \quad (4.26)$$

where η_{T_s} is the learning constant of T_s . Similarly, a perceptron learning of the parameter T iteratively can be given by



$$\frac{\partial E(m, n)}{\partial T} = \frac{\partial E(m, n)}{\partial y(m, n)} \frac{\partial y(m, n)}{\partial T} \quad (4.27)$$

$$1 \leq m \leq M; 1 \leq n \leq N.$$

where

$$\frac{\partial E(m, n)}{\partial y(m, n)} = -(d(m, n) - y(m, n)), \quad (4.28)$$

$$\frac{\partial y(m, n)}{\partial T} = -1. \quad (4.29)$$

$$T_{new} = T_{old} - \eta_T \frac{\partial E(m, n)}{\partial T} \quad (4.30)$$

where η_T is the learning constant of T . For a set of natural images to train the best weighting parameters α_s 、 α_t 、 T_s and T , there may be no solution for perfect noise map. Therefore, we will use the pocket algorithm to our learning phase, i.e., we will save the best α_s 、 α_t 、 T_s and T in the pocket during the course of learning, and pop out the best parameters in our pockets as our best weighting parameters when we finished a long enough learning epoch.

4.3.2 Proposed Training Method

In this section, we propose two training stages for the purpose to maintain image sharpness. In the first stage of our training method, we will choose four typical grayscale natural images with adding the same percentage of impulse noise rate to each natural images as the input training images, then we raster-scanningly train every pixel of training images. And at the end of each row, under the currently learned parameter values of α_s 、 α_t 、 T_s and T , we test the noise and noise free pixel detection accuracy by summing of all four input training images. To enhance the noise detection accuracy, we weighted the noise pixel misclassification by a multiplier constant where value is larger than 1. From our experiment of obtaining good Peak Signal Noise Ratio (PSNR), the multiplier constant is dependent on the impulse noise rate of adding on the image. We have experienced for a good PSNR that multiplier constant equal to 10 for impulse noise rate smaller than 10%, and equal to 100 otherwise. It is to be noted that our proposed learning scheme usually cannot obtain a

perfect noise maps of training images. i.e., the number of wrongly classified noise or noise free pixels is zero for all the training images. Therefore, we will exploit the pocket algorithm to our learning phase. In the first epoch, we will select the best parameters with the smallest number of wrongly classified pixels. We store this number of smallest misclassified pixels and its associated parameter set as our best solution in this learning epoch. Then, we use the best parameters from the first learning epoch as the initial parameters of the second learning epoch. After a long enough learning epochs, we use the best parameters as the initial value of our second training stage.

Our learning algorithm will identify the central pixel of 3×3 window as a noisy pixel if there is a big enough difference with its eight neighbor pixels. Unfortunately, an edge point could be prone to be wrongly classified as a noise pixel for it could produce a big difference with neighbor pixels due to edge effect. To alleviate this shortage, we propose the second stage to retrain the images with edge pixel being identified differently. Similarly as the first stage, we add the edge detection process to our training method in the second stage. When we start training these four input images in the second stage, we first use the best parameter set from the first stage to repair the input training images. Then, we apply Canny edge detector to help us find the edge pixels from these repaired images. Finally, when the current training pixel is an edge pixel, we increase the threshold value by a factor of “1.1” to reduce the probability of edge pixel to be detected as a noisy pixel. By this way, we can achieved good performance by combining these two training stages.

Chapter 5 Experimental Results

In our experiments, we focus on two common types of impulse noise, one is salt-and-pepper noise and the other is random-valued impulse noise. These two types of noise model are described in Section 1.2.1. For the measurement of the restoration quality, we employ the *peak signal-to-noise ratio* (PSNR) performance metric, which is based on the *root-mean squared error* (RMSE). The expression of RMSE and PSNR are defined as:

$$RMSE = \sqrt{\frac{1}{N \times M \times Q} \sum_{i=1}^N \sum_{j=1}^M \sum_{q=1}^Q (x^q(i, j) - o^q(i, j))^2} \quad (5.1)$$

$$PSNR = 20 \times \log \left(\frac{255}{RMSE} \right) \quad (5.2)$$

where M , N are the image dimensions, Q is the number of channels of the image ($Q=1$ for grayscale image), and $o^q(i, j)$ and $x^q(i, j)$ denote the q -th component of the original image vector and the filtered image, at pixel position (i, j) , respectively. For the evaluation of the detail preservation capabilities of the proposed filtering design, the *mean absolute error* (MAE) has been used as

$$MAE = \frac{\sum_{i=1}^N \sum_{j=1}^M \sum_{q=1}^Q |x^q(i, j) - o^q(i, j)|}{N \times M \times Q} \quad (5.3)$$

