Sensors and Materials, Vol. 37, No. 7 (2025) 3219–3235
MYU Tokyo

S & M 4114

Obstacle Avoidance and Navigation of Unmanned Ground Vehicles Using a Type-2 Fuzzy Neural Controller Based on Improved Mantis Search Algorithm

Cheng-Jian Lin,[1*] Chun-Yi Pan,[2] Bing-Hong Chen,[2]
Chen-Hao Huang,[1] and Kuang-Hui Tang[3]

[1]Department of Computer Science & Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan
[2]Ph.D. Program, Prospective Technology of Electrical Engineering and Computer Science, National Chin-Yi University of Technology, Taichung 411, Taiwan
[3]Department of Electronic Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan

(Received January 27, 2025; accepted June 26, 2025)

Keywords: unmanned ground vehicle, fuzzy neural controller, mantis search algorithm, type-2 fuzzy set, navigation control

In this study, we use the type-2 fuzzy neural controller (T2FNC) based on the improved mantis search algorithm (IMSA) for navigation and obstacle avoidance applications of unmanned ground vehicles (UGVs) with differential wheels in unknown environments. In unknown environments, a light detection and ranging sensor is used to capture distance information between UGVs and the surrounding environment. The T2FNC has a five-layer architecture. The first to fifth layers are the input, fuzzified, rule, order reduction process, and output layers, respectively. In the T2FNC, we use the IMSA to adjust the parameters in the network. In addition, the simulated annealing reciprocal local search algorithm is proposed to prevent the traditional MSA from falling into the local optimal solution. Experimental results indicate that the fitness value of the proposed IMSA is 0.983452. Compared with the traditional MSA algorithm, the movement distance and movement time of the proposed T2FNC with IMSA are shortened by 4.5 and 4.14%, respectively. In addition, the experimental results show that the proposed method can have excellent obstacle avoidance and navigation capabilities in unknown environments.

1. Introduction

Recently, unmanned ground vehicles (UGVs) have been widely used in various fields, such as military reconnaissance, logistics, and environmental monitoring. Wang *et al.*(1) combined UGVs with geographic information systems to realize the autonomous navigation of UGVs in an urban traffic environment. Mammarella *et al.*(2) proposed a multistage method and coordinated multiple UGVs to perform agricultural tasks in a vineyard. Appelqvist *et al.*(3) demonstrated the

*Corresponding author: e-mail: cjlin@ncut.edu.tw
https://doi.org/10.18494/SAM5582

UGV technology plan, which includes the design of vehicle platform instrumentation, software architecture, communication connections, and a human–machine interface. Wang *et al.*(4) developed an autonomous collaborative exploration system that utilizes UGVs for exploration operations. Rodríguez *et al.*(5) proposed a collaborative task allocation algorithm for multiple UGVs and a cooperative process for performing search and rescue missions in a mountainous environment.

The navigation performance of the UGV is related to the various environmental sensors it is equipped with. These sensors are responsible for collecting environmental information and transmitting it to the main control computer for processing. Liu *et al.*(6) pointed out that there are some limitations in the performance of different sensors in different application scenarios. Radar(7) detects objects using radio waves. Although it has stable performance under severe weather conditions, it cannot provide the precise contour information of objects. Ultrasonic sensors(8) are often used for close-range detection and obstacle avoidance because of their small size, low cost, and being unaffected by weather. However, owing to the short detection range of ultrasonic sensors, they are difficult to apply in large-scale environmental monitoring. The camera(9) can capture images in the environment, but it lacks depth information. A stereo camera(10) provides depth information but it is affected by lighting conditions and requires high computing resources. Compared with the above-mentioned sensors, light detection and ranging (LiDAR)(11) can operate under various climatic conditions and provide high-resolution environmental depth information. It has long-range detection and a wide field of view, enabling UGVs to conduct precise navigation.

For UGV control, a proportional-integral-derivative controller(12) or sliding mode controller(13,14) is often used. These controllers maintain system stability by adjusting the parameters. However, parameter values are usually determined by empirical methods, which limits flexibility. Therefore, adaptive controllers(15–17) that can automatically adjust parameters in accordance with system changes have emerged. However, as the requirements for system stability and flexibility increase, researchers are gradually turning to fuzzy controllers.(18) Fuzzy controllers are divided into type-1 and type-2 fuzzy controllers. Castillo *et al.*(19) pointed out that the type-2 fuzzy controller can handle more uncertain input and is better than the type-1 fuzzy controller.

In machine learning techniques, the backpropagation (BP)(20) algorithm is often used to adjust the parameters of the controller. However, the BP algorithm has a local optimal solution problem. To overcome this problem, heuristic algorithms in machine learning techniques have become the main method for controller parameter optimization. In the development process of heuristic algorithms, researchers have designed a variety of algorithms inspired by biological evolution, natural physical laws, group animal behavior, and human social behavior. For example, the genetic algorithm (GA)(21) simulates natural selection and genetic mechanisms. The simulated annealing algorithm(22) is a random search based on the metal annealing process. Particle swarm optimization (PSO)(23,24) imitates the search behavior of group animals, and the ant colony optimization algorithm(25) simulates the food search process of an ant colony. Because of the randomness and nondeterminism of these heuristic algorithms, it is easy to fall into the local optimal solution during the global search process. In addition, when these algorithms face highly complex multidimensional optimization problems, their convergence is slow. To overcome these problems, Abdel-Basset *et al.*(26) proposed the mantis search algorithm (MSA) to simulate the hunting process of mantises. Compared with other heuristic algorithms, MSA shows significant advantages in performance indicators and exhibits excellent performance on specific engineering problems.(27–29)

In this study, we use the type-2 fuzzy neural controller (T2FNC) based on the improved mantis search algorithm (IMSA) for navigation and obstacle avoidance applications of UGVs with differential wheels in unknown environments. The major contributions of this study are as follows.

- In unknown environments, a LiDAR sensor is used to capture distance information between UGVs and the surrounding environment.
- The proposed T2FNC has a five-layer architecture. The first to fifth layers are the input, fuzzified, rule, order reduction process, and output layers, respectively.
- The IMSA is proposed to adjust the parameters in the T2FNC. In addition, the simulated annealing reciprocal local search algorithm (SARLSA) is also proposed to prevent the traditional MSA from falling into the local optimal solution. This concept of the proposed IMSA is developed for the first time.

