Applications of Brain-Inspired SOR Network to Controller Design and Knowledge Acquisition

Takanori Koga, Keiichi Horio and Takeshi Yamakawa
Kyushu Institute of Technology
Graduate School of Life Science and Systems Engineering
8080196, 2-4 Hibikino, Kitakyushu. Japan
koga-takanori@edu.brain.kyutech.ac.jp
{horio, yamakawa}@brain.kyutech.ac.jp

Abstract - *In this paper, we propose the SOR network with fuzzy inference based evaluation inspired by brain function. In the proposed method, controller design and knowledge acquisition are achieved simultaneously. All the designer has to do is to describe evaluation rules for the input/output data set sampled by trial and error. In the description process, only designer's commonsense knowledge is required. SOR network extracts practical knowledge from the data set with evaluation, and works as a fuzzy controller after the learning.*

Key words - SOR network, controller design, knowledge acquisition

1 Introduction

The original intent of Self-Organizing Relationship (SOR) network[1] development is to advance the Self-Organizing Map (SOM)[2] from the viewpoint of brain-inspired algorithm. SOR network realizes an approximation of a desired Input/Output (I/O) relationship of a target system by unsupervised learning. SOR network is effective in the case that a set of learning data (desired I/O relationship) is unavailable but the I/O data obtained by trial and error can be evaluated. The criterion for evaluation becomes the key whether the desired I/O relationship will be successfully approximated or not. So far, mathematical expressions and a designer's intuition are employed as objective (quantitative)[1] and subjective (qualitative) evaluations[3], respectively. Furthermore, we have introduced the evaluation method with fuzzy inference and applied it to the controller design[4]. In the method, only experts' commonsense knowledge is required. The method is inspired that human beings evaluate the I/O relationship linguistically as opposed to mathematically when the target system is complex. In this paper, we propose the application of the SOR network with fuzzy inference based evaluation as a knowledge acquisition tool. In learning process, the SOR network extracts practical knowledge on the two-dimensional knowledge layer. Each if-then unit on the knowledge layer stores a crisp if-then rule. By analyzing the knowledge layer (for example, watching U-matrix[5]) the designer can acquire valuable knowledge. Also, SOR network generates the desirable outputs to the actual inputs as a fuzzy controller. In these processes, two fuzzy inference engines work. The one calculates evaluation values as an evaluator and the other does desirable outputs as a controller. The effectiveness of the proposed method is verified

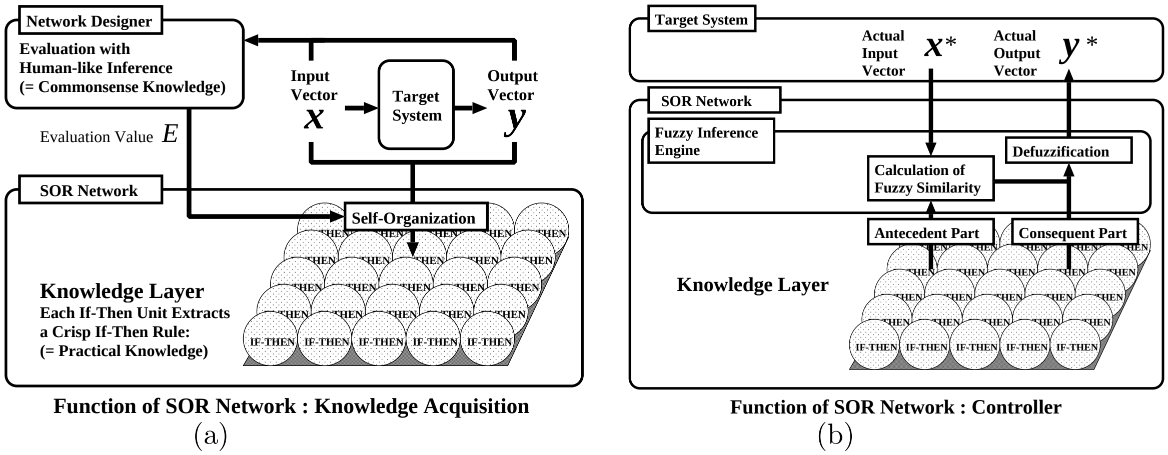


Figure 1: The structure of the SOR network.(a)The learning mode, (b)The execution mode.

through a trailer-truck back-up controller design.

