Interpretable Fuzzy Logic Control for Multi-Robot Coordination in a Cluttered Environment

Yu-Cheng Chang, Ye Shi, Anna Dostovalova, Zehong Cao, Jijoong Kim, Daniel Gibbons, and Chin-Teng Lin, *Fellow, IEEE*,

***Abstract*—Mobile robot navigation is an essential problem in robotics. We propose a method for constructing and training fuzzy logic controllers (FLCs) to coordinate a robotic team performing collision-free navigation and arriving simultaneously at a target location in an unknown environment. Our FLCs are organised in a multi-layered architecture to reduce the number of tunable parameters and improve the scalability of the solution. In addition, in contrast to simple traditional switching mechanisms between target seeking and obstacle avoidance, we develop a novel rule-embedded FLC to improve the navigation performance. Moreover, we design a grouping and merging mechanism to obtain transparent fuzzy sets and integrate this mechanism into the training process for all FLCs, thus increasing the interpretability of the fuzzy models. We train the proposed FLCs using a novel multi-objective hybrid approach combining a genetic algorithm and particle swarm optimisation. Simulation results demonstrate the effectiveness of our algorithms in reliably solving the proposed navigation problem.**

***Index Terms*—Fuzzy logic controller, multi-objective hybrid GA-PSO, Multi-robot navigation, simultaneous arrival.**

I. Introduction

Multi-robot navigation control has attracted interest in recent years because it can be used in various applications, suchas autonomous vehicles in surveillance or territory exploration mission and self-driving cars in the smart transport system [1]–[4]. Missions of a robotic team often require coordinating the order of individual robots to complete assigned tasks. In particular, this study investigates a method to coordinate the arrival time on target for a multiagent robotic team while ensuring no collisions occur. The simultaneous arrival problem has become an emerging application of multiagent systems, especially in transport of goods and services [5], [6]. Such applications usually aim to minimize the overall transport time and the delay of arrival time of a given mission. For example, automatic delivery systems might consider unmanned vehicle routing with simultaneous pickup and delivery to provide the customers with a satisfactory service [7], [8]. There is a significant body of research dedicated to this problem.

In [1], Yao *et al*. developed a dynamical system to adjust both the path length and the speed for each robot to achieve simultaneous arrival. A cooperative localisation technique with a proportional navigation guidance law was used in [2] to ensure that multiple unmanned vehicles could simultaneously reach a moving target in a GPS-denied environment. In [3], Babel *et al*. proposed an optimal path assignment method to coordinate the simultaneous arrival of multiple air vehicles. However, these methods assume that the map of the environment is static and known a priori. Thus, they may not be suitable for more practical scenarios, i.e., multi-robot coordination in a dynamic and unknown environment. Also, neither of these methods is scalable to high-dimensional input spaces.

Fuzzy system is a feasible approach to provide robust control and scalability for robot navigation and has been widely used in such problems. In [9], Pothal *et al*. proposed an adaptive neuro-fuzzy controller for robot navigation in a highly cluttered environment considering robot behaviours such as obstacle avoidance, target seeking and speed control. The controller was trained in a supervised manner. This method relies on the availability of precise input-output training data. Juang *et al*. in [10] used a particle swarm optimisation (PSO)-based algorithm to tune a fuzzy controller applied for robot navigation in unknown environments. This method does not require training data and can be trained in a real-time manner. Chen *et al*. in [11] propose a knowledge-based neural fuzzy controller which adopted a knowledge-based evolutionary strategy to train a fuzzy neural network for mobile robot navigation control. Those studies has shown that fuzzy systems can alleviate the negative impacts of sensor noise and provide robust control for robot navigation.

Another advantage of fuzzy systems is that they possess interpretability because they learn knowledge from data and then represent it in the form of fuzzy rules. Such a mechanism can enable a human to understand the inherent knowledge captured by a fuzzy model. Ishibuchi *et al*. [12] and Cococcioni *et al*. [13] designed interpretable fuzzy systems by minimising the number of fuzzy rules and fixing the partitioning granularity of the input space during the optimisation process. Furthermore, to keep balance between the accuracy and interpretability of fuzzy systems, several approaches using multi-objective optimization [14]–[19] have been developed. The authors of

This work was supported in part by Australia Defence Science & Technology (DST) Group under Agreement RA9124. Research was also sponsored in part by the Australian Research Council (ARC) under discovery grant DP210101093, and US Office of Naval Research Global under Cooperative Agreement Number ONRG - NICOP - N62909-19-1-2058.

Yu-Cheng Chang, Ye Shi and Chin-Teng Lin are with CIBCI lab, Australian Artificial Intelligence Institute, the School of Computer Science, University of Technology, Sydney, NSW 2007, Australia. (Email: Yu-Cheng.Chang@uts.edu.au, Ye.Shi-1@uts.edu.au and Chin-Teng.Lin@uts.edu.au)

Anna Dostovalova, Jijoong Kim and Daniel Gibbons are with Australia Defence Science & Technology (DST) Group, Australia. (E-mail: Anna.Dostovalova@dst.defence.gov.au, Jijoong.Kim@dst.defence.gov.au and Daniel.Gibbons@dst.defence.gov.au)

Zehong Cao is with the School of Information and Communication Technology, University of Tasmania. (E-mail: zehong.cao@utas.edu.au)

[14]–[17] imposed a constraint condition to assign a proper distribution of fuzzy sets in each input variable to preserve the transparency of the fuzzy sets after tuning of their free parameters. Furthermore, a means of measuring the relevance of fuzzy rules has been proposed to identify fuzzy-based redundancy in the feature space, which helps to reduce the complexity of the fuzzy rules and achieve a suitable trade-off in performance between accuracy and interpretability for multi-objective optimisation algorithms [18], [19].

In this work, we consider scenarios in which agents cannot rely on a map because it may be outdated or change mid-mission. To coordinate a team of robots in such an environment, we construct a multi-layered system of FLCs to achieve multi-robot navigation and simultaneous arrival-time control. The control system consists of two levels of controllers: a high-level controller to ensure the same time of arrival for all robots at a destination point and a low-level controller to enable each robot to perform collision-free navigation. Moreover, the interpretability of the FLCs is improved by using transparent fuzzy sets, which are highly distinguishable and can properly represent the universe of discourse of the data space. Specifically, we develop a grouping and merging mechanism to obtain transparent fuzzy sets and integrate this mechanism into the training process for the FLCs. This mechanism is based on a distance metric to measure the similarity between fuzzy sets in terms of the centre and variance of each. Any two fuzzy sets with a similarity above a given threshold will be merged into a single set. The main contributions of this paper are four-fold:

- This paper proposes a scalable FLC-based method to enable multi-robot navigation in an unknown and complex environment. This method is based on a feature-splitting strategy, which can significantly reduce the number of parameters to be learned in the fuzzy rules without compromising navigation performance.
- In contrast to traditional switching mechanisms [10], [20], [21] between target seeking and obstacle avoidance, this paper proposes a novel and robust rule-embedded FLC to improve navigation performance.
- This paper develops a grouping and merging mechanism to obtain more transparent fuzzy sets and integrates this mechanism into the training process for all developed FLCs, thus increasing the interpretability of the FLCs.
- A new multi-objective hybrid method combining a genetic algorithm (GA) and PSO (M-GA-PSO) is presented to train all of the proposed FLCs. The M-GA-PSO method outperforms its single-objective counterpart in terms of both coordination accuracy and navigation performance. Specifically, the proposed method is based on performance and similarity grouping such that each swarm has its own adaptive neighbourhood.

