A 3-D CA-BASED EDGE OPERATOR FOR 3-D IMAGES

S. Wongthanavasu

Department of Computer Science
Faculty of Science
Khon Kaen University
Khon Kaen 40002, Thailand
wongsar@kku.ac.th

C. Lursinsap

Department of Mathematics
Faculty of Science
Chulalongkorn University
Bangkok 10330, Thailand
Chidchanok.L@Chula.ac.th

ABSTRACT

This paper presents a new 3-D edge detector based on a cellular automata model. A uniform cellular automata rule using a 3-D von Neumann neighborhood is proposed for carrying out edge detection on binary and gray-scaled 3-D images. The work shows that a cellular automata-based model provides better edge maps for binary and gray-scaled objects when compared to the 3-D well-known edge operators.

1. INTRODUCTION

Cellular Automata (CAs) are a computational paradigm that can breakthrough the von Neumann bottleneck. They were originally introduced by *Neumann* and *Ulam* [1] to provide a formal framework for investigating the behavior of spatiotemporal dynamic complex systems. Cellular automaton (CA) comprises an array of cells, where each cell can be in one of a finite number of possible states, which is updated synchronously in discrete time steps according to cell rules. A state of a cell at the next time step is determined by its neighboring cell's current states.

There are a number of papers which discuss a general one and two dimensional CA [1], but their applications to image processing are quite rare [2-4]. More specifically, there are rare papers published 3-D image processing based on the 3-D CA model as opposed to some other methods [5-6]. The study by Cattell et al. [2] proposed the theoretical aspects of 2-by-n hybrid CA using specific rules. They applied their works to VLSI testing to show that the proposed CA was superior to other methods. Hernandez et al. [2] presented CA for elementary 2D image enhancement. Wongthanavasu et al. [3] presented a 2-D CA for edge detection on binary images. Moreover, Wongthanavasu et al. [4] presented a 2D CA for edge detection on grayscale images and evaluated its performance in comparison with well-known edge operators.

This paper proposes a 3-D CA with a regular configuration for edge detection on 3-D images. It starts by introducing basic definitions and fundamentals necessary for understanding the proposed CA. Then, an application of CA model to edge detection on binary and grayscale objects is presented in comparison with the well-known edge operators.

2. BASIC DEFINITION OF 3-D CA(S)

Let I denote the set of integer. A 3-D cellular space is a 4-tuple, $(IxIxI, V, \nu_0, f)$, where $IxIxI$ is a set of cartesian product of three integer sets, V is a set of cellular states, ν_0 is a distinguished element of V called the quiescent state, and f is the local transition function from n-tuples of elements of V into V. The function f is subject to the restriction $f(\nu_0, \nu_0, \ldots, \nu_0) = \nu_0$, where ν_0 is the quiescent state. The relevant neighborhood function is a function from $IxIxI$ into 2^{IxIxI} and is thus defined by $g(\alpha) = \{\alpha+\delta_1, \alpha+\delta_2, \ldots, \alpha+\delta_n\}$, for all $\alpha \in IxIxI$, where δ_i $(i = 1,2,\ldots,n) \in IxIxI$ is fixed. For example, a 3-D von Neumann neighborhood consisting of 7 immediate orthogonal neighbors in six directions surrounding the central cell has $g(\alpha) = \{\alpha + (0,0,0), \alpha + (1,0,0), \alpha + (0,1,0), \alpha + (-1,0,0), \alpha + (0,-1,0), \alpha + (0,0,-1), \alpha + (0,0,1)\}$. The neighborhood state function of a cell α at time t is defined by $h^t(\alpha) = (\nu^t(\alpha+\delta_1),\nu^t(\alpha+\delta_2),\ldots,\nu^t(\alpha+\delta_n))$. For 3-D von Neumann neighborhood, the neighborhood state function of the central cell (c) is defined by: $h^t(c) = (\nu^t(c), \nu^t(n),\nu^t(e), \nu^t(s), \nu^t(w), \nu^t(f), \nu^t(r))$, where $\nu^t(c)$ is current state of the central cell, $\nu^t(n)(\nu^t(s))$ for the north (south) cells, $\nu^t(e)(\nu^t(w))$ for the east (west) cells, and $\nu^t(f)(\nu^t(r))$ for the front (rare) cells shown in Figure 1.

