Available online at www.sciencedirect.com
SCIENCE DIRECT®
Neural Networks 19 (2006) 799–811
Neural Networks
www.elsevier.com/locate/neunet
2006 Special Issue

The Self-Organizing Relationship (SOR) network employing fuzzy inference based heuristic evaluation

Takanori Koga*, Keiichi Horio, Takeshi Yamakawa[1]

Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology, 2-4 Hibikino, Wakamatsu-ku, Kitakyushu, 8080196, Japan

Abstract

When human beings acquire a new skill, this usually is accomplished by the summarization of numerous experiences based on their own evaluation criteria. Usually these experiences are obtained by trial and error. The criteria for success and failure are based on our own knowledge or advice given by others. The Self-Organizing Relationship (SOR) network has been inspired by this process and has been proposed to emulate this process computationally. In the previous applications of the SOR network for controller design, the evaluation criteria have been assigned by using mathematical expressions. Generally, however, mathematical expressions of the evaluation criteria become difficult as the complexity of a target system increases. On the other hand, human beings can contrive to express their knowledge for evaluation by using heuristic expressions, although a target system is complicated. In this study, we employ fuzzy inference in order to realize heuristic expressions of the evaluation criteria.

Keywords: Self-organization; Self-Organizing Relationship (SOR) network; Evaluation; Heuristic knowledge; Trailer-truck back-up control; Fuzzy inference

1. Introduction

The original self-organizing relationship (SOR) network (Yamakawa & Horio, 1999) was proposed as an extension of the Self-Organizing Maps (SOM) (e.g. Kohonen, 1982, 2001) in order to realize an approximation of a desirable I/O relationship of a target system. The Counter-Propagation Network (CPN) (Hecht-Nielsen, 1987) is also a typical extension of the SOM, and can approximate a desirable I/O relationship of a target system. The CPN has been applied to various applications (e.g. Whittaker & Cook, 1995; Zupan, Novic, & Ruisanchez, 1997). However, the CPN cannot be applied to system designs when a desirable I/O relationship is unavailable because the CPN requires a supervised signal.

On the other hand, the learning of the SOR network is achieved by a so-called "learning with a critic". In particular, the SOR network can extract a desirable I/O relationship by using learning vectors and their evaluations. The learning vectors imply the I/O relationship of a target system. They are acquired by trial and error. The learning vectors can be subjectively evaluated by intuition of designers or objectively evaluated by mathematical evaluation functions. To employ the SOR network is especially effective in the case that it is difficult or costly to acquire the desirable I/O relationship. This is because the SOR network utilizes even the learning vectors with negative evaluation values for learning. That is, the SOR network can learn from undesirable behaviors as well. One successful application of employing subjective evaluation is image enhancement (Horio, Haraguchi, & Yamakawa, 1999). On the other hand, successful applications of employing objective evaluation are the adaptive control of a DC motor (Horio & Yamakawa, 2000) and power system stabilization (Horio et al., 1999). In these control applications, the objective evaluation values were quantitatively calculated by simple mathematical evaluation functions.

However, in more complicated control systems, e.g. trailer–truck back-up control, it is difficult to describe the designer's knowledge with simple mathematical expressions for evaluation. Trailer–truck back-up control is regarded as a control of a nonholonomic system. Brockett proved that any nonholonomic system cannot be stabilized by a continuous and time-invariant state feedback system (Brockett, 1983).

* Corresponding author.
E-mail addresses: koga-takanori@edu.brain.kyutech.ac.jp (T. Koga), horio@brain.kyutech.ac.jp (K. Horio), yamakawa@brain.kyutech.ac.jp (T. Yamakawa).
[1] Tel.: +81 93 695 6123; fax: +81 93 695 6133.

0893-6080/$ - see front matter
doi:10.1016/j.neunet.2006.05.008

Therefore, generally, many control methodologies consist, mainly, of discontinuous control, switching control, time-variant control and so on. Besides, the existence of an uncontrollable state called the "jack-knife state" makes the control difficult. Therefore, this problem is a well-known benchmark problem not only in the field of nonlinear control (e.g. Kolmanovsky, Reyhanoglu, & McClamroch, 1996; Pomet, 1992; Prieur & Astolfi, 2002; Sampei, Tamura, Kobayashi, & Shibui, 1995; Tilbury, Murray, & Sastry, 1995) but also in the field of softcomputing (e.g. Hong, Won, & Lee, 2001; Ichihashi, Miyoshi, Nagasaka, Tokunaga, & Wakamatsu, 1995; Kong & Kosko, 1992; Nguyen & Widrow, 1989; Tanaka & Sano, 1994).

In order to apply the SOR network to this kind of problem, in this paper, we propose a new objective evaluation method of learning vectors for the SOR network. In this method, evaluation values are calculated by fuzzy inference (e.g. Mamdani & Assilian, 1975; Mamdani, 1977; Takagi & Sugeno, 1985; Zadeh, 1965, 1973, 1989). The heuristic evaluation rules are locally described by fuzzy if–then rules according to certain situations. The fuzzy if–then rules are useful in the case that objects of description involve uncertainty and fuzziness of knowledge. The rules are smoothly switched by employing fuzzy inference. The proposed method requires only commonsense knowledge which judges whether the results of trial and error are good or bad. By using the proposed method, the controller is constructed in a self-organized fashion by the learning of the SOR network. The effectiveness of the proposed method is verified by the results of both computer simulations and practical experiments of the trailer–truck back-up control.

2. Self-Organizing Relationship (SOR) network

The SOR network consists of a set of reference vectors $\boldsymbol{v}_i$ which are associated with the units on the competitive layer (Fig. 1(a)). Usually, the competitive layer is 1D or 2D for visualization of state of the reference vectors (Koga, Horio, & Yamakawa, 2005). The $\boldsymbol{v}_i$ is the *Cartesian product* of reference vectors in the input space $\boldsymbol{w}_i$ and in the output space $\boldsymbol{u}_i$. The $\boldsymbol{w}_i$ and the $\boldsymbol{u}_i$ have N_x and N_y elements, respectively. The operation of the SOR network is divided into two modes, the learning mode (Fig. 1(b)) and the execution mode (Fig. 1(c)). The network can be established, by learning, in order to approximate a desired function $\boldsymbol{y}^* = f(\boldsymbol{x}^*)$.

In the learning mode, the SOR network learns an I/O relationship of a target system not only from good examples but also from bad examples. A learning vector $\boldsymbol{I}_l$ consists of an input vector $\boldsymbol{x}_l$ and an output vector $\boldsymbol{y}_l$. Each learning vector has an evaluation value E_l. The E_l may be assigned by the network designer's intuition or by quantitative evaluation criteria in advance. The value of E_l is positive or negative in accordance with the judgment of the designer, preference of the user or scores of examination. A positive evaluation value causes the self-organization of attraction to the learning vectors, i.e. "attractive learning" and a negative one causes repulsion from the learning vectors, i.e. "repulsive learning". Further explanations of repulsive learning are provided in (e.g. Furukawa, Sonoh, Horio, & Yamakawa, 2005; Yamakawa & Horio, 1999; Yamakawa, Horio, & Sonoh, 2005).

After learning, the reference vectors are arranged primarily in the area where desired I/O vector pairs exist. In the execution mode, the SOR network is used as an I/O relationship generator (Fig. 1(c)).

Although, so far, the SOR network has been mainly used with on-line learning (e.g. Yamakawa & Horio, 1999; Horio et al., 1999; Horio & Yamakawa, 2000; Koga, Horio, & Yamakawa, 2004), on-line learning has some difficulties in essence. For instance, the low stability of a learning process, low repeatability of learning results, excessive burdens imposed on a network designer in parameter tuning, and so on. The main cause of these problems is the order of application of learning vectors. To cope with these problems, batch learning of the SOR network (Yamakawa et al., 2005) was proposed. Furthermore, learning of SOM with attractive and repulsive data (Furukawa et al., 2005) is proposed, and it can be regarded as a learning mode of the SOR network. Both methods employ the concept of "virtual antiparticle", in which a learning vector with a negative evaluation value is virtually converted into one with a positive evaluation value. In contrast with these, we propose the simpler batch-learning law without "virtual antiparticle" in this paper.

2.1. Learning mode of the SOR network

In the learning mode, a set of learning vectors obtained by trial and error is applied to the SOR network together with evaluation values. The learning algorithm of the SOR network is summarized as follows.

Step 0. All reference vectors $\boldsymbol{v}_i = (\boldsymbol{w}_i, \boldsymbol{u}_i)$ $(i = 1, \ldots, N_v)$ are initialized by random numbers.

Step 1. A best matching unit c_l for each learning vector $\boldsymbol{I}_l = (\boldsymbol{x}_l, \boldsymbol{y}_l)$ $(l = 1, \ldots, L)$ is selected with the smallest *Euclidean Distance* as follows.

$$c_l = \arg\min_i \|\boldsymbol{I}_l - \boldsymbol{v}_i\|. \tag{1}$$

Step 2. A Gaussian neighborhood function is calculated as follows.

$$h_{c_l,i}(t) = \exp\left(-\frac{\|\boldsymbol{r}_i - \boldsymbol{r}_{c_l}\|^2}{2\sigma^2(t)}\right), \tag{2}$$

where $\boldsymbol{r}_i$ and $\boldsymbol{r}_{c_l}$ are the positions of a unit i and the best matching unit c_l on the competitive layer. $\sigma(t)$ is the width of the neighborhood function, the value of which decays with time.

