Self-Organizing Homotopy Network

Tetsuo Furukawa
Department of Brain Science and Engineering, Kyushu Institute of Technology
2–4 Hibikino, Wakamtatsu-ku, Kitakyushu 808–0196, Japan
e-mail: furukawa@brain.kyutech.ac.jp

Keywords: homotopy, fiber bundle, SOM^2, mnSOM

Abstract— In this paper, we propose a conceptual learning algorithm called the '*self-organizing homotopy* (SOH)' together with an implementation thereof. As in the case of the SOM, our SOH organizes a homotopy in a self-organizing manner by giving a set of data episodes. Thus it is an extension of the SOM, moving from a '*map*' to a '*homotopy*'. From a geometrical viewpoint, the SOH represents a set of (i.e. multiple) data distributions by a fiber bundle, whereas the SOM represents a single data distribution by a manifold. Therefore, this paper also proposes the concept of '*fiber bundle learning*" as an extension of *manifold learning*. One of the solutions to the SOH is SOM^2, in which every reference vector unit of the conventional SOM is itself replaced by an SOM. Consequently SOM^2 has the ability to represent a fiber bundle, i.e. a product manifold, by using a product space of SOM×SOM. It is also possible to design SOM^n to represent higher order fiber bundles. It is expected that SOHs will play important roles in the fields of pattern recognition, adaptive functions, context understanding, and others, in which nonlinear manifolds and the homotopy play crucial roles.

1 Introduction

Kohonen's self-organizing map (SOM) is capable of acquiring a map from a high-dimensional data vector space to a low-dimensional space, thus representing the distribution of given data vectors naturally. From a geometrical viewpoint, the SOM is an unsupervised learning machine which approximates the given data distribution by fitting a low-dimensional nonlinear manifold. Therefore the SOM incorporates two aspects, namely 'map learning' and 'manifold learning'.

If a map is continuously deformed to another map, this continuous deformation between the two maps is called a *homotopy*. Thus a homotopy is useful for representing the continuous change of a set of data distributions, a set of input-output functions, or a set of system dynamics, for example, which are modulated continuously by the context or the environment. Incorporating this concept, we propose the novel concept of an unsupervised learning machine. Such a learning machine, which is capable of acquiring a homotopy from a series of data sets in a self-organizing manner, is called a "*self-organizing homotopy* (SOH)". As in the case of the SOM, the SOH incorporates two aspects. The first is "*homotopy learning*" and the second is "*fiber bundle learning*". A fiber bundle is a geometrical concept representing a kind of product manifold, which consists of a bundle of fibers connecting a series of congruent manifolds. If a manifold is deformed continuously then the entire trajectory of the manifold forms a fiber bundle. Therefore the fiber bundle is another mathematical aspect of the SOH. In other words, an SOH is a learning machine which represents a group of data distributions by a fiber bundle, i.e. a series of manifolds, whereas the conventional SOM represents a single data distribution by a manifold.

It is well known that the nonlinear manifold plays a crucial role in many pattern classification tasks. Often a class of data vectors is distributed in an inherent nonlinear manifold, and different classes are distributed in different manifolds. In such cases 'pattern classification' means determining the manifold, to which given data vectors should belong. One of the most typical examples of this is face image classification [1]. A set of face images of the same person taken from various camera angles is known to form a 'face manifold', because the continuous change in camera angle causes a continuous change in the face images. Since each set of face images corresponds to its inherent face manifold, the essence of face image classification is to determine the face manifold, to which a given face image belongs.

In the above scenario, a manifold is obtained by changing the camera angle continuously, but there is another way to obtain a manifold of face images. Suppose that the face shape is flexible like a rubber mask. By stretching the rubber mask, we can obtain another continuous change in the face images, i.e. another manifold. The set of face images taken from various view points and of various face shapes forms a product manifold, which consists of a group of weft manifolds (i.e. manifolds of various faces) and of a group of warp manifolds (i.e. manifolds of camera angles). Thus the entire geometric structure becomes a fiber bundle. This is the reason why homotopy learning or fiber bundle learning is important.

Actually the importance of the concept has been recognized in the field of the SOM. The adaptive subspace SOM (ASSOM) and the self-organizing operator map (SOOM) are examples of the architectures that can organize a homotopy [2, 3]. However, ASSOM and SOOM deal with linear cases only. Furthermore these algorithms do not

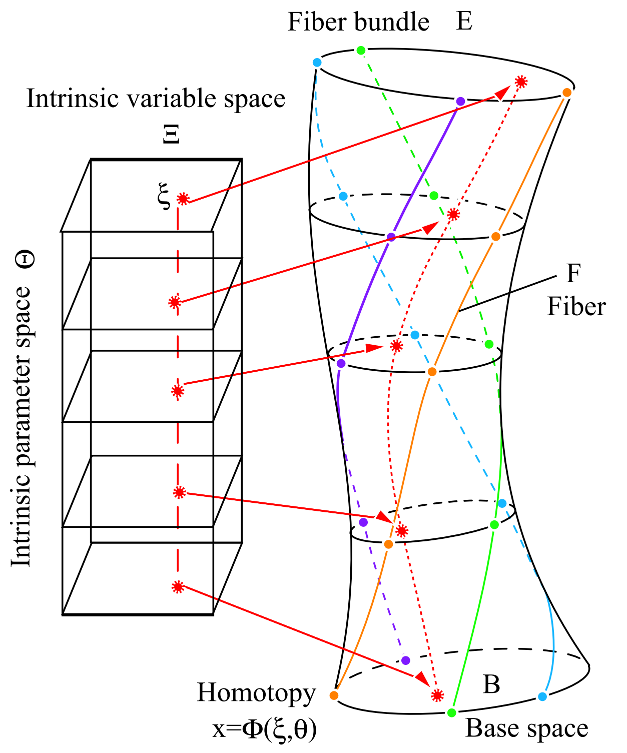


Figure 1: The framework dealt with by SOH.

