2010 12th International Workshop on Cellular Nanoscale
Networks and their Applications (CNNA)

Image Processing Application Using CNN with Dynamic Template

Makasakazu KAWAHARA
Dept. Electrical and Electronic Eng.,
Tokushima University
kawahara@ee.tokushima-u.ac.jp

Takashi INOUE
Dept. Electrical and Electronic Eng.,
Tokushima University
takashi@ee.tokushima-u.ac.jp

Yoshifumi NISHIO
Dept. Electrical and Electronic Eng.,
Tokushima University
nishio@ee.tokushima-u.ac.jp

***Abstract*— In this research, we propose an image processing application using cellular neural networks (CNN) with dynamic template (D-CNN). In D-CNN, the wiring weights of template are dynamically changed at each update by learning. In this study, we also focus on the number of converged value. Thus, some converged values of each cell in the initial input image are changed by effect of next input image in motion picture. Although input image is only still image, we think that if multiple input images are inputted to the D-CNN as motion picture, converged values of each cell are continued to changed. We investigate the converged value by changing input image. The results indicate that D-CNN can be applied to the region segregated processing for the motion picture.**

I. Introduction

Cellular neural networks (CNN) were proposed by Chua and Yang in 1988 [1]. The idea of CNN was inspired from the architecture of the cellular automata and the neural networks. Unlike the original neural networks, the CNN has local connectivity property. Wiring weights of the cells are established by parameters called the template. The performance of the CNN is decided by the template. Also, the CNN has been successfully used for various high-speed parallel signals processing applications such as image processing application [2]-[4], including medical image processing [5][6] and texture segmentation [7]. Usually, the templates of all the cells in the CNN are identical and those values do not change during the processing. This is good for implementation but restrict the performance, namely the conventional CNN can not perform image processing based on the local features of input images.

In the previous study, we have proposed cellular neural networks with dynamic template (D-CNN) [8]. In D-CNN, template is dynamically changed at each update by learning. This learning method is inspired from the rank order learning. Also, we have investigated the output characteristic of D-CNN by using diffusion template. Usually, using diffusion template, values of cells using the conventional CNN converge to one constant value. From the simulation results of the previous study, we confirmed that the converged value of each cell is divided to two or three values. Also, convergence process is much more rapid than that of the conventional CNN. However, input image is only still image. We think that if multiple input images are inputted to the D-CNN as motion picture, converged values of each cell continue to change. Thus, we expect that another application using D-CNN is found from these multiple converged values.

In this study, we propose an image processing application using CNN with dynamic template. We apply D-CNN to motion picture processing through some investigations. From the investigation of converged values by changing input images, we understand the output image. In this way, another image is inputted to the D-CNN, the converged value of each cell continues to change. According to our simulation, the converged value of each cell is divided to more points than the case of one input image. The results indicate that D-CNN can be applied to the region segregated processing for the motion picture.

In the Sec. 2, we review the basic of the standard CNN. In the Sec. 3, we show the algorithm of the proposed D-CNN. In the Sec. 4, simulation results using the proposed D-CNN are shown. In the Sec. 5, we show image processing application using this investigation result. The Section 6 concludes the article.

II. Cellular Neural Networks [1]

In this section, we explain the basic structure of the CNN. The CNN has M by N processing unit circuits called cells. Cells are arranged in a reticular pattern to M line N row. We represent a cell $C(i,j)$ using a variable i which denotes vertical position and a variable j which denotes horizontal position. The cell contains linear and nonlinear circuit elements. The CNN is an array of cells. Each cell is connected to only its neighboring cells according to a template. Usually, the template is the same for all cells except for boundary cells. The CNN has the features of time continuity, spatial discreteness, nonlinearity and parallel processing capability

State equation:

$$\frac{dv_{xij}}{dt} = -v_{xij} + \sum_{k=i-r}^{i+r}\sum_{l=j-r}^{j+r} A_{(i,j;k,l)}v_{ykl}(t) + \sum_{k=i-r}^{i+r}\sum_{l=j-r}^{j+r} B_{(i,j;k,l)}v_{ukl}(t) + I. \quad (1)$$

Output equation:

$$v_{yij}(t) = \frac{1}{2}(|v_{xij}(t) + 1| - |v_{xij}(t) - 1|). \quad (2)$$

978-1-4244-6678-8/10/$26.00 ©2010 IEEE

where v_x, v_y and v_u represent a state, an output and an input of cell, respectively. In the equation (1), A is the feedback template and B is the control template. These and bias I are collectively called general template. In this equation, the control template depends on input value.