Step 3. Updating values of each reference vector are calculated as follows.

For attractive learning $(E_l > 0)$:

$$\Delta \boldsymbol{v}_{i,l}^{(\mathrm{posi})} = \alpha(t) h_{c_l,i}(t) E_l^{(\mathrm{posi})} (\boldsymbol{I}_l^{(\mathrm{posi})} - \boldsymbol{v}_i(t)), \tag{3}$$

For repulsive learning $(E_l < 0)$:

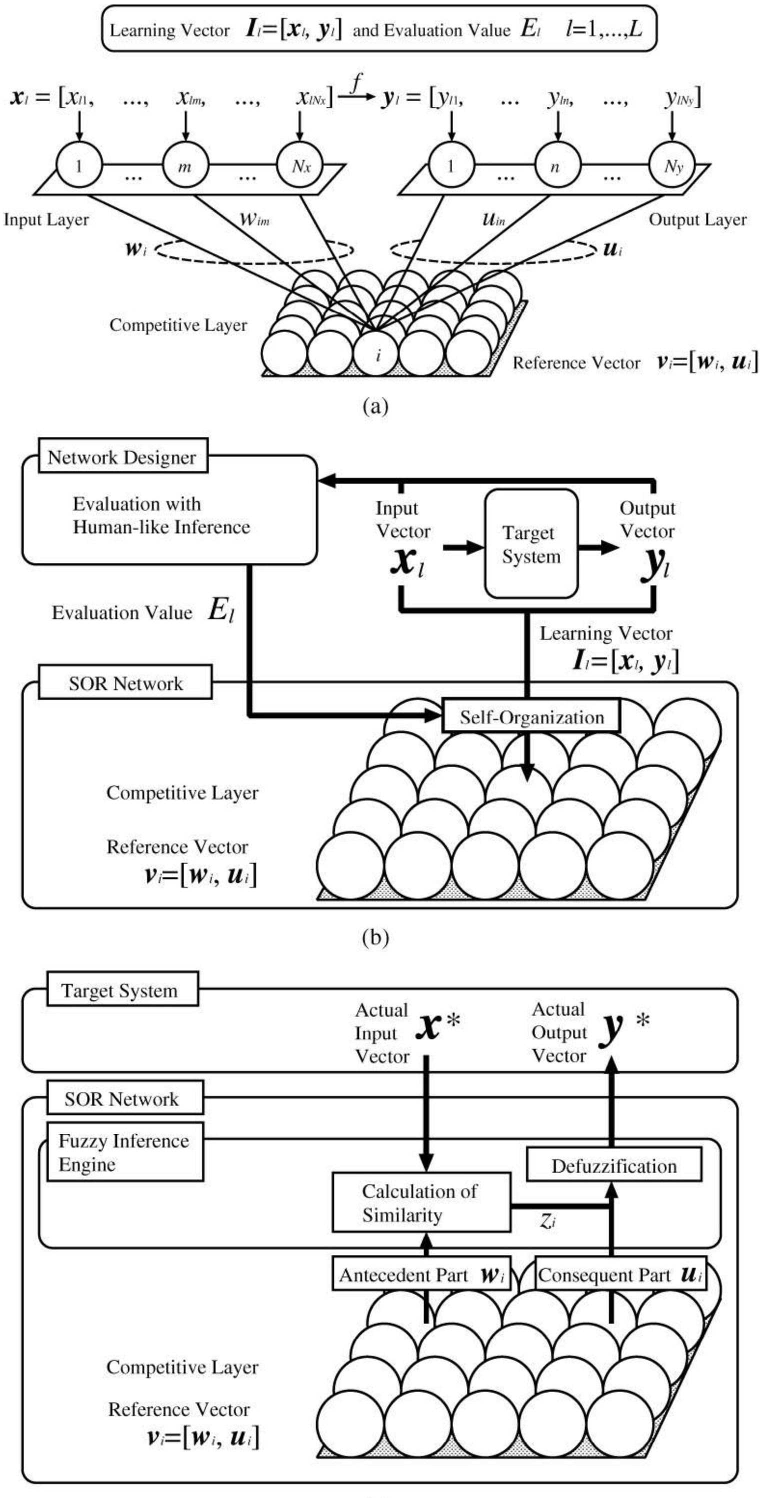


Fig. 1. (a) Architecture of SOR network. (b) Learning mode. (c) Execution mode.

$$\Delta \boldsymbol{v}_{i,l}^{(\text{nega})} = \beta(t) h_{c_l,i}(t) E_l^{(\text{nega})} \exp\left(-\frac{\|\boldsymbol{I}_l^{(\text{nega})} - \boldsymbol{v}_i(t)\|}{\sigma_r}\right) \times \frac{\boldsymbol{I}_l^{(\text{nega})} - \boldsymbol{v}_i(t)}{\|\boldsymbol{I}_l^{(\text{nega})} - \boldsymbol{v}_i(t)\|}. \tag{4}$$

Here, $\Delta \boldsymbol{v}_{i,l}^{(\text{posi})}$ and $\Delta \boldsymbol{v}_{i,l}^{(\text{nega})}$ denote the updating values for the l-th learning vector with positive and negative evaluation values, respectively. $I_l^{(\text{posi})}$ and $I_l^{(\text{nega})}$ are learning vectors with a positive evaluation value $E_l^{(\text{posi})}$ and a negative evaluation value $E_l^{(\text{nega})}$, respectively. Also, σ_r denotes a coefficient deciding the extent of the repulsive effect, $\alpha(t)(>0)$ a coefficient for the attractive learning and $\beta(t)(>0)$ a coefficient for the repulsive learning. Eqs. (3) and (4) are basically same as the on-line updating laws of the original SOR network (Yamakawa & Horio, 1999). Then, each reference vector is finally updated as follows.

$$\boldsymbol{v}_i(t+1) = \boldsymbol{v}_i(t) + \frac{\sum_{l=1}^{L^{(\mathrm{posi})}} \Delta \boldsymbol{v}_{i,l}^{(\mathrm{posi})} + \sum_{l=1}^{L^{(\mathrm{nega})}} \Delta \boldsymbol{v}_{i,l}^{(\mathrm{nega})}}{\sum_{l=1}^{L^{(\mathrm{posi})}} \alpha(t) h_{c_l,i}(t) E_l^{(\mathrm{posi})} + \sum_{l=1}^{L^{(\mathrm{nega})}} \beta(t) h_{c_l,i}(t) |E_l^{(\mathrm{nega})}|}, \quad (5)$$

where $L^{(\mathrm{posi})}$ and $L^{(\mathrm{nega})}$ are the numbers of the learning vectors with positive and negative evaluation values, respectively. All updating values given by Eqs. (3) and (4) are accumulated and normalized in this process.

Step 4. Steps 1 to 3 are repeated with decreasing $\sigma(t)$, $\alpha(t)$ and $\beta(t)$ monotonically. Usually, these values are calculated at each learning step t by the following equations.

$$\sigma(t) = (\sigma(0) - \sigma(\mathrm{final})) \times \exp\left(-\frac{t}{\tau_\sigma}\right) + \sigma(\mathrm{final}), \quad (6)$$

$$\alpha(t) = (\alpha(0) - \alpha(\mathrm{final})) \times \exp\left(-\frac{t}{\tau_\alpha}\right) + \alpha(\mathrm{final}), \quad (7)$$

$$\beta(t) = (\beta(0) - \beta(\mathrm{final})) \times \exp\left(-\frac{t}{\tau_\beta}\right) + \beta(\mathrm{final}), \quad (8)$$

where τ_σ, τ_α and τ_β denote each decay rate.

2.2. Execution mode of the SOR network

The execution mode of the SOR network corresponds to a simplified fuzzy inference with N_v if–then rules (Fig. 1(c)). The output of the network $\boldsymbol{y}^* = [y_1^*, \ldots, y_n^*, \ldots, y_{N_y}^*]$ is the weighted average of $\boldsymbol{u}_i$ by the similarity measure z_i, that is, n-th element of $\boldsymbol{y}^*$ is calculated as follows.

$$y_n^* = \frac{\sum_{i=1}^{N_v} z_i u_{in}}{\sum_{i=1}^{N_v} z_i}. \quad (9)$$

Similarity measures z_i between an actual input vector $\boldsymbol{x}^* = [x_1^*, \ldots, x_m^*, \ldots, x_{N_x}^*]$ and all reference vectors in input space $\boldsymbol{w}_i$ of the units on the competitive layer are calculated by

$$z_i = \exp\left(-\frac{\|\boldsymbol{x}^* - \boldsymbol{w}_i\|^2}{2\gamma_i^2}\right), \quad (10)$$

where γ_i is a parameter representing fuzziness of similarity. The parameter γ_i is decided in consideration of the distribution of the reference vectors by

$$\gamma_i = \frac{1}{S} \sum_{s=1}^{S} \|\boldsymbol{w}_i - \boldsymbol{w}_i^{(s)}\|, \quad (11)$$

where $\boldsymbol{w}_i^{(s)}$ is the s-th nearest reference vector according to the distance to the reference vector $\boldsymbol{w}_i$. The value of S can be assigned empirically, which is explained in Section 5.