The rest of this article is organized as follows. In Sect. 2, we introduce the structure of the T2FNC and the related learning algorithms IMSA and SARLSA. In Sect. 3, we explain the simulation experimental results of navigation control and compare them with the results of other methods. Finally, the results are summarized in Sect. 5 and suggestions for future research are given.

2. Navigation Control of UGV Based on the T2FNC

In this section, we introduce the navigation control of UGV using the T2FNC, as shown in Fig. 1. The NVIDIA Jetson AGX Orin is used to receive information from LiDAR and to determine the T2FNC-based navigation control of the UGV. First, we will explain the structure of the T2FNC. Next, the IMSA learning algorithm with optimized global search capabilities is introduced. Finally, we will explain the design of the navigation fitness functions and evaluate the navigation performance of unmanned vehicles.

2.1 T2FNC

The architecture of the proposed T2FNC is a five-layer architecture comprising the input layer, fuzzy layer, firing layer, output processing layer, and output layer, as shown in Fig. 2. The *j*th IF~THEN~ fuzzy rule is expressed as follows.

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

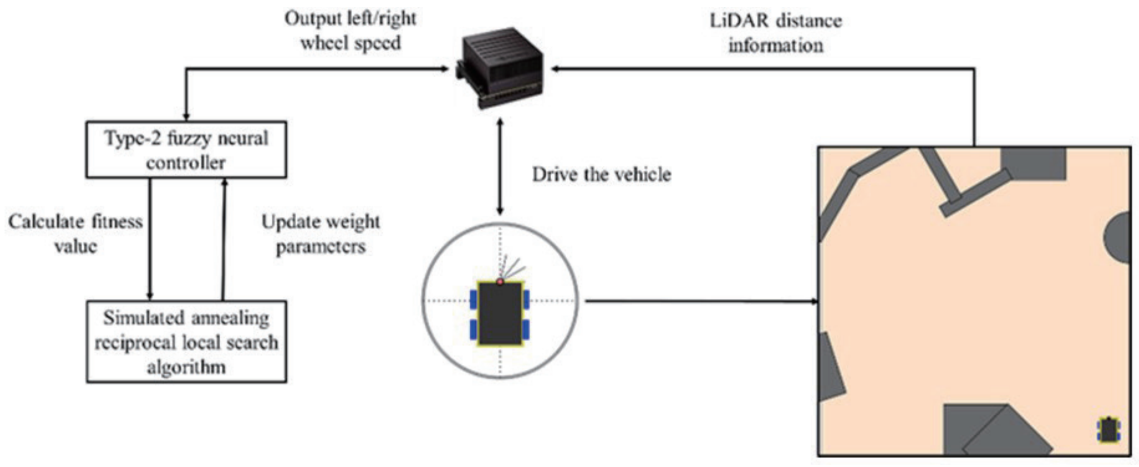

Output left/right wheel speed
LiDAR distance information
Type-2 fuzzy neural controller
Drive the vehicle
Calculate fitness value
Update weight parameters
Simulated annealing reciprocal local search algorithm

Fig. 1. (Color online) T2FNC-based navigation control of UGV.

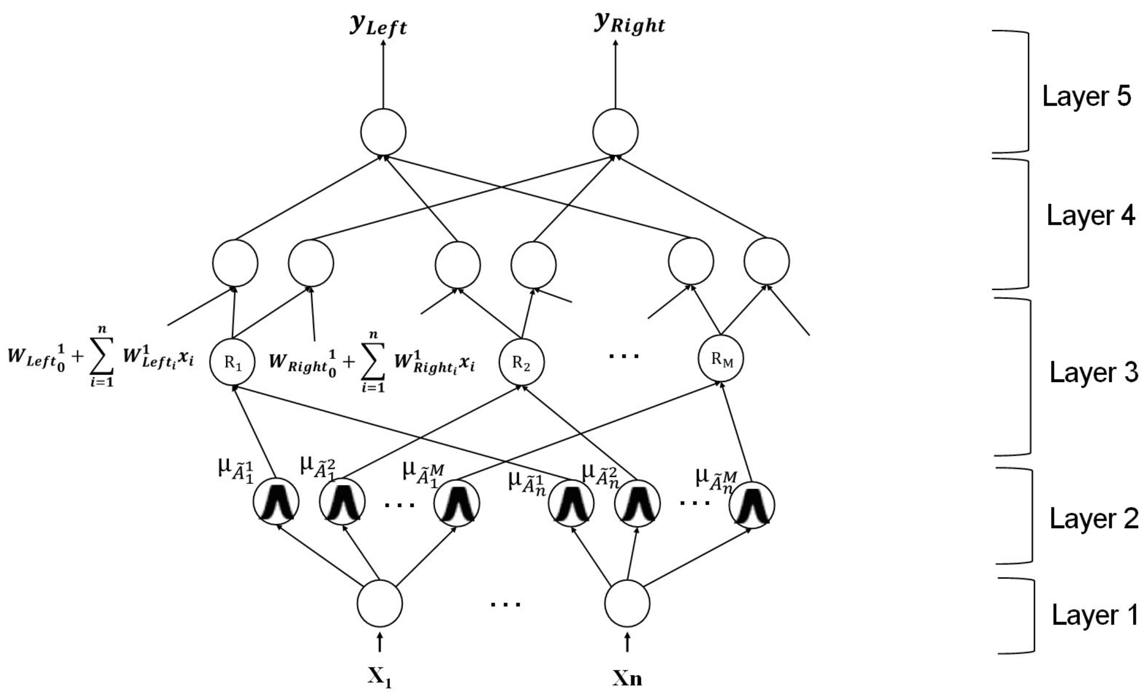


Fig. 2. Architecture of the proposed T2FNC.

Here, $x_1, x_2, \ldots, x_n$ are inputs, $\tilde{A}_1^j, \tilde{A}_2^j, \ldots, \tilde{A}_n^j$ are type-2 fuzzy sets, and $w_0^j + \Sigma_{i=1}^{n} w_i^j x_i$ is the TSK-type consequent part of the fuzzy rule.

Layer 1: This layer receives environmental information from LiDAR and passes this information to the next layer. Therefore, this layer does not need to do any calculations.