have the ability to represent fibers. One of the real solutions to homotopy learning is the SOM^2 proposed by the author [4, 5, 6]. SOM^2 has an arrayed structure of conventional SOM modules (called '*child SOMs*' or '*child maps*'), each of which represents a nonlinear manifold. At the same time, the upper level of the SOM^2 (called the '*parent SOM*' or '*parent map*') represents a continuous change of the child maps. As a result, the entire SOM^2 represents a product manifold created by the product of (*child SOM*)×(*parent SOM*). In addition, a set of reference vectors with the same index of child SOMs forms a continuous string, namely, a fiber. Therefore SOM^2 is capable of representing a fiber bundle as well. In this paper, the theoretical framework of the SOH is first described. SOH is a conceptual algorithm rather than a concrete one, because some of its procedures are left to the users to implement. After that the theory of and algorithm for the SOM^2 are presented as an actual implementation of the SOH. This is followed by some simulation results.

2 Framework

First of all, let us clarify the framework of the situation dealt with by the SOH. As is the case in the ASSOM, *episodes* are the elemental units in homotopy learning. An episode means a set of data vectors observed together, e.g. measured from the same system or obtained in the same context. Let $D_i = \{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,J}\}$ be the i-th episode[1], the distribution of which is represented by a nonlinear manifold U_i in the high-dimensional data space X. Suppose that D_i is generated by a map $\mathbf{x} = \phi_i(\xi)$ and a probability density function (pdf) $p(\xi)$. Here $\xi \in \Xi$ is a intrinsic (hidden) variable and ϕ_i is a homeomorphic map from Ξ to U_i. Thus,

$$\phi_i : \Xi \longrightarrow U_i \tag{1}$$

$$\xi \longmapsto \mathbf{x}. \tag{2}$$

In the case of face images, ξ and $\mathbf{x}$ mean the camera angle and the corresponding face image respectively. For convenience of explanation, let us assume that $\Xi = I^2$ and $p(\xi)$ is a uniform distribution, i.e. $p(x) = 1$ (Figure 1). In this case, U_i is a 2-dimensional nonlinear bounded manifold in X. In this situation, the episode D_i can be obtained by repeating trials J times with $p(\xi)$ and $\phi_i(\xi)$.

Suppose we have I episodes $\{D_1, \ldots, D_I\}$, each of which is generated by the pdf $p(\xi)$ and a set of maps $\{\phi_1(\xi), \ldots, \phi_I(\xi)\}$. Note that $p(\xi)$ is common for all episodes. Furthermore, suppose that the $\{\phi_i(\xi)\}$ are represented by a continuous deformation of a map modulated by an intrinsic (hidden) parameter $\theta \in \Theta$ as follows.

$$\phi_i(\xi) = \Phi(\xi, \theta_i) \tag{3}$$

$\Phi(\xi, \theta)$ is the homotopy underlying the phenomena, and Φ maps the intrinsic product space, i.e., the product of the intrinsic variable and the parameter $\Xi \times \Theta$ to a fiber bundle E.

$$\Phi : \Xi \times \Theta \longrightarrow E \tag{4}$$

In the case of face images, θ is the parameter that determines the face shape. Thus a 3D face shape is determined by the intrinsic parameter θ, and then a map from a camera angle to the corresponding 2D face image is determined. Using the homotopy $\Phi(\xi, \theta)$, a section of the fiber bundle is obtained by fixing θ, while a fiber F_ξ is obtained by fixing ξ. Figure 1 shows a case in which the dimension of the intrinsic parameter space Θ is one, i.e. $\Theta = I$. Furthermore, suppose that θ is also a random variable obeying $q(\theta)$. We can assume $q(\theta)$ is a uniform distribution in I, i.e. $q(\theta) = 1$, without losing generality. Under these circumstances, we obtain a set of parameters $\{\theta_1, \ldots, \theta_I\}$ from I-number of trials with $q(\theta)$, and then making J-number of trials with $p(\xi)$ for each parameter, we obtain an episode set $\{D_i\}$. This is the stochastic generation model of the episodes.

Using this mathematical framework, the aim of the SOH is to estimate the homotopy $\Phi(\xi, \theta)$ from the given episodes in a self-organizing manner. Please note that the generation model is presented to clarify and to simplify the situation dealt with by SOH, and it is usually veiled under the phenomena in the actual tasks.

[1]In this paper, subscripts denote indexes of episodes and data vectors, while superscripts denote indexes of SOHs and SOMs. In addition, capital letters denote the upper limits of the indexes. For example, I and J denote the upper limits of the indexes i and j respectively.

3 The SOH Algorithm

We now introduce the ideal algorithm for an SOH, which has the ability to represent a homotopy $\Psi(\xi,\theta)$. $\psi_\theta(\xi) = \Psi(\xi,\theta)$ maps Ξ to a manifold M_θ.

$$\Psi : \Xi \times \Theta \longrightarrow E \tag{5}$$
$$\psi_\theta : \Xi \longrightarrow M_\theta \tag{6}$$

M_θ is a section of E. We also assume that we have the following two procedures, which can be used to construct the SOH algorithm.