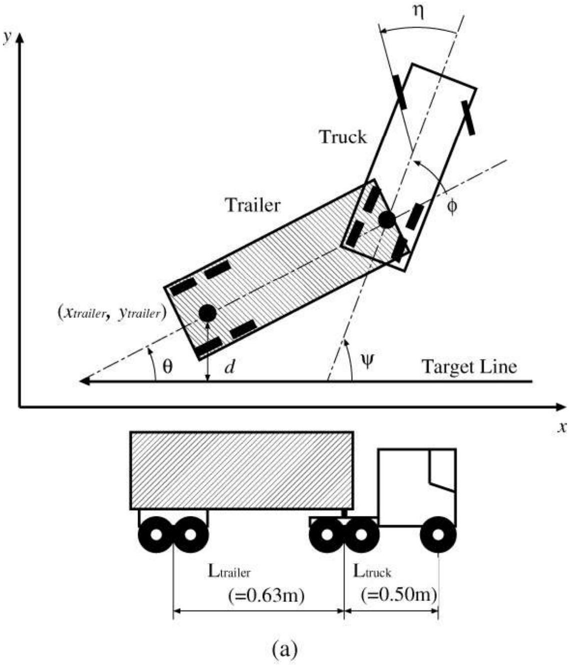
3. Control of a nonholonomic system (trailer–truck back-up control)

In the learning mode of the SOR network, the network is established based on the learning vectors and their evaluation values. In the case that the SOR network is applied to a control problem, the evaluation values should be quantitatively calculated. The evaluation values, in the power system stabilization (Yamakawa & Horio, 1999) and adaptive control of a DC motor (Horio & Yamakawa, 2000), are simply defined by the decrease of error.

However, in more complicated control systems, e.g. trailer–truck back-up control, it is difficult to describe the designer's knowledge with mathematical expressions. A mobile robot such as trailer–truck cannot be directly regulated from any initial positions to a target position. This is because car-like mobile robots are nonholonomic systems (e.g. Laumond, Sekhavat, & Lamiraux, 1998). Nonholonomic systems are characterized by constraint equations involving the time derivatives of the system configuration variables. These equations are non-integrable. Nonholonomic systems cannot be stabilized by continuous and time-invariant state feedback system (Brockett, 1983). Generally, this kind of system should be regulated to a target state via appropriate subgoals. In the field of control theory, there are many papers in which the controllers for nonholonomic systems are designed by using nonlinear control theories (e.g. Kolmanovsky et al., 1996; Pomet, 1992; Prieur & Astolfi, 2002; Sampei et al., 1995; Tilbury et al., 1995). Generally, these methods need advanced mathematical knowledge. In contrast to this, however, there are papers in which softcomputing techniques are employed for nonholonomic systems, in particular trailer–truck back-up control, (e.g. Hong et al., 2001; Ichihashi et al., 1995; Kong & Kosko, 1992; Nguyen & Widrow, 1989; Tanaka & Sano, 1994).

In order to apply the SOR network to this nonholonomic problem, we propose a new method where the learning vectors are evaluated by fuzzy inference with fuzzy control strategies. The description in detail appears in the next section.

A semi-trailer type trailer–truck is used in this study. The trailer is connected to the truck with a rotatable shaft, and is pushed back at one point when the truck backs up. The control objective is to make the trailer–truck follow the target line from any initial state by only backward movement at constant velocity. The trailer–truck control is a severe nonlinear control. The trailer–truck easily meets unstable states, the so-called "jack-knife states". This is because the connection angle (angle from alignment of the trailer and the truck) easily becomes large. Once the connection angle is excessively enlarged, the trailer–truck cannot be controlled any more by only backward movement. A model of the semi-trailer type trailer–truck used in this study is shown in Fig. 2(a). The parameters and variables of the trailer–truck are shown in Table 1. The kinematic model



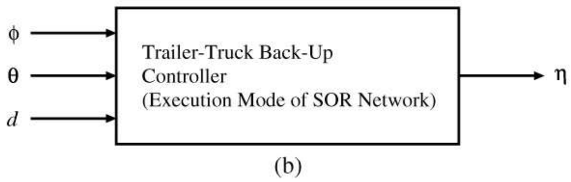


Fig. 2. (a) Model of the semi-trailer type trailer–truck. (b) Trailer–truck back-up controller constructed in this paper.

Table 1
Parameters and variables of the trailer–truck

L_{trailer}	Length of the trailer (=0.63 m)
L_{truck}	Length of the truck (=0.50 m)
v	Velocity of the truck (= − 0.1 m/s)
η	Front wheel angle of the truck $-30° \le \eta \le 30°$
ψ	Angle of the truck $-180° \le \psi \le 180°$
θ	Angle of the trailer $-180° \le \theta \le 180°$
ϕ	Connection angle of the trailer–truck $-45° \le \phi \le 45°$
x_{trailer}	X-coordinate of the trailer
y_{trailer}	Y-coordinate of the trailer
d	Position error against the target line

of the trailer–truck (Ichihashi et al., 1995; Tanaka & Sano, 1994) is described by

$$\psi[k+1] = \psi[k] + \frac{v \cdot \Delta t \cdot \tan \eta[k]}{L_{\mathrm{truck}}}, \tag{12}$$

$$\theta[k+1] = \theta[k] + \frac{v \cdot \Delta t \cdot \sin \phi[k]}{L_{\mathrm{trailer}}}, \tag{13}$$

$$\phi[k] = \psi[k] - \theta[k], \tag{14}$$

$$x_{\mathrm{trailer}}[k+1] = x_{\mathrm{trailer}}[k] + v \cdot \Delta t \cdot \cos \phi[k] \cdot \cos \frac{\theta[k+1] + \theta[k]}{2}, \tag{15}$$

$$y_{\mathrm{trailer}}[k+1] = y_{\mathrm{trailer}}[k] + v \cdot \Delta t \cdot \cos \phi[k] \cdot \sin \frac{\theta[k+1] + \theta[k]}{2}, \tag{16}$$

where k is a time step, and Δt a sampling period ($\Delta t = 1.0$ [s]). In the proposed system, the connection angle 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 η (Fig. 2(b)).

4. Acquisition of learning vectors and their evaluation by fuzzy inference

In order to employ the SOR network to design a controller for a complicated system, i.e. trailer–truck back-up control, we propose a new evaluation method by employing fuzzy inference (e.g. Mamdani & Assilian, 1975; Mamdani, 1977; Takagi & Sugeno, 1985; Zadeh, 1965, 1973, 1989). Fuzzy inference has many features, for example, (1) experiential knowledge which involves uncertainty is easily described by simple form of fuzzy if–then rules, (2) the practical non-linear system can be modeled by a set of fuzzy if–then rules rather than by mathematical expressions, (3) each of the fuzzy if–then rules can be revised individually if it is incorrect.

Fig. 3 shows the procedure of the learning vector acquisition. First, a state $(\phi(k), \theta(k), d(k))$ of the trailer–truck at time step k is randomly given. Then, the front wheel angle $\eta(k)$ is randomly given as an operation at time k. These values are elements of the learning vectors. As a result of operation, the states at $k+1$ and $k+2$ are calculated using the kinematic model given by Eqs. (12)–(16). The designer evaluates the I/O relationship of the controller with fuzzy inference by watching the change of the state of the trailer–truck for two sampling intervals. In order to construct the fuzzy if–then rules for evaluation, four fundamental control strategies are heuristically 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 the jack-knife state.

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

Control Strategy 3
If the trailer–truck moves in a direction opposite to the target direction, the direction of movement should be corrected to make the trailer–truck move in the target direction.

Control Strategy 4
After control strategies 1–3 are accomplished, 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 target line, the connection angle ϕ should be kept zero to avoid divergence.

We should emphasize that it is not so difficult for designers to consider these control strategies even if they do not have enough knowledge about the dynamics of the trailer–truck. In

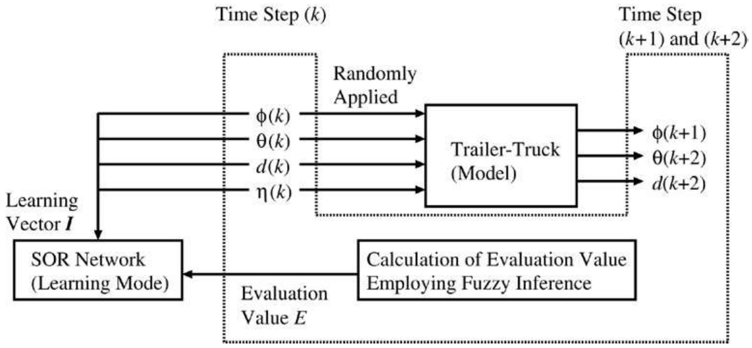


Fig. 3. Procedure of learning vector acquisition and evaluation.