The rest of this paper is structured as follows. In Section II, we design the proposed controllers, identify the tunable parameters that define their fuzzy rules and discuss their input/output structures. In Section III, we describe the M-GA-PSO method of training the FLCs and propose the grouping and merging mechanism to further enhance the interpretability of the FLCs. Section IV gives the details of the training process. Simulations are presented in Section V, and Section VI concludes the paper.

II. Fuzzy Controllers for Multi-Robot Navigation and Coordination

We consider the following scenario. A team of agents is to navigate through a cluttered (possibly dynamic) area to arrive at a designated destination point, which might also change during the mission. Each agent is equipped with a 2D lidar sensor that has an 80 m range and a 180° scanning angle. The agents cannot stop while in transit, and there is a limit on their angular velocity. Such a scenario requires a set of reactive controllers that take readings from the on-board sensors of the agents and are stable with respect to input noise. Fuzzy inference system (FIS) control is a natural choice in these circumstances. To alleviate the dimensional issues of the problem, we create a two-layered system (Fig. 1) for multi-robot navigation and coordination. The lower-level controller (FLC1) is responsible for the safe navigation of an individual robot, and the higher-level controller (FLC2) provides coordination for simultaneous arrival by adjusting the speeds and headings of all robots. As shown in Fig. 2, FLC1 consists of an obstacle avoidance controller and a behaviour selector.

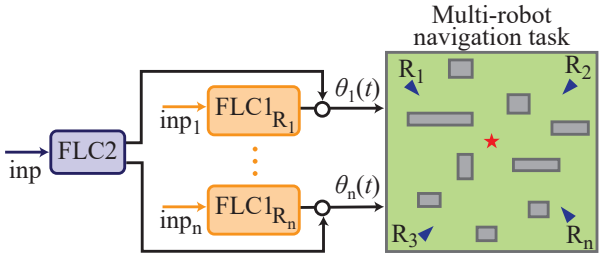


Fig. 1. Block diagram of the control configuration for multi-robot navigation and arrival-time control.

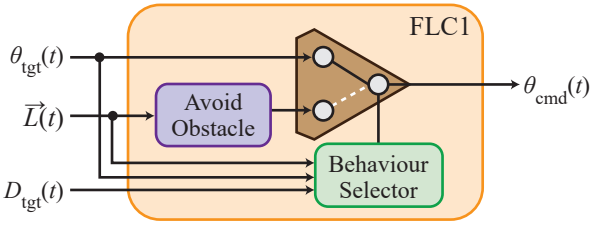


Fig. 2. The low-level individual navigation controller.

A. Scalable FLC for Obstacle Avoidance

The obstacle avoidance controller is constructed as a first-order Takagi-Sugeno-Kang (TSK)-type FIS with 15 fuzzy rules of the following form:

$$R^j: \text{IF } x_1^j \text{ is } A_1^j \text{ AND } x_2^j \text{ is } A_2^j \text{ AND } x_3^j \text{ is } A_3^j,$$
$$\text{THEN } \theta^j = b_0^j + \textstyle\sum_{i=1}^{3} b_i^j x_i^j,$$

where A_1^j, A_2^j, and A_3^j are fuzzy sets defined by Gaussian membership functions, $j = 1, \ldots, 15$, and the AND operation

is implemented as the algebraic product. The inputs are provided by the lidar sensor of the controlled agent, as shown in Fig. 3. For each sector, the shortest distance to a detected obstacle is returned, $L_1, \ldots, L_7$. To improve the scalability

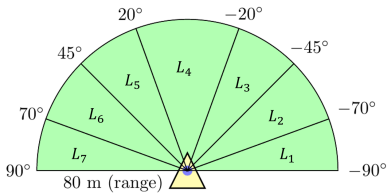


Fig. 3. Scanning area of a 2D lidar system in the simulation setting.

of the controller, we arrange the scanner readings into three groups, $\vec{x}_{right} = (L_1, L_2, L_3)$, $\vec{x}_{front} = (L_3, L_4, L_5)$ and $\vec{x}_{left} = (L_7, L_6, L_5)$, to represent the features on the right side, on the left side and in front of the sensor, respectively. Then, we map them to the inputs of the fuzzy rules, x_1^j, x_2^j, and x_3^j, in the following way:

input	$j = 1, \ldots, 5$	$j = 6, \ldots, 10$	$j = 11, \ldots, 15$
x_1^j	L_1	L_3	L_7
x_2^j	L_2	L_4	L_6
x_3^j	L_3	L_5	L_5

For the last 5 rules, we make use of the left-right symmetry of the lidar sensor. The inputs from the sectors located on the sensor's left-hand side are taken in reverse order to match the corresponding sectors on the right-hand side. This allows us to re-use the membership functions of the first 5 fuzzy rules for their antecedent parts and flip the signs of the parameters defining their consequent parts, thus reducing the number of tunable parameters for this controller from 150 to 100.

B. FLC for Robot Behaviour Selection

In this section, we develop a robust rule-embedded FLC to achieve behaviour switching between the default mode of moving towards the target, i.e., seek behaviour, and avoiding obstacles, i.e., avoid behaviour. The switching rules in [10], [20], [21] are simply based on certain pre-defined conditions to control behaviour switching. It is clear that such simple rules cannot guarantee safe navigation, especially in a cluttered environment. The robot behaviour selector proposed in this paper consists of 6 fuzzy rules with the following structure:

$$
\begin{aligned}
&R^1: \text{IF } \theta_{\text{tgt}} \text{ is } \Theta^F \text{ AND } C^1 \text{ is True, THEN Avoid;}\\
&R^2: \text{IF } \theta_{\text{tgt}} \text{ is } \Theta^R \text{ AND } C^2 \text{ is True, THEN Avoid;}\\
&R^3: \text{IF } \theta_{\text{tgt}} \text{ is } \Theta^L \text{ AND } C^3 \text{ is True, THEN Avoid;}\\
&R^4: \text{IF } \theta_{\text{tgt}} \text{ is } \Theta^F \text{ AND } C^4 \text{ is False, THEN Seek;}\\
&R^5: \text{IF } \theta_{\text{tgt}} \text{ is } \Theta^R \text{ AND } C^5 \text{ is False, THEN Seek;}\\
&R^6: \text{IF } \theta_{\text{tgt}} \text{ is } \Theta^L \text{ AND } C^6 \text{ is False, THEN Seek.}
\end{aligned}
$$

Here, θ_{tgt} is the scaled angle to the target, and Θ^F, Θ^L and Θ^R are fuzzy sets of directions corresponding to the scenarios in which the target is in front or on the left or right side of the robot, respectively.

