Sensors and Materials, Vol. 30, No. 11 (2018) 2499–2516
MYU Tokyo

S & M 1694

Using Interval Type-2 Recurrent Fuzzy Cerebellar Model Articulation Controller Based on Improved Differential Evolution for Cooperative Carrying Control of Mobile Robots

Jyun-Yu Jhang,[1] Cheng-Jian Lin,[2*] Tzu-Chao Lin,[3]
Chao-Chun Chen,[3] and Kuu-Young Young[1]

[1]Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu 300, Taiwan
[2]Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan
[3]Institute of Manufacturing Information and Systems, National Cheng Kung University, Tainan 701, Taiwan

(Received March 28, 2018; accepted October 22, 2018)

Keywords: mobile robot, cooperative carrying, reinforcement learning, differential evolution

In this study, we propose an effective cooperative carrying method for mobile robots in an unknown environment. During the carrying process, the state manager (SM) switches between wall-following carrying (WFC) and toward-goal carrying (TGC) to avoid obstacles and prevent the objects from dropping. An interval type-2 recurrent fuzzy cerebellar model articulation controller (IT2RFCMAC) based on dynamic group differential evolution (DGDE) is proposed for implementing the WFC and TGC of mobile robots. The adaptive wall-following control is developed using the reinforcement learning strategy to realize cooperative carrying control for mobile robots. The experimental results indicated that the proposed DGDE is superior to other algorithms and can complete the cooperative carrying of mobile robots to reach the goal location.

1. Introduction

In recent years, mobile robot navigation control and obstacle avoidance have become important research fields.[1–3] During the navigation process in an unknown environment, a robot has to avoid colliding with obstacles and move towards a goal. In the cooperative carrying process by multiple robots, the mobile robots assist each other to prevent an object from dropping and colliding until they successfully complete the cooperative carrying task. Therefore, developing the appropriate decision for mobile robots to avoid obstacles is an important topic. Zhao designed a fuzzy controller using infrared sensors to keep robots away from obstacles.[4] Gavrilut and Tiponut proposed an expert fuzzy system to replace the complex mathematical model and define the fuzzy rules, but specialists in the field are required to establish fuzzy rules on the basis of their expertise.[5] Zhu and Yang combined fuzzy logic with artificial neural networks (ANNs) and adjusted the controller parameters through a neural

*Corresponding author: e-mail: cjlin@ncut.edu.tw
https://doi.org/10.18494/SAM.2018.2052

ISSN 0914-4935

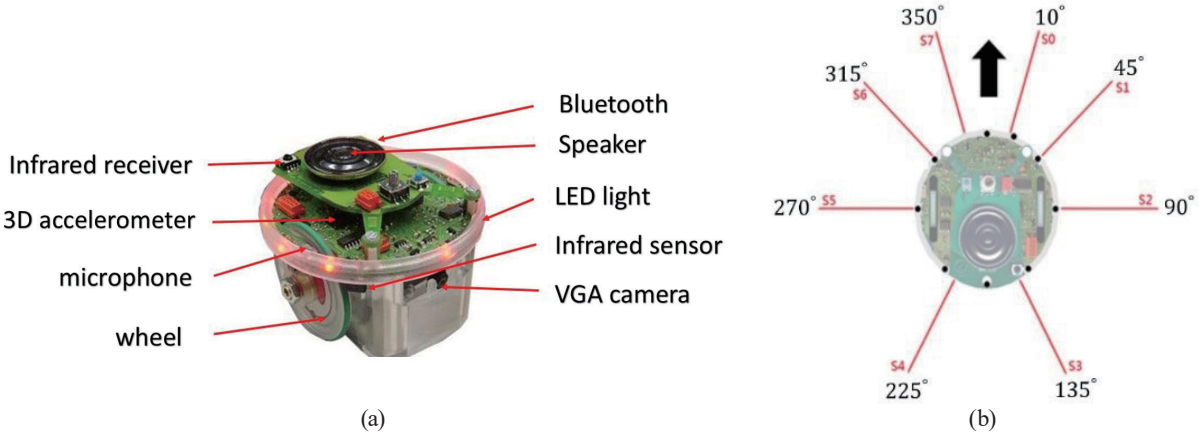
fuzzy network so that the mobile robot can complete a navigation task.[6] These methods only consider the navigation of a single mobile robot.

In cooperative carrying by robots, scholars also use various methods to design the controller.[7–11] Chen *et al.* proposed a strategy for transporting an object to a goal using multiple mobile robots.[7–9] Sakuyama *et al.* presented a methodology for the transport of a large object by mobile robots using small hand carts.[9] The object was placed on top of the mobile robots that carry an object toward a destination. Pereira *et al.* divided the robots into the leader and the follower to complete the cooperative carrying.[10] However, when robots encounter obstacles, it must turn back to move in another direction, which reduces the efficiency of navigation. Yamashita *et al.* adopted a path planning method to calculate the optimized route in a known environment.[11] In a real environment, the input signal contains uncertainties due to noise interferences from the sensors. Therefore, Baklouti *et al.* used an interval type-2 fuzzy neural network for solving uncertain problems.[12]

In this study, we present an effective cooperative carrying and navigation control method for mobile robots in an unknown environment. A state manager switches between two behavioral control modes, wall-following carrying (WFC) and toward-goal carrying (TGC), based on the relationship between the mobile robot and the unknown environment. In order to design a robust controller with antinoise capability, an efficient interval type-2 recurrent fuzzy cerebellar model articulation controller (IT2RFCMAC) based on dynamic group differential evolution (DGDE) is proposed to realize the carrying control and WFC for mobile robots. The experimental results demonstrate that the proposed method can enable the mobile robots to complete the task of cooperative carrying.

2. Description of Mobile Robot

The e-puck mobile robot developed by Ecole Polytechnique Fédérale de Lausanne was adopted in this study, as shown in Fig. 1(a). The mobile robot has been applied to various studies, such as signal processing, robot control, swarm intelligence, coordinated motion, and human–computer interaction.



(a) (b)

Fig. 1. (Color online) The e-puck mobile robot architecture.

The e-puck is a two-wheeled mobile robot with an axle diameter of 4 cm, a height of 5 cm, and a maximum speed of 15 cm per second. The robot contains 8 infrared sensors S_0–S_7 that are distributed around the robot, as shown in Fig. 1(b). The sensors on the right side of the robot are mounted at 10, 45, 90, and 135°. Each infrared sensor can detect a distance between 1 and 5 cm.

