

# Chapter 1 Introduction

## 1.1 Motivation

Edge detection plays an important role in image processing and computer vision. High-level image processing tasks such as image segmentation, object recognition, tracking, stereo analysis, and image coding depend on the quality of the edge detection procedure. The performance of these tasks is therefore tremendously affected by the effectiveness of edge detection. In grayscale edge detection, the Canny edge detector [1] is well-known for its high reliability and sensitivity. It is largely due to its optimal weights in computing pixel's difference, non-maximal suppression technique and thresholding with hysteresis make the output edge map showing well-connected edges. For an image, as the color information lost during grayscale conversion, so that when the edges of the object boundary although have different hues but only little change in the grayscale intensity, edges still most likely cannot be detected successfully. In addition, edge detection sometimes will also enhance the difficulty of its implementation in face of some lower-contrast images.

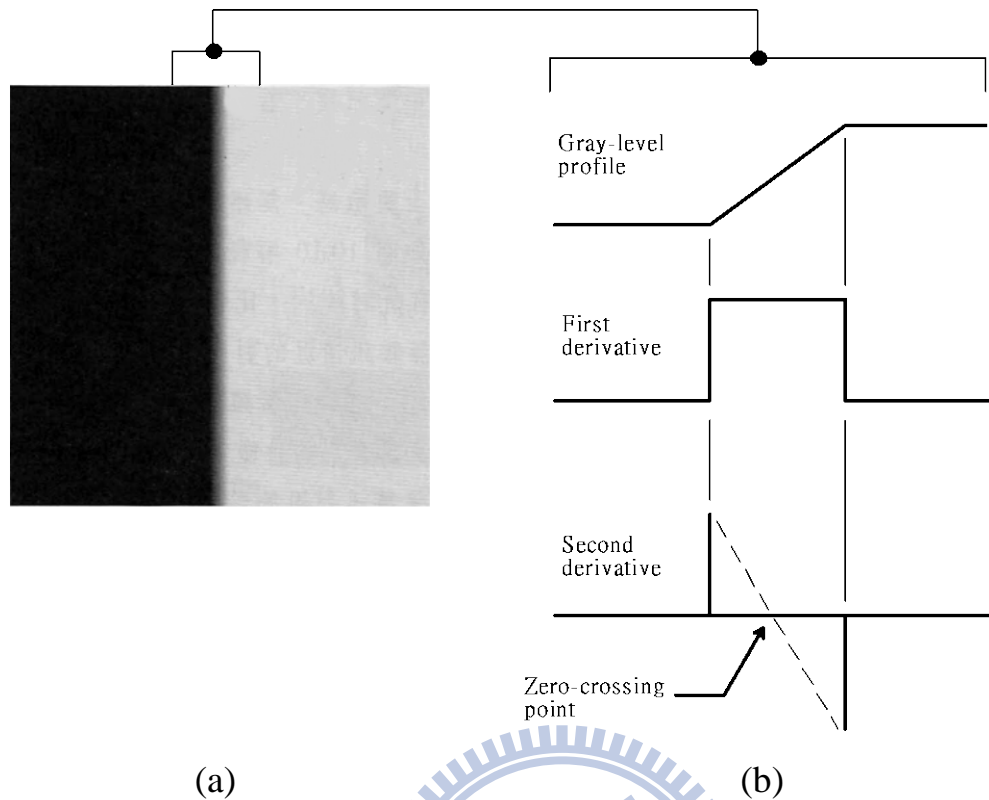
In this thesis, we propose to apply a set of generalized weighted aggregation algorithm to implement the edge detection method with difference for image pixels, and proceeding the numerical correction and update in the size of operating parameters along with the variation of total average accuracy for edge detection. Then, we extend to a set of operating parameters automatic iterative learning mechanism through several repeated operations. We make use of eight grayscale synthetic images with adding different types and rates of random noises as the input images in the parameters automatically learning mechanism. At last, we expect to obtain the best operating parameter set about the performance of edge detection for these eight

input images in a limited number of iterative learning.

## 1.2 Image Edge Detection

### 1.2.1 Image Edge

In the view of imaging science, edge represents the collection of connected pixels between two different objects of an image. Because the edge can recognize the most obvious distinction between different geometric objects, it has become one of the most important features between each geometric object in one image. And edge detection has also become the most important pre-processing step of image signal processing. In the traditional studies of edge detection, the vast majority are based on the concept of first-order and second-order derivative. Figure 1.1 shows the diagram of the image edge and the first and second-order derivative of gray-level intensity; Figure 1.1 (b) indicates a zero-crossing point of the second derivative at the intersection of the horizontal zero-intensity axis and the connection of second derivative. The point not only indicates the location of the image edge, but also specifies the background intensity on both sides of the edge are often far higher or lower than the value on the edge.



**Fig. 1.1** The diagram of image edge and the first and second-order derivative of gray-level intensity, (a) Two regions separated by a vertical edge; (b) The detail near the edge showing a gray-level profile, and the first and second derivatives of the profile.

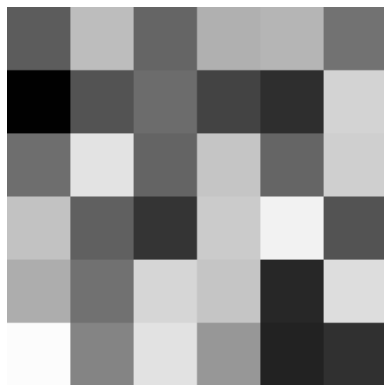
## 1.2.2 Binary Edge Maps

In general, we utilize the binary technique to convert any image to a black-and-white image in order to facilitate the human can easily identify the partition between the edges of objects and the background in the original image when proceeding the edge detection of images. However, due to the amount of data that need to be processed significantly reduce after converting an image to a binary image, it will also enhance the executing speed for other image processing tasks.

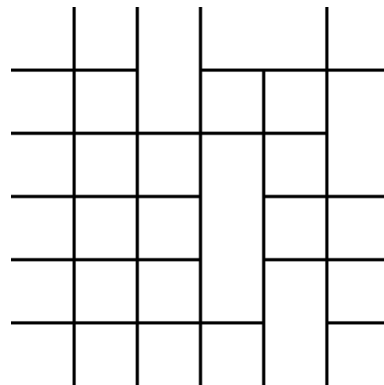
Here we will introduce the basic operation of the binary image processing method:

First, for any image  $I$ , we proceed the necessary numerical or logical operation of edge detection work for gray-level values of each pixel  $I(x, y)$  and its neighbor pixel from left to right, top to bottom respectively, and set an empty array  $B$  with same dimension of image  $I$  in advance. Then, we compare the operation value with the default threshold value, “T”. If the value greater than T, we will mark the corresponding location of pixel  $I(x, y)$  as a black spot which is equal to gray-level value “0” in the array  $B$ , indicating that the pixel belongs to an edge point; Otherwise, it is marked as a white spot which is equal to gray-level value “255” standing for that pixel  $I(x, y)$  is an non-edge point, such as equation 1.1. Finally, we can obtain a binary image clearly depicting the image edges by rendering the array  $B$  in image mode, as shown in Figure 1.2 (b).

$$B(x, y) = \begin{cases} 0(\text{Edge point}), & \text{if } I(x, y) > T \\ 255(\text{Nonedge point}), & \text{if } I(x, y) \leq T \end{cases} \quad (1.1)$$



(a)



(b)

**Fig. 1.2** The result of the gray-scale edge detection by Sobel operator [3],  
 (a) The original gray-level image; (b) The binary edge map obtained by setting the threshold value  $T = 0.001$ .

Since the binary image having the characteristics containing easy to storage, process and identify, it has been used widely in morphology and image recognition processing. Morphology in Imaging Science refers specifically to the technology of extracting the specific components in the images, such as image segmentation, edge detection, thinning, skeleton extraction are the means of the Morphology.

### **1.3 Research Method**

In this thesis, adopting the method proposed by Barrenechea et al. [2], our new edge detection method utilize generalized weighted mean aggregation algorithm to construct interval-valued fuzzy relations. From the weighted mean difference of the central pixel and its 8-neighborhood pixels in a 3x3 sliding window, the upper and lower interval-valued fuzzy relations can be obtained. To increase the edge detection accuracy, we have derived the iterative learning mechanism of the two weighting parameters of the mean aggregation. In the parameters learning phase, we make use of eight grayscale synthetic images with different types and percentages of random noises by the computer program as the input images of the parameters. We have used several edge accuracy indices, which lay different importance multiples of edge-pixel over non-edge-pixel accuracy, so that we can extract the image edge map more sensitively and reliably. Then, we update the weighting parameters of the mean to increase the edge accuracy index of the edge maps of eight images by the steepest descent method cast in discrete formulation. After the learning procedure, we can obtain the best weighting parameters of the mean for the edge detection scheme of all input images through several repeated learning. In our experiments, we have tested by a variety of edge accuracy indices for edge detection, and compare their merits and drawbacks in the edge detection ability of the images.

## 1.4 Thesis Outline

This thesis is organized as follows. The motivation of this study and the basic concept of image edge are introduced in Chapter 1. In Chapter 2, we introduce the method of edge detection for images applying the concept of interval-valued fuzzy relation proposed by Barrenechea et al [2]. In Chapter 3, we describe the concept and method for the iterative learning mechanism of the operating parameter accompanying with the edge detection for images in this thesis, and propose several common random noise types. In Chapter 4, we summarize all experimental results. At last, we integrate all of our studies and discuss the directions for improvement in the future in Chapter 5.



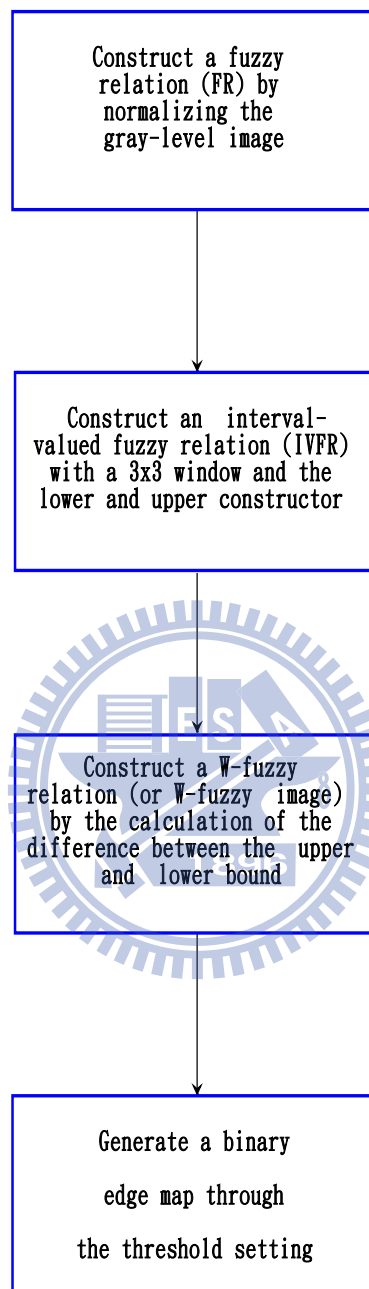
## Chapter 2 Apply Interval-valued Fuzzy Relation to Image Edge Detection

In this chapter, we will introduce to construct interval-valued fuzzy relation (IVFR) [2] of an image by applying  $t$ -norm and  $t$ -conorm (also called  $s$ -norm) in the fuzzy theory to the central pixel and its eight neighbor pixels in a sliding window of the image. For each sliding window, we will calculate the intensity differences between the central pixel and its eight neighbor pixels. To this end, we thus 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.

Thus, we refer to a method proposed by Barrenechea et al [2] making the image edges blurring associated with the interval-valued fuzzy relations. First, we give the definition of an edge in fuzzy terms, since it can indicate that an edge should make it clear that the adjacent pixels having a big enough variation in 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. Finally, we apply the concepts and technology of fuzzy theory in the handler of image edge detection.

Fig. 2.1 illustrates the concepts and steps about the application of interval-valued fuzzy relations in edge detection of images. First of all, we have to normalize the original gray-scale image in order to obtain an image representable by a fuzzy set and an interval-valued fuzzy relation by the lower and upper constructors. We further construct a “W-fuzzy relation” (also called “W-fuzzy edge image” ) that represents the difference between the upper and lower bounds of each interval in an interval-valued matrix. Finally, a binary edge map is generated by threshold

setting from the “W-fuzzy edge image”.



**Fig. 2.1** The flow chart of the application among interval-valued fuzzy relation to image edge detection.

## 2.1 Fuzzy Membership Degree

In the fuzzy theory, fuzzy sets represent the collection of the unclear boundaries



or borders and having specific things. In general, we establish a membership function to represent the relationship of each element to a fuzzy set. The membership function is the basis of fuzzy theory, the purpose of which is to describe some vague phenomena by using the definite and religious mathematical method. It represents the degree of membership of an element belonging to the set by any real number between 0 and 1. If we suppose  $A$  is a fuzzy subset of a universal set  $U$  and the membership function of  $A$  is represented by  $\mu_A$ , then  $\mu_A$  satisfies the following relationship:

$$\mu_A : X \rightarrow [0,1] \quad (2.1)$$

It means that all the values of  $\mu_A(x)$  are between 0 and 1 for all  $x \in U$ . Zadeh ever mentioned that the degree of membership of  $x$  belonging to  $A$  is higher as the value of  $\mu_A(x)$  closer to “1” in the theory of fuzzy sets in 1965. And if the value of  $\mu_A(x)$  approaches to “0”, it indicates the degree of  $x$  belonging to  $A$  is very low. Therefore, it not only can express clearly the degree of membership of the elements belonging to the set but also show all the values of “Transition” between “Yes” and “No” .

## 2.2 Interval-valued Fuzzy Relation

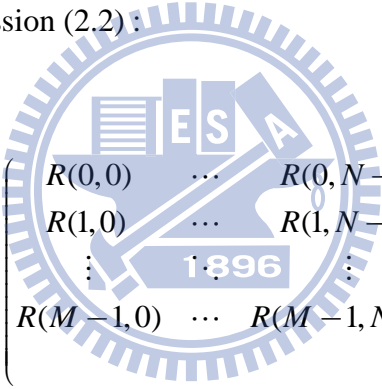
Base on the fuzzy relation, we can define two fuzzy relations of an image, one is “lower” and the other is “upper” constructor to constitute the interval-valued fuzzy relation with the same dimensions of the images [6]. It is to be noticed that we must consider carefully the gray-scale intensity of each pixel and its neighbor pixels contained in a fixed range of the testing image in the procedure, for example, we

consider eight pixels contained in the neighborhood when using a sliding window with size of 3x3.

### 2.2.1 Fuzzy Relation

For a grayscale image with size of  $M \times N$ , we execute the normalization which let all the gray-scale values of pixels in the image converted to a number between 0 and 1 by dividing all of them by the grayscale maximal intensity “255” in advance.

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 described by matrices in the following way, as the expression (2.2):



$$R = \begin{pmatrix} R(0,0) & \dots & R(0,N-1) \\ R(1,0) & \dots & R(1,N-1) \\ \vdots & & \vdots \\ R(M-1,0) & \dots & R(M-1,N-1) \end{pmatrix} \quad (2.2)$$

Besides,  $F(X \times Y)$  represents the set of all fuzzy relations from  $X$  to  $Y$  [8], [9].

