Paper

Associative dynamics of color images in a large-scale chaotic neural network

Makito Oku and Kazuyuki Aihara

1 Department of Mathematical Informatics, Graduate School of Information Science and Technology, The University of Tokyo 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
2 Institute of Industrial Science, The University of Tokyo 4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan

a) oku@sat.t.u-tokyo.ac.jp

Received December 14, 2010; Revised May 19, 2011; Published October 1, 2011

Abstract: In this paper, we report a way to store color images in a large-scale chaotic neural network and to retrieve them by using chaotic dynamics. In the proposed method, color images are converted to binary codes, modified slightly by inverting a few bits, and stored in the network. The results of numerical simulations show that chaotic transitions among stored patterns and their reverse patterns can be observed within a certain range of parameters. We also compare five different coding schemes of color information, which change the appearance of chaotic dynamics. In addition, if connections are restricted in a neighborhood of each unit, a variety of wave patterns are observed.

Key Words: chaotic neural network, image processing, associative memory, large-scale simulation, chaotic itinerancy, traveling wave

1. Introduction

Large-scale simulations of neural networks are getting popular in the last few decades [2–6]. The number of units has reached one million [5], and it is expected to get increased more in future. In such a background, there are the progress of technology that makes large-scale simulations practical and the growing expectation for new findings that would be revealed by large-scale simulations.

Recently, our group succeeded in simulating a large-scale chaotic neural network with one million units [7]. A characteristic feature of the chaotic neural network model [8–11] is that the network's state chaotically wanders among multiple stored patterns. The model is used for the phenomenological modeling such as the dynamical associative memory [10, 12, 13], the endogenous perceptual alternation [14, 15], and birdsong learning [16]. Furthermore, it is also applicable to engineering problems such as pattern recognition [17, 18], image segmentation [19], combinatorial optimization [20–27], and encoding of digital information [28]. Although large-scale chaotic neural networks have been investigated primarily in the context of optimization problems solving [24, 27] or electrical circuits.

A preliminary version of this paper was presented at the 2010 International Symposium on Nonlinear Theory and its Applications (NOLTA 2010) and was published in its proceedings [1]. Major modifications are to add simulation results in the cases of reversible code and neighboring connections.
implementation [29], no one has investigated a large-scale dynamical associative memory model.

We found in our previous study [7] that the network exhibits chaotic itinerancy [30–32] among the stored patterns. We also found that even an incompletely retrieved image—the network output differs from the correct image at many pixels—can evoke a clear perception of the original image. Furthermore, we observed traveling waves in the network if connections are restricted in a neighborhood of each unit. Because of these features, a large-scale dynamical associative memory model appeared to be a useful tool to deepen our insight into complex nonlinear phenomena of neural networks such as chaotic itinerancy and traveling waves.

In this study, we store color images in a large-scale chaotic neural network and investigate how the network retrieves them by using chaotic dynamics. Since the associative memory model can store only binary patterns, the conversion of color images to binary codes is required. We first attempt the most simple and direct way of conversion and then carry out other four conversion methods. After that, we also investigate the case of a locally connected network.

2. Chaotic neural network model

First, we explain the chaotic neural network model [8–11]. It is a map-based model that derives from a series of studies—the McCulloch-Pitts model [33], the Caianiello’s model [34], and the Nagumo-Sato model [35]. As its name implies, this model qualitatively reproduces chaos in nerve membranes observed in squid giant axons [36] and the Hodgkin-Huxley model [37].

In this study, we consider a recurrent neural network with \( N \) units. The external input is constant both in time and spatial domains. Each unit has two internal state variables \( \eta_i \) and \( \zeta_i \) and one output variable \( x_i \). If we adopt the vector representation, \( \eta = \{\eta_1, \ldots, \eta_N\}^T \), \( \zeta = \{\zeta_1, \ldots, \zeta_N\}^T \), and \( x = \{x_1, \ldots, x_N\}^T \), the system’s dynamics can be described by the following difference equations:

\[
\begin{align*}
\eta(t + 1) &= k_f \eta(t) + W x(t), \\
\zeta(t + 1) &= k_r \zeta(t) - \alpha x(t) + a, \\
x(t + 1) &= f(\eta(t + 1) + \zeta(t + 1)).
\end{align*}
\]

Here, \( W \) denotes the \( N \times N \) weight matrix; \( k_f, k_r \in [0, 1] \), the time constants; \( \alpha \geq 0 \), the strength of the refractoriness; and \( a \), which is a vector of constant values \( a \geq 0 \), the bias that includes the external input and the threshold. The first internal state variable \( \eta \) changes in response to the input from other units. The other internal state variable \( \zeta \) depends on the output of each unit reflecting its refractoriness. The output \( x \) is defined by a nonlinear function of the summation of the internal state variables. Here, \( f \) is an operation that applies the following sigmoid function to each element of the argument vector:

\[
f(y) = \frac{1}{1 + \exp(-y/\epsilon)},
\]

where \( \epsilon \) denotes the steepness parameter.