3. Wall-following Control of Mobile Robot

In this section, the proposed IT2RFCMAC based on DGDE is demonstrated to realize wall-following control. The DGDE algorithm is used to adjust the parameters of the IT2RFCMAC during the learning process.

3.1 IT2RFCMAC

Recently, ANNs have been widely applied to various fields.[13–19] Albus proposed the cerebellar model articulation controller (CMAC) in 1975.[20] The CMAC architecture mimics the human cerebellum and has the advantages of reliable learning ability, simple structure, and easy realization.

Figure 2 shows the structure of IT2RFCMAC. X_n represents the input of IT2RFCMAC, whereas Y_L and Y_R represent the left and right wheel speeds of the robot, respectively. To reduce the computational complexity during defuzzification, in this study, we adopted the centers of sets (COS) to implement the reduction process.[21] The fuzzy if-then rule can be expressed as

$$\begin{aligned} &\text{Rule } j: \text{IF } X_1 \text{ is } \tilde{A}_{1,j} \text{ and } X_2 \text{ is } \tilde{A}_{2,j} \text{ and } \ldots \text{ and } X_n \text{ is } \tilde{A}_{n,j}, \\ &\text{THEN } y = a_j^0 + \sum_{i=1}^{n} a_{i,j} X_i^{(1)}, \end{aligned} \tag{1}$$

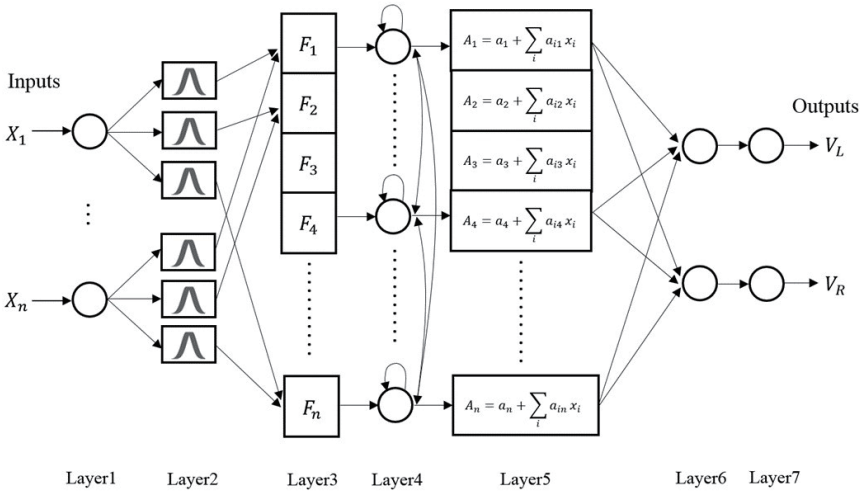


Fig. 2. Structure of an IT2RFCMAC.

where j is the rule number of the fuzzy hypercube, X_i denotes the ith input, $\tilde{A}_{i,j}$ represents the interval type-2 fuzzy set, and y denotes a Takagi–Sugeno–Kang (TSK) linear function in the consequent layer.

The operations of the seven-layer IT2RFCMAC are described as follows.

Layer 1 (input layer): The input data set $X = [X_1, X_2, \ldots, X_n]$ is expressed as

$$Input_i^{(1)} = X_i \text{, and } i = 1, 2, \ldots, n. \tag{2}$$

Layer 2 (Gaussian membership function layer): The input data is converted into fuzzy linguistic information in this layer, which is called fuzzification operation. Each node is defined into an interval type-2 fuzzy set. The Gaussian primary membership function is composed of an uncertain mean $[m^1, m^2]$ and standard deviation (σ). The membership degree of the Gaussian primary membership function ($\mu_{\tilde{A}}$) is expressed as the upper bound $\overline{\mu}_{\tilde{A}}$ and the lower bound $\underline{\mu}_{\tilde{A}}$ and defined as

$$\mu_{\tilde{A}_{i,j}}^{(2)} = exp\left(-\frac{1}{2} \frac{\left[Input_i^{(1)} - m_{i,j} \right]^2}{\left(\sigma_{i,j} \right)^2} \right),\ N\left(m_{i,j}, \sigma_{i,j}; Input_i^{(1)} \right),\ m_{i,j} \in \left[m_{i,j}^1, m_{i,j}^2 \right], \tag{3}$$

$$\overline{\mu}_{\tilde{A}_{i,j}}^{(2)} = \begin{cases} N\left(m_{i,j}^1, \sigma_{i,j}; Input_i^{(1)} \right), & \text{if } Input_i^{(1)} < m_{i,j}^1, \\ 1, & \text{if } m_{i,j}^1 \le Input_i^{(1)} \le m_{i,j}^2, \\ N\left(m_{i,j}^2, \sigma_{i,j}; Input_i^{(1)} \right), & \text{if } Input_i^{(1)} > m_{i,j}^2, \end{cases} \tag{4}$$

and

$$\underline{\mu}_{\tilde{A}_{i,j}}^{(2)} = \begin{cases} N\left(m_{i,j}^2,\ \sigma_{i,j};\ Input_i^{(1)} \right), & \text{if } Input_i^{(1)} \le \dfrac{m_{i,j}^1 + m_{i,j}^2}{2}. \\ N\left(m_{i,j}^1,\ \sigma_{i,j};\ Input_i^{(1)} \right), & \text{if } Input_i^{(1)} > \dfrac{m_{i,j}^1 + m_{i,j}^2}{2}. \end{cases} \tag{5}$$

The output of each node is an interval $[\overline{\mu}_{\tilde{A}_{i,j}}^{(2)}, \underline{\mu}_{\tilde{A}_{i,j}}^{(2)}]$.

Layer 3 (firing layer): Each node is a fuzzy hypercube and uses an algebraic product operation to calculate the firing strengths $F_j^{(3)}$, and is defined as

$$F_j^{(3)} = \left[\overline{f}_j^{(3)}, \underline{f}_j^{(3)} \right],\ j = 1, 2, \ldots, M, \tag{6}$$

$$\overline{f}_j^{(3)} = \prod_{i=1}^{n} \overline{\mu}_{A_{i,j}}^{(2)},\ \underline{f}_j^{(3)} = \prod_{i=1}^{n} \underline{\mu}_{A_{i,j}}^{(2)}, \tag{7}$$

where $\overline{f}_j^{(3)}$ and $\underline{f}_j^{(3)}$ represent the firing strengths of the fuzzy hypercube's upper bound and lower bound, respectively.