III. CNN WITH DYNAMIC TEMPLATE [8]

In this section, we explain the algorithm of D-CNN. In our research, we change input images when a certain calculation times comes. In our D-CNN, the templates are updated at every iterations by rank order learning. The learning steps in our D-CNN are described as follows.

STEP 1: The state values and the output values of all the cells in D-CNN are updated according to the discretized model of Eqs. (1) and (2).

STEP 2: Calculate the comparison of the output value of each cell with the one-step-past outputs of the cell and its neighbor cells like Fig. 1. The comparison equation for the cell (i, j) is described Eq. (3).

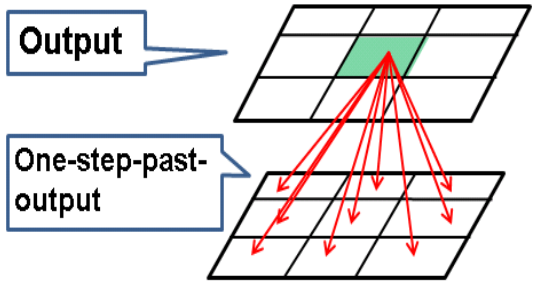


Fig. 1. Comparison of output value of each cell with one-step-past outputs of the cell and its neighbor cells.

Comparison Equation:

$$Dif(i,j;k,l) = |v^{past}_{y,(i,j)} - v^{now}_{y,(k,l)}|. \tag{3}$$

STEP 3: Among the 9 calculated values of $Dif(i, j; k, l)$, the cells with the smallest and the second smallest values are defined as "winner" and "second", respectively. In our update algorithm, we change the learning rate in two elements. By this step, we find the position of cells with the nearest and the second nearest values to the corresponding cell (i, j).

STEP 4: Update the elements of the template corresponding to the positions of the "winner" and the "second". Note that in our proposed learning algorithm only two elements are updated. The update method and the update function are described as follows.

Update Method:

Assume that the template before update is given as Eq. (4).

$$Template^{now}:$$
$$A^{now}_{(i,j)} = \begin{bmatrix} a^{now}_{11} & a^{now}_{12} & a^{now}_{13} \\ a^{now}_{21} & a^{now}_{22} & a^{now}_{23} \\ a^{now}_{31} & a^{now}_{32} & a^{now}_{33} \end{bmatrix},$$
$$B^{now}_{(i,j)} = \begin{bmatrix} b^{now}_{11} & b^{now}_{12} & b^{now}_{13} \\ b^{now}_{21} & b^{now}_{22} & b^{now}_{23} \\ b^{now}_{31} & b^{now}_{32} & b^{now}_{33} \end{bmatrix},$$
$$I^{now}_{(i,j)} = I^{now}. \tag{4}$$

For example, we consider the case that the "winner" is (i, j) and the "second" is $(i-1, j-1)$. In that case, only a^{now}_{22}, b^{now}_{22}, a^{now}_{11} and b^{now}_{11} in Eq. (4) are updated. The threshold value I is not updated in our learning method.

In our update algorithm, we change the learning rate in two elements. The learning rates of the "winner" and the "second" are shown as follows.

Learning rate:

$$R_1 = R_{10}\left(1 - \frac{Number\ of\ calculation}{Number\ of\ calculation_{max}}\right). \tag{5}$$
$$R_2 = R_{20}\left(1 - \frac{Number\ of\ calculation}{Number\ of\ calculation_{max}}\right). \tag{6}$$

In this study, we decide $Number\ of\ calculation_{max}$ in Eqs. (5) and (6) to be set to 10. Namely, the learning rates of "winner" and "second" are changed until 10 calculations. Then, after $Number\ of\ calculation_{max}$ becomes over 10, the learning rates of "winner" and "second" become 0 and the templates are not updated. By using the learning rate, the elements of the template are updated according to the following update equation.