Now we relate the neighborhood state of a cell α at time t to the cellular state of that cell at time $t+1$ by $f(h^t(\alpha)) = \nu^{t+1}(\alpha)$. For 3-D von Neumann neighborhood, $f(h^t(c)) = \nu^{t+1}(c)$. The function f is referred to as the 3-D CA rule and is usually given in the form of a state table, specifying all possible pairs of the form $(h^t(\alpha),\nu^{t+1}(\alpha))$.

0-7803-8554-3/04/$20.00 ©2004 IEEE.

Whenever convenient, we omit the time superscript t from h^t.

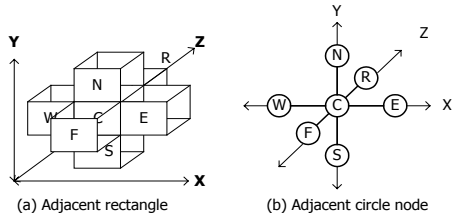


Figure 1. 3-D von Neumann neighborhood.

3. CELLULAR AUTOMATA WITH A REGULAR CONFIGURATION

This section provides a necessary fundamentals and theoretical computation of the proposed 3-D CA using a finite state machine (*FSM*). All arithmetic is performed in 7-based number (number of neighbors in 3-D von Neumann) with a condition.

For our purposes, a finite state machine M comprises n single-bit memory elements and a transition function. The state of the ith memory element at time t is denoted $v^t(i)$. The state of M at time t is denoted v^t. The transition function f determines the state of M at time $t+1$ from the state at time t: that is $v^{t+1} = f(v^t)$. The function f can be specified as n functions $f_1,f_2,...,f_n$, where the i-th function calculates the next state of cell i: that is $v^{t+1}(i) = f_i(v^t(i))$. For simplicity, v is used to denote the present state and v^+ the next state. Similarly, $v(_i)$ denotes the present state and $v^+(i)$ the next state of cell i.

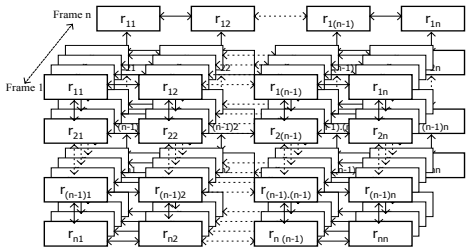


Figure 2. Structure of 3-D *n-by-n-by-n* CA.

Figure 2 shows the interconnection structure of the n-by-n-by-n CA studied here. We use the variables r_{xy}, , *where* $x,y = 1,2,3,...,n,$ to indicate the rule used by the cells, and use a rule vector $[[r_{11},r_{12},...,r_{1n}], [r_{21}, r_{22},..., r_{2n}], ..., [r_{n1}, r_{n2},..., r_{n.n}]]$ to identify a given n-by-n CA at a particular frame. The leftmost, rightmost, topmost and bottom-most cells are assumed to have a constant-0 input on their missing neighbors (*omitted in the picture*). Such a CA is called an *n x n x n CA* with *a regular configuration*. At each time step each cell calculates a new state using its *cell rule*. For our proposed CA, the same rule is applied to all cells, making the CA *uniform*. For a *von Neumann* neighborhood, the cell rule f for cell c calculates its state at the next time step by: $f(h(c)) = v^+(c) = [v(c)+v(n)+v(e)+v(s)+v(w)+v(f)+v(r)] \mod 7$. The $v^+(c)$ value is then filtered by the condition:

If $((v(c)+v(n)+v(e)+v(s)+v(w)+v(f)+v(r)) \mod 7 = 0)$, *in other words* $v^+(c) = 0$, *then* $v^+(c) = 0$, *Otherwise* $v^+(c) = v(c)$.

This rule is summarized in Table 1. According to the uniform CA, the rule r_{xy} used by any cell can be encoded as a single bit "1".

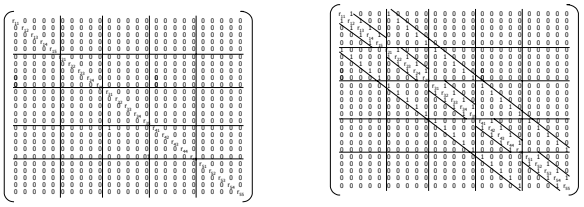
The transition matrix T of a finite state machine (*FSM*) defines the transition function (f) algebraically, so that $v^+ = v.T$. There are two transition matrices to take into account. To calculate the next state of a cell at frame i, the transition matrix T_1, which has an identity block structure, will operate on frame $i-1$ and $i+1$, whereas the transition matrix T_2, which has a tridiagonal block structure, will operate on frame i. For example, the transition matrices (T_1 & T_2) of a below simplified 5-5-3 CA has the block structure shown in Figure 3(a) and 3(b), respectively, where the r_{ij} ($5 \geq i,j \geq 1$) is "*1*" as previously stated.