### 2.2.2 Lower and Upper Constructor

We can define the expressions of the lower bound and the upper bound respectively for a k-tuple pair  $(x_1, x_2, \dots, x_k)$  in which  $x_1, x_2, \dots, x_k \in [0,1]$  as followed:

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

$$\bigotimes_{i=1}^k x_i = S(\bigotimes_{i=1}^k x_i, x_k) = S(x_1, x_2, \dots, x_k) \quad (2.4)$$

In the above expressions, “ $T$ ” and “ $S$ ” stand for one kind of  $t$ -norm and  $s$ -norm operators introduced in fuzzy theory; In this section, we take the common “min” and “max” operators for examples. And the operations of them are expressed as followed:

$$T_M(x, y) = \min(x, y), \quad x, y \in [0, 1] \quad (2.5)$$

$$S_M(x, y) = \max(x, y), \quad x, y \in [0, 1] \quad (2.6)$$

Let  $R \in F(X \times Y)$  be a fuzzy relation. 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  $n \leq \frac{N-1}{2}$ . We define the lower constructor associated with  $T_1, T_2, m$ , and  $n$  in the following way:

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

For all  $(x, y) \in X \times Y$ , and where 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  $n$  indicate that the considered sliding window is a matrix of dimension  $(2m+1) \times (2n+1)$ , which is centered at  $(x, y)$ .

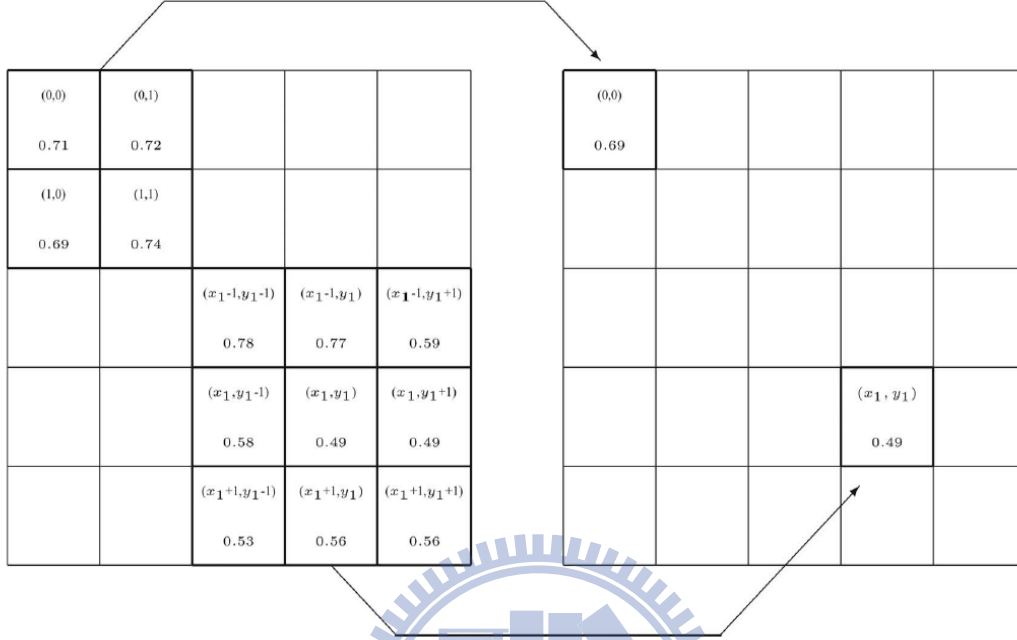
Analogously, if we consider two  $t$ -conorms ( $s$ -norms)  $S_1$  and  $S_2$ , then we define the upper bound of an interval in the following way:

$$U_{S_1, S_2}^{m,n} [R](x, y) = \underset{j=-n}{\overset{i=-m}{S_1}} \left( \underset{j=-n}{\overset{i=-m}{S_2}} (R(x-i, y-j), R(x, y)) \right) \quad (2.8)$$

For simplicity, if  $m=n$ , then we denote  $L_{T_1, T_2}^{m,n}$  and  $U_{S_1, S_2}^{m,n}$  as  $L_{T_1, T_2}^m$  and  $U_{S_1, S_2}^m$ , respectively.

In Fig. 2.2, we graphically illustrate how the lower constructor operation works with  $m=n=1$ , and  $T_1=T_2=T_M$ . For the element  $(0,0)$  and  $(x_1, y_1) \in X \times Y$  in the fuzzy relation, we have:

$$\begin{aligned} L_{T_M, T_M}^1 [R](0, 0) &= \min(\min(0.74, 0.71), \min(0.69, 0.71), \min(0.72, 0.71), \\ &\quad \min(0.71, 0.71)) = 0.69 \\ L_{T_M, T_M}^1 [R](x_1, y_1) &= \min(\min(0.56, 0.49), \min(0.56, 0.49), \min(0.53, 0.49), \\ &\quad \min(0.49, 0.49), \min(0.49, 0.49), \min(0.58, 0.49), \\ &\quad \min(0.59, 0.49), \min(0.77, 0.49), \min(0.78, 0.49)) \\ &= 0.49 \end{aligned}$$



**Fig. 2.2** The diagram of the operation of lower constructor for specific pixels in an image.

### 2.2.3 Interval-valued Fuzzy Matrix

Let  $R \in F(X \times Y)$  be a fuzzy relation. If we consider a lower constructor  $L_{T_1, T_2}^{m, n}$  and an upper constructor  $U_{S_1, S_2}^{m, n}$ , then an interval-valued fuzzy relation  $R^{m, n}$  will be denoted as an  $M \times N$  matrix:

$$R^{m, n} = \begin{pmatrix} [L_{T_1, T_2}^{m, n} [R](0, 0), U_{S_1, S_2}^{m, n} [R](0, 0)] & \cdots & [L_{T_1, T_2}^{m, n} [R](0, N-1), U_{S_1, S_2}^{m, n} [R](0, N-1)] \\ \vdots & \ddots & \vdots \\ [L_{T_1, T_2}^{m, n} [R](M-1, 0), U_{S_1, S_2}^{m, n} [R](M-1, 0)] & \cdots & [L_{T_1, T_2}^{m, n} [R](M-1, N-1), U_{S_1, S_2}^{m, n} [R](M-1, N-1)] \end{pmatrix} \quad (2.9)$$

which all of the elements in this matrix  $R^{m, n}$  represent an interval, as follows:

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

where  $X$  and  $Y$  are two sets  $\{0, 1, \dots, M-1\}$  and  $\{0, 1, \dots, N-1\}$  as mentioned in section 2.2.1, and  $L([0, 1])$  represents the set of all closed subintervals of  $[0, 1]$ . If  $m = n$ , then we denote  $R^{m,n}$  as  $R^m$ .

### 2.3 W-Fuzzy Relation and W-Fuzzy Edge

In this section, we will compute the length between the upper and lower bound of each interval from the given interval-valued fuzzy matrix  $R^{m,n}$ , and construct a new fuzzy relation  $W[R^{m,n}]$  by the values, which is defined as follows:

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

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

where the magnitude of  $W[R^{m,n}](x, y)$  represents the membership degree associated with a pixel in the fuzzy relation of the original image, that means the degree of difference between the gray-scale values of pixels contained in a  $(2m+1) \times (2n+1)$  window for this pixel. Therefore, we can obtain following conclusion with this W-fuzzy relation,  $W[R^{m,n}]$ :

- (1) We have that if the length associated with a pixel is maximal (i.e., in the window considered, we have at least one white pixel and at least one black pixel), then the pixel is always considered an edge.
- (2) We have that if the window centered at  $(x, y)$  has a constant intensity, then the length of the associated interval is zero. Therefore, the pixel will never be considered as part of an edge.

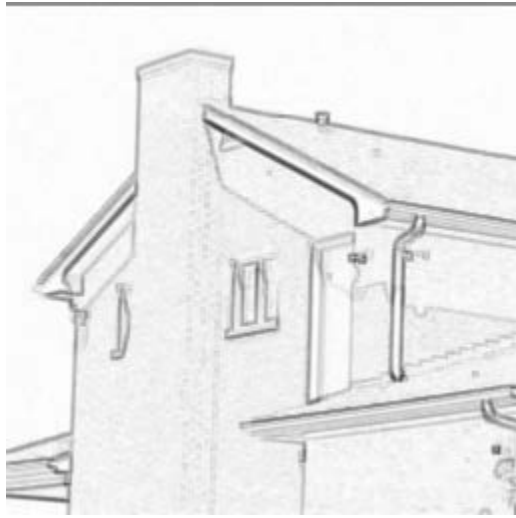
In the fuzzy theory, we call  $W[R^{m,n}]$  the W-fuzzy relation. In the image processing field, due to the relation  $W[R^{m,n}]$  represents a fuzzy edge image that can not show the definite differences of intensity value, but visually captures the intensity changes clearly between a pixel and its neighbor pixels, it is named as a W-fuzzy edge image specifically in the reference proposed by Barrenechea et al [2]; for this reason, this image can be considered as an image that represents edges in a fuzzy way. This fact will enable us to better adjust to the application in which we want to use our edge detector based on W-fuzzy edge images. Fig. 2.3 shows two gray-scale natural images and their W-fuzzy edge images.



(a)



(b)

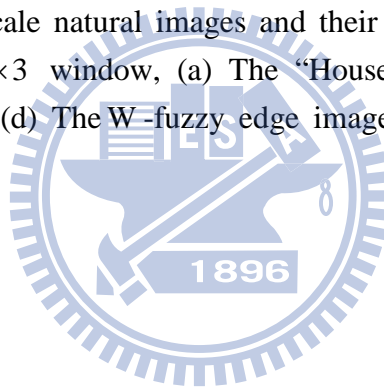


(c)



(d)

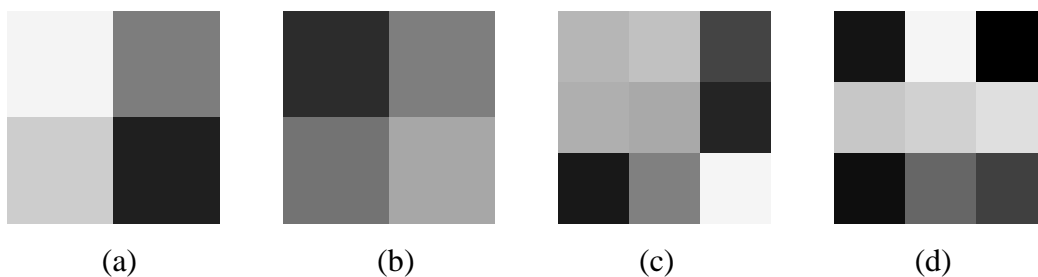
**Fig. 2.3** The gray-scale natural images and their W - fuzzy edge images by using a  $3 \times 3$  window, (a) The “House” image; (b) The “Lenna” image; (c)-(d) The W -fuzzy edge images of “House” and “Lenna” images.

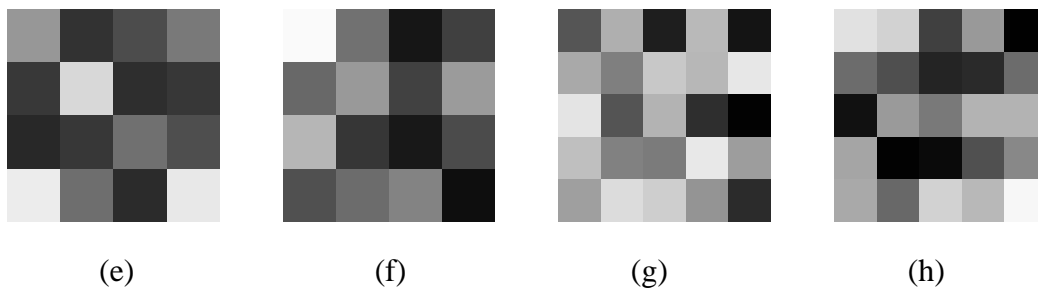




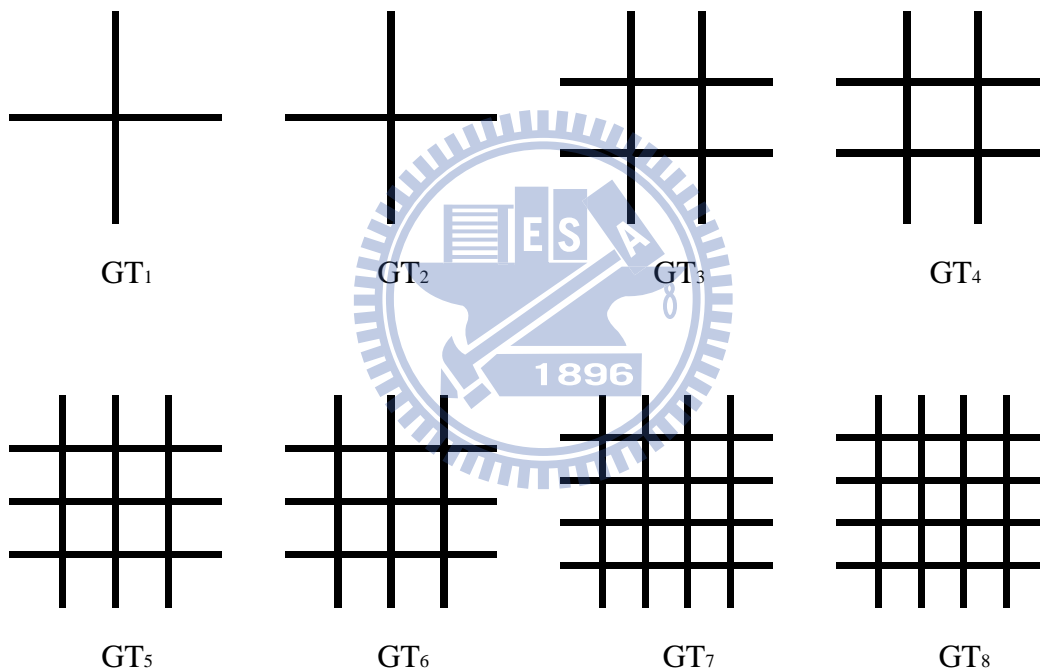
# Chapter 3 The Best Learning of Operating Parameters For Image Edge Detection

In this chapter, we will extend the edge detection method based on the concept of W-fuzzy relation introduced in Chapter 2. We propose an edge detection method of the weighted mean difference aggregated algorithm for pixel values between a central pixel and its 8 neighbor pixels in a  $3 \times 3$  window by using two weighting parameters within  $[0,1]$ , and constitute a set of parameters automatic learning mechanism by several iterative operations. In the learning procedure, we adopt eight synthetic gray-scale images with size of  $60 \times 60$  that the gray-level values are randomly generated by the computer program as the input images for parameter learning, as shown in Fig. 3.1. And then, our principle to modify and update the operating parameters is according to the variation of average accuracy index for edge detection calculated by the binary edge maps obtained from our edge detection method and the individual ground truths (GT) edge maps of eight images in each parameter learning epoch.





**Fig. 3.1** Eight gray-scale synthetic images randomly generated by a computer program with size of  $60 \times 60$ .



**Fig. 3.2** The ground truth edge maps for eight gray-scale synthetic images.

### 3.1 The Introduction of Operating Parameters

In the thesis proposed by Barrenechea et al [2], the selections of  $t$ -norm and  $s$ -norm operators do not have sufficient flexibility, and all of them are nonlinear operands as well as hard to derive a common formulation such as the “max” and

“min” logical operators introduced in Section 2.2.2. In order to improve the disadvantage and increase the accuracy index for edge detection, we develop a weighted mean common equation of linear or power with a set of generalized T-type and S-type operands “ $\alpha_T$ ” and “ $\alpha_S$ ” in order to replace the  $t$ -norm and  $s$ -norm operators. By this way, we can obtain an output value of each pixel in an image like the “W-fuzzy relation” do. At last, the goal of edge detection is achieved through the operation of a threshold.