Because of the continuous activation function, the output \( x \) takes analog values. One interpretation [38] is that they represent graded action potentials [39] or firing rates. From another standpoint [40], the continuous function is introduced for theoretical tractability, but the essential features of neuronal dynamics can be kept if \( \epsilon \) is sufficiently small. By using the continuous activation function, we can analyse orbital instability with Lyapunov exponents [10].

Next, we explain the associative memory model [41–44]. In general, the associative memory model has two phases: the encoding phase and the retrieval phase. In the encoding phase, \( K \) binary patterns \( s^k = \{s^k_1, \ldots, s^k_N\}^T \), \( s^k_i \in \{-1, 1\}, (k = 1, \ldots, K) \) are given. For simplicity, we assume that each pattern contains equal numbers of 1 and \(-1\). Then, the weight matrix is determined by the autocorrelation matrix of the patterns as follows:

\[
W = \frac{1}{K} \sum_{k=1}^{K} s^k (s^k)^T.
\]
In the retrieval phase, one of the stored patterns with perturbation is given as an initial state. Then, the corresponding stored pattern is recovered in finite steps. This phenomenon is called pattern completion. From a dynamical systems viewpoint, each stored pattern corresponds to a stable equilibrium point.

The dynamical associative memory model [10] differs from the conventional associative memory model in that the system’s state is attracted to a stored pattern for a short period, but leaves the pattern after a while, and then visits other patterns. It has been hypothesized [45, 46] that such transitory dynamics of neural networks may ensure rapid access to previously learned patterns when a similar input pattern to them is given. In addition, when stimulated by a novel input, the network shows highly chaotic state that may facilitate learning of the novel input. Recently, it is also experimentally observed that the ongoing activity in the visual area of cats exhibits spontaneous transitions between different patterns [47].

The chaotic neural network model exhibits such behavior in some parameter regions. Neither the order of visiting nor the duration of each stay is predictable although the system’s dynamics is deterministic. This phenomenon is also an example of chaotic itinerancy [30–32]. It is also suggested that, if the stored patterns are mutually orthogonal, there exist a number of invariant subspaces with symmetrical and hierarchical structure in the network dynamics [48]. Such a structure may play a key role for the rich dynamical behavior.

Many previous works on the dynamical associative memory model use approximately 100 units [10, 12, 13]. In order to perform large-scale simulations, one of the major problems is the requirement of a considerably large amount of memory capacity for representing the weight matrix, which increases in $O(N^2)$. However, the all-to-all connection regime used in the associative memory model has high redundancy; even if we remove some amount of the network connections, the qualitative property of dynamics can be retained. Therefore, we use a partially connected network [49, 50] in our simulations. The weight matrix $W = \{w_{ij}\}$ is then changed as follows:

$$w_{ij} = \begin{cases} \frac{1}{K} \sum_{k=1}^{K} s_i^k s_j^k & e_{ij} \in E \\ 0 & \text{otherwise} \end{cases},$$

where $e_{ij}$ denotes a connection from unit $j$ to unit $i$, and $E$ denotes the set of all the connections. Notice that if $K$ is even, $w_{ij}$ can take 0 even if $e_{ij} \in E$. In such cases, the number of non-zero elements of $W$ is less than $|E|$.

### 3. Conversion of color images to binary codes

In our previous study [7], binary images of 1 000 × 1 000 pixels were transformed to $10^6$-dimensional vectors by concatenating all rows and stored in the network. Although binary images are useful for investigating the dynamical associative memory model, applications of the model to multivariate images such as gray-scale images and color images should be appreciated from a practical point of view. In this paper, we only consider color image processing. The same techniques as those proposed here can be applied to gray-scale images as well in a straightforward manner.