In order to explicitly see the advantages and disadvantages of each method, we will normalize the parameters which are RMSE and MAE. These parameters are normalized between 0 and 1; the larger value for the parameters indicates better performances. These two normalized $RMSE_{\text{nor}}$ and MAE_{nor} is expressed as follows:

$$RMSE_{\text{nor}} = \frac{\max(RMSE) - RMSE(i)}{\max(RMSE) - \min(RMSE)} \quad (5.4)$$

$$MAE_{\text{nor}} = \frac{\max(MAE) - MAE(i)}{\max(MAE) - \min(MAE)} \quad (5.5)$$



At last, we compare our s, t aggregation approach to several noise filters such as Adaptive Peer Group (APG) [19], Center Weighted Median Filter (CWMF) [3], Peer Group Filter (PGF) [17], Fast Similarity-based impulsive noise removal Vector Filter (FSVF) [20-23], Switching Median Filter (SMF) [11]. Our first method applies s -norm and t -norm operators respectively to construct the interval-valued fuzzy relations by extensive combinatory trials, without s, t learning mechanism and saturation threshold process, we called it “ST” method. If “ST” method with saturation threshold process, we called it “ST with saturation” method. In order to make our experimental results more representative, we take average of 100 testing images with the same percentage of impulse noise rate to all four training images.

5.1 Results of Salt and Pepper Noise Correction

In this experiments, we have added the salt-and-pepper noise, as shown in Fig. 5.1, to four gray-scale images as our training input images. Based on our proposed weighted mean based interval-valued fuzzy relations for noise detection of noisy images, we have obtained good results by setting the initial values of $\alpha_s = 0.75$, $\alpha_t = 0.25$, learning constants $\eta_s = \eta_t = \eta_T = \eta_{T_s} = 0.0035$, noisy pixel threshold $T = 6$ and saturation threshold $T_s = 18$ in the first training stage. After 9 learning epochs in the first training stage, we select the best parameters as the initial values of our second training stage. In the second stage of our training method, we can obtain the best parameter set after 30 learning epochs. If the percentage of noise ratio is less than 20%, we will correct it by using alpha-trimmed mean filter using ranking central three pixels, when the pixel is regarded as a noisy pixel. Otherwise, we will correct it by using median filter when it is detected as a noisy pixel.

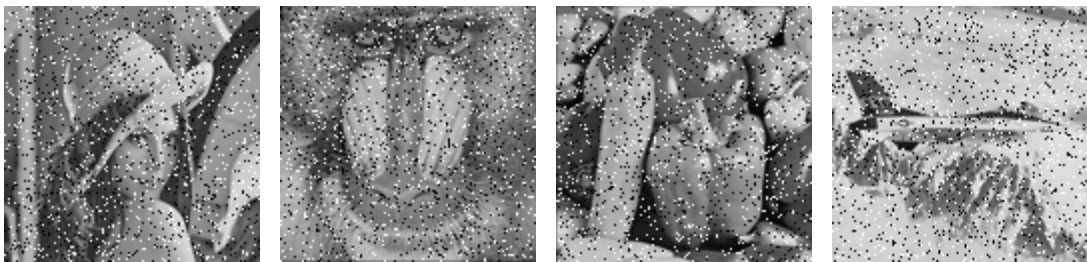


Fig. 5.1. Four 10% salt and pepper noise training images with size of 128×128 .

According to TABLE I, although our method is not the best method, the performance of our method is still above average. When the concentration of the salt and pepper noise is increased in an image, our method is better than the other methods gradually. Fig. 5.2, Fig. 5.3 and Fig. 5.4 show the correction results by different filters of noisy Boat image, with 20%, 40% and 60% salt and pepper noise, respectively.



(a)



(b)



(c)



(d)



(e)



(f)





(g)



(h)



(i)



(j)

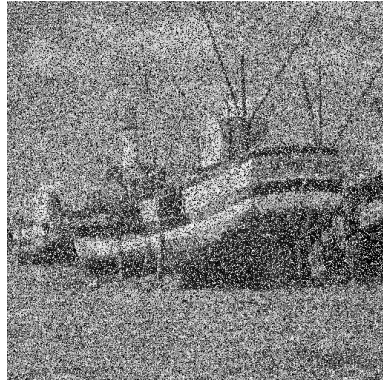


(k)

Fig. 5.2. Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 20% salt and pepper noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method.



(a)



(b)



(c)



(d)



(e)



(f)



(g)



(h)





(i)



(j)

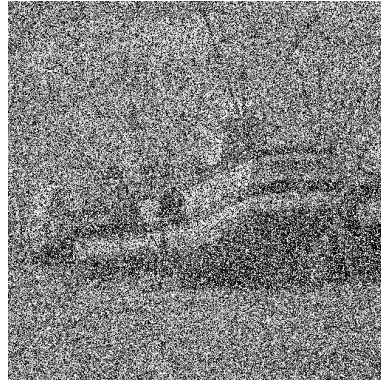


(k)

Fig. 5.3. Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 40% salt and pepper noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method.