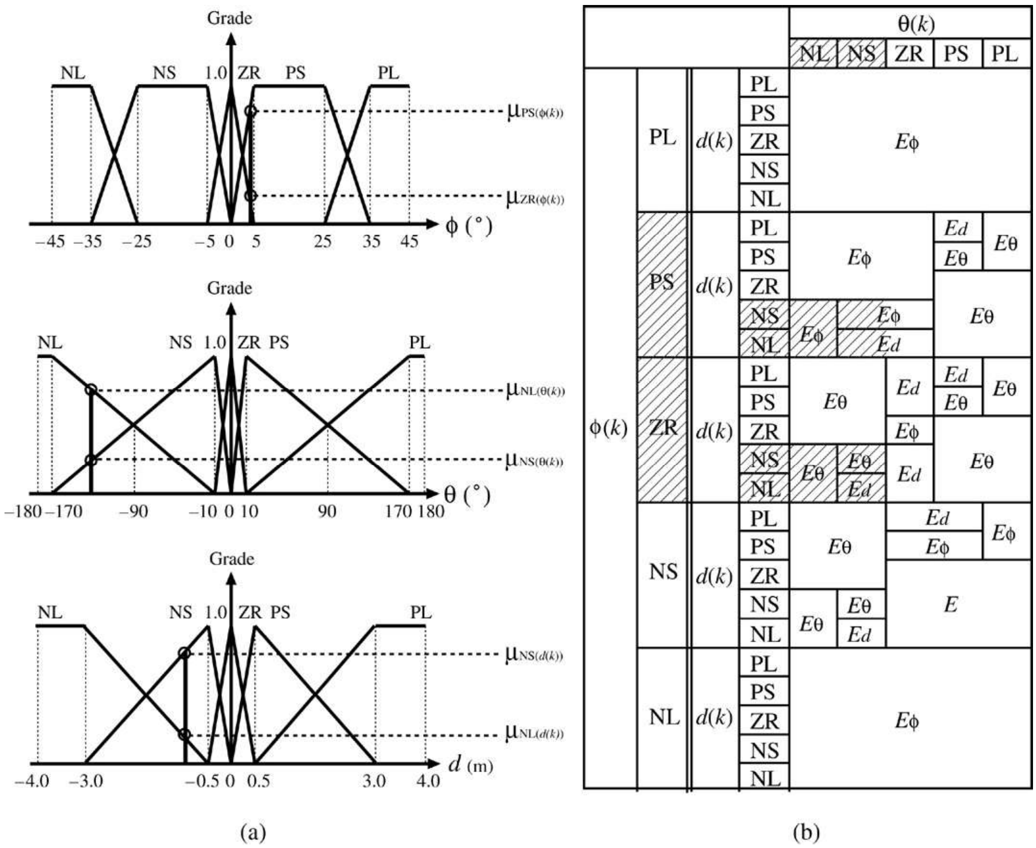


Fig. 4. Fuzzy if–then rules for evaluation. (a) Membership function of antecedent part. (b) Fuzzy if–then rule table.

addition, in these control strategies, how to operate the front wheel angle of the truck, η, is not referred to at all.

These control strategies are represented by fuzzy if–then rules (Eq. (17)) and membership functions as shown in Fig. 4.

$$\text{IF } \phi(k) \text{ is } A_{1j}, \theta(k) \text{ is } A_{qj} \text{ and } d(k) \text{ is } A_{N_{in}j}, \quad \text{THEN } E_j = E_\phi \text{ or } E_\theta \text{ or } E_d \quad j = 1, \ldots, N_r, \tag{17}$$

where j and q are suffixes of rule and input variable, respectively. N_r and N_{in} are the numbers of rule and input variables, respectively. A_{qj} is the label of a fuzzy set. Please note that the suffix of the learning vector l is omitted to clarify the explanation. The antecedent variables of the rules are the state of the trailer–truck. Five fuzzy sets labeled as "PL (Positively Large)", "PS (Positively Small)", "ZR (Approximately Zero)", "NS (Negatively Small)" and "NL (Negatively Large)" are arranged for each input variable.

In actual construction of the fuzzy if–then rule table, one of three consequent variables in Eq. (17) is selected with considering which input variable should be decreased. For example, the antecedent is given as $\phi(k)$ = PL, $\theta(k)$ = "don't care" and $d(k)$ = "don't care". Under this situation, the designer should select E_ϕ as a consequent. This fuzzy if–then rule corresponds to the control strategy 1 and is located in the top square of the rule table in Fig. 4(b). In another case, the consequent of the rule "IF $\phi(k)$ is PS, $\theta(k)$ is PS and $d(k)$ is PL, THEN?" can be assigned to be E_d, because $d(k)$ is PL, which is shown in the square under the top square in the rule table (Fig. 4(b)). In a similar manner, all the consequents of the if–then rules are assigned as shown in Fig. 4(b).

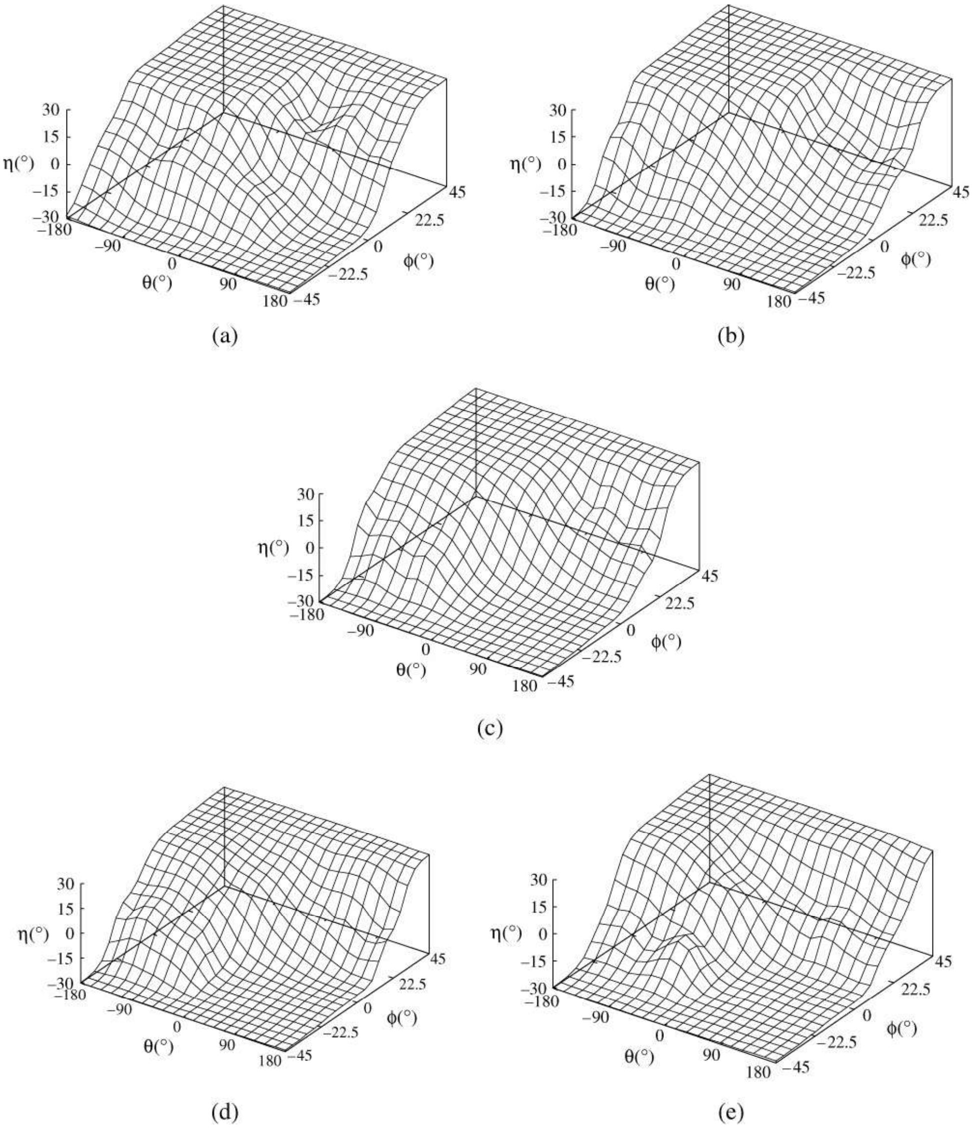


Fig. 5. I/O characteristics of the designed controller. (a) $d = 4.0$ m. (b) $d = 2.0$ m. (c) $d = 0.0$ m. (d) $d = -2.0$ m. (e) $d = -4.0$ m.