Procedure 1: *Procedure 1* is the algorithm for estimating the intrinsic parameter θ_i of the episode D_i. If the manifold U_i is known, then the estimated parameter $\hat{\theta}_i$ is given as

$$\hat{\theta}_i = \arg\min_\theta L(U_i, M_\theta). \tag{7}$$

Here $L(U_\theta, M_\theta)$ is the distance between two homeomorphic manifolds generated by two homotopic maps Φ and Ψ. Thus,

$$L^2(U_i, M_\theta) = \int_{\xi\in\Xi} \|\Phi(\xi,\theta_i) - \Psi(\xi,\theta)\|^2 \, p(\xi)\, d\xi. \tag{8}$$

However U_i, $\Phi(\xi,\theta)$ and θ_i are all unknown in a real example. It is therefore assumed that the episode map $\hat{\phi}_i(\xi)$ is estimated initially, before applying this procedure. The distance between two manifolds is then estimated by

$$\hat{L}^2(U_i, M_\theta) = \int_{\xi\in\Xi} \left\|\hat{\phi}_i(\xi) - \Psi(\xi,\theta)\right\|^2 p(\xi)\, d(\xi). \tag{9}$$

Using this estimated distance, *Procedure 1* can determine $\hat{\theta}_i$, which is regarded as '*the best matching* θ' or '*the winner* θ' of the episode D_i.

Procedure 2: *Procedure 2* is the algorithm for estimating the map of each episode $\{\hat{\phi}_i(\xi)\}$ from the winner map $\psi_{\hat{\theta}_i}(\xi) = \Psi(\xi,\hat{\theta}_i)$. From another point of view, this procedure estimates where each fiber passes through the manifold U_i.

We now describe the entire SOH algorithm, except for the details of how the two procedures given above are implemented. The algorithm consists of an iterative loop with 4 steps.

(i) *Determining the winner sections*
For every episode D_i, the estimated parameter $\hat{\theta}_i$ is calculated by *Procedure 1*. The section $M_{\hat{\theta}_i}$ and $\psi_{\hat{\theta}_i}$ become respectively, *the winner section* and *the winner map* of D_i.

(ii) *Estimating fibers and episode maps*
Using *Procedure 2*, the episode map $\hat{\phi}_i(\xi)$ is estimated from the winner map $\psi_{\hat{\theta}_i}(\xi)$. This step also estimates the positions where fibers pass through U_i.

(iii) *Evaluating learning mass distribution*
Learning masses $m_i(\theta)$ are calculated using the neighborhood function.

$$m_i(\theta) = h\left(\|\theta - \hat{\theta}_i\|\right) \tag{10}$$

where $h(\cdot)$ is the neighborhood function.

(iv) *Updating the homotopy*
Finally, the homotopy represented by the SOH is updated so that it becomes the mass centre of the estimated episode maps with the learning mass distribution.

$$\Psi(\xi,\theta) := \frac{m_1(\theta)\hat{\phi}_1(\xi) + \cdots + m_I(\theta)\hat{\phi}_I(\xi)}{m_1(\theta) + \cdots + m_I(\theta)} \tag{11}$$

The algorithm iterates through these four steps reducing the size of the neighborhood function until the network reaches a steady state.

The four steps can be summarized as follows. In step (i), the $\{\hat{\theta}_i\}$ are estimated by fixing $\Psi(\xi,\theta)$ and $\{\hat{\phi}_i(\xi)\}$, while in step (ii), the $\{\hat{\phi}_i(\xi)\}$ are estimated by fixing $\Psi(\xi,\theta)$ and $\{\hat{\theta}_i\}$. Finally in step (iii) and (iv), $\Psi(\xi,\theta)$ is updated by fixing $\{\hat{\phi}_i(\xi)\}$ and $\{\hat{\theta}_i\}$.

There are several ways to implement *Procedures 1* and *2*, and this affects the performance of SOH. Sophisticated algorithms may produce better results, but they may also consume more computational time. The implementation therefore depends on the users or tasks. One of the simplest and fastest solutions is the SOM^2 algorithm described in the next section.

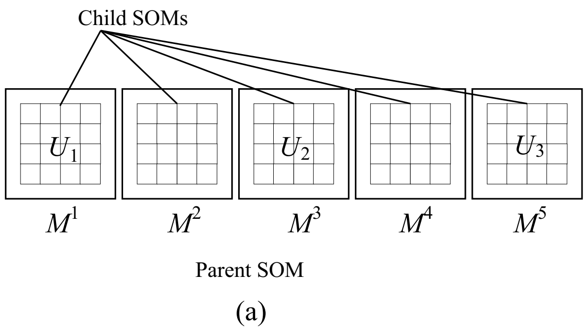
4 SOM^2 Algorithm

In this section SOM^2 is introduced as one of the actual implementations of the SOH. The name "SOM^2" is derived from "SOM×SOM", because it represents a product manifold described by two levels of SOMs.

The architecture of SOM^2 is an assembly of SOM modules arrayed on a lattice, like the reference vector units of the conventional SOM. Each of these SOMs is called a '*child SOM*', and the upper level of the SOM consisting of child SOMs is called the '*parent SOM*'. The SOM^2 in Figure 2 (a) has 5 child SOMs, each of which represents an individual manifold $\{M^1,\ldots,M^5\}$. In this case, the topology of the parent SOM is one-dimensional, but of course it is possible to have 2 or more dimensions. The parent SOM is expected to represent a continuous change of maps represented by the child SOMs, i.e. a homotopy.

