

# Tentacled Self-Organizing Map for Effective Data Extraction

Haruna Matsushita, *Student Member, IEEE*, and Yoshifumi Nishio, *Member, IEEE*

**Abstract**—Since we can accumulate a huge amount of data including useless information in these years, it is important to investigate various extraction method of clusters from data including a lot of noises. The Self-Organizing Map (SOM) attracts attentions for clustering in these years. In our past study, we have proposed a method of using simultaneously two kinds of SOMs whose features are different ( $n$ SOM), namely, one self-organizes the area on which input data are concentrated, and the other self-organizes the whole of the input space. Further, we have applied this method to clustering of data including a lot of noises and have confirmed the efficiency. However, in order to obtain an efficient clustering performance using this method, we must determine the appropriate number of the SOMs used in the method. This problem has been remedied by proposing the Peace SOM (PSOM) method, however, PSOM algorithm must be used after executing the  $n$ SOM method.

In this study, we propose a method of using plural SOMs (TSOM: Tentacled SOM) for effective data extraction, which possesses both abilities of  $n$ SOM and PSOM. Each SOM of TSOM can catch the information of other SOMs existing in its neighborhood and self-organizes with the competing and accommodating behaviors. The behavior of TSOM is investigated with applications to data extraction from input data including a lot of noises. We can confirm that TSOM successfully extracts clusters even in the case that we do not know the number of clusters in advance.

## I. INTRODUCTION

Since we can accumulate a huge amount of data including useless information in these years, it is important to investigate various extraction methods of clusters from data including a lot of noises. Then, the Self-Organizing Map (SOM) attracts attentions in recent years. SOM is an unsupervised neural network introduced by Kohonen in 1982 [1] and is a model simplifying self-organization process of the brain. SOM can obtain a statistical feature of input data and can be applied to a wide field of data classifications [2]-[5]. Although many methods to extract clusters by using SOM have been proposed, it seems to be very difficult to construct a simple method of using SOM for universal input data.

In our past study, we have proposed a method of using simultaneously two kinds of SOMs whose features are different ( $n$ SOM method) [6]. In the  $n$ SOM method, one self-organizes the area on which input data are concentrated, and the other self-organizes the whole of the input space. We call the former  $SOM_L$  and the latter  $SOM_G$ . We have investigated competing behavior of  $n$ SOM caused by the difference of the initial states and the neighborhood functions. Furthermore, we have applied  $n$ SOM to clustering of data

including a lot of noises and have confirmed the efficiency. However,  $n$ SOM method has two disadvantages in order to obtain an efficient clustering performance.

- 1) We must set the initial state of  $SOM_G$  avoiding clusters.
- 2) We must determine the appropriate number of SOMs, namely, we must know the number of clusters in advance.

Regarding the case 1), it is difficult to set the initial state properly in higher-dimensional space. Regarding the case 2), if we use too many SOMs in  $n$ SOM method, some SOMs compete with each other in the same cluster and the one cluster is extracted as wrong plural clusters.

We have also proposed the Peace SOM algorithm [7] which remedies the above problem 2). PSOM possesses both competing and accommodating abilities and is used after executing the  $n$ SOM method. This algorithm makes the competing SOMs in the same cluster to accommodate the cluster, while the SOMs situated far keeping to compete with other SOMs. We have investigated the competing and the accommodating behaviors of PSOM with applications to clustering input data including a lot of noises. However, the learning process is a little bit complicated, because the PSOM algorithm must be used after executing the  $n$ SOM method, namely it is the two-tiered learning.

In this study, we propose a method of using plural SOMs (TSOM: Tentacled SOM) for effective data extraction. TSOM method possesses both abilities of  $n$ SOM and PSOM, namely, TSOM has context sensitive behaviors of competing and accommodating. The plural SOMs of TSOM contains the two kinds of SOM. The two kinds of SOM are  $SOM_L$  and  $SOM_G$ .  $SOM_L$  self-organizes the area on which input data are concentrated, while  $SOM_G$  self-organizes the whole of the input space, in the same way as  $n$ SOM method. In the TSOM algorithm, one  $SOM_G$  and plural  $SOM_L$  are used. Each SOM of TSOM can catch the information of other SOMs existing in its neighborhood and self-organizes with the competing and accommodating behaviors — hence the name “tentacled”. TSOM has the following three key rules.

1. Winner SOM: All SOMs of TSOM have one winner neuron, respectively. However, neurons of all SOMs are NOT updated. One or more winner SOMs are decided for one input data, and the weight vectors of only the winner SOM are updated.
2. The update direction  $y_{G_i}$  of  $SOM_G$ : the update direction of  $SOM_G$  is decided by the positional relation with  $SOM_L$ .
3. Learning function:  $SOM_L$  are updated according to the learning function which is the same function used in PSOM algorithm.

These key roles make the behavior of TSOM interesting. Namely,  $SOM_L$  and  $SOM_L$  which are located close to each

Haruna Matsushita and Yoshifumi Nishio are with the Department of Electrical and Electronic Engineering, Tokushima University, Tokushima, 770-8506, Japan (phone: +81-88-656-7470; fax: +81-88-656-7471; email: {haruna, nishio}@ee.tokushima-u.ac.jp).

other accommodate each other, contrary,  $SOM_G$  and  $SOM_L$  which are located close to each other compete against with each other.  $SOM_L$  and  $SOM_L$ , which are located far apart, also compete against with each other.