Layer 2: This layer is responsible for blurring the output information of the previous layer. Each node is defined as a type-2 fuzzy set. These fuzzy sets handle uncertainty through Gaussian membership functions, which has an uncertain mean value $[m_1, m_2]$ and a fixed standard deviation *σ*. The membership function $\mu_{\tilde{A}}$ of type-2 fuzzy set is

$$\mu_{\tilde{A}_i^j} = \exp\left(-\frac{1}{2}\cdot\left(\frac{x_i-\mu_i^j}{\sigma_i^j}\right)^2\right) \equiv N\left(m_i^j, \sigma_i^j, x_i^j\right), m_i^j \in \left[m_{i1}^j, m_{i2}^j\right]. \tag{2}$$

The degree of membership of the membership function $\mu_{\tilde{A}}$ is called the uncertainty footprint, which is jointly represented by the upper boundary membership function $\overline{\mu}_{\tilde{A}}$ and the lower boundary membership function $\underline{\mu}_{\tilde{A}}$.

$$\overline{\mu}_{\tilde{A}_i^j} = \begin{cases} N\left(m_{i1}^j, \sigma_i^j, x_i^j\right), & x_i < m_{i1}^j, \\ 1\,, & m_{i1}^j \le x_i \le m_{i2}^j, \\ N\left(m_{i2}^j, \sigma_i^j, x_i^j\right), & x_i > m_{i2}^j, \end{cases} \tag{3}$$

$$\underline{\mu}_{\tilde{A}_i^j} = \begin{cases} N\left(m_{i2}^j, \sigma_i^j, x_i^j\right), & x_i < \dfrac{m_{i1}^j + m_{i2}^j}{2}, \\ N\left(m_{i1}^j, \sigma_i^j, x_i^j\right), & x_i > \dfrac{m_{i1}^j + m_{i2}^j}{2}. \end{cases} \tag{4}$$

The output of each node is expressed as an interval, $\left[\overline{\mu}_{\tilde{A}_i^j}\ , \underline{\mu}_{\tilde{A}_i^j}\right]$, which is composed of an upper boundary membership function and a lower boundary membership function, reflecting the range of the uncertainty footprint.

Layer 3: Each node represents a fuzzy rule and uses the algebraic product to implement fuzzy intersection operations. The firing strength F^j of each fuzzy rule is defined as follows.

$$F^j = \left[\overline{f}^j, \underline{f}^j\right] \tag{5}$$

$$\overline{f}^j = \prod_i^n \overline{\mu}_{\tilde{A}_i^j} \text{ and } \underline{f}^j = \prod_i^n \underline{\mu}_{\tilde{A}_i^j} \tag{6}$$

Layer 4: This layer reduces the order of the type-2 fuzzy set and converts it into a type-1 fuzzy set. The center-of-gravity method is used to defuzzify the type-2 fuzzy set to obtain its upper and lower bounds [y_{upper}, y_{lower}]. To reduce the computational complexity of the order-reduction process, the center-of-sets order-reduction method is adopted, and the formula is as follows.

$$y_{upper} = \frac{\sum_{j=1}^{M} \overline{f}^j \left(w_0^j + \sum_{i=1}^{n} w_i^j x_i\right)}{\sum_{j=1}^{M} \overline{f}^j} \tag{7}$$

$$y_{lower} = \frac{\sum_{j=1}^{M} \underline{f}^{j} \left(w_0^j + \sum_{i=1}^{n} w_i^j x_i \right)}{\sum_{j=1}^{M} \underline{f}^{j}} \tag{8}$$

Layer 5: This layer averages the output results of the previous layer to obtain the final output.

$$y = \frac{y_{upper} + y_{lower}}{2} \tag{9}$$

Since the UGV used in this study is a differential wheel, the T2FNC yields two outputs corresponding to the left wheel speed (y_{Left}) and the right wheel speed (y_{Right}).

2.2 Proposed IMSA learning algorithm

In this subsection, we introduce our proposed IMSA learning algorithm. The specific steps of IMSA are as follows.

Step 1: Initialization

In this step, each mantis is randomly positioned in the D-dimensional solution space and initialized, and the N mantises in the mantis swarm are randomly distributed in the solution space. Therefore, a two-dimensional matrix x of size $N \times D$ can be expressed. The position of mantis i at the function evaluation t can be defined by a vector initialized randomly within the lower and upper bounds of the optimization problem[26] as the following formula:

$$\vec{x}_i^t = \vec{x}^l + \vec{r} * \left(\vec{x}^u - \vec{x}^l \right), \tag{10}$$

where $\vec{x}_i^t$ is the current position of the mantis, $\vec{r}$ is a vector with a randomly generated value between –1 and 1, and $\vec{x}^u$ and $\vec{x}^l$ represent the upper and lower bounds, respectively.

Step 2: Coding

The position of the mantis is encoded through parameters. The parameters of each mantis include the mean $\left[m_{i1}^j, m_{i2}^j \right]$, standard deviation σ_i^j, and weights w_0^j and w_i^j in the T2FNC. That is, these parameters are used to describe the state of the mantis in the solution space.

Step 3: Search for prey

In this step, the members of the mantis swarm are divided into two behavioral roles: pursuers and ambush predators. Pursuers actively search for prey and simulate movement behaviors of different steps and directions. The formula is

$$\vec{x}_i^{t+1} = \begin{cases} \vec{x}_i^t + \vec{\tau}_1 * \left(\vec{x}_i^t - \vec{x}_a^t \right) + \left| \vec{\tau}_2 \right| \cdot \vec{U} * \left(\vec{x}_a^t - \vec{x}_b^t \right), \text{ if } r_1 \le r_2, \\ \vec{x}_i^t * \vec{U} + \left(\vec{x}_a^t + \vec{r}_3 * \left(\vec{x}_b^t - \vec{x}_c^t \right) \right) * \left(1 - \vec{U} \right), \text{ otherwise,} \end{cases} \tag{11}$$

where $\vec{\tau}_1$ is a random number generated by the Levy flight strategy, and $\vec{\tau}_2$ is a random number generated in accordance with the normal distribution. To simulate the pursuer's direction of action, the binary vector $\vec{U}$ is used:

$$\vec{U} = f\left(x\right) = \begin{cases} 0, & \text{if } \vec{r}_4 < \vec{r}_5, \\ 1, & \text{otherwise,} \end{cases} \tag{12}$$

where $\vec{r}_4$ and $\vec{r}_5$ are uniformly distributed random vectors between 0 and 1. If $\vec{r}_4$ of the *j*th dimension is less than $\vec{r}_5$, the corresponding $\vec{U}$ will be 0; otherwise, $\vec{U}$ will be 1.