A simplified CA example comprised 3 consecutive 5-by-5 2-D frames utilized in this computation is given. Since the CA is uniform (applied one rule), the *rule vector* (r_{xy}) for the 2-D CA is represented as [[*1,1,1,1,1*], [*1,1,1,1,1*], [*1,1,1,1,1*], [*1,1,1,1,1*], [*1,1,1,1,1*]].

Frame 1: $[[v_{111}, v_{121}, v_{131}, v_{141}, v_{151}], [v_{211}, v_{221}, v_{231}, v_{241}, v_{251}], [v_{311}, v_{321}, v_{331}, v_{341}, v_{351}], [v_{411}, v_{421}, v_{431}, v_{441}, v_{451}]$, $[v_{511}, v_{521}, v_{531}, v_{541}, v_{551}]] = [[1,1,0,0,1], [1,0,1,1,1], [1,1,0,1,1], [1,0,0,1,1], [1,1,1,1,0]]$.

Frame 2: $[[v_{112}, v_{122}, v_{132}, v_{142}, v_{152}], [v_{212}, v_{222}, v_{232}, v_{242}, v_{252}], [v_{312}, v_{322}, v_{332}, v_{342}, v_{352}], [v_{412}, v_{422}, v_{432}, v_{442}, v_{452}]$, $[v_{512}, v_{522}, v_{532}, v_{542}, v_{552}]] = [[1,0,1,1,0], [0,1,1,1,1], [1,1,1,1,1], [1,1,1,0,0], [1,1,1,0,1]]$.

Frame 3: $[[v_{113}, v_{123}, v_{133}, v_{143}, v_{153}], [v_{213}, v_{223}, v_{233}, v_{243}, v_{253}], [v_{313}, v_{323}, v_{333}, v_{343}, v_{353}], [v_{413}, v_{423}, v_{433}, v_{443}, v_{453}]$, $[v_{513}, v_{523}, v_{533}, v_{543}, v_{553}]] = [[1,1,1,1,1], [1,1,1,1,1], [1,1,1,0,1], [1,1,0,0,1], [1,0,1,1,0]]$.



(a) (b)

Figure 3. Transition Matrices: (a) T_1 for frame i-1 and i+1, (b) T_2 for frame i.

For illustration purposes, the states of the next time step (derived from operating matrix T on a state vector v, i.e., $v^+ = v.T$) for cell v_{232} is calculated as follow:

v_{232}^{+} = {[(state vector of frame 1: [v_{111}, v_{121}, v_{131},…,v_{551}]) * (8-th column of matrix T_1)] + [(state vector of frame 2: [v_{112}, v_{122}, v_{132},…,v_{552}]) * (8-th column of matrix T_2)] + [(state vector of frame 3: [v_{113},…,v_{553}]) * (8-th column of matrix T_1)]} mod 7

= {(v_{231}+0+0+…+0+0) + (v_{132} + v_{222} +v_{232} +v_{242} +0+…+v_{332}+0+…+0)+(v_{233} +0+…+0)} mod 7

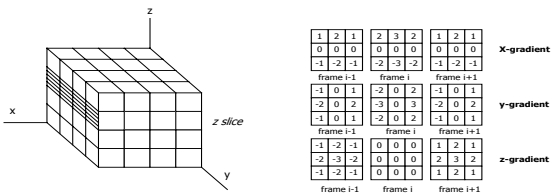
= {(1+0+0+…+0)+(1+1+1+1+1+0+..+0) + (1+0+0+…+0)} mod 7 = 0

Using the next state functions, a sequence of state vectors is evaluated. Obviously, the CA produces the same state vector after the first time step of evolution. That is, the evolution leads to a stable configuration after a specific number of time steps.

4. APPLICATION TO EDGE DETECTION

4.1. Topology of 3-D Images

A 3-D image can be thought either as a 3-D matrix with x-y-z dimensions or as a stack of 2-D images. In the first case, a 3-D image can be considered as a 3-D matrix a[x][y][z](*or a(x,y,z)*) of dimensions N_1xN_2xN_3 where x,y,z denote column, row, and slice (image) coordinates, respectively (Fig.4a). In the latter case x-y refers to coordinates within the image and z refers to the 2-D image (frame). Both representations are equivalent [5].