On the basis of above principles, there are two operating parameters in our learning system, defined respectively as follows:

“ $\alpha_T$ ”: T-type operating parameter. It’s used for replacing the efficacy of t-norm operators, meaning to construct a lower bound of an interval for an image. And we limit its value within [0, 0.5].

“ $\alpha_S$ ”: S-type operating parameter. It’s used for replacing the efficacy of s-norm operators, meaning to construct an upper bound of an interval for an image. And we limit its value within [0.5, 1].

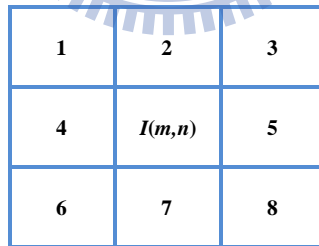
It will lead to the up and down variation of accuracy index for edge detection by applying two designed operating parameters. Then, we correct and update the operating parameters according to this variation of accuracy index in the parameter learning system later.

### **3.2 The Weighted Mean Difference Aggregation Calculation of Pixel Values**

In this research, in addition to have the parameters “ $\alpha_T$ ” and “ $\alpha_S$ ” to determine the strength of T-type and S-type operands, we also promote them to an edge detection method by performing the weighted mean difference calculation for pixel values in an image, and expect to obtain a set of operating parameters with best capability in edge detection through a series of parametric designs. We will introduce the weighted mean difference calculation of “linear” and “quadratic” types for pixel values in the following.

### 3.2.1 The Aggregation Calculation of Linear Type

For a  $M \times N$  image  $I$ , we separately compute the linear weighted mean operating value of gray-scale intensity between the central pixel  $I(m,n)$  and its eight neighbor pixels in a  $3 \times 3$  window by using the generalized weighted operands “ $\alpha_T$ ” and “ $\alpha_S$ ” as shown in following:



**Fig. 3.3** The diagram of a  $3 \times 3$  window centered at pixel  $I(m,n)$  associated with its eight neighbor pixels.

$$\begin{cases} y_{Ti}(I(m,n)) = \alpha_T a_i(I(m,n)) + (1 - \alpha_T) b_i(I(m,n)) \\ y_{Si}(I(m,n)) = \alpha_S a_i(I(m,n)) + (1 - \alpha_S) b_i(I(m,n)) \end{cases}, \quad \alpha_T, \alpha_S \in [0,1] \quad (3.1)$$

$(i = 1, 2, \dots, 8; m = 1, 2, \dots, M, n = 1, 2, \dots, N)$

where the subscript “i” represents eight neighbor pixels and the operating parameters “ $\alpha_T$ ”, “ $\alpha_S$ ” must satisfy  $\alpha_T < \alpha_S$ ,  $a_i(I(m,n))$  and  $b_i(I(m,n))$  represent the larger and smaller gray-scale value respectively between the central pixel  $I(m,n)$  and its neighbor pixels. And then, we further compute the average of eight  $y_{Ti}$  and  $y_{Si}$  as follows:

$$\overline{y_T}(m,n) = \frac{\sum_{i=1}^8 y_{Ti}(I(m,n))}{8}, \quad \overline{y_S}(m,n) = \frac{\sum_{i=1}^8 y_{Si}(I(m,n))}{8} \quad (3.2)$$

At last, we obtain an output value  $y_w$  for each pixel like as the meaning of W-fuzzy relations :

$$\begin{aligned} y_w(m,n) &= \overline{y_S}(m,n) - \overline{y_T}(m,n) \\ &= \frac{(\alpha_S - \alpha_T) \sum_{i=1}^8 (a_i(I(m,n)) - b_i(I(m,n)))}{8} \end{aligned} \quad (3.3)$$

We can determine whether a pixel  $I(m,n)$  belongs to edge pixel or not according to the comparison of this output value and an adaptive threshold  $T$  as shown in the expression 3.4. We also can obtain the degree of intensity difference between the central pixel  $I(m,n)$  and its neighbor pixels in the same time .

$$\begin{cases} I(m,n) \text{ is an edge point, if } y_w(m,n) \geq T \\ I(m,n) \text{ is a nonedge point, if } y_w(m,n) < T \end{cases} \quad (3.4)$$

### 3.2.2 The Aggregation Calculation of Quadratic Type

We try to change  $a_i(I(m,n))$  and  $b_i(I(m,n))$  into  $a_i^2(I(m,n))$  and  $b_i^2(I(m,n))$  in order to strengthen the power of intensity variation between two pixels. Analogously, we execute the weighted mean aggregation calculation with the operands “ $\alpha_T$ ” and “ $\alpha_S$ ” by considering a  $3 \times 3$  window centered at pixel  $I(m,n)$  shown in Fig. 3.3 and calculate the value  $y_T$  and  $y_S$  by taking account of the gray-scale values of pixel  $I(m,n)$  and its eight neighbor pixels as shown in 3.5:

$$\begin{cases} y_{Ti}(I(m,n)) = \alpha_T a_i^2(I(m,n)) + (1 - \alpha_T) b_i^2(I(m,n)) \\ y_{Si}(I(m,n)) = \alpha_S a_i^2(I(m,n)) + (1 - \alpha_S) b_i^2(I(m,n)) \end{cases}, \quad \alpha_T, \alpha_S \in [0,1] \quad (3.5)$$

( $i = 1, 2, \dots, 8; m = 1, 2, \dots, M, n = 1, 2, \dots, N$ )

In the same way, we calculate the average of the eight  $y_{Ti}$  and  $y_{Si}$  as follows:

$$\overline{y_T}(m,n) = \frac{\sum_{i=1}^8 y_{Ti}(I(m,n))}{8}, \quad \overline{y_S}(m,n) = \frac{\sum_{i=1}^8 y_{Si}(I(m,n))}{8} \quad (3.6)$$

Then, the output value  $y_w$  for pixel  $I(m,n)$  is defined as follows:

$$y_w(m,n) = \sqrt{\overline{y_S}(m,n) - \overline{y_T}(m,n)} \quad (3.7)$$

$$= \sqrt{\frac{(\alpha_S - \alpha_T) \sum_{i=1}^8 (a_i^2(I(m,n)) - b_i^2(I(m,n)))}{8}}$$

### 3.3 Parameter Automatic Learning Mechanism

### 3.3.1 The Setting of Initial Operating Parameters and Thresholds

Before proceeding the parameter automatic learning, we must give a set of initial operating parameters for it. By this way, the weighted mean difference aggregation calculation for pixel values will be started from this set of operating parameter, and the repeated numerical correction of operating parameter would be adjusted towards the direction to enhance the capability in edge detection. Besides, it is necessary to set a threshold for deciding whether a pixel belongs to edge or not and obtain the edge detection accuracy index to judge the correcting direction for the parameter value. And the size of threshold usually depends on the nature of input images.

### 3.3.2 The Jacobi Error Correction with Steepest Gradient Method

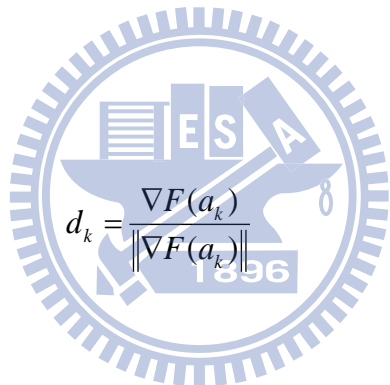
The basic concept of the steepest gradient method [10] is applying the theory of Calculus to search the local extremes of any function  $F(a)$  that a simple expression of it is:

$$a_{k+1} = a_k \pm \eta d_k \quad (k = 0, 1, 2, \dots), \quad d_k = \nabla F(a_k) \quad (3.8)$$

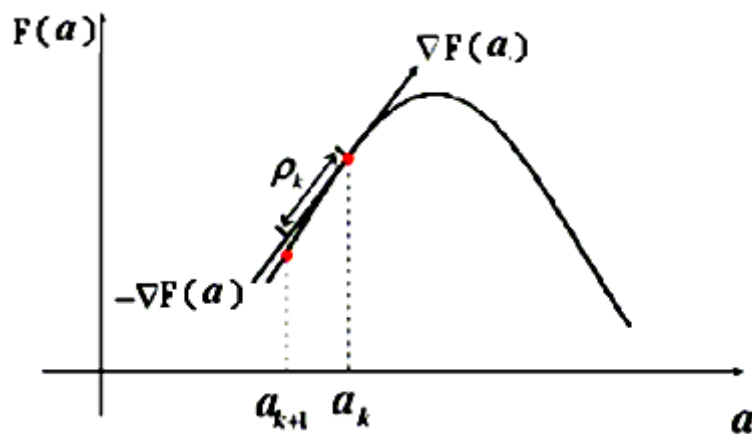
which “ $\eta$ ” is called “learning ratio constant” that is generally set between 0 and 1, and  $d_k$  denote the gradient value at point  $a_k$  in the function  $F$ . For the steepest gradient method, it can reach the maximum value of a function at the soonest speed going along with a certain point in its positive gradient direction; on the contrary, it reach the minimum of a function at the soonest speed going in the negative gradient direction.

First, we have to choose an initial point  $a_0$  and determine the increment or

decrement in the direction of rapidest ascending or descending at  $F(a_0)$  through an appropriate gradient value. Then, we further seek for a new searching direction repeatedly. As shown in Fig. 3.4,  $\nabla F(a_k)$  represents the magnitude in need of changing for the steepest gradient ascent or descent obtained from the first-order partial differential at point  $a$  of the function  $F$ . And the gradient  $\nabla F(a_k)$  of any point  $a_k$  in the function  $F$  is a vector which the direction is increased most rapidly of the value in the function  $F$ ; In contrast, the direction of negative gradient is the one that decreased most rapidly of the value in the function  $F$ . For any point  $F(a_k)$  of function  $F$ , we can define the unit vector of the searching direction of its gradient as follows:

$$d_k = \frac{\nabla F(a_k)}{\|\nabla F(a_k)\|} \quad (3.9)$$


Move on the point  $F(a_k)$ , we can obtain a new point  $a_{k+1} = a_k \pm \rho_k d_k$  by going forward  $\rho_k$  steps along the direction of  $d_k$  or  $-d_k$ .



**Fig. 3.4** The diagram of the steepest gradient method.



Through the iterative calculation repeatedly, all of “ $a$ ” will consist a sequence  $\{a_0, a_1, a_2, \dots, a_k, a_{k+1}, \dots\}$  which must converge to the maximum value of function  $F$  under some specific conditions. The more important is that the extreme solution we get would be just one among the local extremes instead of the global one for this kind of iterative calculation.

Then, we apply the notion of steepest gradient method to search the best operating parameters for edge detection in the parameter learning mechanism. Moreover, we execute the correction and update of parameters according to the variation of total average accuracy for edge detection in the learning mechanism, and perform the weighted mean difference aggregation calculation of pixel values with new operating parameters repeatedly in next iteration. In addition, we adopt the method of Jacobi error correction proposed by M. Emin Yüksel et al in the reference [11] as the basis of parameters correction.

First of all, let  $w$  be a parametric vector consisted of  $\alpha_T(t)$  and  $\alpha_S(t)$  for the  $t$ -th learning epoch as follows:

$$w = \begin{pmatrix} \alpha_T(t) \\ \alpha_S(t) \end{pmatrix}_{2 \times 1} \quad (3.10)$$

And  $\Delta w$  denote a vector consisted of  $\Delta \alpha_T(t)$  and  $\Delta \alpha_S(t)$  as follows:

$$\Delta w = \begin{pmatrix} \Delta \alpha_T(t) \\ \Delta \alpha_S(t) \end{pmatrix}_{2 \times 1} = [J^T(w)J(w) + \mu I]^{-1} J^T(w)E(w) \quad (3.11)$$

In the above expression,  $J$  represents a Jacobi matrix defined as follows:

$$J = \begin{pmatrix} \frac{\partial E}{\partial \alpha_T} & \frac{\partial E}{\partial \alpha_S} \end{pmatrix} \quad (3.12)$$

and  $\mu$  is an arbitrarily small constant for the purpose that we can obtain the vector  $\Delta w$  when  $J^T(w)J(w)$  is a singular matrix (we choose  $\mu = 0.01$  in our research). In addition,  $I$  is a  $2 \times 2$  identity matrix and  $E$  denotes an error function of any pixel  $I_k(m, n)$  in the  $k$ -th input image  $I_k$ :

$$E = GT_k(m, n) - y_{w_k}(m, n) \quad (3.13)$$

For the operation in equation 3.3, the elements in Jacobi matrix are denoted as follows:

$$\begin{aligned} \frac{\partial E}{\partial \alpha_T} &= \frac{\partial(GT_k(m, n) - y_{w_k}(m, n))}{\partial \alpha_T} = \frac{\partial(\overline{y_{T_k}(m, n)})}{\partial \alpha_T} \\ &= \frac{\sum_{i=1}^8 (a_i(I_k(m, n)) - b_i(I_k(m, n)))}{8} \end{aligned} \quad (3.14)$$

$$\frac{\partial E}{\partial \alpha_S} = \frac{\sum_{i=1}^8 (b_i(I_k(m, n)) - a_i(I_k(m, n)))}{8} \quad (3.15)$$

And for the operation in equation 3.7, the results will be as follows:

$$\begin{aligned}
\frac{\partial E}{\partial \alpha_T} &= \frac{\partial(GT_k(m,n) - y_{W_k}(m,n))}{\partial \alpha_T} = \frac{\partial(-\sqrt{y_{S_k}(m,n) - y_{T_k}(m,n)})}{\partial \alpha_T} \quad (3.16) \\
&= \frac{\partial(-\sqrt{y_{S_k}(m,n) - y_{T_k}(m,n)})}{\partial(y_{S_k}(m,n) - y_{T_k}(m,n))} \frac{\partial(y_{S_k}(m,n) - y_{T_k}(m,n))}{\partial \alpha_T} \\
&= \frac{-1}{2\sqrt{y_{S_k}(m,n) - y_{T_k}(m,n)}} \frac{(-\sum_{i=1}^8 (a_i^2(I_k(m,n)) - b_i^2(I_k(m,n))))}{8} \\
&= \frac{1}{2\sqrt{\frac{(\alpha_S - \alpha_T) \sum_{i=1}^8 (a_i^2(I_k(m,n)) - b_i^2(I_k(m,n)))}{8}}} \frac{\sum_{i=1}^8 (a_i^2(I_k(m,n)) - b_i^2(I_k(m,n)))}{8} \\
&= \frac{1}{4} \sqrt{\frac{\sum_{i=1}^8 (a_i^2(I_k(m,n)) - b_i^2(I_k(m,n)))}{2(\alpha_S - \alpha_T)}} \\
\frac{\partial E}{\partial \alpha_S} &= -\frac{1}{4} \sqrt{\frac{\sum_{i=1}^8 (a_i^2(I_k(m,n)) - b_i^2(I_k(m,n)))}{2(\alpha_S - \alpha_T)}} \quad (3.17)
\end{aligned}$$

Next, we will calculate the average of correction for all pixels in these eight images as shown in the following expression:

$$\Delta w_{ave} = \begin{pmatrix} \Delta \alpha_{T_{ave}}(t) \\ \Delta \alpha_{S_{ave}}(t) \end{pmatrix}_{2 \times 1} = \begin{pmatrix} \frac{\sum_{m=1}^M \sum_{n=1}^N \sum_{k=1}^8 \Delta \alpha_{T_k}(m,n;t)}{(M \times N \times 8)} \\ \frac{\sum_{m=1}^M \sum_{n=1}^N \sum_{k=1}^8 \Delta \alpha_{S_k}(m,n;t)}{(M \times N \times 8)} \end{pmatrix}_{2 \times 1} \quad (3.18)$$