(a)



(b)



(c)



(d)



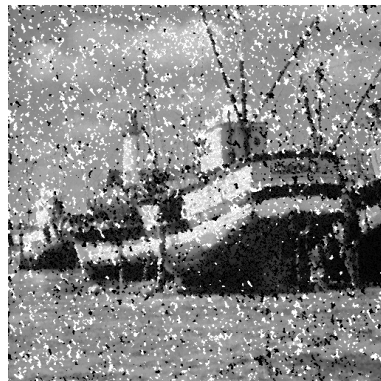
(e)



(f)



(g)



(h)





(i)



(j)



(k)

Fig. 5.4. Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 60% salt and pepper noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method.

TABLE I
THE NOISE REMOVAL RESULTS BY DIFFERENT FILTERS

(a) Corrupted Boat image with 10% salt and pepper noise

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	7.746	1.729	30.349 ₉	0.000	0.000	0.000 ₉
CWMF	4.995	0.825	34.161₂	0.620	0.820	1.440₂
PGF	6.005	1.014	32.561 ₅	0.393	0.649	1.042 ₄
FSVF	5.267	0.882	33.700₃	0.559	0.768	1.327₃
SMF	3.313	0.626	37.729₁	1.000	1.000	2.000₁
ST _{TAE}	6.642	1.417	31.685 ₈	0.249	0.283	0.532 ₈
ST with saturation	6.201	1.107	32.282 ₇	0.349	0.564	0.913 ₇
Our learning ST method (two stage)	5.911	1.065	32.699 ₄	0.414	0.602	1.016 ₅
Our learning ST method (one stage)	6.166	1.098	32.332 ₆	0.357	0.572	0.929 ₆

(b) Corrupted Boat image with 20% salt and pepper noise

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	10.236	3.147	27.929 ₉	0.000	0.000	0.000 ₉
CWMF	8.023	1.795	30.046₃	0.442	0.752	1.194₃
PGF	8.200	2.098	29.854 ₅	0.407	0.584	0.991 ₄
FSVF	7.835	1.835	30.252₂	0.480	0.730	1.210₂
SMF	5.230	1.349	33.762₁	1.000	1.000	2.000₁
ST _{TAE}	9.059	2.820	28.990 ₈	0.235	0.182	0.417 ₈
ST with saturation	8.353	2.192	29.695 ₆	0.376	0.532	0.908 ₆
Our learning ST method (two stage)	8.129	2.250	29.930 ₄	0.421	0.499	0.920 ₅
Our learning ST method (one stage)	8.399	2.223	29.647 ₇	0.367	0.514	0.881 ₇

(c) Corrupted Boat image with 40% salt and pepper noise

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	15.335	6.518	24.418 ₇	0.546	0.151	0.697 ₆
CWMF	21.269	6.070	21.577 ₉	0.000	0.272	0.272 ₉
PGF	13.724	4.747	25.382₂	0.694	0.628	1.322₂
FSVF	18.866	5.788	22.618 ₈	0.221	0.348	0.569 ₇
SMF	10.399	3.367	27.792₁	1.000	1.000	2.000₁
ST _{TAE}	14.935	6.611	24.648 ₅	0.583	0.126	0.709 ₅
ST with saturation	14.441	5.222	24.939 ₄	0.628	0.501	1.129₃
Our learning ST method (two stage)	14.347	5.277	24.996₃	0.637	0.486	1.123 ₄
Our learning ST method (one stage)	15.161	7.080	24.517 ₆	0.562	0.000	0.562 ₈

(d) Corrupted Boat image with 60% salt and pepper noise

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	26.985	14.667	19.508 ₇	0.816	0.493	1.309 ₇
CWMF	48.974	21.161	14.332 ₉	0.000	0.011	0.011 ₉
PGF	26.259	10.449	19.746₂	0.843	0.805	1.648₂
FSVF	45.988	21.310	14.878 ₈	0.111	0.000	0.111 ₈
SMF	22.023	7.823	21.274₁	1.000	1.000	2.000₁
ST _{TAE}	26.895	12.443	19.538 ₆	0.819	0.657	1.476 ₆
ST with saturation	26.865	11.292	19.547 ₅	0.820	0.743	1.563₃
Our learning ST method (two stage)	26.840	12.294	19.556₃	0.821	0.669	1.490 ₄
Our learning ST method (one stage)	26.843	12.309	19.555 ₄	0.821	0.667	1.489 ₅

5.2 Results of Random-Valued Impulse Noise Correction

In this experiments, we have added the random-valued impulse noise, as shown in Fig. 5.3, to four gray-scale images as our training input images. Similarly, we have obtained the good performances by setting the initial values of $\alpha_s = 0.75$, $\alpha_t = 0.25$, learning constants $\eta_s = \eta_t = \eta_T = \eta_{T_s} = 0.0035$, noisy pixel threshold $T = 6$ and saturation threshold $T_s = 18$ in the first training stage. After 9 learning epochs in the first training stage, we select the best parameters as the initial values of our second training stage. In the second stage of our training method, we can obtain the best parameter set after 30 learning epochs. If the percentage of noise ratio is less than 20%, we will correct it by using alpha-trimmed mean filter using ranking central three pixels, when the pixel is regarded as a noisy pixel. Otherwise, we will correct it by using median filter when it is detected as a noisy pixel.

