electronics

MDPI

Article

Cooperative Carrying Control for Multi-Evolutionary Mobile Robots in Unknown Environments

Jyun-Yu Jhang [1], Cheng-Jian Lin [2,*] and Kuu-Young Young [1]

[1] Institute of Electrical and Control Engineering, National Chiao Tung University, Hsinchu 300, Taiwan; o800920@gmail.com (J.-Y.J.); kyoung@mail.nctu.edu.tw (K.-Y.Y.)

[2] Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan

* Correspondence: cjlin@ncut.edu.tw

Received: 13 February 2019; Accepted: 1 March 2019; Published: 6 March 2019

Abstract: This study provides an effective cooperative carrying and navigation control method for mobile robots in an unknown environment. The manager mode switches between two behavioral control modes—wall-following mode (WFM) and toward-goal mode (TGM)—based on the relationship between the mobile robot and the unknown environment. An interval type-2 fuzzy neural controller (IT2FNC) based on a dynamic group differential evolution (DGDE) is proposed to realize the carrying control and WFM control for mobile robots. The proposed DGDE uses a hybrid method that involves a group concept and an improved differential evolution to overcome the drawbacks of the traditional differential evolution algorithm. A reinforcement learning strategy was adopted to develop an adaptive WFM control and achieve cooperative carrying control for mobile robots. The experimental results demonstrated that the proposed DGDE is superior to other algorithms at using WFM control. Moreover, the experimental results demonstrate that the proposed method can complete the task of cooperative carrying, and can realize navigation control to enable the robot to reach the target location.

Keywords: evolutionary mobile robot; fuzzy control; navigation control; cooperative carrying; differential evolution

1. Introduction

Mobile robot control has been widely used in several applications, such as navigation, obstacle avoidance, path planning, and cooperative transport. To enhance the robot control quality, Cupertino [1] adopted a fuzzy controller. The fuzzy controller possesses robustness and an anti-noise ability; the controller can identify and calculate signals with uncertainties. However, to design an applicable fuzzy controller, designers must spend a considerable amount of time analyzing the experimental input and output data of a mobile robot. Thus, machine-learning technology has gradually attracted considerable research attention. Zhu and Yang [2] and Rusu et al. [3] used supervised learning methods to adjust the parameters of the if–then rules in fuzzy neural networks by training data. The disadvantages of supervised learning [2,3] are that it is difficult to collect training data in advance and obtain precise training data.

Recently, reinforcement learning [4] has been widely used in control applications for mobile robots. The method not only can automatically construct a complete fuzzy neural network in the absence of precise training data but also adjust the parameters of the system through the machine leaning algorithm to complete the navigation task. Therefore, designing a mobile robot using evolutionary computing in unknown environments has become a topic of interest. Hsu and Juang [5] implemented a wall-following control using reinforcement learning. A fitness function was used to evaluate a

robot's movement in an unknown environment, and an optimal fitness value was adopted as the reinforcement signal. Anish and Parhi [6] used sensors to measure the distance between a robot and an obstacle. In [6], measured distances were used as inputs, and steering angle and speed were used as outputs for the adaptive network-based fuzzy inference system controller. Juang and Chang [7] adopted four sonar sensors and used the data from these sensors as input signals. In this application, the speeds of the left and right wheel are the outputs of the fuzzy neural network controller. The mobile robot was trained in the training environment by adjusting the parameters of the fuzzy neural network controller. Although the aforementioned methods can be used to successfully complete the navigation task, their performances are not optimal. Because the robot is in a real environment, the input signal contains uncertainties due to noise interferences from the sensors.

Although some researchers [8,9] have used the type-1 fuzzy set to solve uncertain problems, the control performance in the real environment is not optimal. Therefore, Kim and Chwa [10] used an interval type-2 fuzzy set for solving uncertain problems. The membership values of fuzzy sets provide a footprint of uncertainty (FOU) to deal with uncertainties. Thus, the controller design is more flexible and has improved error tolerance. Castillo and Melin [11] reduced the computational complexity of an interval type-2 fuzzy system.

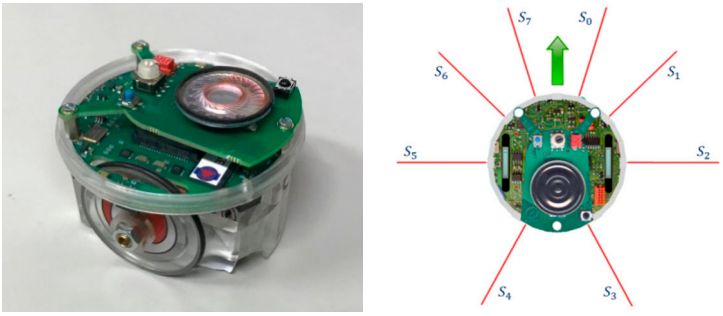
Recently, researchers have used evolutionary algorithms for solving the parameter optimization problem, such as particle swarm optimization (PSO) [12], ant colony optimization (ACO) [13], differential evolution (DE) [14], the artificial bee colony (ABC) algorithm [15], and bacterial foraging optimization (BFO) [16]. In recent years, DE has been widely used in various fields [17,18] and has the advantages of a simple structure, reduced parameter setting requirements, and superior problem solving ability. However, the traditional DE method has a disadvantage, in that it can easily become trapped in a local optimal solution. To eliminate this disadvantage, an improved DE is proposed for solving mobile robot control problems.

The major contributions of this paper are described as follows. First, an efficient interval type-2 fuzzy neural controller (IT2FNC) based on dynamic group differential evolution (DGDE) was designed to implement the carrying control and wall-following mode (WFM) control for mobile robots. In the proposed IT2FNC, a functional link neural network (FLNN) with a nonlinear combination input was added in consequent part of a fuzzy rule. Second, the proposed DGDE used a hybrid method that involves a group concept and an improved DE to overcome the drawbacks of easily trapped into local optimal in the traditional DE algorithm. Second, a manager mode was developed to assist mobile robots in navigation control. Third, the manager mode switched to WFM or toward-goal mode (TGM), based on the relationship between the mobile robot and the unknown environment. Fourth, the proposed control method can complete the task of cooperative carrying and can realize navigation control to enable the robot to reach a target location. Moreover, the experimental results demonstrated that the proposed DGDE learning algorithm is superior to other algorithms at using WFM control.

2. Mobile Robot Specifications

In this study, the e-puck mobile robot developed by the Ecole Polytechnique Fédérale de Lausanne, was adopted, as displayed in Figure 1a. The mobile robot has been widely used in studies, such as in embedded computing, signal processing, inter-robot communication, robot control, feature extraction from images and sounds, swarm intelligence, and cooperative behavior in robotics.

The e-puck mobile robot is a two-wheeled mobile robot with an axle diameter of 4.12 cm and a maximum speed of 15 cm/s. The infrared sensor has a sensing range of approximately 360° and is symmetrical. The sensor performs tasks such as object detection, distance detection, and obstacle avoidance. The sensors S_0–S_3 that are on the right side of the robot are mounted at 10°, 45°, 90°, and 135°, respectively. Each sensor can detect a distance between 1 and 6 cm, as shown in Figure 1b.



(**a**) (**b**)

Figure 1. E-puck mobile robot architecture (**a**) E-puck mobile robot, (**b**) Infrared sensor position.

3. Proposed Type-2 Fuzzy Controller Based on an Evolutionary Algorithm

An IT2FNC was proposed to realize wall-following control. The associated DGDE learning algorithm can be used to adjust the parameters of the IT2FNC.