In Section II, we explain the key points and the learning algorithm of the proposed TSOM algorithm in detail. The competing and the accommodating behaviors of TSOM are investigated in the Section III with applications to data extraction from input data including a lot of noises. For 2 and 3-dimensional input data, extraction ability of clusters is evaluated both visually and quantitatively using a correct answer rate. We also apply TSOM to 5-dimensional input data and confirm the extraction ability for higher-dimensional data. We can see that TSOM successfully extracts clusters even in the case that we do not know the number of clusters in advance.

## II. TENTACLED SELF-ORGANIZING MAP (TSOM)

In this study, we propose a method of using plural SOMs (TSOM: Tentacled SOM) for effective data extraction. The plural SOMs of TSOM contains the two kinds of SOMs, namely  $SOM_G$  and  $SOM_L$ . Each SOM of TSOM exchanges their own position information each other and self-organizes with the competing and accommodating behaviors.

In the TSOM algorithm, we use totally  $n$  SOMs, that is one  $SOM_G$  and  $(n - 1)$   $SOM_L$ ; namely  $SOM_{L1}, SOM_{L2}, \dots, SOM_{L(n-1)}$ . Each SOM consists of  $m$  neurons located at regular 2-dimensional grid. The lattice of the grid is either hexagonal or rectangle. Each neuron  $i$  of  $SOM_G$  has a  $d$ -dimensional weight vector  $w_{Gi} = (w_{Gi1}, w_{Gi2}, \dots, w_{Gid})$ , and each neuron  $i$  of  $SOM_{Ll}$  ( $l = 1, 2, \dots, n - 1$ ) also has the weight vector  $w_{Lli} = (w_{Lli1}, w_{Lli2}, \dots, w_{Llid})$ . The input data is denoted by  $x_j = (x_{j1}, x_{j2}, \dots, x_{jd})$  ( $j = 1, 2, \dots, N$ ).

TSOM has three key rules, namely, “Winner SOM”, “The update direction of  $SOM_G$ ” and “Learning function”. We explain these key points with the learning algorithm in this section.

### A. WINNER SOM DECIDING RULE

We use plural SOMs for TSOM algorithm and find winner neurons in every SOM for one input data by calculating the distance between  $x_j$  and all the weight vector. In other words, each winner neuron is the neuron with the weight vector closest to  $x_j$  in each SOM, namely, all SOMs of TSOM have one winner neuron, respectively. The winner neuron of  $SOM_G$  is denoted by  $c_G$ , and the winner neuron of  $SOM_{Ll}$  is denoted by  $c_{Ll}$ . However, neurons of all SOMs are NOT updated, the weight vectors updated are that of only the winner SOM.

We explain the method for deciding the winner SOM. This plays a very important role in the TSOM algorithm and produces the competing behavior and the accommodating behavior of TSOM. The winner SOM is determined by the position of each winner neuron of each SOM.

### [For $SOM_L$ ]

(1) The winner SOM is  $SOM_{Ll}$  if the winner neuron  $c_{Ll}$  of  $SOM_{Ll}$  has the weight vector closest to  $x_j$  in all the winner neurons (as Fig. 1(a)).

(2) Plural  $SOM_L$  can be the winner SOMs if the distances between  $x_j$  and winner neurons of plural  $SOM_L$  are less than  $R_L(t)$ , namely  $\|w_{Llc_{Ll}} - x_j\| \leq R_L(t)$  (as Fig. 1(b)).

### [For $SOM_G$ ]

(3) The winner SOM is  $SOM_G$  if the winner neuron  $c_G$  of  $SOM_G$  has the weight vector closest to  $x_j$  in all the winner neurons, and all winner neurons of  $SOM_L$  are more than  $R_L(t)$  away from  $x_j$  (as Fig. 1(c)).

(4)  $SOM_G$  can not be the winner if distances between  $x_j$  and winner neurons of  $SOM_L$  are not more than  $R_L(t)$  although the winner neuron  $c_G$  of  $SOM_G$  has the weight vector closest to  $x_j$  in all the winner neurons (as Fig. 1(d)).

In the case (4), the weight vectors  $w_G$  of  $SOM_G$  are updated to minus direction. The value of deciding the update direction of  $SOM_G$  for the input data  $x_j$  can be written as;

$$y_{Gj} = \begin{cases} 1, & \text{if case (3)} \\ -1, & \text{if case (4)} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

In other words,  $SOM_G$  and  $SOM_L$ , which are located close to each other, compete against each other.

$R_L(t)$  plays an tentacle of each SOM and increases with time according to;

$$R_L(t) = R_{Lm} \left\{ 1 - \left( \frac{0.005}{R_{Lm}} \right)^{t/T} \right\}, \quad (2)$$

where  $R_{Lm}$  is a maximum value of  $R_L(t)$ , and  $T$  is the maximum number of the learning.

### B. LEARNING ALGORITHM

The range of the elements of the input data  $x_j$  are assumed to be from 0 to 1. The initial values of all the weight vector of  $SOM_G$  and  $SOM_L$  are given between 0 and 1 at random.

(TSOM1) An input data  $x_j$  is inputted to all the neurons of  $SOM_G$  and  $SOM_L$  at the same time in parallel.