Ambush predators will remain stationary and wait for their prey to enter attack range.

$$\vec{x}_i^{t+1} = \begin{cases} \vec{x}_i^t + \alpha \cdot \left(\vec{x}_{ar}^{'} - \vec{x}_a^t\right), & \text{if } \vec{r}_9 < \vec{r}_{10}, \\ \vec{x}_{ar}^{'} + \left(r_7 \, {*2} - 1\right) * \mu * \left(\vec{x}^l + \vec{r}_8 \, * \left(\vec{x}^u - \vec{x}^l\right)\right), & \text{otherwise.} \end{cases} \tag{13}$$

Here, $\vec{x}_{ar}^{'}$ is the currently existing global optimal solution, and $\vec{x}_a^{\mathrm{t}}$ is a random individual in the mantis group. α is the factor that controls the position of the mantis' head:

$$\alpha = \cos\left(\pi r_6\right) \cdot \mu \, , \tag{14}$$

where μ is the distance factor,

$$\mu = \left(1 - \frac{t}{E}\right), \tag{15}$$

with t being the current number of iterations and E the target number of iterations.

Step 4: Attack prey

In the step of attacking the prey, the position of the mantis is updated in accordance with the dynamic changes occurring during the attack.

$$x_{i,j}^{t+1} = \frac{\left(x_{i,j}^t + x_j^*\right)}{2.0} + v_s \cdot d_{si,j}^t \tag{16}$$

v_s is the attack speed,

$$v_s = \frac{1}{1 + e^{l\varphi}}, \tag{17}$$

where l is a random parameter used to control the attack speed, and φ is the gravity acceleration during attack. $d_{si,j}^t$ is the attack distance.

$$d_{si,j}^{t} = \left(x_j^{*} - x_{i,j}^{t}\right) \tag{18}$$

If the attack fails, the mantis will reorient and launch a new attack:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + r_{13} \cdot \left(x_{a,j}^{t} - x_{b,j}^{t}\right). \tag{19}$$

Equation (20) is used in conjunction with the failure probability P_f to enhance the convergence rate towards the optimal solution in an optimization problem:

$$x_{i,j}^{t+1} = x_{i,j}^{t} + e^{2l} \cdot \cos\left(2l\pi\right) \cdot \left|x_{i,j}^{t} - \vec{x}_{ar,j}^{\,'}\right| + \left(r_{11} \cdot 2 - 1\right) \cdot \left(x_j^{u} - x_j^{l}\right), \tag{20}$$

where P_f is used to determine whether the next operation will be performed.

$$P_f = \alpha \cdot \left(1 - \frac{t}{T}\right) \tag{21}$$

Step 5: Proposed SARLSA local search algorithm

To avoid the mantis algorithm falling into the local optimal solution, a SARLSA local search algorithm is formulated as

$$M_j^{t+1} = M_j^{t} + r_{14} \times \left(M_{Gbest} - Mutual_{vector} \times BF_2\right), \tag{22}$$

$$M_i^{t+1} = M_i^{t} + r_{15} \times \left(M_{Gbest} - Mutual_{vector} \times BF_1\right), \tag{23}$$

$$Mutual\_vector = \frac{M_j^{t+1} + M_i^{t+1}}{2}, \tag{24}$$

where M_j represents the current jth mantis, M_{Gbest} represents the mantis with the current global best solution, BF_1 and BF_2 are interest factors, M_i is the ith mantis randomly selected from the mantis group, and r_{14} and r_{15} are random numbers between −1 and 1.

The mantis will compare the current fitness value $Fit(x_{i,j}^{t})$ with those of the generated new positions $Fit(M_j^{t+1})$ and $Fit(M_i^{t+1})$. If the fitness value of the generated new position is better than the current fitness value, the mantis will choose to jump out of the current position:

$$e^{\left(Fit\left(M_j^{t+1}\right) - Fit\left(x_{i,j}^{t}\right)/T\right)} > r, \tag{25}$$

$$e^{\left(Fit\left(M_i^{t+1}\right)-Fit\left(x_{i,j}^t\right)/T\right)} > r, \tag{26}$$

where T is the temperature coefficient and r is a random number between −1 and 1. If Eq. (25) or Eq. (26) is satisfied, the current solution will generate a new solution by using the proposed SARLSA in accordance with Eqs. (22)–(24); otherwise, the current solution will be randomly regenerated in accordance with Eq. (11).

Step 6: Sexual cannibalism

To ensure that the solutions advantageous for the mantis group will be retained, we simulated the phenomenon of female mantises eating male mantises after mating, which means that solutions with poor performance are eliminated.

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{r}_{16} * \left(\vec{x}_i^t - \vec{x}_a^t\right) \tag{27}$$

$\vec{r}_{16}$ is a randomly generated attraction factor.

Step 7: Termination condition

When the termination condition is met, the optimal solution has been found and the algorithm stops.

Steps 2 to 6 are repeated until the termination condition is met.

2.3 Definition of fitness function

The fitness function is designed to evaluate the performance of the proposed T2FNC. The inputs of the T2FNC are four angles from LiDAR, which are L_0, L_1, L_2, and L_3 with the values of 10°, 40°, 60°, and 90°, respectively. The detection distance range of LiDAR is 100 m. To reduce the amount of computation required by NVIDIA Jetson AGX Orin, the LiDAR detection range is limited to 0.5 to 8 m. The output of the T2FNC determines the left and right wheel speeds of the differential wheel UGV.

The designed fitness function ρ includes four subfitness functions, namely, ρ_1, ρ_2, ρ_3, and ρ_4. The subfitness function ρ_1 is used to evaluate the actual moving distance of the UGV. When the actual moving distance D_{dis} reaches the set end distance (D_{stop}), it means that the UGV has completed one complete training process. In this study, we set D_{stop} to 250 m.

$$\rho_1 = 1 - \frac{D_{dis}}{D_{stop}} \tag{28}$$

The subfitness function ρ_2 is used to estimate the distance between the UGV and the wall and to calculate its average distance during movement.

$$\rho_2 = \frac{\sum_{t=1}^{T_{all}} I_{dis}(t)}{T_{all}} \tag{29}$$

With this subfitness function, we expect the UGV to maintain a stable distance from the wall. In this case, the value of I_{dis} is close to zero.

$$I_{dis}(t) = \left| L_3(t) - d_{wall} \right| \tag{30}$$

d_{wall} represents the fixed distance to the wall and is set to 2 m.