3.1. Interval Type-2 Fuzzy Neural Controller

In this section, the structure of the IT2FNC is introduced. Figure 2 displays the structure of the IT2FNC. $X_1, \cdots, X_n$ represents the input of IT2FNC, whereas Y_{Left} and Y_{Right} represent the left and right wheel speed of the robot, respectively. To reduce the computational complexity of the order reduction during defuzzification, this study adopted the centers of sets (COS) [11,19] to conduct the order reduction process. A functional link neural network (FLNN) with a nonlinear combination input was added in consequent part of a fuzzy rule [20]. Figure 2 presents the five-layer structure of an IT2FNC. The IT2FNC consists of an input layer, a membership function layer, a firing layer, a consequent layer, and an output layer. The if–then rule can be expressed as follows:

$$\begin{aligned} \text{Rule } j: \text{ IF } x_1 \text{ is } \widetilde{A}_{1j} \text{ and } x_2 \text{ is } \widetilde{A}_{2j} \ldots \text{ and } x_i \text{ is } \widetilde{A}_{ij} \ldots \text{ and } x_n \text{ is } \widetilde{A}_{nj} \\ \text{THEN } y_j = \sum_{k=1}^{M} \omega_{kj} \varnothing_k, \\ = \omega_{1j} \varnothing_1 + \omega_{2j} \varnothing_2 + \ldots + \omega_{Mj} \varnothing_M, \end{aligned} \tag{1}$$

where n is the number of inputs, x_i represents the ith input, y_j denotes the output of the jth fuzzy rule, $\widetilde{A}_{1j}, \widetilde{A}_{2j}, \ldots, \widetilde{A}_{nj}$ represent the interval type-2 fuzzy sets, ω_{kj} is the link weight, $\varnothing_k$ represents the basis trigonometric function, and M is the number of basis functions.

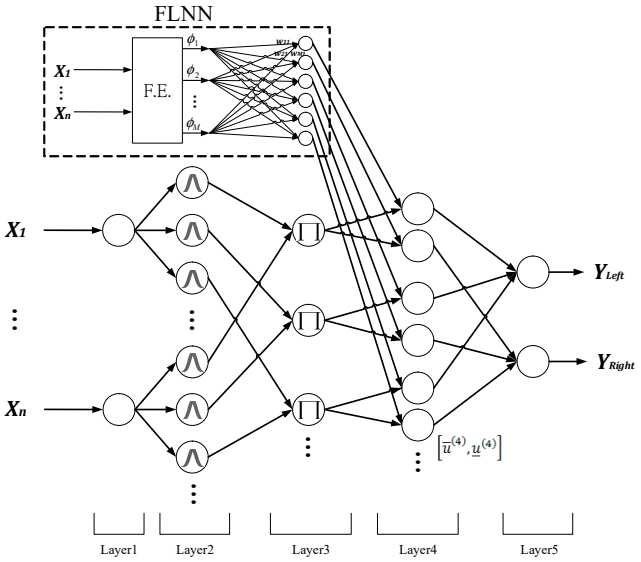


Figure 2. Structure of an interval type-2 fuzzy neural controller (IT2FNC).

The five-layer structure of the IT2FNC is described as follows:

Layer 1 (input layer): This layer only imports the input data into the next layer:

$$u_i^{(1)} = x_i. \tag{2}$$

Layer 2 (membership function layer): This layer performs the fuzzification. Each node in this layer defines an interval type-2 fuzzy set, as displayed in Figure 3. The Gaussian primary membership function has an uncertainty mean $[m_{ij1},\ m_{ij2}]$ and standard deviation σ_{ij} and is expressed as follows:

$$u_{ij}^{(2)} = exp\left(-\frac{\left[u_i^{(1)} - m_{ij}\right]^2}{\sigma_{ij}^2}\right) \equiv N\left(m_{ij},\ \sigma_{ij};\ u_i^{(1)}\right),\ m_{ij} \in \left[m_{ij1},\ m_{ij2}\right]. \tag{3}$$

The membership degree of the Gaussian primary membership function $u_{ij}^{(2)}$ is called the footprint of uncertainty (FOU) and is expressed as the upper bound $\overline{u}_{ij}^{(2)}$ and the lower bound $\underline{u}_{ij}^{(2)}$. The membership degree is expressed as follows:

$$\overline{u}_{ij}^{(2)}\left(u_i^{(1)}\right) = \begin{cases} N\left(m_{ij1},\ \sigma_{ij};\ u_i^{(1)}\right), & if \quad u_i^{(1)} < m_{ij1} \\ 1, & if \quad m_{ij1} \le u_i^{(1)} \le m_{ij2}, \\ N\left(m_{ij2},\ \sigma_{ij};\ u_i^{(1)}\right), & if \quad u_i^{(1)} > m_{ij2} \end{cases} \tag{4}$$

and

$$\underline{u}_{ij}^{(2)}\left(u_i^{(1)}\right) = \begin{cases} N\left(m_{ij2},\ \sigma_{ij};\ u_i^{(1)}\right), & if \quad u_i^{(1)} \le \frac{m_{ij1}+m_{ij2}}{2} \\ N\left(m_{ij1},\ \sigma_{ij};\ u_i^{(1)}\right), & if \quad u_i^{(1)} > \frac{m_{ij1}+m_{ij2}}{2} \end{cases}. \tag{5}$$

The output of each node is represented by the following interval: $\left[\overline{u}_{ij}^{(2)}, \underline{u}_{ij}^{(2)}\right]$.

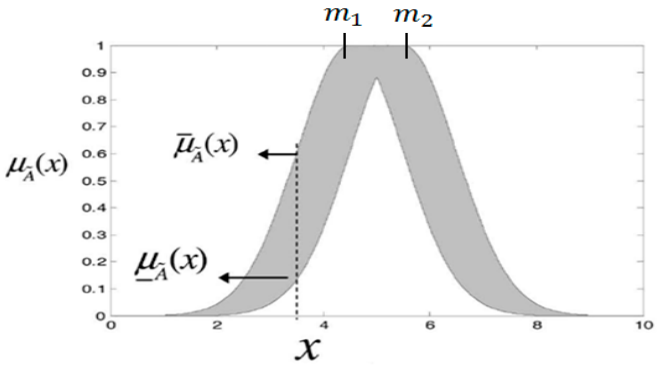


Figure 3. Interval type-2 fuzzy sets.

Layer 3 (firing layer): Each node is a rule node and uses an algebraic product operation to obtain the firing strengths $\overline{u}_j^{(3)}$ and $\underline{u}_j^{(3)}$ of each rule node. The firing strength of each rule node is defined as follows:

$$\overline{u}_j^{(3)} = \prod_i \overline{u}_{ij}^{(2)} \text{ and } \underline{u}_j^{(3)} = \prod_i \underline{u}_{ij}^{(2)}, \tag{6}$$

where $\prod_i \overline{u}_{ij}^{(2)}$ and $\prod_i \underline{u}_{ij}^{(2)}$ represent the firing strength of the interval's upper bound and lower bound, respectively.

Layer 4 (consequent layer): Interval type-2 fuzzy sets are reduced to interval type-1 fuzzy sets through a type-reduction operation. The traditional type-2 order reduction method is based on highly complex calculation. Therefore, the centers of sets (COS) method [11] is adopted for implementing the reduction process and is described as follows:

$$Y(x) = [y_l, y_r] = \frac{\int_{a^1} \cdots \int_{a^M} \int_{f^1 \in [\underline{f}^1, \overline{f}^1]} \cdots \int_{f^M \in [\underline{f}^M, \overline{f}^M]} 1}{\frac{\sum_{i=1}^{M} f^i a^i}{\sum_{i=1}^{M} f^i}}, \tag{7}$$

$$\underline{u}^{(4)} = y_l = \frac{\sum_{j=1}^{R} \underline{u}_j^{(3)} \left(\sum_{k=1}^{M} \omega_{kj} \phi_k \right)}{\sum_{j=1}^{R} \underline{u}_j^{(3)}}, \tag{8}$$

and

$$\overline{u}^{(4)} = y_r = \frac{\sum_{j=1}^{R} \overline{u}_j^{(3)} \left(\sum_{k=1}^{M} \omega_{kj} \phi_k \right)}{\sum_{j=1}^{R} \overline{u}_j^{(3)}}, \tag{9}$$

where $\sum_{k=1}^{M} \omega_{kj} \phi_k$ represents a nonlinear combination of FLNN inputs, ω_{kj} represents the link weight, and $\varnothing_k$ represents the functional expansion of FLNN inputs. The functional expansion is based on basis trigonometric functions and defined as follows:

$$[\phi_1, \phi_2, \cdots, \phi_M] = [x_1, \sin(\pi x_1), \cos(\pi x_1), \cdots, x_n, \sin(\pi x_n), \cos(\pi x_n)], \tag{10}$$

where M = 3 × n is the number of basis functions and n is the number of inputs.