(TSOM2) Distances between  $x_j$  and all the weight vectors of all SOM are calculated. The winner neuron of each SOM is the neuron with the weight vector closest to  $x_j$  in each SOM;

$$\begin{aligned} c_G &= \arg \min_i \{ \|w_{Gi} - x_j\| \}, \\ c_{Ll} &= \arg \min_i \{ \|w_{Lli} - x_j\| \}, \end{aligned} \quad (3)$$

where  $\| \cdot \|$  is the distance measure, in this study, Euclidian distance.

(TSOM3) One or more winner SOMs are determined by the distance between  $x_j$  and the winner neurons, according to II. A.

(TSOM4) The weight vectors of neurons of  $SOM_G$  and  $SOM_L$  are updated.

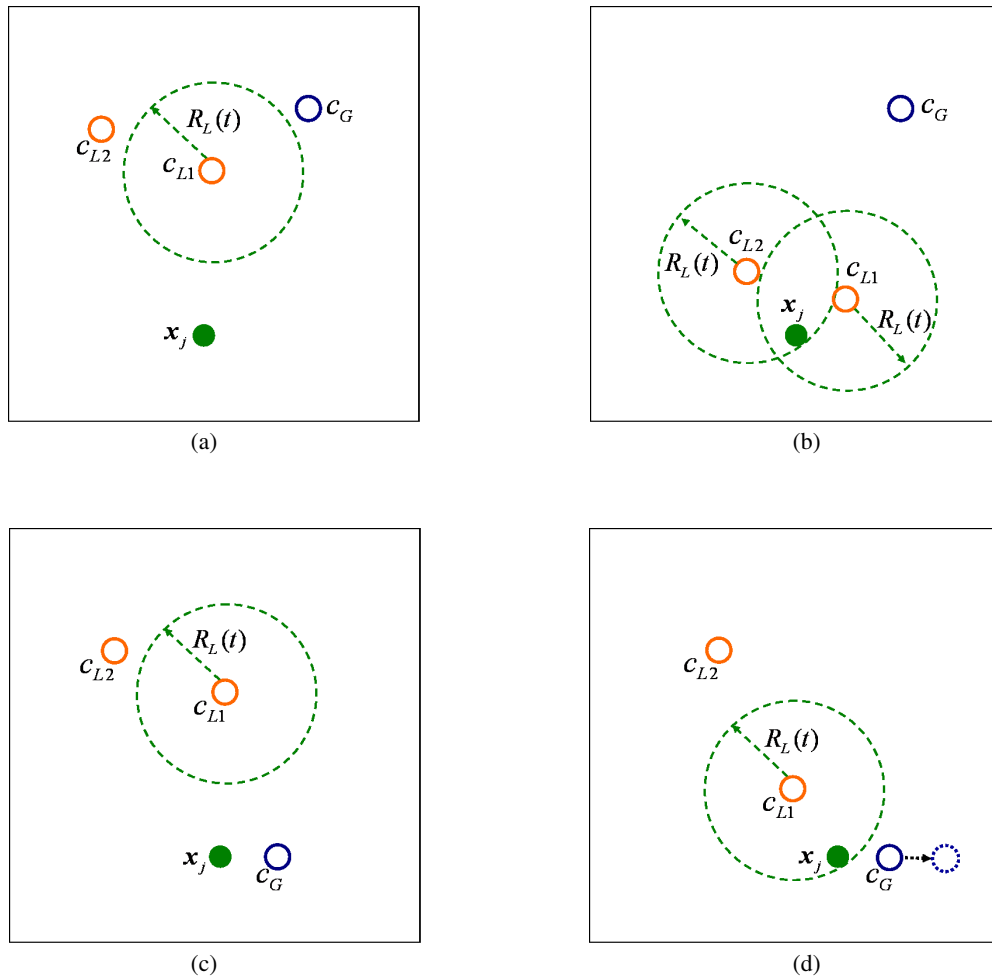


Fig. 1. Example of winner SOM and the location of winner neurons. (a) Winner SOM is  $SOM_{L1}$ . (b) Winner SOMs are  $SOM_{L1}$  and  $SOM_{L2}$ . (c) Winner SOM is  $SOM_G$ . (d) Winner SOM is  $SOM_{L1}$ .

The weight vectors of neurons in  $SOM_G$  are updated according to the following equation;

$$\mathbf{w}_{G_i}(t+1) = \mathbf{w}_{G_i}(t) + y_{G_j} \cdot h_{G_{c_G},i}(t)(\mathbf{x}_j - \mathbf{w}_{G_i}(t)), \quad (4)$$

where  $y_{G_j}$  is the value of deciding the update direction of  $SOM_G$  for  $\mathbf{x}_j$  determined by Eq. (1), and  $h_{G_{c_G},i}(t)$  is the neighborhood function of  $SOM_G$  described as follows;

$$h_{G_{c_G},i}(t) = \alpha(t) \exp\left(-\frac{\|\mathbf{r}_i - \mathbf{r}_{c_G}\|^2}{2\sigma^2(t)}\right), \quad (5)$$

where  $\mathbf{r}_i$  is the vectorial locations on the display grid of neuron  $i$ ,  $\alpha(t)$  is the learning rate, and  $\sigma(t)$  corresponds to the widths of the neighborhood function. Both  $\alpha(t)$  and  $\sigma(t)$  decrease monotonically with time according to the following equation;

$$\alpha(t) = \alpha(0)(1 - t/T), \quad \sigma(t) = \sigma(0)(1 - t/T), \quad (6)$$

where  $T$  is the maximum number of the learning.