$C^{[1,6]}$ are defined as follows:

$$
\begin{aligned}
C^1 :=& R^{front,1} \text{ OR } R^{right,1} \text{ OR } R^{left,1},\\
C^2 :=& R^{front,2} \text{ OR } R^{right,2} \text{ OR } R^{left,2},\\
C^3 :=& R^{front,3} \text{ OR } R^{right,3} \text{ OR } R^{left,3},\\
C^4 :=& \text{Inverse of } C^1,\\
C^5 :=& \text{Inverse of } C^2,\\
C^6 :=& \text{Inverse of } C^3,
\end{aligned}
\tag{1}
$$

where

$$
\begin{aligned}
R^{front,1} =& (L_3 \text{ is } M_{\text{F}} \text{ AND } d_{L_3} \text{ is } Z_d),\\
R^{front,2} =& (L_4 \text{ is } M_{\text{F}} \text{ AND } d_{L_4} \text{ is } Z_d),\\
R^{front,3} =& (L_5 \text{ is } M_{\text{F}} \text{ AND } d_{L_5} \text{ is } Z_d),\\
R^{right,1} =& (L_1 \text{ is } M_{\text{R}} \text{ AND } d_{L_1} \text{ is } Z_d),\\
R^{right,2} =& (L_2 \text{ is } M_{\text{R}} \text{ AND } d_{L_2} \text{ is } Z_d),\\
R^{right,3} =& (L_3 \text{ is } M_{\text{R}} \text{ AND } d_{L_3} \text{ is } Z_d),\\
R^{left,1} =& (L_5 \text{ is } M_{\text{L}} \text{ AND } d_{L_5} \text{ is } Z_d),\\
R^{left,2} =& (L_6 \text{ is } M_{\text{L}} \text{ AND } d_{L_6} \text{ is } Z_d),\\
R^{left,3} =& (L_7 \text{ is } M_{\text{L}} \text{ AND } d_{L_7} \text{ is } Z_d).
\end{aligned}
\tag{2}
$$

Here, M_{F}, M_{L} and M_{R} are fuzzy sets of distances corresponding to the scenarios in which the target is in front or on the left or right side of the robot, respectively; Z_d is predefined by the user and determines the status of reaching the target, i.e., True or False; and $d_{L_i} = L_i - d_{\text{tgt}}$, where d_{tgt} is the distance to the target. The above fuzzy rules can be interpreted as follows: If there is an obstacle between the target and the robot, then one of the rules R_1, R_2 and R_3 is triggered, and the robot changes its behaviour to Avoid. If the robot has a direct line to the target, then one of the rules R_4, R_5 and R_6 is triggered, and the robot changes its behaviour to Seek.

C. FLC for Arrival-Time Coordination

FLC2 is a centralised controller that synchronises the robots' arrival times at the target. During each control loop, the agents are ranked in ascending order based on their (estimated) distances to the target. Let X_{rank_1} denote the robot that is the closest to the destination point, and let X_{rank_n} denote the robot that is the farthest away. In this paper, we do not take all of the robots' speeds as input information, as this is not a scalable approach. Instead, we directly control only the speeds of the closest robot and the farthest one, and the speeds of the others can then be obtained through interpolation. FLC2 takes five inputs: the movement speeds of robots X_{rank_1} and X_{rank_n}, v_1 and v_n); their distances to the target, D_1 and D_n; and $\Delta D = D_1 - D_n$. The controller is constructed as a zero-order TSK fuzzy system with 5 inputs, 2 outputs and 10 fuzzy rules of the following form:

$$
R^j: \text{IF } D_1 \text{ is } B_1^j \text{ AND } \ldots \text{ AND } \Delta D \text{ is } B_5^j, \text{ THEN } \vec{\alpha} \text{ is } \vec{a}^j.
$$

Here, $\vec{a}^j \in \mathbb{R}^2$ is the consequent component of rule j , and $B_1^j, \ldots, B_5^j$ are Gaussian membership functions. There are 120

tunable parameters in FLC2. We used the weighted average defuzzification method [22] to calculate the output values for all FLCs in this study. The outputs of the FLC2 are the speed scaling factors scaling factors $\vec{\alpha} = [\alpha_1, \alpha_n] \in [0.5, 1.5]$ for the first- and last-ranked robots, respectively. The scaling factors for the rest of the robots are calculated as follows:

$$\alpha_i = \frac{i-1}{n-1}(\alpha_n - \alpha_1) + \alpha_1, \ i = 2, \ldots, n-1. \tag{3}$$

The updated velocities are calculated as $\hat{v}_i = \alpha_i v_i$. The velocities are allowed to vary within $\pm 50\%$ of the user-defined value. To make the lengths of the robots' paths roughly equal, FLC2 also adjusts each robot's heading angle. The heading adjustment for robot i in time step t is calculated as follows:

$$\theta_{\text{adj},i}(t) = \theta_{\text{tgt},i}(t) + \beta_i \theta_{\text{max\_adj}}, \tag{4}$$

where $\theta_{\text{tgt},i}(t)$ is the target bearing (in the body coordinates), $\theta_{\text{max\_adj}}$ is the maximum allowed angular adjustment, and β_i is a scaling factor that determines the strength of heading angle adjustment and is calculated from D_n, D_i (the estimated distances to the target for robots n and i calculated in time step $(t-1)$) and α_i as follows:

$$\beta_i = \sqrt{(1 - D_i/D_n)/\alpha_i}. \tag{5}$$

III. Multi-Objective Hybrid GA-PSO Algorithm

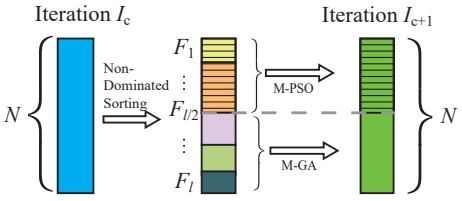


Fig. 4. Learning configuration in the M-GA-PSO algorithm.

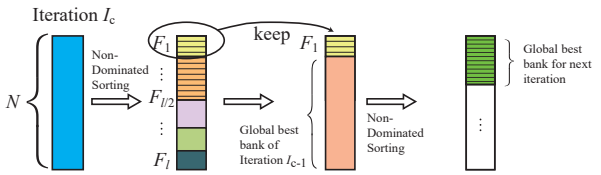


Fig. 5. The process of updating the local-best bank.

In M-GA-PSO, each particle or individual represents a whole solution to a specific problem. A block diagram of the procedure for the creation of new individuals is shown in Fig. 4. In each iteration, non-dominated sorting is performed to categorise N new solutions into l fronts based on their performances. A front is formed of individuals that cannot be dominated by each other because their fitness values are not directly comparable. Then, the top half of the ranked individuals are updated by a multi-objective PSO algorithm (M-PSO), and the rest of the solutions are updated by a multi-objective GA (M-GA). In every iteration, after new solutions are generated, M-GA-PSO updates the local-best bank (Fig. 5). Suppose that there are N new solutions categorised into l fronts. The first-ranked solutions are kept and mixed with those in the old local-best bank. Non-dominated sorting is applied to the mixed set of local-best solutions, and only the first-ranked individuals form the new local-best bank for the next iteration.