Update Equation:

$$a^{updated}_{winner} = a^{now}_{winner} - R_1(v^{past}_{y,(i,j)} - v^{now}_{y,(i,j)}). \tag{7}$$
$$a^{updated}_{second} = a^{now}_{second} - R_2(v^{past}_{y,(i,j)} - v^{now}_{y,(i,j)}). \tag{8}$$

R_1 and R_2 decrease according to the Eqs. (5) and (6). The initial learning rates are given as follows.

Initial Learning rate:

$$Winner \ : \ R_{10} \quad (0 \le R_{10} \le 0.1). \tag{9}$$
$$Second \ : \ R_{20} = R_{10}/4. \tag{10}$$

After the update using Eqs. (7) and (8), the updated template is shown as follows. In Eq. (11), $a^{updated}_{11}$ and $a^{updated}_{22}$ are the updated values. Also, $b^{updated}_{11}$ and $b^{updated}_{22}$ are updated

similarly.

$$
\begin{aligned}
&Template^{updated}: \\
&A_{(i,j)}^{updated} = \begin{bmatrix} a_{11}^{updated} & a_{12}^{now} & a_{13}^{now} \\ a_{21}^{now} & a_{22}^{updated} & a_{23}^{now} \\ a_{31}^{now} & a_{32}^{now} & a_{33}^{now} \end{bmatrix}, \\
&B_{(i,j)}^{updated} = \begin{bmatrix} b_{11}^{updated} & b_{12}^{now} & b_{13}^{now} \\ b_{21}^{now} & b_{22}^{updated} & b_{23}^{now} \\ b_{31}^{now} & b_{32}^{now} & b_{33}^{now} \end{bmatrix}, \\
&I_{(i,j)}^{updated} = I^{now}. \qquad (11)
\end{aligned}
$$

STEP 5: The steps from 1 to 4 are repeated.

These learning steps inspired from the rank order learning.

IV. Simulation Results

In this section, we show the simulation results for multiple input images using D-CNN. In the first step of this simulation, an initial template is set to D-CNN. The elements of the initial template are updated by using updated method in the previous section. We choose the diffusion template [8] as an initial template in this study as follows.

Initial Template:

$$
\begin{aligned}
&A = \begin{bmatrix} 0.1 & 0.15 & 0.1 \\ 0.15 & 0 & 0.15 \\ 0.1 & 0.15 & 0.1 \end{bmatrix}, \; B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
&I = 0. \qquad (12)
\end{aligned}
$$

Input image diffuses without limit by using the diffusion template. Namely, all values of cells converge to one constant value.

In this simulation, we investigate two change patterns of input images. First, we investigate convergence process when the background of an image is bright. Figure 2 shows simulation results for two patterns of input images. The difference between the first pattern and the second pattern is whether the second input image is Fig. 2(b) or (c). Namely, in the first pattern, input image changes from Fig. 2(a) to Fig. 2(b) and in the second pattern, input image changes from Fig. 2(a) to Fig. 2(c).

Figures 2(d)-(f) shows the simulation results for Fig. 2(a)-(c). Figure 2(d) shows the output image between 0 $[\tau]$ and 10 $[\tau]$. In Fig. 2(d), all the cells have same converge value. However, in Fig. 2(e) and (f), other converged value appear by changing input image. Namely, we can say that the converged value of the first input image changes by the next inputted image.

Second, we investigate convergence process when the background of an image is dark. Figure 3 shows simulation results for 2 patterns of input images. Changing pattern of input image