The subfitness function ρ_3 is used to estimate the average angle between the UGV and the wall.

$$\rho_3 = \frac{\sum_{t=1}^{T_{all}} \left| \theta(t) - 90^\circ \right|}{T_{all}} \tag{31}$$

θ is the angle between the UGV and the wall. We expect the UGV to be able to move parallel to the wall as

$$\theta(t) = \cos^{-1}\left(\frac{angle(t)^2 + L_3{}^2 - L_2{}^2}{2 \times L_3 \times angle(t)} \right). \tag{32}$$

θ is the angle between L_2 and L_3 in the LiDAR information and *angle*(*t*) is the angle calculated using the cosine formula.

$$angle(t) = \sqrt{L_3{}^2 + L_2{}^2 - 2 \cdot L_3 L_2 \cos\left(30^\circ\right)} \tag{33}$$

The subfitness function ρ_4 is used as an indicator to estimate the obstacle avoidance distance of the UGV. When the UGV encounters an obstacle, it needs to make turns and maintain stability during linear motion. If the angle between the UGV and the wall is 90^{o}, and the distance returned by the sensing front (L_0) is less than 6 m, the turning angle is calculated.

$$\rho_4 = 1 - \frac{\sum_{t=1}^{T_{all}} L_{md} - L_0}{\sum_{t=1}^{T_{all}} L_{md}} \tag{34}$$

Here, L_{md} is the maximum detection distance of LiDAR sensing, which is set to 8 m in this study. L_{md} is compared with the data returned by L_0 to obtain the indicator of whether the UGV should turn or keep moving in a straight line.

Finally, the fitness function can be expressed by combining the four subfitness functions.

$$\rho = \frac{1}{1+\left(\rho_1 + \rho_2 + \rho_3 + \rho_4\right)} \tag{35}$$

3. Experimental Results

To verify the performance of the proposed IMSA-based T2FNC, we designed a 50×50 m training environment, as shown in Fig. 3. The fitness values and movement distances of various controllers for obstacle avoidance control and wall-following behavior are analyzed in this environment.

3.1 Parameter settings

In MSA, six parameters need to be preset: the buffer length of optimal solutions (A), the probability of attack failure (p_a), the probability of exchange between the exploration and development stages (p_e), the restoration coefficient of the relationship between pursuer and ambusher (R_c), the gravitational acceleration of the mantis attack (G), and the probability of cannibalism (P_c). The values of these parameters are set to A = 1.0, p_a = 0.5, p_e = 0.5, R_c = 2.0, G = 6.0, and P_c = 0.2. To ensure the fairness of comparisons between algorithms, each algorithm was trained for 1000 iterations. The initialization parameters of all algorithms are default values.

3.2 Experimental results in the training and testing environments

To clearly display the output results of the T2FNC, we use different colors to represent the angle changes. For example, red means the UGV moves in a straight line, blue means the UGV turns left, and green means the UGV turns right. The color definitions are as follows.

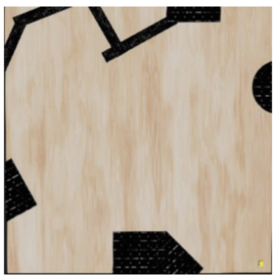


Fig. 3. (Color online) Schematic diagram of training environment.

$$\begin{cases} red, & \text{if } \left| y_{Left} - y_{Right} \right| \le 0.2 \\ green, & \text{if } y_{Left} - y_{Right} > 0.2 \\ blue, & \text{if } y_{Right} - y_{Left} > 0.2 \end{cases} \tag{36}$$

If the speed difference between the two wheels of the UGV is less than 0.2, the UGV is moving forward and is indicated in red. That is, we choose a threshold to filter out some small deviations that are not turns. If the left wheel speed is greater than the right wheel speed by 0.2, the UGV turns right and is indicated in green; conversely, if the right wheel speed is greater than the left wheel speed by 0.2, the UGV turns left and is indicated in blue.

The evaluation indicators of the controller include the fitness function (F), the movement distance of the UGV (D), and the time it takes for the UGV to circle the environment (T). Table 1 shows the experimental results of obstacle avoidance control using various algorithms in the training environment. In this table, the fitness function, movement distance, and movement time of the proposed T2FNC with IMSA are 0.983452, 200 m, and 47.1 s, respectively. The fitness function and movement time of the proposed IMSA method are better than those of the other methods.(21,22,26,30,31) Although the IMSA method's driving distance of 200 m is longer than the GA(22) method's 190 m, the fitness function and moving time of the IMSA method are better than those of the GA method.

Figure 4 shows the movement trajectories of the T2FNC based on various algorithms for wall-following control in the training environment. In the figure, the movement trajectory of IMSA moves smoothly and is clearly better than those of the other methods. In particular, when the UGV passes through turns and square obstacle areas, the IMSA can maintain stable speed and direction and reduce unnecessary pauses and deviations.

In addition, we also designed a 50×50 m^2 testing environment to verify the performance of the proposed IMSA-based T2FNC. Table 2 shows the experimental results of obstacle avoidance control using various algorithms in the testing environment. In this table, the fitness function, movement distance, and movement time of the proposed T2FNC with IMSA are 0.979115, 222 m, and 53.1 s, respectively. The fitness function, moving distance, and moving time of the proposed T2FNC with IMSA are better than those of the other methods,(21,26,30,31) except that the moving distance of the GA method (210 m)(21) is better than that of our method.

Table 1
Experimental results of obstacle avoidance control in the training environment.

Methods	Evaluation indicators		
	Training environment		
	F	D (m)	T (s)
GA(21)	0.927236	190	48.6
PSO(23)	0.966063	205	51.6
MSA(26)	0.982070	205	49.2
ABC(30)	0.954546	213.5	72.4
DE(31)	0.958116	200	51.3
Proposed IMSA	0.983452	200	47.1