We first introduce the most simple and direct way of conversion. The RGB (red, green, and blue) color space is one of the basic color image representations. RGB values are normally represented by integer values from 0 to 255. Each value is encoded in 8 bits; 24 bits represent the complete information of a pixel. By concatenating these 24-bit binary codes, we can directly convert the 24-bit RGB color images to binary codes of length $24 \times (\text{the number of pixels})$.

<table>
<thead>
<tr>
<th>Statistics</th>
<th>Target value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sum_{i=1}^{N} s_i^k$</td>
<td>0</td>
</tr>
<tr>
<td>$\sum_{i=1}^{N} s_i^k s_i^l$</td>
<td>0.08N</td>
</tr>
<tr>
<td>$\sum_{i=1}^{N} s_i^k s_i^l s_i^m$</td>
<td>$-0.08N$</td>
</tr>
</tbody>
</table>

Table 1. List of adjusted statistics of the stored patterns.
The obtained binary codes of 1 and 0 are transformed to those of 1 and −1, and then their statistics, as shown in Table I, are adjusted by inverting bits with the number as small as possible. In particular, lower bits in each byte are preferentially selected for the adjustment because they have smaller effects to the original values. This preprocessing balances the numbers of 1 and −1 in each binary pattern as well as equalizes the amount of overlaps among patterns. We do not make the patterns completely orthogonal to each other. Rather, we make any two patterns weakly correlated. We also control the relationship among three patterns. These adjustments are perhaps necessary for the emergence of chaotic behavior. However, the relationship between the statistics of patterns and chaotic behavior is not yet clear.

In addition to the abovementioned conversion method, we carry out four other methods. Two of them are the usage of different color spaces, and the other two are the usage of different binary numeral systems.

First, we use the \( YIQ \) color space, which is used by the television system in the U.S. and Japan. \( YIQ \) values of an image are obtained from \( RGB \) values of the image by using the linear transformation as follows [51]:

\[
\begin{pmatrix}
Y \\
I \\
Q
\end{pmatrix} = \begin{pmatrix}
0.2990 & 0.5870 & 0.1140 \\
0.5957 & -0.2745 & -0.3213 \\
0.2115 & -0.5226 & -0.3111
\end{pmatrix}
\begin{pmatrix}
R \\
G \\
B
\end{pmatrix},
\]

(7)

where \( RGB \) values are scaled from 0 to 1. \( Y \in [0, 1] \), \( I \in [-0.5958, 0.5957] \), and \( Q \in [-0.5226, 0.5226] \). The first component \( Y \) provides the luminance value, and the other two components describe the chrominance information.

In our simulations, \( YIQ \) values that are converted from the original \( RGB \) values are scaled from 0 to 255 and quantized. Then, we obtain a 24-bit binary code per pixel as in the case of \( RGB \) images.

Second, we use \( HSV \) (hue, saturation, and value), also known as \( HSB \) (hue, saturation, and brightness), which is commonly used in image processing because it is more natural to human vision. In the determination of \( HSV \), we consider a cylindrical coordinate because hue is a circular quantity. Let \( M = \text{max}(R, G, B) \), \( m = \text{min}(R, G, B) \), and \( C = M - m \). If \( R \neq G \), \( G \neq B \), and \( B \neq R \) hold (thus \( M, C \neq 0 \)), the \( HSV \) color space is defined as follows [52]:

\[
H = \begin{cases}
60^\circ (G - B)/C \mod 360^\circ & (M = R) \\
60^\circ (B - R)/C + 120^\circ & (M = G) \\
60^\circ (R - G)/C + 240^\circ & (M = B)
\end{cases}
\]

(8)

\[
S = C/V, \quad V = M.
\]

(9)

(10)

In addition, \( S = 0 \) if \( C = 0 \), \( V \neq 0 \), and \( V = 0 \) if \( M = 0 \). \( H \in [0^\circ, 360^\circ] \), \( S \in [0, 1] \), and \( V \in [0, 1] \). Usually a certain value is assigned if a value is undefined (\( H = 0 \) if \( C = 0 \), for example). As in the case of \( YIQ \) values, \( HSV \) values are scaled from 0 to 255 and quantized in order to obtain a 24-bit binary code per pixel.