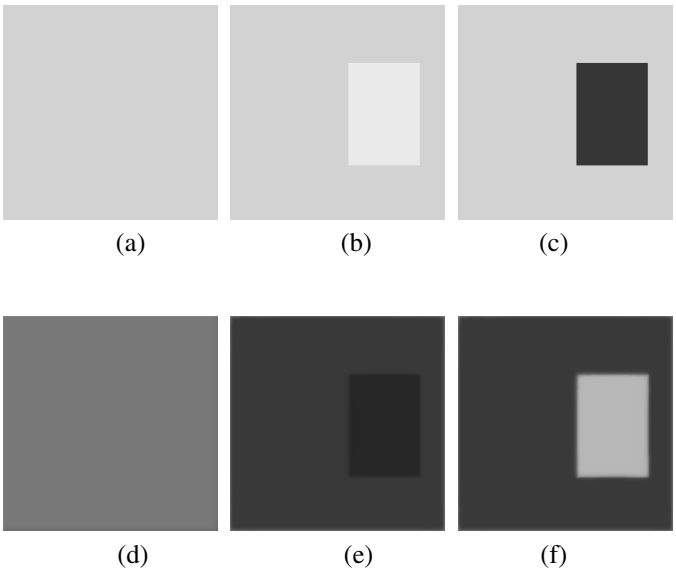


Fig. 2. Simulation result. (a) Input image1. (b) Input image2 (after 10 $[\tau]$). (c) Input image3 (after 10 $[\tau]$). (d) Output image before input image is changed. (e) Output image1 after input image turn to Fig. 2(b). (f) Output image2 after input image turn to Fig. 2(c).

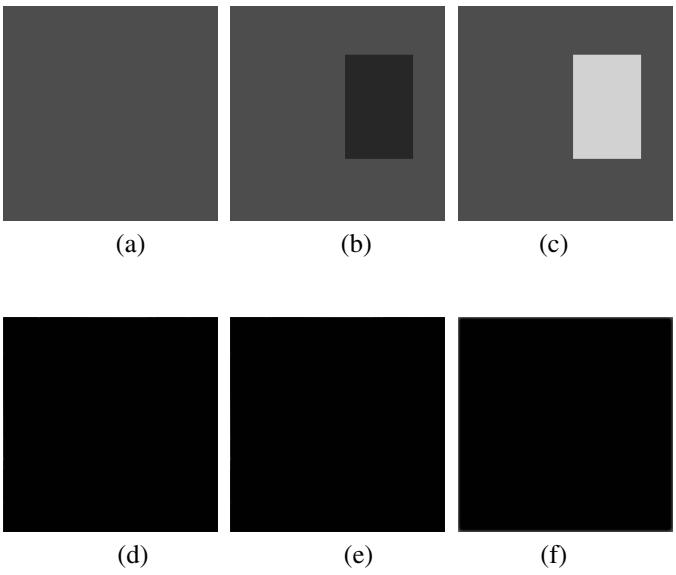


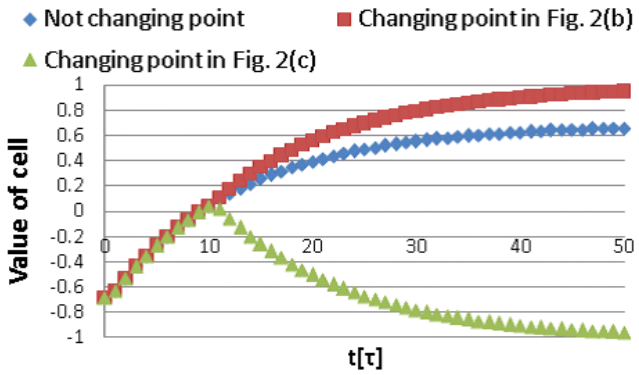
Fig. 3. Simulation results. (a) Input image1 (initial input image). (b) Input image2 (after 10 $[\tau]$). (c) Input image3 (after 10 $[\tau]$). (d) Output image before input image is changed. (e) Output image1 after input image turn to Fig. 2(b). (f) Output image2 after input image turn to Fig. 2(c).

is the same as in Fig. 2. In Fig. 3, we confirmed that the output image converges to one constant value whatever the value of changed area by changing input image. In Figs. 3(d)-(f), converged value is same despite changing input images. Namely, the converged value depends on whether the value of background is over 0 or not.

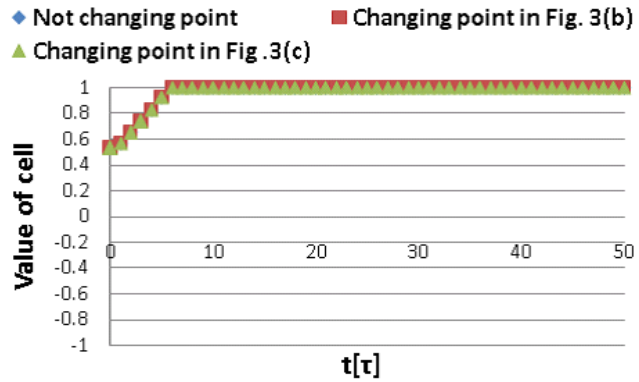
Next, we investigate the behavior of the convergence process by changing input image.