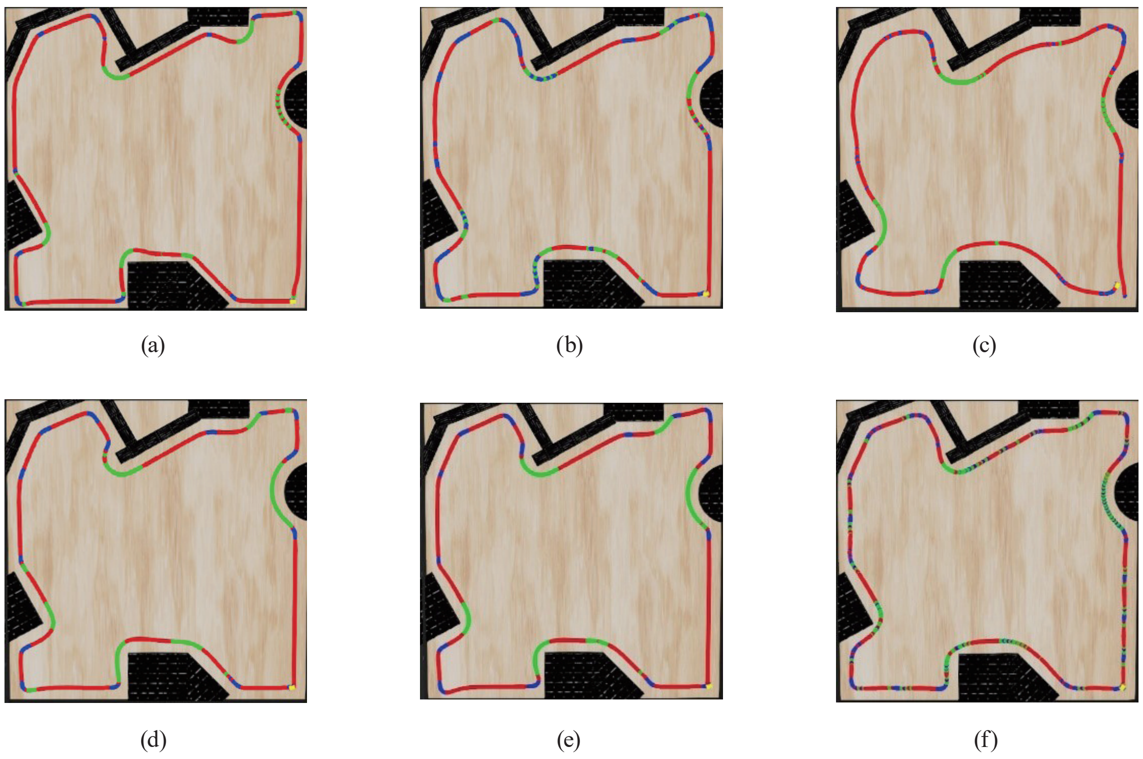
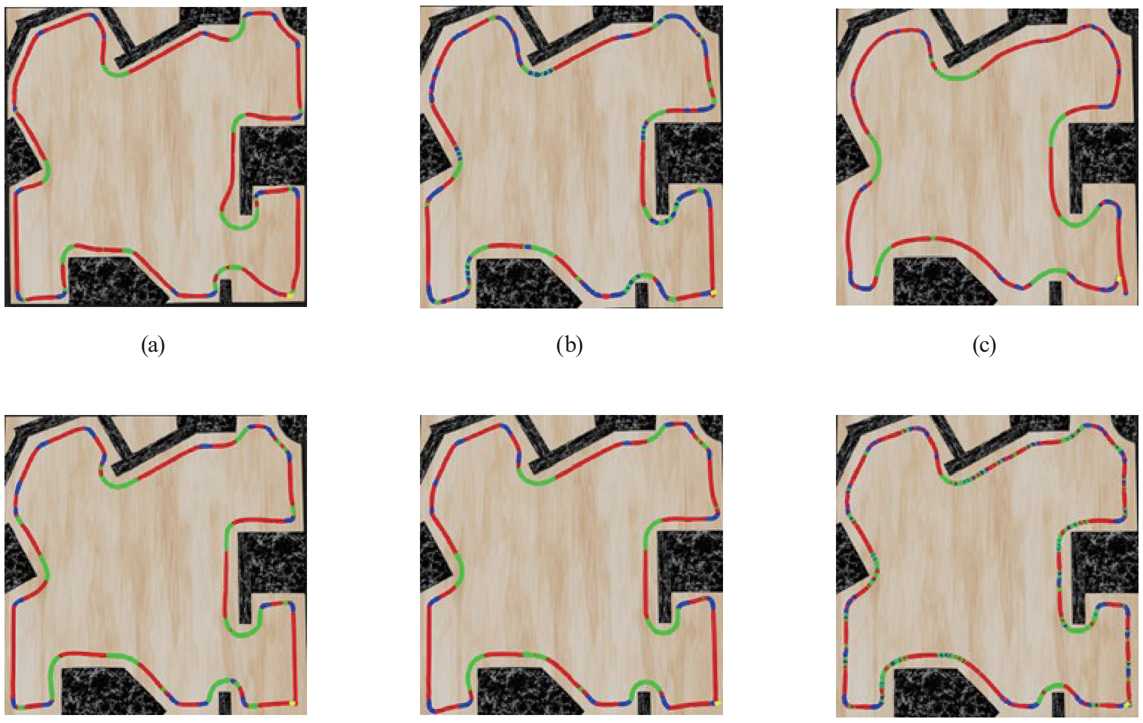


Fig. 4. (Color online) UGV movement trajectories using T2FNC based on (a) ABC, (b) DE, (c) GA, (d) PSO, (e) MSA, and (f) IMSA in the training environment.

Table 2
Experimental results of obstacle avoidance control in the testing environment.

Methods	Evaluation indicators		
	Testing environment		
	F	*D* (m)	*T* (s)
GA[21]	0.923144	210	57.5
PSO[23]	0.966212	232	55.4
MSA[26]	0.977064	232	55.3
ABC[30]	0.923896	246	104.9
DE[31]	0.958383	225	57.4
IMSA	0.979115	222	53.1

Figure 5 shows the movement trajectories of the T2FNC based on various algorithms for wall-following control in the testing environment. In this figure, the moving distance of the proposed IMSA [Fig. 5(f)] is 10 m shorter than those of PSO [Fig. 5(d)] and MSA [Fig. 5(e)]. In the figure, the movement trajectory of IMSA is smooth and clearly better than those of the other methods.



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

Fig. 5. (Color online) UGV movement trajectories using T2FNC based on (a) ABC, (b) DE, (c) GA, (d) PSO, (e) MSA, and (f) IMSA in the testing environment.

3.3 Experimental results of navigation control

The navigation control system of this study is divided into two parts: wall-following control (i.e., obstacle avoidance control) and movement control toward the target. First, as shown in Fig. 6, we divide the LiDAR detection range of the vehicle into four areas: P_1 from +15° to −15°, P_2 from −15° to −90°, P_3 from +15° to +90°, and P_4 180° behind the vehicle. When the vehicle moves forward, if the obstacle is located in any area from P_1 to P_3, the navigation control will switch to wall-following control to allow the vehicle to bypass an obstacle. When the obstacle no longer falls within the detection area, navigation control switches to movement control toward the target.