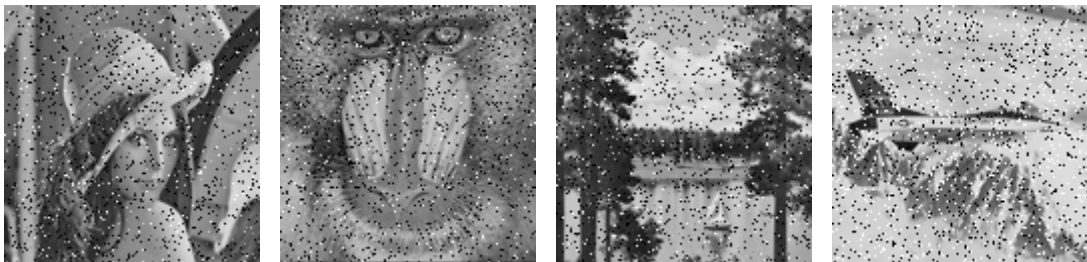
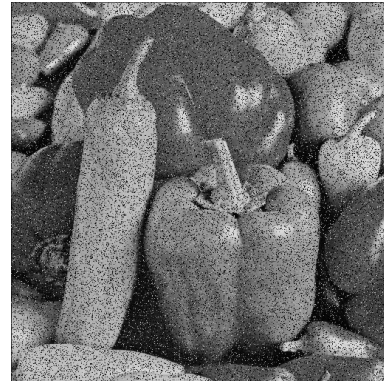


Fig. 5.5. Four 10% random-valued impulse noise training images with size of 128×128 .

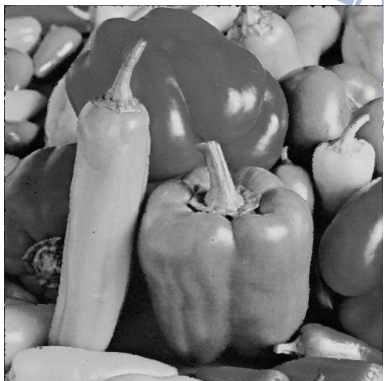
According to TABLE II, the performance of our proposed method is above average no matter what the sample corruption probability is. Fig. 5.5 and Fig. 5.6 show the correction results by different filters of noisy Pepper image, with 20% and 40% random-valued impulse noise, respectively. Fig. 5.7 shows the correction results by different filters of noisy Boat image with 60% random-valued impulse noise.



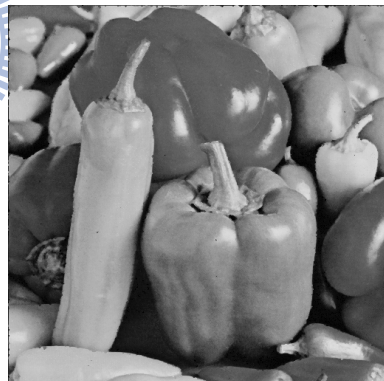
(a)



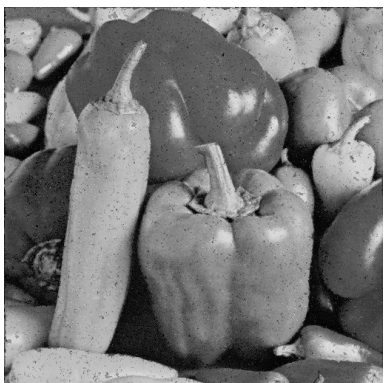
(b)



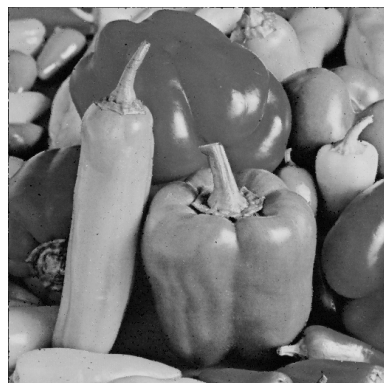
(c)



(d)



(e)



(f)