If the winner SOM is  $SOM_{Ll}$ , the weight vectors of neurons in  $SOM_{Ll}$  are updated, according to;

$$\mathbf{w}_{Ll_i}(t+1) = \mathbf{w}_{Ll_i}(t) + h_{Ll_{c_{Ll}},i}(t)(\mathbf{x}_j - \mathbf{w}_{Ll_i}(t)), \quad (7)$$

The function  $h_{Ll_{c_{Ll}},i}(t)$  is the neighborhood function of  $SOM_{Ll}$  and it is described as follows;

$$h_{Ll_{c_{Ll}},i}(t) = p_{c_{Ll}}(t) \exp\left(-\frac{\|\mathbf{r}_i - \mathbf{r}_{c_{Ll}}\|^2}{2\sigma^2(t)}\right), \quad (8)$$

where  $p_{c_{Ll}}(t)$  is the learning function, and it is explained in the next subsection.

**(TSOM5)** The steps from (TSOM1) to (TSOM4) are repeated for all the input data, namely from  $j = 1$  to  $j = N$ .

### C. LEARNING FUNCTION

In our past study, we have proposed the learning function. The learning function produces the competing behavior and the accommodating behavior of  $SOM_{Ll}$ . The value of the learning function is determined by the distance between the input vector  $\mathbf{x}_j$  and the winner neuron  $c_{Ll}$  of  $SOM_{Ll}$ , according to;

$$p_{c_{Ll}}(t) = \alpha(t) \exp\left(-\frac{\|\mathbf{w}_{Ll_{c_{Ll}}} - \mathbf{x}_j\|^2}{2\sigma_P^2}\right), \quad (9)$$

where  $\mathbf{w}_{Ll_{c_{Ll}}}$  is the weight vector of the winner neuron  $c_{Ll}$  of  $SOM_{Ll}$ ,  $\sigma_P$  corresponds to the width of the learning

function, and  $\alpha(t)$  corresponds to the maximum value of the learning function according to Eq. (6).

It is desirable to choose  $\sigma_P$  as a small value such as less than 0.1, because this is a key point to decide their competing behavior and accommodating behavior. Namely, when plural SOM<sub>L</sub> are winner SOM, some winner SOM<sub>L</sub> whose winner neurons are near the input data are significantly updated and this means that these SOM<sub>L</sub> tend to accommodate the input data each other. Conversely, some winner SOM<sub>L</sub> whose winner neurons are far apart from the input data are updated very little and this means that these SOM<sub>L</sub> keep to compete with other SOM<sub>L</sub>.

In other words, some SOM<sub>L</sub>, which are located close to each other, accommodate each other, contrary, some SOM<sub>L</sub>, which are located far apart, compete against each other.

### III. APPLICATION TO DATA EXTRACTION

#### A. 2-dimensional input data

First, we consider 2-dimensional input data as shown in Fig. 2(a). The input data is generated artificially as follows. Total number of the input data  $N$  is 1600. 25% of the input data are distributed within a range from 0.2 to 0.3 horizontally and from 0.7 to 0.8 vertically, and these data are called the cluster  $C_1$ . 50% of the input data are distributed in another cluster  $C_2$ , whose horizontal-values follow the normal distribution  $N(0.7, 0.04)$ , and the vertical-values  $N(0.2, 0.0016)$ . The remaining 25% of the input data are distributed between 0 and 1 at random.

Because the input data include 2 clusters, we use one SOM<sub>G</sub> and two SOM<sub>L</sub> ( $n = 3$ ). Each SOM has 100 neurons ( $10 \times 10$ ), namely 3 SOMs have totally 300 neurons, and these neurons are arranged on a 2-dimensional grid. We repeat the learning four times for all input data. The parameters of the learning are chosen as follows;

$$\alpha(0) = 0.5, \sigma(0) = 3, \sigma_P = 1/16, R_{L_m} = 0.2.$$

The simulation result is shown in Fig. 2(b). We can see that two SOM<sub>L</sub> stay around the two clusters by the competing behavior of TSOM method.

In order to extract only clusters of the input data including a lot of noises, we calculate the distance between the input data  $x_j$  and the weight vectors in SOM<sub>L<sub>l</sub></sub> after learning of TSOM. If the calculated distance is smaller than a threshold value  $R$ , the input data  $x_j$  is classified into the cluster corresponding to SOM<sub>L<sub>l</sub></sub>. Figures 2(c) and (d) show the input data classified into the clusters corresponding to SOM<sub>L<sub>1</sub></sub> and SOM<sub>L<sub>2</sub></sub>, respectively ( $R = 0.05$ ). As we can see from the figures, SOM<sub>L</sub> can successfully extract the clusters, and the noises are removed by SOM<sub>G</sub>. This is the result of TSOM when we know the number of the clusters in advance.

Although, the  $k$ -means method is known to be not useful for the data including a lot of noises, we carry out the  $k$ -means method for the same input data for the comparison. Figure 3 shows the results obtained by using the  $k$ -means method, where the number of the cluster is set as  $k = 3$

(namely, two clusters and noises). We can see that the clusters obtained by the  $k$ -means method include a lot of noises.