A. Investigation of converged value

In this section, we investigate the converged value of each cell in detail. In order to investigate the converged value, we investigate the output value of each cell in each step.



(a)



(b)

Fig. 4. The changing process of v_y. (a) The convergence process in Fig. 2. (b) The convergence process in FIg. 3.

Figure 4 shows the convergence process in Figs. 2 and 3. When the calculation time comes 10 $[\tau]$, next image is inputted. These graphs correspond to the output value of each cell. In this investigation, we take $C_{(i,j)}$=$(183, 104)$ which change v_u by changing input image.

From Fig. 4(a), we can see that if the background of image is near to white, the changing point in Fig. 2(c) appears with an inversed color. On the other hand, the changing point in Fig. 2(b) changes darker.

In Fig. 4(b), we can see that if the background of image is near to black, the changing points in Figs. 3(b) and (c) appear with a black color. From simulation results, we should analyze why the inversed color appears when the background of image near to -1. For this analysis, we examine the elements of the updated template in D-CNN.

B. Investigation of updated template

Next, we investigate updated template in Figs. 2 and 3 for analysis. In this examination, we focus on the B template. Because, in the state equation (1), B template is influenced by the input value V_u.

First, the updated template for Fig. 2 is shown as follows.

$$Updated\ Template\ in\ Fig.\ 2:$$

$$A = \begin{bmatrix} 0.363 & 0.15 & 0.1 \\ 0.15 & -0.016 & 0.15 \\ 0.1 & 0.15 & 0.1 \end{bmatrix},$$

$$B = \begin{bmatrix} -0.064 & 0 & 0 \\ 0 & -0.016 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$I = 0. \tag{13}$$

From this updated template, we can see that the elements of B template is negative in Fig. 2. If the next input is dark (plus value), the output is bright (negative value) because of the negative B elements. While the input is bright (minus value), the output is dark (positive value). Therefore, the changed area is detected as inversed color.

Next, the updated template for Fig. 3 is shown as follows.

$$Updated\ Template\ in\ Fig.\ 3:$$

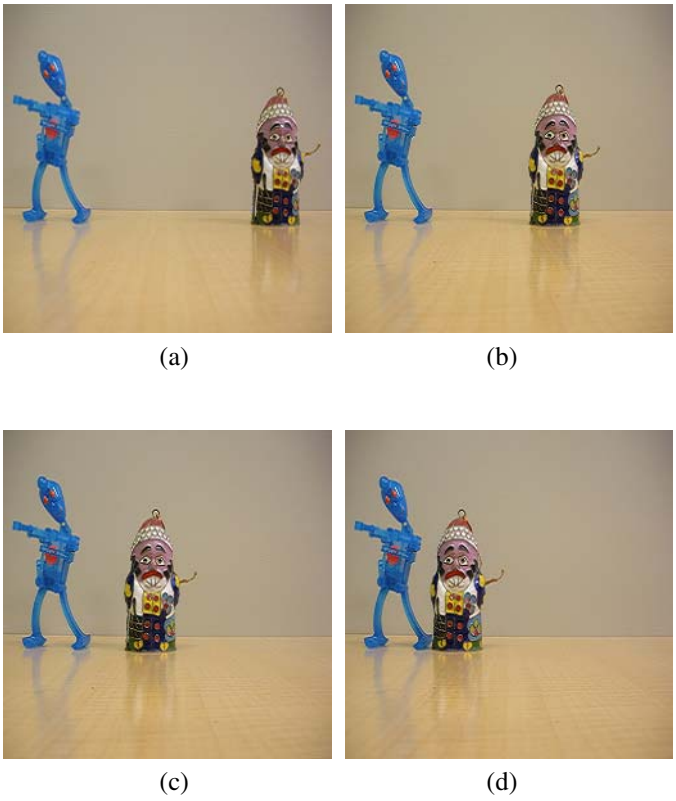
$$A = \begin{bmatrix} 0.156 & 0.15 & 0.1 \\ 0.15 & 0.014 & 0.15 \\ 0.1 & 0.15 & 0.1 \end{bmatrix},$$

$$B = \begin{bmatrix} 0.056 & 0 & 0 \\ 0 & 0.0141 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$I = 0. \tag{14}$$

From this updated template, we can see that the elements of B template is negative in Fig. 2. If the next input is dark (plus value), the output is bright (negative value) because of the negative B elements. While the input is bright (minus value), the output is dark (positive value). Therefore, the changed area is detected as inversed color.

