International Journal of Applied Science and Engineering
2011. 9, 3: 143-160

Salt and Pepper Noise Reduction by Cellular Automata

Chih-Yu Hsu [a], Ta-Shan Tsui[b,c], Shyr-Shen Yu[b,*], and Kuo-Kun Tseng[d]

[a] *Department of Information & Communication Engineering, Chaoyang University of Technology, Taichung, Taiwan, R.O.C.*
[b] *Department of Computer Science and Engineering, National Chung Hsing University, Taichung, Taiwan, R.O.C.*
[c] *Department of Applied Mathematics, National Chung Hsing University, Taichung, Taiwan, R.O.C.*
[d] *Department of Computer Science and Technology, Shenzhen Graduate School, Harbin Institute of Technology. HIT Campus of ShenZhen University Town, XiLi, ShenZhen, China*

Abstract: An algorithm and software based on the concept of cellular automata was developed for removing salt and pepper noise efficiently. The paper shows the software programming for developing an algorithm and software called Cellular Automata Image Denoising (CAID) toolkit. This paper presents the CAID toolkit using examples and discusses how the CAID toolkit is designed. Matlab code was used to develop a software program for removing salt and pepper noise in gray and color images. We hope the code will help the researchers who are interested in the Image Denoising for research of image processing.

Keywords: Cellular automata; salt and pepper noise; software algorithm; Matlab Code.

1. Introduction

In this paper, a method based on cellular automata and string matching rules is proposed to remove Salt and Pepper noise efficiently. In order to make our research reproducible [1], we are trying to not only describe the developed algorithms to "sufficient" precision in a paper, but also to give readers access to all the information (e.g., code, data, schemes) for their researches [2-3]. In the words of Buckheit and Donoho [4]: "The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.". We try to share the details of the algorithms of Salt and Pepper Noise Reduction and the Programming codes. The Salt and Pepper Noise Reduction Programming has been developed in a Matlab language for our algorithms. The developed algorithms and their associated programs are known as the CAID toolkit. Our goal is to describe the concept that underlies CAID toolkit and to present examples of programs in the CAID toolkit. A complicated system can be understood by understanding its parts, and by understanding the relations between those parts and their immediate neighbors. For the purpose of clarity to show our algorithms, we only discuss the gray level image now.

Image processing applies filters with convolution operations on the images. Filters are designed as specific blocks and are used as masks for convolution operations. Using cellular automaton for image processing

* Corresponding author; e-mail: pyu@nchu.edu.tw
Accepted for Publication: May, 1, 2011

should involve the relation between the automaton and image processing. A digital image with size $m \times n$ is a bi-dimensional array with that each element called a pixel and with a gray level or color value. Each pixel can be characterized by the triplet (i, j, k) where (i, j) represents its position in the array and k represents gray level or color. The image can be then considered as a particular configuration state of a cellular automaton that has as cellular space on the array defined by the image. Each site in the array corresponds to a pixel. Cellular automaton growth is used for image processing [5-7] and is controlled by predefined rules, usually controlled by program, or by some grammar rules.

Cellular automata (CA) [8, 9] consist of a regular grid of cells (pixels) and each cell (pixel) can be in only one of a finite number of possible states. For a gray level image, each cell (pixel) has 256 states that are composed of integer numbers from 0 to 255. The state of a cell (pixel) is determined by the previous states of a surrounding neighborhood of cells (pixels) and is updated synchronously in discrete time steps. The identical rules contained in each cell is essentially a finite state machine, usually specified in the form of grammar rules with an entry for every possible neighborhood configuration of states. Rosin[8] used the sequential floating forward search (SFFS) method [10] and made an experiment on filtering to overcome salt and pepper noise with RMS error criterion between the denoised and original image. However, the experiment is difficult to reproduce because the author did not provide source codes and data. The goal of the paper is to use examples to explain how to implement cellular automata to remove salt and pepper noise. Our algorithms only compared with the median filter because Matlab has the functions of median filter.

2. The CAID Toolkit

The CAID toolkit is developed to remove Salt and Pepper noise efficiently and it is based on string matching with cellular automata rules. Cellular Automata based on string matching rules to remove Salt and Pepper noise efficiently is explained in this section. Typical median filtering set the value of an output pixel to be the median of the neighborhood pixels. A block for masking the neighborhood pixels is used as a moving window to screen the whole image. Sorting algorithm should be run on the neighborhood pixels in the masked area before finding the median of the neighborhood pixels. The median value is used to replace the center pixel of the window. Because of no detection of the noise, the false pixels may be generated by some situations. Our proposed algorithm use Cellular Automata based on string matching rules to identify the pixels of Salt and Pepper noise, then the median filter is used only on these identified pixels. Besides, noisy pixels can be captured by string keywords and replaced by the value, which is the median of the neighborhood pixels.

2.1. Synthetic image and noise

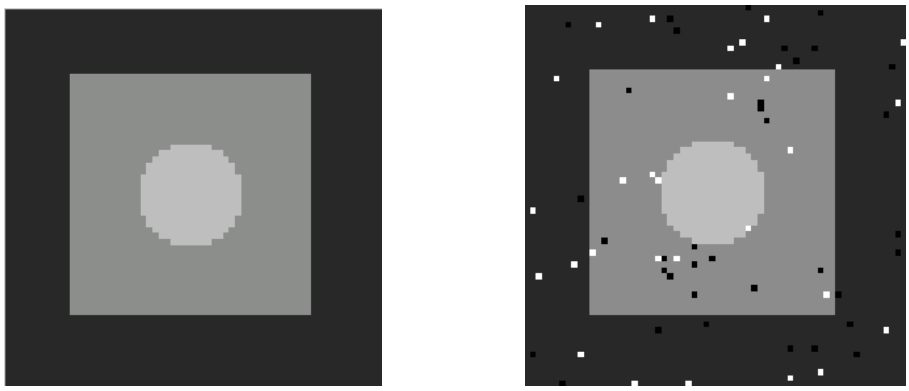
A synthetic grayscale image is crated by Matlab codes. The following codes are used to display a synthetic grayscale image as shown in Figure 1(a) and the algorithm is described as follows:

1. A square area is generated with the size 64x64 and each pixel has gray level value 100.
2. Another square block area is built in the previous square area with the size 41x41 and each pixel has gray level value 140.
3. A circle is built with a radius 8 and it has the center point at middle of the squares.