Layer 5 (output layer): The node output of layer 5 is defuzzified by computing the average of $\overline{u}^{(4)}$ and $\underline{u}^{(4)}$. The crisp value y is obtained as follows:

$$y = \frac{\overline{u}^{(4)} + \underline{u}^{(4)}}{2} = u^{(5)}. \tag{11}$$

3.2. Proposed DGDE

DE has the advantages of fast convergence and simple implementation. However, some problems, such as low precision and becoming easily trapped into local optima, are encountered when complex problems are solved. Therefore, an efficient DGDE algorithm is proposed to overcome the shortcomings of traditional DE. The steps of DGDE are described as follows:

Step 1: Initialization and coding

By setting the parameter vectors of DE and randomly initializing the target vector $X_{i,G}$, the mathematical model is expressed as follows:

$$X_{i,G} = \left[x_{i,G}^1, x_{i,G}^2, \cdots, x_{i,G}^D \right] \tag{12}$$

where $i = 1, 2, \cdots, NP$; NP is the number of population; $X_{i,G}$ represents the ith parameter vector in Gth generation, and D is the number of dimensions.

The proposed DGDE is used to adjust the parameters of the IT2FNC. All the parameters in the IT2FNC are coded into one vector. Each vector represents an IT2FNC. The adjustable parameters in each IT2FNC are the uncertainty mean m_{ij}, standard deviation σ_{ij}, displacement value of the uncertainty mean d_{ij}, and link weight ω_{kj}. Where the $d_{ij} = m_{ij2} - m_{ij1} > 0$. The coding format is presented in Figure 4.

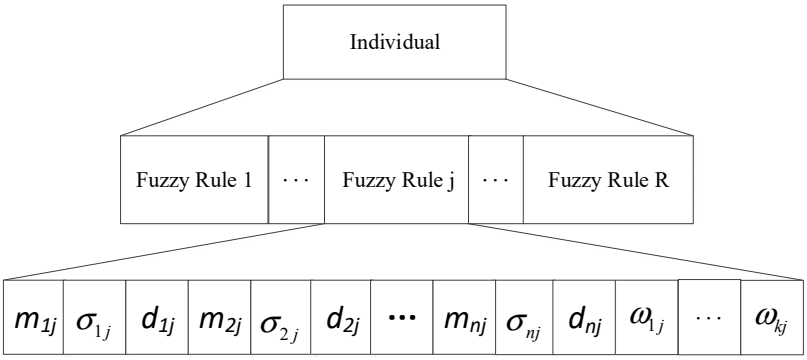


Figure 4. Coding format of the vector.

Step 2: Vector group

The fitness values of all parameter vectors are sorted in descending order. The initial group number of all vectors is set to zero (Figure 5).

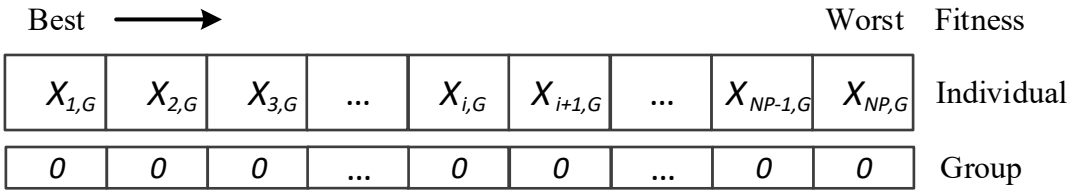


Figure 5. Fitness values of the parameter vectors in descending order.

The computation of the fitness values for the individuals in the population will be discussed later, see Equations (24)–(30). The highest fitness value of the vectors is set as the new group leader. This implies that the group number is updated from zero to one, as shown in Figure 6. On the basis of the average distance difference and the average fitness difference between these ungrouped vectors (i.e., group number 0) and the group leader, the threshold value of similarity comprises the threshold values of fitness and distance:

$$DIS^g = \sum_{i=1}^{NP}\sum_{j=1}^{D}\sqrt{\left(L_j^g - X_j^i\right)^2},\ if\ X^i \text{ is ungrouped,} \tag{13}$$

$$FIT^g = \sum_{i=1}^{NP}\left|Fit(L^g) - Fit\left(X^i\right)\right|,\ if\ X^i \text{ is ungrouped,} \tag{14}$$

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

$$Average\_Fitness\ (AFIT^g) = \frac{FIT^g}{NI}, \tag{16}$$

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

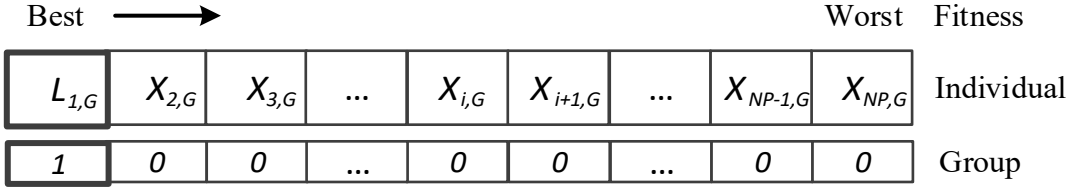


Figure 6. The first group leader with the highest fitness value.

The difference of the distance values $\left(Dis^i\right)$ and the difference of the fitness values Fit^i between the ungrouped vectors and the leader vectors are calculated as follows:

$$Dis^i = \sum_{j=1}^{D} \sqrt{\left(L_j^g - X_j^i\right)^2}, \tag{17}$$

$$Fit^i = \left|Fit(L^g) - Fit\left(X^i\right)\right| \tag{18}$$

If the $Dis^i < ADIS^g$ and $Fit^i < AFIT^g$ conditions are satisfied for a vector, then the vector is similar to the gth group leader. Therefore, these vectors are grouped together and the group number is updated to g. Otherwise, the vectors are not grouped together. Figure 7 shows that the vectors $X_{2,G}$ and $X_{i,G}$ are similar to the *1*st group leader.

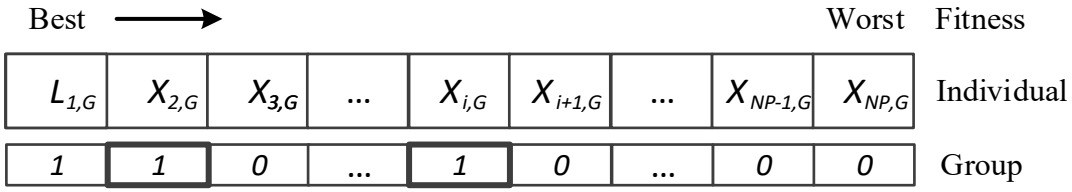


Figure 7. Generation of the first group members.

Next, if any ungrouped vectors exist, the ungrouped vector with the highest fitness value is set as the group leader in a new group. Figure 8 presents the vector $X_{3,G}$ from the ungrouped vectors that has the highest fitness value and is set as a new group leader (i.e., the 2nd group leader). The grouping process is completed when no ungroup vectors exist.

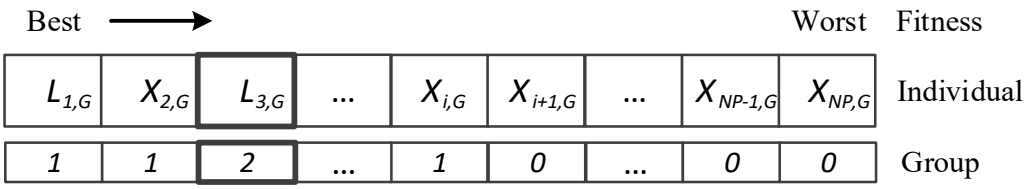


Figure 8. Generation of the second group leader.

