Biological Cybernetics (2018) 112:495–508
https://doi.org/10.1007/s00422-018-0775-9

Biological Cybernetics

ORIGINAL ARTICLE

Application of chaos in a recurrent neural network to control in ill-posed problems: a novel autonomous robot arm

Seiji Kuwada[1] · Tomoya Aota[1] · Kengo Uehara[1] · Shigetoshi Nara[1]

Received: 18 May 2016 / Accepted: 6 August 2018 / Published online: 20 August 2018

Abstract
Inspired by a viewpoint that complex/chaotic dynamics would play an important role in biological systems including the brain, chaotic dynamics introduced in a recurrent neural network was applied to robot control in ill-posed situations. By computer experiments we show that a model robot arm without an advanced visual processing function can catch a target object and bring it to a set position under ill-posed situations (e.g., in the presence of unknown obstacles). The key idea in these works is adaptive switching of a system parameter (connectivity) between a chaos regime and attractor regime in a neural network model, which generates, depending on environmental circumstances, either chaotic motions or definite motions corresponding to embedded attractors. The adaptive switching results in useful functional motions of the robot arm. These successful experiments indicate that chaotic dynamics is potentially useful for practical engineering control applications. In addition, this novel autonomous arm system is implemented in a hardware robot arm that can avoid obstacles and reach for a target in a situation where the robot can get only rough target information, including uncertainty, by means of a few sensors, as indicated in the appendix, A1 and A2.

Keywords Robot arm · Adaptive control · Functional chaos · Recurrent neural network · Ill-posed problem

1 Introduction

For the last few decades, biological science and brain science have made tremendous advances. However, the mechanisms of advanced biological functions remain beyond our understanding. Generally speaking, it is well known that biological systems have excellent functions not only in information processing but also in well-regulated control, which works adaptively in various environments. However, it is quite difficult to understand biological control mechanisms using conventional methodologies, even in a small insect (Huber and Thorson 1985).

When we try to understand these mechanisms with successive operations of decomposed parts or elements based on certain neuron models, then, in the face of immense complexity in the behavioral function systems, such a decompositional method will more or less encounter two difficulties: one is a *combinatorial explosion* and the other is a *divergence of algorithmic complexity*. Unfortunately, these difficulties have not yet been overcome.

On the other hand, chaotic dynamics experimentally observed in biological systems, including brains, have attracted great interest (Hayashi et al. 1982; Babloyantz and Destexhe 1986; Skarda and Freeman 1987; Arhem et al. 2000). These articles suggest that biological functions could work under a novel *dynamical mechanism* in information processing and control. On this viewpoint, chaotic dynamics in artificial neural networks has also attracted great interest in relation to neuroscience and brain science (Skarda and Freeman 1987; Aihara et al. 1990; Tsuda 1991, 2001; Kaneko and Tsuda 2003; Fujii et al. 1996; Liljenström 1995). More generally, not only chaos in the brain (Yao and Freeman 1990) but also the effects of disorder in the brain have been considered by many researchers, for example in an intensive book edited by Arhem, Blomberg, and Liljenström, (Arhem et al.

Communicated by J. Leo van Hemmen.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s00422-018-0775-9) contains supplementary material, which is available to authorized users.

✉ Shigetoshi Nara
nara@ec.okayama-u.ac.jp

[1] Department of Electrical and Electronic Engineering, Graduate School of Natural Science and Technology, Okayama University, 3-1-1 Tsushima-naka, Kita-ku, Okayama 700-8530, Japan

2000), as well as from the viewpoint of complex dynamics (Liljenström 1995).

Nara and Davis studied chaotic dynamics in neural networks and other systems from a functional viewpoint and proposed a novel idea to harness the onset of complex nonlinear dynamics in information processing or control systems (Nara and Davis 1992). In particular, Nara and Davis introduced chaotic dynamics into a recurrent neural network (RNN) consisting of binary neurons by means of adjusting only one system parameter (input connectivity from other neurons), and they proposed that constrained chaos might be useful dynamics to solve complex problems, such as ill-posed problems. As a prototype functional experiment, they applied chaotic dynamics to solving, for instance, a memory search task set in an ill-posed context (Nara and Davis 1992, 1997; Nara et al. 1993, 1995; Kuroiwa et al. 1999; Nara 2003). The memory search task was executed by switching between two dynamical regimes. One was a regime where memories are stable attractors, and the other a regime where there is chaotic wandering among memories. Moreover, the idea has been extended to challenging applications of chaotic dynamics in control systems. Chaotic dynamics introduced into RNNs was applied to controlling the movement of an object through a two-dimensional maze (labyrinth) to reach for a target (Suemitsu and Nara 2004) or to reach for a target moving along different trajectories (Li et al. 2008; Yoshinaka et al. 2012). In these articles, a simple coding method is employed for a projection of the higher-dimensional neural state dynamics to lower-dimensional motion increments. A simple control algorithm is proposed whose key point is that, by means of adaptive switching between two regimes, a weakly chaotic (or strange attractor) regime and a strongly chaotic regime, complex and ill-posed problems can be solved. The aforementioned control tasks were successfully executed in computer experiments and in a hardware implementation as well (Yoshinaka et al. 2012). The results showed that constrained chaotic behaviors can give better performance in solving these ill-posed problems than random jumping in state space.

In this paper, we extend the aforementioned approach to a different control problem, the control of an robot arm. Thus, we give further evidence that chaotic dynamics could be useful in robotics and biological control systems. We mainly consider the control of a single robot arm and hardware implementation; we also briefly touch on the control of a pair of competing robot arms. In the first control task, a robot arm tries to catch a (static or moving) target object and bring it to a set position under ill-posed circumstances. That is, (a) it has no advanced visual processing function, so it can only get rough directional information on the target including uncertainty, and (b) there exist unknown obstacles, which means that knowledge about the obstacles is not given to the robot arm. We show that our computer experiments were successful and also that the theoretical systems were successfully implemented in a hardware robot arm and in the present technical stage can actually avoid obstacles and reach for the target. For the problem of two robot arms, we show only preliminary results.

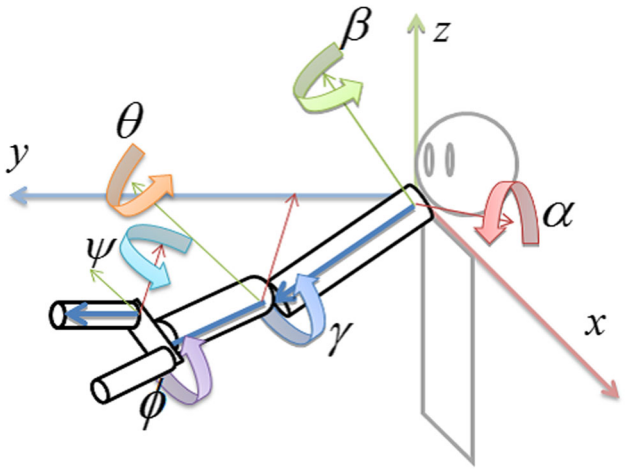


Fig. 1 Our arm model in Euler angle scheme using a left-handed coordinate system

2 An autonomous robot arm system driven by a neural network model

2.1 Construction of arm system

Our arm model is shown in Fig. 1. It has six degrees of freedom, and each is quantitatively specified using an Euler angle scheme as shown in the figure. The components of the arm model are the shoulder (the origin of upper arm rotation with three angles, α, β, γ, and length l_u), elbow (the origin of lower arm rotation with two angles, θ, ϕ, and length l_l), wrist and its coplanar two fingers (1 angle, ψ between them to grip an object, and the finger length l_f, the hand width l_w), where wrist bending is not introduced in this work. For convenience of description, we employed a *right-handed coordinate system*.

The arm positions during arbitrary motions are calculated from the initial position,

$$\boldsymbol{r}_0 = l_u \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + l_l \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + l_w \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + l_f \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix},$$

shown in Fig. 2, by successive transformations consisting of rotations with respect to the given axes defining Euler angles and corresponding spatial translations to the positions of the elbow and wrist. Note that the figure was drawn using *InsilicoIDE*, which is free software on a platform called *Physiome*

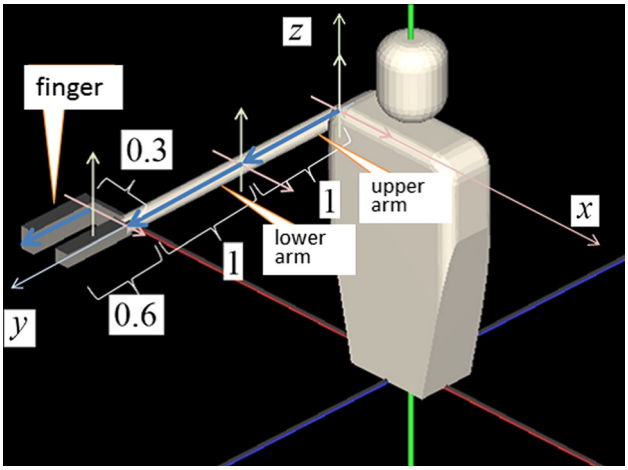


Fig. 2 Initial position and attitude. The numerical values of each length, $l_u = 1.0, l_l = 1.0, l_w = 0.3, l_f = 0.6$, adopted in this work are shown. The figure is drawn using *InsilicoIDE* (see reference in text)

