sensors

MDPI

Article

Interval Type-2 Neural Fuzzy Controller-Based Navigation of Cooperative Load-Carrying Mobile Robots in Unknown Environments

Chun-Hui Lin [1], Shyh-Hau Wang [1] and Cheng-Jian Lin [2,*]

[1] Department of Computer Science & Information Engineering, Nation Cheng Kung University, Tainan 701, Taiwan; P78071044@mail.ncku.edu.tw (C.-H.L.); shyhhau@mail.ncku.edu.tw (S.-H.W.)
[2] Department of Computer Science & Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan
* Correspondence: cjlin@ncut.edu.tw; Tel.: +886-4-2392-4505 (ext. 8753)

Received: 20 October 2018; Accepted: 24 November 2018; Published: 28 November 2018

Abstract: In this paper, a navigation method is proposed for cooperative load-carrying mobile robots. The behavior mode manager is used efficaciously in the navigation control method to switch between two behavior modes, wall-following mode (WFM) and goal-oriented mode (GOM), according to various environmental conditions. Additionally, an interval type-2 neural fuzzy controller based on dynamic group artificial bee colony (DGABC) is proposed in this paper. Reinforcement learning was used to develop the WFM adaptively. First, a single robot is trained to learn the WFM. Then, this control method is implemented for cooperative load-carrying mobile robots. In WFM learning, the proposed DGABC performs better than the original artificial bee colony algorithm and other improved algorithms. Furthermore, the results of cooperative load-carrying navigation control tests demonstrate that the proposed cooperative load-carrying method and the navigation method can enable the robots to carry the task item to the goal and complete the navigation mission efficiently.

Keywords: evolutionary robot; navigation control; fuzzy control; wall-following control; cooperative carrying; interval type-2 neural fuzzy controller; artificial bee colony algorithm; grouping strategy

1. Introduction

The robotics is rapidly progressed in recent years. Many researchers [1–4] have applied robots to various fields. Paul et al. [1] proposed a biomimetic robotic fish for informal science learning. Christopher et al. [2] presented a new robotic harvester that can autonomously harvest sweet pepper in protected cropping environments. Michail et al. [3] designed an autonomous robotic vehicle for monitoring the difficult fields to access or dangerous for humans. Maurizio et al. [4] developed effective emotion-based assistive behaviors for a socially assistive robot intended for natural human-robot interaction scenarios with explicit social and assistive task functionalities.

Several real-life tasks are easy for humans but difficult for robots. Robots do not possess brains to think as humans. Therefore, if people can make robots think like humans and judge things more flexibly, then robots can be used in various ways. Recent studies have evaluated robotic applications such as obstacle avoidance, path navigation, load carrying, and path planning.

In the real environment, the noise and interference cause sensor uncertainty. To solve complicated problems in engineering, we cannot successfully use classical methods because of various uncertainties typical for those problems. Several theories can be considered as mathematical tools for dealing with uncertainties. For a stochastically stable phenomenon in probability theory, it should exist a limit of the sample mean in a long series of trials. To test the existence of the limit, a large number of

trials must be performed. Interval mathematics have arisen as a method of taking into account the errors of calculations by constructing an interval estimate for the exact solution of a problem. This is useful in many cases, but the methods of interval mathematics are not sufficiently adaptable for problems with different uncertainties. The most appropriate theory, for dealing with uncertainties is the fuzzy set theory [5] developed. Fuzzy set is defined by employing the fuzzy membership function. The fuzzy set in [5] is also called type-1 fuzzy set. The fuzzy set operations based on the arithmetic operations with membership functions. Rough set theory [6] is similar to fuzzy set theory, however the uncertain and imprecision in this approach is expressed by a boundary region of a set, and not by a partial membership as in fuzzy set theory. Rough set is defined by topological operations called approximations, thus this definition also requires advanced mathematical concepts. Molodtsov [7] did not impose only one way to set the membership function and used an adequate parametrization to soft set theory. In this study, because the fuzzy set theory is progressing rapidly at the present time, we focus on the fuzzy set for solving control problems.

Some researchers have used the type-1 fuzzy system to robot control. Algabri et al. [8] proposed the fuzzy logic control system for controlling mobile robots and used evolutionary algorithm to adjust the parameters of a fuzzy controller. Lee et al. [9] adopted a fuzzy neural network based on reinforcement learning to implement the robot's navigation control. Fathinezhad et al. [10] used supervised fuzzy sarsa learning (SFSL) to guide the robots' navigation in environments that have obstacles. SFSL combines supervised learning and reinforcement learning to reduce learning and failure times. Infrared sensors were used in the study conducted by Anish [11]. These infrared sensors transmitted the distance between the robot and the obstacle to the adaptive network-based fuzzy inference system (ANFIS) controller. According to the aforementioned results, although the type-1 fuzzy system can successfully solve the navigation control and avoid collision problems, the control performance of robots is not good enough.

Recently, some researchers have extended the type-1 fuzzy set to the type-2 fuzzy set in a fuzzy system for solving robotic control [12–15], data classification [16], function approximation [17,18], and automatic control [19,20]. The type-2 fuzzy set incorporate uncertainty about the membership function into fuzzy set theory. Thus, type-2 fuzzy sets generalize type-1 fuzzy sets and more uncertainty can be handled. In [16–20], the experimental results have shown that the type-2 fuzzy set has a better performance than the type-1 fuzzy set for dealing with uncertainties. The output set corresponding to each rule of the type-2 fuzzy system is a type-2 fuzzy set. The type-reduction combines all these output sets and then performs a centroid calculation on this type-2 fuzzy set. The center-of-sets defuzzification method [21] is to replace each rule consequent set by a singleton situated at its centroid and then find the centroid of the type-1 set comprised of these singletons. However, general type-2 fuzzy systems are computationally intensive because type-reduction is very intensive. Things simplify a lot when secondary membership functions are set to interval sets. In this case, the secondary memberships are either zero or one. Therefore, the interval type-2 fuzzy set was proposed by [22,23] and adopted to reduce the computational complexity in this study.

To resolve control problems, the backpropagation (BP) algorithm is used commonly to adjust the parameters of neural fuzzy controllers. However, the fast convergence ability of BP might cause the system to fall into local optimal solutions instead of global optimal solutions. Hence, evolutionary algorithms, such as the particle swarm optimization (PSO) [24], ant colony optimization [25], differential evolution (DE) [26], and artificial bee colony (ABC) [27], are proposed to find global optimal solutions. To speed up the convergence time without falling into local optimal solutions and improve accuracy, the group strategy is imported in the ABC algorithm. The proposed dynamic group artificial bee colony (DGBAC) balances the employed bees' searching ability and changes the movement equations of employed bees and onlooker bees.

The goal of this paper proposes a cooperative load-carrying method of mobile robots to carry the task item and navigate in an unknown environment. The behavior mode manager is used efficaciously in the navigation control to switch between two behavior modes, wall-following mode (WFM) and

goal-oriented mode (GOM), according to various environmental conditions. Additionally, an interval type-2 neural fuzzy controller (IT2NFC) based on DGABC is proposed in this paper. Reinforcement learning was used to develop the WFM adaptively. The proposed IT2NFC-based DGABC has several advantages, such as (1) Only small amount of parameters are required; (2) Interval type-2 fuzzy sets are used to reduce sensor-sensing noise and disturbance; (3) The effectiveness and robustness of the proposed controller are improved. Based on the aforementioned advantages, the behavior mode manager makes the moving path of the mobile robot smoother and automatically switches between the two modes to complete the task.

2. Structure of IT2NFC

In this section, the IT2NFC is introduced. The five layers in this structure are as follows: input layer, fuzzification layer, firing layer, output processing layer, and output layer. In Figure 1, X_1~X_n represent the inputs of the IT2NFC, which are the distances between the nearby objects and the robot's infrared sensors. Y_{Left} and Y_{Right} indicate the outputs of the IT2NFC, which are the speeds of the right and left wheels, respectively.