2 Self-Organizing Relationship (SOR) Network

The SOR network consists of a knowledge layer with N if-then units. The operation of SOR network is divided into two modes; the learning mode (Fig.1(a)) and the execution mode (Fig.1 (b)). In the learning mode, the learning of SOR network is achieved in a similar fashion as SOM[2] by the stable batch learning algorithm[6]. The input vectors and the output vectors have n and m elements, respectively. The j-th if-then unit on the knowledge layer has reference vectors in the input space $\boldsymbol{w}_j = [w_{j1}, \cdots, w_{jn}]$ and in the output space $\boldsymbol{u}_j = [u_{j1}, \cdots, u_{jm}]$. After the learning, the practical knowledge of the target system is stored in each if-then unit on the knowledge layer. In the execution mode, the SOR network generates actual outputs by working as a fuzzy inference engine.

2.1 Learning Mode of SOR Network

In the learning, the random I/O vector pair $\boldsymbol{I}_l = [\boldsymbol{x}_l, \boldsymbol{y}_l] = [x_{l1}, \ldots, x_{ln}, y_{l1}, \ldots, y_{lm}]$ is applied, as the learning vector, to the input and output layers together with the evaluation E_l for the I/O vector pair as shown in Fig.1(a). The evaluation E_l may be assigned by the network designer, given by the intuition of the user or obtained by examining the system under test. The value of E_l is positive and negative in accordance with judgment of the designer, preference of the user or score of an examination. The positive E_l causes the self-organization of attraction to the learning vector and the negative one does that of repulsion from the learning vector. The batch learning algorithm of SOR network is summarized as follows.

Step 0 Let $\boldsymbol{v}_j$ be the *Cartesian product* of $\boldsymbol{w}_j$ and $\boldsymbol{u}_j$. All reference vectors $\boldsymbol{v}_j (j = 1, \cdots, N)$ are initialized by random numbers.

Step 1 A winner unit $winner_l$ for the learning vector I_l is selected by the smallest *Euclidean Distance* by:

$$winner_l = \arg\min_j \parallel \boldsymbol{I}_l - \boldsymbol{v}_j \parallel . \tag{1}$$

Step 2 A coefficient of neighboring effect $\xi_{j,l}$ and a coefficient of the repulsive learning ζ_j are calculated as follows:

$$\xi_{j,l} = \begin{cases} E_l h(j, winner_l) & for \quad E_l \geq 0 \\ E_l \frac{\alpha}{\eta_r^2} \exp(-\frac{\|\boldsymbol{v}_{winner_l} - \boldsymbol{I}_l\|^2}{2\eta_r^2}) h(j, winner_l) & for \quad E_l < 0, \end{cases} \tag{2}$$

$$\zeta_j = 2 \sum_{l \in V_j} \mid \xi_{j,l} \mid, \tag{3}$$

$$h(j, winner_l) = \exp(-\frac{\parallel r_j - r_{winner_l} \parallel^2}{2\eta(t)^2}), \tag{4}$$

where, r_j, r_{winner_l} are coordinates of the j-th and the winner unit on the knowledge layer, respectively. $\eta(t)$ is a width of neighboring function at the learning step t. V_j is defined as a set of learning vectors with a negative evaluation value in the voronoi region of the reference vector $\boldsymbol{v}_j$.

Step 3 After all learning vectors are applied, the reference vectors are updated as follows:

$$\boldsymbol{v}_j(t+1) = (1-\epsilon)\boldsymbol{v}_j(t) + \epsilon \frac{\sum_{l=1}^{L} \xi_{j,l} \boldsymbol{I}_l + \sum_{j=1}^{N} \zeta_j \boldsymbol{v}_j(t)}{\sum_{l=1}^{L} \mid \xi_{j,l} \mid}, \tag{5}$$

where, ϵ is a learning rate.