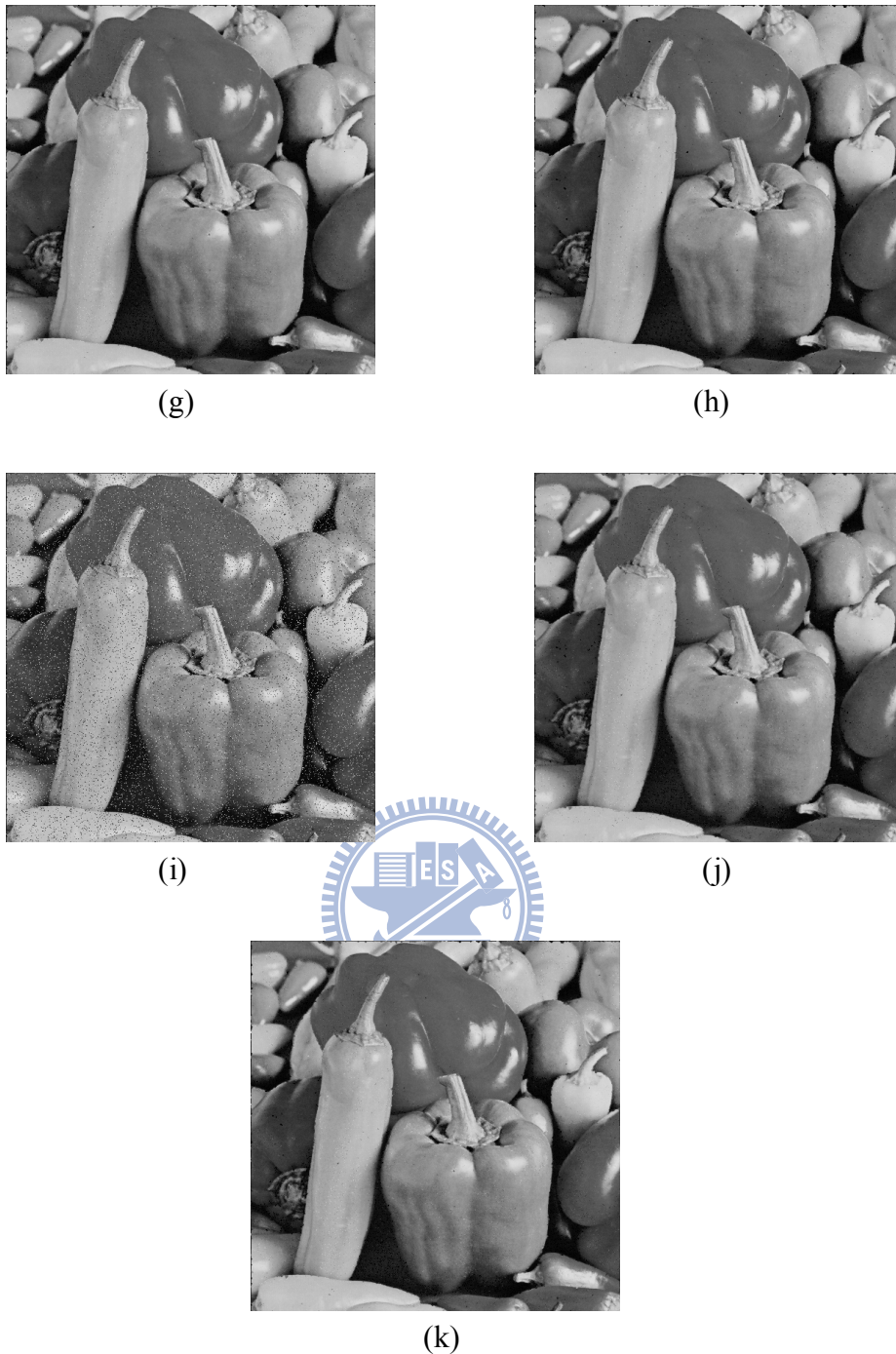
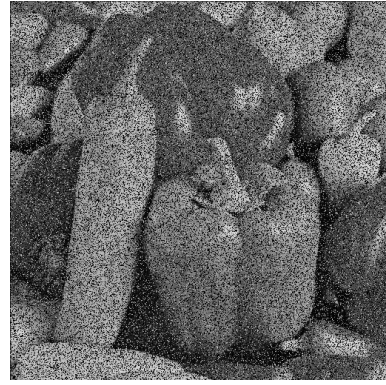


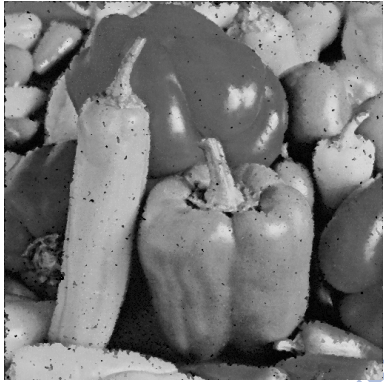
Fig. 5.6. Noisy pixel correction results of Pepper image filtered by different filters. (a) Original image. (b) Corrupted image with 20% random-valued impulse noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method.



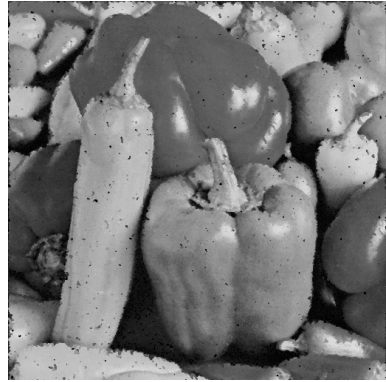
(a)



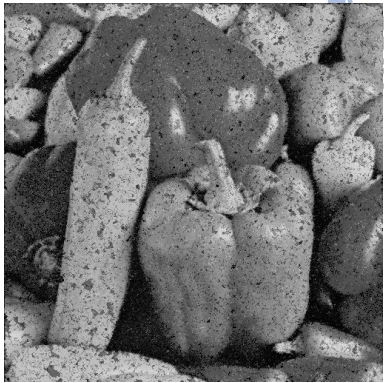
(b)