A. M-GA

The crossover operation of M-GA is performed when the local-best bank has more than two entries. A new particle vector $\vec{s}_{\text{new}} = (s^1_{\text{new}}, \ldots, s^M_{\text{new}})$ is created as follows:

$$s^j_{\text{new}} = \begin{cases} r_c P^j_1 + (1 - r_c) P^j_2, & j = 1, \ldots, \tau, \\ (1 - r_c) P^j_1 + r_c P^j_2, & j = \tau + 1, \ldots, M, \end{cases} \tag{6}$$

where r_c is a random number from $[0, 1]$, $\vec{P}_1$ and $\vec{P}_2$ are two parents randomly selected from the local-best bank, τ denotes a randomly selected crossover site, and M is the length of the solution vector. After the crossover operation, mutations occur with a pre-defined probability p_m. The mutation operation is implemented as follows:

$$s^j_{\text{new}} \leftarrow s^j_{\text{new}} + c_m r^j_m (s^j_{\max} - s^j_{\min}), \quad j = 1, \ldots, M, \tag{7}$$

where r^j_m is a random number from $[-1, 1]$, c_m is a coefficient that restricts the strength of mutation, and $\vec{s}_{\max}$ and $\vec{s}_{\min}$ are the upper and lower bounds of the solution space, respectively. The initial velocity vector of each new solution is assigned by means of the following formula:

$$\vec{v}_{\text{new}} = c_m(\vec{r}_1(\vec{P}_1 - \vec{s}_{\text{new}}) + \vec{r}_2(\vec{P}_2 - \vec{s}_{\text{new}})), \tag{8}$$

where $\vec{r}_1$ and $\vec{r}_2$ are vectors of random numbers on $[0, 1]$.

If there is only one global solution in the bank, then M-GA performs an asexual reproduction operation to generate new solutions. This operation is implemented via Gaussian sampling. Suppose that there is only one local-best solution P^g. Each element of P^g is set as the centre of a Gaussian density function, and the standard deviation is set to 0.1. The asexual reproduction operation is followed by a mutation operation to prevent the new solutions from being too similar to each other.

B. M-PSO

In addition to M-GA, M-GA-PSO incorporates the velocity and position update mechanism of M-PSO to ensure that all solutions explore a relatively optimal area during the optimisation process. The swarm is divided into N_g groups in the same manner as in [10]. Neighbours are particles within the same group. The neighbourhood best position, $\vec{P}^g_z$, is the leading particle in group z, selected as follows:

$$\vec{P}^g_z = \underset{1 \le k \le n}{\arg\max}\, \text{SPD}(\vec{s}_i, \vec{P}^g_k). \tag{9}$$

Here, n is the size of the global-best bank, and SPD is a metric based on the similarity and fitness of the particles, defined as

$$\text{SPD}(\vec{s}_i, \vec{P}^g_k) = \frac{\|\vec{s}_i - \vec{P}^g_k\| D_p(\vec{P}^g_k)}{\|\vec{f}(\vec{s}_i)\| - \|\vec{f}(\vec{P}^g_k)\|}, \tag{10}$$

where $\vec{s}_i$ is the i-th particle in the group, $\vec{P}_k^g$ is the k-th local-best solution, and $\vec{f} = [f_i^1, \ldots, f_i^{N_{\text{obj}}}]$ is an N_{obj}-dimensional objective vector. To discriminate among the solutions in a front, we introduce a crowding distance $D_p(\cdot)$. The solutions are sorted by their crowding distance values in ascending order. The crowding distance of the i-th solution $D_p(\vec{s}_i)$ is obtained as follows:

$$D_p(\vec{s}_i) = \frac{1}{N_{\text{obj}}} \sum_{j=1}^{N_{\text{obj}}} u_i^j, \tag{11}$$

with

$$u_i^j = \frac{|f_{i+1}^j - f_{i-1}^j|}{f_{\max}^j - f_{\min}^j}, \quad j = 1, \ldots, N_{\text{obj}}, \tag{12}$$

where $f_{\max}^j$ and $f_{\min}^j$ are the maximum and minimum values, respectively, of the j-th objective and i is the running index for the solutions in a group. The crowding distances of the two boundary solutions are both set to 1.

The velocity $\vec{v}_i$ of each particle i in group z is updated using its individual best position, $\vec{P}_i$, and the neighbourhood best position, $\vec{P}_z^g$:

$$\begin{aligned} &\vec{v}_i(t+1) = \\ &\quad \chi((\vec{v}_i(t) + c_1\vec{r}_1(\vec{P}_i - \vec{s}_i(t)) + c_2\vec{r}_2(\vec{P}_z^g - \vec{s}_i(t))), \end{aligned} \tag{13}$$

where χ is a constriction factor that controls the magnitude of $\vec{v}_i$ and c_1 and c_2 are positive acceleration coefficients. This method forces different groups to locate different optima and thus reduces the chance of all solutions becoming trapped in a local minimum. Each particle changes its position in accordance with the following formula:

$$\vec{s}_i(t+1) = \vec{s}_i(t) + \vec{v}_i(t+1). \tag{14}$$

C. Interpretability Improvement

This study considers the transparency of fuzzy sets to improve the interpretability of an FLC. Specifically, transparent fuzzy sets should 1) be sufficiently distinguishable and 2) represent the universe of discourse of the input variables. Additionally, the antecedent of the fuzzy model should be as simple as possible. Generally, a smaller number of fuzzy sets is more desirable. In this paper, we propose a grouping and merging mechanism to obtain transparent fuzzy sets and integrate the proposed method into the FLC training process.

The grouping mechanism finds similar fuzzy sets in a partition of an input variable by using a distance measure and a pre-defined distance threshold δ_g. Given input x_i and under the assumption that B_i^j is a fuzzy set of rule j, the Gaussian membership function can be described in the following form:

$$\mu_i^j(x_i) = \exp\left\{ -\frac{\left(x_i - m_i^j\right)^2}{{\sigma_i^j}^2} \right\}, \tag{15}$$

where μ_i^j is the membership value of B_i^j with input x_i. Then, the distance between two fuzzy sets in input variable i, B_i^j and B_i^l, is defined as follows:

$$\rho_m\left(B_i^p, B_i^q\right) = \sqrt{\left(m_i^p - m_i^q\right)^2 + \left(\sigma_i^p - \sigma_i^q\right)^2}. \tag{16}$$

A cut-off distance ρ_{cutoff} is set for grouping fuzzy sets in an input variable. The fuzzy set grouping algorithm is shown in Algorithm 1. To group fuzzy sets in input variable i, the fuzzy sets are sorted in ascending order of their centres as follows: $\widetilde{B}_i^1, \ldots, \widetilde{B}_i^r$. The grouping algorithm sequentially evaluates the distance ρ_m starting from $\widetilde{B}_i^1$, and those fuzzy sets that are located within ρ_{cutoff} will be grouped in the first group, G_1. If a fuzzy set B_i^j satisfies $\rho_m\left(\widetilde{B}_i^1, \widetilde{B}_i^j\right) > \rho_{\text{cutoff}}$, then B_i^k will be assigned to a new group, G_2. The algorithm continuously evaluates ρ_m between each pair of fuzzy sets B_i^j until every fuzzy set is assigned to a group. After the fuzzy sets have been grouped, a reference centre is generated for each group. Every reference centre is obtained by calculating the arithmetic mean of the centres of the fuzzy sets assigned to the corresponding group. The reference centres are then used to guide the fuzzy sets belonging to the same group; the fuzzy sets of input variable i are updated as follows:

$$\hat{m}_i^p = m_i^p + r_3(m_i^{G_j} - m_i^p), \ \widetilde{B}_i^p \in G_j, \tag{17}$$