The file name of the codes that implemented the above algorithms is Synthetic image.m.

An image with Salt and Pepper Noise is displayed in Figure 1(b) that is generated by

the Matlab function “imnoise()” with a parameter (d), which is the noise density. All numerical parameters are normalized and they correspond to operations with images with intensities ranging from 0 to 1. This function affects some pixels in the image and their function value is approximately equal to values of elements in the image array plus the parameter (d). Our algorithm, which has a noise detector, is different from the classical median filter. The pixels with the greatest gray level (255) and the lowest gray level (0) represent the salt and pepper, respectively. Thresholding method is used to separate the noise from the objects and background in the image. The binary image is obtained by a threshold.



(a) (b)

Figure 1. (a) a synthetic grayscale image and (b) with Salt and Pepper Noise(d=0.02).

(a) (b) (c)

Figure 2. (a) a binary image obtained by labeling white pixels that whose gray level is greater than 250, (b) a binary image obtained by labeling white pixels that whose gray level is lower than 10 and (c) a binary image obtained by is the union set of the noise in (a) and (b).

.

In Figure 2(a) and Figure 2(b), threshold 250 is used to select the pixels of Salt Noise and threshold 10 is used to select the pixels of Pepper Noise. These thresholds are flexible to be chosen for the separation of the Salt and Pepper Noise from the other objects in the image. All the pixels in Figure 2(a) and Figure 2(b) are noise to be removed and the union of the noise pixels is shown in Figure 2(c).

In the next subsection, we are going to introduce the classical median filter that will be used to compare the results by our developed algorithms.

2.2. Noise Filters

The classical noise filters use a two dimensional mask matrix to compute convolutions on an input image. Considering a filter matrix F with dimensions (M_F, N_F) and an image matrix I with dimension (M_I, N_I), the equation of the 2-D discrete convolution is described as the following.

$$\sum_{m=0}^{M_F}\sum_{n=0}^{N_F} F(m,n)\times I(i-m, j-n) \qquad (1)$$

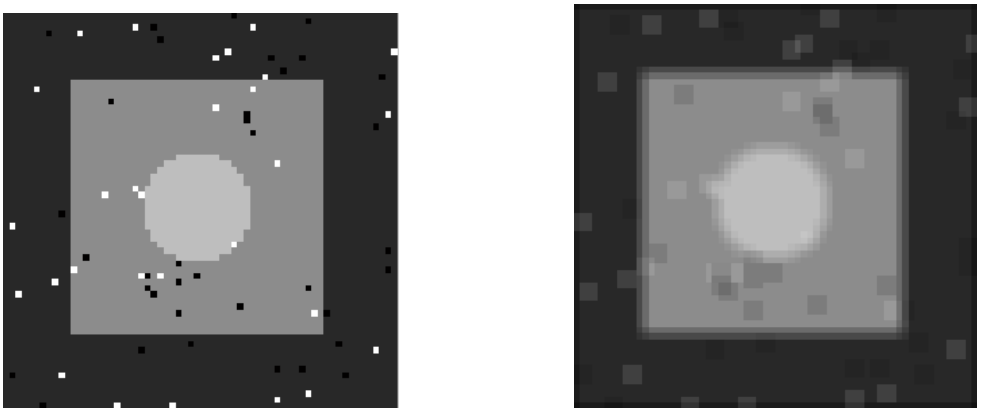
Where a two dimensional matrix I represents an image and $I(i, j)$ is the gray level at the pixel whose location is indexed by the symbol (i, j) and constrained by $0 \le i \le M_I + M_F - 1$ and $0 \le j \le N_I + N_F - 1$. An example of a filter matrix F with sizes 3x3 is shown as the following.

$$F = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) \\ f(1,0) & f(1,1) & f(0,2) \\ f(2,0) & f(2,1) & f(0,2) \end{bmatrix} \qquad (2)$$

For an averaging filter with size 3x3, the filter is matrix represented as the following.

$$F_{avg} = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} \qquad (3)$$

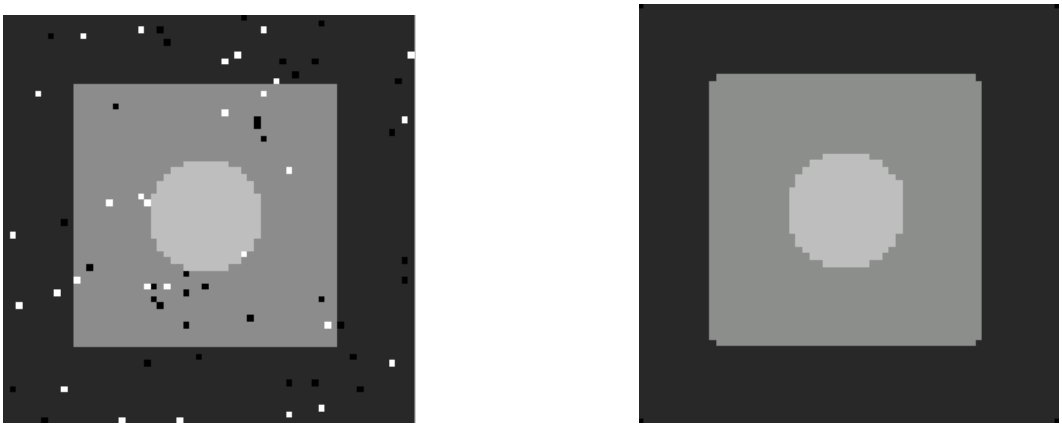
The image with Salt and Pepper Noise as shown in Figure 3(a) is denoised by the averaging filter as the equation (3). The result image is blurred as shown in Figure 3(b) and the noise is not removed.