Step 4 Steps 1 to 3 are repeated decreasing the width of neighboring function $\sigma(t)$

2.2 Execution Mode of SOR Network

After the learning, SOR network is ready to use as the I/O relationship generator as shown in Fig.1(b). The w_{ji} is the i-th element of the j-th reference vector in the input space characterizing the j-th if-then unit. The u_{jk} is the k-th element of the j-th reference vector in the output space characterizing the j-th if-then unit. The actual input vector $\boldsymbol{x}^*$ is applied to SOR network. The similarity measure z_j between $\boldsymbol{x}^*$ and the reference vector $\boldsymbol{w}_j$ of j-th if-then unit in the knowledge layer is defined by:

$$z_j = \exp(-\frac{\|\boldsymbol{x}^* - \boldsymbol{w}_j\|^2}{2\gamma_j^2}), \tag{6}$$

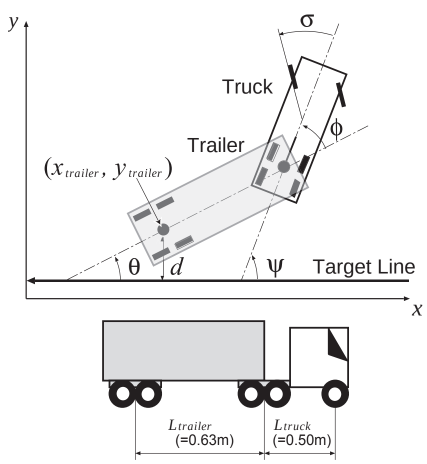
where, γ_j is a parameter representing fuzziness of similarity. The k-th element y_k^* of the output vector $\boldsymbol{y}^*$is calculated by:

$$y_k^* = \frac{\sum_{j=1}^{N} z_j u_{jk}}{\sum_{j=1}^{N} z_j}. \tag{7}$$

The output of the network $\boldsymbol{y}^* = [y_1^*, \ldots, y_m^*]$ represents the weighted average of $\boldsymbol{u}_j$ by the similarity measure z_j. The execution process of SOR network corresponds to a fuzzy inference with N if-then rules.*Appl*

3 Controller Design Employing SOR Network

3.1 Trailer-Truck Back-Up Control



$L_{trailer}$	Length of the trailer (=0.63m)
L_{truck}	Length of the truck (=0.50m)
v	Velocity of the truck (= -0.1m/s)
σ	Front wheel angle of the truck -30 ≤ σ ≤ 30 [deg]
ψ	Angle of the truck -180 ≤ ψ ≤ 180 [deg]
θ	Angle of the trailer -180 ≤ θ ≤ 180 [deg]
ϕ	Connection angle of trailer-truck -45 ≤ ϕ ≤ 45 [deg]
$x_{trailer}$	X- coordinate of trailer
$y_{trailer}$	Y- coordinate of trailer
d	Position error against target line

Figure 2: The model of the trailer-truck and its specification used in this paper.

In this study, a semi-trailer type trailer-truck back-up control is achieved. The model of the semi-trailer type trailer-truck used in this study is shown in Fig.2. The trailer-truck back-up control is a severe nonlinear control. In addition, the trailer-truck easily meets unstable states. This is because the connection angle between the trailer and the truck easily becomes large. Once the connection angle is excessively enlarged, the trailer-truck can not be controlled any more by only backward movement. This is called as "jackknife state". In this study, the control objective is to make the trailer-truck follow the straight target line from any initial state by only backward movement at constant velocity. This problem is regarded as design of a 3-input 1-output regulator which does not require strict optimality. The SOR network, which works as the trailer-truck controller, is established by the learning with learning vectors. The learning vectors are obtained from the kinematic model of the trailer-truck. The kinematic model of the trailer-truck is described by:

$$\psi[k+1] = \psi[k] + \frac{v\Delta t \tan\sigma[k]}{L_{truck}}, \tag{8}$$

$$\theta[k+1] = \theta[k] + \frac{v\Delta t \sin\phi[k]}{L_{trailer}}, \tag{9}$$

$$\phi[k] = \psi[k] - \theta[k], \tag{10}$$

$$x_{trailer}[k+1] = x[k] + v\Delta t \cos\phi[k] \cos\frac{\theta[k+1]+\theta[k]}{2}, \tag{11}$$

$$y_{trailer}[k+1] = y[k] + v\Delta t \cos\phi[k] \sin\frac{\theta[k+1]+\theta[k]}{2}, \tag{12}$$

where, k represents a time sampling, and Δt a sampling interval (Δt = 1.0[sec]). In the proposed system, the angle between the trailer and the truck ϕ, the angle of the trailer θ, and the distance between the trailer-truck and the target line d are the inputs to the SOR network, and the SOR network generates the front wheel angle σ. The procedure of the learning vector acquisition is as mentioned below. At first, the state of the trailer-truck at k $(\phi(k), \theta(k), d(k))$ is randomly given. Then, the front wheel angle $(\sigma(k))$ is randomly given as an operation at time k. These values are elements of the learning vector. As a result of operation, the state at $k+1$ is calculated using the kinematics model. The designer should evaluate the I/O relationship of the controller with fuzzy inference by watching the change of the state of the trailer-truck between time moment k and $k+1$. To evaluate the result of trial and error with mathematical expressions is difficult. However, to evaluate the result linguistically with fuzzy if-then rules is easy. Therefore, we employed fuzzy inference based evaluation. In order to make the fuzzy if-then rules for evaluation, four fundamental control strategies are contrived as follows.