where m_i^p is the centre of fuzzy set $\widetilde{B}_i^p$, $\hat{m}_i^p$ is the updated centre and $r_3 \in [0, 1]$ is a uniform random number. The update of the centres occurs after the position update of M-PSO; fuzzy sets in the same group will move closer to each other during the learning process. Significant overlap between neighbouring fuzzy sets may occur due to the changes in their centres and widths. Let the degree of overlap be denoted by δ; given sorted fuzzy sets $\left\{\widetilde{B}_i^p, \ \widetilde{B}_i^q, \ldots\right\} \in G_j$, the degree of overlap between $\widetilde{B}_i^p$ and $\widetilde{B}_i^q$ can be evaluated as follows:

$$\delta_{pq} = \max\left(\mu_i^p(m_i^q), \ \mu_i^q(m_i^p)\right), \tag{18}$$

where $\mu_i^p(m_i^q)$ is the membership value of $\widetilde{B}_i^p$ fed with m_i^q, the centre of $\widetilde{B}_i^q$, and $\mu_i^q(m_i^p)$ is the membership value of $\widetilde{B}_i^q$ fed with m_i^p. A higher δ_{pq} indicates greater overlap between $\widetilde{B}_i^p$ and $\widetilde{B}_i^q$. If δ_{pq} is larger than the pre-set threshold value δ_{th}, then these two neighbouring fuzzy sets will be merged together. The centre of the new fuzzy set is set equal to the average of the centres of the two merged fuzzy sets, and similarly, the width is set to the average width. The parameters of the two merged fuzzy sets in the solution vector will be replaced by the parameters of the new fuzzy set. Fig. 6 shows the pipeline of the interpretable fuzzy controller training process based on M-GA-PSO.

IV. Controller Training

All three controllers introduced in Section II are trained separately in a cascaded manner. We firstly train the robot to move along the boundary of an obstacle, which can be used in the second and third phases of training as the obstacle avoidance behaviour. After the first phase of training is complete, the behaviour selector part of FLC1 is trained to navigate a robot in a cluttered environment towards its destination point. The parameters of the obstacle avoidance controller are fixed during this phase of training. Finally, FLC2 is trained to coordinate a group of robots (each carrying an already trained FLC1) to achieve simultaneous arrival. The maps used for training are shown in Fig. 7. The separated training strategy

Algorithm 1 Fuzzy Set Grouping Algorithm

Require: $\widetilde{B}_i^1, \ldots, \widetilde{B}_i^r$, fuzzy sets sorted in ascending order
  Initialisation: Set empty groups $G_j \leftarrow \emptyset (1 \leq j \leq r)$
  $j \leftarrow 1$, set the index of group j to 1
  $G_j \leftarrow \widetilde{B}_i^1$, assign $\widetilde{B}_i^1$ to the first group
  for $p = 1, \ldots, r$ **do**
    for $q = p+1, \ldots, r$ **do**
      if $\rho_m\left(\widetilde{B}_i^p, \widetilde{B}_i^q\right) < \rho_{\text{cutoff}}$ **then**
        $G_j \leftarrow G_j \cup \widetilde{B}_i^q$, assign B_i^q to G_k
      else
        $p \leftarrow q$
        $j \leftarrow j+1$
      end if
    end for
  end for

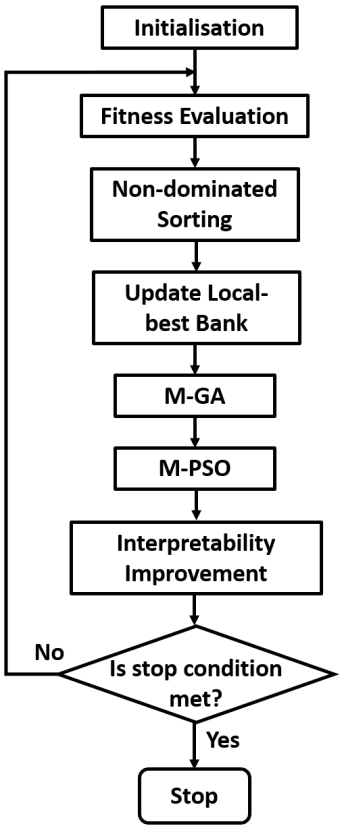
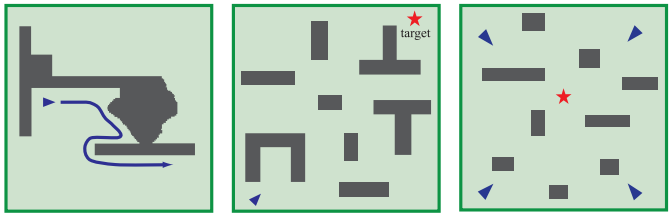


Fig. 6. Flowchart of the M-GA-PSO algorithm.

enables a complex task to be decomposed into several simple sub-tasks to reduce the complicity of the training. Furthermore, each controller can learn a particular function and ensure proper performance for a sub-task. We, therefore, can design a fuzzy controller performing boundary following movement, which can be used in different environments with various shapes of obstacles.

A. Phase 1—Training for Obstacle Avoidance

During phase 1 of training, the robot moves around the obstacle shown in Fig. 7(a) and stops when the following



(a) (b) (c)

Fig. 7. Maps for the 3 phases of controller training.

constraint is violated:

$$\min(L_1, L_2, L_3, L_4, L_5, L_6, L_7) > D_{\min}. \tag{19}$$

Here, $D_{\min}$ is a user-specified value which determines the collision condition of the robot, i.e. $D_{\min} = 0.1$ m. When the robot stops, the number of time steps taken is recorded as T_{total}. The distance from the obstacle that the robot maintains in every time step is used as the other criterion for performance evaluation. Thus, we have two objectives, f_1 and f_2:

$$f_1 = \frac{\sum_{t=1}^{T_{\text{total}}} |L_7(t) - D_{\text{BF}}|}{T_{\text{total}}}, \quad f_2 = 1/T_{\text{total}}. \tag{20}$$

D_{BF} is a user-specified value.

B. Phase 2—Collision-Free Navigation

A robot equipped with boundary-following knowledge is now placed in a more complex environment (Fig. 7(b)). The goal is to safely navigate from the starting point to the target location relying only on sensor data. The boundary-following behaviour is fixed. The second part of FLC1 acts as a switch between two modes: moving towards the target and avoiding an obstacle. The fitness value of a solution is determined by running a simulation. The simulation stops either when a pre-set number of steps is reached or when the distance to the target location becomes less than a user-defined value D_{miss}. The robot's trajectory is given a rating based on its final distance from the target (D_{final}), the number of collisions with obstacles (N_{c}), and the total length of the trajectory (L). This gives us two objective functions:

$$f_3 = N_{\text{c}} + P, \quad f_4 = L + P, \tag{21}$$

where P is a penalty function defined as

$$P = \begin{cases} 0, & \text{if } D_{\text{final}} \leq D_{\text{miss}}, \\ 1000 D_{\text{final}}, & \text{if } D_{\text{final}} > D_{\text{miss}}. \end{cases} \tag{22}$$

The penalty value is added to both objective functions in (21) to gradually eliminate particles that fail to reach the target.