(a) (b)

Figure 3. (a) an image with Salt and Pepper Noise and (b) the denoised image by averaging filter.

Median filter is the median value of an array that is use as a mask. As described in Matlab help function, Median filtering is a nonlinear operation often used in image processing to reduce "salt and pepper" noise. A median filter is more effective than convolution when the goal is to reduce noise and to preserve edges simultaneously. We used the function medfilt2() to perform median filtering of the image matrix in two dimensions. Each output pixel contains the median value in the 3-by-3 neighborhood around the corresponding pixel in the input image. The function medfilt2() pads the image with 0's on the edges, so the median values for the points on the corners of the edges might appear distorted.



(a) (b)

Figure 4. (a) an image with Salt and Pepper Noise and (b) the denoised image by median filter.

The image with Salt and Pepper Noise as shown in Figure 4(a) is denoised by the median filter and the result image is shown in Figure 4(b). The noise is removed but the color of the four corners is changed. Our algorithm is developed to overcome the drawback of the median filter by cellular automata represented with some rules to detect the noised pixels.

2.3. Cellular Automata

An image with Salt and Pepper Noise has pixels with the greatest gray level (255) and the lowest gray level (0). The thresholding method is used to separate the noise from the objects and background as shown in Figure 2(c).

We use a mask with the size 4x4 and some keywords to detect the Salt and Pepper Noise. A mask is as shown in Table 1 and it is reshaped to a one dimensional array in Table 2. The mask is first located at the left top of the image. The cell with the number 0 represents the pixel at (0, 0) and the cell with the number 16 represents with the pixel at (3,3). The route of the moving mask is arranged from the left column to the right and then it is moved from the first column to the bottom row. The mask screens the whole image until the mask reaches the bottom of the image.

Table 1. A mask is indexed numbers for each cell in a square block.

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

Table 2. A mask is indexed numbers and reshaped to a one dimensional array.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

There are some rules to detect noise by the mask. For every cell at the boundary of the mask in Table 1 can not be a noise. The cell is labeled as class zero if it is not a noise and label class one if it is a noise. We represent the class zero by gray color and class one by white color as shown in Table 3.

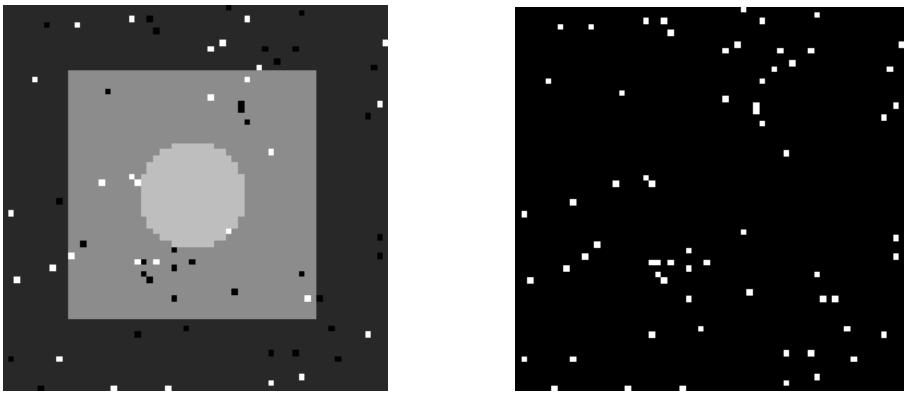
Table 3. A mask is labeled with class zero by gray color and class one by white color.

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

In Table 3, the mask has four pixels that are influenced by noises. We use a binary string matrix [0000011001100000] to represent the case as in the table 4. Consider the two strings str1 and str2, we use Matlab function strcmp() returns logical 0 (false) if the strings str1 and str2 are not identical. If the strings str1 and str2 are identical, the function returns logical 1 (true). We replace the gray level of the pixels 6,7,10,11 by the median of the values of the other pixels numbered.

Table 4. The mask in Table 3 is reshaped to one dimensional array.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16



(a) (b)

Figure 5. (a) the denoised image by mask that is the string matrix [0000011001100000], and (b) the binary image with noise pixels that were not removed.

Comparing the images in Figure 2(c) and Figure 5(b), the number of noise is not reduced very much because there are few isolated pixel groups that are composed of four pixels influenced by noises. We consider four masks in Table 5 to remove these isolated pixels that are influenced by noises.

There are four cases Table 5(a), (b), (c) and (d) with masks having three white color cells.
Table 5. Four masks having three white cells are labeled with class zero by gray color and class one by white color.

In Table 5, the four corresponding binary string matrixes are [0000001001100000], [0000011000100000], [0000010001100000], and [0000011001000000] each of which represents each case.

Table 5. Four masks having three white cells are labeled with class zero by gray color and class one by white color.