(c)



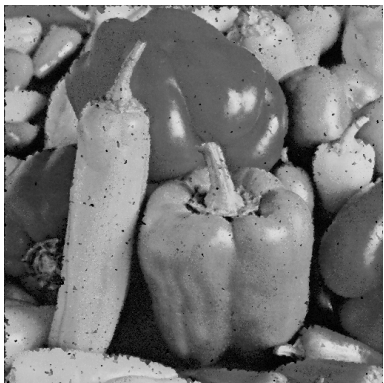
(d)



(e)



(f)

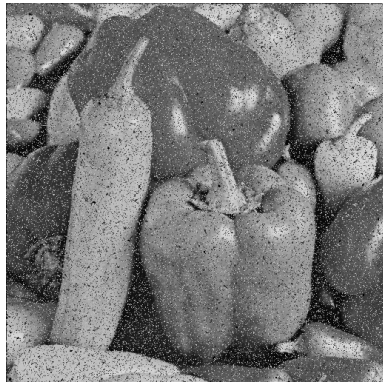


(g)



(h)





(i)



(j)

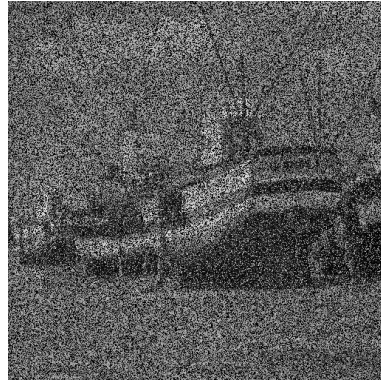


(k)

Fig. 5.7. Noisy pixel correction results of Pepper image filtered by different filters. (a) Original image. (b) Corrupted image with 40% random-valued impulse noise. (c) –(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method.



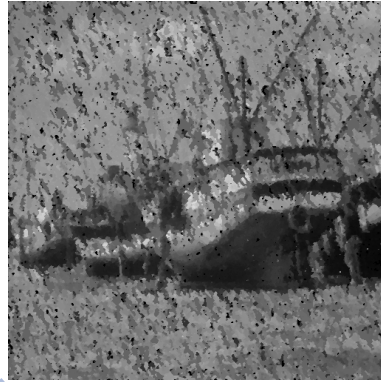
(a)



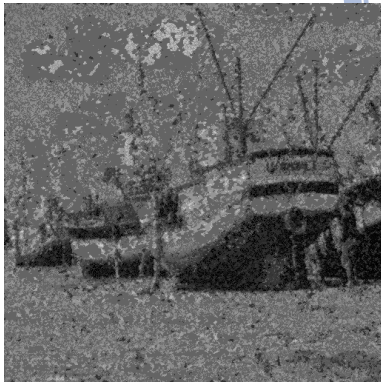
(b)



(c)



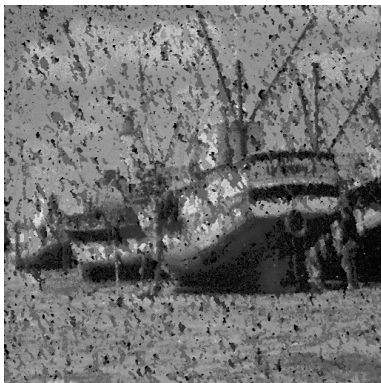
(d)



(e)



(f)



(g)

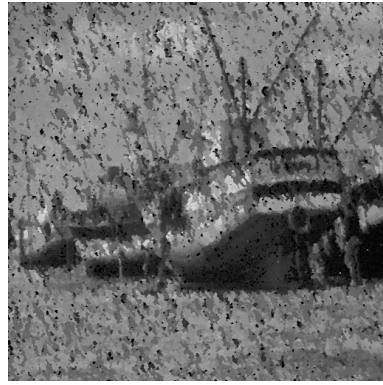


(h)





(i)



(j)



(k)

Fig. 5.8. Noisy pixel correction results of Boat image filtered by different filters. (a) Original image. (b) Corrupted image with 60% random-valued impulse noise. (c)–(i) are filtering results. Image filtering results filtered by (c) our proposed filter with two stages. (d) our proposed filter with one stage. (e) Adaptive Peer Group (APG). (f) Center Weighted Median Filter (CWMF). (g) Peer Group Filter (PGF). (h) Fast Similarity-based impulsive noise removal Vector Filter (FSVF). (i) Switching Median Filter (SMF). (j) “ST” method. (k) “ST with saturation” method.

TABLE II
THE NOISE REMOVAL RESULTS BY DIFFERENT FILTERS

(a) Corrupted Pepper image with 10% random-valued impulse noise.