C. Phase 3—Training for Arrival-Time Coordination

For arrival-time coordination, a team of 4 agents is placed in a single scenario, as shown in Fig. 7(c). Each robot is equipped with a trained FLC1 and a 2D lidar sensor. A single destination point is selected. The scenario set-up is fixed during the training process. The evaluation of controller performance involves a complete simulation run from start to

end until either all robots arrive at their target or the simulation time limit is reached. At the end of the run, the robots are ranked by their arrival times, and the first (T_1) and last (T_n) of these times are used to generate two objective values as follows:

$$f_5 = |T_n - T_1|, \quad f_6 = (T_1 + T_n)/2. \tag{23}$$

When f_5 reaches its minimum, the robots arrive at their destination point simultaneously; f_6 keeps all robots moving at their maximum speeds by minimising the average arrival time of the two robots positioned at opposite ends of the ranking.

M-GA-PSO vs. PSO

To compare the optimisation performance of M-GA-PSO with that of single-objective PSO, we conducted 10 runs for each training phase using each algorithm for statistical evaluation. When using PSO, we combined the objective functions defined in (20)–(23) using cascading weights:

$$\begin{aligned} f_{\text{PSO}_1} &= 0.1f_1 + 10f_2, \\ f_{\text{PSO}_2} &= 10f_3 + 0.1f_4, \\ f_{\text{PSO}_3} &= 10f_5 + 0.1f_6. \end{aligned} \tag{24}$$

From each run of M-GA-PSO, we selected the fitness pair with the smallest combined value for comparison. The results are summarised in Table I. In Table I, M-GA-PSO considers the transparency of the fuzzy sets during the optimisation process, while M-GA-PSO No Interpret. excludes the interpretability improvement and does not consider the transparency of the fuzzy sets. In our opinion, M-GA-PSO succeeds in forcing the particles to explore a wider area and thus finds a better solution, and the proposed interpretability improvement mechanism has no impact on the control performance of M-GA-PSO.

V. Results and Discussion

A. FLC1 Behaviour Selector vs. Simple Switching Mechanism

To compare the performance of the FLC1 behaviour selector with that of hand-coded switching logic, we ran 20 robots in a test environment. The results are shown in Fig. 8. Panel (a) shows the trajectories produced by robots equipped with both parts of FLC1: FLC1-AC (the obstacle avoidance controller) and FLC1-BS (the embedded behaviour selector). Panel (b) shows the trajectories produced by robots equipped with FLC1-AC and a hand-coded switching mechanism [10]. The hand-code switching mechanism includes a fuzzy controller for obstacle avoidance behaviour. This fuzzy controller had left and right rule in a symmetric manner to deal with obstacles locating on the left and right side, respectively. The hand-code switching mechanism determines the robot's left or right avoidance behaviours by comparing the distance between the left-side and the right-side obstacle. For example, if the sensor reading of the left-side is greater than the right side, then the robot performs the right-side avoidance behaviour. This simple control logic might cause rapid switches between the left or right avoidance behaviours when the robot is straightforwardly approaching a convex corner of an obstacle, as shown in Fig. 8 (b), because there was not much difference between the sensor readings of the left-side and right-side. In this situation, the robot cannot determine its behaviour and switches between the left and right-side avoidance behaviour, leading the robot to collide with the obstacle.

TABLE I
Comparison of M-GA-PSO with and without the interpretability improvement and PSO for controller training.

		f_1	f_2	f_3	f_4	f_5	f_6
M-GA-PSO	AVG	4.2051	0.0033	0.0	186.83	0.41	432.9
	STD	0.9760	0.0	0.0	3.14	0.053	35.3
M-GA-PSO No Interpret.	AVG	4.0651	0.0033	0.0	185.24	0.48	430.62
	STD	0.7247	0.0	0.0	2.58	0.045	32.6
PSO	AVG	5.4971	0.0036	0.0	187.57	0.63	458.6
	STD	0.7080	0.0005	0.0	1.56	0.097	30.8

We observe that FLC1-BS (the embedded behaviour selector) achieves collision-free navigation more reliably than its hand-coded counterpart, as shown in Fig. 8 (a). The proposed embedded behaviour selector includes fuzzy rules that received the sensor readings from the right side, on the left side and the front to determine movements of the avoidance behaviour. The designed fuzzy rules successfully improve the flaw of hand-coded switching mechanism; prevent the rapid switches between the left or right avoidance behaviours. As a result, the proposed embedded behaviour selector can deal with more complicated shape obstacles and environments than the hand-coded switching mechanism.

B. Arrival-Time Coordination

To give FLC2 the ability to handle teams of various sizes without retraining, we constructed it such that it directly controls only two agents (the farthest from and the nearest to the destination point) by producing two speed adjustment factors for those agents. For the rest of the team, the speed adjustment factors are calculated using (3). FLC2 also controls the heading angles of the agents via (4) and (5).

Figs. 9 and 10 show the results for controller-managed teams of 10, 20 and 30 agents in two different test environments. In Fig. 9, the target point is set to $(x, y) = (100, 350)$. The controller achieves a 1-time-step time difference between the fastest robot and the slowest robot and takes 788 time steps to complete the navigation task when controlling 30 robots. When the number of robots is reduced to 10, the controller achieves zero time difference with a 786 time-step completion time. In Fig. 10, the target point is set to $(x, y) = (400, 100)$. FLC2 achieves perfect time control for all three teams, but the completion time increases with the size of the team. It takes a team of 10 agents 795 time steps to complete the task, whereas a team of 30 needs 924 time steps. These simulation results demonstrate that the proposed controller has good scalability and robustness; without any retraining, it is able to coordinate varying numbers of robots with acceptable performance.

C. Interpretable Fuzzy Controller for Arrival-Time Coordination

This section presents the knowledge learned by the interpretable fuzzy controller for arrival-time coordination. Table II

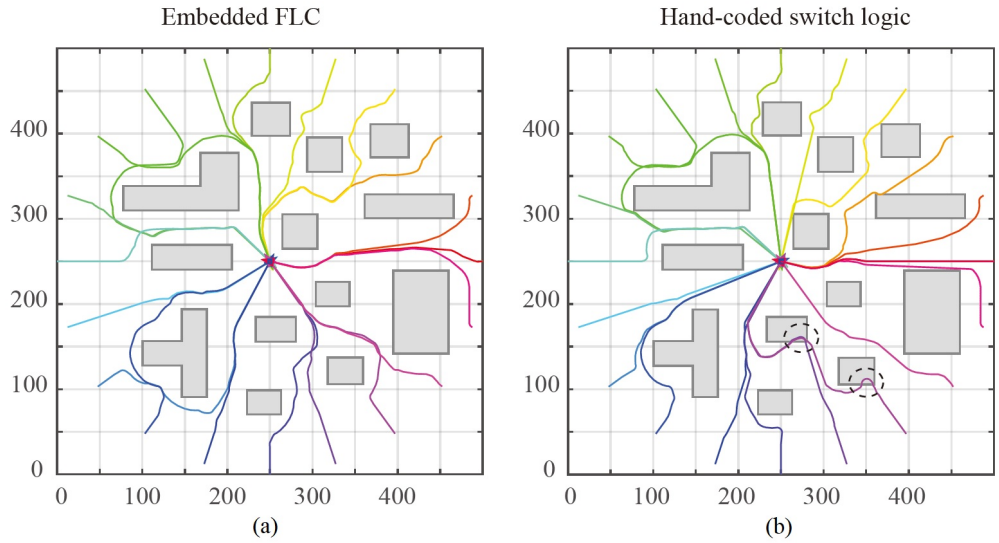


Fig. 8. Trajectories produced by (a) the trained behaviour selector and (b) a hand-coded switching mechanism as in [10], [23]. Each line represents a robot's trajectory.