Layer 4 (recurrent layer): In this layer, feedback connections are added to embed temporal relations in the network. The output $O_j^{(4)}$ is combined with the current firing strength and all the previous fuzzy hypercubes, and is expressed as

$$O_j^{(4)} = \left[\overline{o}_j^{(4)}, \underline{o}_j^{(4)} \right], \tag{8}$$

where

$$\begin{aligned} \overline{o}_j^{(4)} &= \left[\sum_{k=1} \left(\lambda_{k,j}^q \times \overline{o}_k^{(4)}(t-1) \right) \right] + \left(1 - r_j^q\right) \times \overline{f}_j^{(3)}, \\ \underline{o}_j^{(4)} &= \left[\sum_{k=1} \left(\lambda_{k,j}^q \times \underline{o}_k^{(4)}(t-1) \right) \right] + \left(1 - r_j^q\right) \times \underline{f}_j^{(3)}, \end{aligned} \tag{9}$$

where $r_j^q = \sum_{k=1}^{M} \lambda_{k,j}^q$ and $\lambda_{k,j}^q = \frac{R_{k,j}^q}{M} \left(0 \le R_{k,j}^q \le 1\right)$ are the recurrent weights of the current and previous firing strengths of each rule, respectively. M is the number of fuzzy hypercubes, $\lambda_{k,j}^q$ represents the random number of recurrent weight between [0,1], and $\overline{o}_k^{(4)}(t-1)$ and $\underline{o}_k^{(4)}(t-1)$ represent the upper bound and lower bound of the previous output $O_j^{(4)}$, respectively.

Layer 5 (consequent layer): This layer adopts the TSK function instead of the traditional fuzzy inference of the consequent, and the output in layer 5 is defined as

$$A_j^{(5)} = \left[\overline{A}_j^{(5)}, \underline{A}_j^{(5)} \right], \tag{10}$$

where

$$\begin{aligned} \overline{A}_j^{(5)} &= a_j^0 + \sum_{i=1}^{n} a_{i,j} \times Input_i^{(1)}, \\ \underline{A}_j^{(5)} &= a_j^0 + \sum_{i=1}^{n} a_{i,j} \times Input_i^{(1)}, \end{aligned} \tag{11}$$

where a_j^0 and $a_{i,j}$ represent a constant of the TSK linear function weight, and the output $A_j^{(5)}$ is

expressed as the upper bound $\bar{A}_j^{(5)}$ and the lower bound $\underline{A}_j^{(5)}$.

Layer 6 (defuzzification layer): The traditional type-2 order reduction method is a highly complex calculation. Thus, the type-2 fuzzy sets are converted to type-1 fuzzy sets by the type-reduction[21] method. The crisp output value $[y_r^{(6)}, y_l^{(6)}]$ is obtained using a center-of-gravity defuzzification method and is described as

$$y_r^{(6)} = \frac{\sum_{j=1}^{M} \bar{o}_j^{(4)} \times \bar{A}_j^{(5)}}{\sum_{j=1}^{M} \bar{o}_j^{(4)}}, \quad y_l^{(6)} = \frac{\sum_{j=1}^{M} \underline{o}_j^{(4)} \times \underline{A}_j^{(5)}}{\sum_{j=1}^{M} \underline{o}_j^{(4)}}. \tag{12}$$

Layer 7 (output processing layer): The output is defuzzified by computing the average of $y_r^{(6)}$ and $y_l^{(6)}$, and the crisp value of $y^{(7)}$ is obtained as

$$y^{(7)} = \frac{y_r^{(6)} + y_l^{(6)}}{2}. \tag{13}$$

3.2 Proposed DGDE algorithm

Evolutionary algorithms can solve optimization problems by imitating some aspects of natural evolution, such as ant colony optimization (ACO),[22] particle swarm optimization (PSO),[23] differential evolution (DE),[24] and the artificial bee colony (ABC) algorithm.[25] The traditional DE method has the advantages of reduced parameters, fast convergence, and global optimum search ability, and has been applied in various fields successfully.[26–30] However, it still has several disadvantages, such as poor accuracy and becoming easily trapped in a local optimal solution. To eliminate this disadvantage, an efficient DGDE algorithm is proposed to overcome the shortcomings of traditional DE in this study. The steps of DGDE are described in detail below.

Step 1 initialization and coding: All the parameters of each IT2RFCMAC are coded into one vector. The coding format is presented in Fig. 3. The adjustable parameters in each fuzzy hypercube consist of a Gaussian uncertainty mean ($m^1_{i,j}$), standard deviation ($\sigma_{i,j}$), displacement

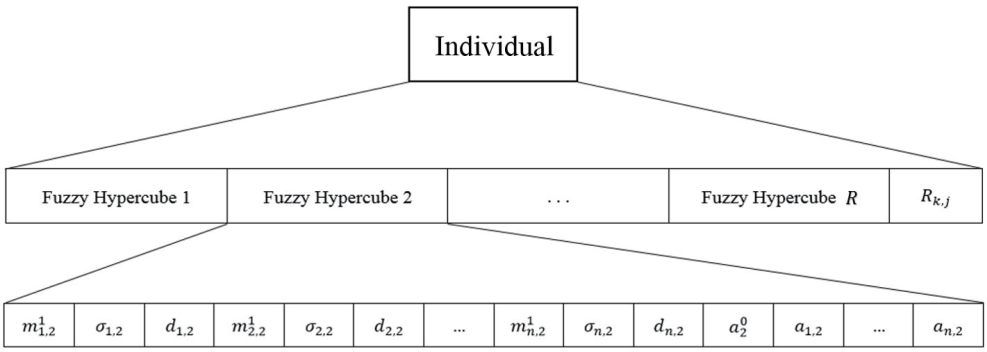


Fig. 3. Coding format of the vector.

value of the uncertainty mean ($d_{i,j}$), recurrent weight ($R_{k,j}$), and TSK linear function weights (a^0_j and $a_{i,j}$).

Step 2 ranking the fitness values of all vectors: When the fitness value of each vector X_i is calculated, it is sorted according to the fitness value from the best to the worst. The initial group number of all vectors is set to zero.