Table 3 shows the experimental results of obstacle avoidance control using various algorithms. In this table, the movement time of the proposed IMSA is only 29.4 s, which is significantly better than those of the other methods.(21,23,26,30,31) Furthermore, the moving distance of the proposed IMSA is close to that of the GA method (23) and shorter than those of the other methods.(21,26,30,31)

UGV movement trajectories of navigation control using various methods are shown in Fig. 7. In the figure, the motion trajectory of the proposed IMSA algorithm is smoother and more stable than those of the other algorithms, especially when the UGV encounters a sharp turn. Compared

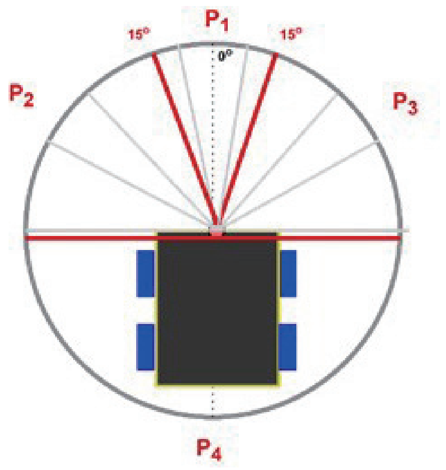
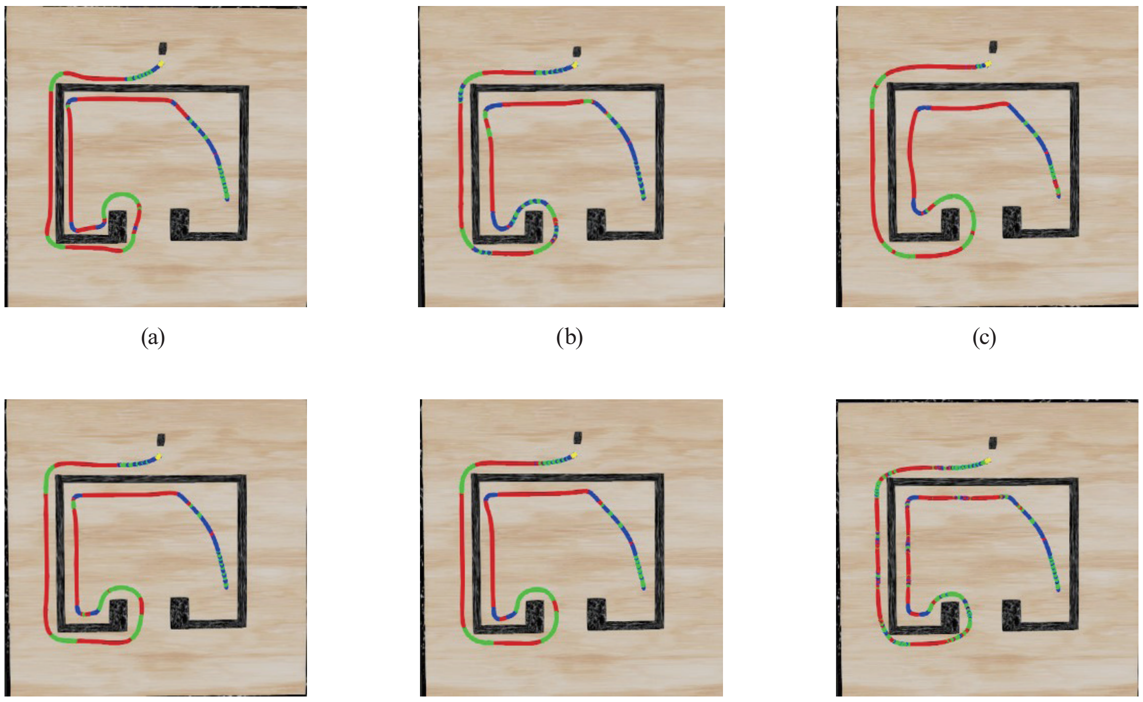


Fig. 6. (Color online) Four zones in the vehicle environment.

Table 3
Experimental results of obstacle avoidance control.

Method	Evaluation	
	D (m)	T (s)
GA(21)	128.3	33.5
PSO(23)	135.5	32.4
MSA(26)	134.3	30.9
ABC(30)	138.6	52.6
DE(31)	132.8	35.7
IMSA	129.4	29.4



(a) (b) (c)

(d) (e) (f)

Fig. 7. (Color online) UGV movement trajectories of navigation control using T2FNC based on (a) ABC, (b) DE, (c) GA, (d) PSO, (e) MSA, and (f) IMSA.

with other algorithms, IMSA maintains vehicle stability more effectively and reduces the number of unnecessary deflections. In addition, IMSA successfully avoids endless loops during navigation.

4. Conclusions

In this study, a T2FNC based on the IMSA was proposed for navigation and obstacle avoidance applications of UGVs in unknown environments. A LiDAR sensor was used to capture distance information between UGVs and the surrounding environment. The IMSA learning algorithm was adopted to adjust the parameters in the T2FNC. In addition, the SARLSA was employed to avoid the traditional MSA from falling into the local optimal solution. In wall-following control, the fitness function, movement distance, and movement time of the proposed T2FNC with IMSA were 0.983452, 200 m, and 47.1 s, in the training environment and 0.979115, 222 m, and 53.1 s, in the test environment, respectively. In navigation control, the movement distance and movement time of the proposed T2FNC with IMSA were 129.4 m and 29.4 s, respectively, which are better than those of the other methods. Compared with the traditional MSA algorithm, the movement distance and movement time of the proposed T2FNC with IMSA were reduced by 4.5% and 4.14%, respectively.

The current T2FNC control system uses one-dimensional LiDAR data as input to detect obstacles and navigate vehicles, and has also obtained good experimental results. However, in practical applications, we often face problems of high/low and thick/thin obstacles, which will affect the navigation stability of the system. To overcome this problem, we will use three-dimensional LiDAR data in future research to further enhance the system's sensing capabilities. This will provide the system with a more detailed interpretation of its surroundings, including the height and depth of obstacles. Moreover, in future research, we will also consider the energy efficiency of the proposed wall-following control system when evaluating its performance.