Figure 2 (b) shows a simulation result after learning 3 episodes. In this case, 3 episodes D_1, D_2 and D_3 are given, which are distributed in square shape manifolds U_1, U_2 and U_3. After learning these episodes, the SOM^2 sorts the given episodes so that they show a continuous change of the maps naturally. In addition the SOM^2 interpolates between the given manifolds, such that intermediate child SOMs represent intermediate distributions between episodes. This means that these intermediate child SOMs generated their maps where there were no data points. Furthermore we can draw a set of strings, i.e. *fibers*, between the child SOMs, by connecting the reference vectors with the same index. Such fibers represent the gradual change of these manifolds. Therefore the SOM^2 also represents a fiber bundle. Note that the child SOMs are organized by mutual interactions between themselves. If they were to construct their maps independently, the SOM^2 would not be able to make the appropriate interpolations between given episodes.

Now let us suppose that we have an SOM^2 with K child SOMs, each of which has L reference vectors. Thus there are $K \times L$ reference vectors denoted as $\{\mathbf{w}^{k,l}\}$. Let W^k refer to the jointed reference vectors of the k-th child SOM, i.e. $W^k = (\mathbf{w}^{k,1}, \ldots, \mathbf{w}^{k,L})$. Therefore W^k represents the k-th section M^k of the fiber bundle E. Now suppose that we have an additional set of I child SOMs which are not built into the SOM^2 architecture (i.e. not shown in Figure 2).



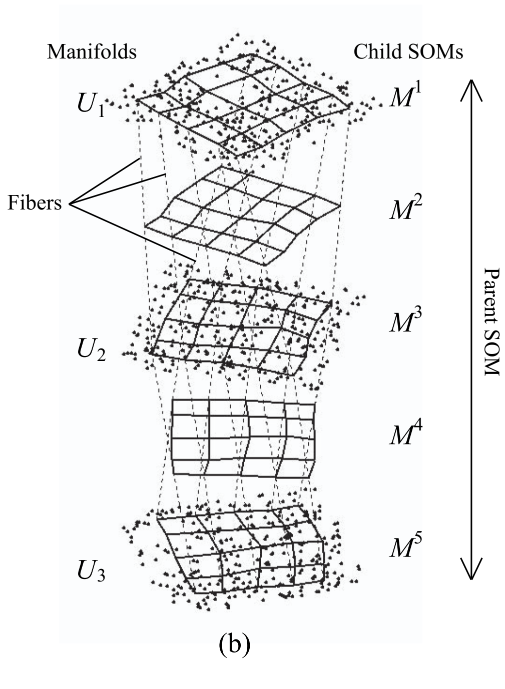


Figure 2: (a) The SOM^2 architecture. (b) Results of estimating a homotopy from 3 episodes using SOM^2.

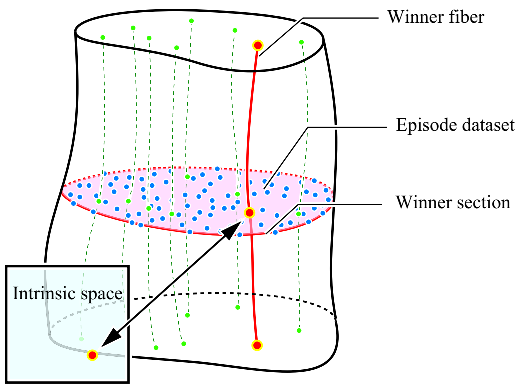


Figure 3: How to determine the winner section and the winner fiber in SOM^2

These child SOMs are expected to represent the estimated episode maps. Let $\{\mathbf{v}_i^l\}$ be the reference vectors of the additional set of SOMs, then V_i denotes the jointed vectors $V_i = (\mathbf{v}_i^1, \ldots, \mathbf{v}_i^L)$. V_i represents the distribution of the i-th episode, i.e. U_i. This additional set of child SOMs is useful only in the explanation of the algorithm. It is not at all needed in the actual implementation thereof.

In the algorithm for SOM^2, the mean square error between an episode and a child SOM is evaluated to estimate the distance between them. This is *Procedure 1* of the SOM^2. In *Procedure 2*, the SOM^2 executes the batch-SOM algorithm only once to estimate each episode map. Using these procedures, the SOM^2 algorithm can be described as follows.

(i) *Determining the winner section*

To determine the winner section of each episode, the mean square error between data points and the closest reference vectors is evaluated for every combination of episode and child SOM. Let $l_{i,j}^{k,*}$ denote the winner index of $\mathbf{x}_{i,j}$ in the k-th child SOM. Thus,

$$l_{i,j}^{k,*} \triangleq \arg\min_l e_{i,j}^{k,l} \tag{12}$$

$$e_{i,j}^{k,l} \triangleq \left\| \mathbf{x}_{i,j} - \mathbf{w}^{k,l} \right\|^2 . \tag{13}$$

Then the mean square error between the i-th episode and the k-th Episode is defined as

$$E_i^k \triangleq \frac{1}{J} \sum_{j=1}^{J} e^k(\mathbf{x}_{i,j}). \tag{14}$$

The index of the winner child SOM, i.e. the winner section, is determined as follows.