Third, we use the gray code. The color space is \( RGB \), but the gray code is used instead of the ordinary binary code for transforming 0–255 integer values into 8-bit codes. We refer to the ordinary binary code as the binary code in the following text. In the gray code, every two successive values differs only in one bit. A binary code representation of an integer \( n \) can be converted to a gray code representation by \( n \oplus \lfloor n/2 \rfloor \), where \( \oplus \) denotes an XOR operation.

Finally, we use another binary numeral system in which any 8-bit code and its inverted code represent the same value, which we call the reversible code. This code can represent effectively 128 tones. To transform a 0–127 integer value into an 8-bit code, first we take the 7-bit binary code representation. Then, we add a random bit to the head of it. If it is 1, the remaining 7 bits are inverted. Otherwise, they are unchanged. On the other hand, when we evaluate an 8-bit code, the last 7 bits are inversely interpreted if the most significant bit is 1. This code corresponds to taking the absolute value of the one’s complement system.

511
In each of the five conversion methods, network output can be decoded in the opposite manner. If the recovered \( RGB \) values are outside the specified range, we reset the values larger than 255 to 255 and the negative values to 0.

4. Simulations and results

In the following simulations, we set \( k_f = 0.8 \), \( k_r = 0.9 \), \( a = 6.4 \), \( \alpha = 12 \), and \( \epsilon = 0.015 \) unless otherwise noted. Each unit receives input from 100 units selected at random. These connections include those of value 0, which are removed before carrying out the simulations. As an initial condition, \( \eta_i(0) \) takes a random value that is uniformly distributed in \([0,1]\), and \( \zeta_i(0) \) is set to 0.

The source code of the program is written in the C programming language with Message Passing Interface (MPI). The program is run on a cluster of four Linux server machines that have two 3.0 GHz dual-core processors and 8.0 GB RAM each. It takes approximately 1 second to compute one step of simulation.

As stored patterns, we select four standard color test images from the USC-SIPI image database [53], as shown in Fig. 1. The color information is represented in the \( RGB \) space in the original images, and the \( RGB \) values range from 0 to 255. We resize them to \( 256 \times 256 \) pixels, and convert them to binary codes. Thus, the number of units, or the length of the binary codes, is \( 24 \times 256^2 = 1,572,864 \).

![Fig. 1. Original color images for stored patterns [53].](image-url)

![Fig. 2. (A) An example of time series of the decoded network output displayed with 10 step intervals. (B) Representative snapshots corresponding to almost complete retrieval of the stored patterns and the reverse ones.](image-url)
First, we directly convert the color images, which is represented in the RGB space, by using binary code. In the preprocessing, 1.1–3.9% (Ave. 2.6%) of the bits in each binary code are inverted. The root-mean-square (RMS) error per color component of a pixel is 0.67.

Figure 2 shows a typical time series of the decoded network output. The four stored patterns and their reverse patterns are retrieved successively. A few cycles of oscillation between a stored pattern and its reverse one are also observed.

To analyze the time series of the network output $x(t)$, we calculate the Hamming distances between the quantized network output and the four binary codes (see Fig. 3). The downward and upward peaks correspond to the retrievals of the stored patterns and the reverse ones, respectively. Such
chaotic behavior can continue for a long time, at least $10^5$ steps, as shown in Fig. 4. The frequency distribution of the retrieval of the patterns is apparently not uniform, and it also depends on the initial conditions, i.e., the most frequently retrieved pattern differs with different initial conditions. On the other hand, the maximum Lyapunov exponent does not vary much with initial conditions, which is estimated to be $0.664 \pm 0.003$.

Next, we try the other four coding schemes: the $YIQ$ space with binary code, the $HSV$ space with binary code, the $RGB$ space with gray code, and the $RGB$ space with reversible code. The RMS errors due to the preprocessing are 1.58, 1.23, 0.70, and 1.36 for the cases of the $YIQ$, $HSV$, gray code, and reversible code, respectively. Chaotic behavior similar to the previous case is observed in all the four cases, as shown in Fig. 5.

The appearance of the decoded network output is also similar to that in the previous case when stored patterns are retrieved, but qualitatively different when reverse patterns are retrieved except in

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{images.png}
\caption{Comparison of different encoding schemes. The decoded network output is displayed with 10 step intervals. (A) $YIQ$ with binary code, (B) $HSV$ with binary code, (C) $RGB$ with gray code, and (D) $RGB$ with reversible code.}
\end{figure}
the YIQ case, as shown in Fig. 6. In the HSV case, the reverse patterns tend to be dark. In the case of the gray code, the reverse patterns have some discontinuous changes of color (shadow in the hat, for example). On the other hand, in the case of the reversible code, the appearance of a stored pattern and that of the reverse pattern are the same.