Step 3 vector group: The vector with the highest fitness value is set as the new group leader, and the group number is updated to one. Then, the average distance difference and the average fitness difference between these ungrouped particles and the group leader in group number zero are calculated. The definition is

$$DIS^g = \sum_{i=1}^{NP}\sum_{j=1}^{D}\sqrt{\left(L_j^g - X_j^i\right)^2}\ ,\ \text{if } X^i\, Group = 0\ , \tag{14}$$

$$Average\_Distance\left(ADIS^g\right) = \frac{DIS^g}{NI}\ , \tag{15}$$

$$FIT^g = \sum_{i=1}^{NP}\left|Fit\left(L^g\right) - Fit\left(X^i\right)\right|\ ,\ \text{if } X^i\, Group = 0\ , \tag{16}$$

$$Average\_Fitness\left(AFIT^g\right) = \frac{FIT^g}{NI}\ , \tag{17}$$

where NP represents the number of parameter vectors, D represents the encoded dimension, $L^g{}_j$ denotes the jth dimension of the gth group leader, NI is the total number of ungrouped vectors (group number 0), and $ADIS^g$ and $AFIT^g$ represent the distance threshold value and fitness threshold value, respectively, of the gth group.

The distance difference (Dis^i) and fitness difference (Fit^i) are calculated to determine whether the ungrouped vectors are similar to the leader vectors.

$$Dis^i = \sum_{j=1}^{D}\sqrt{\left(L_j^g - X_j^i\right)^2} \tag{18}$$

$$Fit^i = \left|Fit\left(L^g\right) - Fit\left(X^i\right)\right| \tag{19}$$

If the conditions $Dis^i < ADIS^g$ and $Fit^i < AFIT^g$ are satisfied for a vector, then the vector is similar to the gth group leader. These vectors are grouped and the group number is updated to g.

If any ungrouped particles exist, repeat Steps 1 to 3. The ungrouped vector with the highest fitness value is set as the group leader in a new group (i.e., the 2nd group leader). The grouping process is completed when no ungroup vectors exist.

Step 4 mutation: In this study, a dynamic grouping strategy and a new mutation method are proposed for improving the traditional DE algorithms. The modified formula of mutation is expressed as

$$U_{i,G+1} = X_{best,G} + F\left(X_{rL,G} - X_{r1,G}\right) + F\left(X_{r2,G} - X_{r3,G}\right), \tag{20}$$

where F is the mutation weight factor, $X_{best,G}$ is the best fitness vector, and $X_{rL,G}$ is a random leader selected from all the group leaders.

The traditional DE method is easily trapped into a local optimum. Therefore, the random leader as the base vector is proposed to effectively increase the search ability. The mutated vector in Eq. (20) revolves around the best vector and enhances the search ability in the solution space.

Steps 5 and 6 are recombination and selection, respectively, which are the same as those in the traditional DE method. The pseudocode of DGDE is as shown in Fig. 4.

3.3 Wall-following control of mobile robots

A reinforcement learning strategy is utilized by the IT2RFCMAC to realize wall-following control for the mobile robot. The IT2RFCMAC has four inputs (S_0, S_1, S_2, and S_3) and two outputs. The input S_i is the distance that is measured by the infrared sensor. The outputs are the rotational speeds V_L and V_R of the two wheels. The block diagram for the wall-following control of the mobile robot is as shown in Fig. 5.

Start

Step 1 Initialization;

Step 2 (Optimization via Dynamic Group Differential Evolution):

repeat

2.1 Calculate average distance ($ADIS^g$) and average fitness ($AFIT^g$)

$Average\_Distance(ADIS^g) = \frac{DIS^g}{NI}$, $Average\_Fitness\ (AFIT^g) = \frac{FIT^g}{NI}$

2.2 Calculate the distance (Dis^i) and fitness (Fit^i) between individual and leader in each group:

$Dis^i = \sum_{j=1}^{D} \sqrt{\left(L_j^g - X_j^i\right)^2}$, $Fit^i = \left|Fit(L^g) - Fit(X^i)\right|$

If($Dis^i < ADIS^g$ and $Fit^i < AFIT^g$)

Classify into the same group and update the group number as g;

2.3 Mutation:

$U_{i,G+1} = X_{best,G} + F\left(X_{rL,G} - X_{r1,G}\right) + F\left(X_{r2,G} - X_{r3,G}\right)$;

2.4 Recombination;

2.5 Selection;

Until Generation over;

End

Fig. 4. Pseudocode of DGDE.

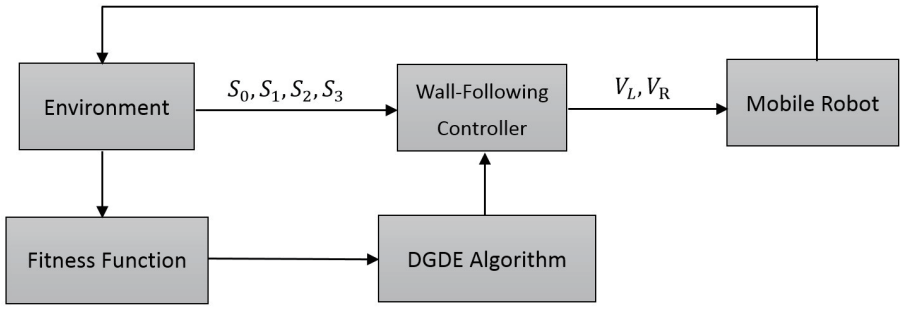


Fig. 5. Mobile robot wall-following control block diagram.

Figure 6 presents the 1.7 × 1.4 m^2 training environment. In order to allow the mobile robots to experience various environments, the training environment consists of straight lines, corners, right-angled corners, and a U curve.

To avoid collision with obstacles and deviation from the wall during the wall-following control learning process, three terminal conditions are adopted for wall-following control learning.

1. A larger total moving distance of the mobile robot than the maximum permitted distance of the training environment indicates that the mobile robot successfully moved in a circular path in an unknown environment.
2. The mobile robot is defined as having collided with the wall when the measured distance from any infrared sensor is less than D_1 (i.e., D_1 is set to 1 cm).
3. The mobile robot is defined as having deviated from the wall when the measured distance of the sensor S_2 is greater than D_2 (i.e., D_2 is set to 1 cm).