And we regard  $\Delta \alpha_{T_{ave}}(t)$  and  $\Delta \alpha_{S_{ave}}(t)$  as the basic unit of correction for the

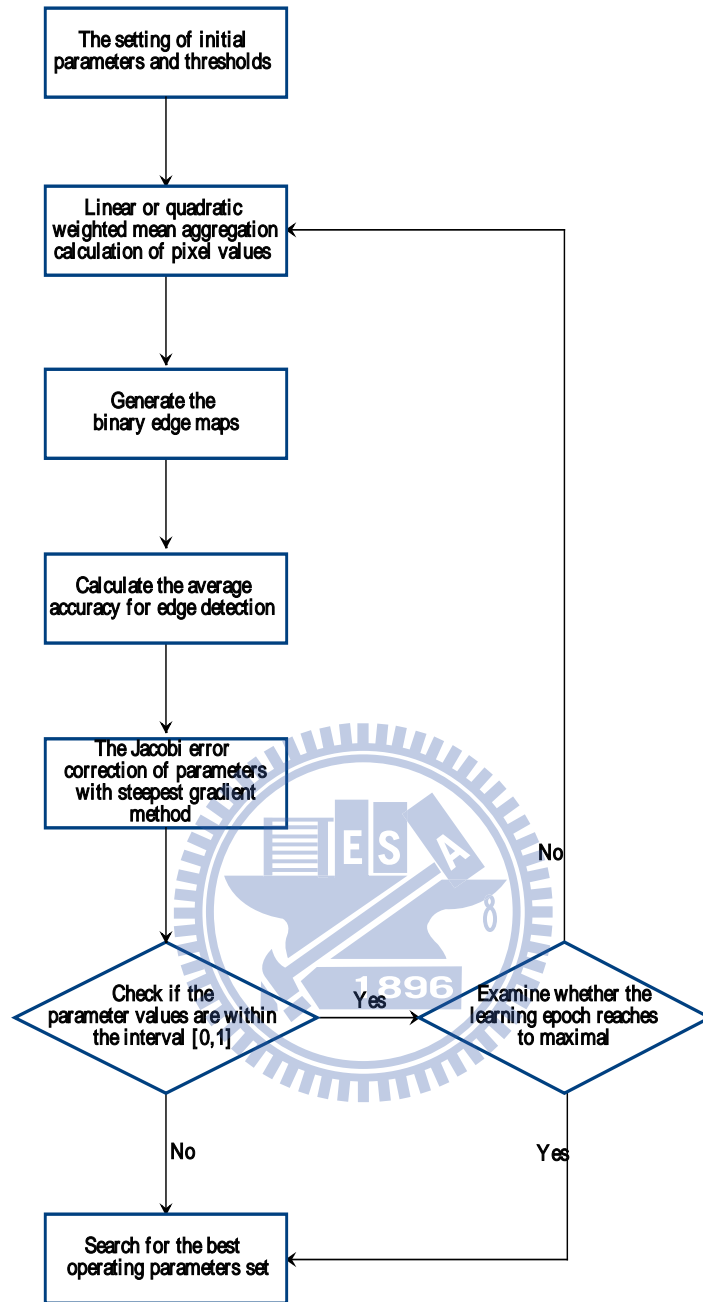
operating parameters “ $\alpha_T$ ” and “ $\alpha_S$ ” in next epoch as shown in expression 3.19:

$$\begin{cases} \alpha_T(t+1) = \alpha_T(t) \pm \eta \Delta \alpha_{T_{ave}}(t) \\ \alpha_S(t+1) = \alpha_S(t) \pm \eta \Delta \alpha_{S_{ave}}(t) \end{cases} \quad (3.19)$$

which the learning ratio constant “ $\eta$ ” can be arbitrarily chosen an adaptive value, and the increment or decrement of parameters “ $\alpha_T$ ” and “ $\alpha_S$ ” depends on the variation of average accuracy index in edge detection of all input images.

### 3.3.3 The Best Operating Parameters

In the end of each learning epoch, we calculate the total average edge detection accuracy of all working images and find out the best operating parameters  $\alpha_{T_{opt}}$  and  $\alpha_{S_{opt}}$  corresponding to the highest accuracy through several iterative learning. From the experimental results, we find it will affect the value of best operating parameters more or less with different initial operating parameters or different accuracy indices for edge detection. Fig. 3.5 roughly depicts the procedure of entire parameter automatic learning mechanism.



**Fig. 3.5** The flow chart of the automatic learning mechanism for operating parameters.

### 3.4 Random Noise

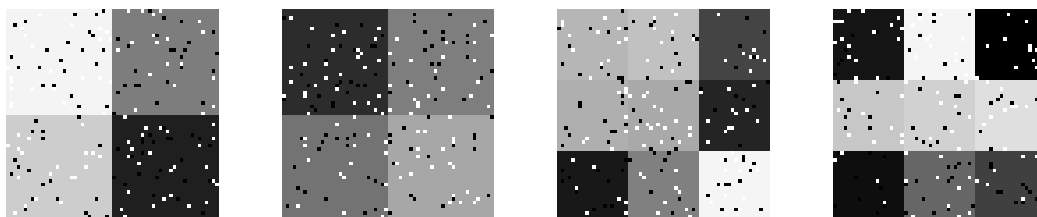
In machine vision systems, noise often arises out of the using of cameras or video capture cards. In addition, the video signal may be subjected to interference in the

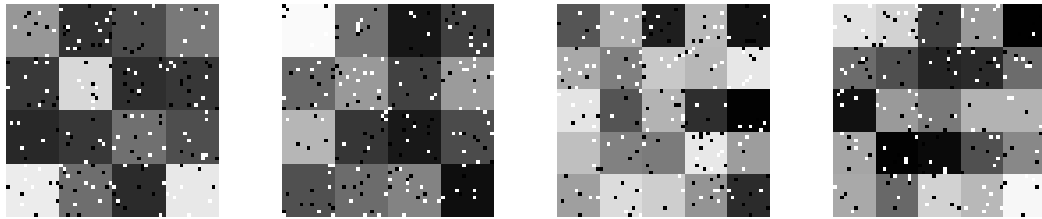
transmission of an image from place A to place B, and the probability of occurrence for random noise is higher as the transmission distance is longer. Thus, in this thesis, we mix “Impulse” and “Gaussian” noise randomly in the original gray-scale input images by MATLAB program without loss of generality, and further use these images to proceed the parameter automatic learning for the purpose of best edge detection.

### 3.4.1 Impulse Noise

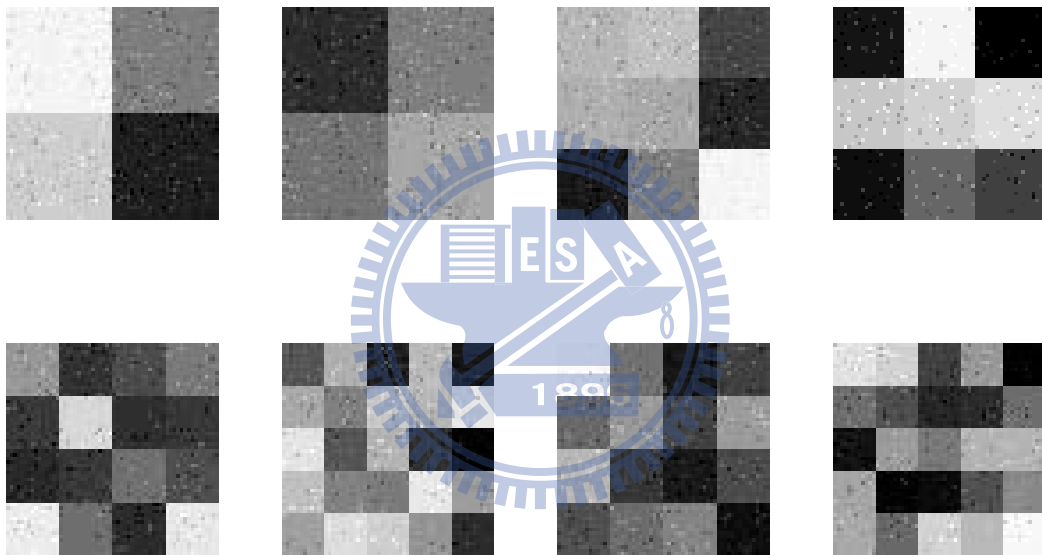
Impulse noise is usually a black or white dot in an image. Because its color is similar to “pepper” black or “salt” white, we also call it the “salt and pepper noise” that the gray-scale intensity “0” represents for “pepper” noise, and the intensity “255” is in behalf of “salt” noise in the field of imaging science. Such noise usually has a higher frequency and is out of tune with the adjacent pixels, so it is easy to be identified. In our research, we try to dope impulse noise with the amount of 5, 8, or 10 percent in the original gray-scale images of Fig. 3.1 as shown in Fig. 3.6.

Besides, we also test by mixing one kind of impulse random noise within an arbitrary positive fixed value  $P$  in the original gray-scale image. The way of operation is to increase or decrease the gray-scale intensity with an positive number not greater than  $P$  for 5~10 percent amount of pixels chosen at random in an image as shown in Fig. 3.7.





**Fig. 3.6** Eight gray-scale images with 5% impulse random noise.



**Fig. 3.7** Eight gray-scale images with 10% impulse random noise within a fixed value  $P=50$ .

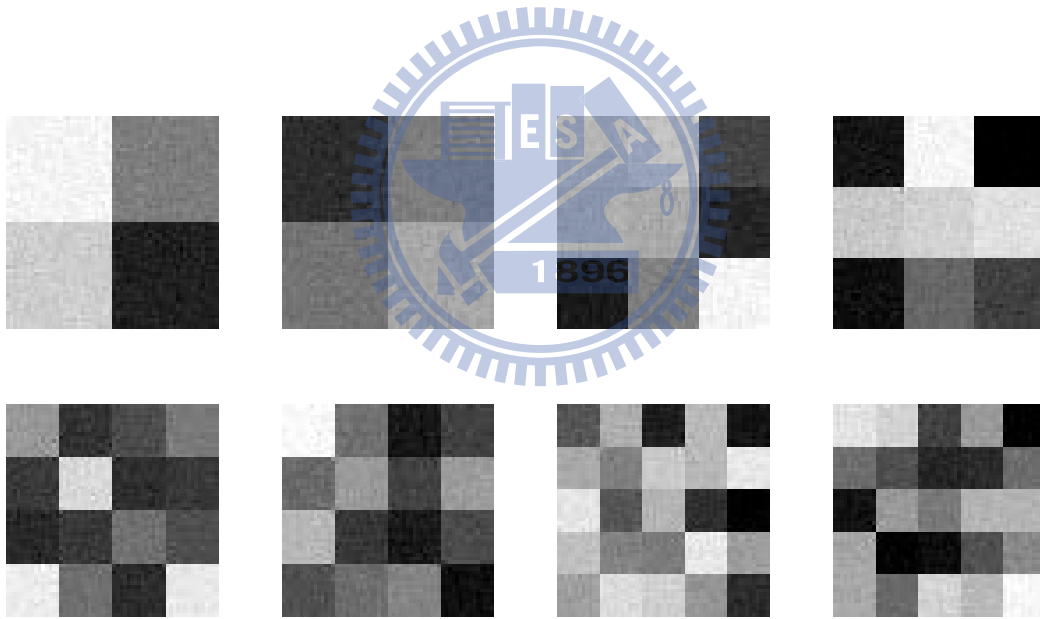
### 3.4.2 Gaussian Noise

In general situation, it would be quite possible that there exists Gaussian noise as the pixel values we capture are different for each time. And the expression of

Gaussian function is shown as follows:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (3.20)$$

which “ $\mu$ ” denotes the mean value, and “ $\sigma$ ” denotes the standard deviation. However, due to we mix Gaussian noise in original images by the instruction “imnoise” of MATLAB program, it is necessary to normalize the gray-scale images in advance in order to match the instruction. After doping the Gaussian noise in these images, we convert them back to the normal gray-scale images as shown in Fig. 3.8.

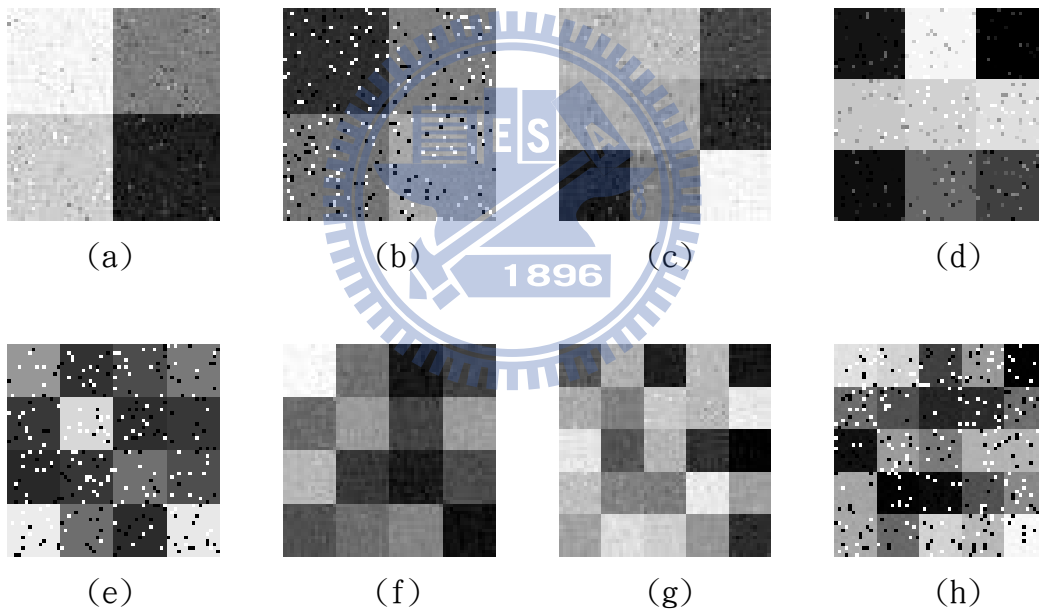


**Fig. 3.8** Eight gray-scale images with Gaussian random noise ( $\mu = 0$ ,  $\sigma=6$ ).



## Chapter 4 Experimental Results

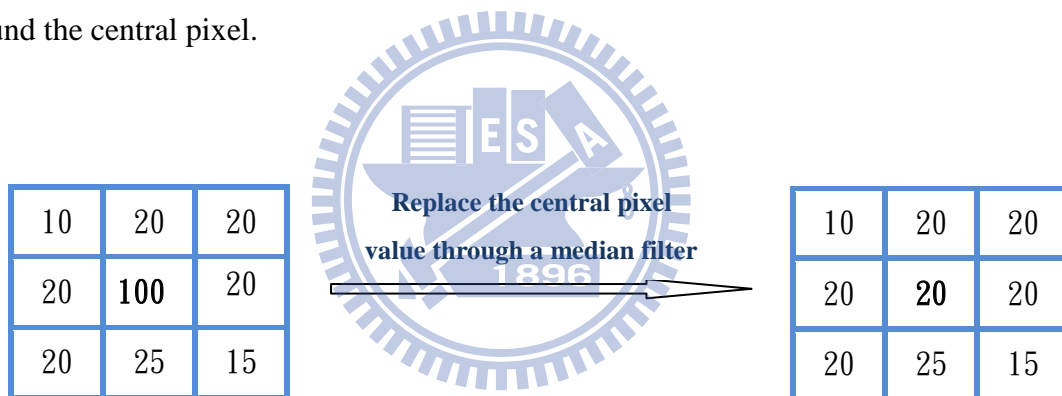
In the experiments, we try to mix some different proportions and types of random noises for eight gray-scale images in Fig. 3.1 respectively. As shown in Fig.4.1, in order to prevent too many noise pixels misjudged as edge for the edge detection result, we make a preprocessing to remove a few noise in these images by a  $3 \times 3$  median filter before proceeding the parameters automatic learning. And then, we perform the parameters learning for edge detection with linear and quadratic weighted mean difference algorithms of image pixels introduced in Section 3.2.