The consequent part is given as a constant value given by Eqs. (18)–(20).

$$E_\phi = \tanh\left(\frac{|\phi(k)| - |\phi(k+1)|}{a_\phi}\right), \tag{18}$$

$$E_\theta = \tanh\left(\frac{|\theta(k)| - |\theta(k+2)|}{a_\theta}\right), \tag{19}$$

$$E_d = \tanh\left(\frac{|d(k)| - |d(k+2)|}{a_d}\right). \tag{20}$$

These evaluation values are defined as the decrease of the error normalized with the sigmoid function ranging from -1.0 to $+1.0$. The reason why the sigmoid function is used is to emphasize the evaluation near 0. The coefficients a_ϕ, a_θ and a_d decide the slope of the function, and the values are decided as 0.6 times the maximum decrease of the error ($a_\phi = 3.0°$, $a_\theta = 3.8°$ and $a_d = 0.06$ m).

In the fuzzy inference, the product–sum–gravity method (e.g. Mizumoto, 1990) is employed. First, the matching grade of each rule λ_j is calculated as follows.

$$\lambda_j = \prod_{q=1}^{N_{in}} \mu_{A_{qj}}(x_q). \tag{21}$$

Here, $\mu_{A_{qj}}(x_q)$ is membership grade of the fuzzy set A_{qj}, where x_q is the input variable, i.e. $x_1 = \phi(k)$, $x_2 = \theta(k)$ and $x_3 = d(k)$. Finally, the evaluation value E is calculated by the following equation.

$$E = \frac{\sum_{j=1}^{N_r} \lambda_j E_j}{\sum_{j=1}^{N_r} \lambda_j}. \tag{22}$$

Below is a demonstration of a concrete example of the fuzzy inference process. Suppose that a state at time step k is given by $\phi(k) = 4.0°$, $\theta(k) = -130.0°$ and $d = -1.0$ m. First, each fuzzy membership grade $\mu_{A_{qj}}(x_q)$ is derived as shown in

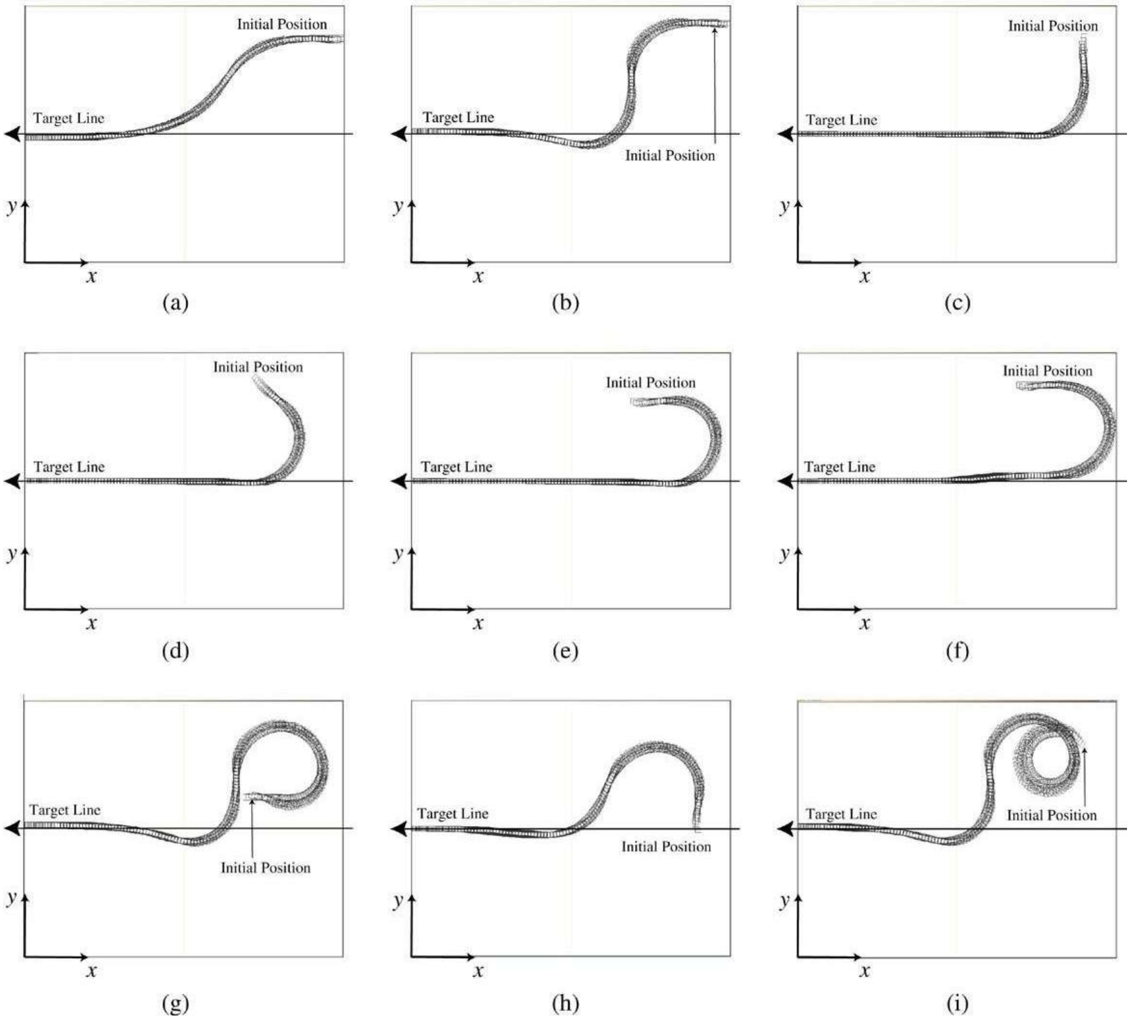


Fig. 6. Trajectories of the trailer–truck. Initial state: (a) $\phi = 0°, \theta = 0°, d = 3.0$ m. (b) $\phi = 0°, \theta = 0°, d = 3.5$ m. (c) $\phi = 0°, \theta = 90°, d = 2.0$ m. (d) $\phi = 0°, \theta = 135°, d = 2.5$ m. (e) $\phi = 0°, \theta = 180°, d = 2.5$ m. (f) $\phi = 0°, \theta = 180°, d = 3.0$ m. (g) $\phi = 0°, \theta = -180°, d = 1.0$ m. (h) $\phi = 0°, \theta = -90°, d = 1.0$ m. (i) $\phi = -45°, \theta = 0°, d = 3.0$ m.

Fig. 4(a). The concrete values are $\mu_{\mathrm{ZR}}(\phi(k) = 4.0) = 0.2$, $\mu_{\mathrm{PS}}(\phi(k) = 4.0) = 0.8$, $\mu_{\mathrm{NS}}(\theta(k) = -130.0) = 0.25$, $\mu_{\mathrm{NL}}(\theta(k) = -130.0) = 0.75$, $\mu_{\mathrm{NS}}(d(k) = -1.0) = 0.8$ and $\mu_{\mathrm{NL}}(d(k) = -1.0) = 0.2$. Next, the matching grade of each fuzzy if–then rule is calculated by Eq. (21). In this case, eight rules, i.e. the meshed-square in the rule table (Fig. 4(b)), have non-zero values. The concrete values are $\lambda_{(\phi=\mathrm{PS},\theta=\mathrm{NL},d=\mathrm{NS})} = 0.8 \times 0.75 \times 0.8 = 0.48$, i.q. $\lambda_{(\phi=\mathrm{PS},\theta=\mathrm{NL},d=\mathrm{NL})} = 0.12$, $\lambda_{(\phi=\mathrm{PS},\theta=\mathrm{NS},d=\mathrm{NS})} = 0.16$, $\lambda_{(\phi=\mathrm{PS},\theta=\mathrm{NS},d=\mathrm{NL})} = 0.04$, $\lambda_{(\phi=\mathrm{ZR},\theta=\mathrm{NL},d=\mathrm{NS})} = 0.12$, $\lambda_{(\phi=\mathrm{ZR},\theta=\mathrm{NL},d=\mathrm{NL})} = 0.03$, $\lambda_{(\phi=\mathrm{ZR},\theta=\mathrm{NS},d=\mathrm{NS})} = 0.04$ and $\lambda_{(\phi=\mathrm{ZR},\theta=\mathrm{NS},d=\mathrm{NL})} = 0.01$. Thus, final the E value is derived by Eq. (22) as follows.

$$E = \frac{(0.48+0.12+0.16)E_\phi + (0.12+0.03+0.04)E_\theta + (0.04+0.01)E_d}{0.48+0.12+0.16+0.04+0.12+0.03+0.04+0.01}. \tag{23}$$

Here, E_ϕ, E_θ and E_d can be obtained by Eqs. (18)–(20), two time steps after a certain front wheel angle is given.

Thus one learning vector $(\phi(k), \theta(k), d(k), \eta(k))$ is now acquired with the evaluation value E.

5. Computer simulation

The computer simulation of trailer–truck back-up control employing the proposed method is achieved. The acquisition of the learning vectors and evaluation with fuzzy inference is done by the method of the description in the previous section. In the learning of the SOR network, 6561 learning vectors are regularly sampled. In this regard, 100 learning vectors, which are assigned to be $(\phi, \theta, d, \eta, E_l) = (0.0, 0.0, 0.0, 0.0, 1.0)$, are deliberately added in order to ensure the stability around the target state. The number of learning iterations is 200, the number of units on the 2D competitive layer is 625 (25×25), $\alpha(0) = 1.0$, $\alpha(\text{final}) = 0.01$, $\beta(0) = 0.1$, $\beta(\text{final}) = 0.005$, $\sigma(0) = 25.0$, $\sigma(\text{final}) = 0.25$, $\sigma_r = 0.1$, $\tau_\alpha = 30.0$, $\tau_\beta = 30.0$ and $\tau_\sigma = 30.0$. In the execution mode, the parameter S in Eq. (11) is empirically assigned to be $S = 3$, by considering the distribution of the reference vectors.