$$k_i^* \triangleq \arg\min_k E_i^k \tag{15}$$

(ii) *Estimating fibers and episode maps*

In the second step, the positions where fibers pass through every section are estimated first. In SOM^2, data point $\mathbf{x}_{i,j}$ is assumed to belong to the nearest fiber in the winner section. Thus the index of the winner fiber $l_{i,j}^{*,*}$ is denoted as $l_{i,j}^{*,*}$. This situation is depicted in Figure 3. After determining the winner fibers, the episode maps are estimated as follows.

$$\mathbf{v}_i^l = \sum_{j=1}^{J} \beta_{i,j}^l \, \mathbf{x}_{i,j} \tag{16}$$

In this equation, $\beta_{i,j}^l$ is the normalized learning rate calculated from the neighborhood function of the child SOM.

$$\beta_{i,j} = \frac{h_c\left[d_c(l, l_{i,j}^{*,*});\ \sigma_c(t)\right]}{\sum_{j'=1}^{J} h_c\left[d_c(l, l_{i,j'}^{*,*});\ \sigma_c(t)\right]} \tag{17}$$

$h_c[l;\sigma]$ is the neighborhood function at the child level, and $\sigma_c(t)$ denotes the neighborhood size at the time of calculation t. These formulas mean that the episode map V_i is estimated by applying the batch-SOM algorithm once, with the initial state $W^{k_i^*}$.

(iii) Evaluating learning masses

In the third step, the learning rates at the parent level are calculated.

$$\alpha_i^k = \frac{h_p\left[d_p(k, k_i^*;\ \sigma_p(t))\right]}{\sum_{i'=1}^{I} h_p\left[d_p(k, k_{i'}^*);\ \sigma_p(t))\right]} \tag{18}$$

$h_p[l;\sigma]$ and $\sigma_p(t)$ are the neighborhood function of the parent SOM and its size respectively.

(iv) Updating the homotopy

Finally all reference vectors are updated so that every child SOM represents the mass centre of the episode maps with the learning masses.

$$W^k = \sum_{i=1}^{I} \alpha_i^k \, V_i \tag{19}$$

By combining (16) and (19), we obtain the following update algorithm.

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I}\sum_{j=1}^{J} \alpha_i^k \, \beta_{i,j}^l \, \mathbf{x}_{i,j} \tag{20}$$

Since (20) does not contain the episode map $\{\mathbf{v}_i^l\}$, we can update SOM^2 without estimating each episode map $\{V_i\}$. We iterate through these steps, reducing the neighborhood size until the network reaches a steady state.

The above algorithm for SOM^2 may sound very complicated, but the reason for this is that it is described from the viewpoint of the SOH. We can simplify the SOM^2 algorithm as follows.

(i) Child competition phase. The winner unit is determined by (13) within each child SOM for every data vector.

(ii) Parent competition phase. The winner map is determined by (15) for every episode.

(iii) Evaluating child neighborhood phase. The neighborhood function at the child level is evaluated by (17).

(iv) Evaluating parent neighborhood phase. The neighborhood function at the parent level is evaluated by (18).

(v) Updating phase. All reference vectors are updated by (20).

The algorithm therefore has a nested structure of child and parent levels.

Interestingly, it is also possible to construct an additional nested structure, as in the nesting of Russian dolls. For example, SOM^2 can be a module of a meta-class of the SOM. Thus SOM×SOM×SOM, i.e. SOM^3, is also possible [4]. In the case of SOM^3, the update algorithm is described as follows.

$$\mathbf{w}^{l,m,n} = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K} \alpha_i^l \, \beta_{i,j}^m \, \gamma_{i,j,k}^n \, \mathbf{x}_{i,j,k} \tag{21}$$

$$\tag{22}$$

Here α_i^l, $\beta_{i,j}^m$ and $\gamma_{i,j,k}^n$ are the normalized neighborhood functions at parent, child, and grandchild levels respectively, given by

$$\alpha_i^l = \frac{h_\alpha\left[d_\alpha(l, l_i^*),\ \sigma_\alpha(t)\right]}{\sum_{i'=1}^{I} h_\alpha\left[d_\alpha(l, l_{i'}^*),\ \sigma_\alpha(t)\right]} \tag{23}$$

$$\beta_{i,j}^m = \frac{h_\beta\left[d_\beta(m, m_{i,j}^{*,*}),\ \sigma_\beta(t)\right]}{\sum_{j'=1}^{J} h_\beta\left[d_\beta(m, m_{i,j'}^{*,*}),\ \sigma_\beta(t)\right]} \tag{24}$$

$$\gamma_{i,j,k}^n = \frac{h_\gamma\left[d_\gamma(n, n_{i,j,k}^{*,*,*}),\ \sigma_\gamma(t)\right]}{\sum_{k'=1}^{J} h_\gamma\left[d_\gamma(n, n_{i,j,k'}^{*,*,*}),\ \sigma_\gamma(t)\right]}. \tag{25}$$

It is easy to extend the n-th order case, namely SOM^n, that represents the $(n-1)$ order of a homotopy. In this context, the conventional SOM is regarded as SOM^1, while a single reference vector unit (i.e. a Hebb neuron) corresponds to SOM^0. Therefore the conventional model is a member of the SOM^n family.