**Fig. 4.1** Eight gray-scale synthetic images mixed with different types of random noises, (a) With 5% impulse noise within a fixed value  $P=80$  and Gaussian noise ( $\mu=0, \sigma=5.5$ ), (b) With 8% impulse noise and Gaussian noise ( $\mu=0, \sigma=4.5$ ), (c) With 10% impulse noise within a fixed value  $P=50$  and Gaussian noise ( $\mu=0, \sigma=3.5$ ), (d) With 10% impulse noise within a fixed value  $P=80$ , (e) With 8% impulse noise, (f) With Gaussian noise ( $\mu=0, \sigma=6$ ), (g) With 8% impulse noise within a fixed value  $P=30$  and Gaussian noise ( $\mu=0, \sigma=4.5$ ), (h) With 10% impulse noise and Gaussian noise ( $\mu=0, \sigma=3$ ).

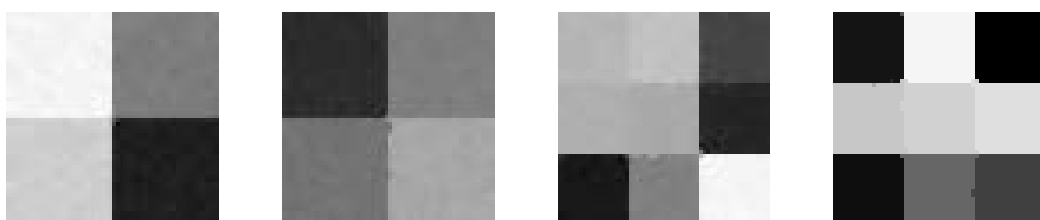
## 4.1 Median Filter

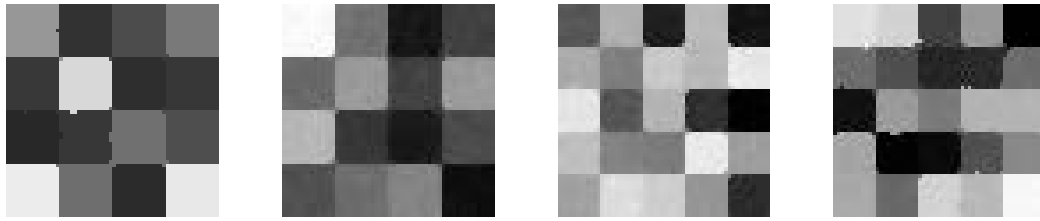
In general, using a feasible filter can make a smoothing effect for images without changing the structure of image pixels. Specifically, the median filter can remove the stronger high frequency noise effectively but still keep the specific edge shape and not blur an image easily. The method of operation is to sort all the pixel values in a window and seek for the median value to replace the central pixel in the window. As shown in Fig.4.2, we sort nine pixel values in a  $3 \times 3$  window in ascending order as the sequence  $\{10, 15, 20, 20, 20, 20, 20, 25, 100\}$  and find out the median “20” to replace the original central pixel value. Then, it can exactly filter out the isolated high-brightness noise “100” and roughly keep the same for the brightness value around the central pixel.



**Fig. 4.2** The diagram of the action of a median filter.

Fig.4.3 shows the results of eight gray-scale images mixed with random noise through the action of a median filter.





**Fig.4.3** The results of eight images in Fig. 4.1 through the action of a median filter.

## 4.2 Accuracy Calculation

Because we take the variation of the total average accuracy for edge detection as the reference basis of the correction direction of parameters in our algorithm, we have to establish an accuracy formulation for providing the learning mechanism in advance.

First, we have to determine a ground truth map (GT) for every input image which depends on the gray-scale intensity difference between a pixel and its four neighbor pixels in this research in order to obtain the calculation of edge detection accuracy for convenience. If there is at least one of four neighbor pixels different with the central pixel, then we will label the coordinate corresponding to the location of this pixel as a black dot standing for edge in the ground truth map like the binary image introduced in Section 1.2.2.

Next, by scanning and comparing the ground truths and binary edge maps produced by our edge detection algorithm, we can define the following two representative accuracy formulations:

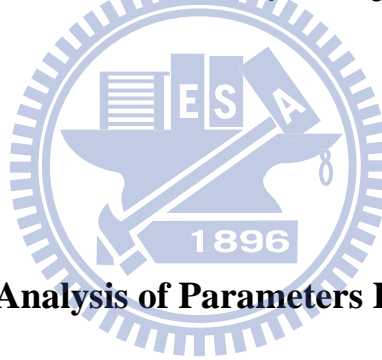
**A. Edge-weighted Accuracy:**

$$\left( 1 - \frac{r \left( \begin{array}{l} \text{The number of misjudged} \\ \text{edge pixels in GT} \end{array} \right) + \left( \begin{array}{l} \text{The number of misjudged} \\ \text{nonedge pixels in GT} \end{array} \right)}{r \left( \begin{array}{l} \text{The number of} \\ \text{edge pixels in GT} \end{array} \right) + \left( \begin{array}{l} \text{The number of} \\ \text{nonedge pixels in GT} \end{array} \right)} \right) \times 100\%$$

where “r” denotes an importance multiple for edge pixels in the ground truth map.

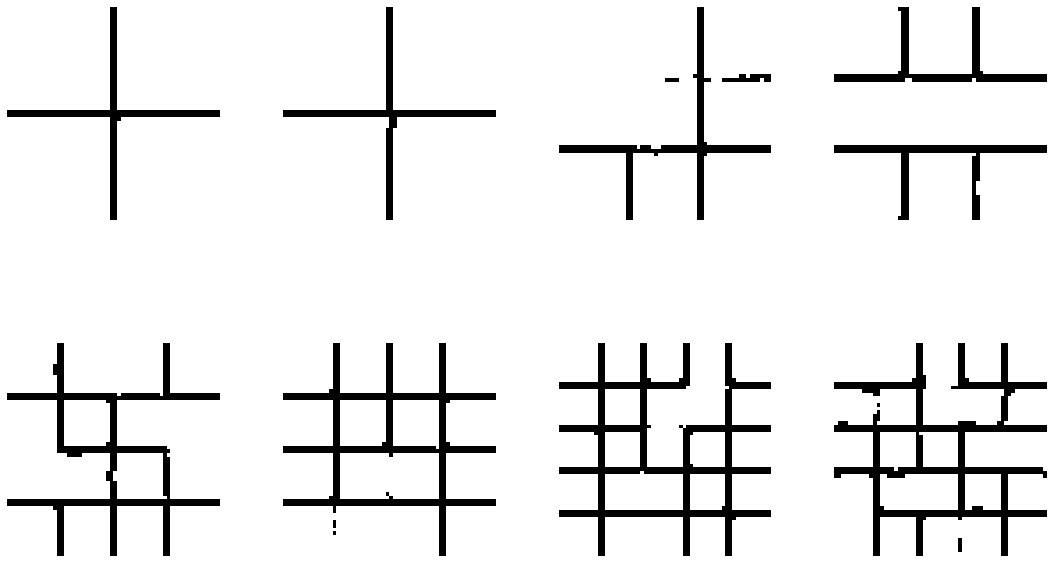
**B. The Average Accuracy of Edge-pixel and Nonedge-pixel:**

$$\left( \frac{\begin{array}{l} \text{The number of edge pixels} \\ \text{correctly detected in GT} \end{array}}{\begin{array}{l} \text{The number of edge pixels in GT} \end{array}} + \frac{\begin{array}{l} \text{The number of nonedge pixels} \\ \text{correctly detected in GT} \end{array}}{\begin{array}{l} \text{The number of nonedge pixels in GT} \end{array}} \right) \div 2 \times 100\%$$

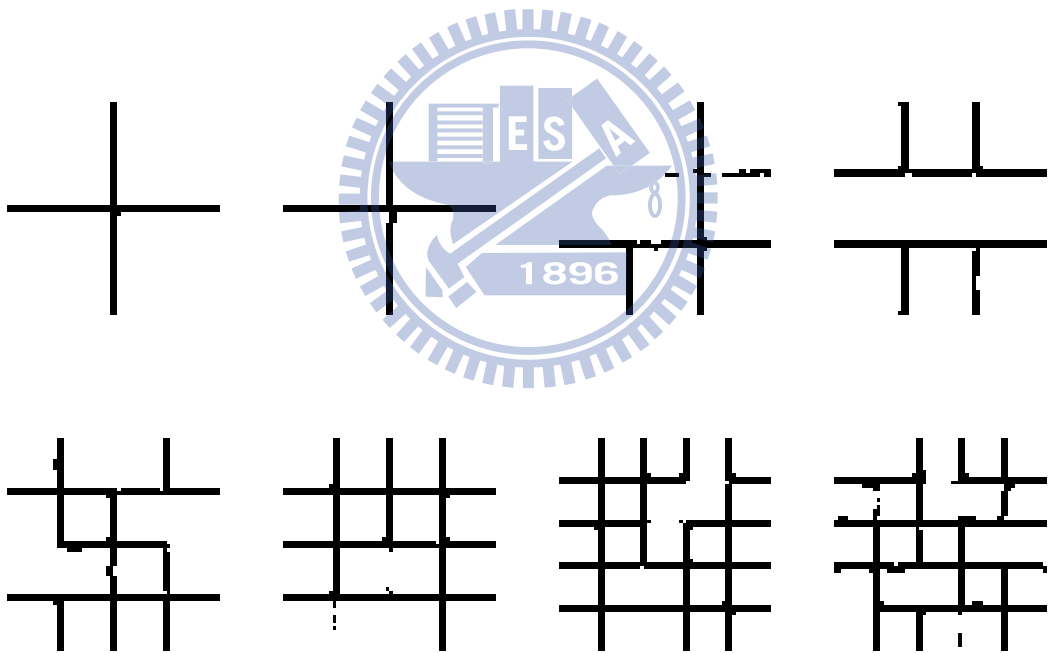


**4.3 The Results and Analysis of Parameters Learning**

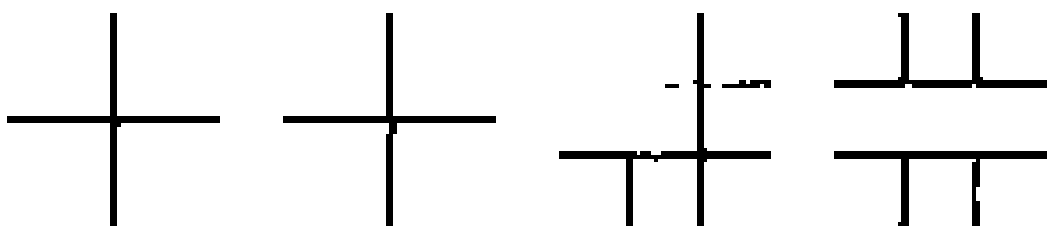
Fig.4.4-6 are the results of best edge detection for parameter learning with the linear weighted mean difference aggregation calculation of pixel values under the combinations of three different initial operating parameter sets and three kinds of accuracy formulations by using the eight gray-scale images mixed with random noise in Fig. 4.1; Fig. 4.7-9 are the results of best edge detection with the quadratic weighted mean difference formulation of pixel values. Table 4.1-2 and Table 4.4-5 show the results of best operating parameters and edge detection accuracy under nine different conditions respectively for Fig.4.4-6 and Fig.4.7-9. Besides, the ratio relationships between the number of edge-pixel and nonedge-pixel for eight original gray-scale images in Fig. 3.1 are recorded in Table 4.3.

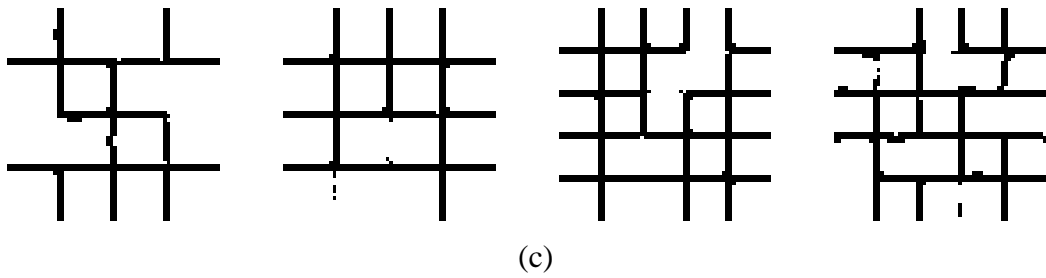


(a)

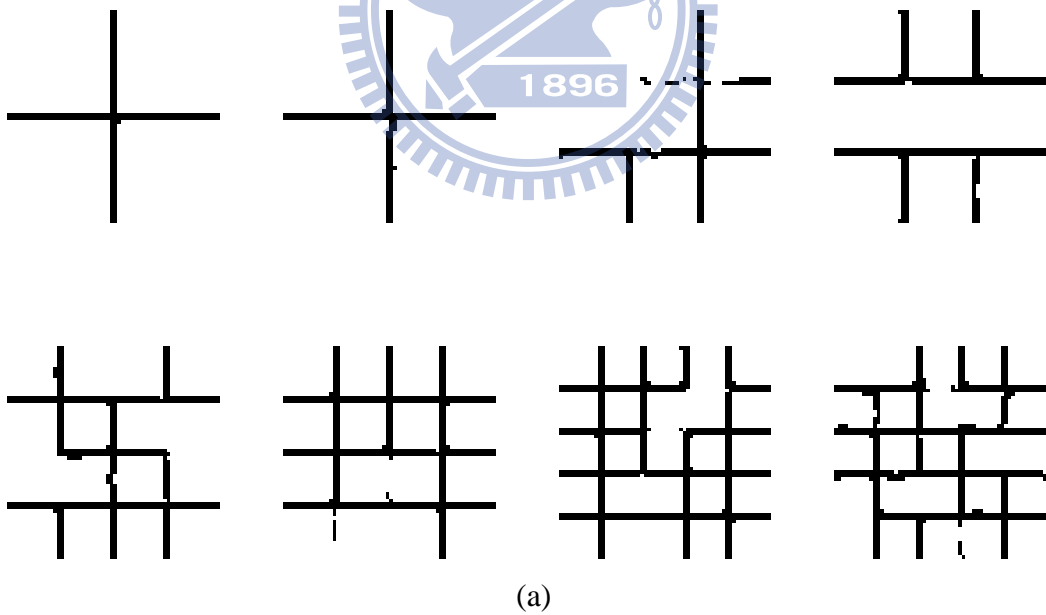


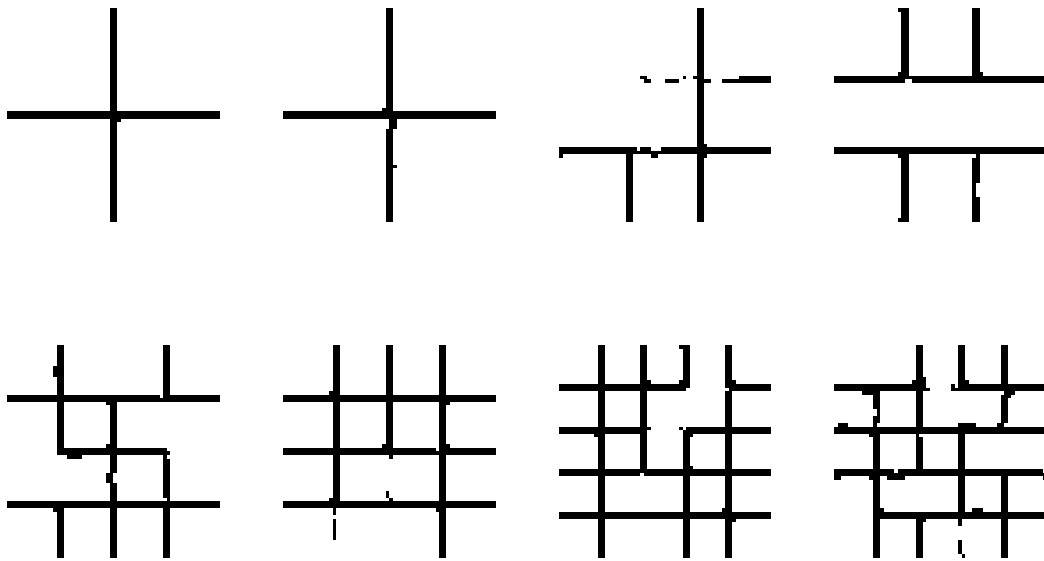
(b)