In this study, a fitness function that is a combination of three subfitness functions is proposed to evaluate the performance of mobile robot wall-following control. The three subfitness functions are for the total moving distance (SF_1), the distance between the robot and the wall (SF_2), and the degree of parallelism between the robot and the wall (SF_3).

(1) SF_1: When the robot moving distance $R_{distance}$ is greater than the default value R_{stop}, set $R_{distance} = R_{stop}$. This indicates that the robot has successfully moved in a circular path in the training environment.

$$SF_1 = R_{stop} - R_{distance} \tag{21}$$

(2) SF_2: To maintain a fixed distance between the robot and the wall in the wall-following process, SF_2 is defined as the average of $W_d(t)$, where $W_d(t)$ denotes the distance between the robot and the wall at each time step.

$$W_d(t) = \left| S_2(t) - d_{wall} \right| \tag{22}$$

$$SF_2 = \frac{\sum_{t=1}^{T_{stop}} W_d(t)}{T_{stop}} \tag{23}$$

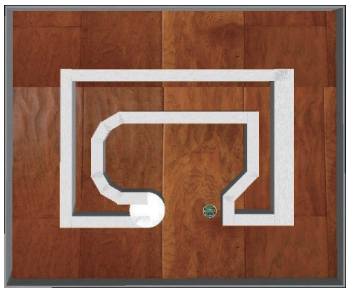


Fig. 6. (Color online) Training environment of mobile robot wall-following control.

(3) SF_3: According to the law of cosines, $x(t)$ is equal to RS_2 when the robot sensor S_2 is parallel to the wall, and the angle between the robot and the wall is 90°, as shown in Fig. 7.

$$RS_1 = r + D_1\,, \quad RS_2 = r + D_2 \tag{24}$$

$$x(t) = \sqrt{RS_1{}^2 + RS_2{}^2 - 2RS_1RS_2\cos(45°)} \tag{25}$$

Here, r is the radius of the robot, and D_1 and D_2 denote the distances between the sensors S_1 and S_2. SF_3 is defined as

$$SF_3 = \frac{\sum_{t=1}^{T_{stop}} \left| RS_2 - x(t) \right|}{T_{stop}}\,. \tag{26}$$

The fitness function for evaluating the wall-following learning performance is a combination of the three subfitness functions (SF_1, SF_2, and SF_3) and is defined as

$$F(\bullet) = \frac{1}{1 + (SF_1 + SF_2 + SF_3)}\,. \tag{27}$$

3.4 Experimental results of wall-following control

In this section, the performance of the proposed DGDE method is compared with those of other evolutionary algorithms. The initialization parameters of the DGDE algorithm consist of the number of vectors (NP), crossover rate (CR), mutation weighting factor (F), generation, and number of fuzzy hypercubes, as presented in Table 1.

Moreover, different numbers of fuzzy rules are considered in the performance evaluation. The IT2RFCMAC with six fuzzy hypercubes was more efficient than that with five or seven fuzzy hypercubes, as shown in Table 2. The number of successful runs represents the number of times that the robot moved successfully in a circular path in the training environment.

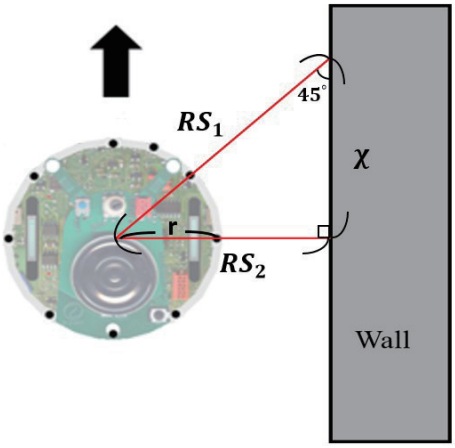


Fig. 7. (Color online) Definition the degree of parallelism.

Table 1
Initial parameters of DGDE.

NP	*CR*	*F*	Generation	Hypercubes
30	0.9	0.5	3000	5, 6, and 7

Table 2
Performance with different numbers of fuzzy hypercubes.

	Number of hypercubes		
Fitness value	5	6	7
Best	0.920	0.925	0.912
Worst	0.853	0.868	0.837
Average	0.889	0.906	0.873
STD	0.017	0.015	0.018
Number of successful runs	10	10	10

Table 3
Performances of various algorithms.

	Fitness value				Number of successful runs
	Best	Worst	Average	STD	
DGDE	0.925	0.868	0.906	0.015	10
DE[24]	0.889	0.835	0.868	0.014	8
JADE[29]	0.914	0.861	0.889	0.013	10
Rank-DE[30]	0.910	0.857	0.878	0.012	10
ABC[25]	0.824	0.774	0.803	0.016	7

Table 3 shows the performance of different algorithms. The proposed DGDE achieved superior fitness values and more successful runs compared with other algorithms. The paths of the robot moving with wall-following control using various evolutionary algorithms are shown in Fig. 8.

4. Cooperative Carrying by Multi-evolutionary Mobile Robots

In this section, we introduce the method of cooperative carrying for mobile robots. The training environment consists of a leader robot and a follower robot, the distance between the two robots is set as 15 cm, and a rectangular object is placed on the two robots. During the cooperative carrying process, the leader explores the front environment and the follower assists the leader to achieve obstacle avoidance, as shown in Fig. 9.