We can extend the SOM^2 further. In the case of SOM^2, the architecture of both parent and child levels consists of SOMs, but it is also possible to use neural gas (NG), radial basis function (RBF), or other maps and manifolds representing networks [4]. For example, the architecture consisting of a parent SOM and child NGs, known as NG×SOM,

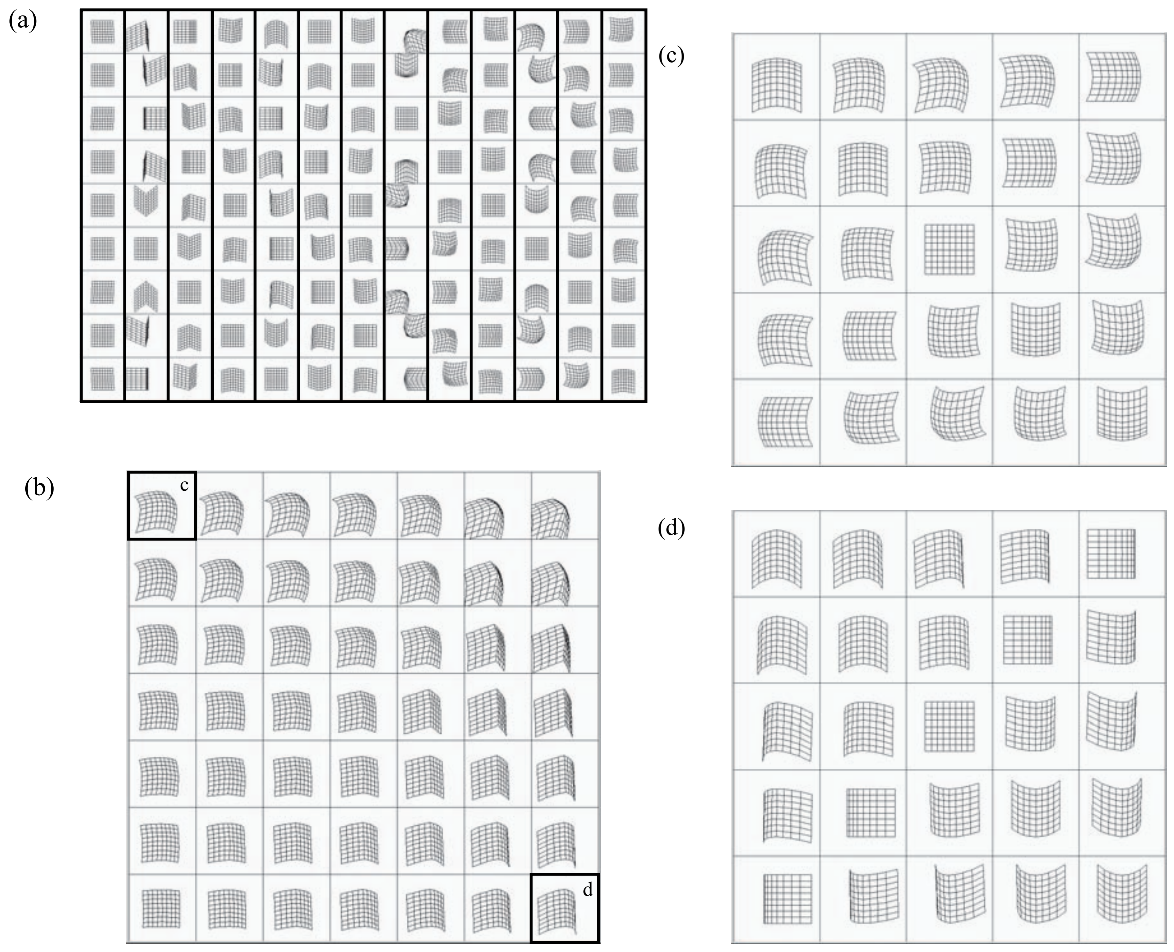


Figure 4: Map of 3D objects generated by an SOM^2 constructed from 2D images. (a) 13 episodes used in the simulation (thick boxes). Each 2D image is vectorized by jointing the (x, y) coordinates of mesh nodes. (b) The parent map represents various 3D shapes. (c) and (d) Child maps represent various viewpoints.

would be a good solution if the topology of episodes are unknown. NG×SOM also deals with the case in which every data distribution does not form a manifold, e.g. a topology with some branches or with some singular points. Growing type architectures are also available to use at the parent and/or child level, and we are currently developing this model.

5 Simulation results

In the first simulation, a map of 3D objects is generated by SOM^2 from sets of 2D images. We prepared 13 episodes of 2D images of 3D objects shown in Figure 4 (a). Each 2D image is vectorized by giving (x, y) coordinates of the grid points, i.e. $\mathbf{x} = (x_1, y_1, x_2, y_2, \ldots, x_N, y_N)$. N is the number of grid points, with $N = 81$ in this simulation. After learning these episodes, the parent SOM shows a map of 3D shapes (Figure 4 (b)). The SOM^2 successfully interpolates between the given objects, and the entire map represents a continuous change of the 3D shapes. Figure 4 (c) and (d) are the child maps (indicated by thick boxes in Figure 4 (b)). Each child map shows a continuous change of the viewpoint, by interpolating intermediate 2D images between the given ones. Note that the corresponding units of child SOMs represent the view from the same angle.

Figure 5 is a map of faces generated by SOM^2. In this simulation, every episode consists of a set of facial images of the same person taken from various view points. Since the order of the images was randomly shuffled, no clues were given as to camera angles. Sets of facial images of 12 people, namely 12 episodes, were given to an SOM^2 without any image preprocessing. Every subject was taken a set of photographs with 75×75 pixels from 17 different angles. Thus each episode consists of 17 data vectors, the dimension of which is 5225.