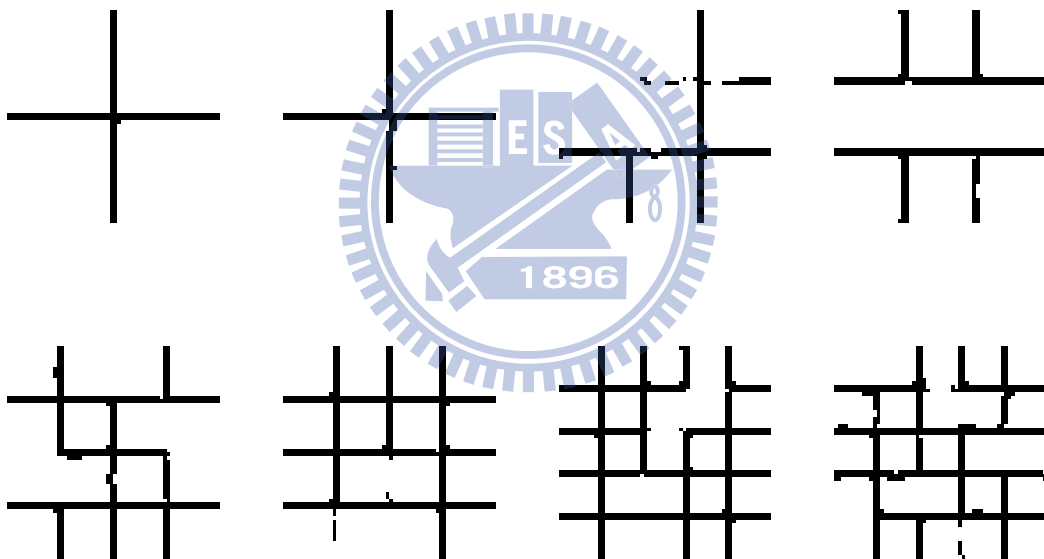


**Fig. 4.4** The results of best edge detection for parameters learning with the linear weighted mean difference algorithm of image pixel values (Initial values:  $\alpha_T=0.25$ ,  $\alpha_S=0.75$ ;  $\eta=0.88$ ,  $T=7$ ), (a) Correct based on the accuracy with edge-pixel weighted of multiple “1”; (b) Correct based on the accuracy with edge-pixel weighted of multiple “4”; (c) Correct based on the average accuracy of edge-pixel and nonedge-pixel.



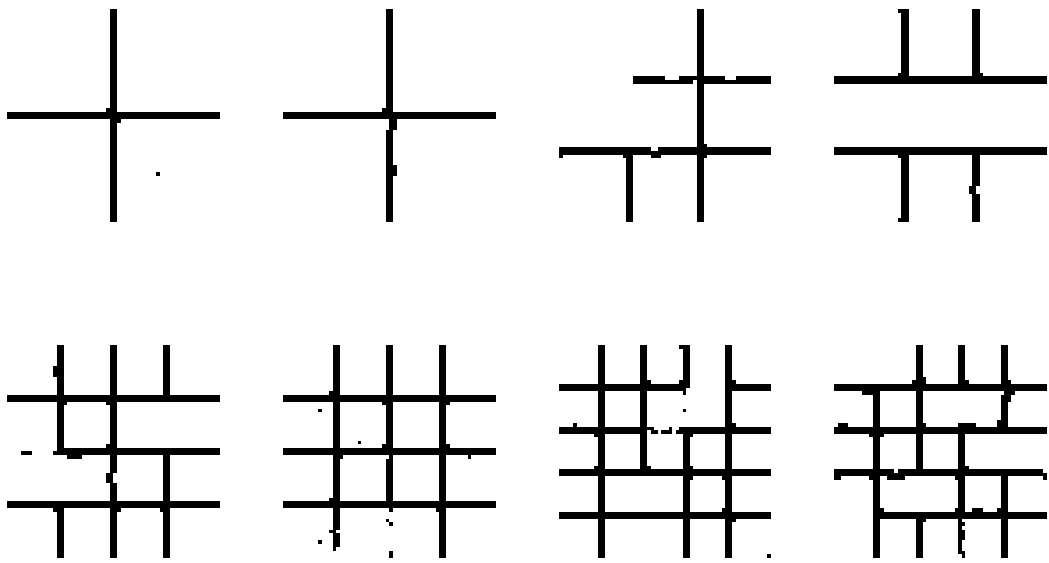


(b)

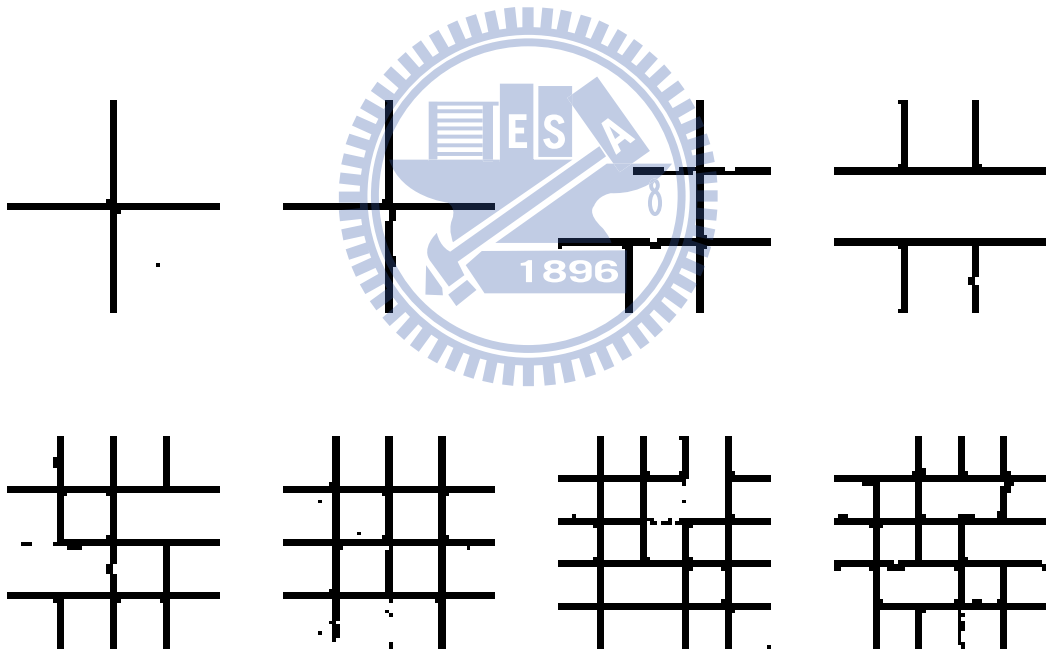


(c)

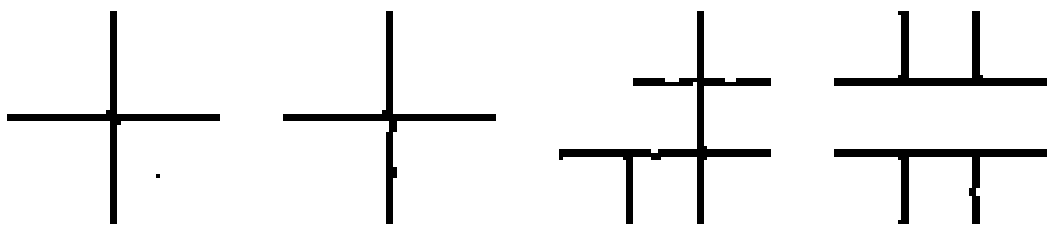
**Fig. 4.5** The results of best edge detection for parameters learning with the linear weighted mean difference algorithm of image pixel values (Initial values:  $\alpha_r=0.15$ ,  $\alpha_s=0.55$ ;  $\eta=0.7$ ,  $T=5$ ), (a) Correct based on the accuracy with edge-pixel weighted of multiple “1”; (b) Correct based on the accuracy with edge-pixel weighted of multiple “4”; (c) Correct based on the average accuracy of edge-pixel and nonedge-pixel.



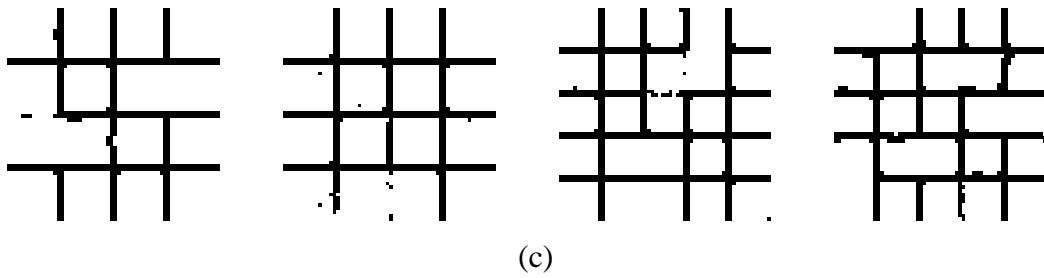
(a)



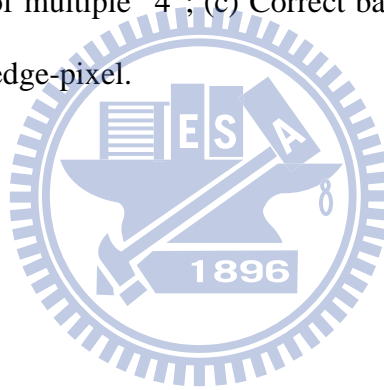
(b)







**Fig. 4.6** The results of best edge detection for parameters learning with the linear weighted mean difference algorithm of image pixel values (Initial values:  $\alpha_r=0.42$ ,  $\alpha_s=0.58$ ;  $\eta=0.8$ ,  $T=3$ ), (a) Correct based on the accuracy with edge-pixel weighted of multiple “1”; (b) Correct based on the accuracy with edge-pixel weighted of multiple “4”; (c) Correct based on the average accuracy of edge-pixel and nonedge-pixel.



**TABLE 4.1** The results of best operating parameters and average accuracy through 80 epoches with the linear weighted mean difference aggregation algorithm of pixel values for eight images in Fig. 4.1.

Initial Values	$\eta$	Accuracy Formulation	Best Average Accuracy(%) <sub>1</sub>	The Best Operating Parameters
$\alpha_T = 0.25$ $\alpha_S = 0.75$ (T=7)	0.88	A(r=1)	97.26 <sub>3</sub>	$\alpha_T = 0.1540, \alpha_S = 0.8460$
		A(r=4)	93.40	$\alpha_T = 0.1540, \alpha_S = 0.8460$
		B	92.68	$\alpha_T = 0.1540, \alpha_S = 0.8460$
$\alpha_T = 0.15$ $\alpha_S = 0.55$ (T=5)	0.7	A(r=1)	97.31 <sub>2</sub>	$\alpha_T = 0.0901, \alpha_S = 0.6099$
		A(r=4)	93.59	$\alpha_T = 0.0901, \alpha_S = 0.6099$
		B	94.66	$\alpha_T = 0.0901, \alpha_S = 0.6099$
$\alpha_T = 0.42$ $\alpha_S = 0.58$ (T=3)	0.8	A(r=1)	97.84 <sub>1</sub>	$\alpha_T = 0.2960, \alpha_S = 0.7040$
		A(r=4)	95.16	$\alpha_T = 0.2960, \alpha_S = 0.7040$
		B	94.59	$\alpha_T = 0.2960, \alpha_S = 0.7040$

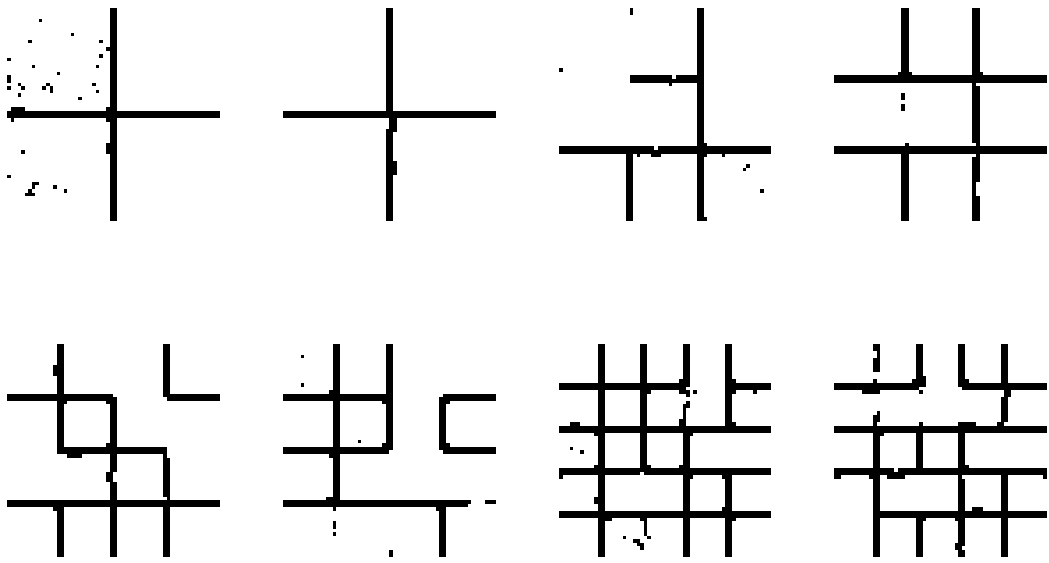
**Note1:** “Average Accuracy” means that the average of all individual accuracy of eight images with some kind of accuracy formulations.