5.1. I/O characteristics of SOR network

The I/O characteristic of the designed controller is represented in 4-dimensional space (3-input 1-output); hence, the I/O characteristic is shown for five cases of d in Fig. 5(a)–(e). All figures show the saturation in the front wheel angle η, which limits the larger angle to avoid the jack-knife state. This avoidance action shows the first priority over any other operations (Control Strategy 1).

Once the trailer–truck falls into the jack-knife state, it becomes uncontrollable, even if any other variables are assigned well. The curvature of each figure in Fig. 5 represents the operation to correct the direction of trailer–truck movement (Control Strategies 2 and 3), and the operation to regulate the position d and angle θ (Control Strategy 4). It is emphasized that the nonlinear I/O relationship necessary for adequate control can be self-organizingly established by the learning vectors with evaluation, which are obtained by trial and error.

5.2. Trajectories of trailer–truck

The trajectories of the trailer–truck controlled by the designed controller are shown in Fig. 6(a)–(i). In each figure, the work space is $10.0\ \text{m} \times 8.0\ \text{m}$. The trailer–truck is controlled to follow the target line from any initial state avoiding the jack-knife state. From the line-symmetric initial states for the target line, the trailer–truck can be completely regulated to the target line. These results show that the sophisticated operation is successfully earned by the learning of the SOR network without any expert knowledge, even though the designer does not give the detail of operation expressly.

5.3. Effectiveness of repulsive learning

Figs. 7–9 show experimental results which verify the effectiveness of repulsive learning. In the repulsive learning, the ratio of the final values of α and β is the dominant parameter. In these experiments, four controllers are established. Controller 1 is the above-mentioned one, i.e. a finely-tuned controller ($\beta(0) = 0.1$ and $\beta(\text{final}) = 0.005$). Controllers 2–4 are established under basically the same conditions as Controller 1. Controller 2 is established under the condition $\beta(0) = 0.0$ and $\beta(\text{final}) = 0.0$. This means that Controller 2 is established by only attractive learning. Controller 3 is established under a somewhat excessive repulsive effect, $\beta(0) = 0.1$ and $\beta(\text{final}) = 0.01$. Controller 4 is established under an extreme excessive repulsive effect, $\beta(0) = 0.1$ and $\beta(\text{final}) = 0.05$.

The comparison of control results among four controllers is shown in Fig. 7. Two typical initial states are set: Fig. 7(a) $\phi = 0°, \theta = 90°, d = 2.0$ m; Fig. 7(b) $\phi = 0°, \theta = 180°, d = 2.5$ m. All the controllers can finally regulate the trailer–truck to the target line. In particular, the SOR network can regulate the trailer–truck to the target line even if repulsive learning is omitted (i.e. using Controller 2). The lack of repulsive learning cannot be fatal.

Fig. 8 shows the difference of I/O characteristics between Controller 1 and Controller 2. In the figure, η_{diff} represents the difference of the front wheel angle η between two controllers, i.e. η of Controller 1 minus η of Controller 2. This result shows that the obtained I/O relationships of two controllers involve notable differences. The trajectories of the trailer–truck which are controlled by Controller 1 and Controller 2 are shown in Fig. 9. Fig. 9(a), (b) are the results by using Controller 1 and Fig. 9(c), (d) are those by using Controller 2. Under the situation without employing repulsive learning, the SOR network cannot control the trailer–truck to avoid actively the undesirable state. Therefore, some undesirable phenomena such as overshoot or low-speed convergence are observed.

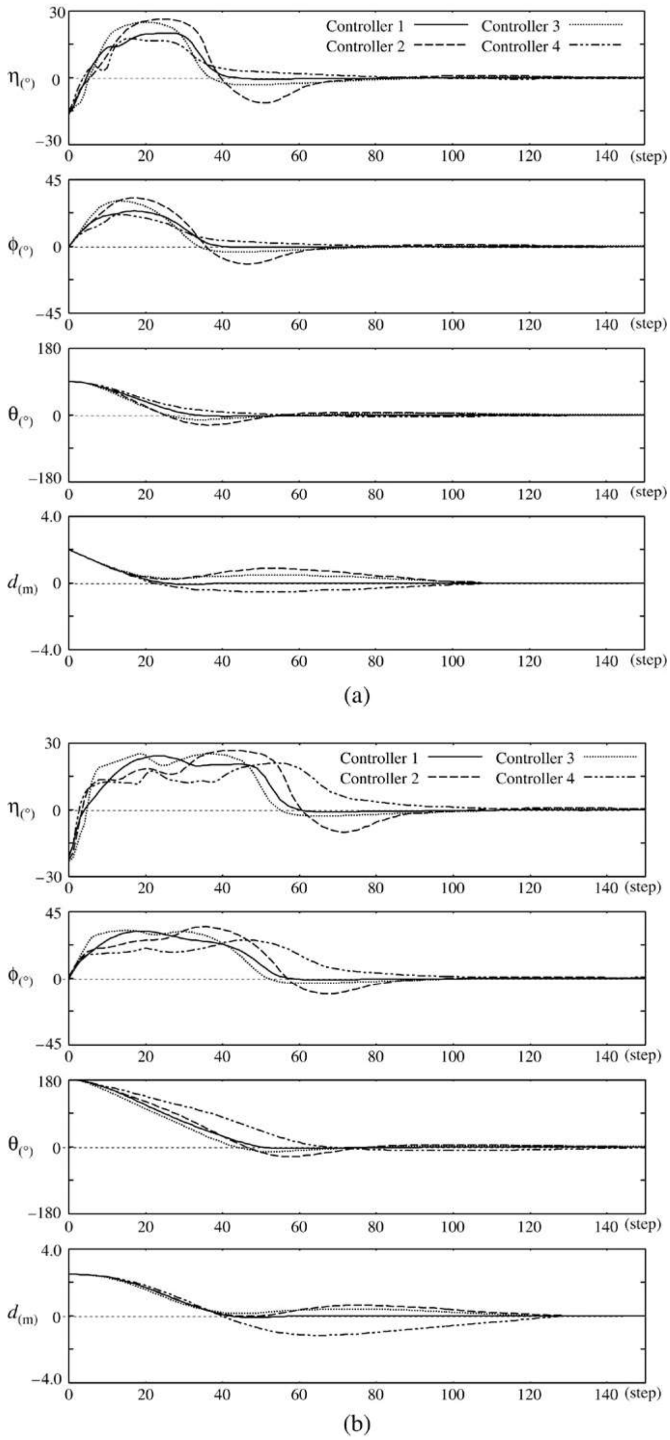


Fig. 7. Comparison of control results using Controllers 1–4. Initial state (a) $\phi = 0°, \theta = 90°, d = 2.0$ m, (b) $\phi = 0°, \theta = 180°, d = 2.5$ m.

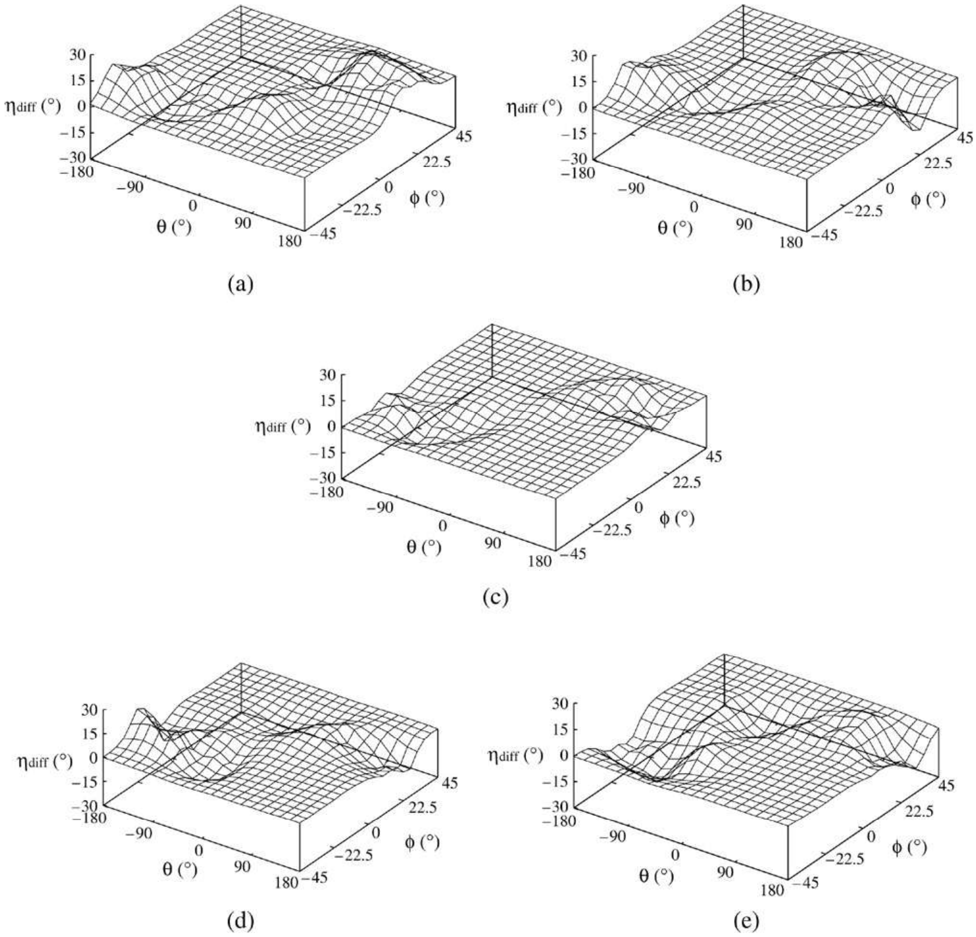


Fig. 8. Difference of I/O characteristics between Controller 1 and Controller 2. (a) $d = 4.0$ m. (b) $d = 2.0$ m. (c) $d = 0.0$ m. (d) $d = -2.0$ m. (e) $d = -4.0$ m.