Next, we assume that we do not know the appropriate number of clusters in advance. We simulate for the same input data using one SOM<sub>G</sub> and three SOM<sub>L</sub>, namely the number of SOM<sub>L</sub> is more than the number of clusters. We use the same parameters as Fig. 2. Figures 4(a)-(h) show the learning process of TSOM, and the simulation result is Fig. 4(h). We can see that all SOM<sub>L</sub> stay around the two clusters and SOM<sub>G</sub> self-organizes only noises. We observe this in more detail, the two SOM<sub>L</sub>, which stay around the same cluster  $C_2$ , are accommodating in one cluster without competing although one SOM<sub>L</sub> is redundant. On the other hand, SOM<sub>L</sub> situated on the other cluster  $C_1$  is competing with other two SOM<sub>L</sub> without accommodating.

Let us consider the learning process. In the early stage of learning as Figs. 4(a) and (b), SOM<sub>G</sub> is self-organizing the cluster  $C_1$ . However, SOM<sub>L</sub> edges out SOM<sub>G</sub> and is self-organizing this cluster as Fig. 4(d), because SOM<sub>G</sub> are updated to the minus direction, according to Eq. (1). In the meanwhile, two SOM<sub>L</sub> are competing in the same cluster  $C_2$  as Figs. 4(a)-(d), however, these two SOM<sub>L</sub> are accommodating in one cluster as the learning progressed. This is because SOM<sub>L</sub> and SOM<sub>L</sub>, which are located close to each other, accommodate each other, according to Eq. (8), namely, some SOM<sub>L</sub> whose winner neurons are close to the input data are significantly updated.

We carry out the extraction of clusters from Fig. 4(h). Figures 5(a)-(c) show that the input data are classified into the clusters corresponding to SOM<sub>L<sub>1</sub></sub>, SOM<sub>L<sub>2</sub></sub> and SOM<sub>L<sub>3</sub></sub>, respectively ( $R = 0.05$ ).

In order to investigate the ability of the TSOM algorithm, we carry out the simulations with excessive number of SOMs. The parameters are the same as Fig. 2. Figure 6 shows the result using six SOMs for the same input data, namely 5 SOM<sub>L</sub> just for 2 clusters. As we can see from the figures, TSOM can extract 2 clusters as if they know the number of the clusters.

In order to evaluate the clustering ability of TSOM quantitatively, we define the correct answer rate  $R_{CI}$  as follows;

$$R_{CI} = \frac{N_{rI} - N_{eI}}{N_{CI}}, \quad (I = 1, 2, \dots), \quad (10)$$

where  $N_{CI}$  is the true number of the input data within the cluster  $C_I$ ,  $N_{rI}$  is the obtained number of the desired input data within  $C_I$ , and  $N_{eI}$  is the obtained number of undesired input data out of  $C_I$ . The calculated results are summarized in Table I. When the number of SOM<sub>L</sub> is equal to the number of the clusters ( $n=3$ ), we can evaluate the effectiveness of the method of using TSOM method by this index value.

#### B. 3-dimensional input data

Next, we carry out simulation for 3-dimensional input data shown in Fig. 7(a). The input data include two clusters and a lot of noises out of the clusters.

Figure 7(b) shows the learning result of TSOM ( $n = 3$ ), and the same result from another angle is shown in Fig. 7(c).

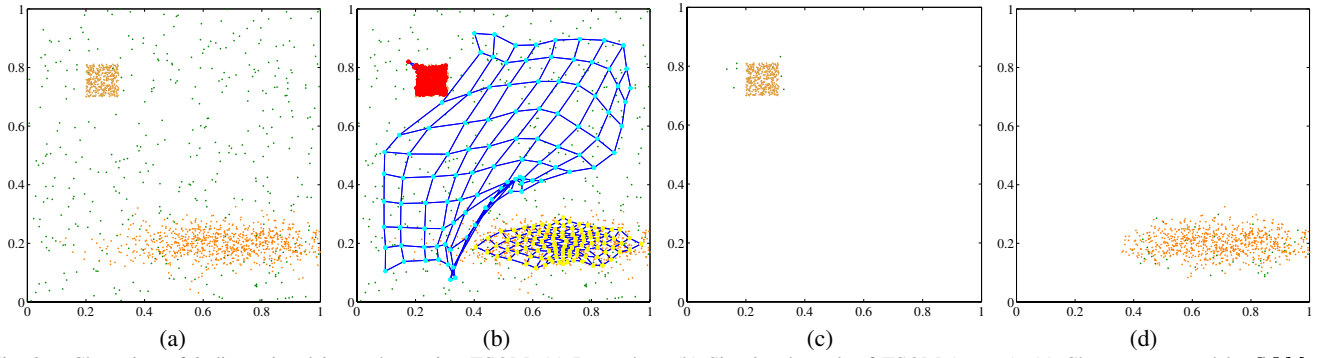


Fig. 2. Clustering of 2-dimensional input data using TSOM. (a) Input data. (b) Simulated result of TSOM ( $n = 3$ ). (c) Clusters extracted by  $SOM_{L1}$ . (d) Clusters extracted by  $SOM_{L2}$ .