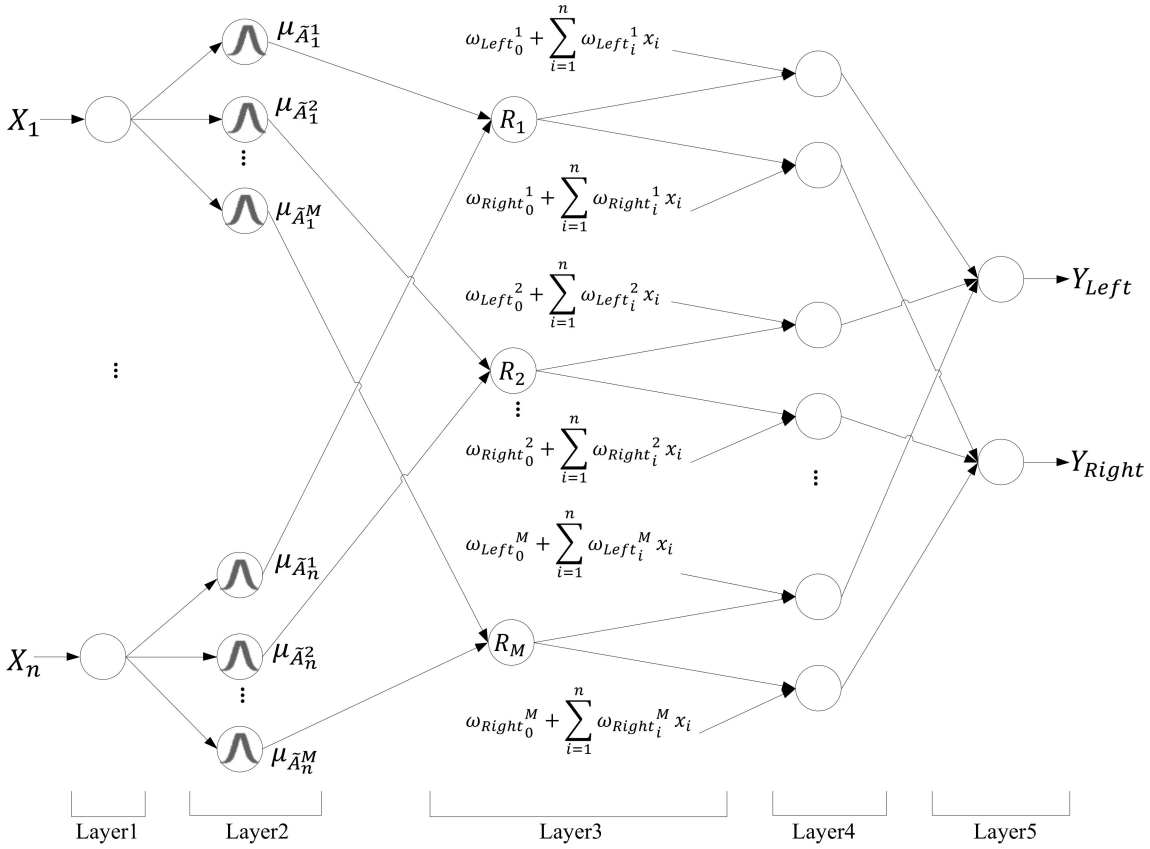


Figure 1. Structure of IT2NFC.

To improve the performance of the fuzzy system, the order reduction method, established by O. Castillo [19], was used in this paper. The reduction of order in the system can decrease the computational complexity. The rule in the IT2NFC is illustrated as follows:

$$\begin{aligned} &\text{Rule } j: \text{ IF } x_1 \text{ is } \widetilde{A}_1^j \text{ and } x_2 \text{ is } \widetilde{A}_2^j \ldots \text{and } x_n \text{ is } \widetilde{A}_n^j \\ &\text{THEN } y_{Left} \text{ is } \omega_{Left_0}^j + \textstyle\sum_{i=1}^{n} \omega_{Left_i}^j x_i, \\ &y_{Right} \text{ is } \omega_{Right_0}^j + \textstyle\sum_{i=1}^{n} \omega_{Right_i}^j x_i \end{aligned} \tag{1}$$

where x_i is the input variable, $\widetilde{A}_i^j$ is an IT2FS, y_{Left} and y_{Right} are the output variables, $\omega_0^j + \sum_{i=1}^{n} \omega_i^j x_i$ is the output of the Takagi–Sugeno–Kang linear function, i represents the input from 1 to n, and j is the number of the rule.

1. Input Layer

In this layer, the input data is sent to the next layer directly without any computation.

$$x_i = X_i \tag{2}$$

2. Fuzzification Layer

In this layer, named the membership function layer, the input data is calculated in a fuzzy manner. Each node is defined as an IT2FS. Figure 2 displays that each membership function is one Gaussian membership function and has an uncertainty mean $[m_1, m_2]$ and a fixed standard deviation σ.

$$u_{\widetilde{A}_i^j} = \exp\left(-\frac{1}{2}\frac{\left[x_i - m_i^j\right]^2}{\left(\sigma_i^j\right)^2}\right) \equiv N\left(m_i^j, \sigma_i^j; x_i\right), m_i^j \in \left[m_{i1}^j, m_{i2}^j\right] \tag{3}$$

where m_i^j and σ_i^j are the mean and deviation in the ith input in the jth Gaussian membership function. However, horizontal shifting causes the upper bound $\overline{u}_{\widetilde{A}}$ and lower bound $\underline{u}_{\widetilde{A}}$ of the membership function. The interval can be represented as $\left[\overline{u}_{\widetilde{A}_i^j}, \underline{u}_{\widetilde{A}_i^j}\right]$.

$$\overline{u}_{\widetilde{A}_i^j} = \begin{cases} N\left(m_{i1}^j, \sigma_i^j; x_i\right), & x_i < m_{i1}^j \\ 1, & m_{i1}^j \le x_j \le m_{i2}^j \\ N\left(m_{i2}^j, \sigma_i^j; x_i\right), & x_i > m_{i2}^j \end{cases} \tag{4}$$

and

$$\underline{u}_{\widetilde{A}_i^j} = \begin{cases} N\left(m_{i2}^j, \sigma_i^j; x_i\right), & x_i \le \frac{m_{i1}^j + m_{i2}^j}{2} \\ N\left(m_{i1}^j, \sigma_i^j; x_i\right), & x_i > \frac{m_{i1}^j + m_{i2}^j}{2} \end{cases} \tag{5}$$

3. Firing Layer

In this layer, each node denotes the one rule node. Take the product of each fuzzy set and calculate the firing strength F^j. of each rule.

$$F^j = \left[\underline{f}^j, \overline{f}^j\right] \tag{6}$$

$$\overline{f}^j = \prod_i^n \overline{u}_{\widetilde{A}_i^j} \; and \; \underline{f}^j = \prod_i^n \underline{u}_{\widetilde{A}_i^j} \tag{7}$$

where $\underline{f}^j$ and $\overline{f}^j$ denote the upper and lower bounds of the firing strength.

4. Output Processing Layer

In this layer, the center-of-sets order reduction operation is used to produce IT1FS, and then the center-of-gravity method is used to de-fuzzy the set and obtain the output value $[y_l, y_r]$.

$$Y(x) = [y_l, y_r] = \int_{a^1} \cdots \int_{a^M} \int_{f^1 \in [\underline{f}^1, \overline{f}^1]} \cdots \int_{f^M \in [\underline{f}^M, \overline{f}^M]} 1 / \frac{\sum_{i=1}^{M} f^i a^i}{\sum_{i=1}^{M} f^i} \tag{8}$$

The output of the fuzzy set is defined as the following:

$$y_l = \frac{\sum_{j=1}^{M} \underline{f}^j \left(w_0^j + \sum_{i=1}^{n} w_i^j x_i \right)}{\sum_{j=1}^{M} \underline{f}^j} \tag{9}$$

and

$$y_r = \frac{\sum_{j=1}^{M} \overline{f}^j \left(w_0^j + \sum_{i=1}^{n} w_i^j x_i \right)}{\sum_{j=1}^{M} \overline{f}^j} \tag{10}$$

where y_l and y_r indicate the upper and lower bound values of the output, w_i^j represents the linked weights of each node, M represents the number of the rule, and n means the number of the input.

5. Output Layer

In this layer, the pervious output value is averaged to get the certain output from the neural network y.

$$y = \frac{y_l + y_r}{2} \tag{11}$$

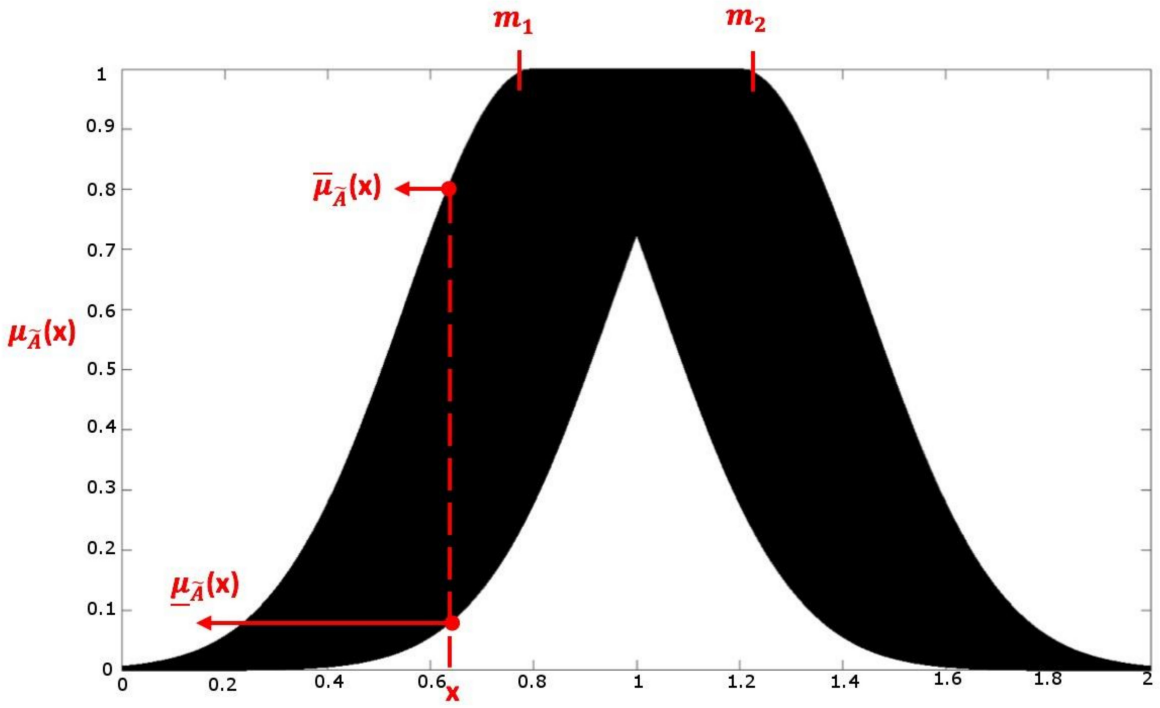


Figure 2. Interval type-2 fuzzy sets.