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(a)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

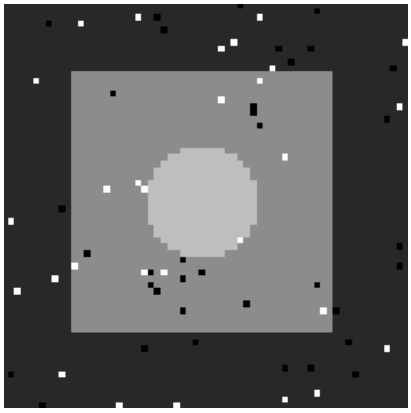
(b)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

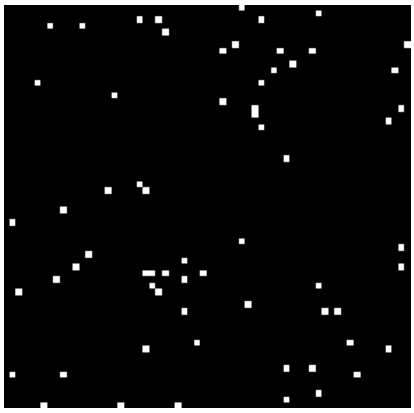
(c)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(d)



(a)



(b)

Figure 6. (a) the denoised image by mask that are the string matrixes in Table 3 and Table 5 and (b) the binary image with noise pixels that were not removed.

Comparing the images in Figure 6(b) and Figure 5(b), the number of noise is not reduced very much because there are few isolated pixel groups that are composed of three pixels influenced by noises. We consider six masks in Table 6 to remove those isolated pixel groups that are composed of two pixels influenced by noises. There are six cases as (a), (b), (c), (d), (e) and (f) with masks having two white cells in Table 6.

Table 6. Six masks having two white cells are labeled with class zero by gray color and class one by white color.

The six corresponding binary string matrixes are [0000010001000000], [00000000011- 00000], [0000001000100000], [0000011- 000000000], [0000010000100000], and [0000001001000000], each of which represents each case.

Comparing the images in Figure 7(b) and

Figure 6(b), the number of noise is not reduced very much because there are few isolated pixel groups that are composed of two pixels influenced by noises. We consider four masks in Table 7 to remove those isolated pixel groups that are composed of one pixel influenced by noise. There are four cases (a), (b), (c) and (d) with masks having only a white cell color in Table 7.

The four corresponding binary string matrixes are [0000010000000000], [0000001-000000000] [0000000001000000], and [0000000000100000] each of which represents each case.

Table 6. Six masks having two white cells are labeled with class zero by gray color and class one by white color.

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(a)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(b)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(c)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

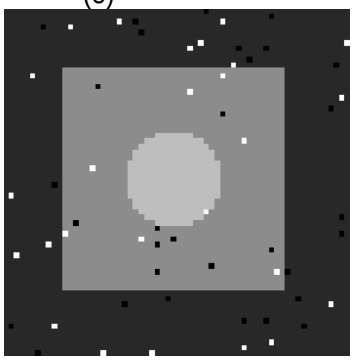
(d)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

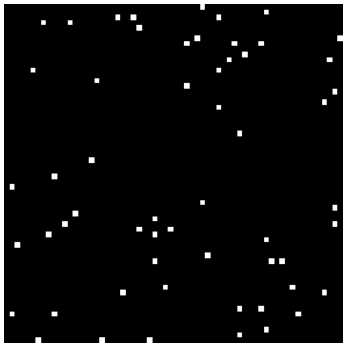
(e)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(f)



(a)



(b)

Figure 7. (a) the denoised image by masks that are the string matrixes in Table 3, Table 5 and Table 6 and (b) the binary image with noise pixels that were not removed.

Table 7. Four masks having one cell are labeled with class zero by gray color and class one by white color.

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(a)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(b)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(c)

1	5	9	13
2	6	10	14
3	7	11	15
4	8	12	16

(d)

The four corresponding binary string matrixes are [0000010000000000], [0000001000000000] [0000000001000000], and [0000000000100000] each of which represents each case.

(a) (b)

Figure 8. (a) the denoised image by mask comparing with the string matrixes in Table 3, Table 5, Table 6 and Table 7, and (b) the binary image with noise pixels that were not removed.

Comparing Figure 8(a) and Figure 8(b), a lot of pixel groups with one isolated pixel in a neighbourhood were removed, but the gray levels of two pixels at the boundary of the inner square were not correctly changed. Because the replacement by the median value of gray levels of surrounding pixels excludes itself, the error for selection of gray level occurs at the boundary of the image. To solve the problem, we should consider adding some rule for boundary detection and the modified rules are as following.

"When the noise pixels occurs at boundary, the selection of gray level for the noise replacement has to consider more than the general noise pixels."

For example, we consider Table 7(a) to implement the above rule as the following.

"**If** gray levels of cells (1, 2, 3) are the same value ($GI(1) = GI(2) = GI(3)$), there exists a boundary if the gray level $GI(1)$ of

the cell (1) is not equal to the gray level $GI(5)$ of the cell (5). The value of gray level could be replaced by the median of gray level of the pixels at cells (5, 7, 9, 10 and 11).
Else if gray levels of cells (1, 5, 9) are the same value ($GI(1)=GI(5)=GI(9)$), there exists a boundary if the gray level $GI(1)$ of the cell (1) is not equal to the gray level $GI(2)$ of the cell (2).The value of gray level could be replaced by the median of gray level of the pixels at cells (2, 3, 7, 10 and 11).
Else if gray levels of cells (9, 10, 11) are the same value ($GI(9)=GI(10)=GI(11)$), there exists a boundary if the gray level $GI(9)$ of the cell (11) is not equal to the gray level $GI(5)$ of the cell (5).The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 5 and 7).
Else if gray levels of cells (3, 7, 11) are the same value ($GI(3)=GI(7)=GI(11)$), there exists a boundary if the gray level $GI(3)$ of the cell (3) is not equal to the gray level $GI(2)$ of the cell (2).The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 5, 9 and 10).
else