Filter	RMSE	MAE	PSNR	RMSE- normalized	MAE- normalized	SUM
APG	7.900	1.687	30.179 ₈	0.714	0.627	1.341 ₈
CWMF	4.054	0.663	35.976₁	1.000	1.000	2.000₁
PGF	4.726	0.806	34.644 ₆	0.950	0.948	1.898 ₆
FSVF	4.329	0.726	35.407₂	0.980	0.977	1.957₂
SMF	17.490	3.408	23.275 ₉	0.000	0.000	0.000 ₉
ST _{TAE}	5.199	0.967	33.814 ₇	0.915	0.889	1.804 ₇
ST with saturation	4.608	0.808	34.862 ₅	0.959	0.947	1.906 ₅
Our learning ST method (two stage)	4.422	0.792	35.221₃	0.973	0.953	1.926₃
Our learning ST method (one stage)	4.499	0.828	35.072 ₄	0.967	0.940	1.907 ₄

(b) Corrupted Pepper image with 20% random-valued impulse noise.

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	13.040	4.107	25.826 ₈	0.637	0.517	1.154 ₈
CWMF	6.319	1.501	32.119₁	1.000	1.000	2.000₁
PGF	7.359	1.817	30.796 ₅	0.944	0.941	1.885 ₅
FSVF	6.859	1.570	31.408₂	0.971	0.987	1.958₂
SMF	24.826	6.894	20.233 ₉	0.000	0.000	0.000 ₉
ST _{TAE}	7.874	2.123	30.207 ₇	0.916	0.885	1.801 ₇
ST with saturation	7.375	1.820	30.777 ₆	0.943	0.941	1.884 ₆
Our learning ST method (two stage)	7.056	1.729	31.161₃	0.960	0.958	1.918₃
Our learning ST method (one stage)	7.189	1.849	30.999 ₄	0.953	0.936	1.889 ₄

(c) Corrupted Pepper image with 40% random-valued impulse noise.

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	28.480	13.748	19.040 ₈	0.347	0.108	0.455 ₈
CWMF	13.614	4.578	25.452₁	1.000	0.976	1.976₁
PGF	14.835	5.094	24.707 ₆	0.946	0.927	1.873 ₆
FSVF	15.138	4.327	24.532₃	0.933	1.000	1.933₂
SMF	36.392	14.894	16.911 ₉	0.000	0.000	0.000 ₉
ST _{TAE}	15.269	5.786	24.456 ₇	0.927	0.862	1.789 ₇
ST with saturation	14.750	4.981	24.756 ₄	0.950	0.938	1.888 ₄
Our learning ST method (two stage)	14.695	4.910	24.789₂	0.953	0.945	1.898₃
Our learning ST method (one stage)	14.781	5.119	24.738 ₅	0.949	0.925	1.874 ₅

(d) Corrupted Boat image with 60% random-valued impulse noise.

Filter	RMSE	MAE	PSNR	RMSE-normalized	MAE-normalized	SUM
APG	45.189	33.151	15.030 ₈	0.259	0.000	0.259 ₉
CWMF	37.469	18.380	16.658 ₂	0.987	1.000	1.987 ₁
PGF	37.329	20.461	16.690 ₁	1.000	0.859	1.859 ₂
FSVF	43.470	21.081	15.316 ₇	0.397	0.817	1.214 ₇
SMF	47.939	26.934	14.517 ₉	0.000	0.421	0.421 ₈
ST _{TAE}	38.733	23.599	16.370 ₅	0.868	0.647	1.515 ₅
ST with saturation	37.978	21.201	16.540 ₄	0.939	0.809	1.748 ₄
Our learning ST method (two stage)	37.924	20.893	16.553 ₃	0.944	0.830	1.774 ₃
Our learning ST method (one stage)	38.933	24.292	16.325 ₆	0.849	0.600	1.449 ₆

Chapter 6 Conclusions

In this thesis, we integrate the weighted mean aggregation and Interval-Valued Fuzzy Relation (IVFR) for detecting noise of an image. For each 3×3 sliding window, the upper and lower weighted mean aggregations of central pixel and its eight neighbor pixels can be calculated, which constitute the interval-valued fuzzy relations. To counter the over-weighting of a big difference term, we introduce a saturation threshold transfer function for pixel difference values. Moreover, the difference between the upper and lower aggregations reflects the degree of intensity variation between central pixel and its eight neighbor pixels. That is, we will identify the central pixel as a noise candidate if it is larger than threshold. Along this line of reasoning, the learning formula of the weighting parameters are derived to decrease the noise detection error of an image. However, there could be no solution for perfect noise map. Therefore, we have exploited the pocket algorithm to our learning algorithm.

The effectiveness of our noise detection scheme is verified by various impulse noise images. Finally, our designed model is applied to noise detection of natural image. Results indicate that the method we proposed is proven to be more superior than the other noise detection and correction algorithms.