3. Structure of DGABC

3.1. Structure of ABC

The ABC, proposed by Karaboga in 2005, is a global optimization algorithm based on swarm intelligence. The advantages of ABC include fewer control parameters, parallel computing, global searching, simple computing, easy implementation, and robustness. The ABC consists of employed bees, onlooker bees, and scout bees. In the traditional algorithm, the employed bees search for a new food source within previous searching experiments and share the resource messages to onlooker bees. The onlooker bees wait for the messages from the employed bees and then search for the new food

source according to employed bees' information. The scout bees search all the food sources. If the food source gains do not improve, then the scout bees search a new food source in the search area randomly. The traditional ABC is introduced in the following steps [23].

Step 1: Initialization

Initialize the total number of food sources (FN), the maximum number of epochs (MEN), and the limit number of each non-improved food source (limit). Put each employed bee in the solution space randomly and set the location as a food source. The fitness of the food source is named the food source gains.

$$x_i^j = x_{min}^j + rand[0,1]\left(x_{max}^j - x_{min}^j\right) \tag{12}$$

where x_i^j is the initial value of the ith employed bee in the jth dimension, x_{min}^j and x_{max}^j are the minimum and maximum values of jth dimension in the search space, and the $rand[0,1]$ is the random number from 0 to 1.

Step 2

Calculate the profits of the food source after each employed bee moves to a new food source.

$$v_i^j = x_{ij} + rand[-1,1]\left(x_i^j - x_k^j\right) \tag{13}$$

where v_i^j is the new location of the ith employed bee in the jth dimension, x_i^j is the previous location of the ith employed bee in the jth dimension, x_k^j is the location of the randomly chosen employed bee k in the jth dimension, and the $rand[-1,1]$ is a random number from -1 to 1.

Step 3

Use roulette wheel selection (Equation (14)) to determine which food source should be searched by each onlooker bee. The onlooker bee searches for the nearby food source and then calculates the profits of that food source.

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \tag{14}$$

where P_i is the probability of the ith food source being selected, fit_i is the profits of the ith food source, and FN is the number of the food source.

Step 4

If the food source gains does not improve after limit times, then abandon this food source. According to Equation (12), the scout bee will find a new food source and replace the abandoned food source.

Step 5

Record the highest food source gains and set it as the best solution in the algorithm.

Step 6

Recursively run the algorithm until the current epoch is equal to MEN. If YES, then stop the algorithm and output the best solution. If NO, then return to step 2.

3.2. Structure of DGABC

The proposed DGABC is used to adjust the parameters in the IT2NFC. DGABC balances the search ability of the traditional ABC and avoids the early converge problem. The proposed DGABC uses the dynamic grouping strategy and reforms the movement equations of the employed and onlooker

bees to improve the convergence speed and the algorithm's performance. The evolution flowchart is displayed in Figure 3.

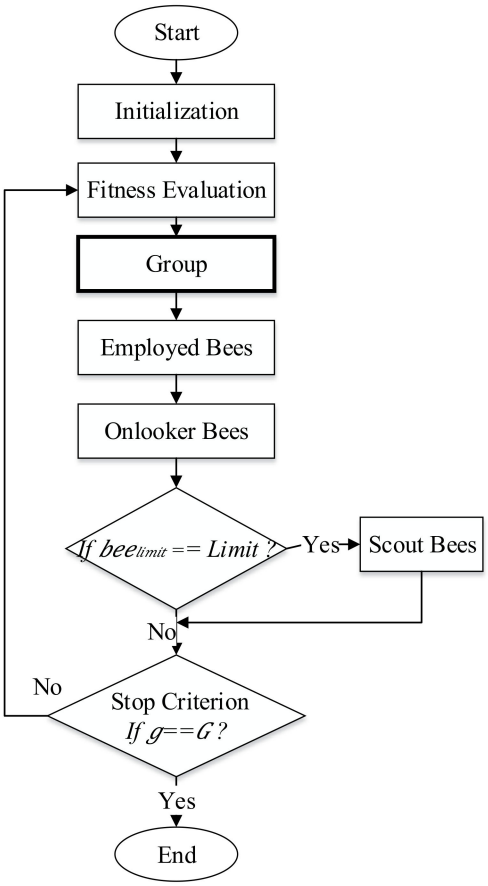


Figure 3. Flowchart of DGABC.

In DGABC, each bee represents an IT2NFC and a set of parameters. Figure 4 depicts the coding of all the parameters including the uncertain mean $\left[m_{i1}^{j}, m_{i2}^{j}\right]$, standard deviation σ_i^j, and link weights $w_{Left_0}^{j}, w_{Left_i}^{j}, w_{Right_0}^{j}, w_{Right_i}^{j}$.

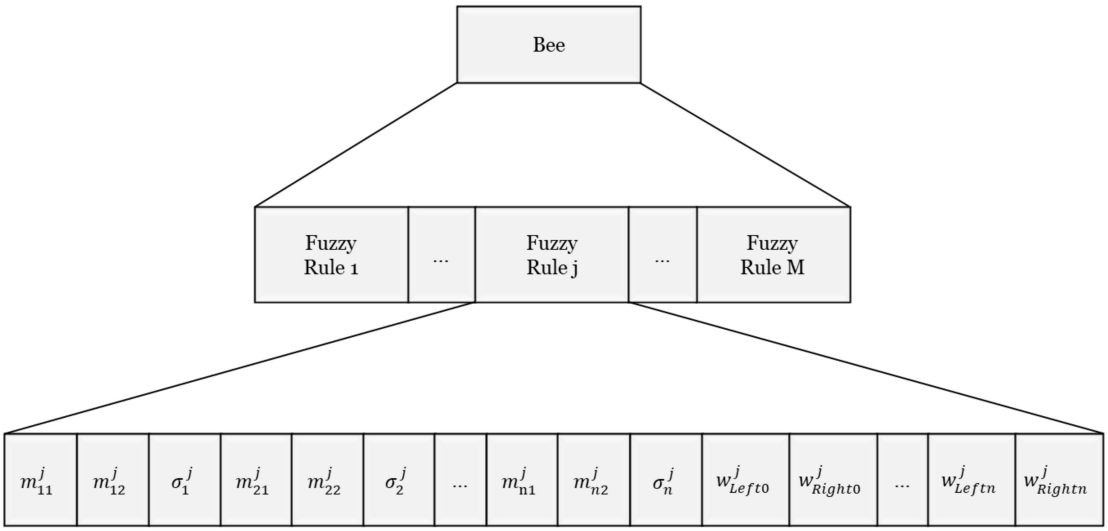


Figure 4. Coding of a bee.

In the traditional ABC, the number of employed and onlooker bees is equal. However, improper setting of the number of employed and onlooker bees might result in the unequal searching ability of global and local searching and lower the performance.

The group strategy is used to bring parity to the searching competence. Bees are separated into groups dynamically according to their performance. The best bee is selected as the leader. To maintain consistency with the traditional ABC, in which employed bees lead onlooker bees, the leader in each group is selected as an employed bee and the rest of the bees in the group are selected as onlooker bees. During training, the proportion of two types of bees can be adjusted automatically.

Using similarity evaluation to categorize individuals can prevent the following three cases (depicted in Figure 5).

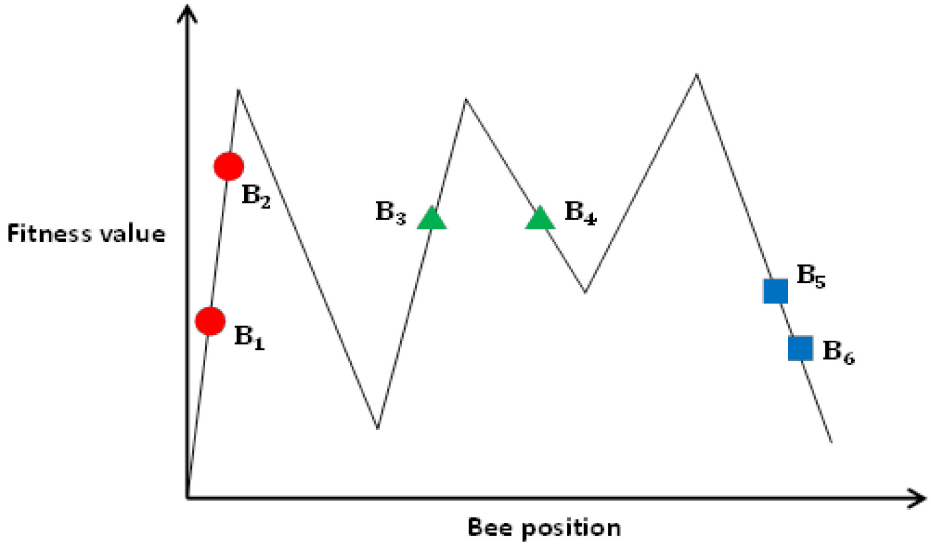


Figure 5. Similarity evaluation of individuals.

Case 1:

B_1 and B_2 are in a similar location. However, the fitness values are very different.