Step 3: Mutation

Two new mutation methods are proposed for improving the disadvantages of the traditional DE algorithms. The modified formula is expressed as follows:

Mutation method 1:

$$U_{i,G+1} = X_{rL,G} + F(X_{r1,G} - X_{r2,G}) \tag{19}$$

Mutation method 2:

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

where $U_{i,G+1} = \left[u_{i,G+1}^1, u_{i,G+1}^2, \cdots, u_{i,G+1}^D\right]$, F is the mutation weight factor. Moreover, the weight factor is set to 0.5 as a general rule, $X_{best,\,G}$ is the best fitness vector, and $X_{rL,G}$ is a random leader selected from all the group leaders. Because the traditional DE method easily falls into local optima, this study employed the random leader as the base vector to increase the search ability effectively, as presented in Equation (19). In Equation (20), the best vector is set as the base vector, and three random vectors and one random leader vector are used. The mutated vector in Equation (20) revolves around the best vector and enhances the search ability in the solution space.

Step 4: Recombination

The recombination operation crosses the mutation vector with the target vector and generates a new vector $V_{i,G+1}$. The definition of $V_{i,G+1}$ is expressed as follows:

$$V_{i,G+1} = \left[v^1_{i,G+1}, v^2_{i,G+1}, \cdots, v^D_{i,G+1}\right] \tag{21}$$

$$v^j_{i,\,G+1} = \begin{cases} u^j_{i,G+1}, & if\ rand^j(0,1) \le CR \\ x^j_{i,G+1}, & otherwise \end{cases} \quad j = 1 \ldots D \tag{22}$$

where $rand^j(0,1)$ represents random values between zero and one in jth dimension, and CR is the crossover rate. The higher CR value represents the higher similarity between the vector and the mutation vector.

Step 5: Selection

The fitness values of vectors are evaluated for selecting the target vectors in the next generation. If the fitness value of a vector is less favorable than the current target vector, the target vector will remain in the next generation. The selection operation is described as follows:

$$X_{i,G+1} = \begin{cases} V_{i,G+1}\ , if\ Fit(V_{i,G+1}) & > Fit(X_{i,G}) \\ X_{i,G}, & otherwise \end{cases} \tag{23}$$

The flowchart of DGDE is presented in Figure 9.

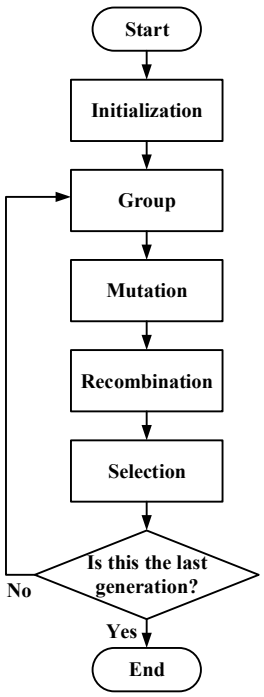


Figure 9. Flowchart of the proposed dynamic group differential evolution (DGDE).

3.3. Wall-Following Control of Mobile Robots

Recently, some researchers have proposed the wall-following control of mobile robots for using reinforcement learning. Jhang et al. [21] used an interval type-2 recurrent fuzzy cerebellar model articulation controller (IT2RFCMAC). They adopted a Takagi–Sugeno–Kang (TSK) in the consequent part in IT2RFCMAC. The TSK is a linear function and simple implementation. However, in complex problems, nonlinear functions are better in terms of performance than linear functions. Therefore, in this study a nonlinear functional link neural network of the proposed IT2FNC was used as the

consequent part to improve control performance. The proposed IT2FNC that based on DGDE is demonstrated. Wall-following control was utilized by the IT2FNC to control the mobile robot. A reinforcement learning strategy was used to adjust the controller parameters of the proposed IT2FNC. The block diagram for the wall-following control of the mobile robot is presented in Figure 10. The four input signals of the proposed IT2FNC are the S_0, S_1, S_2, and S_3 distances, which are measured by the infrared sensor. The outputs of the IT2FNC are the rotational speeds V_L and V_R of the two wheels.

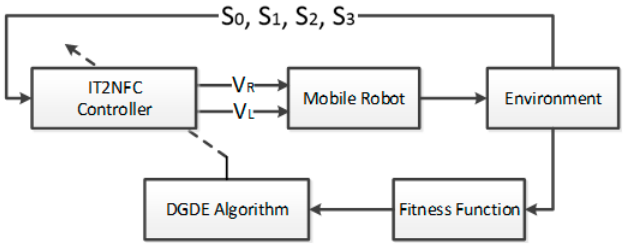


Figure 10. Block diagram of the mobile robot wall-following control.

To allow mobile robots to be used in different environments, the training environment in this study featured straight lines, corners, right-angled corners, and slope lines. Figure 11 presents the 1.7×1.6 m^2 training environment.

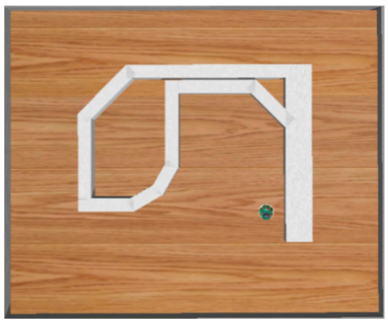


Figure 11. Training environment of the mobile robot wall-following control.

To avoid collision with obstacles and deviation from the wall during the wall-following control learning process, three terminal conditions of wall-following control learning were specified:

1. If the total moving distance of the mobile robot was larger than the predefined maximal distance of the training environment, the mobile robot successfully moved in a circular path in an unknown environment.
2. The mobile robot collided with the wall when the measured distance from any infrared sensor was less than 1 cm, as displayed in Figure 12a.
3. The mobile robot deviated from the wall when the measured distance S_2 was greater than 6 cm, as displayed in Figure 12b.

(**a**) (**b**)

Figure 12. Mobile robot (**a**) collides with the wall (**b**) deviates from the wall.

The proposed DGDE was used to train the parameters of the IT2FNC. Each vector in the DGDE represents a solution of an IT2FNC. When any terminal condition during the wall-following control learning process is satisfied, a fitness function was used to evaluate the performance of the mobile robot. The proposed fitness function comprises three sub-fitness functions—SF_1, SF_2, and SF_3. SF_1, SF_2, and SF_3 represent the total moving distance, the distance between the robot and the wall, and the degree of parallelism between the robot and the wall, respectively. The three sub-fitness functions are defined as follows:

(1) SF_1: If the moving distance R_{dis} was greater than the predefined value R_{stop}, the robot successfully moved around a circular path in the training environment and set $R_{dis} = R_{stop}$. The sub-fitness function SF_1 is defined as follows:

$$SF_1 = R_{stop} - R_{dis} \tag{24}$$

(2) SF_2: The goal of the wall-following control was to maintain a fixed distance between the side of the robot and the wall. Therefore, the sub-fitness function SF_2 is defined as the average of $WD(t)$, where $WD(t)$ represents the distance between the side of the robot and the wall at each time step, and is defined as follows:

$$WD(t) = |S_2(t) - d_{wall}| \tag{25}$$

$$SF_2 = \frac{\sum_{t=1}^{T_{stop}} WD(t)}{T_{stop}} \tag{26}$$

where d_{wall} is the pre-defined fixed distance (i.e., d_{wall} = 4 cm), as presented in Figure 13a. T_{stop} is the total number of time steps in a learning process. If the robot remains at a fixed distance from the wall, the SF_2 value is equal to zero.

(3) SF_3: This sub-fitness function was used for evaluating the degree of parallelism between the robot and the wall. If the robot was parallel to the wall, the angle θ between the robot and wall was 90°. On the basis of the law of cosines, $x(t)$ must have the same value as that of RS_2, as presented in Figure 13b.

$$RS_1 = r + \delta_1 \, , RS_2 = r + \delta_2 \tag{27}$$

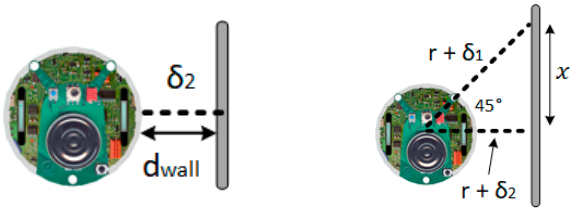
$$x(t) = \sqrt{{RS_1}^2 + {RS_2}^2 - 2RS_1RS_2\cos(45^\circ)} \tag{28}$$

where r is the radius of the robot; δ_1 and δ_2 represent the distance between the sensor 1 and the wall and that between sensor 2 and the wall, respectively, and SF_3 represents the average value of the degree of parallelism during movement. If the robot is parallel to the wall, SF_3 is equal to zero.

$$SF_3 = \frac{\sum_{t=1}^{T_{stop}} |RS_2 - x(t)|}{T_{stop}} \tag{29}$$

Therefore, the proposed fitness function is defined as follows:

$$F(\cdot) = \frac{1}{1 + (SF_1 + SF_2 + SF_3)} \tag{30}$$



(**a**) (**b**)

Figure 13. Definition of (**a**) d_{wall} and (**b**) degree of parallelism.