(Physiome 2012). Explicit descriptions of transformations are written as follows. First, let us consider transformations of the coordinate system due to rotating operations (Fig. 1). In the initial coordinate system fixed to each part of the arm, the three orthogonal unit vectors are written $[\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z]$, which are represented as column vectors,

$$\boldsymbol{e}_x =^t [1,0,0]\,, \boldsymbol{e}_y =^t [0,1,0]\,, \boldsymbol{e}_z =^t [0,0,1]\,,$$

where t means transpose. When we rotate this coordinate system with respect to the origin around an arbitrary axis represented as a unit vector, $\boldsymbol{q} =^t [\lambda, \mu, \nu]$, with a rotating angle, θ, then the transformed orthogonal unit vectors are

$$\left[\boldsymbol{e}_x^\theta, \boldsymbol{e}_y^\theta, \boldsymbol{e}_z^\theta\right] = \boldsymbol{M}(\boldsymbol{q},\theta)\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}, \tag{1}$$

where $\boldsymbol{M}(\boldsymbol{q},\theta)$ is the rotation matrix, given as

$$\boldsymbol{M}(\boldsymbol{q},\theta) = \begin{bmatrix} \cos\theta + \lambda^2(1-\cos\theta) & \lambda\mu(1-\cos\theta) - \nu\sin\theta & \nu\lambda(1-\cos\theta) + \mu\sin\theta \\ \lambda\mu(1-\cos\theta) + \nu\sin\theta & \cos\theta + \mu^2(1-\cos\theta) & \mu\nu(1-\cos\theta) - \lambda\sin\theta \\ \nu\lambda(1-\cos\theta) - \mu\sin\theta & \mu\nu(1-\cos\theta) + \lambda\sin\theta & \cos\theta + \nu^2(1-\cos\theta) \end{bmatrix}.$$

The vector from shoulder to elbow (the upper arm vector) in the initial attitude is represented as $\boldsymbol{r}_{u0} = \boldsymbol{e}_y l_u$, where l_u is the length of the upper arm. Thus, the transformed vector due to a movement of the upper arm is calculated as

$$\boldsymbol{R}_u = \boldsymbol{M}\left(\boldsymbol{e}_z^\alpha, \beta\right)\boldsymbol{M}\left(\boldsymbol{e}_x, \alpha\right)\boldsymbol{e}_y l_u, \tag{2}$$

where $\boldsymbol{e}_z \to \boldsymbol{e}_z^\alpha$ indicates the rotation axis (unit) vector of the second rotation, β, after the first rotation of Euler angle α. The third rotation with respect to the transformed y-axis ($\boldsymbol{M}(\boldsymbol{e}_y^{\beta\alpha}, \gamma)$ corresponding to $\boldsymbol{e}_y \to \boldsymbol{e}_y^{\beta\alpha}$) is executed; however, it does not affect the attitude of the upper arm, so it is abbreviated. Using the same styles of representation, we can specify the attitude of the lower arm, $\boldsymbol{R}_l$, as

$$\boldsymbol{R}_l = \boldsymbol{M}\left(\boldsymbol{e}_z^{\gamma\beta\alpha}, \theta\right)\boldsymbol{M}\left(\boldsymbol{e}_y^{\beta\alpha}, \gamma\right)\boldsymbol{M}\left(\boldsymbol{e}_z^\alpha, \beta\right)\boldsymbol{M}\left(\boldsymbol{e}_x, \alpha\right)\boldsymbol{e}_y l_l + \boldsymbol{R}_u. \tag{3}$$

Similarly, the attitude vectors of the wrist and finger are

$$\begin{aligned}\boldsymbol{R}_w = {} & \boldsymbol{M}\left(\boldsymbol{e}_y^{\theta\gamma\beta\alpha}, \phi\right)\boldsymbol{M}\left(\boldsymbol{e}_x^{\gamma\beta\alpha}, \theta\right)\boldsymbol{M}\left(\boldsymbol{e}_y^{\beta\alpha}, \gamma\right)\boldsymbol{M}\left(\boldsymbol{e}_z^\alpha, \beta\right) \\ & \times \boldsymbol{M}\left(\boldsymbol{e}_x, \alpha\right)\boldsymbol{e}_y l_w + \boldsymbol{R}_l, \end{aligned} \tag{4}$$

$$\begin{aligned}\boldsymbol{R}_{fi} = {} & \boldsymbol{M}\left(\boldsymbol{e}_z^{\phi\theta\gamma\beta\alpha}, \psi\right)\boldsymbol{M}\left(\boldsymbol{e}_y^{\theta\gamma\beta\alpha}, \phi\right)\boldsymbol{M}\left(\boldsymbol{e}_x^{\gamma\beta\alpha}, \theta\right) \\ & \times \boldsymbol{M}\left(\boldsymbol{e}_y^{\beta\alpha}, \gamma\right)\boldsymbol{M}\left(\boldsymbol{e}_z^\alpha, \beta\right)\boldsymbol{M}\left(\boldsymbol{e}_x, \alpha\right)\boldsymbol{e}_y l_{fi} + \boldsymbol{R}_w. \end{aligned} \tag{5}$$

These formulas give the motions of our arm model with six degrees of freedom, with the attitude of the upper arm specified by α and β, the lower arm by γ and θ, the wrist by ϕ, and the finger by ψ. There exist many other arm models, but let us consider this one in our work.

Now, when we consider a humanoid robot arm, then the six (Euler) angles cannot take arbitrary values. Thus, we restrict the dynamic range of each angle parameter to be within the range shown in what follows, where the parameters are written with the suffix "real":

$$\begin{aligned} -\pi/2 \le {} & \alpha_{\text{real}}(t) \le \pi/2, \\ -\pi/2 \le {} & \beta_{\text{real}}(t) \le \pi/2, \\ 0 \le {} & \gamma_{\text{real}}(t) \le \pi/2, \\ 0 \le {} & \theta_{\text{real}}(t) \le \pi \\ -\pi/2 \le {} & \phi_{\text{real}}(t) \le 0, \\ -\pi/2 \le {} & \psi_{\text{real}}(t) \le \pi/6. \end{aligned} \tag{6}$$

Let us note that our arm model does not correspond to complete motions of natural arms, for instance ignoring the five degrees of freedom of fingers, and so on. Moreover, considering the need to restrict the dynamic range of variables within finite ranges during motion, we introduce new parameters, represented by the suffix "para," $\alpha_{\text{para}}(t)$, $\beta_{\text{para}}(t)$, ... Utilizing these auxiliary variables, we introduce the following

scheme to determine the six Euler angles at time t:

$$\begin{aligned}
\alpha_{\text{real}}(t) &= \frac{\pi}{2}\sin\left(\alpha_{\text{para}}(t)\right)\\
\beta_{\text{real}}(t) &= \frac{\pi}{2}\sin\left(\beta_{\text{para}}(t)\right),\\
\gamma_{\text{real}}(t) &= \frac{\pi}{4}\sin\left(\gamma_{\text{para}}(t)\right)+\frac{\pi}{4},\\
\theta_{\text{real}}(t) &= \frac{\pi}{2}\sin\left(\theta_{\text{para}}(t)\right)+\frac{\pi}{2},\\
\phi_{\text{real}}(t) &= \frac{\pi}{4}\sin\left(\phi_{\text{para}}(t)\right)-\frac{\pi}{4},\\
\psi_{\text{real}}(t) &= \frac{\pi}{3}\sin\left(\psi_{\text{para}}(t)\right)-\frac{\pi}{6},
\end{aligned} \tag{7}$$

where the auxiliary variables $\alpha_{\text{para}}(t)$, $\beta_{\text{para}}(t)$, ... are determined by neuronal activity patterns at each time step t, as shown in the next subsection. Hence, whatever the six auxiliary variables take, the dynamic ranges of the six Euler angles are kept within the natural range of humanoid arm motion. In the next subsection, we propose a method to determine the six auxiliary variables utilizing neuron activity patterns, which gives *motion coding via neuron activity patterns*

2.2 Driving of arm using activity pattern of recurrent neuron network

In this study, let us employ a RNN to drive the arm model. Specifically, we employ an asymmetrical RNN consisting of N binary neurons. Historically speaking, RNN models had greatly advanced since Hopfield applied a recurrent model to content-accessible memories (Hopfield 1982, 1984). The model used in the present work is an extended type of that model, which has been reviewed in many articles (Anderson and Rosenfeld 1988, 1990). It should be noted that the formulation described in what follows is similar to our previous paper (Li et al. 2008); however, considering the aims of this journal and to avoid misunderstanding or confusion, let us use the terms *network node* and *activity (pattern)*. The network model is shown in Fig. 3. Its updating obeys the rule defined by the following formula:

$$\begin{aligned}
S_i(t+1) &= \text{sgn}\left(\sum_{j\in G_i(r)} W_{ij}S_j(t)\right),\\
\text{sgn}(u) &= \begin{cases} +1 & u \geq 0,\\ -1 & u < 0,\end{cases}
\end{aligned} \tag{8}$$

where $S_i(t) = \pm 1$ $(i = 1 \sim N)$ represents the activity state of the ith node at time t. W_{ij} is an asymmetrical synaptic weight from node S_j to node S_i, where W_{ii} is taken to be 0. $G_i(r)$ means a spatial configuration set with connectivity r $(0 < r < N)$ that is a fan-in number for node S_i. At a certain time t, the *activity state of a node* in the network