The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 7, 8, 9, 11, 10, 12, 13, 14, 15 and 16).
end"

For the Table 7(b), the rules are statements as the following.
"**If** gray levels of cells (2, 3, 4) are the same value ($GI(2)=GI(3)=GI(4)$), there exists a boundary if the gray level $GI(2)$ of the cell (2) is not equal to te gray level $GI(6)$ of the cell (6).The value of gray level could be replaced by the median of gray level of the pixels at cells (6, 8, 10, 11 and 12).
Else if gray levels of cells (2, 6, 10) are the same value ($GI(2)=GI(6)=GI(10)$), there exists a boundary if he gray level $GI(2)$ of the cell (2) is not equal to the gray level $GI(3)$ of the cell (3). The value of gray level could be replaced by the median of gray level of the pixels at cells (3, 4, 8, 11 and 12).
Else if gray levels of cells (10, 11, 12) are the same value ($GI(10)=GI(11)=GI(12)$), there exists a boundary if the gray level $GI(10)$ of the cell (10) is not equal to the gray level $GI(6)$ of the cell (6). The value of gray level could be replaced by the median of gray level of the pixels at cells (2, 3, 4, 6 and 8).
Else if gray levels of cells (4, 8, 12) are the same value ($GI(4)=GI(8)=GI(12)$), there exists a boundary if the gray level $GI(4)$ of the cell (4) is not equal to the gray level $GI(3)$ of the cell (3). The value of gray level could be replaced by the median of gray level of the pixels at cells (2, 3, 6, 10 and 11).
else

The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 6, 8, 9, 11, 10, 12, 13, 14, 15 and 16).
end"

For the Table 7(c), the rules are statements as the following.
"**If** gray levels of cells (5, 6, 7) are the same value ($GI(5)=GI(6)=GI(7)$), there exists a boundary if he gray level $GI(5)$ of the cell (5) is not equal to the gray level $GI(9)$ of the cell (9).The value of gray level could be replaced by the median of gray level of the pixels at cells (9, 11, 13, 14 and 15).
Else if gray levels of cells (5, 9, 13) are the same value ($GI(5)=GI(9)=GI(13)$), there

exists a boundary if the gray level $GI(5)$ of the cell (5) is not equal to the gray level $GI(6)$ of the cell (6).The value of gray level for replacement could be replaced by the median of gray level of the pixels at cells (6, 7, 11, 14 and 15).

Else if gray levels of cells (13, 14, 15) are the same value ($GI(13) = GI(14) = GI(15)$), there exists a boundary if the gray level $GI(13)$ of the cell (13) is not equal to the gray level $GI(9)$ of the cell (9).The value of gray level could be replaced by the median of gray level of the pixels at cells (5, 6, 7, 9 and 11).

Else if gray levels of cells (7, 11, 15) are the same value ($GI(7) = GI(11) = GI(15)$), there exists a boundary if the gray level $GI(7)$ of the cell (7) is not equal to the gray level $GI(6)$ of the cell (6).The value of gray level could be replaced by the median of gray level of the pixels at cells (5, 6, 9,13 and 14).

else

The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15 and 16).

end"

For the table 7(d), the rules are statements as the following.

"**If** gray levels of cells (6, 7, 8) are the same value ($GI(6) = GI(7) = GI(8)$), there exists a boundary if the gray level $GI(6)$ of the cell (6) is not equal to the gray level $GI(10)$ of the cell (10).The value of gray level for replacement could be replaced by the median of gray level of the pixels at cells (10, 12, 14, 15 and 16).

Else if gray levels of cells (6, 10, 14) are the same value ($GI(6) = GI(10) = GI(14)$), there exists a boundary if the gray level $GI(6)$ of the cell (6) is not equal to the gray level $GI(7)$ of the cell (7).The value of gray level could be replaced by the median of gray level of the pixels at cells (10, 12, 14, 15 and 16).

Else if gray levels of cells (14, 15, 16) are the same value ($GI(14) = GI(15) = GI(16)$), there exists a boundary if the gray level $GI(14)$ of the cell (14) is not equal to the gray level $GI(10)$ of the cell (10).The value of gray level could be replaced by the median of gray level of the pixels at cells (6, 7, 8, 10 and 12).

Else if gray levels of cells (8, 12, 16) are the same value ($GI(8) = GI(12) = GI(16)$), there exists a boundary if the gray level $GI(8)$ of the cell (8) is not equal to the gray level $GI(7)$ of the cell (7). The value of gray level could be replaced by the median of gray level of the pixels at cells (6, 7, 10,14 and 15).

else

The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15 and 16).

end"

If we replace the noised pixels with the value selected by the modified rules, the denoised image is as shown in Figure 9. Comparing Figure 8(a) and Figure 8(b), a lot of pixel groups with one isolated pixel in a neighbourhood were removed, but the gray levels of two pixels at the boundary of the inner square were not correctly changed. Because the replacement by the median value of gray levels of surrounding pixels excludes itself, the error for selection of gray level occurs at the boundary of the image. To solve the problem, we should consider adding some rule for boundary detection and the modified rules are as following.

"When the noise pixels occurs at boundary, the selection of gray level for the noise replacement has to consider more than the general noise pixels."