Figure 14 shows the block diagram of learning process of wall-following control. Each solution represents an IT2FNC controller, which is evaluated by the fitness function. Additionally, it automatically adjust parameters of IT2FNC using evolutionary strategies. The best solution will be replaced when a better solution exits in each generation.

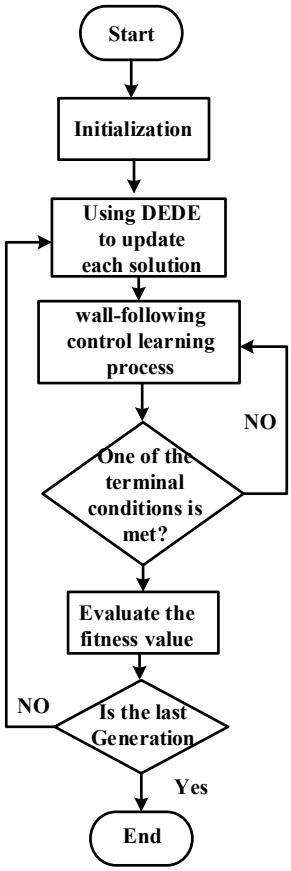


Figure 14. Block diagram of learning process of wall-following control.

3.4. Experimental Results of the Wall-Following Control

To verify the effectiveness of the proposed method, the performance of the WFM controller while using the proposed DGDE-1 (mutation method 1) and DGDE-2 (mutation method 2) were compared with the performance of the WFM control while using other methods. Each method was evaluated 10 times to verify the stability of each algorithm.

The initial parameters of the DGDE are the number of the population (NP), crossover rate, generation, weighting factor of mutation, and number of fuzzy rules, as presented in Table 1. Moreover, we considered different fuzzy rule numbers for performance evaluation. Table 2 presents the performance evaluation results of different fuzzy rule numbers. The IT2FNC with six fuzzy rules was more efficient than those with five and seven fuzzy rules.

Table 1. Initial parameters of DGDE.

NP	CR	F	Generation	Rule
30	0.9	0.5	3000	5,6,7

Table 2. Performance evaluation of different fuzzy rule numbers. STD: standard deviation.

Number of Rules / Fitness Value	DGDE-1			DGDE-2		
	5	**6**	**7**	**5**	**6**	**7**
Best	0.932	0.961	0.948	0.949	0.962	0.953
Worst	0.865	0.891	0.821	0.908	0.919	0.891
Average	0.903	0.933	0.911	0.923	0.942	0.913
STD	0.017	0.012	0.200	0.012	0.009	0.016
Number of successful runs	10	10	10	10	10	10

Table 3 presents the performance evaluations of different algorithms. In this table, the performance indexes include the best fitness function, the worst fitness function, the average fitness function, the standard deviation (STD), the number of successful runs, and computation time for one training. The number of successful runs is the number of times the robot moved successfully around a circular path in the training environment. Figure 15 presents the learning curves of the WFM control when various evolutionary algorithms are used. The proposed DGDE achieved superior fitness values and successful runs than other methods. On the other hand, the proposed method also compares with the IT2RFCMAC controller [21]. Table 4 shows that the proposed IT2FNC with nonlinear functional link neural network performs better than IT2RFCMAC with linear TSK architecture [21].

Table 3. Performance comparison of various algorithms in the wall-following control behavior.

Evaluation Items / Algorithms	Fitness Value				Number of Success Runs	Computation Time (H:M:S)
	Best	**Worst**	**Average**	**STD**		
DGDE-1	0.961	0.891	0.933	0.012	10	5:01:39
DGDE-2	0.962	0.919	0.942	0.009	10	4:38:56
JADE [22]	0.950	0.860	0.911	0.029	10	10:11:03
Rank-DE [23]	0.958	0.867	0.922	0.025	10	18:21:05
DE [14]	0.941	0.262	0.786	0.184	8	1:03:38
PSO [12]	0.947	0.206	0.738	0.257	7	5:49:45
ABC [15]	0.932	0.354	0.735	0.149	8	2:57:23

Table 4. Performance comparison of different network architecture.

Evaluation Items / Algorithms	IT2FNC		IT2RFCMAC [23]
	DGDE-1	**DGDE-2**	**DGDE**
Best	0.961	0.962	0.925
Worst	0.891	0.919	0.868
Average	0.933	0.942	0.906

To verify the WFM control performance of different learning algorithms, a training environment and two unknown testing environments were created, as presented in Figures 16–18. The testing environment in Figure 17 focuses on many difficult and large curves, whereas the other testing environment in Figure 18 focuses on many right angle curves. The fitness value was used to evaluate the WFM, and the detailed comparison results are presented in Table 5.

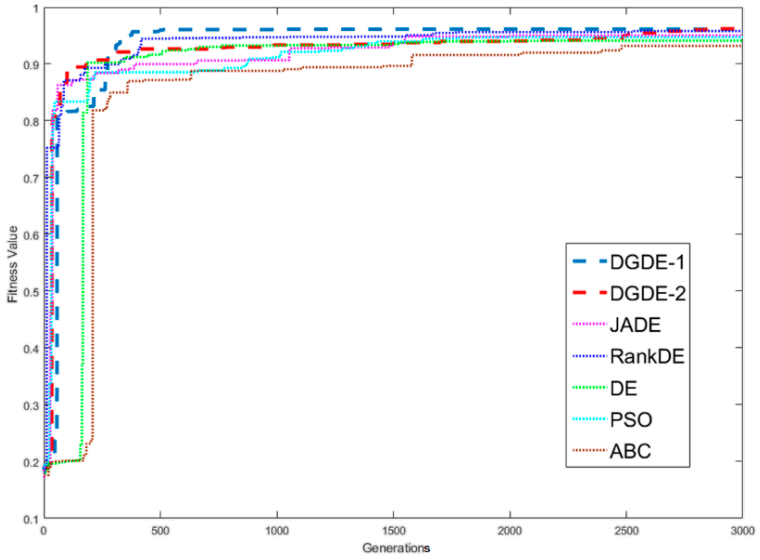
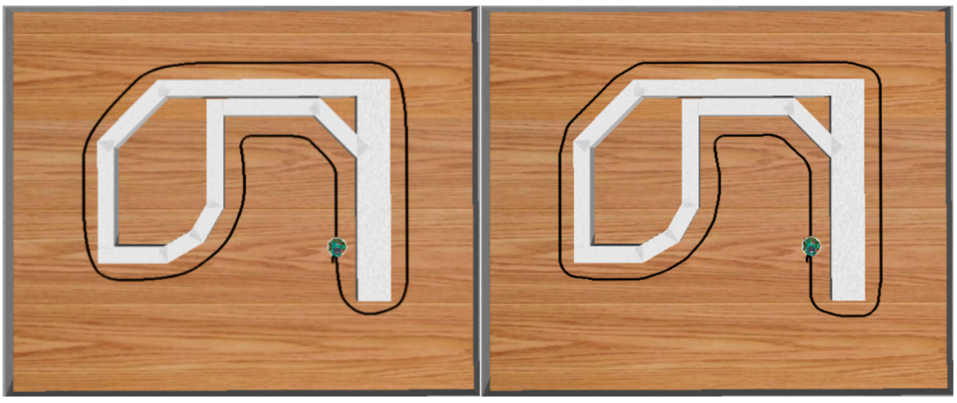


Figure 15. Learning curves of the wall-following control when various evolutionary algorithms are employed.