(a) (b)

Figure 4. (a) 3-D image representation as a 3-D array, (b) 3-D Sobel's coefficients.

Each pixel of a 2-D image belonging to a stack of images, or equivalently, each position in the 3-D image, is called a voxel (point).

4.2. Sobel and Laplacian Edge Operators

The proposed 3-D CA is applied to a test set of 3-D images. For comparison purposes, 3-D Laplacian of Gaussian and Sobel edge operators [5] were tested on the set of 3-D images. 3-D Sobel edge operator emphasizes edge details using three *3-by-3-by-3 edge convolution operators*, one detecting X-gradients, the second Y-gradients, and the third Z-gradients then using the sum of absolute value of each gradient to determine the magnitude and orientation of an edge. The coefficients of 3D Sobel's convolution mask are shown in Figure 4b. 3-D Laplacian edge operator (∇^2) uses *a 3-by-3-by-3 spatial convolution mask* to approximate edge maps [8], which is defined as follow:

$$\nabla^2 f(x,y,z) = \frac{\partial^2 f(x,y,z)}{x^2} + \frac{\partial^2 f(x,y,z)}{y^2} + \frac{\partial^2 f(x,y,z)}{z^2}$$

For identification purposes, an *edge* on a binary image is defined as *a path* separating an object (*black cluster*) from a background (*white cluster*) and three consecutive frames of 320x200 images are utilized for being processed. Figures 5-6 shows our approach and the compared methods as applied to the frame 31 of the test binary images.

5. EXTENSION TO 256 GRAY-SCALED IMAGES

The CA-based model is now extended to 3-D grayscale images (*8-bit images*). Instead of two states, the *256* possible states are now considered. By considering all cell rules given in Table 1, which deals with binary states, we encounter the relationship between the neighborhood state function (*f(v)*) and the state of a central cell at the next time step (v_c^{+}) by the following condition: *If* $v_c <= v_{max} - v_{min}$ *then* $v_c^{+} = v_c$, *otherwise* $v_c^{+} = v_{max} - v_{min}$, where v_c is the current state of the central cell, and v_{max}(v_{min}) is the *maximum (minimum)* states in the considering neighborhood, respectively. It is obvious that the central cell will change its state when its current state is greater than the difference between the *maximum* and the *minimum* states in its neighborhood. We can take the conditional rule to carry out the computation.

6. DISCUSSION AND CONCLUSION

A 3-D uniform CA using a 3-D von Neumann neighborhood with a regular configuration is utilized to model and determine the magnitude and orientation of edges on 3-D binary and 256 grayscale images. For a comparative evaluation, Sobel and Laplacian of Gaussian edge operators were compared. In processing on 3-D CT head binary image, 3-D CA edge operator provides the quality of edge maps much better than 3-D Laplacian and Sobel (Fig.6). It provides clear, unbroken, and one pixel-wide edge maps, whereas Sobel gives thick edges and adjacent boundaries tend to merge. In grayscale image case, CA shows the promising results that its edge maps are clear, thin, and better than Sobel and Laplacian on average (Fig. 7). In sense of 3-D object recognition, the quality of edge detection is directly relevant to the recognition performance. The edge maps resulting from 3-D CA edge operator are suggestive a better performance comparable to the compared methods.

h(c)	v^+_c	h(c)	v^+_c	h(c)	v^+_c	h(c)	v^+_c
0000000	0	0100000	0	1000000	1	1100000	1
0000001	0	0100001	0	1000001	1	1100001	1
0000010	0	0100010	0	1000010	1	1100010	1
0000011	0	0100011	0	1000011	1	1100011	1
0000100	0	0100100	0	1000100	1	1100100	1
0000101	0	0100101	0	1000101	1	1100101	1
0000110	0	0100110	0	1000110	1	1100110	1
0000111	0	0100111	0	1000111	1	1100111	1
0001000	0	0101000	0	1001000	1	1101000	1
0001001	0	0101001	0	1001001	1	1101001	1
0001010	0	0101010	0	1001010	1	1101010	1
0001011	0	0101011	0	1001011	1	1101011	1
0001100	0	0101100	0	1001100	1	1101100	1
0001101	0	0101101	0	1001101	1	1101101	1
0001110	0	0101110	0	1001110	1	1101110	1
0001111	0	0101111	0	1001111	1	1101111	1
0010000	0	0110000	0	1010000	1	1110000	1
0010001	0	0110001	0	1010001	1	1110001	1
0010010	0	0110010	0	1010010	1	1110010	1
0010011	0	0110011	0	1010011	1	1110011	1
0010100	0	0110100	0	1010100	1	1110100	1
0010101	0	0110101	0	1010101	1	1110101	1
0010110	0	0110110	0	1010110	1	1110110	1
0010111	0	0110111	0	1010111	1	1110111	1
0011000	0	0111000	0	1011000	1	1111000	1
0011001	0	0111001	0	1011001	1	1111001	1
0011010	0	0111010	0	1011010	1	1111010	1
0011011	0	0111011	0	1011011	1	1111011	1
0011100	0	0111100	0	1011100	1	1111100	1
0011101	0	0111101	0	1011101	1	1111101	1
0011110	0	0111110	0	1011110	1	1111110	1
0011111	0	0111111	0	1011111	1	1111111	0