References

- [1] Ho-Ming Lin and Alan, "Median filters with Adaptive Length," *IEEE transactions of the circuits and systems*, vol. 35, no. 6, June 1988.
- [2] O. Yli-Harja, J. Astola, and Y. Neuvo, "Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation," *IEEE Trans. Signal Processing*, vol. 39, pp. 395–410, Feb. 1991.
- [3] S.-J. Ko and Y. H. Lee, "Center weighted median filters and their applications to image enhancement," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 984–993, Sept. 1991.
- [4] C Boncelet. Image noise models. In: A Bovik, editor. *Handbook of image and video processing*. New York: Academic Press, 2000.
- [5] Lopez DB, Mendoza FH, Ramirez JM. Noise in color digital images. In: *Proceedings of the midwest symposium on circuits and systems*, pp. 403–6, Aug. 1999.
- [6] J. Zheng, K. P. Valavanis, and J. M. Gauch, "Noise removal from color images," *J. intell. Robot. Syst.*, vol. 7, no. 3, pp. 257-285, June 1993.
- [7] Plataniotis KN, Venetsanopoulos AN. In *Color image processing and applications*. Berlin: Springer; 2000.
- [8] G. Arce and J. Paredes, "Recursive weighted median filters admitting negative weights and their optimization," *IEEE Tr: on Signal Proc.*, vol. 48, nr. 3, March 2000.
- [9] R. Lukac, V. Fischer, G. Motyl, and M. Drutarovsky, "Adaptive video filtering framework," *Int. J. Imag. Syst. Technol.*, vol. 14, no. 6, pp. 223–237, Dec. 2004.
- [10] R. Lukac, "Adaptive vector median filtering," *Pattern Recognit. Lett.*, vol. 24, pp. 1889–1899, Aug. 2003.

- [11] P.-E. Ng and K.-K. Ma, "A switching median filter with boundary discriminative noise detection for extremely corrupted images," *IEEE Trans. Image Process.*, vol. 15, no. 6, pp. 1506–1516, Jun. 2006.
- [12] E. Barrenechea, H. Bustince, and C. Lopez-Molina, "Construction of interval-valued fuzzy relations with application to the generation of fuzzy edge images," *IEEE Trans. Fuzzy System.*, vol. 19, no. 5, Oct 2011.
- [13] J. H. Hong, S. Campbell, and P. Yeh, "Optical pattern classifier with Perceptron learning," *Applied Optics*, vol. 29, no. 20, pp. 3019–3025, 1990.
- [14] M. E. Yuksel and E. Besdok, "A simple neuro-fuzzy impulse detector for efficient blur reduction of impulse noise removal operators for digital images," *IEEE Trans. Fuzzy Syst.*, vol.12, no. 6, pp. 854–865, Dec. 2004.
- [15] H. Bustince and P. Burillo, "Structures on intuitionistic fuzzy relations," *Fuzzy Sets Syst.*, vol. 78, pp. 293–303, 1996.
- [16] H. Bustince and P. Burillo, "Mathematical analysis of interval-valued fuzzy relations: Application to approximate reasoning," *Fuzzy Sets Syst.*, vol. 113, pp. 205–219, 2000.
- [17] Y. Deng, C. Kenney, M. S. Moore, and B. S. Manjunath, "Peer group filtering and perceptual color image quantization," in *Proc. IEEE Int. Symp. Circuits and Systems*, vol. 4, 1999, pp. 21–24.
- [18] B. Smolka and A. Chydzinski, "Fast detection and impulsive noise removal in color images," *Real-Time Imag.*, vol. 11, no. 5–6, pp. 389–402, Oct.–Dec. 2005.
- [19] Y. K. Peng, "Non-Uniformity and Bad Pixel Correction for NIR image," *Master Thesis*, Elect. and Con. Eng. Dept., Chiao Tung Univ., Taiwan, 2012.
- [20] B. Smolka, K. N. Plataniotis, R. Lukac and A. N. Venetsanopoulos, 'Similarity based impulsive noise removal in color images'. *Proc. IEEE Int. Conf. Image Process.*, Barcelona, Spain, 2003, pp. 105–108

- [21] B. Smolka, R. Lukac, A. Chydzinski, K. N. Plataniotis, and W. Wojciechowski, “Fast adaptive similarity based impulsive noise reduction filter,” *Real-Time Imag.*, vol. 9, no. 4, pp. 261–276, Aug. 2003.
- [22] B. Smolka, A. Chydzinski, K. N. Plataniotis, and A. N. Venetsanopoulos, “New filtering technique for the impulsive noise reduction in color images,” *Mathematical Problems in Engineering*, vol. 2004, no. 1, pp. 79-91.
- [23] B. Smolka, A. Chydzinski, K. Wojciechowski, K. N. Plataniotis, and A. N. Venetsanopoulos, “Self-adaptive algorithm for impulsive noise reduction in color images,” *Pattern Recognit.*, vol. 35, no. 8, pp. 1771–1784, 2002.
- [24] S. Q. Zhang and M. A. Karim, “A new impulse detector for switching median filters,” *IEEE Signal Process. Lett.*, vol. 9, no. 11, pp. 360–363, Nov. 2002.