For example, we consider Table 7(a) to implement the above rule as the following.
"**If** gray levels of cells (1, 2, 3) are the same value ($GI(1)=GI(2)=GI(3)$), there exists a boundary if the gray level $GI(1)$ of the cell (1) is not equal to the gray level $GI(5)$ of the cell (5). The value of gray level could be replaced by the median of gray level of the pixels at cells (5, 7, 9, 10 and 11).
Else if gray levels of cells (1, 5, 9) are the same value ($GI(1)=GI(5)=GI(9)$), there exists a boundary if the gray level $GI(1)$ of the cell (1) is not equal to the gray level $GI(2)$ of the cell (2).The value of gray level could be replaced by the median of gray level of the pixels at cells (2, 3, 7, 10 and 11).
Else if gray levels of cells (9, 10, 11) are the same value ($GI(9)=GI(10)=GI(11)$), there exists a boundary if the gray level $GI(9)$ of the cell (11) is not equal to the gray level $GI(5)$ of the cell (5).The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 5 and 7).
Else if gray levels of cells (3, 7, 11) are the same value ($GI(3)=GI(7)=GI(11)$), there exists a boundary if the gray level $GI(3)$ of the cell (3) is not equal to the gray level $GI(2)$ of the cell (2).The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 5, 9 and 10).
else
The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 7, 8, 9, 11, 10, 12, 13, 14, 15 and 16).
end"
For the Table 7(b), the rules are statements as the following.
"**If** gray levels of cells (2, 3, 4) are the same value ($GI(2)=GI(3)=GI(4)$), there exists a boundary if the gray level $GI(2)$ of the cell (2) is not equal to te gray level $GI(6)$ of the cell (6).The value of gray level could be replaced by the median of gray level of the pixels at cells (6, 8, 10, 11 and 12).
Else if gray levels of cells (2, 6, 10) are the same value ($GI(2)=GI(6)=GI(10)$), there exists a boundary if he gray level $GI(2)$ of the cell (2) is not equal to the gray level $GI(3)$ of the cell (3). The value of gray level could be replaced by the median of gray level of the pixels at cells (3, 4, 8, 11 and 12).
Else if gray levels of cells (10, 11, 12) are the same value ($GI(10)=GI(11)=GI(12)$), there exists a boundary if the gray level $GI(10)$ of the cell (10) is not equal to the gray level $GI(6)$ of the cell (6). The value of gray level could be replaced by the median of gray level of the pixels at cells (2, 3, 4, 6 and 8).
Else if gray levels of cells (4, 8, 12) are the same value ($GI(4)=GI(8)=GI(12)$), there exists a boundary if the gray level $GI(4)$ of the cell (4) is not equal to the gray level $GI(3)$ of the cell (3). The value of gray level could be replaced by the median of gray level of the pixels at cells (2, 3, 6, 10 and 11).
else
The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 6, 8, 9, 11, 10, 12, 13, 14, 15 and 16).
end"
For the Table 7(c), the rules are statements as the following.
"**If** gray levels of cells (5, 6, 7) are the same value ($GI(5)=GI(6)=GI(7)$), there exists a boundary if he gray level $GI(5)$ of the cell (5) is not equal to the gray level $GI(9)$ of the cell (9).The value of gray level could be re-

placed by the median of gray level of the pixels at cells (9, 11, 13, 14 and 15).

Else if gray levels of cells (5, 9, 13) are the same value ($GI(5) = GI(9) = GI(13)$), there exists a boundary if the gray level $GI(5)$ of the cell (5) is not equal to the gray level $GI(6)$ of the cell (6).The value of gray level for replacement could be replaced by the median of gray level of the pixels at cells (6, 7, 11, 14 and 15).

Else if gray levels of cells (13, 14, 15) are the same value ($GI(13) = GI(14) = GI(15)$), there exists a boundary if the gray level $GI(13)$ of the cell (13) is not equal to the gray level $GI(9)$ of the cell (9).The value of gray level could be replaced by the median of gray level of the pixels at cells (5, 6, 7, 9 and 11).

Else if gray levels of cells (7, 11, 15) are the same value ($GI(7) = GI(11) = GI(15)$), there exists a boundary if the gray level $GI(7)$ of the cell (7) is not equal to the gray level $GI(6)$ of the cell (6).The value of gray level could be replaced by the median of gray level of the pixels at cells (5, 6, 9,13 and 14).

else

The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15 and 16).

end"

For the table 7(d), the rules are statements as the following.

"**If** gray levels of cells (6, 7, 8) are the same value ($GI(6) = GI(7) = GI(8)$), there exists a boundary if the gray level $GI(6)$ of the cell (6) is not equal to the gray level $GI(10)$ of the cell (10).The value of gray level for replacement could be replaced by the median of gray level of the pixels at cells (10, 12, 14, 15 and 16).

Else if gray levels of cells (6, 10, 14) are the same value ($GI(6) = GI(10) = GI(14)$), there exists a boundary if the gray level $GI(6)$ of the cell (6) is not equal to the gray level $GI(7)$ of the cell (7).The value of gray level could be replaced by the median of gray level of the pixels at cells (10, 12, 14, 15 and 16).

Else if gray levels of cells (14, 15, 16) are the same value ($GI(14) = GI(15) = GI(16)$), there exists a boundary if the gray level $GI(14)$ of the cell (14) is not equal to the gray level $GI(10)$ of the cell (10).The value of gray level could be replaced by the median of gray level of the pixels at cells (6, 7, 8, 10 and 12).