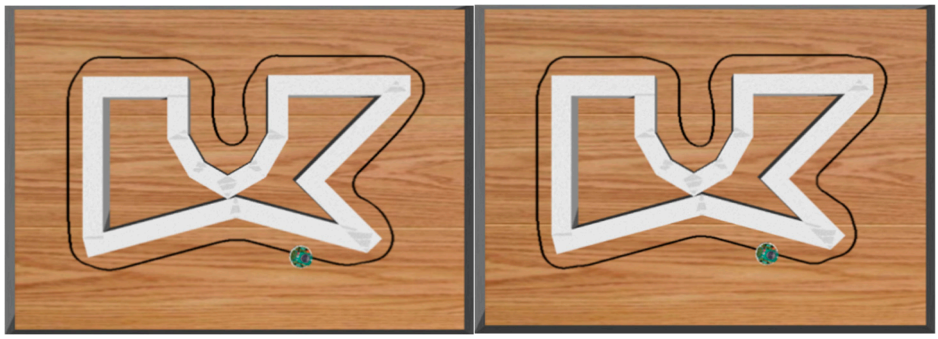
Table 5. Fitness value of various algorithms in the testing environments.

Evaluation Items / Algorithms	Fitness Value		
	Training Environment	Testing Environment 1	Testing Environment 2
DGDE-1	0.961	0.901	0.864
DGDE-2	0.962	0.899	0.872
JADE [22]	0.950	0.895	0.862
Rank-DE [23]	0.958	0.874	0.789
DE [14]	0.941	fail	fail
PSO [12]	0.947	0.828	0.721
ABC [15]	0.932	fail	fail



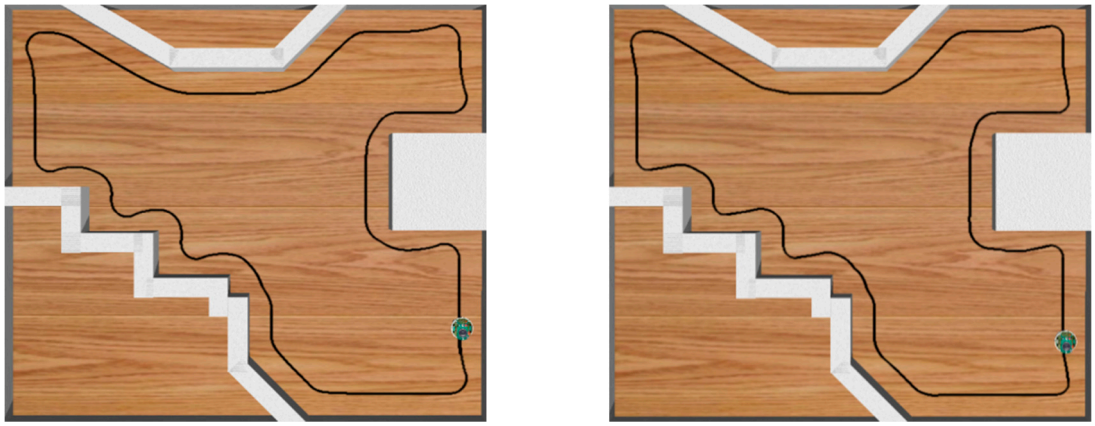
(**a**) (**b**)

Figure 16. Moving path of the robot in the training environment: (**a**) DGDE-1 and (**b**) DGDE-2.



(**a**) (**b**)

Figure 17. Moving path of the robot in the testing environment 1: (**a**) DGDE-1 and (**b**) DGDE-2.



(**a**) (**b**)

Figure 18. Moving path of the robot in the testing environment 2: (**a**) DGDE-1 and (**b**) DGDE-2.

4. Cooperative Carrying and Navigation Control of Multi-Evolutionary Mobile Robots

In this section, the cooperative carrying and navigation control of multi-evolutionary mobile robots is discussed. Figure 19 shows that the distance between two robots R_d was set to 15 cm and a rectangular object was placed on the two robots. The front and rear robots represent the leader and follower, respectively. In the experiments, the leader explored the front environment, and the follower assisted the leader in lifting objects to achieve obstacle avoidance and prevent objects being dropped during cooperative carrying.

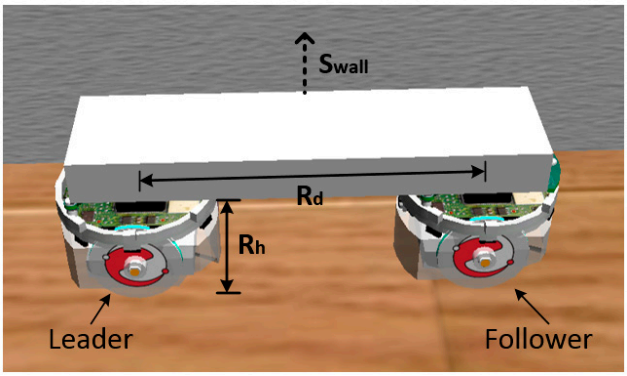


Figure 19. Cooperative carrying of two mobile robots.

4.1. Wall-Following Control of the Cooperative Carrying Method

A dual controller is proposed for the cooperative carrying of two mobile robots. An auxiliary controller was incorporated in the follower robot to learn the WFM of cooperative carrying. The block diagram is presented in Figure 20. The auxiliary controller contained five input signals and two output signals. The inputs are the sensed distances by the follower's sensors (S_0, S_1, S_2, and S_3) and the distances between two robots (R_d). The outputs are the rotational speeds V_L and V_R of the two wheels.

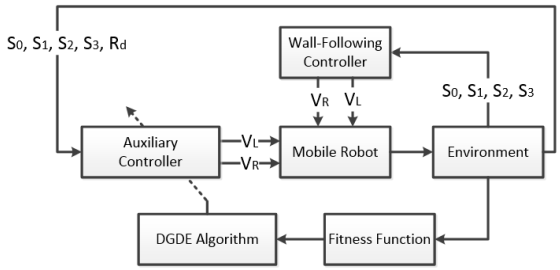


Figure 20. Block diagram of the cooperative carrying of mobile robots.

To implement cooperative carrying in an unknown environment, the training environment featured straight lines, smooth curves, U-shaped curves, and continuous curves to train the follower's auxiliary controller. Figure 21 displays the 1.5×1.4 m^2 training environment.

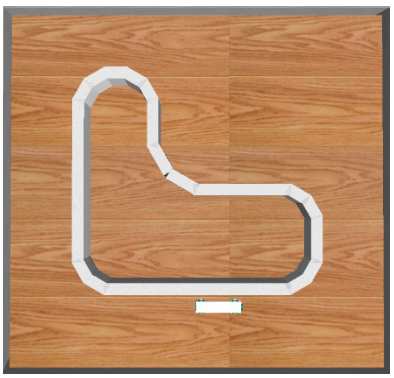


Figure 21. Training environment for cooperative carrying.

To avoid collision with obstacles and objects being dropped during the cooperative carrying learning process, five terminal conditions were specified:

(1) If the measured distance from one of the sensors in the follower robot is less than 1 cm, the follower robot collides with the obstacles.
(2) If the measured distance of the sensor S_2 is higher than 6 cm, the follower robot deviates from the wall.
(3) If the measured distance between the leader robot and follower robot R_d is less than 10 cm or higher than 20 cm, the leader and follower robots are inferred to be very close or very far.
(4) If the measured distance of the sensor is less than the height R_h of the robot, the object is dropped by robots.
(5) If the measured distance S_{wall} of the sensor is less than 1 cm or greater than 7.5 cm, the object approaches the wall or deviates from the wall.

When one of the aforementioned conditions is satisfied, the cooperative carrying of the robots has failed. If the robots engaging in cooperative carrying can successfully move around a circular path in the training environment, the auxiliary controller completes the training process. The time step of the robot moving in the training environment is defined as the fitness function $F(\cdot)$:

$$F(\cdot) = T_{stop} \tag{31}$$

4.2. Navigation Control of Cooperative Carrying

This paper also proposes an effective navigation control method for cooperative carrying in unknown environments. The manager mode automatically selects between the TGM and WFM on the basis of the relative position of the mobile robot and the target location.