Finally, we investigate another network structure where each unit receives input only from its neighboring region, as shown in Fig. 7(A). The neighboring region is a square region centered at the pixel related to the target unit, and the region contains \(2d(2d+1)^2\) units in total. From that region, 441 units selected at random project to the target unit. The boundary condition is periodic, and the network structure is topologically equivalent to a torus. The images are partitioned into blocks of size 16×16 pixels, and in each block the adjustment of statistics of patterns is performed. As in the case of the previous simulations, the connections with value 0 are removed before carrying out the simulations. Initial conditions are also the same to the previous cases. Parameters are set to \(k_f = 0.7\), \(k_r = 0.9\), \(a = 6.4\), \(\alpha = 20\), and \(\epsilon = 0.015\). The RGB color space with reversible code is used.

Figures 7(B–E) show the representative patterns of the decoded network output. We observe
multiple clusters with moving boundaries in the transient dynamics, as shown in Fig. 7(B). These clusters merge over time, and eventually some sort of stationary wave pattern emerges. For example, plane waves moving in the same direction, two groups of such waves crossing each other, and spiral waves can be observed with different initial conditions, as shown in Figs. 7(C–E). We also find that spiral waves emerge in pairs that are spinning in the opposite directions. The maximum Lyapunov exponent varies with initial conditions, but it is always estimated to be positive regardless of the eventual wave patterns.

5. Discussions

In this section, we discuss what we found in the numerical simulations. First, it has been confirmed that chaotic transitions among stored patterns and their reverse patterns can be observed if we store color images in the network, as shown in Figs. 2 and 3. The positive maximum Lyapunov exponent is another evidence for the existence of chaos, and the lifetime of the chaotic behavior in the considered network appeared to be significantly longer than that of a smaller network [10], as shown in Fig. 4.

The frequency distribution of the retrieval of the patterns is apparently not uniform and depends on the initial conditions despite of the careful adjustment of the statistics of the patterns. This may be because each trajectory of the network state might be restricted in a particular subspace dependent on the initial state [54]. Thus, a specific pattern and its reverse one are more frequently retrieved than the others. On the other hand, there is another possibility that the apparent bias in the frequency distribution may disappear in a much longer time scale.

In addition, it appeared that the color images decoded from the network output, as shown in Fig. 2, can evoke a clear perception of the original images even when the retrieval is not complete and the Hamming distance between the quantized network output and the corresponding binary code is not so small. These observations are also consistent with the results of our previous work in which binary images were used as stored patterns [7]. The reason is probably that the human vision would be less sensitive to the perturbation in the image with higher resolution, as demonstrated in Fig. 8.

Next, we found that the use of different color coding schemes can change the appearance of chaotic dynamics, as shown in Fig. 5. In particular, the appearance of reverse patterns differs significantly, as shown in Fig. 6. This is probably because the bitwise inversion operation may have a qualitatively different effect under different coding schemes in general. For example, the operation maps a 0–255 integer value $n$ to $255 - n$ if the binary code is used. Thus, the effect of the operation obviously depends on the coordinate system.

However, we have obtained quite similar reverse patterns in the RGB and YIQ spaces. The reason why YIQ does not change the appearance can be explained as follows. In our simulation, the YIQ values obtained by using Eq. (7) are scaled from 0 to 255 as follows:
where $\tilde{Y}$, $\tilde{I}$, and $\tilde{Q}$ denote the rescaled values. For simplicity, here we do not consider quantization of the values. Then, the bitwise inverted values in the YIQ space are described as $(Y_{\text{inv}}, I_{\text{inv}}, Q_{\text{inv}})^T = (255 - \tilde{Y}, 255 - \tilde{I}, 255 - \tilde{Q})^T$. Let $A$ and $S$ denote the translation matrix in Eq. (7) and the scaling matrix in Eq. (11), respectively. If we move back to the RGB space again, the corresponding values are as follows:

$$ A^{-1}S^{-1}\left\{ \frac{1}{255} \begin{pmatrix} Y_{\text{inv}} \\ I_{\text{inv}} \\ Q_{\text{inv}} \end{pmatrix} - \begin{pmatrix} 0 \\ 0.5 \\ 0.5 \end{pmatrix} \right\} = A^{-1}S^{-1}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix} \approx \begin{pmatrix} 1 - R \\ 1 - G \\ 1 - B \end{pmatrix}. $$

(12)

These values indeed correspond to the bitwise inverted values in the RGB space, given that here the RGB values are scaled to $[0, 1]$. Therefore, the bitwise inversion operation in the YIQ space happens to have the same effect as in the RGB space. Interestingly, the same is true for few other linear transformations of the RGB space such as YUV and $I_1I_2I_3$. However, a linear transformation in general does not have such a property.