Control Strategy 1 If the connection angle between the trailer and the truck ϕ is large, it should be decreased to avoid falling into jackknife state.

Control Strategy 2 If the trailer-truck is directed away from the target line, the direction should be corrected to make the trailer-truck approach the target line.

Control Strategy 3 If the trailer-truck moves in an opposite direction to the target direction, the direction should be corrected to make the trailer-truck move in the target direction.

Control Strategy 4 If control strategies 1-3 are satisfied, the trailer-truck should be controlled to follow the target line considering the distance d and the angle θ. Once the trailer-truck follows the straight target line, the connection angle ϕ should be kept zero to avoid divergence.

These control strategies are represented by the fuzzy if-then rules and the membership functions shown in Table 1 and Fig.3, respectively. Five membership functions labeled as “PL”, “PS”, “ZR”, “NS” and “NL” are arranged for each input variable. The antecedent variables of the rules are the state of the trailer-truck $(\phi(k), \theta(k), d(k))$ at k, and the consequent variable of the rules is the evaluation value calculated by Eq.13-15.

$$E_\phi = \tanh(\frac{|\phi(k)|-|\phi(k+1)|}{a_\phi}), \tag{13}$$

$$E_\theta = \tanh(\frac{|\theta(k)|-|\theta(k+1)|}{a_\theta}), \tag{14}$$

$$E_d = \tanh(\frac{d(k)|-|d(k+1)|}{a_d}). \tag{15}$$

These evaluation values are the decrease of error normalized with the hyperbolic tangent function ranging from -1.0 to 1.0. The reason why the hyperbolic tangent function is used is to emphasize the evaluation nearby 0. The coefficients a_ϕ, a_θ and a_d are used to decide the slope of the function, and the values are decided as 0.6 times of maximum decrease of error (a_ϕ=3.0[deg], a_θ=3.8[deg] and a_d=0.06[m]). Finally, the evaluation value for the operation

Table 1: The fuzzy if-then rule table for evaluation. The consequent part of the rules represents an evaluation value. In the rule table, E_Φ,E_θ and E_d represent evaluation value calculated by Eq.13-15. For example, if Φ is ZR, θ is ZR and d is ZR then $E = E_\Phi$.

<table>
<tr><td colspan="2" rowspan="4"></td><td colspan="25">ϕ</td></tr>
<tr><td colspan="5">NL</td><td colspan="5">NS</td><td colspan="5">ZR</td><td colspan="5">PS</td><td colspan="5">PL</td></tr>
<tr><td colspan="5">d</td><td colspan="5">d</td><td colspan="5">d</td><td colspan="5">d</td><td colspan="5">d</td></tr>
<tr><td>NL</td><td>NS</td><td>ZR</td><td>PS</td><td>PL</td><td>NL</td><td>NS</td><td>ZR</td><td>PS</td><td>PL</td><td>NL</td><td>NS</td><td>ZR</td><td>PS</td><td>PL</td><td>NL</td><td>NS</td><td>ZR</td><td>PS</td><td>PL</td><td>NL</td><td>NS</td><td>ZR</td><td>PS</td><td>PL</td></tr>
<tr><td rowspan="5">θ</td><td>NL</td><td colspan="5" rowspan="5">$E\phi$</td><td colspan="2">$E\theta$</td><td colspan="3" rowspan="2">$E\theta$</td><td colspan="2">$E\theta$</td><td colspan="3" rowspan="2">$E\theta$</td><td colspan="2">$E\phi$</td><td colspan="3" rowspan="3">$E\phi$</td><td colspan="5" rowspan="5">$E\phi$</td></tr>
<tr><td>NS</td><td>Ed</td><td>$E\theta$</td><td>Ed</td><td>$E\theta$</td><td rowspan="2">Ed</td><td rowspan="2">$E\phi$</td></tr>
<tr><td>ZR</td><td colspan="3" rowspan="3">$E\phi$</td><td rowspan="2">$E\phi$</td><td rowspan="2">Ed</td><td colspan="2">Ed</td><td>$E\phi$</td><td colspan="2">Ed</td></tr>
<tr><td>PS</td><td colspan="3" rowspan="2">$E\theta$</td><td>$E\theta$</td><td>Ed</td><td colspan="3" rowspan="2">$E\theta$</td><td>$E\theta$</td><td>Ed</td></tr>
<tr><td>PL</td><td colspan="2">$E\phi$</td><td colspan="2">$E\theta$</td><td colspan="2">$E\theta$</td></tr>
</table>