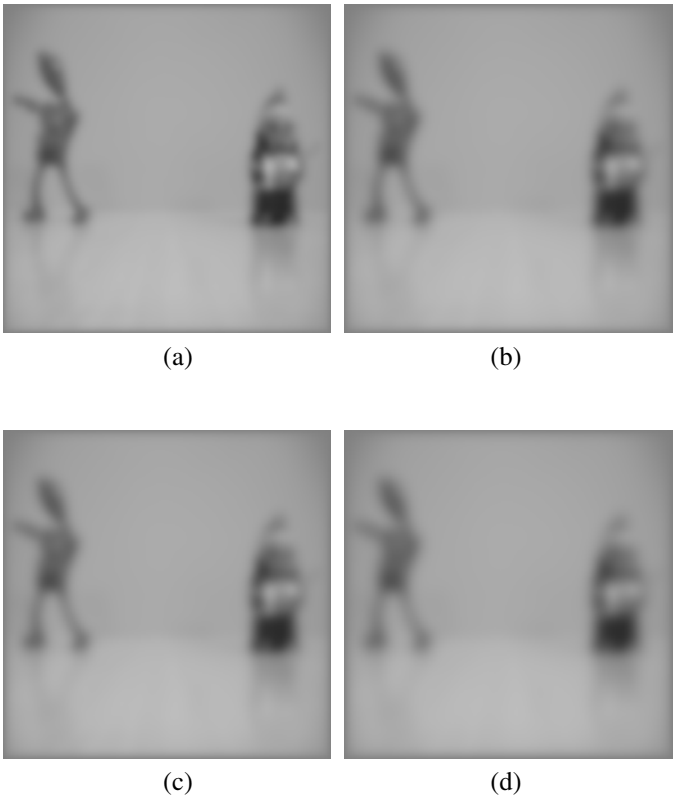
From this updated template, we can see that the elements of B template is positive in Fig. 3. We expected the exactly opposite results to the previous one. However, the results are different. For Fig. 3, the values of the output converge to 1 very quickly (see Fig. 4(b)) and new input cannot influence to invert the color. Namely, the converged value of the changed area dose not depend only on the color of the next input image, but also on the convergence speed for the initial image.



(a) (b)

(c) (d)

Fig. 5. Motion picture. (a) Input image 1 (from 0 to 10 $[\tau]$). (b) Input image 2 (from 10 to 20 $[\tau]$). (c) Input image 3 (from 20 to 30 $[\tau]$). (d) Input image 4 (after 30 $[\tau]$).



(a) (b)

(c) (d)

Fig. 6. Simulation results using conventional CNN for Fig. 5. (a) Output image 1 ($[\tau]$=10). (b) Output image 2 ($[\tau]$=20). (c) Output image 3 ($[\tau]$=30). (c) Output image 4 ($[\tau]$=40).

From the simulation results in this section, we feel that the region segmentation where the object moves or not in motion pictures is possible using D-CNN.

V. Motion picture processing

In this section, we try to apply D-CNN to motion picture processing.

Figure 5 shows 4 input images as a motion picture. We can recognize that the position of the doll gradually changed from right to left through Figs. 5 (a) to (d) like a motion picture. Also, the other object and the background do not change.

Figures 6 and 7 show the simulation results using the conventional CNN and D-CNN for learning rate $R_{10} = 0.1$, respectively, in the input images of Fig. 5 using the initial template of Eq. (12). In Fig. 6, the output images of the conventional CNN are not influenced by the change of the input image, because the B template is set as 0. However, in Fig. 7, the initial area filled with black and changing area is detected. Furthermore, the final position of the moving object is detected and the initial position of the moving object is filled with black in Fig. 7(c).