On the other hand, we have obtained darker reverse patterns in the HSV space than in the RGB space. We found that this tendency is not specific to the input images. Rather, the reason is probably that relatively dark color occupies a large volume in the HSV space. Thus, the bitwise inversion operation in the HSV space tends to decrease brightness of images. Also, the discontinuous color changes in the case of the gray code is not specific to the input images. It is probably because, unlike the binary code, the bitwise inversion operation to the gray code maps some two successive values to largely different ones.

We suggest that all the proposed coding schemes except YIQ are equally appropriate for the visualization of the chaotic dynamics. The best coding scheme may depend on images and purposes of the visualization. However, if one tries to store many images to the network, the reversible code would be most suitable. The reversible code has a unique feature that the statistics of the binary codes converted from the original images are effectively the same with those of random patterns. This is because the most significant bit in each 8-bit code can be randomly assigned independently of the input images, and thus whether the remaining 7 bits are inverted or unchanged is randomly decided. This feature is beneficial when the number of images is large, because in that case the image preprocessing does not work well.

Finally, we found that if connections are restricted in a neighborhood of each unit, a variety of wave patterns emerge, as shown in Fig. 7. As far as we know, we are the first to report the emergence of traveling waves in a locally connected associative memory model. It is not surprising that this observation is consistent with many other researches. In fact, traveling waves or propagating waves are ubiquitous phenomena in the real neural circuits [55–65]. In addition, many theoretical analyses and numerical simulations [2, 3, 5, 66–71] also report the possibility of the emergence of traveling waves in locally connected neural networks. We believe that large-scale numerical simulations like ours are especially useful for understanding of traveling waves, because the phenomena involve a large number of units and has a long time scale. In particular, spiral waves are suggested to be observable only in large size networks [3].

The observation of traveling waves in the associative memory model is important for two reasons. First, it provides a new mechanism of the spontaneous state transitions in the associative memory model. The standard view is that the network as a whole changes to another state, whereas the new mechanism is that traveling waves continuously elicit local alternations of cell assemblies. Indeed, spontaneous state transitions are observed in the real brain, and the associative memory model are often used to interpret such phenomena. Thus, the new mechanism based on traveling waves should be worth considering. Second, the traveling waves observed in this study transmit a higher-order statistical properties with the spatial activity patterns of neurons, instead of population firing rates.

\[
\begin{pmatrix}
\tilde{Y} \\
\tilde{I} \\
\tilde{Q}
\end{pmatrix} = 255\left\{ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/1.1914 & 0 \\ 0 & 0 & 1/1.0452 \end{pmatrix} \begin{pmatrix} Y \\ I \\ Q \end{pmatrix} + \begin{pmatrix} 0 \\ 0.5 \\ 0.5 \end{pmatrix} \right\},
\]

(11)
or phases of oscillations. This suggests that such higher-order properties may also be propagated in the real brain, which would be hard to detect in spatially smoothed signals such as local field potentials.

6. Conclusion
In conclusion, we have stored 24-bit RGB color images of 256×256 pixels in a large-scale chaotic neural network with approximately 1.6 million units. To convert the color images represented by integer values from 0 to 255, we used an 8-bit binary representation. We have observed chaotic transitions among the stored images and their reverse images. We have also found that the use of different color coding schemes can change the appearance of chaotic dynamics. Furthermore, we have observed a variety of wave patterns if units are only locally connected. As a future work, it would be valuable if more images can be dealt with by the network.

Acknowledgments
This research is partially supported by the Japan Society for the Promotion of Science, a Grant-in-Aid for JSPS Fellows (21-937) and FIRST Aihara Innovative Mathematical Modelling Project.

References


“The USC-SIPI image database.” (http://sipi.usc.edu/database/).


H.R. Wilson and J.D. Cowan, “A mathematical theory of the functional dynamics of cortical