Case 2:

B_3 and B_4 have similar fitness values. However, the distance between them is large.

Case 3:

B_5 and B_6 have similar fitness and are located close to each other. Based on similarity evaluation, and B_6 are similar individuals.

Some researchers [20] have used thresholds to group the individuals. However, the fixed threshold results in uneven number of groups in early and late epochs. Therefore, dynamic adjustment of the threshold is proposed in this paper.

The DGABC steps are as follows (depicted in Figure 6):

1. Dynamic Grouping

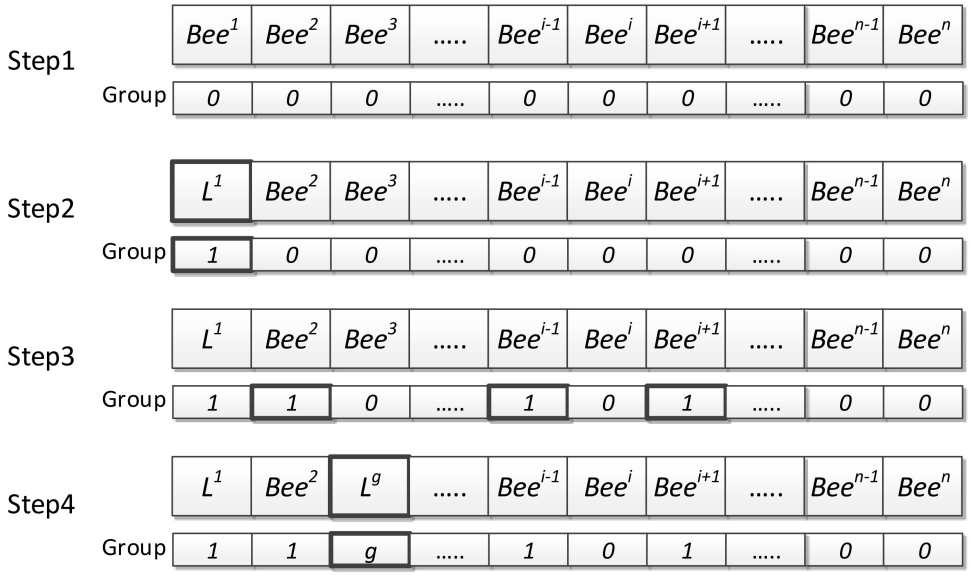


Figure 6. Dynamic grouping method.

Step 1: Sorting

Sort the bees from the best to worst and initiate 0 to all the groups.

Step 2: Calculate the threshold value of similarity.

Find the fittest bee among the ungrouped bees. Then, select the fittest bee as the new group leader, name the group g, and calculate the fitness threshold.

$$DIS^g = \sum_{i=1}^{SN} \sum_{j=1}^{D} \sqrt{\left(Leader_j^g - Bee_j^i\right)^2}, \quad if\ Bee^i\ is\ ungrouped \tag{15}$$

$$FIT^g = \sum_{i=1}^{SN} \left| Fit(Leader^g) - Fit\left(Bee^i\right) \right|, \quad if\ Bee^i\ is\ ungrouped \tag{16}$$

$$Average\_Distance(ADIS^g) = \frac{DIS^g}{NC} \tag{17}$$

$$Average\_Fitness(AFIT^g) = \frac{FIT^g}{NC} \tag{18}$$

where $ADIS^g$ and $AFIT^g$ are the distance and fitness thresholds, respectively, in group g, $Leader_j^g$ is the leader location in group g, $Fit(Leader^g)$ is the fitness value of leader in group g, SN is the total number of bees, is the dimension, and NC is the total number of bees in group 0.

Step 3: Evaluate the similarity

Calculate the distance $\left(Dis^i\right)$ and fitness Fit^i between ungrouped bee Bee^i and the leader.

$$Dis^i = \sum_{j=1}^{D} \sqrt{\left(Leader_j^g - Bee_j^i\right)^2} \tag{19}$$

$$Fit^i = \left| Fit(Leader^g) - Fit\left(Bee^i\right) \right| \tag{20}$$

If $Dis^i < ADIS^g$ and $Fit^i < AFIT^g$, then the bee Bee^i is similar to the group leader. Place them into the same group and change the group number to g.

Step 4: Checking

Check whether there are ungrouped bees. If YES, then go back to step 1 and select the fittest bee in the ungrouped bees as the new group leader and repeat step 1 to step 3. If NO, all the bees are grouped and grouping is complete.

2. Employed Bees Phase

In the traditional ABC, employed bees randomly select their food source searching direction. In the proposed method, the searching direction is improved. The global best solution is used in the movement equation to enable the employed bees to move to the better solution. Moreover, the proposed method maintains the random search mechanism.

$$v_j^i = x_j^i + \varnothing 1_j^i \left(x_j^i - x_j^{best} \right) + \varnothing 2_j^i \left(x_j^i - x_j^k \right) \tag{21}$$

where x_j^{best} is the best solution in the group and $\varnothing 1_j^i$ and $\varnothing 2_j^i$ are random numbers between $[-1, 1]$.

3. Onlooker Bees Phase

The grouping strategy alters the traditional onlooker bees' searching method to the leader (employed bees). The assigned leader must guide the teammates (onlooker bees) to search for a food source.

$$v_j^i = x_j^i + \varnothing_j^i \left(x_j^i - x_j^{Leader} \right) \tag{22}$$

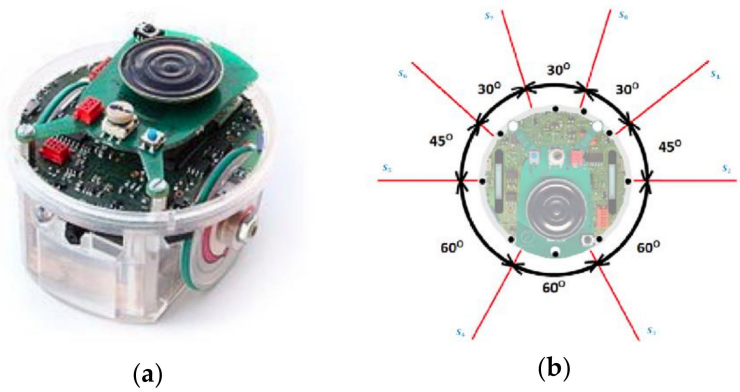
where $\varnothing_j^i$ is a random number between [−1, 1] and x_j^{Leader} is the location of the leader in the group.

4. Control of the Mobile Robot

In this paragraph, the mobile robot is introduced first and then the method of wall-following control is explained in the following section.

4.1. Description of the Mobile Robot

E-puck mobile robots, new small-sized mobile robots manufactured by EPFL, were used in this research. As displayed in Figure 7a, the robot was controlled by the DSPIC processor. Several standard configurations such as a proximal infrared sensor, voice sensor, accelerometer, and camera are included in the robot. The mobile robots connect to each other through Bluetooth and use Bluetooth to communicate with the computers. Nowadays, e-puck mobile robots are widely used. Applications for e-buck include mobile mechanical engineering, real-time programs, interpolation systems, signal transmission, image transmission, combination of sounds and images, human–computer interaction, and robot internal communication.



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

Figure 7. (**a**) E-puck mobile robot; (**b**) E-puck 360° infrared sensors.

Figure 7b shows that e-puck is a two-wheeled mobile robot with eight 360° surrounding infrared sensors from S_0–S_7. The diameter of e-puck is 7 cm, height is 5 cm, and maximum speed is 15 cm/s. The maximum distance between the mobile robot and nearby objects is 6 cm, and the minimum distance is 1 cm.

4.2. Wall-Following Control of the Mobile Robot

In this study, RL was used to implement the wall-following behavior. The robot can evaluate the performance by defining the suitable fitness functions in the training environment even when predefined rules and pre-specified training data are not provided. The wall-following control learning flowchart is displayed in Figure 8.

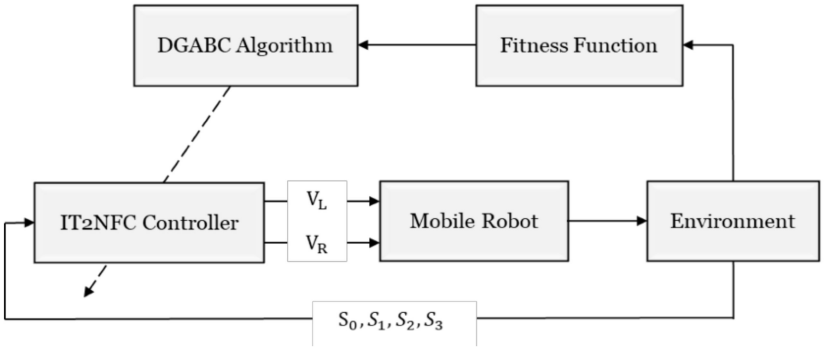


Figure 8. Flowchart of wall-following control learning.

The IT2NFC controller has four input signals (S_0, S_1, S_2, and S_3) and two output signals (V_L and V_R). The input signal S_i denotes the distance from the *i*th infrared sensor and the output signals represent the turning speeds of the left and right wheels.

Figure 9 exhibits the training environment. The goal is to enable the mobile robot to adapt to any type of unknown environments including straight lines, sharp corners, obtuse corners, smooth corners, and U-curve corners.

Figure 9. Training environment.