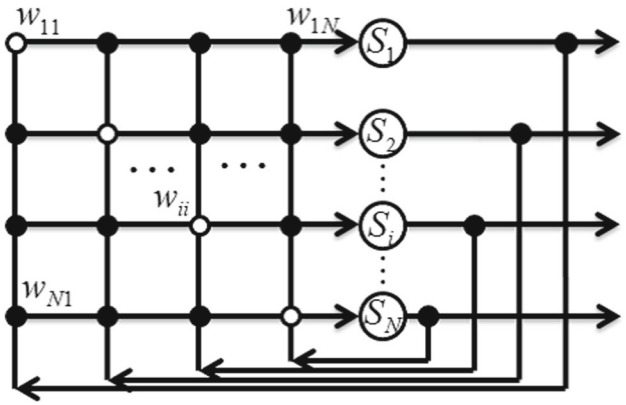


Fig. 3 Fully interconnected RNN model

can be represented as an N-dimensional state vector $\boldsymbol{S}(t)$, which is also called a memory pattern in the context of an associative memory model. The updating rule shows that the time development of the state pattern $\boldsymbol{S}(t)$ is determined by two factors, the synaptic weight matrix $\{W_{ij}\}$ and the spatial configuration set G parameterized by connectivity r. The synaptic weight matrix is chosen so that, at full connectivity $r = N$, there are multiple periodic cycle attractors, and the state vector $\boldsymbol{S}(t)$ converges to one of them depending on the initial value. In our study, an orthogonalized learning method (pseudo-inverse method) is utilized to determine W_{ij}, so that W_{ij} is defined by

$$W_{ij} = \sum_{\mu=1}^{L}\sum_{\lambda=1}^{K}\left(\xi_\mu^{\lambda+1}\right)_i \cdot \left(\xi_\mu^{\lambda}\right)_j^{\dagger}, \tag{9}$$

where $\{\,\xi_\mu^\lambda \mid \lambda = 1\ldots K, \mu = 1\ldots L\,\}$ is a memory (attractor) pattern set, K is the number of memory patterns included in a cycle, and L is the number of memory cycles. $\xi_\mu^{\lambda\dagger}$ is the conjugate vector of ξ_μ^λ that satisfies $\xi_\mu^{\lambda\dagger}\cdot\xi_{\mu'}^{\lambda'} = \delta_{\mu\mu'}\cdot\delta_{\lambda\lambda'}$, where δ is Kronecker's delta. This method was confirmed to be effective at avoiding spurious attractors (Nara and Davis 1992, 1997; Nara et al. 1993, 1995; Nara 2003; Suemitsu and Nara 2005). For example, in Fig. 4, we show the case, $K = 6$, $L = 4$, and $N = 400$, where the activity states of $N = 20 \times 20 = 400$ neurons are represented by bit patterns, which means the activity or nonactivity state and the six patterns are taken as certain face patterns. As the network evolves with the updating rules shown in Eq. (9) for enough time steps, an initial state pattern will converge into one of the embedded limit cycle attractors.

Therefore, in the case of full connectivity, $r = N$, the network can function as a conventional associative memory. If the state pattern $\boldsymbol{S}(t)$ is one of the memory patterns, ξ_μ^λ, then the next output $\boldsymbol{S}(t+1)$ will be the next memory pattern of the cycle, $\xi_\mu^{\lambda+1}$. If the state pattern $\boldsymbol{S}(t)$ is near one of the

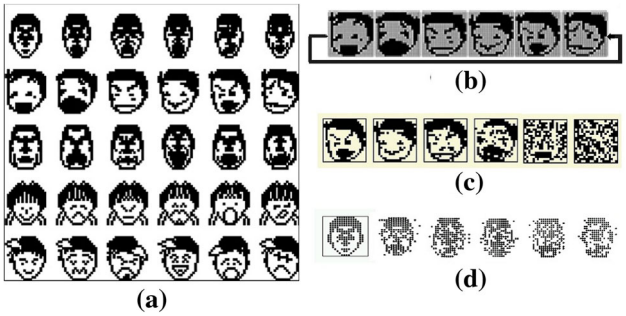


Fig. 4 An example of embedded attractor patterns employed in our previous papers and their destabilization by reducing connectivity r in the synaptic connection matrix. **a** Limit cycle attractors consisting of six patterns per cycle, totaling five cycles, where each state vector is represented by a $N = 400 = 20 \times 20$ bit pattern. **b** Updating with $r = 400$ (full connectivity) reproduces a cycle attractor. **c** Updating with $r = 8$ produces the case where any initial pattern falls into chaos. **d** Updating with $r = 50$ gives weak chaos, where the orbit in 400-dimensional state space does not so much leave the original cycle attractor as retains a nonperiodic orbit, so we could crudely control chaos by changing the connectivity. It should be noted that the preceding examples are only to "guide the eyes" to show the destabilization of attractors by reducing connectivity r. In this work, the embedded attractors are shown in Fig. 6

memory patterns, $\boldsymbol{\xi}^{\lambda}_{\mu}$, the output sequence $\boldsymbol{S}(t + kK)$ $(k = 1, 2, 3 \ldots)$ will converge to the memory pattern $\boldsymbol{\xi}^{\lambda}_{\mu}$. In other words, for each memory pattern, there is a set of state patterns, called a memory basin B^{λ}_{μ}. If $\boldsymbol{S}(t)$ is in the memory basin B^{λ}_{μ}, then the output sequence $\boldsymbol{S}(t + kK)$ $(k = 1, 2, 3 \ldots)$ will converge to the memory pattern $\boldsymbol{\xi}^{\lambda}_{\mu}$.

On the other hand, if connectivity r is reduced sufficiently by pruning the synaptic link matrix, the attractors become unstable. Various aspects of the dynamics at reduced connectivity are discussed in our previous studies (Nara and Davis 1992; Nara et al. 1993, 1995; Nara and Davis 1997; Nara 2003) on the network state $\boldsymbol{S}(t)$. In particular, it was found that, typically, in a regime of small r, there occurs chaotic wandering that repeatedly visits all the areas of state space that were attractor basins at full connectivity.

Therefore, the dynamics in $r \sim N$ is called a *limit cycle attractor regime*, and $r << N$, a *strange (or chaotic) attractor regime* hereafter. We apply these two different dynamics to executing complex functions of an robot arm, as shown in subsequent sections.

It should be noted that, generally speaking, chaotic dynamics also could be realized by reduced connection weights in all of the connection configurations. However, as stated in our previous paper (Nara and Davis 1992), the other reducing methods of connection weight to null, except that reducing the fan-in number produced no chaos in these neural systems. In our various trials of reducing connections except the fan-in number, the updating of neural states finally converges toward spurious attractors. It is plausible that random pruning of the connection weight would causes random fluctuations in the collective field applied to each neuron, but the key point is whether the fluctuation keeps a wide dispersion or not. Along unstable (chaotic) orbits realized in this paper and our previous paper as well, the collective fields do not obey a Gaussian process, which was confirmed by one of the authors (S. Nara) in his past numerical experiments.

2.3 Designing attractors for controlling our robot arm

Biological data show that the number of nodes in brains varies from species to species, and the human brain has more than 100 billion (10^{11}) nodes, but a human only has, roughly speaking, several hundred muscles to perform common motions. The motions are controlled by neural systems, in particular, the motions of muscles are governed by the activity of an enormous number of motor neurons (Nicolelis 2001). Therefore, comparing the number of muscles and the number of neurons to control them in the brain, the coding of control signals from neurons to muscles could be quite redundant. Thus, we try to utilize dynamical behaviors generated by the neural network in the present model, consisting of a relatively large number of nodes to control a small number of motion variables implemented in a robot arm.

We confirmed that the required properties of chaotic dynamics in our network do not so sensitively depend on the size of the node number (Nara 2003) in the range $N = 200 \sim 900$. However, if N is too small, chaotic dynamics cannot occur; but if N is too large, it results in excessive computing time. Therefore, the number of nodes in the network is taken as $N = 400$ in this paper. At a certain time t, the state pattern $\boldsymbol{S}(t)$ in the network can be represented by a 400-dimensional state vector, while the robot arm moves in three-dimensional space in which an attitude of the arm is specified by a 6-dimensional vector in the parameter space. Therefore, we must transform a 400-dimensional state vector into a 6-dimensional vector by a certain coding.