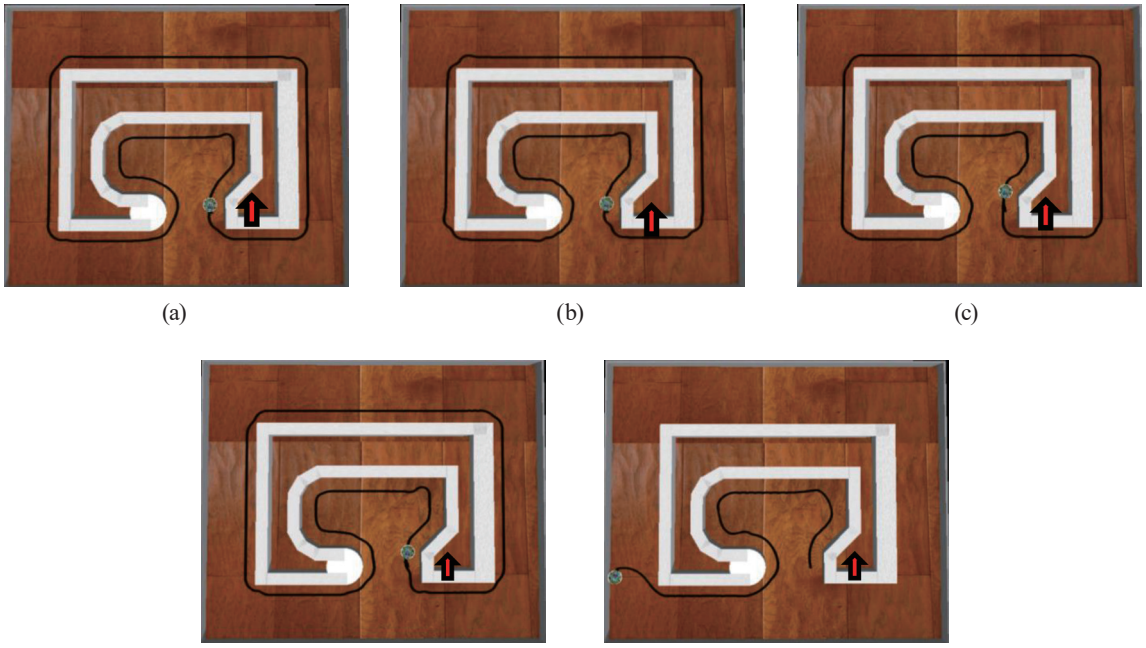


Fig. 8. (Color online) Paths of the robot moving in the training environment: (a) DGDE, (b) DE, (c) JADE, (d) Rank-DE, and (e) ABC.

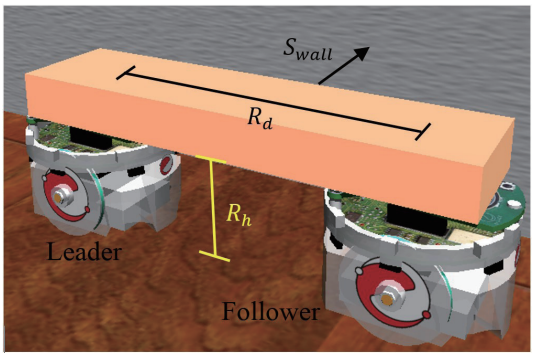


Fig. 9. (Color online) Cooperative carrying by two mobile robots.

4.1 Cooperative carrying method of WFC

A dual controller for cooperative carrying by two mobile robots is proposed. A cooperation controller, which contained five input signals and two output signals, was added for the follower robot to learn WFC. The inputs are the distances sensed by the follower robot's sensors (S_0, S_1, S_2, and S_3) and the distance R_d between two robots. The outputs are the rotational speeds V_L and V_R of the two wheels. Figure 10 presents the block diagram of cooperative carrying by two mobile robots.

The training environment was established with straight lines, smooth curves, continuous curves, and U-shaped curves, to train the follower's cooperation controller. Figure 11 shows the 1.7×1.4 m^2 training environment.

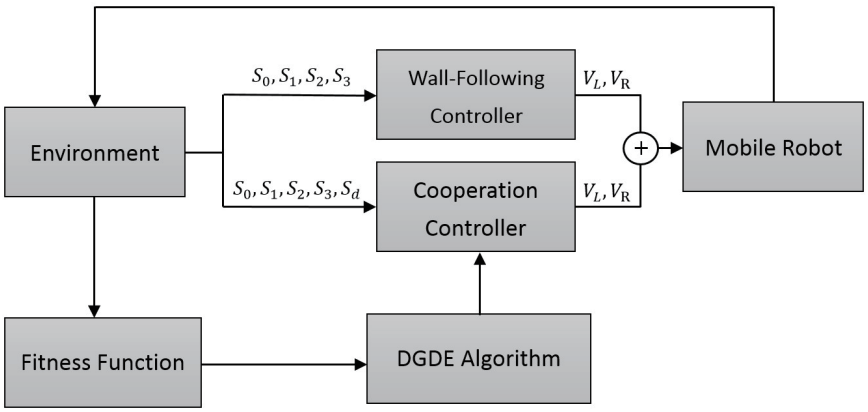


Fig. 10. Structural diagram of cooperative carrying method.

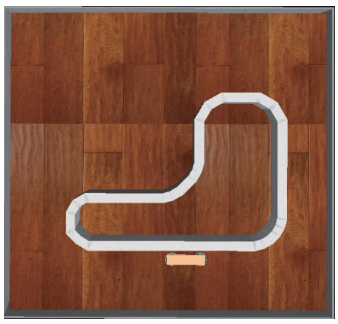


Fig. 11. (Color online) Training environment for cooperative carrying.

During the cooperative carrying learning process, five terminal conditions are proposed to prevent collision with obstacles and objects being dropped.

(1) When any of the follower's sensors detects a distance of less than WD_1 (where WD_1 is equal to 1 cm), it indicates that the follower robot has collided with the wall.

(2) When the distance detected by the follower's sensor S_2 is greater than WD_2, it shows that the follower robot deviates from the wall.

(3) When the measured distance between the leader robot and the follower robot (R_d) is greater than WD_3 or less than WD_4 (where WD_3 and WD_4 are equal to 20 and 10 cm, respectively), the leader and follower robots are deduced to be too far apart or too close together, respectively.

(4) When the measured distance of the robot sensor is less than the height R_h (where R_h is equal to 5 cm), the object has been dropped by the robots.

(5) When the measured distance S_{wall} of the sensor is less than WD_5 or greater than WD_6 (where WD_5 and WD_6 are equal to 1 and 7.5 cm, respectively), it indicates that the object has approached the wall or deviated from the wall.

Cooperative carrying by the robots fails during the training and learning process when the aforementioned conditions are satisfied. A fitness function is proposed to evaluate the efficiency of the cooperative carrying process.

$$F(\bullet) = T_{step} / S_{stop} \tag{28}$$

4.2 Experimental results of the wall-following carrying

The performances of WFC control using our DGDE method and using other methods were compared. Each method was evaluated 10 times to verify the stability of each algorithm in this study. The initial parameters of WFC control using the DGDE algorithm are presented in Table 4.

Table 5 shows the performance evaluations of different algorithms. The proposed WFC control using the DGDE method achieved superior fitness values and more successful runs compared with other methods. In this study, a training environment was created to verify the WFC control performance of different learning algorithms and the path of the robot moving with WFC control using various evolutionary algorithms, as shown in Fig. 12.