Else if gray levels of cells (8, 12, 16) are the same value ($GI(8) = GI(12) = GI(16)$), there exists a boundary if the gray level $GI(8)$ of the cell (8) is not equal to the gray level $GI(7)$ of the cell (7). The value of gray level could be replaced by the median of gray level of the pixels at cells (6, 7, 10,14 and 15).

else

The value of gray level could be replaced by the median of gray level of the pixels at cells (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15 and 16).

end"

If we replace the noised pixels with the value selected by the modified rules, the denoised image is as shown in Figure 9.

(a) (b)

Figure 9. (a) the denoised image is modified by using the rules for gray level replacement and (b) the binary image with noise pixels that were not removed.

The mistake of changing colour is solved by the modified replacement rule in Figure 8(a), but in Figure 9(a) and Figure 9(b), some noise pixels at the boundary of the image were not removed. They were not detected by all the tables because the mask is designed to follow the fact that the noised pixels are at the center pixels of the mask surrounded by normal pixels.

2.4 Boundary pixels detection and replacements

We try to detect the noise pixels at the boundary of the image and replace their gray level. But we try to find a method without changing or adding the detection rules. Figure 2(c) is a map we used to detect the noised pixels. For example, we enlarge the map by adding four new boundaries. If the size of original image is m by n, we make an enlarged map with the size (m+2) by (n+2). The new upper and lower boundaries are 1x(n+2) row matrixes with zeros. The added left and right boundaries are (m+2)x1 column matrixes with zeros. If we use the enlarged map in Figure 10(a), the noised images is obtained in Figure 10(b).

3. Experimental results and discussions

The Root Mean Square (RMS) Error is used as the performance measure for comparing our results with those of the median filter.

3.1. Root Mean Square (RMS) Error

The RMS Error is defined as follows:

$$RMS\ \ Error = \sqrt{\frac{1}{m \times n}\sum_{i=1}^{m}\sum_{j=1}^{n}\left(f(i,j)-g(i,j)\right)^2}$$

.

The functions $f(i,j)$ and $g(i,j)$ are original and denoised image respectively. The numbers m and n are the size of the original image.

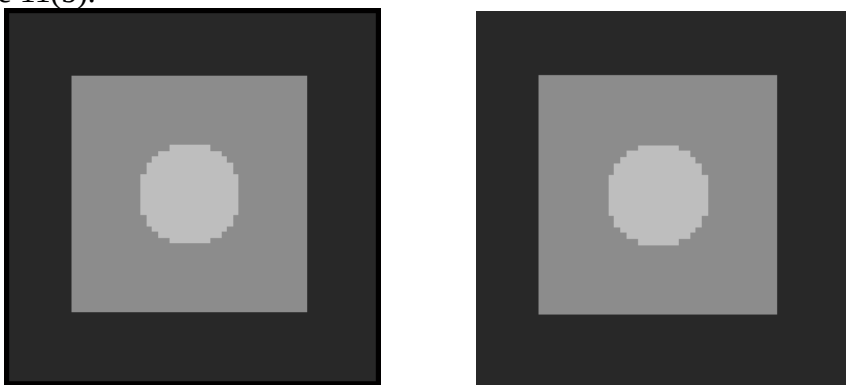
3.2. Results and Discussions

There are one gray image and one color image contaminated by Salt and Pepper used for testing our algorithm.

3.2.1. Gray Image Test

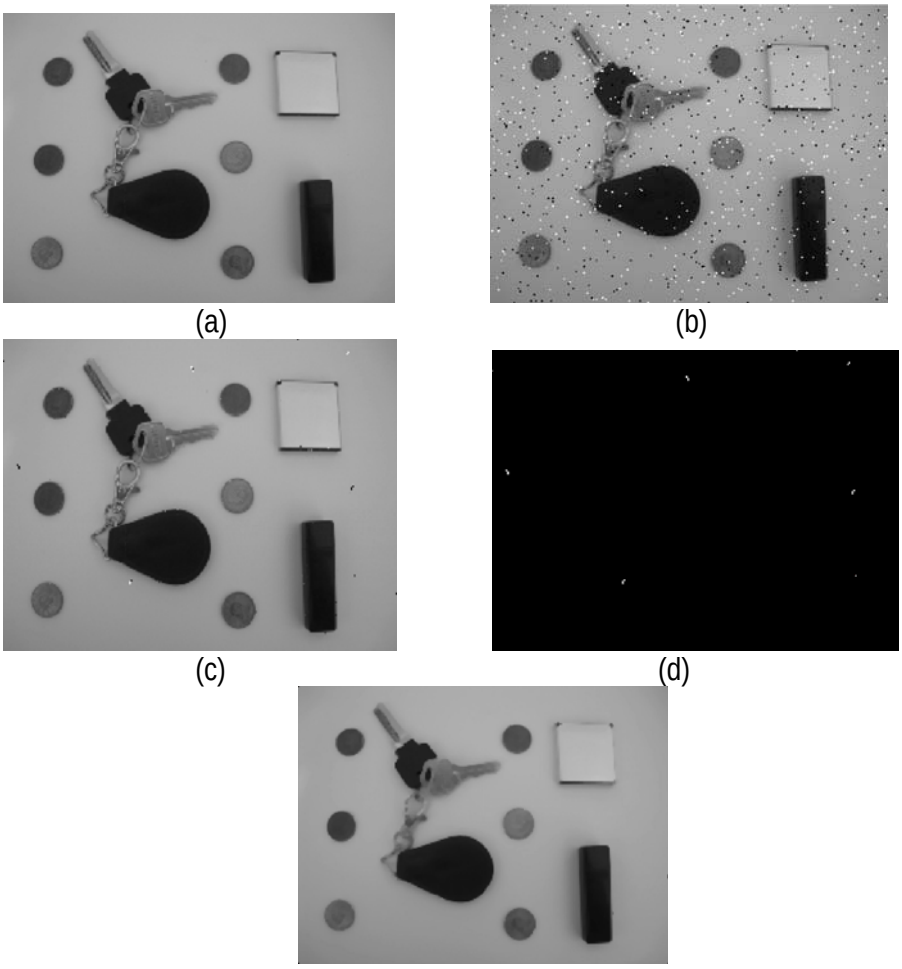
A gray image in Figure 11(a) is a picture taken from a set of objects on a paper. The

image contaminated by Salt and Pepper Noise as shown in Figure 11(b).