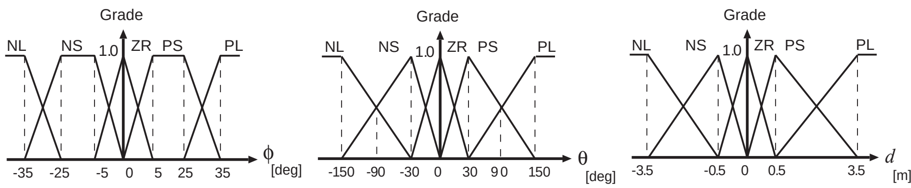


Figure 3: Membership functions of the antecedent part of rules.

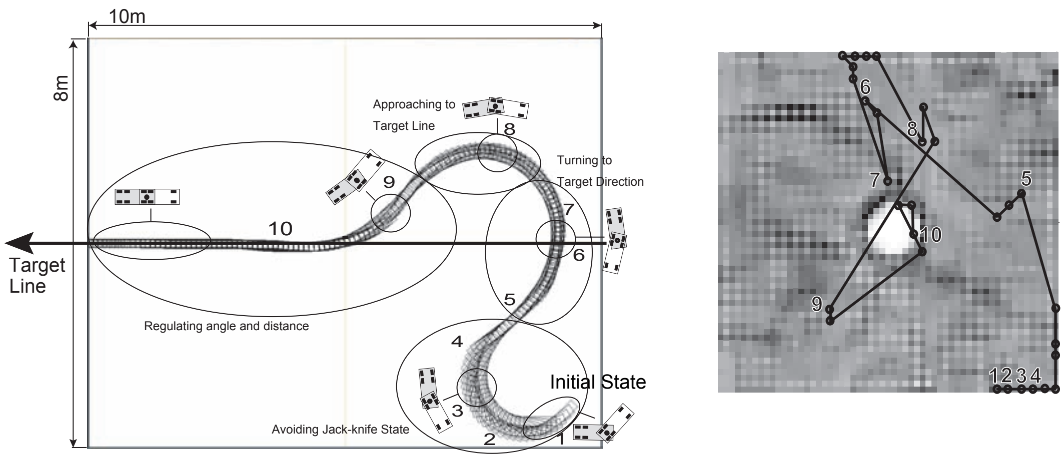
σ in a certain state (*i.e.* the I/O relationship of the controller) is calculated as a result of the fuzzy inference with the product-sum-gravity method. It is not so difficult to consider these control strategies and the fuzzy if-then rules even if a designer does not have enough knowledge about dynamics of the trailer-truck. Additionally, in these control strategies and fuzzy if-then rules, it is not referred at all how to operate the front wheel angle of the truck.

3.2 Computer Simulation Result

The computer simulation of the trailer-truck back-up control employing the proposed method is achieved. The acquisition of the learning vectors and evaluation with fuzzy inference are achieved in the method of the description in the previous section. The number of the learning vectors for the SOR network is 50,000. In the learning of the SOR network, the number of learning iterations is 50, the number of if-then units on the knowledge layer is 900 (30×30), $\alpha = 0.1$, $\eta_r = 0.2$ and $\epsilon = 0.5$. The η decreases from 50 to 1 exponentially. In the execution mode of the SOR network, the parameter γ_j is decided in consideration of the distribution of the weight vectors by:

$$\gamma_j = \frac{1}{P}\sum_{p=1}^{P} \|\boldsymbol{w}_j - \boldsymbol{w}_j^{(p)}\|, \tag{16}$$

where, $\boldsymbol{w}_j^{(p)}$ is the p-th unit according to the distance to the weight vector $\boldsymbol{w}_j$. In particular, the value of P implies the value of γ_j. The value of P is decided in accordance with the number of neighboring if-then units on the knowledge layer. In this simulation, $P = 4$, because the structure of the knowledge layer used in this simulation is a two-dimensional lattice structure.