Before proposing our coding method, let us reconsider motions of our robot arm for preparation. At certain time t_0, the robot is assumed to be at the initial position shown in Fig. 2. At any time t after t_0, the arm assumes a certain attitude, which is shown in Fig. 1. Since the arm has six mechanical degrees of freedom, the motion of the arm at each time step includes six actions, so three-dimensional motions of the arm are generated by six successive operations applied to the four vectors, as written in Eqs. 2, 3, 4, and 5, where all angles are given by Eq. 7. Therefore, to perform three-dimensional motion of the arm using the dynamical behavior of the neural network, the 400-dimensional state pattern $\boldsymbol{S}(t)$ must be transformed into the 6 angle variables,

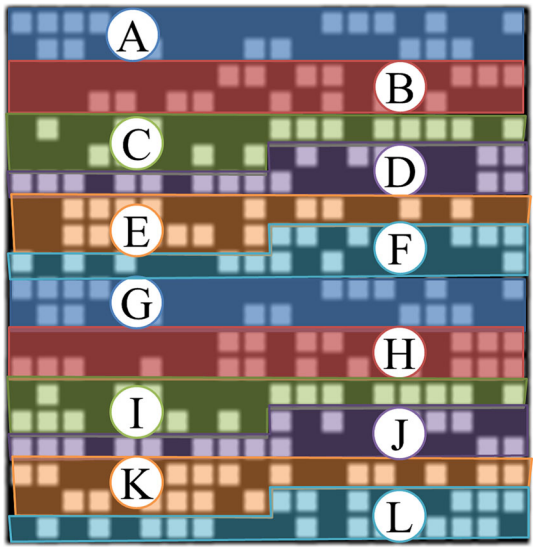


Fig. 5 Twelve fragment vectors of an activity pattern are coded into increments of 6 parametric angle variables by taking inner products between pairs of fragment vectors. These are $(1/N_{A,G})\boldsymbol{S}_A \cdot \boldsymbol{S}_G$, $(1/N_{B,H})\boldsymbol{S}_B \cdot \boldsymbol{S}_H$, $(1/N_{C,I})\boldsymbol{S}_C \cdot \boldsymbol{S}_I$, $(1/N_{D,J})\boldsymbol{S}_D \cdot \boldsymbol{S}_J$, $(1/N_{E,K})\boldsymbol{S}_E \cdot \boldsymbol{S}_K$, $(1/N_{F,L})\boldsymbol{S}_F \cdot \boldsymbol{S}_L$, which give an example of coding using the activity pattern of many nodes

$\alpha_{\text{para}}(t)$, $\beta_{\text{para}}(t)$, $\gamma_{\text{para}}(t)$, $\theta_{\text{para}}(t)$, $\phi_{\text{para}}(t)$, and $\psi_{\text{para}}(t)$. In other words, the preceding six variables must be coded to $\boldsymbol{S}(t)$. Thus, our proposal is as follows. First, the time course of the six parameter angles is defined as

$$\chi'_q(t+1) = \chi'_q(t) + \Delta\chi'_q(t), \tag{10}$$

where χ'_q represents successively α'_{para}, β'_{para}, γ'_{para}, θ'_{para}, ϕ'_{para}, ψ'_{para}. Second, increments of six variables at time t, $\Delta\chi'_q(t)$, are coded into neural activity pattern in 400-dimensional space, as shown in Fig. 5. Thus, for example,

$$\Delta\alpha'_{\text{para}}(t) = \frac{1}{40}\sum_{i=1}^{40} S_i(t)\, S_{i+200}(t), \tag{11}$$

and so on.

As defined in Eq. 11 and Fig. 5, the six parametric angles $(\alpha'_{\text{para}}, \beta'_{\text{para}}, \gamma'_{\text{para}}, \theta'_{\text{para}}, \phi'_{\text{para}}, \psi'_{\text{para}})$ are determined using A-G(α'_{para}) and B–H(β'_{para}) neuron subgroups (80 neurons) and C–I(γ'_{para}), D–J(θ'_{para}), E–K(ϕ'_{para}), and F–L(ψ'_{para}) neuron subgroups (60 neurons), respectively, so the coding includes a high level of redundancy in the sense that one degree of freedom is determined by more than several tens of neurons. For all parameter angles, $-1 \le \Delta\chi'_q(t) \le +1$. These six increments are transformed into increments of the originally defined parameter angles in units of radians (Eq. 7) using the following formula:

$$\Delta\chi_q(t) = Q_0\frac{\pi}{180}\Delta\chi'_q(t), \tag{12}$$

where Q_0 is a constant determined empirically to produce a smooth arm motion; it is taken to be 2.5 in the present work. This results in

$$-Q_0\frac{\pi}{180} \le \Delta\chi_q(t) \le Q_0\frac{\pi}{180}.$$

Using $\Delta\chi_q(t)$, the six parameter angles are updated at each time step as

$$\chi_q(t+1) = \chi_q(t) + \Delta\chi_q(t). \tag{13}$$

These updated angles are inserted into Eq. 7, and the new arm attitude at time $t+1$ is obtained by executing the six actions represented in Eq. 2, 3, 4, and 5, which result in new positions of the elbow, wrist, and finger.

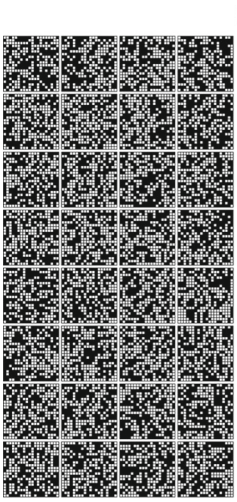
Using this coding, when we design an appropriate limit-cycle attractor consisting of cyclic patterns, decoding them into arm attitudes gives cyclic arm motions in three-dimensional space. Therefore, the next problem consists in the design of activity patterns corresponding to desired arm motions. In the present work, we employ eight limit-cycle attractors consisting of four patterns per cycle, as shown in Fig. 6. Each pattern consists of 12 blocks (Fig. 5), and the pattern of each block is designed so as to satisfy the given values of $\Delta\chi'_q = (1/N_{\kappa,\rho})\boldsymbol{S}_\kappa \cdot \boldsymbol{S}_\rho$ using a random number generator, where the variable correspondences are

$$(q|\kappa,\rho) = (\alpha|A,B),(\beta|G,H),(\gamma|C,I),(\theta|D,J),(\phi|E,K),(\psi|F,L),$$

so $\Delta\chi'_q$ $(\chi_q = \alpha,\beta,\gamma,\theta,\phi,\psi)$ becomes one of $1, 0$, or -1. The reason we employ eight limit-cycle attractors to embed four stationary arm motions will be given later (Fig. 7).

3 Functional experiments in ill-posed settings solved by chaos of neural network

Now let us show that chaos occurring in neural network dynamics could be useful not only in complex information processing but also in complex control under ill-posed situations. Actually, in previous papers we reported several such situations, for instance, memory search (Nara and Davis 1992; Nara et al. 1993, 1995), image synthesis (Nara and Davis 1997), solving a two-dimensional maze (labyrinth) (Suemitsu and Nara 2004; Li et al. 2008), and simultaneous multichannel signal transfers via a nonlinear medium (Soma et al. 2015).



$$\begin{matrix} (\Delta\alpha_{para} & \Delta\beta_{para} & \Delta\gamma_{para} & \Delta\theta_{para} & \Delta\phi_{para} & \Delta\psi_{para}) \\ (+1 & 0 & 0 & +1 & 0 & 0) \\ (0 & +1 & 0 & +1 & 0 & 0) \\ (0 & 0 & +1 & 0 & 0 & +1) \\ (0 & 0 & 0 & 0 & -1 & +1) \\ (-1 & 0 & 0 & -1 & 0 & 0) \\ (0 & -1 & 0 & -1 & 0 & 0) \\ (0 & 0 & -1 & 0 & 0 & -1) \\ (0 & 0 & 0 & 0 & +1 & -1) \end{matrix}$$

Fig. 6 Eight limit-cycle attractors consisting of four patterns per cycle, where they are designed so as to give constant angle increments to corresponding Euler angle parameters, shown in the right brackets. The coding introduced in Eqs. 10, 11, 12, 13, and 7 gives the four kinds of stationary arm motions shown in the next figure

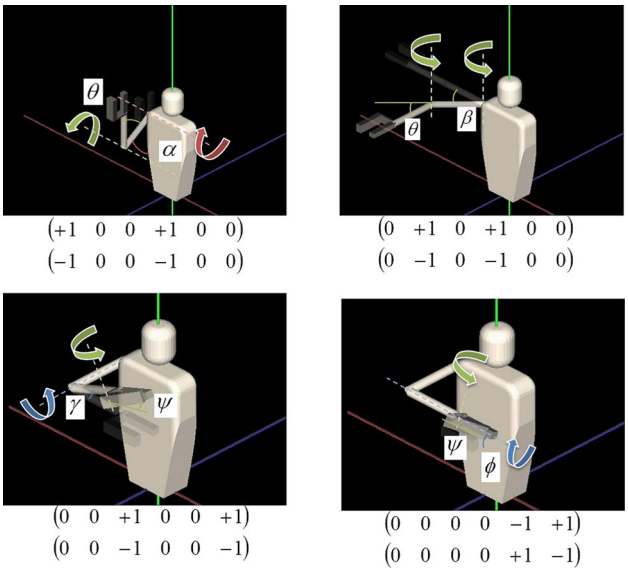


Fig. 7 Four stationary arm motions by embedded limit-cycle attractors shown in previous figures, where each two attractors indicated in the figure give the same stationary arm motion, but angle increments have opposite signs, which means that the motion starts in the opposite direction