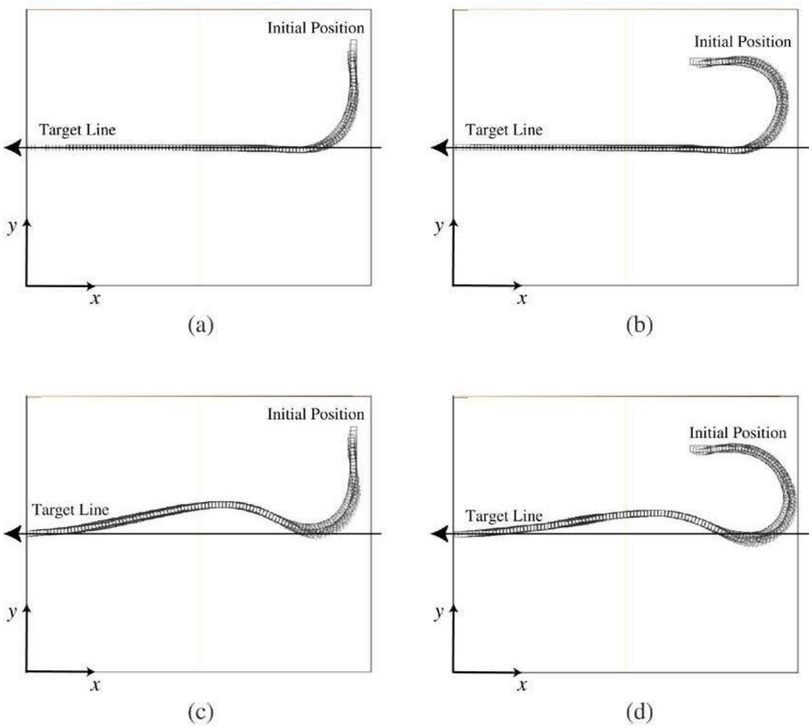


Fig. 9. Comparison of trajectories of the trailer–truck. (a), (b) and (c), (d) are results controlled by Controller 1 and Controller 2, respectively. Initial Position: (a), (c) $\phi = 0°, \theta = 90°, d = 2.0$ m, (b), (d) $\phi = 0°, \theta = 180°, d = 2.5$ m.

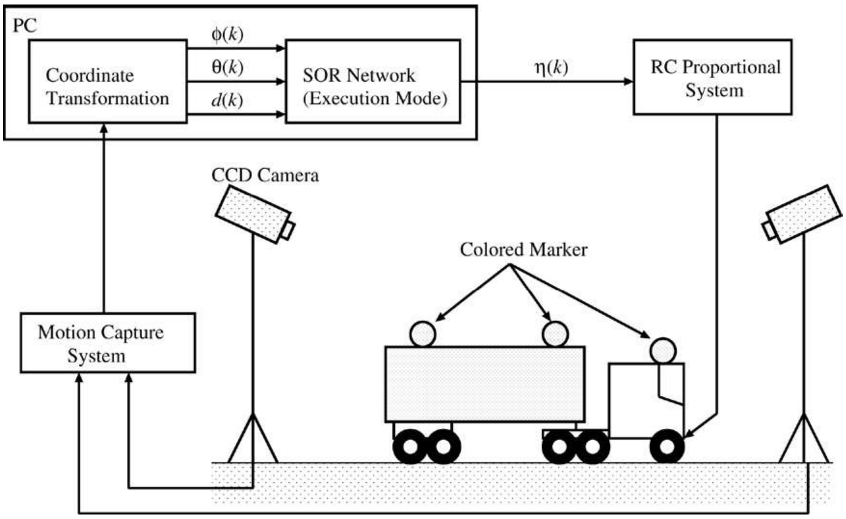


Fig. 10. Trailer–truck control system for the practical experiment.

In contrast, extreme excessive repulsive learning also breaks down the successful control (i.e. using Controller 4 in Fig. 7). The result using Controller 3 is similar to the result using Controller 1.

In this experiment, as the parameter β(final) is close to 0.005, i.e. the finely-tuned parameter, the performance of the controller tends to become good. In this regard, a quantity of difference around this parameter is permissible.

6. Practical experiment

The experimental system is shown in Fig. 10 and the flow of processing is as follows. The motion capture system with two CCD cameras captures the coordinates of the three markers attached to the trailer–truck, and the coordinates are input to the PC through the digital I/O board to calculate the angles ϕ, θ and the distance d. Then the front wheel angle η is calculated as the output of the SOR network. The value of η is sent to the truck through the D/A converter and the remote control proportional system. The reference vectors of the SOR network have been established in the same manner as in Section 5 (Computer simulation). Fig. 11 shows an experimental result. A photograph was taken every two seconds. The black line on the floor is the target line. The initial values of the distance and angles are $\phi = 0°$, $\theta = 90°$ and $d = 2.0$ m as shown in Fig. 11(a). The point of view is switched after a lapse of 16 s. The right side of Fig. 11(h) indicates almost the same position as the one in the left side of Fig. 11(i). The angle of the trailer becomes smaller and the trailer–truck finally follows the target line. It is confirmed that the trailer–truck can follow the target line even in the case of other initial values.

7. Conclusions

In this paper, we proposed a new evaluation algorithm for the learning vectors for the SOR network, in which fuzzy inference was employed. The proposed method was applied to trailer–truck back-up control, which is a well-known benchmark problem as a control of nonholonomic system. We verified the effectiveness of the proposed method both in the computer simulation and a practical experiment with a remote-controlled trailer–truck. In the proposed method, all the designer has to do is to acquire the learning vectors by trial and error and to evaluate the vectors with fuzzy if–then rules constructed according to the designer's commonsense knowledge. The proposed method is much easier than a controller design method employing nonlinear control theory. This is because the proposed method does not require any advanced mathematical knowledge and expressions. Furthermore, the proposed method is a model-free approach.

Seemingly, to employ the rule-based fuzzy logic controller which should control the trailer–truck is effective because the design is linguistically achieved. However, the designer should be well-informed about the desirable I/O relationships of the controller. However, such special knowledge (i.e. the desirable I/O relationship) is not always available at any time. Furthermore, in the practical design process, fine adjustment of the parameters is required. On the other hand, the fuzzy if–then rules used in the proposed algorithm can be constructed without expert special knowledge about the desirable I/O relationships of the controller.

In the learning of the SOR network, undesirable I/O relationships obtained by trial and error are actively utilized through repulsive learning. This approach is somewhat similar to Reinforcement learning (Sutton & Barto, 1998). In Reinforcement learning, a controller is established by the interaction with the environment. Generally, however, Reinforcement learning is a search-based algorithm, and it requires a huge number of trials and errors. In contrast, the SOR network can be regarded as an algorithm which aims to utilize profoundly the designer's knowledge.