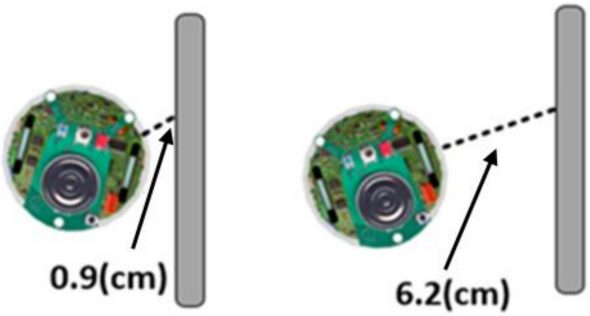
To prevent the robot from moving far away from the walls or colliding with the obstacles, three terminate actions were designed in this study.

Action 1:

The total moving distance must be more than the preset maximum distance (the distance of navigating the training environment), which means that the robot successfully detours the training environment once.

Action 2:

As Figure 10a indicates, the detected distance from one of the sensors is less than 1 cm, which means that the robot hits the wall.



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

Figure 10. (**a**) Robot hits the wall; (**b**) Robot deviates from the wall.

Action 3:

As Figure 10b illustrates, the detected distance from one of the side sensors is more than 6 cm, which means that the robot is deviating from the wall.

To evaluate the performance during the learning, the fitness function is proposed in this paper. The fitness function has three sub-fitness functions, namely the total distance of the robot, the average distance between the robot and the wall, and the parallel degree between the robot and the wall (SF_1, SF_2, and SF_3).

1. SF_1

If the robot's moving distance R_{dis} is close to the preset distance R_{total}, then the robot is close to detouring the training environment.

$$SF_1 = R_{total} - R_{dis} \tag{23}$$

If the robot's moving distance is more than the preset distance R_{total}, then the robot traverses the training environment successfully and $SF_1 = 0$.

2. SF_2

To ensure the robot and the wall stay at a constant distance, the distance $WD(t)$ between the robot's side sensor $S_2(t)$ and the wall is calculated in every time step, as indicated in Figure 11a.

$$WD(t) = |S_2(t) - d_{wall}| \tag{24}$$

$$SF_2 = \frac{\sum_{t=1}^{T_{stop}} WD(t)}{T_{stop}} \tag{25}$$

where d_{wall} is the expected distance between the wall and T_{stop} is time step. In this study, d_{wall} was selected as 4 cm.

If the robot stays at a constant distance with the wall, then $SF_2 = 0$.

3. SF_3

To ensure the robot remains parallel with the wall, the law of cosines is used on the distances from the robot's front right sensor and the side sensor to calculate the length $x(t)$. Like Figure 11b.

$$RS_1 = r + S_1, RS_2 = r + S_2 \tag{26}$$

$$x(t) = \sqrt{RS_1{}^2 + RS_2{}^2 - 2RS_1RS_2\cos(45^\circ)} \tag{27}$$

where r is the radius of the robot, RS_1 is the distance from the sensor S_1, and RS_2 is the distance from the sensor S_2. If the side sensor S_2 is vertical to the wall, then $x(t)$ is equal to and $SF_3 = 0$.

$$SF_3 = \frac{\sum_{t=1}^{T_{stop}} |RS_2 - x(t)|}{T_{stop}} \tag{28}$$

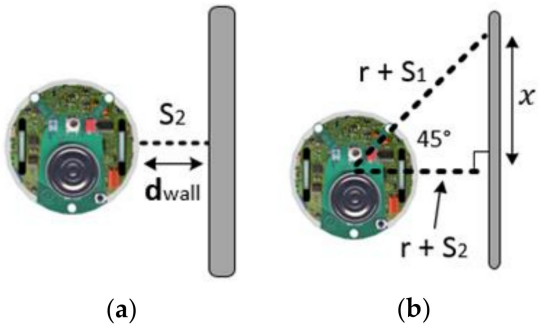


Figure 11. (**a**) Average distance; (**b**) Parallel degree.

Therefore, the fitness function $\overline{F}$ can be defined as the following:

$$\overline{F} = \frac{1}{1 + (SF_1 + SF_2 + SF_3)} \tag{29}$$

5. Cooperative Load-Carrying and Navigation Control of Mobile Robots

In this section, the cooperative load-carrying control of multi-mobile robots is depicted. First, set two mobile robots in the same direction and place the load item upon them. The front robot is the leader and the other is the follower. The distance between the two robots is 15 cm. Figure 12 demonstrates that the leader explores the environment and guides the follower. The follower assists the leader in carrying the item.

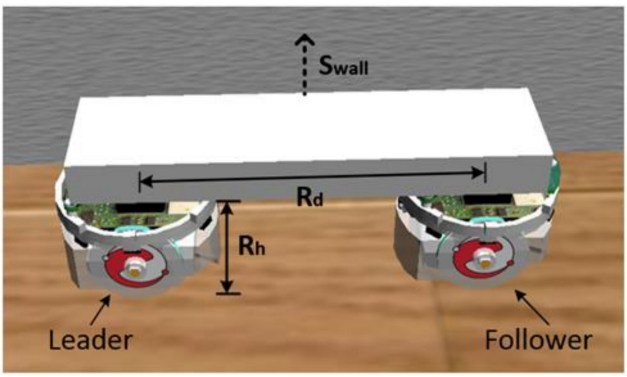


Figure 12. Cooperative load-carrying of two robots.

5.1. Cooperative Load-Carrying Wall-Following Control of Mobile Robots

The wall-following control method is used for the leader and the follower. The auxiliary controller is attached to the follower. Figure 13 presents the flowchart of cooperative load-carrying wall-following control. The auxiliary controller has five inputs, which consist of the distances detected from the four sensors on the follower (S_0, S_1, S_2, and S_3) and the distance between the leader (R_d). The two outputs (V_L and V_R) are the speeds from left and right wheels, respectively. Figure 14 exhibits the

training environment that includes straight lines, sharp corners, obtuse corners, smooth corners, and U-curve corners.

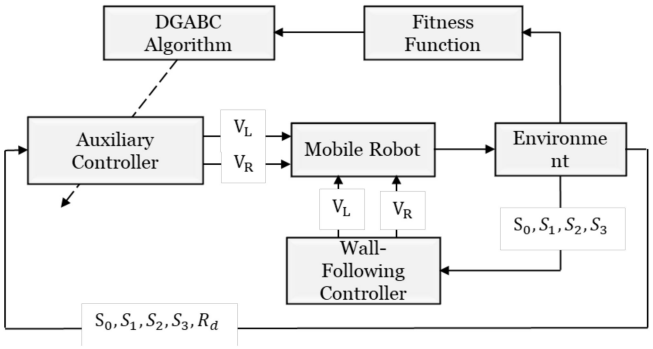


Figure 13. Cooperative load-carrying wall-following control.

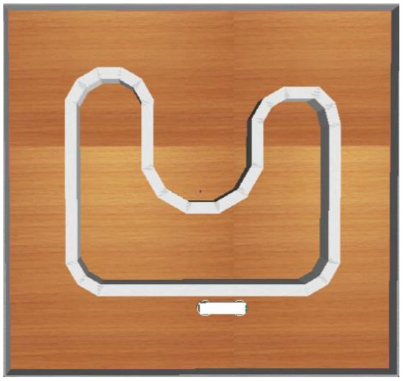


Figure 14. Training environment for cooperative load-carrying wall-following control.

In this control method, the leader only follows the wall and the follower assists the leader to carry the item and adjusts its own output from the auxiliary controller to match the leader's output. To prevent the follower and leader from staying at an unsafe distance or failing the mission, the following five terminations are designed in this study.

Case 1

If one of the sensors on the follower reaches a distance less than 1 cm, then the follower hits the wall.

Case 2

If the follower's side sensor reaches a distance more than 6 cm, then the follower deviates from the wall.

Case 3

If Rd is less than 10 cm or more than 20 cm, then the leader and the follower are too close to each other or too far away from each other, respectively. Improper value of Rd might cause the leader to make a wrong detection or the item to fall.

Case 4

If the sensor under the item is smaller than the robot's height (R_h), then the task fails.

Case 5

If the side sensor detects a distance less than 1 cm or more than 7.5 cm, then the item is too close to the wall or too far away from the wall.

In practical testing experiments, the side and bottom sensors are removed. When the robots encounter a termination case, the fitness value $F(\cdot)$ is calculated by the robot's time steps. Then, the robots return to the initial point and restart the program again until they traverse the training environment successfully.

$$F(\cdot) = T_{stop}/5000 \tag{30}$$

A maximum time step of 5000 is allocated in this study. The longer the robots move, the higher the fitness value gets.

5.2. Cooperative Load-Carrying Navigation Control of Mobile Robots

Known and unknown environments are the two types of environments in existence. A navigation control method to effectively assist mobile robots to navigate unknown environments is proposed in this paper. The behavior mode manager was used to switch between the goal-oriented mode (GOM) and the wall-following mode (WFM). The robots were initialized in the GOM on detecting the obstacles, then switched to the WFM. Otherwise, the robots remain in the GOM.

- GOM

In an unknown environment, the goal position is the only information available to the robots. Therefore, Figure 15 shows an angle θ_{RG} and the robot moves toward the goal.

$$\theta_{RG} = \theta_{Robot} - \theta_{Goal} \tag{31}$$

where θ_{Robot} is the angle between the robot and the axis x and θ_{Goal} is the angle between the goal and the axis x.