4.3 Experimental results of cooperative carrying control

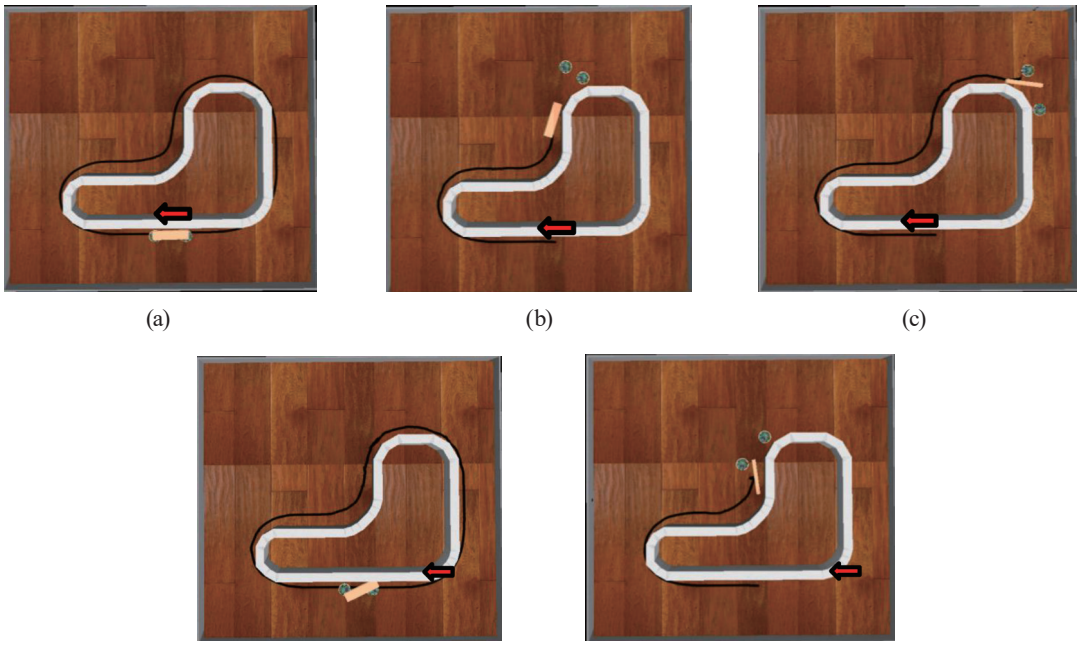
The proposed method is used to verify the performance of navigation control. Two different test environments were created to test whether the robots successfully complete cooperative carrying and navigation control. Experimental results for the two test environments are shown in Fig. 13. The effectiveness of cooperative carrying control was evaluated on the basis of the average distance (*RD*) between the two robots and the average distance (*AD*) between the follower robot and the wall. If *RD* is large, the two robots do not remain apart at a suitable distance during cooperative carrying control, and the object falls easily. On the other hand, if *AD* is small or large, it means that the robots pass the curves with poor efficiency and the object falls easily. Performance evaluation results of cooperative carrying control are shows in Table 6. The experimental results show that the proposed cooperative carrying control method enables successful completion of navigation control in an unknown environments.

Table 4
Initial parameters of WFC control.

NP	*CR*	*F*	Generation	Hypercubes
30	0.9	0.5	2000	6

Table 5
Fitness value of various algorithms in the test environment.

	Fitness value				Number of successes
	Best	Worst	Average	STD	
DGDE	0.859	0.735	0.813	0.028	10
DE[24]	0.438	0.221	0.358	0.046	5
JADE[29]	0.707	0.529	0.590	0.038	8
Rank-DE[30]	0.721	0.553	0.643	0.036	8
ABC[25]	0.399	0.242	0.322	0.054	4



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

Fig. 12. (Color online) Paths of moving robot with WFC control in the training environment: (a) DGDE, (b) DE, (c) JADE, (d) Rank-DE, and (e) ABC.

(a) (b)

Fig. 13. (Color online) Navigation control of cooperative carrying in (a) test environment 1 and (b) test environment 2.

Table 6
Performance evaluation results of cooperative carrying control.

Algorithm	Evaluation items			
	Test environment 1		Test environment 2	
	RD (cm)	*AD* (cm)	*RD* (cm)	*AD* (cm)
DGDE	15.73	3.78	16.23	3.68

5. Conclusion

We proposed an effective IT2RFCMAC controller for cooperative carrying control in an unknown environment. The parameters of IT2RFCMAC are trained in an unknown environment through reinforcement learning. The proposed DGDE learning algorithm

uses dynamic grouping and local search methods to improve the convergence stability of the traditional DE method. In addition, the proposed state manager automatically switches between WFC and TGC during navigation control in cooperative carrying. Experimental results demonstrated that the control performance of the proposed method is superior to those of the other methods in WFC, and cooperative carrying control of mobile robots in unknown environments was successfully accomplished.