3.1 Complex motions of robot arm driven by chaos occurring in neural network dynamics

As shown in the previous section (Sect. 2.2), in the limit-cycle attractor regime ($r \sim N$), embedded attractors have wide basins in the N-dimensional state space of the network, which means that, even when starting from an arbitrary given random pattern, updated state vector following Eq. (8) converges to one of the embedded limit-cycle attractors. Thus,

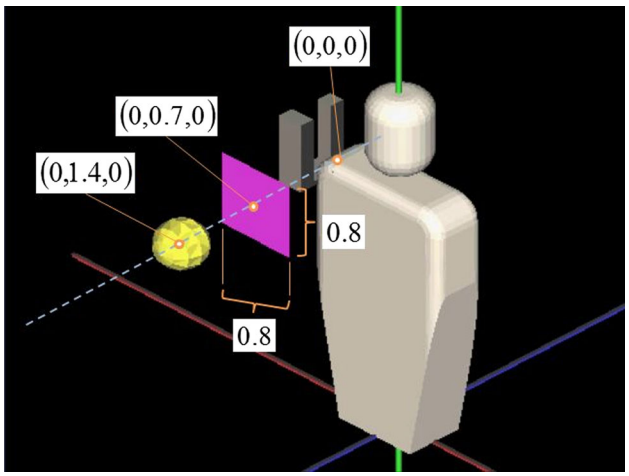


Fig. 8 Initial attitude of robot arm in present task setting and position of obstacle with a set size. Also, the position of the target is shown, where the shape is a sphere and the radius is 0.2

the arm motion quickly becomes one of the embedded stationary motions. On the other hand, in the chaos regime ($r << N$), the activity pattern of the neural network wanders in state space and repeatedly visits all the areas of state space that were attractor basins at full connectivity. This means that the coded motion introduced in Sect. 2.3 produces complex movement that appears chaotic, which is not random motion but includes many fragments of stationary motions successively and in combination at each time step of the motions.

3.2 Solving ill-posed problems with robot arm using adaptive switching between chaotic and definite motions

The task is set as follows:

(1) Robot arm must grip a target object and return it to the set position.
(2) Obstacles prevent arm from arbitrarily making a motion in three-dimensional space.
(3) Robot arm has no preknowledge about the position or size of the obstacles.
(4) Robot arm has no advanced visual processing function but can catch only rough directions including uncertainty about the target object.
(5) Robot arm can recognize the existence of obstacles when any part of the arm comes in contact with those obstacles.

These settings indicate that a given task is ill-posed, in the sense that, first, there is no guarantee of the existence of a solution and, second, there is no guarantee of the uniqueness of a solution, if one is even found. A more practical configuration is shown in Fig. 8.

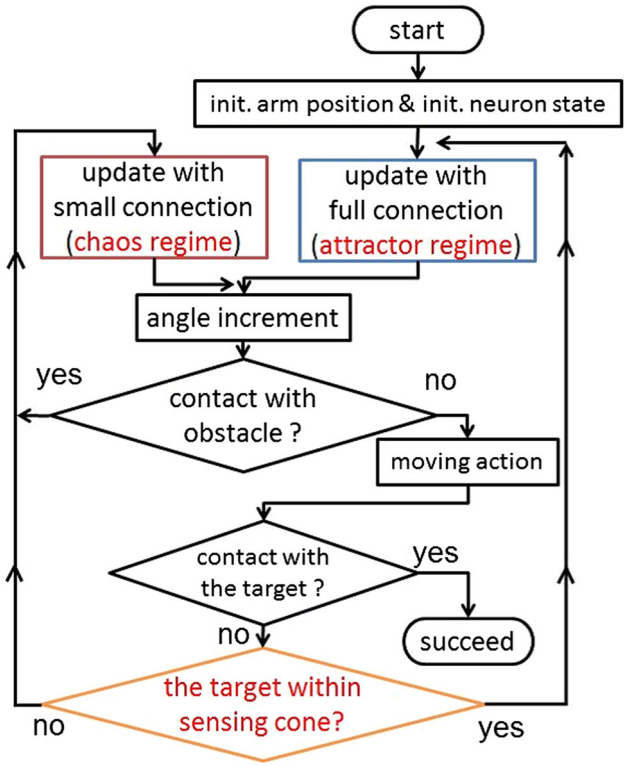


Fig. 9 Algorithm used in solving actions related to the target using adaptive switching between attractor regime ($r \sim N$) and chaos regime ($r << N$)

3.2.1 Moving actions until the target is reached

The key idea for solving this ill-posed problem is that, by means of adaptive switching between a limit-cycle attractor regime and a strongly chaotic attractor regime, complex problems can be solved with simple rule(s). More practically:

(1) At each time step, the robot arm checks whether the target exists within a certain range of direction. If the check produces *no existence*, then the connectivity is switched to $r << N$, which results in chaotic motions to search for the target.
(2) If the robot finds the target direction with given uncertainty (*existence within a given direction range*), the neural connectivity is switched from $r << N$ to $r \sim N$, which results in definite motions that correspond to one of the embedded attractors.
(3) At every time step, if any part of the arm contacts any obstacle, then the connectivity is switched to $r << N$, which results in chaotic motions to avoid the obstacle and to keeping looking for the other motions near the target.

A comprehensible description of the foregoing algorithm is shown in Fig. 9.

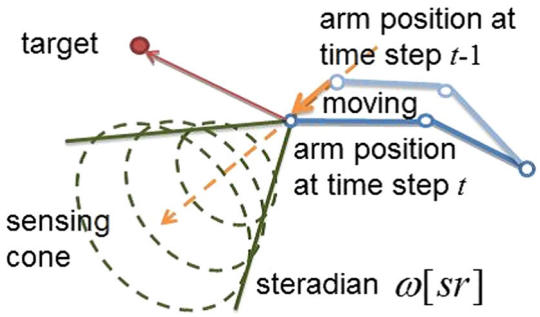


Fig. 10 A set tolerance in recognizing the rough direction of a target represented by a sensing cone having steradian ω

Now, let us discuss the description "*rough direction including uncertainty about the target object*" or "*existence within a given direction range*" in a little more depth. At every time step from $t-1$ to t, we can specify the axis connecting the two center positions of the finger (thumb and the other) and a sensing cone, the top of which is on the center of the finger at time t and the axis is forward of the aforementioned direction (Fig. 10). Note that the center of the finger is defined as the position connecting the center of the mother finger and that of the other finger. This means that the arm system has sensors that can detect whether the target exists within a certain steradian ω, as shown in Fig. 10.

To show that the preceding algorithm is effective, we repeat the computer experiments with the following conditions:

(1) The number of trials is 1000.
(2) If the robot arm cannot reach for the target in less than 5000 time steps, then the trial is regarded as unsuccessful.
(3) As the initial condition of each trial, a $20 \times 20 = 400$ bit pattern and a configuration of small connectivity r, the total number of which is ${}_{400}C_r$ for a given r, are prepared using a random number generator.
(4) The success rate is evaluated for connectivity

$$r = 10, 20, 30, \ldots, 90, 100.$$

The result is shown in Fig. 11 and a successful result in Fig. 12.

3.2.2 Gripping actions associated with reaching for a target

Associated with successfully reaching for a target, the next important action is gripping the target with two fingers consisting of the mother finger and the only other finger in the present model. At every step in carrying out the arm motion, we check the following conditions for the positional relation between the target and the two fingers:

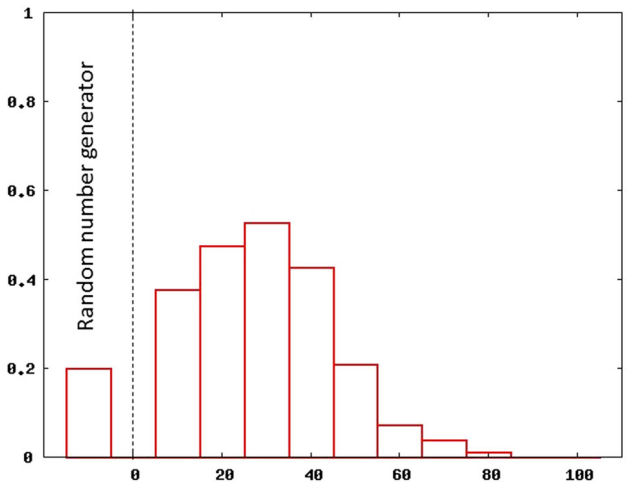


Fig. 11 Success rate in 1000 trials vs. r, in task of only reaching for the target (not yet gripping), where the success rate of the same actions in the case where only the activity update at each time step is replaced by random patterns made by a random number generator is shown

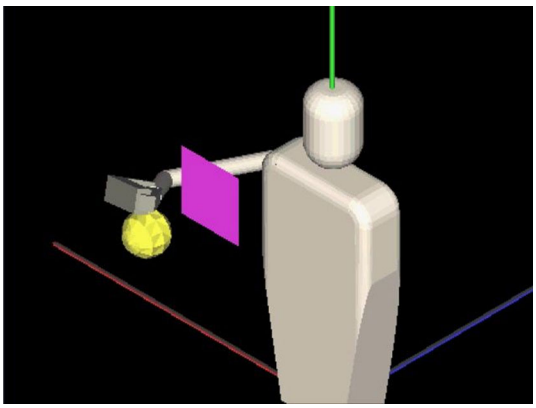


Fig. 12 Example of successful trials, in task of only reaching for the target (not yet gripping)

(1) A circle exists that is the cross section of the sphere across the four-sided polygon plane consisting of the mother finger and the other finger, and the diameter is within a set tolerance. In the present work, a set value is $\pm 30\% \times$ [the diameter of the target sphere].
(2) The center of the circle exists within a set angle opened at the center of the mother finger. In the present work, a set value is $\pm \pi/12$.