In our future work, we aim to develop the SOR network not only as a controller but as a knowledge acquisition tool. The learning algorithm of the SOR network is basically the same as

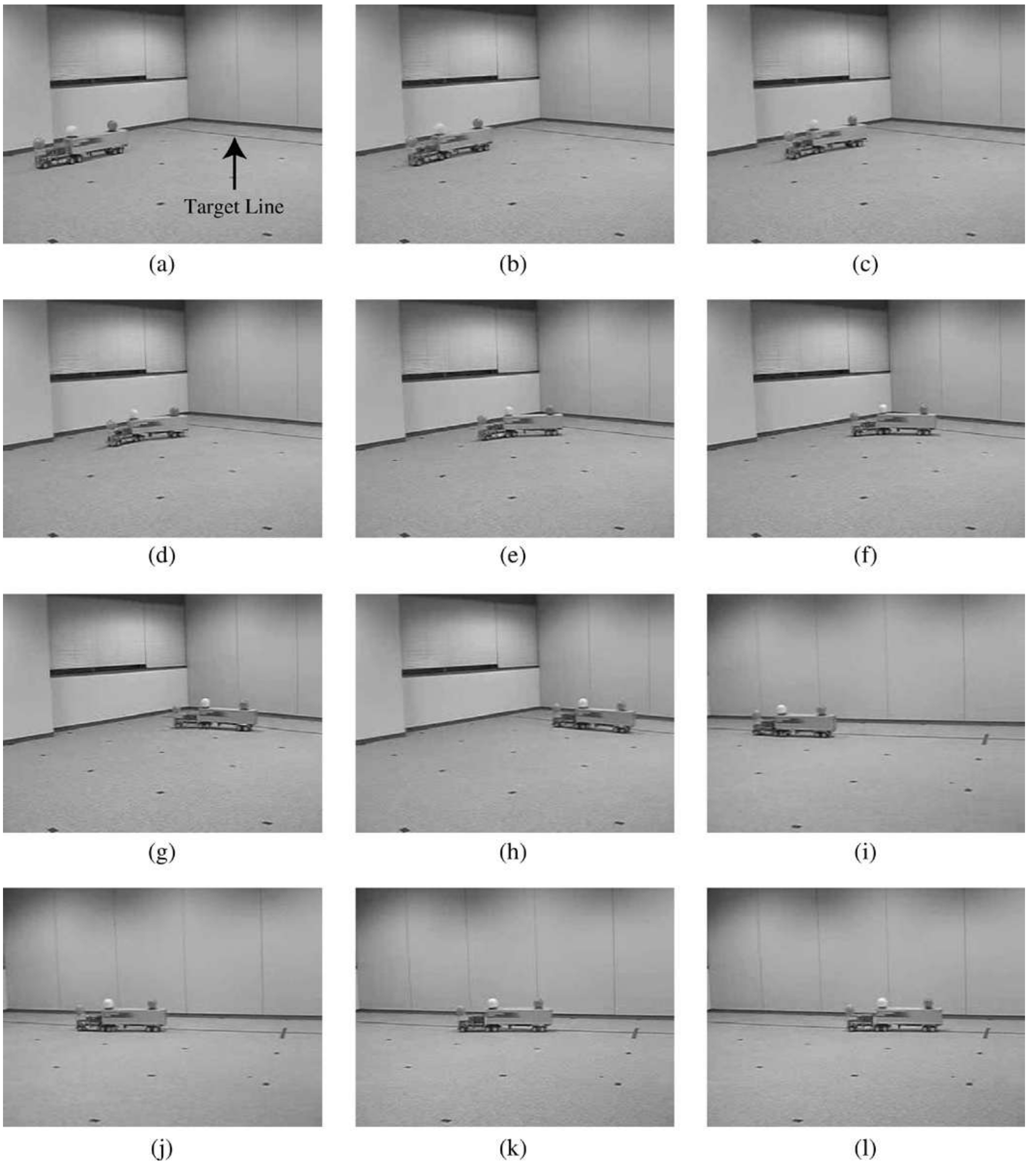


Fig. 11. Practical experimental result. The black line on the floor is the straight target line. (a) Initial position. (b)–(k) Intermediate steps. (l) Final position. The photographs were taken at intervals of two seconds.

the SOM. Therefore, the information stored in the units of the competitive layer can be analyzed as mentioned in Koga et al. (2005). Finally, we aim at a construction of a system in which the designer and the SOR network act on each other in order to improve their knowledge.

Acknowledgements

This work was supported by a 21st Century Center of Excellence Program, "World of Brain Computing Interwoven out of Animals and Robots (PI: T. Yamakawa)", granted in 2003 to the Department of Brain Science and Engineering (Graduate School of Life Science and Systems Engineering), Kyushu Institute of Technology by Japan Ministry of Education, Culture, Sports, Science and Technology. This work is also supported by Grant-in-Aid for Scientific Research (A), 2002, 14205038.

References

Brockett, R. W. (1983). Asymptotic stability and feedback stabilization. In R. W. Brockett, R. Millman, & H. Sussmann (Eds.), *Differential geometric control theory* (pp. 181–191). Boston: Birkhauser.

Furukawa, T., Sonoh, S., Horio, K., & Yamakawa, T. (2005). Batch learning algorithm of SOM with attractive and repulsive data. In *Proceedings of 5th workshop on self-organizing maps* (pp. 413–420).

Hecht-Nielsen, R. (1987). Counter-propagation networks. In *Proceedings of the IEEE first international conference on neural networks*, vol. 2 (pp. 19–32).

Hong, C. S., Won, J. M., & Lee, J. S. (2001). Multi-thread evolutionary programming and its application to truck-and-trailer backer-upper control. *IEICE Transactions on Fundamentals*, *E84-A(2)*, 597–603.

Horio, K., Haraguchi, T., & Yamakawa, T. (1999). An intuitive contrast enhancement of an image data employing the self-organizing relationship (SOR). In *Proceedings of international joint conference on neural networks* (pp. 10–16).

Horio, K., & Yamakawa, T. (2000). Adaptive self-organizing relationship network and its application to adaptive control. In *Proceedings of the 6th international conference on softcomputing and information/intelligent systems* (pp. 299–304).

Ichihashi, H., Miyoshi, T., Nagasaka, K., Tokunaga, M., & Wakamatsu, T. (1995). A neurofuzzy approach to variational problems by using gaussian membership functions. *International Journal of Approximate Reasoning*, *13*(4), 287–302.

Koga, T., Horio, K., & Yamakawa, T. (2004). Self-organizing relationship (SOR) network with fuzzy inference based evaluation and its application to trailer-truck back-up control. In *Proceedings of 11th international conference on neural information processing* (pp. 368–374).

Koga, T., Horio, K., & Yamakawa, T. (2005). Applications of brain-inspired SOR network to controller design and knowledge acquisition. In *Proceedings of 5th workshop on self-organizing maps* (pp. 687–694).

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*, 59–69.

Kohonen, T. (2001). *Self-organizing Maps* (3rd ed.). Berlin: Springer.

Kolmanovsky, I., Reyhanoglu, M., & McClamroch, N. H. (1996). Switched mode feedback control laws for nonholonomic systems in extended power form. *Systems & Control Letters*, *27*(1), 29–36.

Kong, S. G., & Kosko, B. (1992). Adaptive fuzzy-systems for backing-up a truck-and-trailer. *IEEE Transactions on Neural Networks*, *3*(2), 211–223.

Laumond, J. P., Sekhavat, S., & Lamiraux, F. (1998). Guidelines in nonholonomic motion planning for mobile robots. *Robot Motion Planning and Control*, 229.

Mamdani, E. H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, *7*(1), 1–13.

Mamdani, E. H. (1977). Applications of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, *26*(12), 1182–1191.

Mizumoto, M. (1990). Fuzzy controls by product-sum-gravity method. In *Advancement of fuzzy theory and systems in China and Japan* (pp. c1.1–c.1.4). International Academic Publishers.

Nguyen, D., & Widrow, B. (1989). The truck backer-upper: An example of self-learning in neural network. In *Proceedings of international joint conference on neural networks* (pp. 357–363).

Pomet, J. B. (1992). Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift. *Systems & Control Letters*, *18*(2), 147–158.

Prieur, C., & Astolfi, A. (2002). Robust stabilization of chained systems via hybrid control. In *Proceedings of the 41st IEEE conference on decision and control* (pp. 522–527).

Sampei, M., Tamura, T., Kobayashi, T., & Shibui, N. (1995). Arbitrary path tracking control of articulated vehicles using nonlinear control theory. *IEEE Transactions on Control Systems Technology*, *3*(1), 125–131.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. The MIT Press.

Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics*, *15*(1), 116–132.

Tanaka, K., & Sano, M. (1994). A robust stabilization problem of fuzzy control systems and its application to backing up control of a truck-trailer. *IEEE Transactions on Fuzzy Systems*, *2*(2), 119–133.

Tilbury, D., Murray, R. M., & Sastry, S. S. (1995). Trajectory generation for the N-trailer problem using goursat normal form. *IEEE Transactions on Automatic Control*, *40*(5), 802–819.

Whittaker, A. D., & Cook, D. F. (1995). Counterpropagation neural network for modeling a continuous correlated process. *International Journal of Production Research*, *33*(7), 1901–1910.

Yamakawa, T., & Horio, K. (1999). Self-organizing relationship (SOR) network. *IEICE Transactions on Fundamentals*, *E82-A*, 1674–1678.

Yamakawa, T., Horio, K., & Sonoh, S. (2005). Batch learning of the self-organizing relationship (SOR) network. *Neural Information Processing Letters and Reviews*, *8*(3), 31–38.

Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, *8*, 338–353.

Zadeh, L. A. (1973). Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man and Cybernetics*, *3*(1), 28–44.

Zadeh, L. A. (1989). Knowledge representation in fuzzy logic. *IEEE Transactions on Knowledge and Data Engineering*, *1*, 89–100.

Zupan, J., Novic, M., & Ruisanchez, I. (1997). Kohonen and counterpropagation artificial neural networks in analytical chemistry. *Chemometrics and Intelligent Laboratory Systems*, *38*, 1–23.