(1) Toward-Goal Mode

In the navigation control of the unknown environment, the robot used infrared sensors to detect the object. In order to turn towards the goal position, the mobile robot calculates the angle difference θ_{TG} between the current direction of the robot and the target direction, as presented in Figure 22:

$$\theta_{TG} = \theta_{Robot} - \theta_{Goal} \tag{32}$$

where θ_{Robot} is the angle between the mobile robot and the x axis and θ_{Goal} is the angle between the goal and the x axis in inertial coordinate system.

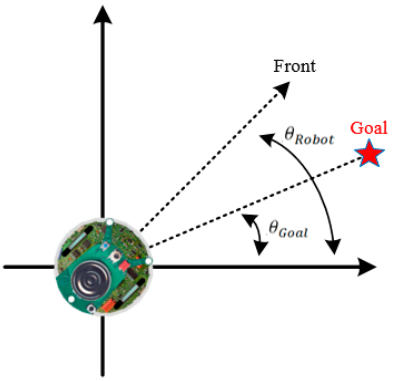


Figure 22. Angle between the robot and the target location.

To avoid objects being dropped during the cooperative carrying process, both the follower and leader move in the same direction and at the same speed, as displayed in Figure 23.

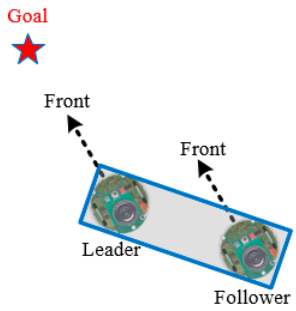


Figure 23. Follower and leader move in the same direction and at the same speed.

(2) Manager Mode

The robot is divided into three zones—O_1, O_2, and O_3, as displayed in Figure 24. In the process of TGM, the manager mode switches to the left WFM or the right WFM on the basis of the zone of the robot O_i and on the basis of which sensor S_i (i = 0, 2, 3, . . . ,7) detected an obstacle.

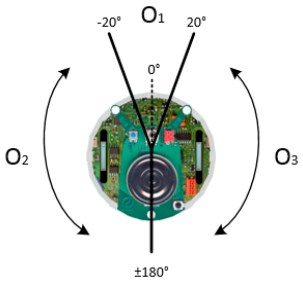


Figure 24. Divided zones of the mobile robot.

- **Left wall-following control:**

  (i) The goal direction is located at O_1, and S_7 or S_6 detects obstacles.
  (ii) The goal direction is located at O_2, and S_7, S_6, and S_5 detect obstacles.

- **Right wall-following control:**

  (i) The goal direction is located at O_1, and S_0 or S_1 detects obstacles.
  (ii) The goal direction is located at O_3, and S_0, S_1, and S_2 detect obstacles.

Before switching to the WFM, the manager mode determines which robot is closest to the obstacle. If the leader robot encounters an obstacle, it will execute a prerotation process. This aim of this process is to enable the two robots to approach the wall for a smooth navigational control, as displayed in Figure 25.

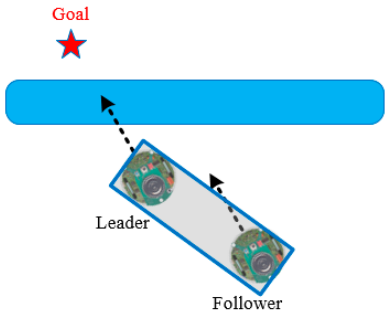


Figure 25. Leader robot encounters an obstacle.

The leader robot turns in order to be positioned parallel to the wall and records the angle of rotation θ_L. Then, this message is sent to the follower robot. The follower robot turns to $\pi - \theta_L{}^\circ$, maintains a fixed distance from the leader robot, and moves toward the obstacle. The manager mode of the two robots switch to WFM until the pre-rotation process of the follower robot is completed. The pre-rotation process is presented in Figure 26.

If the follower robot encounters an obstacle, the manager mode switches to the WFM; this process is displayed in Figure 27. If the target direction is located in O_1 of the follower robot and the distance between the sensor on right side S_1 (the sensor of left side S_6) and the wall is greater than 6 cm, the object being held by the two robots is inferred to have passed the obstacle. Then, the manager mode switches to the TGM, as displayed in Figure 28.

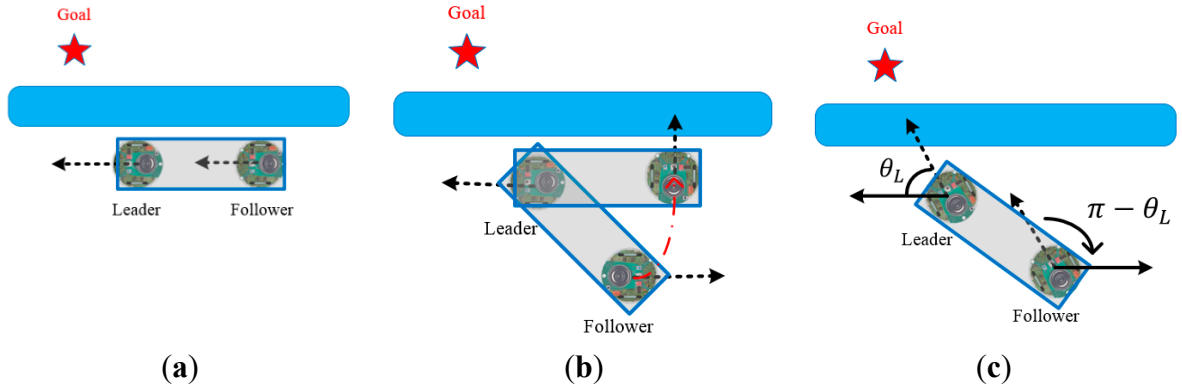


Figure 26. (**a**) Follower robot turns $\pi - \theta_L{}^\circ$; (**b**) Pre-rotation process; (**c**) Manager mode of the two robots switch to the wall-following mode (WFM).

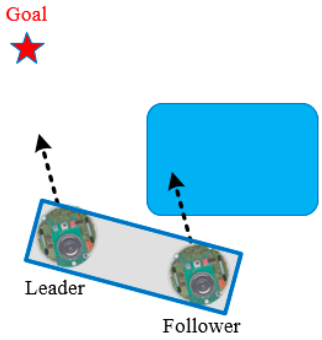
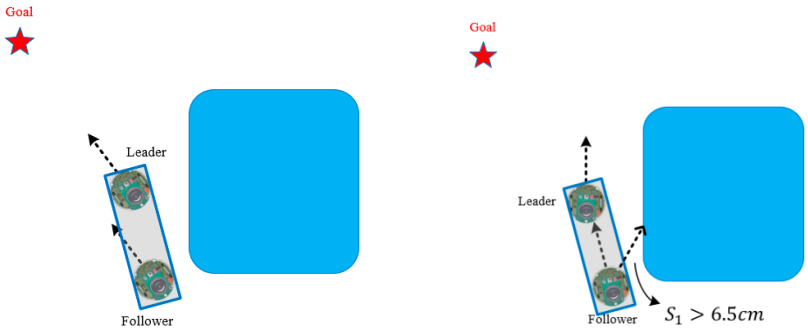


Figure 27. Follower robot encounters an obstacle.



(**a**) Toward-goal mode (**b**) Passes the obstacle

Figure 28. Object on the two robots passes the obstacle.

4.3. Experimental Results of Cooperative Carrying Control

In this subsection, the proposed method is used to verify the success of cooperative carrying control in unknown environments. Figure 29 demonstrates that the robots complete the wall-following control of cooperative carrying in the training environment. Moreover, to verify the performance of navigation control, two different test environments were created for testing whether the robots successfully accomplished cooperative carrying and navigation control. The experimental results of the two test environments are presented in Figure 30. In this experiment, the average distance (RD) between the two robots and the average distance (FWD) between the follower robot and the wall were evaluated. The results are presented in Table 6. If the RD is large, the two robots (i.e., the leader robot and the follower robot) are not at a suitable distance during the cooperative transport control, and the object drops easily. However, if the FWD is very large or very small, the robots pass the curves with poor efficiency, and the object is easily moved and dropped.