The map shown in Figure 5 (a) is not the map of front face images, but is a map of face manifolds. Figure 5 (b) and (c) show the child maps. Both child maps have successfully organized a continuous change of images based on the camera angle. Furthermore, the corresponding units

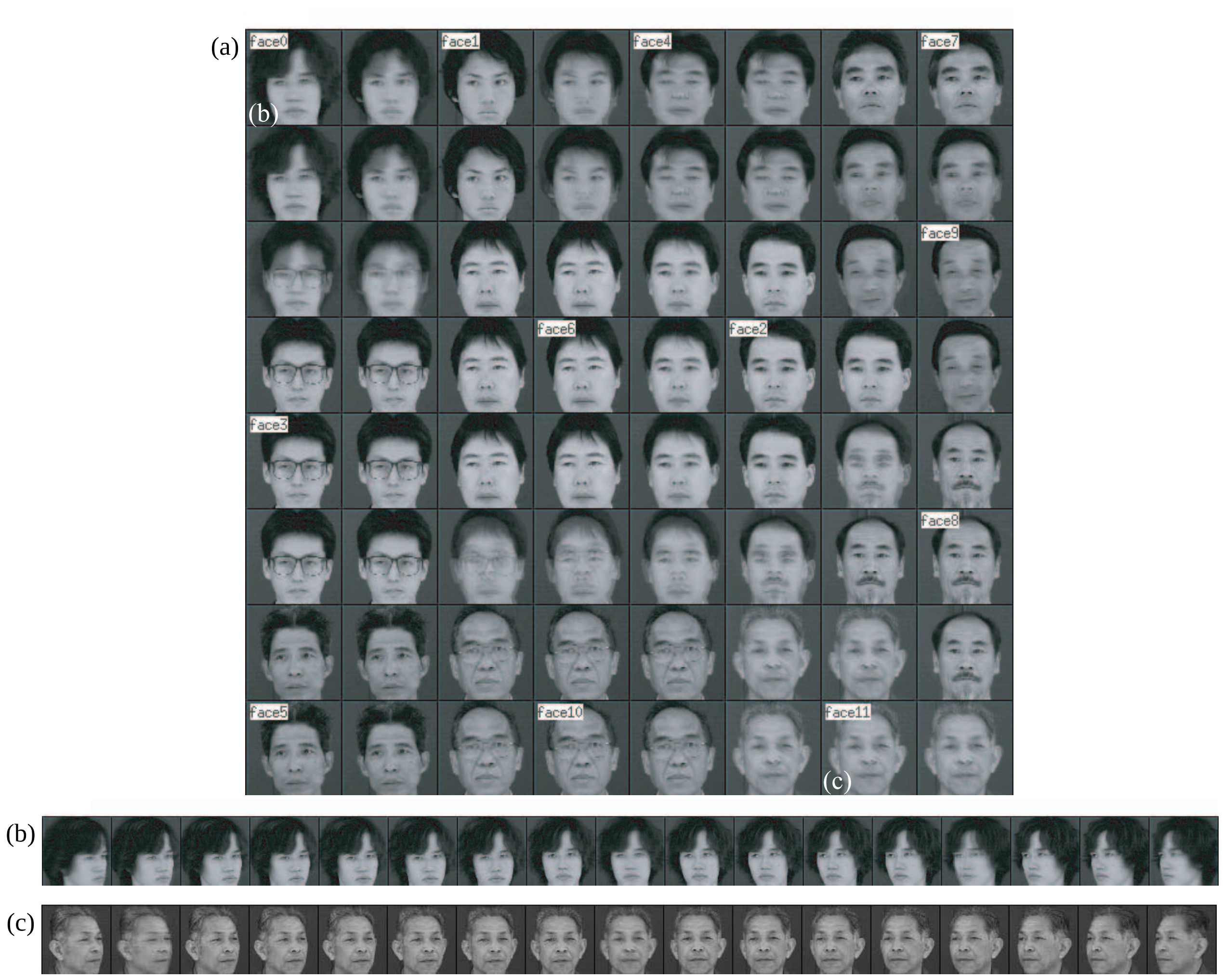


Figure 5: ‘Face map’ generated by SOM^2. The parent map (a) and the child maps of face0 (b) and face11 (c).

of each child SOM represent the same camera angle. These results are consistent in all child SOMs. Thus, it can be concluded that the SOM^2 represents a fiber bundle of face images. These properties would be useful in simultaneous recognition of both face and pose.

Figure 6 is a map of alphabet shapes organized by NG×SOM. In this example, every alphabet shape is represented by a number of small dots, and the coordinate of each dot is regarded as a data vector. Thus every letter is resolved into 200 dots, then a set of coordinates $\{(x_1, y_1), \ldots, (x_{200}, y_{200})\}$ is regarded as an episode observed from the letter. 26 handwritten letters, i.e. 26 episodes were given to the NG×SOM, and the SOM^2 organized a map, in which similar shapes of letters were located closer together in the map.

These simulation results were always consistent in every trial, suggesting the robustness of this method.

6 Discussion

It is possible to design more sophisticated SOH algorithms than SOM^2. For example, the performance of estimating a homotopy would be improved by using the iterative closest point (ICP) algorithm. This type of SOH would be expected to generate a better alphabet map. However, inserting an iterative process into the iteration of the SOH algorithm may increase the calculation cost.