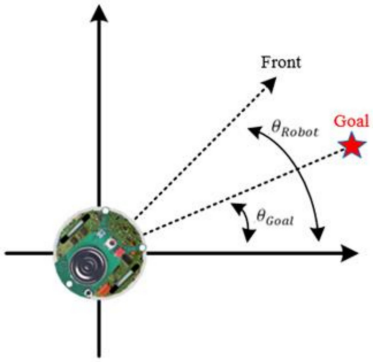


Figure 15. Angle θ_{RG}.

Figure 16, in the GOM, the follower moves in the same direction and with the same speed with the leader.

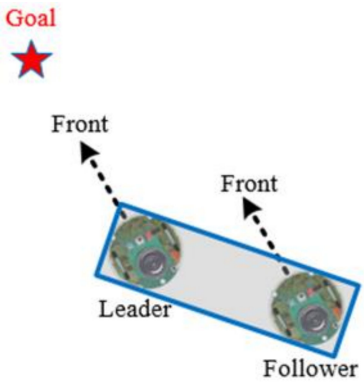


Figure 16. GOM.

- Behavior mode manager

Figure 17 shows the three areas of the robot O_1, O_2, and O_3. Depending on which side of the sensors detect the goal (O_i) or which sensors detect the obstacles (S_i), the behavior mode manager determines which side of the wall the robots should follow.

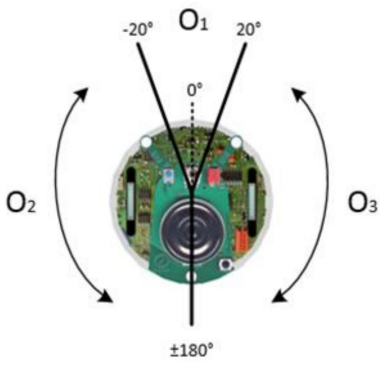


Figure 17. The area distribution of a robot.

Left-side wall-following:

The goal is in O_1 and S_7 or S_6 senses the obstacles.
The goal is in O_2 and S_7, S_6, and S_5 sense the obstacles.

Right-side wall-following:

The goal is in O_1 and S_0 or S_1 senses the obstacles.
The goal is in O_3 and S_0, S_1, and S_2 sense the obstacles.

Prior to the robot's switch to the WFM, the behavior mode manager deduces which robot detects the obstacles first. Figure 18 displays that if the leader detects the obstacles, then the leader will pre-rotate first to stay parallel to the wall. The leader then sends θ_L to the follower, and the follower rotates $\pi - \theta_L$ degrees. Once both robots are within a safe distance and parallel to the wall, the WFM is executed. After they pass the obstacles, the GOM switches back again. Figure 19 analyses the actions of the follower.

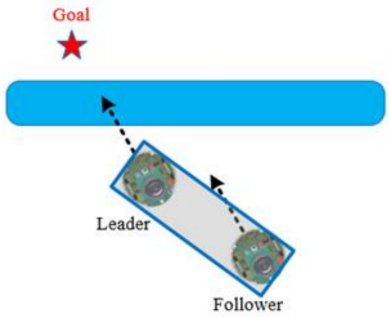


Figure 18. Leader detects the obstacle.

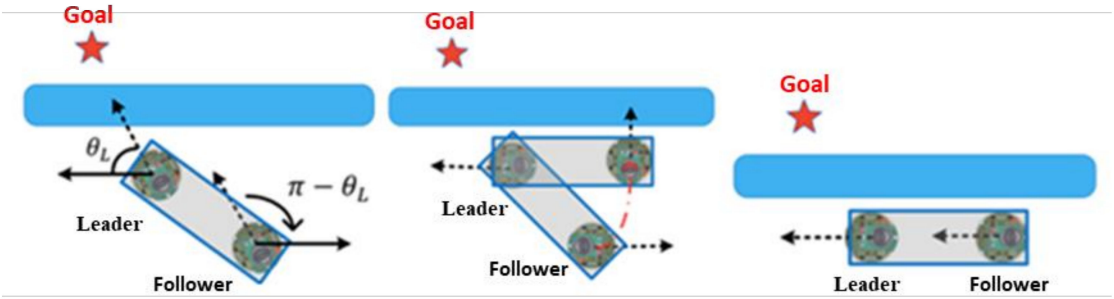


Figure 19. Actions of the follower.

Figure 20 reveals that if the follower detects the obstacle, then both the robots switch to the WFM directly.

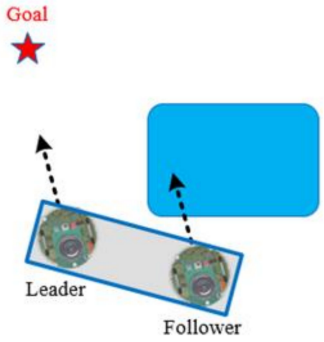


Figure 20. Follower detects the obstacle.

As shown in Figure 21, during the WFM, if the obstacle is in the follower's O_1 area and the right-side sensor S_1 detects a distance larger than 6 cm (if the robots are in the left-side WFM, then the distance is detected by the left-side sensor S_6), then the item on the robots is past the obstacle. The behavior mode manager can switch the WFM back to the GOM.

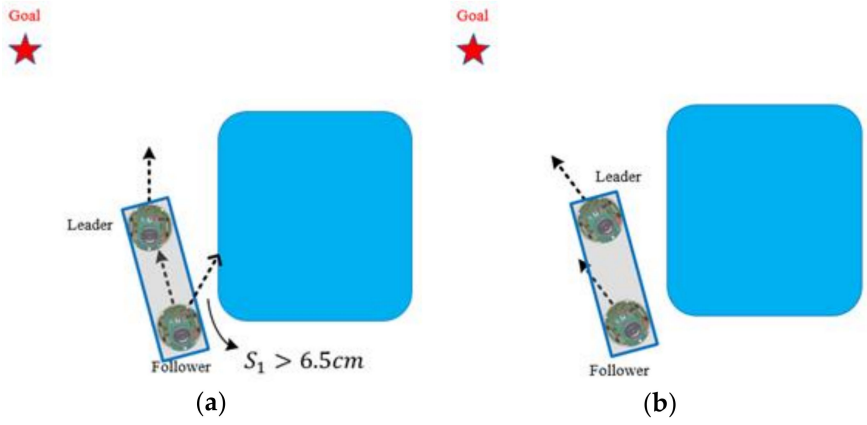


Figure 21. (**a**) Distance between follower and obstacle is larger than 6.5 cm; (**b**) Robots switch the WFM back to the GOM.

6. Experimental Results

Simulation is very important in the robot design process, and the developed algorithm can be quickly verified. At the same time, for robot learners, simulation tools can greatly reduce the cost of learning. The robot simulation platform integrates a physics engine which can calculate motion, rotation and collision according to the physical properties of the object. The simulation tools commonly used in robots are Gazebo, V-REP, and Webot. Table 1 shows the features of various

robot simulator software. In Table 1, the V-Rep and Webots support Linux, mac OS, and Windows development environments, whereas the Gazebo only supports Linux development environment. The V-Rep simulation software has less functional support than Gazebo and Webots. Therefore, in this study Webot is used as a robot simulation platform to verify the proposed IT2NFC controller based on DGABC learning algorithm and perform the WFM control and cooperative load-carrying of mobile robots.

Table 1. Features of various robot simulator software.

Software	Platforms Supported	Dynamic Collision Avoidance	Relative End Effectors	Off-Line Programming	Real-Time Streaming Control of Hardware
Gazebo	Linux	Yes	Yes	Yes	Yes
V-Rep	Linux, mac OS, Windows	No	No	No	No
Webots	Linux, mac OS, Windows	Yes	Yes	Yes	Yes

6.1. Results of WFM Control

The performance and stabilization for DGABC were compared with those for other algorithms. Best, worst, average, standard deviation (STD), and the number of successful runs after ten runs and the average executing times are mentioned in Table 2. The number of successful runs demonstrates that the mobile robots can traverse the training environment one time successfully from the start point to the goal during the simulation. And the learning curves of each algorithms are shown in Figure 22. Table 2 and Figure 23 are under the same conditions. In Table 2, the proposed method exhibited a higher learning effect than improved ABC, ABC, and DE. In addition, the average executing time of proposed method is shorter than those of other methods.

Table 2. Performance comparison of various algorithms.