References

1 X. Wang, M. Wang, and Y. Zhang: Proc. 33rd Chinese Control Conf. (IEEE, 2014) 647–652. https://doi.org/10.1109/ChiCC.2014.6896701

2 M. Mammarella, L. Comba, A. Biglia, F. Dabbene, and P. Gay: Proc. IEEE Int. Workshop on Metrology for Agriculture and Forestry (IEEE, 2020) 224–229. https://doi.org/10.1109/MetroAgriFor50201.2020.9277573

3 P. Appelqvist, J. Knuuttila, and J. Ahtiaine: Mechatronic Systems Applications, A. Milella and G. Cicirelli, Eds. (InTech, London, UK, 2010) p. 15. https://doi.org/10.5772/8919

4 L. Wang, D. Cheng, F. Gao, F. Cai, J. Guo, M. Lin, and S. Shen: Proc. 2018 Int. Symp. Experimental Robotics (2020) 59–71. https://doi.org/10.48550/arXiv.1806.02487

5 M. Rodríguez, A. Al-Kaff, Á. Madridano, D. Martín, and A. de la Escalera: Proc. 2020 Int. Conf. Unmanned Aircraft Systems (IEEE, 2020) 110–116. https://doi.org/10.1109/ICUAS48674.2020.9213974

6 Q. Liu, Z. Li, S. Yuan, Y. Zhu, and X. Li: Sensors **21** (2021) 1354. https://doi.org/10.3390/s21041354

7 S. M. Patole, M. Torlak, D. Wang, and M. Ali: IEEE Signal Process. Mag. **34** (2017) 22. https://doi.org/10.1109/MSP.2016.2628914

8 A. H. Pech, P. M. Nauth, and R. Michalik: Proc. 18th Int. Conf. Smart Technologies (IEEE, 2019) 1–5. https://doi.org/10.1109/EUROCON.2019.8861933

9 B. Han, Y. Wang, Z. Yang, and X. Gao: IEEE Trans. Intell. Transp. Syst. **21** (2020) 3046. https://doi.org/10.1109/TITS.2019.2923752

10 K. Zhu, J. Li, and H. Zhang: Proc. 2nd Int. Conf. Information Technology and Electronic Commerce (IEEE, 2014) 152–156. https://doi.org/10.1109/ICITEC.2014.7105591

11 S. Kraemer, M. E. Bouzouraa, and C. Stiller: Proc. IEEE Intelligent Vehicles Symp. (IEEE, 2019) 1543–1548. https://doi.org/10.1109/IVS.2019.8814079

12 A. Haytham, Y. Z. Elhalwagy, A. Wassal, and N. M. Darwish: Proc. 2014 Int. Conf. Engineering and Technology (IEEE, 2014) 1–8. https://doi.org/10.1109/ICEngTechnol.2014.7016751

13 Y. Shtessel, C. Edwards, L. Fridman, and A. Levant: Sliding Mode Control and Observation, Y. Shtessel, C. Edwards, L. Fridman, and A. Levant, Eds. (Birkhäuser, Switzerland, 2014) p. 43. https://doi.org/10.1007/978-0-8176-4893-0

14 P. Wang, S. Gao, L. Li, S. Cheng, and L. Zhao: IEEE Access **7** (2019) 64984. https://doi.org/10.1109/ACCESS.2019.2917507

15 F. N. Martins, W. C. Celeste, R. Carelli, M. Sarcinelli-Filho, and T. F. Bastos-Filho: Control Eng. Pract. **16** (2008) 1354. https://doi.org/10.1016/j.conengprac.2008.03.004

16 S. Haykin: Adaptive Control Design and Analysis, G. Tao, Ed. (Wiley, New York, 2003) p. 90. https://doi.org/10.1002/0471459100

17 K. Zhang, Q. Sun, and Y. Shi: IEEE Trans. Neural Networks and Learning Systems **32** (2021) 5554. https://doi.org/10.1109/TNNLS.2020.3048305

18 S. K. Oh, H. J. Jang, and W. Pedrycz: Expert Sys. Appl. **38** (2011) 11217. https://doi.org/10.1016/j.eswa.2011.02.169

19 O. Castillo, L. Amador-Angulo, J. R. Castro, and M. Garcia-Valdez: Inf. Sci. **354** (2016) 257. https://doi.org/10.1016/j.ins.2016.03.026

20 S. I. Horikawa, T. Furuhashi, and Y. Uchikawa: IEEE Trans. Neural Networks and Learning Systems **3** (1992) 801. https://doi.org/10.1109/72.159069

21 J. H. Holland: Sci. Am. **267** (1992) 66. https://doi.org/10.1038/scientificamerican0792-66

22 S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi: Science **220** (1983) 671. https://doi.org/10.1126/science.220.4598.671

23 J. Kennedy and R. Eberhart: Proc. Int. Conf. Neural Networks (IEEE, 1995) 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

24 C. J. Lin, B. H. Chen, and J. Y. Jhang: IEEE Access **11** (2023) 107917. https://doi.org/10.1109/ACCESS.2023.3315741

25 M. Dorigo, M. Birattari, and T. Stutzle: IEEE Comput. Intell. Mag. **1** (2006) 28. https://doi.org/10.1109/MCI.2006.329691

26 M. Abdel-Basset, R. Mohamed, M. Zidan, M. Jameel, and M. Abouhawwash: Comput. Methods Appl. Mech. Eng. **415** (2023) 116200. https://doi.org/10.1016/j.cma.2023.116200

27 G. Moustafa, H. Alnami, S. H. Hakmi, A. M. Shaheen, A. R. Ginidi, M. A. Elshahed, and H. S. E. Mansour: IEEE Access **12** (2024) 2674. https://doi.org/10.1109/ACCESS.2023.3344679

28 G. Moustafa, H. Alnami, S. H. Hakmi, A. Ginidi, A. M. Shaheen, and F. A. Al-Mufadi: Biomimetics **8** (2023) 490. https://doi.org/10.3390/biomimetics8060490

29 Z. Fei, Y. Li, and S. Yang: Sensors **24** (2024) 3174. https://doi.org/10.3390/s24103174

30 M. S. Kiran and A. Babalik: J. Comput. Commun. **2** (2014) 108. https://doi.org/10.4236/jcc.2014.24015

31 R. Storn and K. Price: Proc. IEEE Int. Conf. Evolutionary Computation (IEEE, 1996) 842–844. https://doi.org/10.1109/ICEC.1996.542711