If the conditions are satisfied, then the experiment finishes with the arm successfully gripping the target.

A comprehensible drawing of these conditions is shown in Fig. 13. It should be noted that the mentioned tolerances are typical values and the results are not very sensitive for these parameter values, although the results of the computer experiments are not shown. The result of the evaluation of the success rate (reaching + gripping) is shown in Fig. 14,

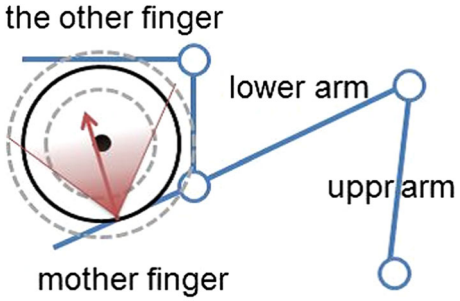


Fig. 13 Conditions for successful gripping by robot arm

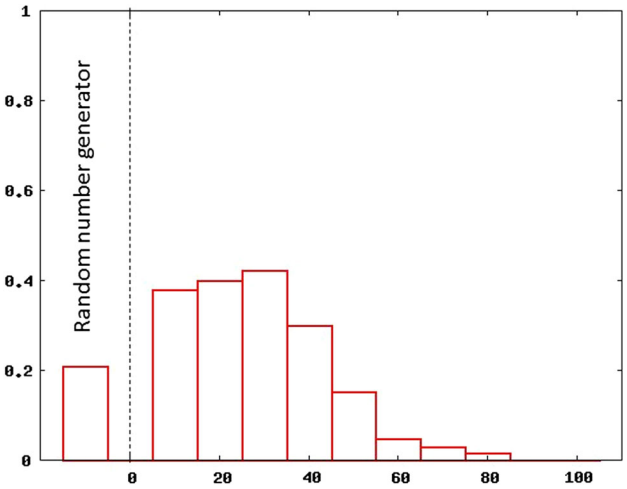


Fig. 14 Success rate in 1000 trials vs. r, in task of reaching for the target and gripping, where the success rate of the same actions in the case where only updating of activity at each time step is replaced by random patterns made by a random number generator is shown

calculated under the same conditions stated in Fig. 11, and one of the successful results is shown in Fig. 15.

3.2.3 Taking back actions after gripping the target

The final moving action is to return the arm to the initial position while avoiding obstacles. The condition for retracting the arm is only to change the target object to the initial position of the robot arm's hand. Thus, the basic algorithm is the same as those shown in Fig. 9, where the condition *the target* is replaced by *the starting position of the hand*. Thus, the results could indicate that the final attitudes of the arm are not the initial ones, but the position of the hand ($\sim$ the position of the center of the gripped target sphere) is the same as the starting position.

3.2.4 Success rate of task: reaching, gripping, retracting

The success rate is shown in Fig. 16, where the computer experiments are done under the same conditions as those

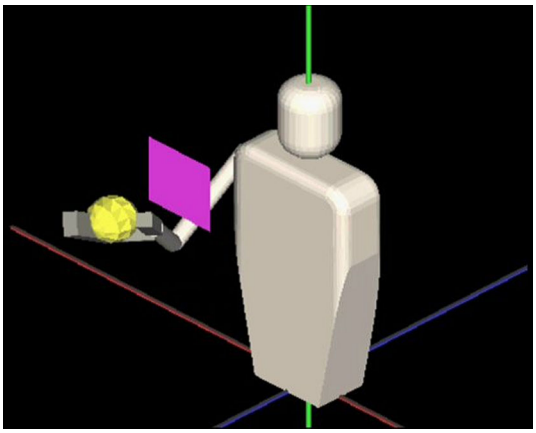


Fig. 15 Example of successful trials, in the task of reaching for the target and gripping

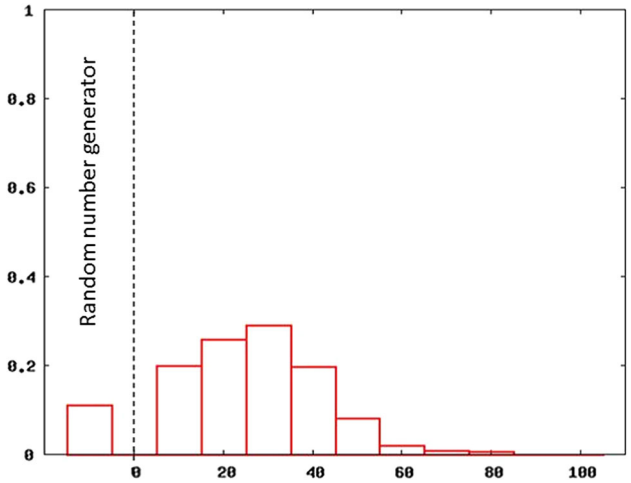


Fig. 16 Success rate in 1000 trials vs. r, in the task of reaching for the target, gripping, and retracting, where the success rate of the same actions in the case where only the activity of updating at each time step is replaced by random patterns made by a random number generator is shown

mentioned in Sect. 3.2.1 regarding the evaluation of statistical data on the success rate.

A detailed discussion of these results is given in Sect. 4.

3.3 A trial of hardware implementation for a single-arm system

It is quite important to attempt a hardware implementation to confirm the practical feasibility of the theoretical findings and computer experiment based on them. In this work, as an initial stage of this, we introduce one of our trials of hardware implementation, where the settings defined in the theoretical treatments are slightly changed according to technological problems associated with hardware implementation. The machine experiments indicate that the robot is able to avoid obstacles and reach for the target in a situation

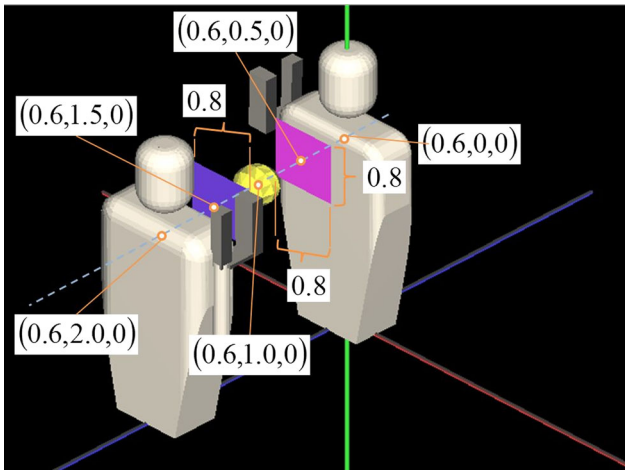


Fig. 17 Initial setting in experiment on competing behaviors to take a target in competition with another robot arm

where the robot can obtain only rough target information, including uncertainty, using a few sensors. The hardware implementation is not yet completed due to the technological difficulty of setting down an isolated object and gripping it with robot fingers. Thus, the actual implemented contents at the present stage are indicated in the appendix A1 and A2.

3.4 Competing behaviors of two independent robot arm in connection with taking only one target object

As shown in previous sections, chaos in a RNN is useful when searching for a target and avoiding obstacles under ill-posed situations. This is a case of a single-robot arms, so it is quite natural that functional experiments are extended to a case of two independent robot arms, in particular, to competing behaviors of two independent robot arms in connection with taking only one target object. This type of functional experiment is inspired by biological behaviors in which two animals are competing for something, particularly food.

An example of the initial setting in our computer experiment is shown in Figs. 17 and 18; in the experiment, one of the two robot arms was able to take the target after some fighting with the other arm.

The results shown above are preliminary results, so the details of the present study will be reported in a future paper.