Evaluation / Algorithm	Fitness Value				Number of Success Runs	Average Executing Times(s)
	Best	Worst	Average	STD		
PSO [24]	0.912	0.905	0.908	0.003	7	227
DE [26]	0.872	0.858	0.864	0.006	8	207
ABC [27]	0.895	0.857	0.876	0.016	5	230
IABC [28]	0.926	0.905	0.917	0.009	8	199
DGABC	**0.895**	**0.881**	**0.889**	**0.006**	**10**	**183**

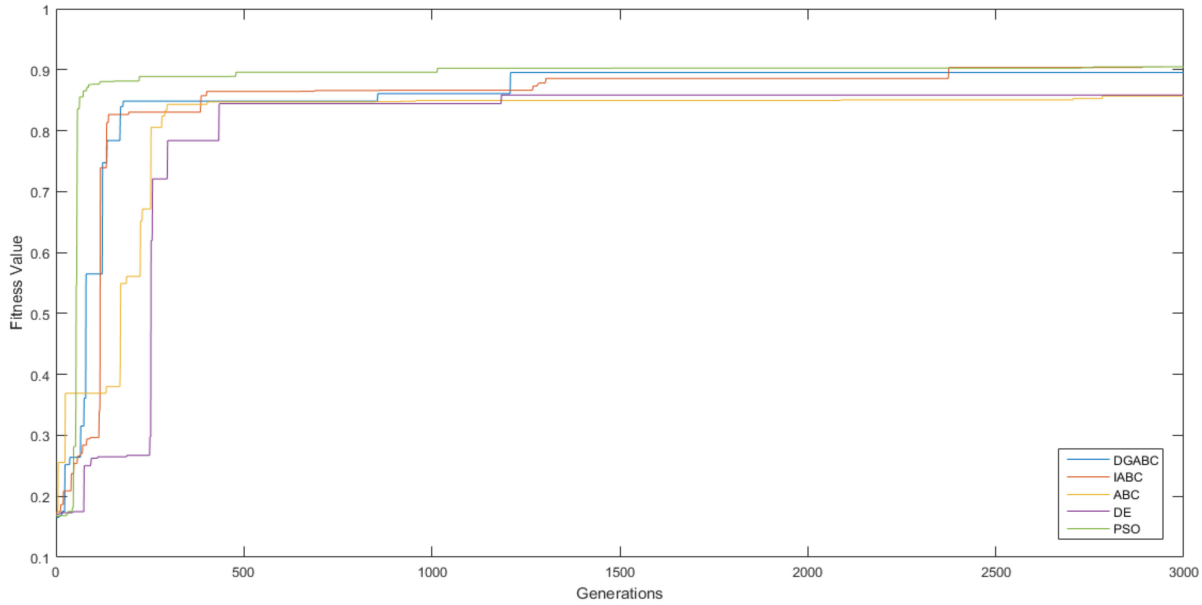


Figure 22. Learning curves of wall-following control in various algorithms.

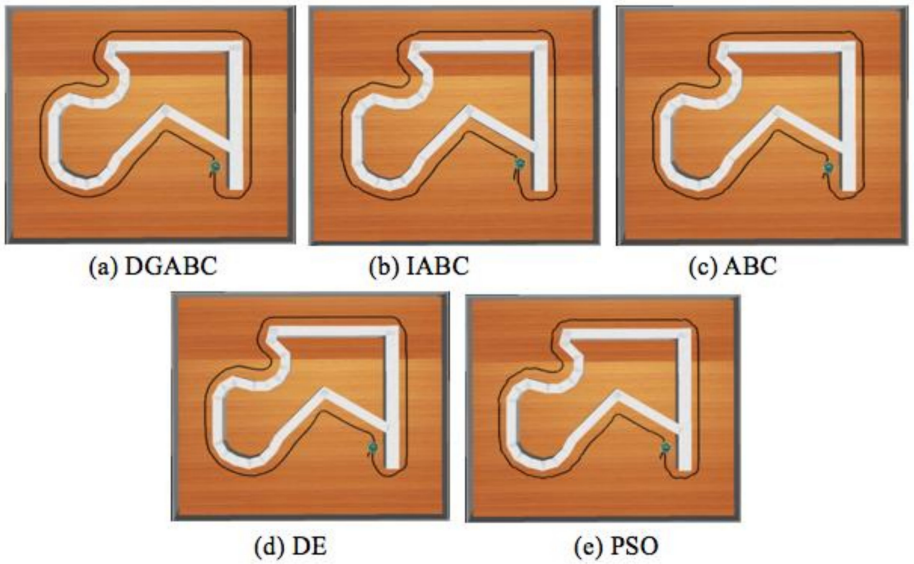


Figure 23. Moving path of various algorithms in the training environment.

In addition, we have also compared the proposed IT2NFC (type-2 fuzzy sets) based on DGABC with the IT1NFC (type-1 fuzzy sets) based on DGABC. Table 3 shows the performance comparison of IT1NFC and IT2NFC. The experimental results show the performance of IT2NFC (type-2 fuzzy sets) is better than the IT1NFC (type-1 fuzzy sets)"

Table 3. Performance comparison of IT1NFC and IT2NFC.

Evaluation / Algorithm	Fitness Value				Number of Success Runs	Average Executing Times (s)
	Best	Worst	Average	STD		
IT1NFC (type-1)	0.889	0.856	0.878	0.011	9	198
IT2NFC (type-2)	0.895	0.881	0.889	0.006	10	183

6.2. Results of Cooperative Load-Carrying WFM Control

The results revealed that the trained control method was implemented for cooperative load-carrying mobile robots. Table 4 compares the performance of the proposed WFM with that of other algorithms in the simulation. The better performance and higher stabilization for cooperative load-carrying WFM control in DGABC is demonstrated in Figure 24. Figure 25 shows the moving path of two mobile robots in the training environment. The PSO did not display any advantage in cooperative load-carrying WFM control. Therefore, the proposed DGABC method exhibited better performance in this control.

Table 4. Performance of cooperative load-carrying WFM control.

Evaluation / Algorithm	Fitness Value				Number of Successful Runs	Average Executing Times (s)
	Best	Worst	Average	STD		
PSO [24]	0.522	0.160	0.325	0.131	5	193
DE [26]	1	0.446	0.632	0.245	6	172
ABC [27]	0.326	0.1	0.195	0.087	3	179
IABC [28]	0.958	0.247	0.583	0.250	7	165
DGABC	1	0.764	0.921	0.105	10	141

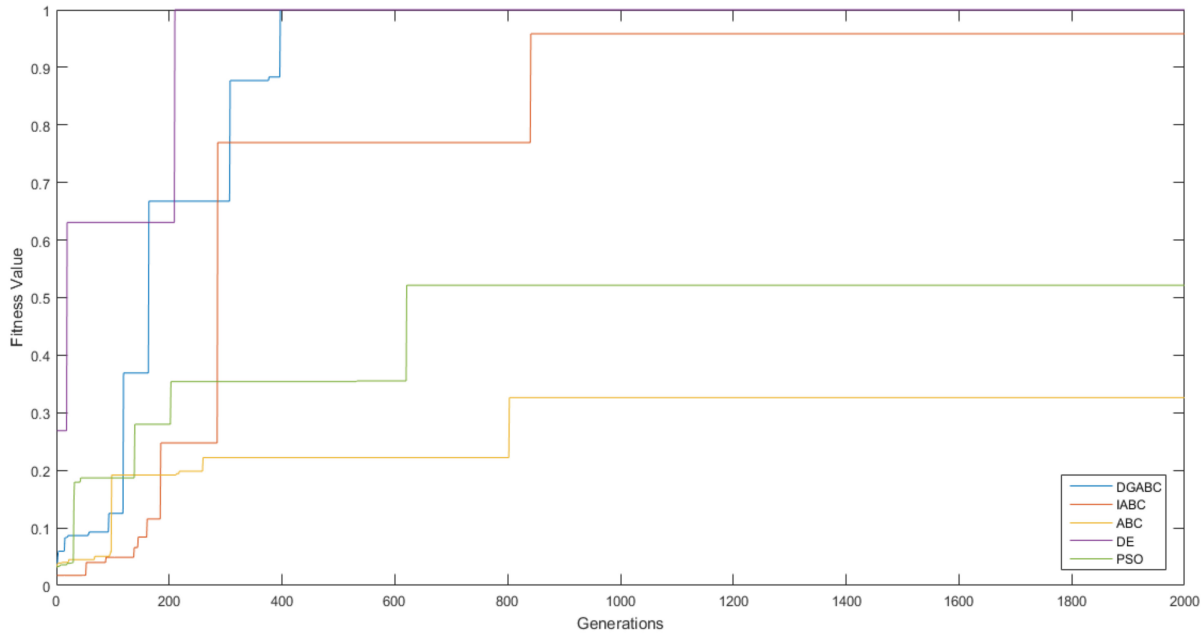


Figure 24. Learning curves of various algorithms.

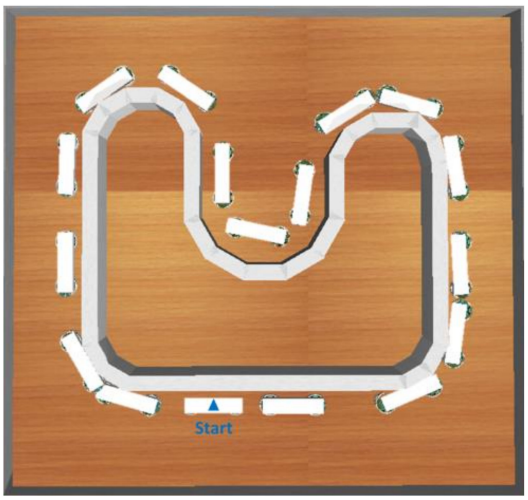
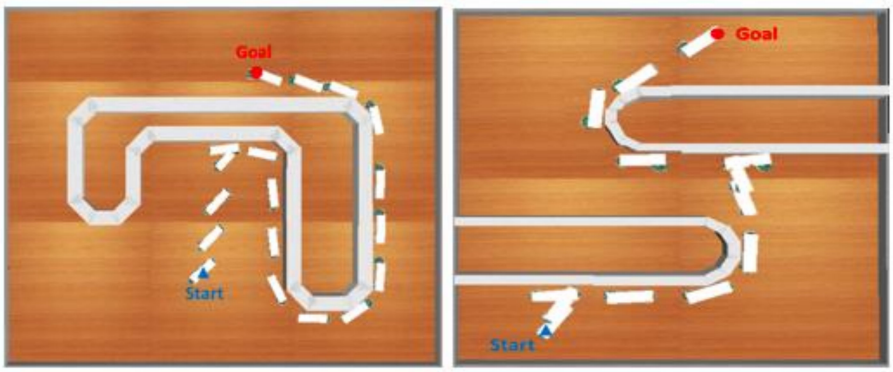


Figure 25. Moving path of the proposed method.