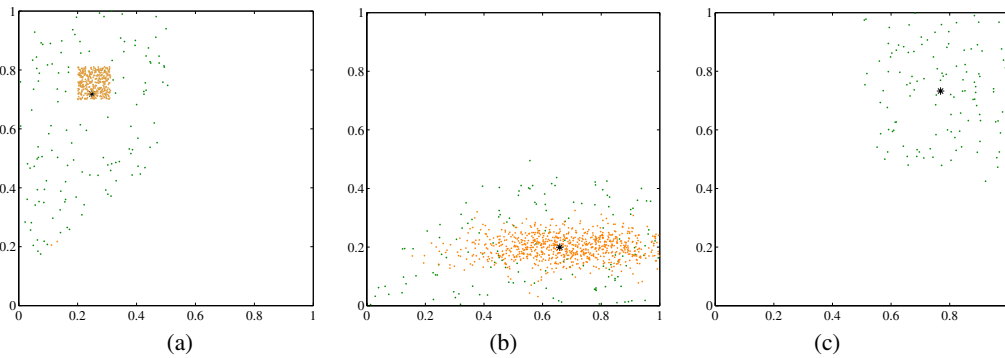


Fig. 3. Extraction of clusters by  $k$ -means method ( $k = 3$ ). (a) Cluster1. (b) Cluster2. (c) Noises.

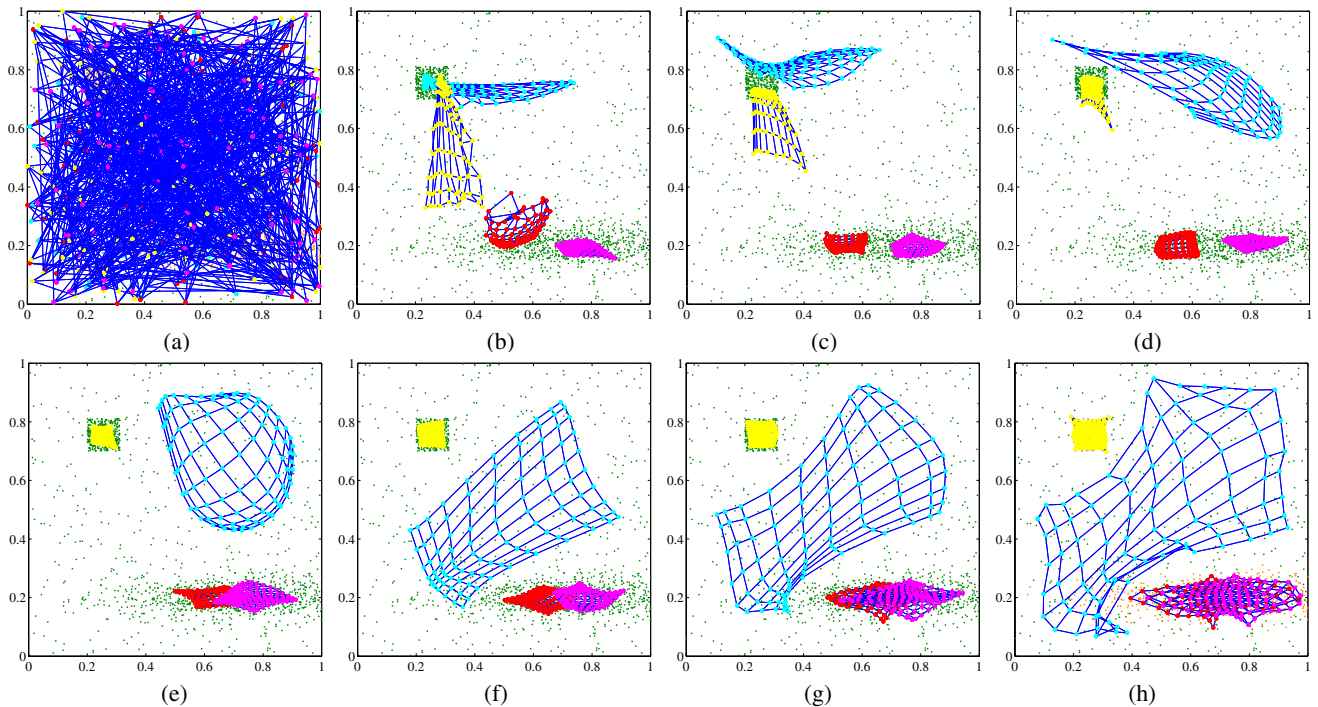


Fig. 4. Clustering using TSOM with inappropriate number of  $SOM_L$ . (a) Initial state ( $t = 0$ ). (b)  $t = 200$ . (c)  $t = 400$ . (d)  $t = 600$ . (e)  $t = 1000$ . (f)  $t = 2000$ . (g)  $t = 3000$ . (h) Simulation results ( $t = 6400$ ).

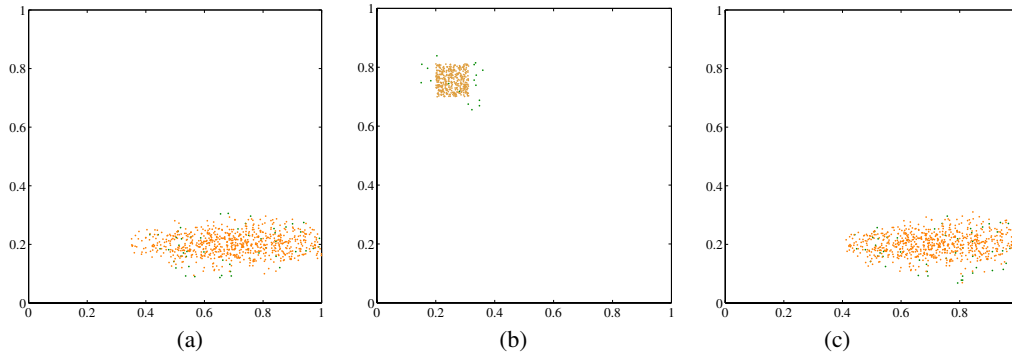


Fig. 5. Input data extracted by  $SOM_L$  of TSOM. (a)  $SOM_{L1}$  (b)  $SOM_{L2}$ . (c)  $SOM_{L3}$ .