4 Discussion

Our arm model and the computer experiments are motivated by arm motions in which adaptive functions are observed in many animals, not only in mammals but also in insects, in various natural environments. Often the situations in which arm motions are made result in *ill-posed settings* with respect to

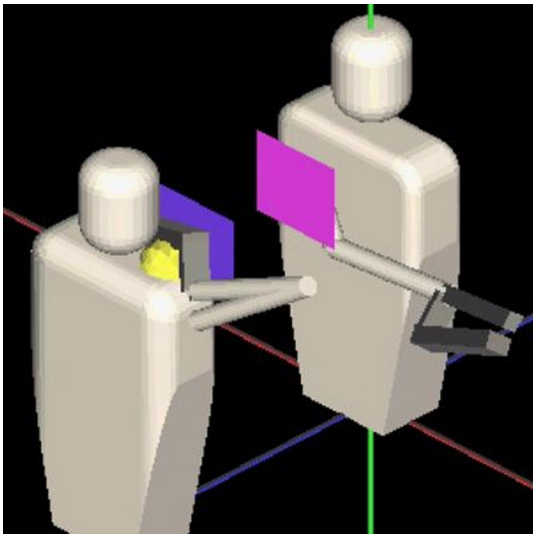


Fig. 18 Example of a typical case where an robot arm successfully takes the target back to one side having won out against the other arm

solving certain problems, for instance, taking an object with the hand and bringing it back to a convenient position under difficult circumstances, for instance, in dark surroundings and in the presence of obstacles.

There are three key ideas related to realizing autonomous and adaptive properties in the present robot arm working in ill-posed situations. First is the property of adaptive switching between a limit-cycle attractor regime and chaotic attractor regime depending on inputs with given uncertainty. Second, in a chaotic regime, various fragments of embedded attractors transiently appear and vanish, which results in the evocation of mixed or combined activity patterns simultaneously consisting of many limit-cycle attractor fragments. These autonomous dynamics may seem disorderly, but they include abundant dynamic structures, as discussed in our previous papers and elsewhere (Arhem et al. 2000). In this work, they strongly depend on selected connectivity, so improvements in terms of more adaptive control, including different connectivities depending on various visual or behavioral situations, for instance, reaching, grasping, and retracting, would bring advanced functions. However, such improvements represent the next issues we plan to tackle and will be studied in future work. Third, the assumed coding of arm motions includes high rates of redundancy, which means that the six degrees of freedom are coded into and decoded from a large number of nodes assigned to each degrees of freedom, in the present work, 60 or 80 nodes per degrees of freedom.

Neural chaos during the computer experiment on the two situations where there is no detected signal within the sensing cone and there is a signal when contact is made with obstacles in the moving directions of an arm produces complex motions that are not random but reflect dynamic and transient combinations of embedded attractors, and it enables the robot arm to determine the target direction and the motions necessary to avoid obstacles, both more effectively than at least random motion, as indicated in Figs. 11, 14, and 16. These three figures also indicate that there is a certain optimum connectivity resulting in the greatest success rate of trials, which means that the usefulness of chaos depends on the dynamic localization structures of chaotic orbits in the high-dimensional state space of network activity. This feature was commonly observed in our previous functional experiments of neural chaos, in memory search (Nara and Davis 1992; Nara et al. 1993, 1995), image synthesis (Nara and Davis 1997), movement control (Suemitsu and Nara 2004; Li et al. 2008), and simultaneous multichannel signal transfers via a nonlinear medium (Soma et al. 2015). Therefore, clarifying what dynamic structures are best suited for indicating enhanced performance awaits future investigations.

5 Summary and concluding remarks

We have shown and demonstrated that adaptive switching between a chaotic regime and a limit-cycle attractor regime depending on external situations indicates that chaos in systems with large but finite degrees of freedom are useful not only for solving ill-posed problems but also for practical realization of our idea. *Complex functions can be carried out by simple rules using chaotic dynamics*, as shown in our previous papers (Nara and Davis 1992; Nara et al. 1995; Kuroiwa et al. 1999; Nara 2003; Li et al. 2008; Yoshinaka et al. 2012). One of the authors (S.N.) has put forth a hypothesis that chaos could play an important role in biological functions; however, the evidence for this has not yet been established by mathematical treatment but only by means of computer experiments, case by case in practical situations. The present work is an example of piling up of experimental results. Let us summarize the paper's results as follows:

(1) A novel model of an robot arm is proposed that is driven by *either* chaos introduced into a recurrent neural network model by pruning synaptic connectivity *or* oscillatory patterns (limit-cycle attractors) embedded in fully connected synaptic connections obtained by the application of a pseudo-inverse learning rule (orthogonalized learning rule) using conventional methods.
(2) A few tasks carried out in ill-posed settings are given to the robot arm. An example is reaching for a target object, where the object's position is unknown to the robot and there are obstacles that the robot has no preknowledge of. The robot arm has sensors that enable it to determine the existing direction of the target, but there is *including considerable uncertainty*.
(3) Computer experiments show that adaptive switching between an attractor regime and chaos regime produces

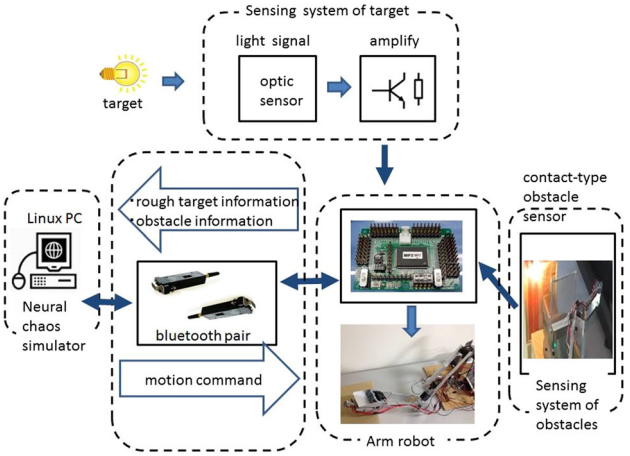


Fig. 19 Block diagram of hardware implementation of our arm system

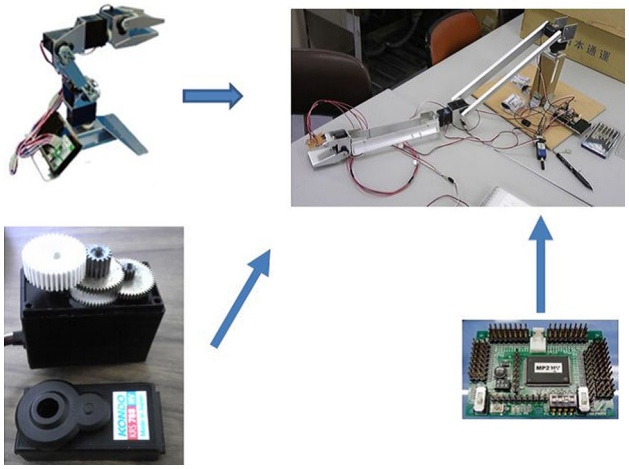


Fig. 20 (1) Original robot arm called RT002. (2) Reformed robot arm with improved servomotors. (3) New, stronger servomotors. (4) Control board, motion processor 2HV(MP2). The items in (3) and (4) are from Kondo Kagaku Co., LTD

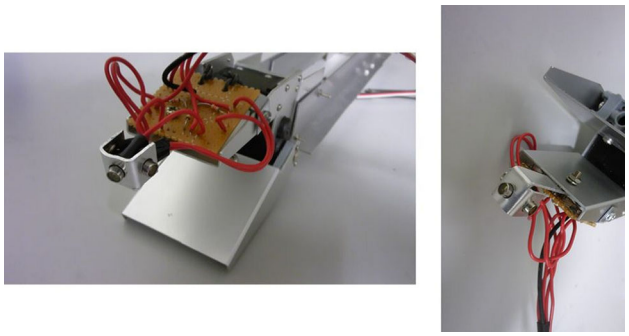


Fig. 21 Three semiconductor optic sensors attached at the front edge of the fingers of our robot arm. Each sensor has a rather wide area for detecting light

successful trials within a plausible execution time, which are shown in computer experiments using a 400 binary network node model, where the limit-cycle attractor regime is in a fully connected state and the chaos regime is in a reduced connected state (a small number of input connections per node = fan-in number reduction).

(4) This system is extended to a case of two independent robot arms, in which the two robots compete with each other to take only one target object. This type of functional experiment was inspired by biological behaviors in which two animals compete for something, especially food. However, only a few preliminary results are shown, and detailed experiments and data analysis are left for future studies.

(5) A trial on a hardware implementation of our proposals are given in the appendix A1 and A2. Though it is also at the preliminary stage because of technological difficulties, it suggests that a practical engineering application of chaos is possible.

Acknowledgements This work was supported in part by Grant-in-Aid #26280093 of the Ministry of Education of Science, Sports & Culture of the Japanese government and by the Cooperative Research Program of the Network Joint Research Center for Materials and Devices.

Appendix A Hardware implementation trial for a single-arm system

Appendix A1 Preparation of our hardware system

The block diagram of our hardware system is shown in Fig. 19. In our study, an robot arm with sensors was constructed using a robot arm from R.T. Corporation, which was adapted by us so as to have five degrees of freedom; the result is shown in Fig. 20.

The length of the arm is too short to carry out our plan with regard to arm behaviors, so the arm was adapted using appropriate aluminum plate bars and, instead of the original version, the stronger servomotors were installed, as shown in the figure. Nonetheless, the rotation freedom of the wrist was not implemented because of a few technical difficulties at this stage, in particular due to too much heavy load on the servomotor of the robot arm shoulder.