(a) (b)

Figure 4: A simulation result. (a)The trajectory of the trailer-truck. (b)The U-matrix of the knowledge layer.

(a) (b) (c) (d)

Figure 5: The component plane representation. (a)ϕ. (b)θ. (c)d. (d)σ.

A trajectory of the trailer-truck controlled by the designed controller is shown in Fig.4(a). In the figure, the lengths of the work space are 10.0m and 8.0m for the x and y axes, respectively. The trailer-truck is controlled to follow the straight target line from a initial state. It is confirmed that the trailer-truck can follow the target line from other initial states. These results show that the sophisticated operation is successfully earned by the learning of the SOR network, even though the designer does not give the detail of the operation expressly. Fig.4(b) shows the U-matrix of the knowledge layer after the learning. The line in the figure represents a history of the best matching unit for the input states. The indexes from 1 to 10 correspond to each index in Fig.4(a). After the learning, the SOR network extracts symbolic knowledge on the knowledge layer. Each if-then unit stores crisp if-then rules. Fig.5(a)-(d) show the component plane representation for each state variable θ, d, ϕ, and operational value σ. The value of each variable is indicated by gray level. The black and white regions represent negative and positive values, respectively. The designer can know how to operate the front wheel angle σ by watching the if-then rule stored in each unit. The designer can also know similar knowledge by analyzing the clusters on the U-Matrix due to the if-then rules are topologically mapped on the knowledge layer.

4 Conclusions

In this paper, we proposed the SOR network with fuzzy inference based evaluation and its application to controller design and knowledge acquisition. The SOR network extracted practical knowledge (*i*.e. desired I/O relationship) from I/O data set which had been obtained by trial and error and evaluation based on designer's commonsense knowledge. The extracted knowledge was visualized by the two-dimensional knowledge layer. Furthermore, the SOR network worked as a fuzzy controller realizes the trailer-truck back-up control. The designer can obtain useful knowledge and easily analyze the state of the controller by watching the knowledge layer. Traditional fuzzy neural networks can also extract such if-then rules, but can not visualize the relationship among rules. In addition, traditional fuzzy neural networks just summarize the I/O relationship of the target system by supervised learning. On the other hand, the SOR network extracts practical knowledge from commonsense knowledge.

Additionally, the designer can easily feed the information obtained from the knowledge layer back to the evaluation rule description. The designer can partially modify the evaluation rule because the evaluation rules are described by decentralized fuzzy if-then rules. The obtained U-Matrix also implies the information about the density of the reference vectors in the I/O space. The designer can plan the strategy for the sampling of the learning vectors. Through these processes, the designer and the SOR network act on each other and improve themselves interactively.

Acknowledgements

This work was supported by a COE program (center #J19) granted to Kyushu Institute of Technology by MEXT of Japan. We would also like to thank Prof. Tetsuo Furukawa for his enthusiastic discussion about a batch learning algorithm of the SOM with attractive and repulsive data.

References

[1] T. Yamakawa and K. Horio, "Self-organizing relationship (SOR) network," *IEICE Transaction on Fundamentals*, Vol.E82-A, pp.1674-1678, 1999.

[2] Teuvo Kohonen, *Self-Organizing Maps*, Springer, 1997.

[3] K. Horio ,T. Haraguchi and T. Yamakawa, "A Intuitive Contrast Enhancement of an Image Data Employing the Self-Organizing Relationship (SOR) Network," *Proc. of IJCNN'99*, pp.10-16, 1999.

[4] T. Koga, K. Horio and T. Yamakawa, "Self-Organizing Relationship (SOR) Network with Fuzzy Inference Based Evaluation and Its Application to Trailer-Truck Back-Up Control," *Proc. of ICONIP'04*, pp.368-374, 2004.

[5] A. Ultsch and H.P. Siemon, "Kohonen's self-organizing feature maps for exploratory data analysis," *In Proc. INNC'90*, pp.305-308, 1990.

[6] T. Furukawa, S. Sonoh, K. Horio and T. Yamakawa, "Batch Learning Algorithm of SOM with Attractive and Repulsive Data," *Proc. of WSOM'05*, 2005.