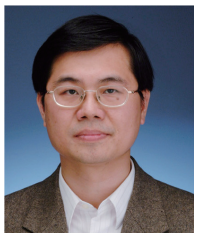
References

1 C. F. Juang and Y. C. Chang: IEEE Trans. Fuzzy Syst. **19** (2011) 379. https://doi.org/10.1109/TFUZZ.2011.2104364
2 Z. D. Wang, E. Nakano, and T. Takahashi: IEEE Trans. Syst. Man Cybern. Part A Syst. Humans **33** (2003) 537. https://doi.org/10.1109/TSMCA.2003.817396
3 R. Jarvis: 2008 IEEE Conf. Robotics, Automation and Mechatronics (IEEE, 2008) 146–149. https://doi.org/10.1109/RAMECH.2008.4681359
4 R. Zhao: 2015 Int. Automatic Control Conf. (CACS, 2015) 19–24. https://doi.org/10.1109/CACS.2015.7378359
5 I. Gavrilut and V. Tiponut: 12th WSEAS Int. Conf. Systems (WSEAS, 2008) 205–209.
6 A. Zhu and S. X. Yang: IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **37** (2007) 610. https://doi.org/10.1109/TSMCC.2007.897499
7 J. Chen, M. Gauci, and R. Grob: 2013 IEEE Int. Conf. Robotics and Automation (IEEE, 2013) 863–869. https://doi.org/10.1109/ICRA.2013.6630674
8 J. Chen, M. Gauci, W. Li, A. Kolling, and R. Grob: IEEE Trans. Rob. **31** (2015) 307. https://doi.org/10.1109/TRO.2015.2400731
9 T. Sakuyama, J. Figueroa, Y. Miyazaki, and J. Ota: 2012 IEEE Int. Conf. Robotics and Biomimetics (IEEE, 2012) 2108–2113. https://doi.org/10.1109/ROBIO.2012.6491280
10 G. A. S. Pereira, B. S. Pimentel, L. Chaimowicz, and M. F. M. Campos: Proc. 2002 IEEE Int. Conf. Robotics and Automation (IEEE, 2002) 281–286. https://doi.org/10.1109/ROBOT.2002.1013374
11 A. Yamashita, T. Arai, J. Ota, and H. Asama: IEEE Trans. Rob. Autom. **19** (2003) 223. https://doi.org/10.1109/TRA.2003.809592
12 N. Baklouti and A. M. Alimi: 2017 Int. Conf. Advanced Systems and Electric Technologies (IC_ASET, 2017) 481–486. https://doi.org/10.1109/ASET.2017.7983740
13 J. G. Wang, S. C. Tai, and C. J. Lin: Neural Comput. Appl. **29** (2018) 201. https://doi.org/10.1007/s00521-016-2551-x
14 J. G. Wang, S. C. Tai, and, C. J. Lin: Opt. Eng. **55** (2016) 083104. https://doi.org/10.1117/1.OE.55.8.083104
15 C. J. Lin and H. Y. Lin: Int. J. Adv. Rob. Syst. **14** (2017) 1. https://doi.org/10.1177/1729881417720872
16 S. Sunphorka, B. Chalermsinsuwan, and P. Piumsomboon: J. Energy Inst. **90** (2017) 51. https://doi.org/10.1016/j.joei.2015.10.007
17 M. Duguleana and G. Mogan: Expert Syst. Appl. **62** (2016) 104. https://doi.org/10.1016/j.eswa.2016.06.021
18 Y. C. Hsu and S. F. Lin: IFAC Proc. **41** (2008) 6530–6535. https://doi.org/10.3182/20080706-5-KR-1001.01101
19 H. W. Peng, S. F. Wu, C. C. Wei, and S. J. Lee: Appl. Soft Comput. **32** (2015) 481. https://doi.org/10.1016/j.asoc.2015.03.059
20 J. S. Albus: J. Dyn. Syst. Meas. Contr. **97** (1975) 220. https://doi.org/10.1115/1.3426922
21 O. Castillo and P. Melin: Appl. Soft Comput. **12** (2012) 1267. https://doi.org/10.1016/j.asoc.2011.12.010
22 M. Mavrovouniotis and S. Yang: Appl. Soft Comput. **13** (2013) 4023. https://doi.org/10.1016/j.asoc.2013.05.022
23 F. Marini and B. Walczak: Chemom. Intell. Lab. Syst. **149** (2015) 153. https://doi.org/10.1016/j.chemolab.2015.08.020
24 S. Das, S. S. Mullick, and P. N. Suganthan: Swarm Evol. Comput. **27** (2016) 1. https://doi.org/10.1016/j.swevo.2016.01.004
25 W. F. Gao, L. L. Huang, S. Y. Liu, and C. Dai: IEEE Trans. Cybern. **45** (2015) 2827. https://doi.org/10.1109/TCYB.2014.2387067
26 S. Das and P. N. Suganthan: IEEE Trans. Evol. Comput. **15** (2010) 4. https://doi.org/10.1109/TEVC.2010.2059031
27 N. L. Tsakiridis, J. B. Theocharis and G. C. Zalidis: 2017 IEEE Int. Conf. Fuzzy Systems (IEEE, 2017) 1–7. https://doi.org/10.1109/FUZZ-IEEE.2017.8015563

28 X. Xue, Z. Dong, W. Xiang, and J. Lu: 2012 9th Int. Conf. Fuzzy Systems and Knowledge Discovery (IEEE, 2012) 409–413. https://doi.org/10.1109/FSKD.2012.6233937
29 J. Zhang and A. C. Sanderson: IEEE Trans. Evol. Comput. **13** (2009) 945. https://doi.org/10.1109/TEVC.2009.2014613
30 W. Gong and Z. Cai: IEEE Trans. Cybern. **43** (2013) 2066. https://doi.org/10.1109/TCYB.2013.2239988

About the Authors

Jyun-Yu Jhang received his B.S. and M.S. degrees from the Department of Computer Science and Information Engineering of National Chin-Yi University of Technology, Taichung, Taiwan, in 2013 and 2015, respectively. He is currently pursuing his Ph.D. degree at the Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan. His current research interests include fuzzy logic theory, type-2 neural fuzzy systems, evolutionary computation, machine learning, and computer vision and applications. (o800920@gmail.com)



Cheng-Jian Lin received his Ph.D. degree in electrical and control engineering from National Chiao-Tung University, Taiwan, R.O.C., in 1996. Currently, he is a Distinguished Professor at the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung City, Taiwan, R.O.C. His current research interests are machine learning, pattern recognition, intelligent control, image processing, and evolutionary robot. (cjlin@mail.cyut.edu.tw)

Tzu-Chao Lin received his M.S. degree from the Department of Computer Science and Information Engineering of Chaoyang University of Technology, Taichung, Taiwan, in 2005. He is currently pursuing his Ph.D. degree at the Institute of Manufacturing Information and Systems, National Cheng-Kung University, Taiwan. His research interests include intelligent control, mobile robots, and evolutionary computation.
(polo1004@gmail.com)

Chao-Chun Chen received his Ph.D. degree from the Department of Computer Science and Information Engineering of National Cheng-Kung University, Taiwan, in 2004. Currently, he is an associate professor at the Institute of Manufacturing Information and Systems, National Cheng-Kung University, Taiwan. His research interests include mobile/wireless data management, sensor networks, spatiotemporal databases, and web information retrieval. (chaochun@mail.ncku.edu.tw)

Kuu-Young Young received his B.S. degree in electrical engineering from National Taiwan University, Taiwan, in 1983, and M.S. and Ph.D. degrees in electrical engineering of Northwestern University, Evanston, IL, in 1987 and 1990, respectively. Currently, he is a professor at the Institute of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu, Taiwan. His current research interests are intelligent control, image processing, and robotic control. (kyoung@mail.nctu.edu.tw)