In our theoretical simulations, a recognizing cone region with a certain steradian range is introduced at the front edge of the arm, whose axis faces the forward direction of arm edge motion at the most recent time step (Fig. 10). Thus, a sensor is necessary whether or not the target exists within the cone region. Simplifying this condition, we set three semiconductor optic sensors (ST-1KB, sensing peak wavelength: 800 nm, KODENSHI SY CORP.) at the edge of the fingers, each of which has a rather wide area for detecting light (Fig. 21). An incandescent light bulb (KR100V90WCA, Toshiba) was used as the target lamp.

Fig. 22 An obstacle before setting used in hardware experiments

As the main theme of this work, we consider the arm system as having six degrees of freedom. However, in the hardware implementation in this appendix, let us confine ourselves to five degrees of freedom, in which the freedom of a rotatable and bendable wrist is replaced by a fixed one, for the following reason. When we introduce a movable wrist, we must attach a servomotor for the wrist, then the total weight of the hand ahead of the shoulder increases rather dramatically, and no strong enough servomotor could be found to control the shoulder.

As noted in the previous section, we must place an obstacle between the target and the robot arm. We employ a system in which contact signals between arm and obstacle are generated on the side of the obstacle. As our obstacle, we choose a rectangular plate; called a touch panel, it can detect when contact is made with an object by a change in the electric sheet resistance when an unknown object makes contact by applying a little pressure on the surface. Furthermore, a narrow aluminum sheet is attached on all side edges of this plate, and the appropriate electrical voltage is applied between the sheet and the arm to detect the side edge contact of the arm. The plate before setting is shown in Fig. 22

Appendix A2 The results of hardware action trials

In the actual action trials, based on the many problems generated by technical restrictions, we changed slightly the algorithms or conditions theoretically given in the main text, where we discard their description one by one. However, the principle and aims proposed in this paper are strictly retained. First, we show the successive snapshots in the process of reaching for the target in the case *without obstacles* in Fig. 23. Second, we show the case *with an obstacle* in Fig. 24.

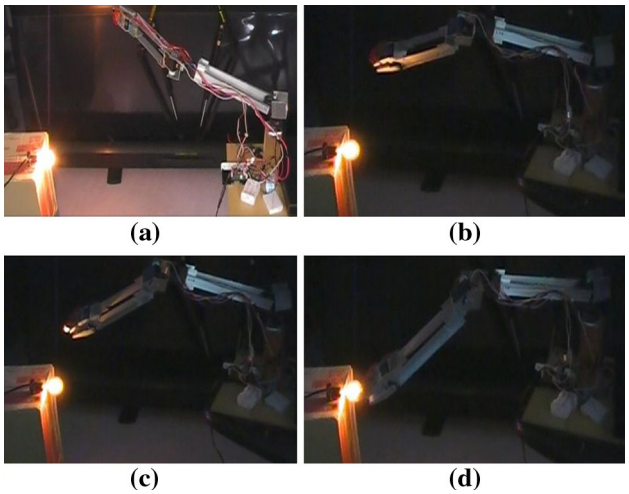


Fig. 23 Example of successful reaching action trial in the case *without obstacles*, where snapshots are taken by indefinite time sampling. In **b**, **c**, and **d**, the background is kept dark to make the optic sensor on the finger detect the target light easily

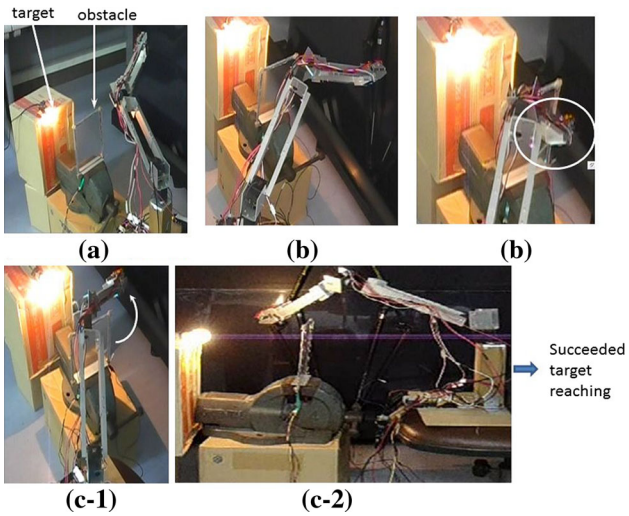


Fig. 24 Example of successful reaching action trial in the case *with an obstacle*, where snapshots are taken by indefinite time sampling

References

Aihara K, Takabe T, Toyoda M (1990) Chaotic neural networks. Phys Lett A 114:333–340

Anderson JA, Rosenfeld E (eds) (1988) NEUROCOMPUTING. The MIT Press, Cambridge

Anderson JA, Rosenfeld E (eds) (1990) NEUROCOMPUTING 2. The MIT Press, Cambridge

Arhem P, Blomberg C, Liljenström H (2000) Disorder versus order in brain functioning—essays in theoretical neurophysics. World Scientific Publ. Co, London

Babloyantz A, Destexhe A (1986) Low-dimensional chaos in an instance of epilepsy. Proc Natl Acad Sci USA 83:3513–3517
Fujii H, Itoh H, Ichinose N, Tsukada M (1996) Dynamical cell assembly hypothesis—theoretical possibility of spatiotemporal coding in the cortex. Neural Netw 9:1303–1350
Hayashi H, Ishizuka S, Ohta M, Hirakawa K (1982) Chaotic behavior in the Onchidium giant neuron under sinusoidal stimulation. Phys Lett A 88:435–438
Hopfield JJ (1982) Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci USA 79:2554–2558
Hopfield JJ (1984) Neurons with graded response have collective computational properties like those of two-state neurons. Proc Natl Acad Sci USA 81:3088–3092
Huber F, Thorson H (1985) Cricket auditory communication. Sci Am 253:60–68
Kaneko K, Tsuda I (2003) Chaotic itinerancy. Chaos 13(3):926–936
Kuroiwa J, Nara S, Aihara K (1999) Functional possibility of chaotic behaviour in a single chaotic neuron model for dynamical signal processing elements. In: 1999 IEEE international conference on systems, man, and cybernetics (SMC'99), Tokyo, October, 1999, vol 1, p 290
Li Y, Kurata S, Morita S, Shimizu S, Munetaka D, Nara S (2008) Application of chaotic dynamics in a recurrent neural network to control: hardware implementation into a novel autonomous roving robot. Biol Cybern 99:185–196
Liljenström H (1995) Autonomous learning with complex dynamics. Int J Intell Syst 10:119–153
Nara S (2003) Can potentially useful dynamics to solve complex problems emerge from constrained chaos and/or chaotic itinerancy? Chaos 13(3):1110–1121
Nara S, Davis P (1992) Chaotic wandering and search in a cycle memory neural network. Prog Theor Phys 88:845–855
Nara S, Davis P (1997) Learning feature constraints in a chaotic neural memory. Phys Rev E 55:826–830
Nara S, Davis P, Kawachi M, Totuji H (1993) Memory search using complex dynamics in a recurrent neural network model. Neural Netwo 6:963–973
Nara S, Davis P, Kawachi M, Totuji H (1995) Chaotic memory dynamics in a recurrent neural network with cycle memories embedded by pseudo-inverse method. Int J Bifurc Chaos Appl Sci Eng 5:1205–1212
Nicolelis MAL (2001) Actions from thoughts. Nature 409:403–407
Physiome (2012) http://www.physiome.jp/index.html. It should be noted that, generally speaking, Platform sites in web-system is often not permanent but rather improved and/or changed occasionally. So, readers should be careful when they access via internet
Skarda CA, Freeman WJ (1987) How brains make chaos in order to make sense of the world. Behav Brain Sci 10:161–195
Soma K, Mori R, Sato R, Furumai N, Nara S (2015) Simultaneous multichannel signal transfers via chaos in a recurrent neural network. Neural Comput 27:1083–1101
Suemitsu Y, Nara S (2004) A solution for two-dimensional mazes with use of chaotic dynamics in a recurrent neural network model. Neural Comput 16(9):1943–1957
Suemitsu Y, Nara S (2005) Emergence of unstable itinerant orbits in a recurrent neural network model. Phys Lett A 344(2):220–228
Tsuda I (1991) Chaotic itinerancy as a dynamical basis of Hermeneutics in brain and mind. World Futures 32:167–184
Tsuda I (2001) Toward an interpretation of dynamic neural activity in terms of chaotic dynamical systems. Behav Brain Sci 24(5):793–847
Yao Y, Freeman WJ (1990) Model of biological pattern recognition with spatially chaotic dynamics. Neural Netw 3:153–170
Yoshinaka R, Kawashima M, Nabeta K, Li Y, Nara S (2012) Adaptive control of robot systems with simple rules using chaotic dynamics in quasi-layered recurrent neural networks. In: Madani K, Correia AD, Rosa A, Filipe J (eds) Computational intelligence. Springer, Berlin Heidelberg, pp 287–305