Table 1. A state table of the proposed 3-D CA using a 3-D von Neumann neighborhood.

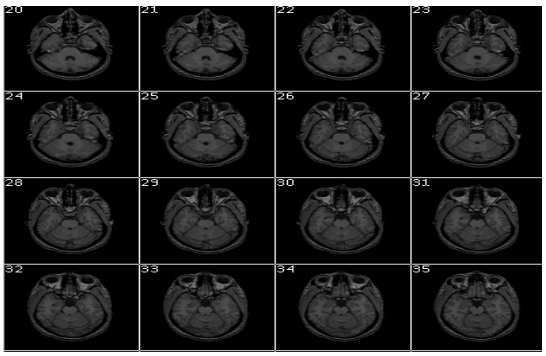
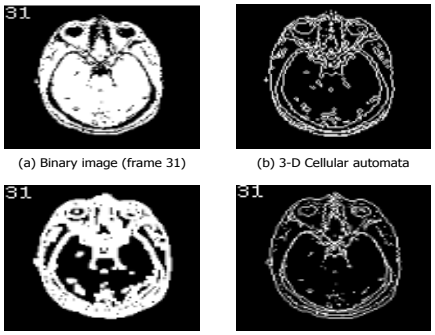


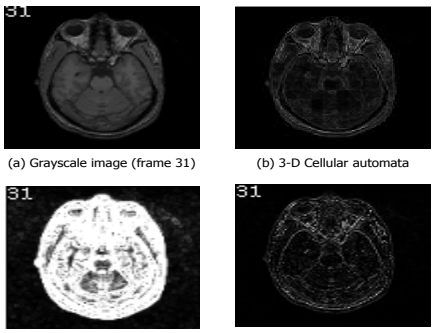
Figure 5. Partial frames of 3-D gray-scaled head images (Originally CT image data courtesy of the Visible Human Project, National Library of Medicine, Bethesda, Maryland).



(a) Binary image (frame 31) (b) 3-D Cellular automata

(c) 3-D Sobel (d) 3-D Laplacian

Figure 6. Edge maps resulting from 3-D edge operators as applied to a binary frame 31 (obtained from Fig.5).



(a) Grayscale image (frame 31) (b) 3-D Cellular automata

(c) 3-D Sobel (d) 3-D Laplacian

Figure 7. Edge maps resulting from 3-D edge operators as applied to a grayscale frame 31 (Fig.5) .

7. REFERENCES

[1] Sipper, M., *Evolution of Parallel Cellular Machine: The Cellular Programming Approach.* LNCS, 1194, Springer, 1997.

[2] G. Hernandez and J.J. Herrmann, "Cellular Automata for Elementary Image Enhancement," *Graphical Models and Image Processing (GMIP),* Vol. 4, No. 58, pp. 82-89, 1996.

[3] S. Wongthanavasu and R. Sadananda, "Pixel-level Edge Detection Using a Cellular Automata-Based Model," *Advances in Intelligent Systems: Theory and Applications.* The Netherlands: IOS Press, vol. 59, pp. 343-351, 2000.

[4] S. Wongthanavasu and R. Sadananda, "A CA-based Edge Operator and Its Performance Evaluation," *Journal of Visual Communication and Image Representation,* Elsevier Science, San Diego, U.S.A., vol. 14, no. 2, pp. 83-96, 2003.

[5] Nikolaidis, N. and I. Pitas, *3-D Image Processing Algorithms,* John Wiley & Sons, Inc., New York, 2001.

[6] M. Bomans, K. Hohne, U. Tiede, and M. Riemer, "3-D segmentation of MR images of the head for 3-D display," *IEEE Trans. on Medical Imaging*, vol. 9, no. 2, pp. 177-183, 1990.