**TABLE 4.2** The individual edge accuracy of eight images in Fig.4.1 with different operating parameters of TABLE 4.1.

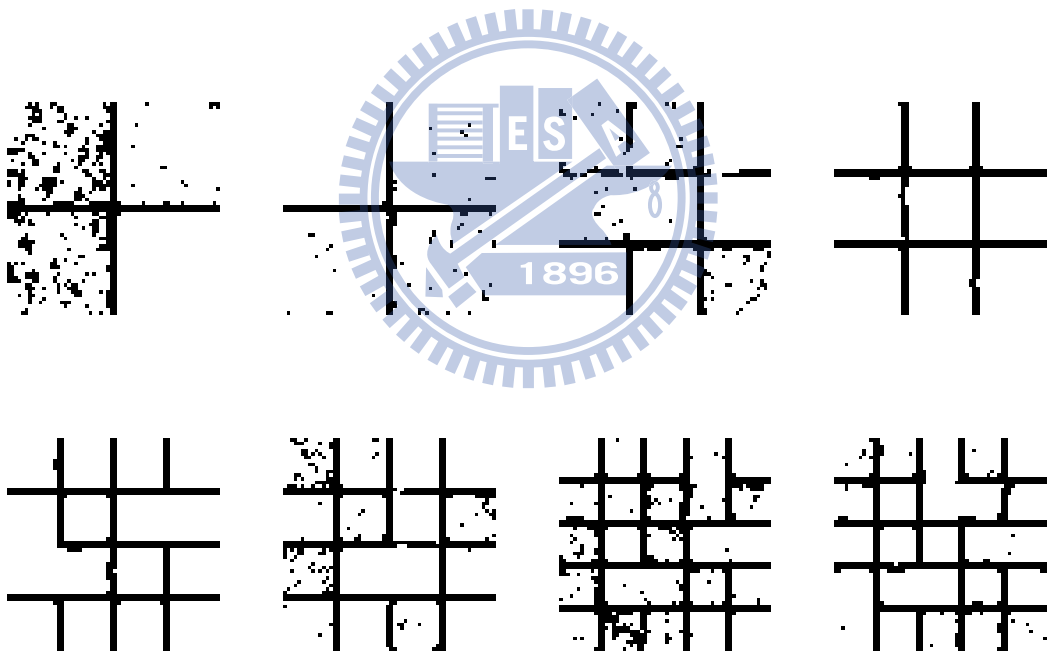
The image number in parameters Fig. 4.1 with three accuracy indices		(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
$\alpha_T = 0.1540$ $\alpha_S = 0.8460$ (T=7)	A(r=1)	99.97	99.83	95.31	97.67	95.64	97.78	96.89	94.97
	A(r=4)	99.98	99.65	86.64	93.51	89.58	94.60	93.26	89.97
	B	99.99	99.32	82.06	91.32	89.25	94.43	94.05	91.06
$\alpha_T = 0.0901$ $\alpha_S = 0.6099$ (T=5)	A(r=1)	99.97	99.78	95.44	97.67	95.67	97.86	96.86	95.25
	A(r=4)	99.98	99.61	87.16	93.57	89.65	94.87	93.24	90.65
	B	99.99	99.29	82.79	91.41	89.32	94.71	94.03	91.65
$\alpha_T = 0.2960$ $\alpha_S = 0.7040$ (T=3)	A(r=1)	99.92	99.72	96.31	97.75	97.36	98.81	97.06	95.78
	A(r=4)	99.93	99.56	89.76	93.93	94.13	97.65	94.02	92.29
	B	99.96	99.26	86.31	91.91	93.95	97.58	94.68	93.05

**TABLE 4.3** The number of edge-pixel and nonedge-pixel and the ratio relationship of them for ground truth edge maps in Fig. 3.1.

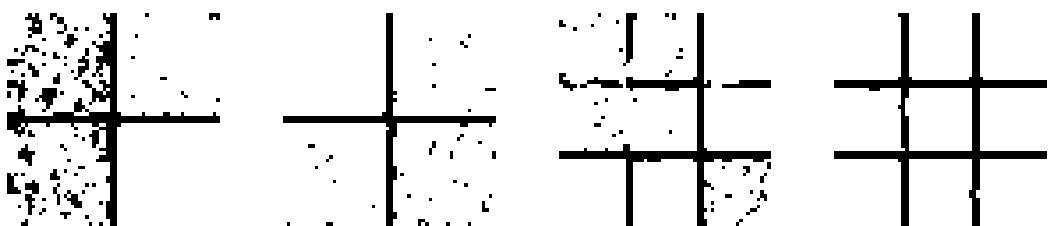
	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
The number of edge pixels	236	236	464	464	684	684	896	896
The number of nonedge pixels	3364	3364	3136	3136	2916	2916	2704	2704
The ratio of nonedge-pixel and edge-pixel	14.25	14.25	6.76	6.76	4.26	4.26	3.02	3.02

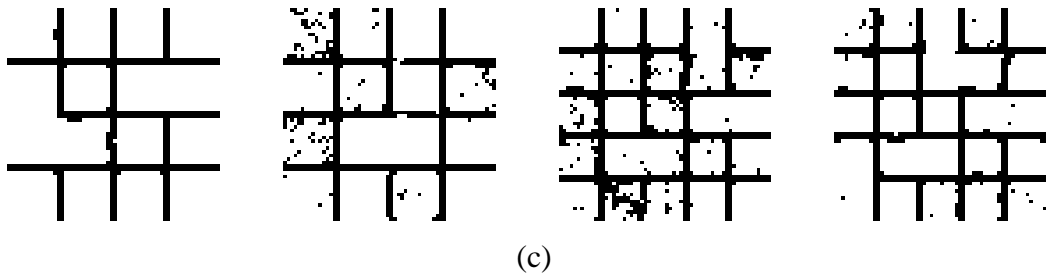


(a)

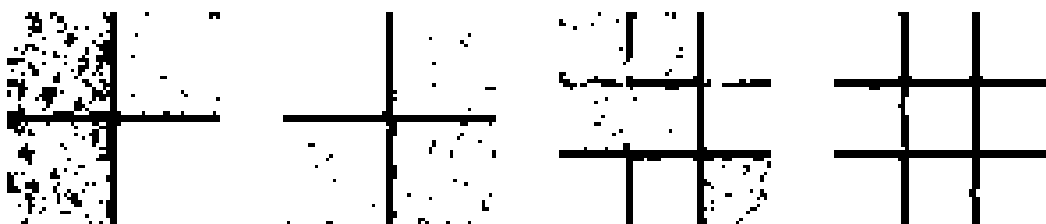
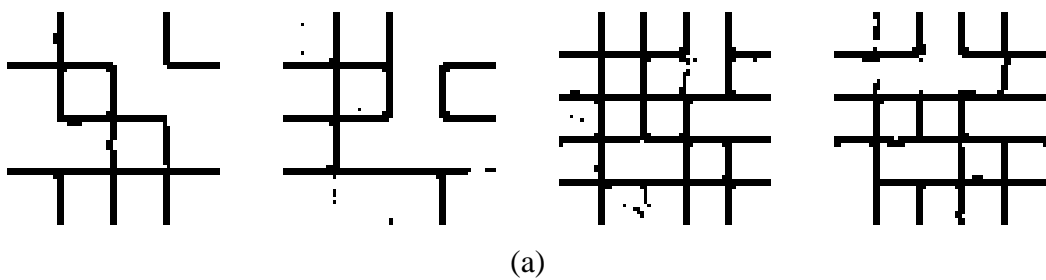
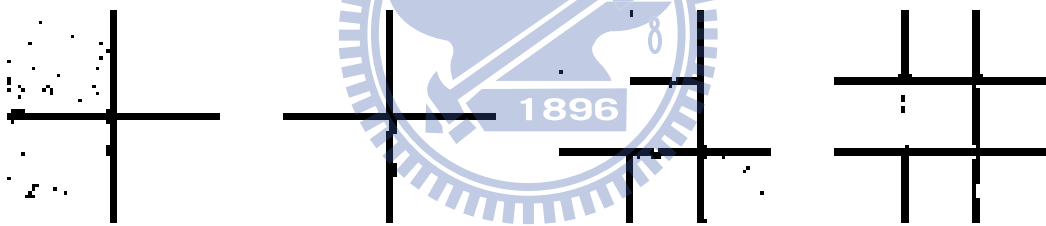


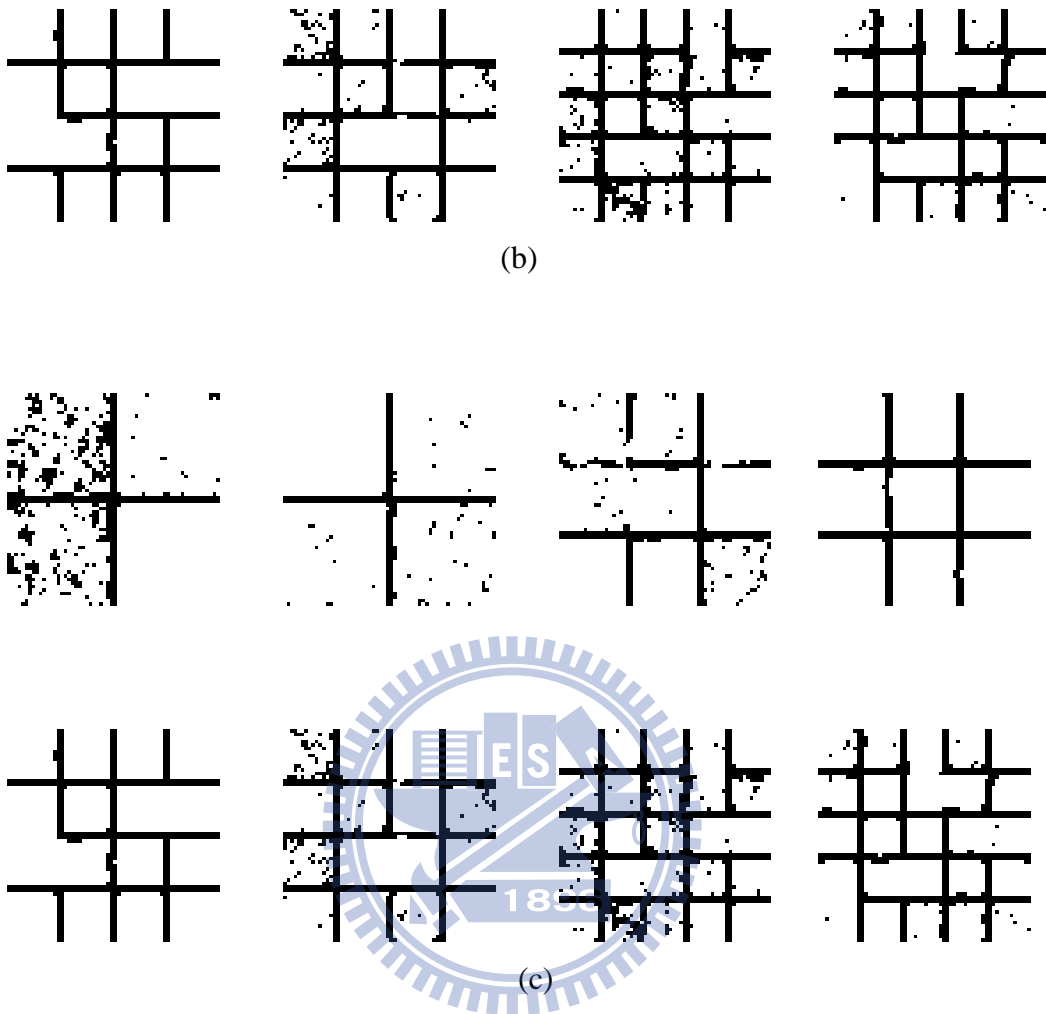
(b)



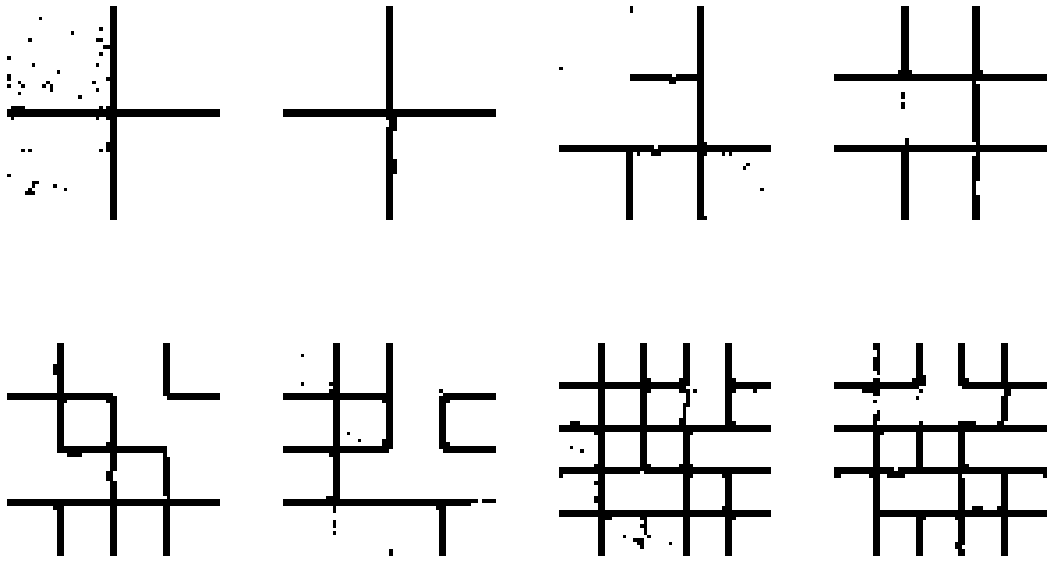


**Fig. 4.7** The results of best edge detection for parameters learning with the quadratic weighted mean difference algorithm of image pixel values (Initial value:  $\alpha_r=0.25$ ,  $\alpha_s=0.75$ ;  $\eta=0.001$ ,  $T=15$ ), (a) Correct based on the accuracy with edge-pixel weighted of multiple “1”; (b) Correct based on the accuracy with edge-pixel weighted of multiple “4”; (c) Correct based on the average accuracy of edge-pixel and nonedge-pixel.