Though many manifold learning algoirthms have been proposed, there are few fiber bundle learning algorithms as far as the author has investigated. Therefore it is hard to compare with other methods. One can use kernel methods, auto-associative neural networks (sand-clock type multilayer perceptron), nonlinear-PCA and other types of manifold learning algorithms as the modules of the SOH. In all cases though, we need to build in a process of estimating fibers, i.e. corresponding points between manifolds. Without this process, modules would express manifolds independently, and the intermediate modules would never represent intermediate manifolds. The mutual interactions be-

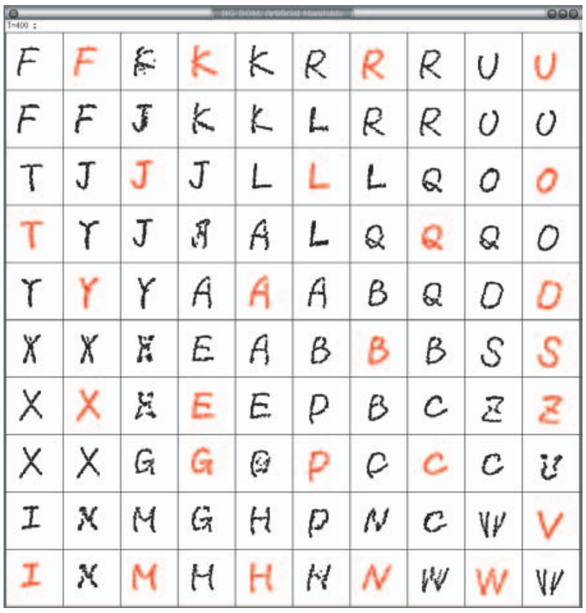


Figure 6: Map of alphabet shapes generated by NG×SOM

tween child modules are thus essential for estimating a fiber bundle.

Apart from the SOH, it seems possible to design other architectures representing a homotopy. For example, multi-layer perceptrons (MLPs) with plan units or parametric bias units can represent a set of functions by switching the additional units [7]. However there is no theoretical assurance that the network represents intermediate function shapes by setting the intermediate values of the additional units. Simulation results have also shown that these methods do not represent a homotopy well [8]. The modular network SOM with MLP modules can deal with a homotopy [9, 10, 11], and would be a good solution in some cases. But its ability is limited to the cases when trivial fibers exist in the data distributions. Some other architectures may work well in similar easy cases. However, iterative and reciprocal estimation of sections and fibers, like SOH, would be required to solve the problem in general cases.

7 Conclusion

In this paper, novel learning concepts of ‘homotopy learning’ and ‘fiber bundle learning’ have been proposed. We have also proposed the SOH algorithm, which is the theoretical description of the learning scheme. One of the solutions for the SOH is SOM^2, which has previously been proposed by the author. In this paper, the SOM^2 algorithm has been re-introduced from the viewpoint of the SOH. The concept of SOM^2 has been generalized to the case of SOM^n, which comprises the conventional SOM case as SOM^1. SOH and its variations are not only applicable to pattern classification tasks, but would also be useful in cases related to multi-functions, multi-systems and others, modulated by the context or the environment.

Acknowledgements

This work was partially supported by the 21th Century Center of Excellence Program to Kyushu Institute of Technology (Center #J19) by MEXT of Japan. This work was also partially supported by a Grant-in-Aid for Scientific Research (C) granted by MEXT of Japan. The facial data in this paper has been used with permission of Softopia Japan, Research and Development Division, HOIP Laboratory.

References

[1] J. Jiang, L. Zhang and T. Furukawa, “Improving the generalization of Fisherface by training class selection using SOM^2,” *Lecture Notes in Computer Science*, Vol.4233, pp.278–285, 2006.

[2] T. Kohonen, S. Kaski and H. Lappalainen, “Self-organized formation of various invariant-feature filters in the adaptive-subspace SOM,” *Neural Computation*, Vol.9, pp.1321–1344, 1993.

[3] T. Kohonen, “Generalization of the self-organizing map,” *Proc. of IJCNN93*, pp.457–462, 1993.

[4] T. Furukawa, “SOM^2 as SOM of SOMs,” *Proc. of WSOM2005*, pp.545–552, 2005.

[5] T. Furukawa, “SOM of SOMs: Self-organizing map which maps a group of self-organizing maps,” *Lecture Notes in computer Science*, Vol.3696, pp.391–396, 2005.

[6] T. Furukawa, “An extension SOM from ‘map’ to ‘homotopy’,” *Lecture Notes in Computer Science*, Vol.4232, pp.958–967, 2006.

[7] J. Tani, M. Ito, and Y. Sugita, “Self-organization of distributedly represented multiple behavior schemata in a mirror system: Reviews of robots using RNNPB,” *Neural Networks*, Vol.17, pp.1273-1289, 2004.

[8] T. Ohkubo, K. Tokunaga and T. Furukawa, “Self-organizing homotopy networks: Comparisons between modular network SOM, SOM of SOMs, and parametric bias method,” Brain-inspired IT III, International Congress Series, (in press).

[9] K. Tokunaga, T. Furukawa and S. Yasui, “Modular network SOM: Extension of SOM to the realm of function space,” *Proc. of WSOM2003*, pp.173–178, 2003.

[10] T. Furukawa, K. Tokunaga, K. Morishita and S. Yasui, “Modular network SOM (mnSOM): From vector space to function space,” *Proc. of IJCNN2005*, pp.1581–1586, 2005.

[11] S. Kaneko, K. Tokunaga, and T. Furukawa, “Modular network SOM: The architecture, the algorithm and applications to nonlinear dynamical systems,” *Proc. of WSOM2005*, pp.537–544, 2005.