(a) (b)

Figure 10. (a) a denoised image with the added boundaries and (b) the black boundary strips were removed. The noise pixels on the boundary are removed as shown in the Figure 10(b) and the result is better than that of the median filter in Figure 4(b), respectively.



(a) (b)

(c) (d)

(e)

Figure 11. (a) original gray image, (b) the noised image with Salt Pepper Noise, (c) the denoised image, (d) the binary image with noise pixels that were not removed, and (e) the denoised image by the median filter.

The denoised image is shown in Figure 11(c) and the binary image with noise pixels not removed is shown in Figure 11(d). The noise groups include more than four pixels and they can be removed in the future if the mask is developed for removing these kinds of noise groups. The denoised image by the median filter is shown in Figure 11(e). The RMS Errors of median filter and our algorithm are 1.7824 and 0.5153. The result shows that our algorithm has better performance for removing Salt and Pepper Noise because our algorithm has the capability of detection of noise pixels.

3.2.2. Color Image Test

A color image in Figure 12(a) is a picture taken from the Sun Moon Lake , which is a famous scenic spot in the central Taiwan. The image contaminated by Salt and Pepper Noise is shown in Figure 12(b).

(a) (b)

Figure 12. A color image (a) without and (b) with Salt and Pepper Noise (d= 0.0005).

(a) (b)

Figure 13. (a) a gray image with Salt and Pepper Noise, and (b) binary image by Thresholding method

We use the process of gray level replacement before the noise detection. A gray image in Figure 13(a) is obtained from Figure 12(b) by converting RGB image to grayscale image. Figure 13(b) is a binary image by applied Thresholding method on the the image in Figure 13(a). The upper threshold is 200 and the lower threshold is 55.

There are three channels of a color image: red, green, and blue channels. We used Figure

13(b) to detect the noise pixels and replace color of noised pixels. The codes are the same as those used for replacing the value of the gray level of the gray image. For each channel, the gray image is denoised and then three denoised gray images of the three channels are combined together to get a color denoised image as in Figure 14.

(a) (b)

Figure 14. (a) a color image with Salt and Pepper Noise (d= 0.0005), and (b) a denoised color image obtained by removing the Salt and Pepper Noise.

As shown in Figure 14(a) and Figure 14(b), we can find that some pixels of noise were removed. Some groups having more than 4 noise pixels are not removed because our mask is suitable for groups with only less than 4 noise pixels

4. Conclusions

Cellular automaton for reduction of image noise was proposed. We used examples to explain the algorithm and the method to design the algorithm for the detection of noised pixels and replacement of the gray level of the noised pixel. Synthetic image is used to test comparing performances between our algorithm and median filter. There were four pixels with error gray level by using median filter to remove the Salt and Pepper Noise. Our algorithm overcomes the drawbacks of median filter by cellular automata to detect the pixels influenced by noises and to replace them with some rules. A real gray image and a color image were used to demonstrate that our proposed algorithm has very good performance. In the future, masks will be developed for removing the noise groups including more than four pixels.

Acknowledgement

We wish to express the appreciation to the National Science Council, Taiwan for the financial support on this research (contract number NSC 99-2115-M-324 001).

References

[1] Wikipedia, 2009. Reproducibility, Wikipedia, [Online]. Available: http://en.wikipedia.org/w/index.php?title=Reproducibility&oldid=262130461

[2] Knuth, D. E. 1984. Literate programming, *The Computer Journal*, 27: 97-111.

[3] Claerbout, J., 1992. Electronic documents give reproducible research a new meaning, *in Proc. 62nd Ann. Int. Meeting of the Soc. of Exploration Geophysics.*

[4] Buckheit, J. B. and Donoho, D. L. 1995. *"WaveLab and reproducible research"*,

Dept. of Statistics, Stanford Univ., Tech. Rep. 474. [Online]. Available: http://www-stat.stanford.edu/~donoho/Reports/1995/wavelab.pdf

[5] Krajcik, Vit. "*Cellular automata*". Faculty of Electrical Engineering and Information Technologies Slovak University of Technology Bratislava, Slovak Republic.

[6] Popovici, Adriana and Popovici, Dan. "*Cellular automata in image processing*". Departments of Computer Science and Mathematics, University of the West Timisoara.

[7] Zhang, Qing, Sato, Youetsu, Takahashi, Jun ya, Muraoka, Kazunobu, and Chiba, Norishige. 1999 Simple cellular automaton-based simulation of ink behaviour and its application to suibokuga-like 3d rendering of trees. *The Journal of Visualisation and Computer Animation*, 10: 27-37.

[8] Rosin, P. L. 2006. Training Cellular Automata for Image Processing, *IEEE Transactions on Image Processing*, 15: 2076-2087.

[9] Rosin, P. L. 2010. Image Processing using 3-State Cellular Automata, *Computer Vision and Image Understanding*, 114: 790-802.

[10] Pudil, P., Novovicova, J., and Kittler, J. V. 1994. Floating search methods in feature selection, *Pattern Recognition Letters*, 15: 1119-1125.