Table 6. Evaluation of the cooperative carrying performance.

Evaluation Items / Algorithms	Training Environment		Testing Environment 1		Testing Environment 2	
	RD (cm)	FWD (cm)	RD (cm)	FWD (cm)	RD (cm)	FWD (cm)
DGDE-1	16.18	3.81	16.54	3.61	17.03	3.42
DGDE-2	15.43	3.96	15.76	3.89	17.16	4.23

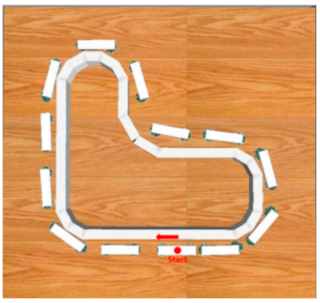
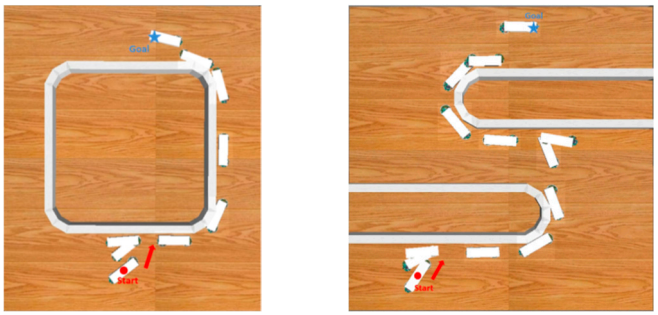


Figure 29. Wall-following control of cooperative carrying in the training environment.



(**a**) (**b**)

Figure 30. Navigation control of cooperative carrying in (**a**) test environment 1 and (**b**) test environment 2.

5. Conclusions

Aiming at the navigation control for cooperative carrying in an unknown environment, this study proposed an IT2FNC based on a dynamic group differential evolution to realize the carrying control and WFM control for mobile robots. At the same time, the developed DGDE learning algorithm adopts dynamic grouping and local search methods, which enhance the search ability and convergence stability of the traditional DE method, and is used to adjust IT2FNC parameters. On this basis, manager mode is established to assist mobile robots in navigation control. The manager mode automatically selects between the WFM or the TGM base on the relative position between the mobile robot and the target location. In addition, the pre-rotation mechanism was employed to accomplish cooperative carrying control. In the training process, the best fitness function, the worst fitness function, the average fitness function, the standard deviation (STD), the number of successful runs, and the computation time of the proposed DGDE were 0.962, 0.919, 0.942, 0.009, 10, and 4:38:56, respectively. Although DE and ABC are shorter than the proposed method at the computation time, DE and ABC had only eight successful runs during 10 runs. Experimental results revealed that the proposed method achieved a superior WFM performance than other methods and successfully accomplished the navigation control of cooperative carrying to target locations in unknown environments. Since the trained controller using reinforcement learning needs to take a long time, future research work needs to implement the proposed learning algorithm on chip to improve the learning speed.

Author Contributions: Revised manuscript, K.-Y.Y. and J.-Y.J.; Conceptualization and Methodology, J.-Y.J., K.-Y.Y. and C.-J.L.; Software, J.-Y.J.; Writing—Original Draft Preparation, C.-J.L.

Funding: This research was funded by Ministry of Science and Technology of the Republic of China, Taiwan (No. MOST 107-2221-E-167-023).

Conflicts of Interest: The authors declare that there is no conflict of interests regarding the publication of this manuscript.

References

1. Cupertino, F.; Giordano, V.; Naso, D.; Delfine, L. Fuzzy control of a mobile robot. *IEEE Robot. Autom. Mag.* **2006**, *13*, 74–81. [CrossRef]
2. Zhu, A.; Yang, S.X. Neurofuzzy-based approach to mobile robot navigation in unknown environments. *IEEE Trans. Syst. Man Cybern. C Appl. Rev.* **2007**, *37*, 610–621. [CrossRef]
3. Rusu, P.; Petriu, E.M.; Whalen, T.E.; Cornell, A.; Spoelder, H.J.W. Behavior-based neuro-fuzzy controller for mobile robot navigation. *IEEE Trans. Instrum. Meas.* **2003**, *52*, 1335–1340. [CrossRef]
4. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 1998.
5. Hsu, C.H.; Juang, C.F. Multi-objective continuous-ant-colony-optimized FC for robot wall-following control. *IEEE Comput. Intell. Mag.* **2013**, *8*, 28–40. [CrossRef]
6. Anish, P.; Parhi, D.R. Multiple mobile robots navigation and obstacle avoidance using minimum rule based ANFIS network controller in the cluttered environment. *Int. J. Adv. Robot. Automn.* **2016**, *1*, 1–11.
7. Juang, C.F.; Chang, Y.C. Evolutionary group-based particle swarm-optimized fuzzy controller with application to mobile robot navigation in unknown environments. *IEEE Trans. Fuzzy Syst.* **2011**, *19*, 379–392. [CrossRef]
8. Mendel, J.M. Advances in type-2 fuzzy sets and systems. *Inf. Sci.* **2007**, *177*, 84–110. [CrossRef]
9. Mendel, J.M. Type-2 fuzzy sets and systems: An overview. *IEEE Comput. Intell. Mag.* **2007**, *2*, 20–29. [CrossRef]
10. Kim, C.J.; Chwa, D. Obstacle avoidance method for wheeled mobile robots using interval type-2 fuzzy neural network. *IEEE Trans. Fuzzy Syst.* **2015**, *23*, 677–687. [CrossRef]
11. Castillo, O.; Melin, P. A review on the design and optimization of interval type-2 fuzzy controllers. *Appl. Soft. Comput.* **2012**, *12*, 1267–1278. [CrossRef]
12. Kennedy, J.; Eberhart, R. Particle swarm optimization. *IEEE Int. Conf. Neural Netw.* **1995**, *4*, 1942–1948.
13. Dorigo, M.; Caro, G.D. Ant colony optimization: A new meta-heuristic. In Proceedings of the 1999 Congress on Evolutionary Computation, Washington, DC, USA, 6–9 July 1999; Volume 2, pp. 1470–1477.
14. Storn, R.; Price, K. Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [CrossRef]
15. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
16. Passino, K.M. Biomimicry of bacterial foraging for distributed optimization and control. *IEEE Control Syst. Mag.* **2002**, *22*, 52–67.
17. Zhou, Y.; Wang, J.; Zhou, Y.; Qiu, Z.; Bi, Z.; Cai, Y. Differential evolution with guiding archive for global numerical optimization. *Appl. Soft Comput.* **2016**, *43*, 424–440. [CrossRef]
18. Das, S.; Mullick, S.S.; Suganthan, P.N. Recent advances in differential evolution—An updated survey. *Swarm Evol. Comput.* **2016**, *27*, 1–30. [CrossRef]
19. Chang, J.Y.; Lin, Y.Y.; Han, M.F.; Lin, C.T. A functional-link based interval type-2 compensatory fuzzy neural network for nonlinear system modeling. In Proceedings of the 2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011), Taipei, Taiwan, 27–30 July 2011; pp. 939–943.
20. Chen, C.H.; Su, M.T.; Lin, C.J.; Lin, C.T. A hybrid of bacterial foraging optimization and particle swarm optimization for evolutionary neural fuzzy classifier design. *Int. J. Fuzzy Syst.* **2014**, *16*, 422–433.
21. Jhang, J.Y.; Lin, C.J.; Lin, T.C.; Chen, C.C.; Young, K.Y. Using Interval Type-2 Recurrent Fuzzy Cerebellar Model Articulation Controller Based on Improved Differential Evolution for Cooperative Carrying Control of Mobile Robots. *Sens. Mater.* **2018**, *30*, 2499–2516. [CrossRef]

22. Zhang, J.; Sanderson, A.C. JADE: Adaptive differential evolution with optional external archive. *IEEE Trans. Evol. Comput.* **2009**, *13*, 945–958. [CrossRef]
23. Gong, W.; Cai, Z. Differential evolution with ranking-based mutation operators. *IEEE Trans. Cybern.* **2013**, *43*, 2066–2081. [CrossRef] [PubMed]