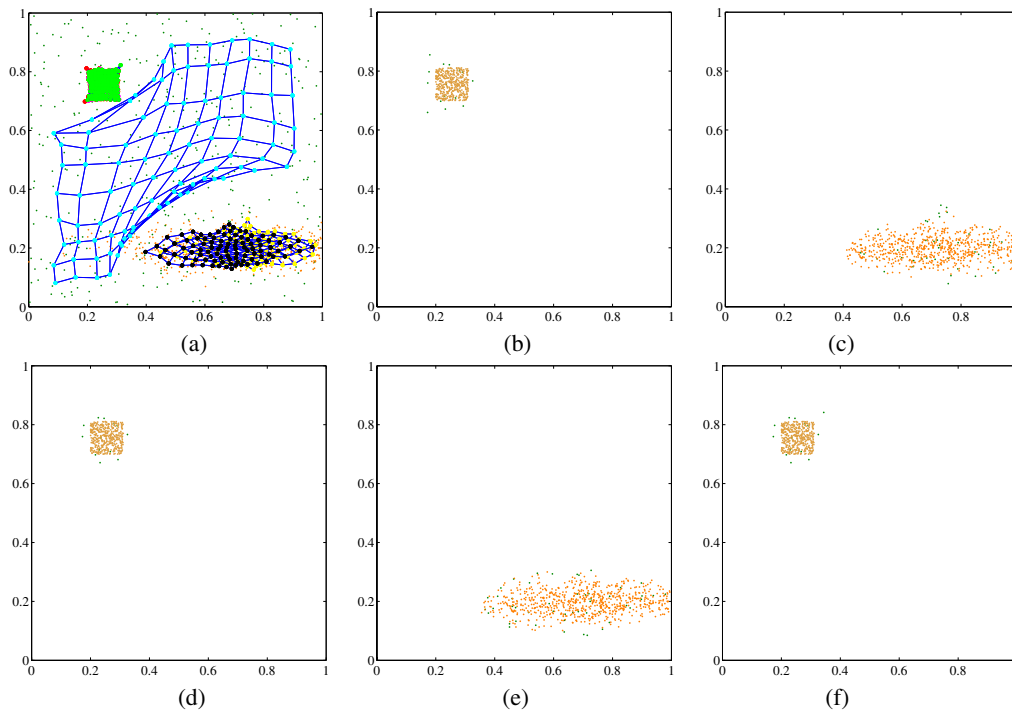


Fig. 6. Cluster extraction using excessive number of SOMs. (a) Learning result of TSOM ( $n = 6$ ). (b)  $SOM_{L1}$  (c)  $SOM_{L2}$ . (d)  $SOM_{L3}$ . (e)  $SOM_{L4}$ . (f)  $SOM_{L5}$ .

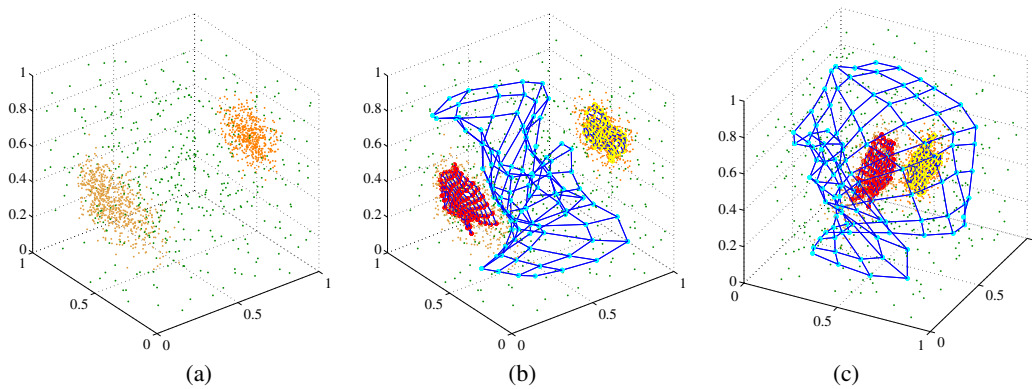


Fig. 7. Clustering of 3-dimensional input data using TSOM. (a) Input data. (b) Learning result of TSOM ( $n = 3$ ) (c) Result from another angle.

TABLE III  
5-DIMENSIONAL GAUSSIAN DATA.

No.		Coordinate axes					Probability [%]
		$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	
$C_1$	Mean value	0.8	0.3	0.7	0.2	0.7	15
	Variance	0.0016	0.0081	0.0036	0.0009	0.0081	
$C_2$	Mean value	0.35	0.8	0.3	0.7	0.2	20
	Variance	0.0036	0.0004	0.01	0.0081	0.0036	
$C_3$	Mean value	0.6	0.9	0.1	0.1	0.9	15
	Variance	0.0004	0.0016	0.0009	0.0025	0.0009	
$C_4$	Mean value	0.2	0.1	0.8	0.4	0.3	20
	Variance	0.0016	0.01	0.0004	0.0225	0.0025	

TABLE I  
CORRECT ANSWER RATE [%] FOR 2-DIMENSIONAL INPUT DATA.