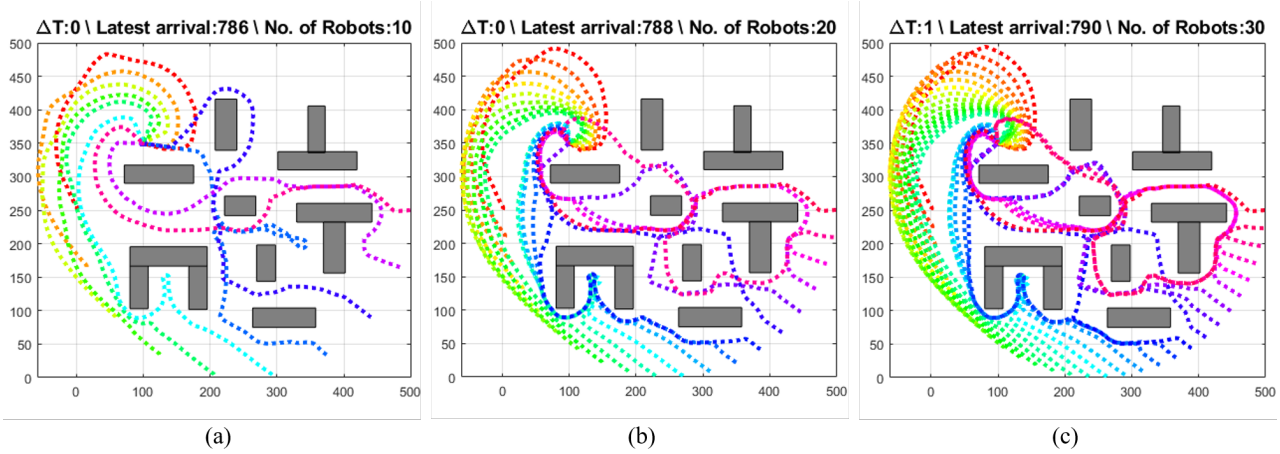


Fig. 9. Arrival-time coordination of robotic teams of various sizes. The numbers of robots in (a), (b) and (c) are 10, 20 and 30, respectively.

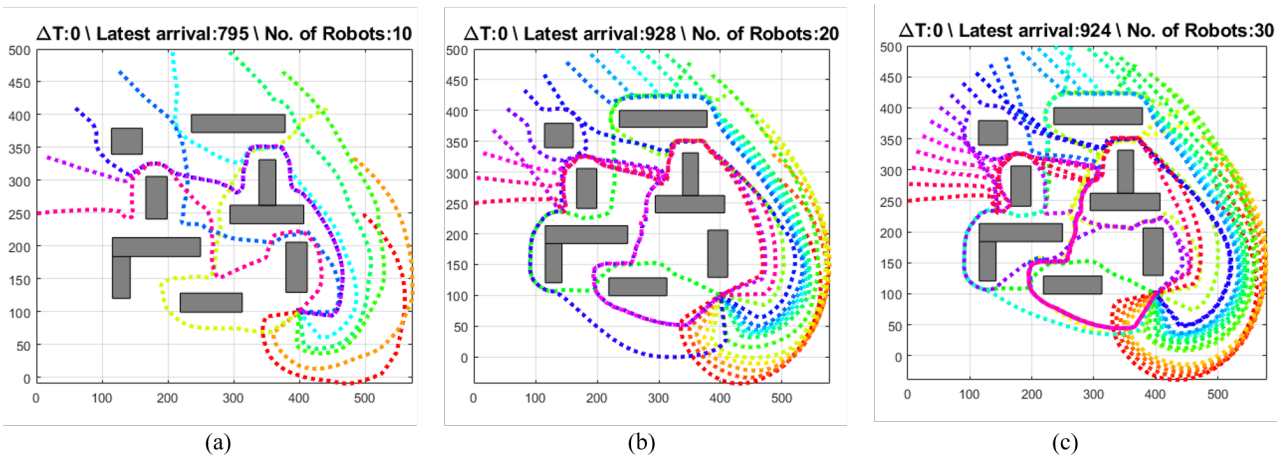


Fig. 10. Arrival-time coordination of robotic teams of various sizes. The numbers of robots in (a), (b) and (c) are 10, 20 and 30, respectively.

TABLE II
FUZZY RULES LEARNED BY M-GA-PSO WITH THE INTERPRETABILITY IMPROVEMENT

Rules	**Input 1**	**Input 2**	**Input 3**	**Input 4**	**Input 5**	α_1	α_n
Rule 1	B_1^2 (Middle Near)	B_2^4 (Very Far)	B_3^4 (Very Fast)	B_4^1 (Slow)	B_5^3 (Distant)	0.8635	1.9476
Rule 2	B_1^1 (Reaching)	B_2^1 (Very Near)	B_3^1 (Very Slow)	B_4^1 (Slow)	B_5^1 (Closing)	1.5919	1.6165
Rule 3	B_1^5 (Very Far)	B_2^4 (Very Far)	B_3^3 (Fast)	B_4^1 (Slow)	B_5^3 (Distant)	0.8785	1.8372
Rule 4	B_1^1 (Reaching)	B_2^2 (Near)	B_3^4 (Very Fast)	B_4^1 (Slow)	B_5^2 (Not Distant)	0.3147	1.7453
Rule 5	B_1^1 (Reaching)	B_2^1 (Very Near)	B_3^2 (Slow)	B_4^1 (Slow)	B_5^1 (Closing)	1.3919	1.6165
Rule 6	B_1^3 (Middle)	B_2^3 (Middle)	B_3^4 (Very Fast)	B_4^2 (Fast)	B_5^1 (Closing)	0.8568	0.7833
Rule 7	B_1^5 (Very Far)	B_2^4 (Very Far)	B_3^3 (Fast)	B_4^3 (Very Fast)	B_5^2 (Not Distant)	1.5635	1.9476
Rule 8	B_1^3 (Middle)	B_2^4 (Very Far)	B_3^1 (Very Slow)	B_4^1 (Slow)	B_5^3 (Distant)	0.5705	1.9722
Rule 9	B_1^2 (Middle Near)	B_2^3 (Middle)	B_3^4 (Very Fast)	B_4^1 (Slow)	B_5^4 (Very Distant)	0.9477	0.8103
Rule 10	B_1^4 (Far)	B_2^4 (Very Far)	B_3^2 (Slow)	B_4^2 (Fast)	B_5^1 (Closing)	1.1705	0.7254