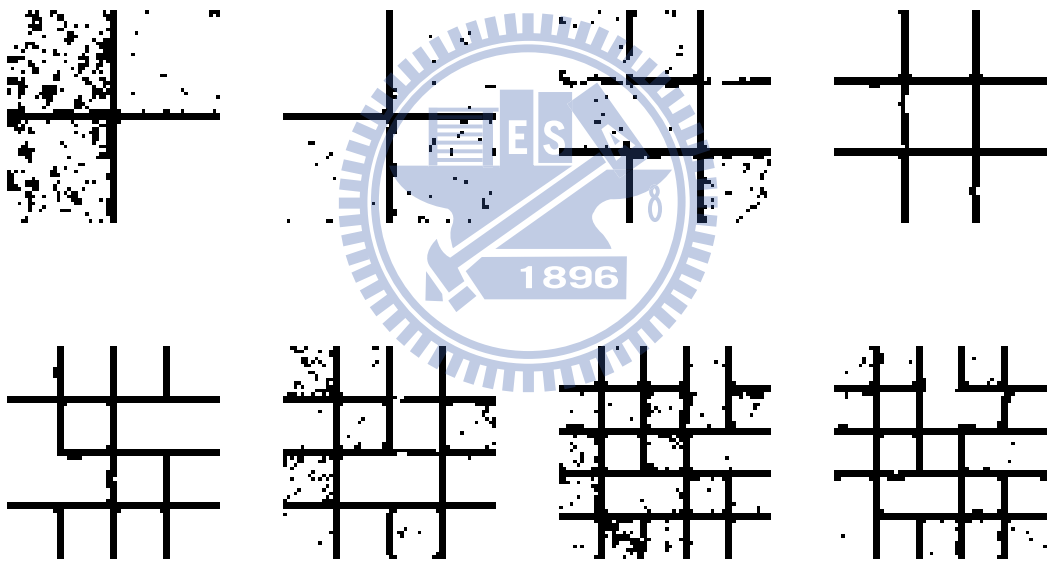




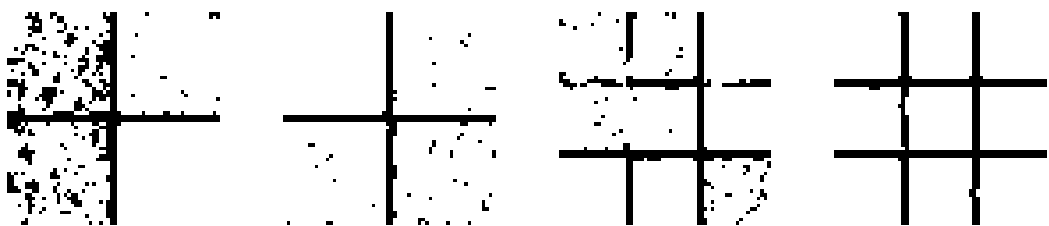
**Fig. 4.8** The results of best edge detection for parameters learning with the quadratic weighted mean difference algorithm of image pixel values (Initial values:  $\alpha_r = 0.15$ ,  $\alpha_s = 0.55$ ;  $\eta = 0.0005$ ,  $T=12$ ), (a) Correct based on the accuracy with edge-pixel weighted of multiple “1”; (b) Correct based on the accuracy with edge-pixel weighted of multiple “4”; (c) Correct based on the average accuracy of edge-pixel and nonedge-pixel.

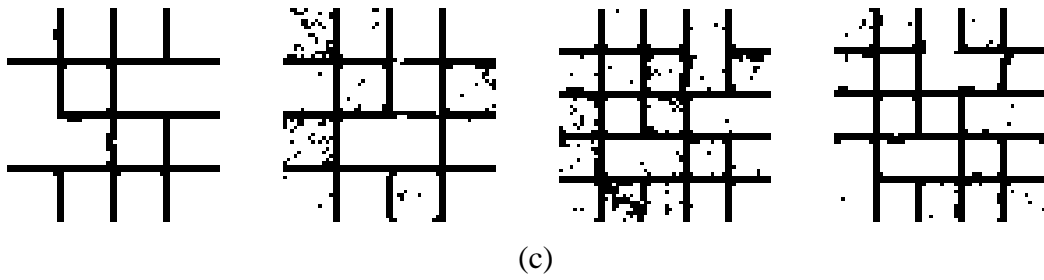


(a)

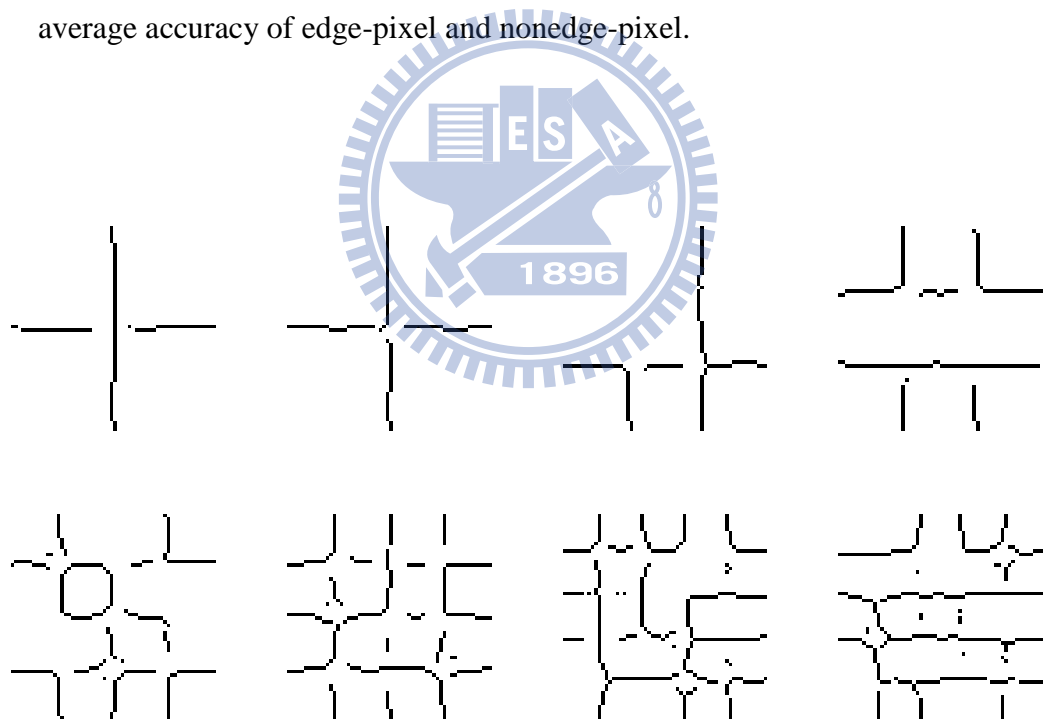


(b)





**Fig. 4.9** The results of best edge detection for parameters learning with the quadratic weighted mean difference algorithm of image pixel values (Initial values:  $\alpha_r=0.42$ ,  $\alpha_s=0.58$ ;  $\eta=0.0003$ ,  $T=10$ ), (a) Correct based on the accuracy with edge-pixel weighted of multiple “1”; (b) Correct based on the accuracy with edge-pixel weighted of multiple “4”; (c) Correct based on the average accuracy of edge-pixel and nonedge-pixel.



**Fig. 4.10** The results of Canny edge detector ( $\sigma=2$ ,  $T_{Low}=0.03$ ,  $T_{High}=0.15$ ) for eight images mixed with random noise in Fig. 4.1.



**TABLE 4.4** The results of best operating parameters and average accuracy through 80 epoches with the quadratic weighted mean difference aggregation algorithm of pixel values for eight images in Fig. 4.1.

Initial Values	$\eta$	Accuracy Calculation	Best Average Accuracy(%)	The Best Operating Parameters
$\alpha_T = 0.25$ $\alpha_S = 0.75$ (T=15)	0.001	A(r=1)	96.70	$\alpha_T = 0.5003, \alpha_S = 0.6159$
		A(r=4)	94.39	$\alpha_T = 0.4397, \alpha_S = 0.6489$
		B	94.69	$\alpha_T = 0.4397, \alpha_S = 0.6489$
$\alpha_T = 0.15$ $\alpha_S = 0.55$ (T=12)	0.0005	A(r=1)	96.70	$\alpha_T = 0.3620, \alpha_S = 0.4360$
		A(r=4)	94.38	$\alpha_T = 0.3234, \alpha_S = 0.4569$
		B	94.67	$\alpha_T = 0.3234, \alpha_S = 0.4569$
$\alpha_T = 0.42$ $\alpha_S = 0.58$ (T=10)	0.0003	A(r=1)	96.70	$\alpha_T = 0.4894, \alpha_S = 0.5423$
		A(r=4)	94.38	$\alpha_T = 0.4636, \alpha_S = 0.5564$
		B	94.67	$\alpha_T = 0.4636, \alpha_S = 0.5564$

**TABLE 4.5** The individual edge accuracy of eight images in Fig. 4.1 with different operating parameters sets of TABLE 4.4.

The image The best number in parameters Fig. 4.1 with three accuracy indices		(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
$\alpha_T = 0.5003$ $\alpha_S = 0.6159$ (T=15)	A(r=1)	98.78	99.69	95.42	98.86	94.86	94.19	96.69	95.14
$\alpha_T = 0.4397$ $\alpha_S = 0.6489$ (T=15)	A(r=4)	90.67	98.45	92.69	99.12	93.93	95.10	92.78	92.41
	B	94.02	98.85	91.19	98.95	93.75	95.09	92.69	92.98
$\alpha_T = 0.3620$ $\alpha_S = 0.4360$ (T=12)	A(r=1)	98.78	99.69	95.42	98.86	94.86	94.19	96.69	95.14
$\alpha_T = 0.3234$ $\alpha_S = 0.4569$ (T=12)	A(r=4)	90.69	98.47	92.53	99.12	93.93	95.10	92.81	92.41
	B	94.04	98.87	90.97	98.95	93.75	95.09	92.73	92.98
$\alpha_T = 0.4894$ $\alpha_S = 0.5423$ (T=10)	A(r=1)	98.56	99.72	95.36	98.89	94.86	94.28	96.67	95.28
$\alpha_T = 0.4636$ $\alpha_S = 0.5564$ (T=10)	A(r=4)	90.69	98.47	92.53	99.12	93.93	95.10	92.81	92.41
	B	94.04	98.87	90.97	98.95	93.75	95.09	92.73	92.98

From the experimental results of the linear type weighted mean aggregation algorithm for edge detection, we find that the boundary pixels between adjacent blocks having similar gray-scale intensity are hard to be detected out for an image. For example, as the fourth image in the first row of Fig 4.4 (b) shows, the boundaries of three connected blocks in the middle row are not detected successfully due to the gray-scale values for these blocks in the original image are 198, 208, 222 respectively very close. But on the other hand, as the datum shown in TABLE 4.1, the best operating parameters almost not be affected by the choose of edge accuracy indices for this kind of algorithm.

For the edge detection of quadratic type algorithm, it can enhance the difference of gray level between adjacent pixels and detect out more edge pixels. But in the meanwhile, it is more likely to magnify the noise points and lead to more misjudged edge pixels. And we can get from TABLE 4.4 that it is possible to reach the same effect in edge detection with different sets of initial operating parameters. Besides, from the experimental results, it also exhibits that median filter only has better suppressing effect for impulse type noise, but is useless for Gaussian noise.

In addition, due to the non-maximum suppression of thinning processing, the lines representing for edge are often one-pixel width in the binary edge map generated by the popular Canny edge detector as shown in Fig. 4.10. And the Canny edge detector even can suppress the noise of different strength effectively by its internal Gaussian filtering processing. Anyway, using our edge detection algorithm can better emphasize details and silhouettes of the objects for an image.

## 4.4 Edge Detection for Natural Images

For natural images, it is unable to determine a ground truth edge map for calculating the edge detection accuracy. Thus, we try to pick out two sets of operating parameters that perform better for edge detection in TABLE 4.1 and TABLE 4.4 to test the natural image “Lenna” by the corresponding edge detection algorithm and to obtain some better results for edge detection by appropriately adjusting the size of thresholds as shown in Fig. 4.11.

By comparing with the results of our edge detection algorithm and Canny edge detector, we hold that it will make the outlines portrayed deeper and show the details more clearly in an image by using our edge detection algorithm.



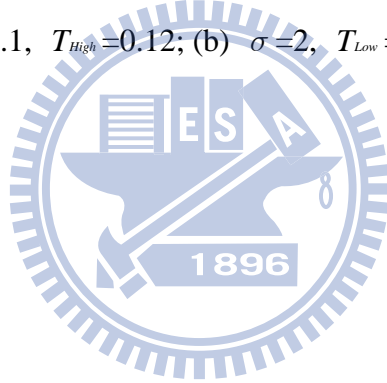
(a)



**Fig.4.11** (a) The natural image “Lenna”; (b)-(c) The results of linear weighted mean difference aggregation algorithm for edge detection with the best operating parameter sets ( $\alpha_T = 0.0901, \alpha_S = 0.6099$ ) and ( $\alpha_T = 0.2960, \alpha_S = 0.7040$ ) in TABLE 4.1, and the threshold T is equal to 3 and 2.5 respectively; (d)-(e) The results of quadratic weighted mean difference aggregation algorithm for edge detection with the best operating parameter sets ( $\alpha_T = 0.5003, \alpha_S = 0.6159$ ) and ( $\alpha_T = 0.4894, \alpha_S = 0.5423$ ) in TABLE 4.4, and the threshold T is equal to 13.5 and 9.5 respectively.



**Fig.4.12** The results of Canny edge detector[11] for natural image “Lenna”,  
(a)  $\sigma = 1.6$ ,  $T_{Low} = 0.1$ ,  $T_{High} = 0.12$ ; (b)  $\sigma = 2$ ,  $T_{Low} = 0.1$ ,  $T_{High} = 0.12$ .



## Chapter 5 Conclusion and Future Prospects

In this thesis, we integrate the technique based on the concept of interval-valued fuzzy relations to construct an fuzzy image that show the degree of intensity variation between neighbor pixels in a gray-scale image obviously, and apply the linear or quadratic type weighted mean difference aggregation calculation of pixel values between a central pixel and its eight neighbor pixels in a  $3 \times 3$  sliding window combined with a set of generalized operating parameters to implement the edge detection for eight synthetic gray-scale images mixed with different types and percentages of random noise. Besides, we further develop a parameter automatic learning mechanism by calculating the edge detection accuracy and the concept of steepest descent method, and test 80 iterations by the linear and quadratic type algorithms with three sets of different initial operating parameters and three kinds of edge detection accuracy indices. From the experimental results, although we test with different conditional setting, but it is possible to achieve same effect for edge detection. On the other side, despite it can better highlight the boundaries between adjacent regions of similar gray-scale pixels with the quadratic type aggregation calculation of image pixel values, but relatively, it also cause some noise pixels magnified and misjudged as edge pixels. So we have to filter out a few noise with tremendous intensity variation by a median filter in advance to make the response of edge detection more robust.

At last, we also choose some pairs of best operating parameters in the application of edge detection for the natural “Lenna” image. By the comparison of our edge detection method and the popular Canny edge detector, we find that it can better emphasize the object details and show the silhouettes strongly by using our edge

detection method. Out of the promotion in the edge detection capability, the efficiency of performance for some high-level tasks of image processing such as image segmentation or object recognition would also escalate.

In the future, we intend to establish an automatic adjustment system of thresholds and learning ratio constants in the parameter learning mechanism according to the image contents in order to prevent the unreliability and inadaptability for setting by testers themselves.





## References

- [1] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, 1986.
- [2] 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 Systems.*, vol 19, no. 5, October 2011.
- [3] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Ed., Prentice Hall, New Jersey, 2002.
- [4] H. Bustince and P. Burillo, "Structures on intuitionistic fuzzy relations," *Fuzzy Sets Syst.*, vol. 78, pp. 293–303, 1996.
- [5] 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.
- [6] H. Bustince, E. Barrenechea, M. Pagola, and R. Orduna, "Construction of interval type 2 fuzzy images to represent images in grayscale: False edges," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, London, U.K., 2007, pp. 73–78.
- [7] H. Bustince, E. Barrenechea, and M. Pagola, "Generation of interval-valued fuzzy and Atanassov's intuitionistic fuzzy connectives from fuzzy connectives and from  $K_\alpha$  operators: Laws for conjunctions and disjunctions, amplitude," *Int. J. Intell. Syst.*, vol. 23, pp. 680–714, 2008.
- [8] H. Bustince, E. Barrenechea, M. Pagola, and J. Fernandez, "Interval-valued fuzzy sets constructed from matrices: Application to edge detection," *Fuzzy Sets Syst.*, vol. 160, pp. 1819–1840, 2009.
- [9] J. Barzilai and J.M. Borwein, "Two point step size gradient methods," *IMA J.*

*Numer. Anal.*, vol. 8, pp. 141–148, 1988.

- [10] M. E. Yüksel 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.
- [11] M.D. Heath, S. Sarkar, T. Sanocki, and K.W. Bowyer, “A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 12, pp. 1338-1359, Dec. 1997.