$n$	Method	$SOM_{Ll}$				
		1	2	3	4	5
3	$k$ -means	81.78	66.00	-	-	-
	TSOM	87.36	97.73	-	-	-
4	TSOM	85.48	95.62	83.08	-	-
6	TSOM	96.50	84.01	96.97	86.09	96.74

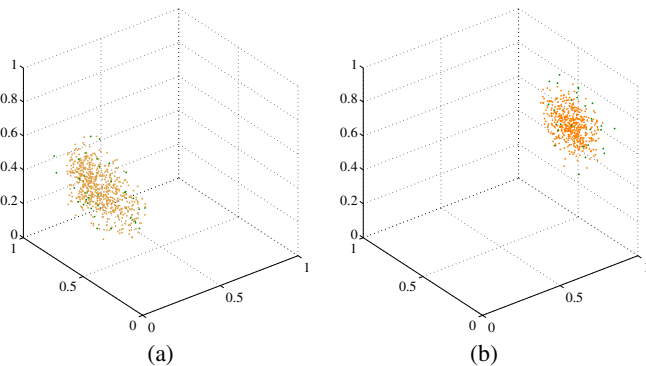


Fig. 8. 3-D input data extracted by  $SOM_L$ . (a)  $SOM_{L1}$ . (b)  $SOM_{L2}$ .

We can see that two  $SOM_L$  stay around the two clusters and  $SOM_G$  self-organizes the area of noises. The input data classified into the clusters corresponding to each  $SOM_L$  are shown in Figs. 8(a) and (b), respectively ( $R = 0.15$ ). We can confirm that the noises are removed by  $SOM_G$  and only the cluster part can be extracted very well.

Furthermore, we carry out this simulation for the same input data using TSOM ( $n = 5$ ) and  $k$ -means method ( $k = 3$ ). The correct answer rates are summarized in Table II. As we can see from this table, the clustering ability of using TSOM method is effectual whether or not we know the number of clusters.

TABLE II  
CORRECT ANSWER RATE [%] FOR 3-DIMENSIONAL INPUT DATA.

$n$	Method	$SOM_{Ll}$			
		1	2	3	4
3	$k$ -means	65.98	59.29	-	-
	TSOM	89.75	94.26	-	-
5	TSOM	93.85	87.67	90.49	94.47

### C. 5-dimensional input data

Furthermore, we perform the simulation for 5-dimensional input data of 1600 points. This data include four clusters and a lot of noises, and the four clusters are generated by random Gaussian data as shown in Table III.

We carry out simulations for the cases of  $n = 5$  (which is the same number of clusters) and  $n = 7$  (we do not know the number of cluster). The parameters of the learning for  $n = 5$  and  $n = 7$  are chosen as follows;

$$n = 5 \Rightarrow \alpha(0) = 0.7, \sigma(0) = 4, \sigma_P = 1/16, R_{Lm} = 0.2,$$

$$n = 7 \Rightarrow \alpha(0) = 0.7, \sigma(0) = 4, \sigma_P = 1/16, R_{Lm} = 0.25.$$

The results of the calculated correct answer rates ( $R = 0.3$ ) are summarized in Table. IV.

TABLE IV  
CORRECT ANSWER RATE [%] FOR 5-DIMENSIONAL INPUT DATA.

$n$	Method	$SOM_{Ll}$					
		1	2	3	4	5	6
5	TSOM	80.00	97.93	96.35	93.69	-	-
7	TSOM	96.77	95.15	63.52	95.79	70.39	98.37

## IV. CONCLUSIONS

In this study, we have proposed the Tentacled SOM (TSOM) algorithm which is a method of using plural SOMs. TSOM possesses both abilities of  $n$ SOM and PSOM. Each SOM of TSOM can catch the information of other SOMs existing in its neighborhood and self-organizes with the competing and accommodating behaviors. We have investigated its behaviors. Furthermore, we have applied TSOM to extract clusters in the case that we do not know the number of the clusters in advance, and have confirmed the efficiency.

## REFERENCES

- [1] T. Kohonen, *Self-organizing Maps*, Berlin, Springer, vol. 30, 1995.
- [2] Y. Cheng, "Clustering with Competing Self-Organizing Maps," *Proc. of International Joint Conference on Neural Networks*, vol. IV, 1992, pp. 785-790.
- [3] W. Wan and D. Fraser, "M2dSOMAP: Clustering and Classification of Remotely Sensed Imagery by Combining Multiple Kohonen Self-Organizing Maps and Associative Memory," *Proc. of International Joint Conference on Neural Networks*, vol. III, 1993, pp. 2464-2467.

- [4] J. Vesanto and E. Alhoniemi, "Clustering of the Self-Organizing Map," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, 2002, pp. 586-600.
- [5] I. Lapidot, H. Guterman and A. Cohen, "Unsupervised Speaker Recognition Based on Competition Between Self-Organizing Maps," *IEEE Transactions on Neural Networks*, 2002, vol. 13, no. 4, pp. 877-887.
- [6] H. Matsushita and Y. Nishio, "Competing Behavior of Two Kinds of SOMs and its Application to Clustering," *Proc. of Workshop on Self-Organizing Maps*, 2005, pp. 355-362.
- [7] H. Matsushita and Y. Nishio, "Competing and Accommodating Behaviors of Peace SOM," *Proc. of IEEE International Symposium on Circuits and Systems*, 2006.