From this simulation results, we can say that D-CNN is effective for the motion picture processing. Also,the converged value for a new image is different from that of the initial input image.

VI. Conclusions

In this study, we have proposed an image processing application using cellular neural networks with dynamic template (D-CNN). In D-CNN, the template was changed by rank order learning. Also, we investigated the convergence for changing output image and motion pictures. From the simulation results, we confirmed a law of changing process of template of D-CNN. Realizing more intelligent tasks depending on inputs are our future research.

References

[1] L. O. Chua and L. Yang, "Cellular Neural Networks:Theory," IEEE Trans. Circuits Syst., vol. 32, pp. 1257-1272, Oct. 1988.

[2] F. Dirk and T. Ronald, "Coding of Binary Image Data using Cellular Neural Networks and Iterative Annealing," Proc. of ECCTD'03, vol. 1, pp. 229-232, Sep. 2003.

[3] M. Namba and Z. Zhang, "Cellular Neural Network for Associative Memory and Its Application to Braille Image Recognition," Proc. of IJCNN'06, pp. 4716-4721, Jul. 2006.

[4] H. Koeppl and L.O. Chua, "An Adaptive Cellular Nonlinear Network and its Application," Proc. of NOLTA'07, pp. 15-18, Sep. 2007.

[5] R. Perfetti, E. Ricci, D. Casali, and G. Costantini "Cellular Neural Networks With Virtual Template Expantion for Retinal Vessel Segmentation," IEEE Trans. Circuits Syst. I, vol. 54, pp. 141-145, Feb. 2007.

[6] T. Szabó and P. Szolgay, "An Analogic Diffusion-Based Algorithm for the Segmentation of CT Images," Proc. of NOLTA'07, pp. 237-239, Sep. 2003.

[7] C.T. Lin, C.H. Huang and S.A. Chen, "CNN-Based Hybrid-Order Texture Segregation as Early Vision Processing and its Implementation on CNN-UM," IEEE Trans. Circuits Syst. I, vol. 54, pp. 2277-2287, Oct. 2007.

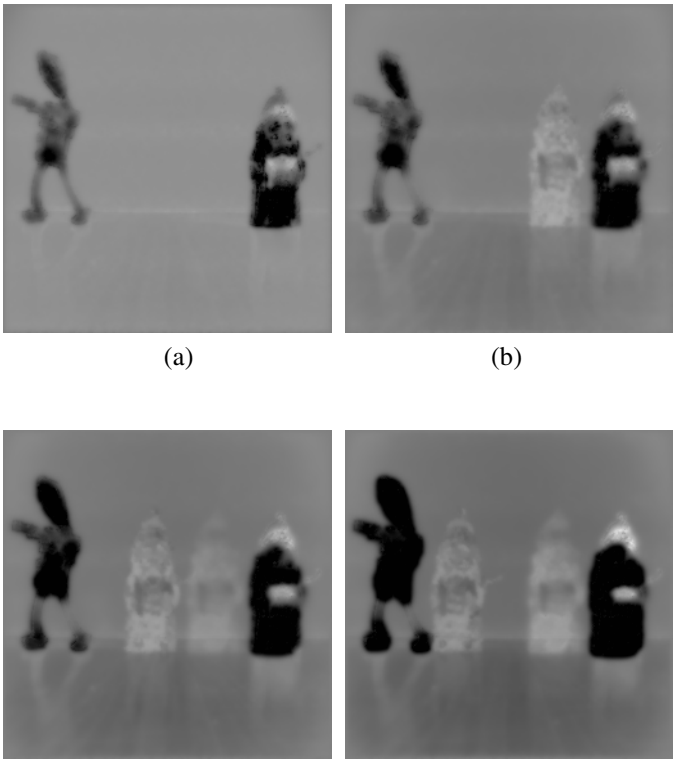


Fig. 7. Simulation results using D-CNN for Fig. 5 (R_{10}=0.1). (a) Output image 1 ([τ]=10). (b) Output image 2 ([τ]=20). (c) Output image 3 ([τ]=30). (c) Output image 4 ([τ]=40).

[8] M. Kawahara, T. Inoue and Y. Nishio, "Cellular Neural Network with Dynamic Template and its Output Characteristics," Proc. of IJCNN'09, pp. 1552-1558, Jun. 2009