6.3. Results of Cooperative Load-Carrying Navigation Control

Implementation of the proposed method in cooperative load-carrying mobile robots in unknown environments is demonstrated in Figure 26. In this experiment, the average distance between two mobile robots (RAD) and the average distance between the wall and the follower robot (FAD) were the two critical standards for evaluating the performance of the proposed method. As shown in Table 5, if the RAD value is too large, the two mobile robots do not stay within the safe distance, which might cause the object to fall. On the other hand, if the FAD value is too large or too small, then the WFM does not perform satisfactorily when the mobile robots are turning around the corner. This might cause the object to shift during the mission, the object to fall, and thus, the mission to fail.



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

Figure 26. Moving path of cooperative load-carrying navigation control in (**a**) testing environment 1 and (**b**) testing environment 2.

Table 5. Performance of cooperative load-carrying navigation control.

Evaluation / Algorithm	Testing Environment 1		Testing Environment 2	
	RAD (cm)	FAD (cm)	RAD (cm)	FAD (cm)
DGABC	15.64	3.72	17.08	3.81

6.4. Discussion

In this study, we have successfully implemented the cooperative load-carrying task of two mobile robots by using the proposed IT2NFC-based DGABC. Next, the proposed method will be extended to

three and more mobile robots for the cooperative load-carrying task. For example, Figure 27 shows the cooperative load-carrying task of three mobile robots. First, the wall-following control method of the cooperative load-carrying task is used for the leader, the follower-1, and the follower-2. Second, the auxiliary controller is attached to the followers based on the description in Section 5.1. Finally, the behavior mode manager in navigation control is also used to switch between the GOM and the WFM in Section 5.2. Therefore, the cooperative load-carrying task of three mobile robots would be implemented. According to the aforementioned steps, the proposed method can be successfully extended to the cooperative load-carrying task of three and more mobile robots.

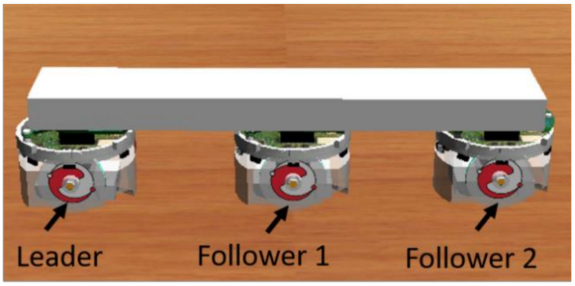


Figure 27. Cooperative load-carrying of three robots.

7. Conclusions

The proposed IT2NFC-based DGABC improves the search capacity and shortens the convergence speed for avoiding falling into local optimal solutions. Using the proposed method, the mobile robots can develop the controller adaptively because DGABC does not need any predefined rule set nor does it require pre-specified training data. According to the environmental situations, mobile robots use the behavior mode manager to switch between WFM and GOM. Additionally, the pre-rotate mechanism ensures the follower robot follows the wall perfectly and allows the leader robot to complete the cooperative load-carrying task in unknown environments.

The cooperative load-carrying task is so complex for the robots that several factors, such as the robot speed, the object payload, and the working environment, should be taken into consideration. Therefore, we focus on implementing the cooperative load-carrying task of two mobile robots in this study. In the future work, we will consider implementing the cooperative load-carrying task of two and more real mobile robots in the unknown environments.

Author Contributions: Formal analysis, C.-H.L.; Methodology, C.-H.L., S.-H.W. and C.-J.L.; Software, C.-J.L.; Supervision, S.-H.W. and C.-J.L.; Writing—original draft, C.-H.L. and C.-J.L.

Funding: This research was funded by the Ministry of Science and Technology of the Republic of China, Taiwan grant number MOST 106-2221-E-167-016.

Acknowledgments: The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan for financially supporting this research under Contract No. MOST 106-2221-E-167-016.

Conflicts of Interest: The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

1. Paul, P.; Raymond, L.; Maurizio, P. Robotic Fish: Design and Characterization of an Interactive iDevice-Controlled Robotic Fish for Informal Science Education. *IEEE Robot. Autom Mag.* **2015**, *22*, 86–96.
2. Christopher, L.; Andrew, E.; Christopher, M.; Adam, W.T.; Tristan, P. Autonomous Sweet Pepper Harvesting for Protected Cropping Systems. *IEEE Robot. Autom. Lett.* **2017**, *2*, 872–879.
3. Michail, P.; Konstantinos, K.; Christos, D.; Georgios, S. Design of an Autonomous Robotic Vehicle for Area Mapping and Remote Monitoring. *Int. J. Comput. Appl.* **2017**, *12*, 36–41.
4. Maurizio, F.; Junichi, T.; Goldie, N. Promoting Interactions Between Humans and Robots Using Robotic Emotional Behavior. *IEEE Trans. Cybern.* **2015**, *46*, 2911–2923.

5. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [CrossRef]
6. Pawlak, Z. Rough set theory and its Applications. *J. Telecommun. Inf. Technol.* **2002**, *3*, 7–10.
7. Molodtsov, D. Soft set theory-First results. *Comput. Math. Appl.* **1999**, *37*, 19–31. [CrossRef]
8. Algabri, M.; Mathkour, H.; Ramdane, H.; Alsulaiman, M. Comparative study of soft computing techniques for mobile robot navigation in an unknown environment. *Comput. Hum. Behav.* **2015**, *50*, 42–56. [CrossRef]
9. Lee, C.L.; Lin, C.J.; Lin, H.Y. Smart Robot Wall-Following Control Using a Sonar Behavior-based Fuzzy Controller in Unknown Environments. *Smart Sci.* **2017**, *5*, 160–166. [CrossRef]
10. Fathinezhad, F.; Derhami, V.; Rezaeian, M. Supervised fuzzy reinforcement learning for robot navigation. *Appl. Soft Comput.* **2016**, *40*, 33–41. [CrossRef]
11. 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. Autom.* **2016**, *1*, 1–11.
12. Mendel, J.M. Advances in type-2 fuzzy sets and systems. *Inf. Sci.* **2007**, *177*, 84–110.
13. Mendel, J.M. Type-2 fuzzy sets and systems: An overview. *IEEE Comput. Intell. Mag.* **2007**, *2*, 20–29. [CrossRef]
14. Zaheer, S.A.; Choi, S.H.; Jung, C.Y.; Kim, J.H. A modular implementation scheme for nonsingleton type-2 fuzzy logic systems with input uncertainties. *IEEE/ASME Trans. Mech.* **2015**, *20*, 3182–3193. [CrossRef]
15. 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]
16. Nguyen, T.; Khosravi, A.; Creighton, D.; Nahavandi, S. Medical data classification using interval type-2 fuzzy logic system and wavelets. *Appl. Soft Comput.* **2015**, *30*, 812–822. [CrossRef]
17. Zarandi, M.H.F.; Rezaee, B.; Turksen, I.B.; Neshat, E. A type-2 fuzzy rule based expert system model for stock price analysis. *Expert Syst. Appl.* **2009**, *36*, 139–154. [CrossRef]
18. Melin, P.; Castillo, O. A review on type-2 fuzzy logic applications in clustering, classification and pattern recognition. *App. Soft Comput.* **2014**, *21*, 568–577. [CrossRef]
19. Tai, K.; El-Sayed, A.-R.; Biglarbegian, M.; Gonzalez, C.I.; Castillo, O.; Mahmud, S. Review of recent type-2 fuzzy controller applications. *Algorithms* **2016**, *9*, 39. [CrossRef]
20. Bay, O.F.; Yatak, M.O. Type-2 fuzzy logic control of a photovoltaic sourced two stages converter. *J. Intell. Fuzzy Syst.* **2018**, *35*, 1103–1117. [CrossRef]
21. Karnik, N.N.; Mendel, J.M. Type-2 fuzzy logic systems: Type-reduction. In Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics, San Diego, CA, USA, 14 October 1998; pp. 2046–2051.
22. Liang, Q.; Mendel, J.M. Interval type-2 fuzzy logic systems: Theory and design. *IEEE Trans. Fuzzy Syst.* **2000**, *8*, 535–550. [CrossRef]
23. 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]
24. Kaveh, A. *Advances in Metaheuristic Algorithms for Optimal Design of Structures*; Springer: Cham, Switzerland, 2017; pp. 11–43.
25. 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; pp. 1470–1477.
26. Zheng, L.M.; Zhang, S.X.; Tang, K.S.; Zheng, S.Y. Differential evolution powered by collective information. *Inf. Sci.* **2017**, *399*, 13–29. [CrossRef]
27. Barbosa, H. *Ant Colony Optimization Techniques and Applications*; Intech Open: Rijeka, Croatia, 2013; ISBN 978-953-51-1001-9.
28. Tsai, P.W.; Pan, J.S.; Liao, B.Y.; Chu, S.C. Enhanced artificial bee colony optimization. *Int. J. Innov. Comput. Inf. Control* **2009**, *5*, 5081–5092.