reveals the M-GA-PSO-designed fuzzy rules. We set the number of fuzzy rules to 10, and there are five input variables and two consequent components in each rule. The fuzzy controller here is fed D_1, D_n, v_1, v_n, and ΔD. D_1 and D_n are the distances between the target and robots X_{rank_1} and X_{rank_n}, respectively. v_1 and v_n are the movement speeds of robots X_{rank_1} and X_{rank_n}, respectively. ΔD is the difference in the remaining distances of robots X_{rank_1} and X_{rank_n} to the target, that is, $\Delta D = D_1 - D_n$. The outputs of the fuzzy controller are α_1 and α_n for the speed regulation of robots X_{rank_1} and X_{rank_n}, respectively. Fig. 11 shows the corresponding fuzzy sets in the five input variables. Compared to the fuzzy sets learned without the interpretability improvement, as shown in Fig. 12, the transparency of the fuzzy sets in Fig. 11 is improved, allowing humans to understand the learned knowledge captured in the M-GA-PSO-designed fuzzy rules. For example, $Rule\ 1$ represents the situation in which X_{rank_1} is moving "Very Fast" and at a "Middle Near" distance to the target and X_{rank_n} is moving at a "Slow" speed and a "Very Far" distance. Consequently, $Rule\ 1$ slows down X_{rank_1} with $\alpha_1 = 0.8635$ and speeds up X_{rank_n} with $\alpha_1 = 1.9476$. If both X_{rank_1} and X_{rank_n} are located far from the target and are moving very fast, then the firing strength of $Rule\ 7$ will become large; the controller will speed up both X_{rank_1} and X_{rank_n}. On the other hand, if X_{rank_1} is approaching the target and moving fast while X_{rank_n} is still at a middle distance and moving slowly, then the firing strength of $Rule\ 4$ will become large; the controller will reduce the movement speed of X_{rank_1} and accelerate X_{rank_n} to catch up with X_{rank_1}.

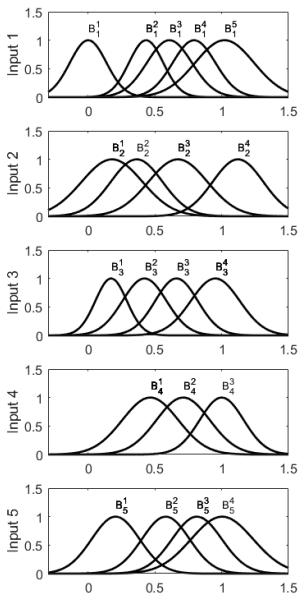


Fig. 11. Fuzzy sets learned by M-GA-PSO with the interpretability improvement.

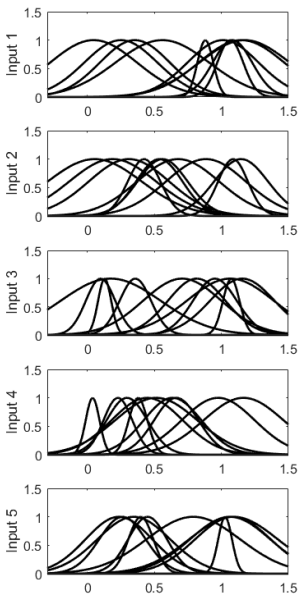


Fig. 12. Fuzzy sets learned by M-GA-PSO without the interpretability improvement.

VI. CONCLUSIONS

This paper investigated the problem of multi-robot navigation and simultaneous arrival coordination in an unknown environment. We presented a two-layer structure of FLCs to ensure safe navigation with coordination of the arrival

times. All three proposed controllers (for obstacle avoidance, behaviour selection and arrival-time coordination) are based on FISs and utilise the self-constructing neural fuzzy inference network (SONFIN) [22] methodology for the construction of fuzzy rules. By splitting the large-scale feature space into several smaller ones, we reduced the number of parameters to be learned without compromising the performance of the low-level controller. Additionally, in contrast to traditional behaviour switching mechanisms, we proposed a novel rule-embedded FLC to improve the navigation performance. Moreover, we developed a grouping and merging mechanism to obtain transparent fuzzy sets and integrated this mechanism into the training process for all FLCs, thus increasing the interpretability of the fuzzy models. To design these controllers automatically and efficiently, we developed the hybrid M-GA-PSO method, which simultaneously leverages the exploration capability of a GA and the convergence capability of PSO. Simulation results demonstrate that with our approach, various numbers of robots can be successfully controlled to safely navigate and reach their target simultaneously. In future work, we will attempt to develop a systematic method for optimising the hyper-parameters of such cascaded models, including the number of rules in the controllers.

REFERENCES

[1] P. Yao and S. Qi, "Obstacle-avoiding path planning for multiple autonomous underwater vehicles with simultaneous arrival," *Science China Technological Sciences*, vol. 62, no. 1, pp. 121–132, 2019.

[2] S. Misra, A. Chakraborty, R. Sharma, and K. Brink, "Cooperative simultaneous arrival of unmanned vehicles onto a moving target in gps-denied environment," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 5409–5414.

[3] L. Babel, "Coordinated target assignment and uav path planning with timing constraints," *Journal of Intelligent & Robotic Systems*, vol. 94, pp. 857–869, June 2019.

[4] F. Ingrand and M. Ghallab, "Deliberation for autonomous robots A survey," *Artificial Intelligence*, vol. 247, pp. 10–44, June 2017.

[5] M. Khosravi, S. Enayati, H. Saeedi, and H. Pishro-Nik, "Multi-purpose drones for coverage and transport applications," *IEEE Transactions on Wireless Communications*, pp. 1–1, February 2021.

[6] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey." *Networks*, vol. 72, no. 4, pp. 411–458, December 2018.

[7] H. Park, D. Son, B. Koo, and B. Jeong, "Waiting strategy for the vehicle routing problem with simultaneous pickup and delivery using genetic algorithm," *Expert Systems with Applications*, vol. 165, p. 113959, 2021.

[8] N. Peinecke and A. Kuenz, "Deconflicting the urban drone airspace," in *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*. IEEE, 2017, pp. 1–6.

[9] J. K. Pothal and D. R. Parhi, "Navigation of multiple mobile robots in a highly clutter terrains using adaptive neuro-fuzzy inference system," *Robotics and Autonomous Systems*, vol. 72, pp. 48–58, October 2015.

[10] C.-F. Juang and Y.-C. Chang, "Evolutionary-group-based particle-swarm-optimized fuzzy controller with application to mobile-robot navigation in unknown environments," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 2, pp. 379–392, April 2011.

[11] C.-H. Chen, C.-J. Lin, S.-Y. Jeng, H.-Y. Lin, and C.-Y. Yu, "Using ultrasonic sensors and a knowledge-based neural fuzzy controller for mobile robot navigation control," *Electronics*, vol. 10, no. 4, p. 446, February 2021.

[12] H. Ishibuchi, T. Murata, and I. B. Türkşen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets and Systems*, vol. 89, no. 2, pp. 135–150, July 1997.

[13] M. Cococcioni, P. Ducange, B. Lazzerini, and F. Marcelloni, "A pareto-based multi-objective evolutionary approach to the identification of mamdani fuzzy systems," *Soft Computing*, vol. 11, no. 11, pp. 1013–1031, Sept. 2007.

[14] M. J. Gacto, R. Alcalá, and F. Herrera, "Integration of an index to preserve the semantic interpretability in the multiobjective evolutionary rule selection and tuning of linguistic fuzzy systems," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 3, pp. 515–531, Jun. 2010.

[15] P. Pulkkinen and H. Koivisto, "A dynamically constrained multiobjective genetic fuzzy system for regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 161–177, Feb. 2010.

[16] R. Alcala, M. J. Gacto, and F. Herrera, "A fast and scalable multi-objective genetic fuzzy system for linguistic fuzzy modeling in high-dimensional regression problems," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 4, pp. 666–681, Aug. 2011.

[17] C.-F. Juang, T.-L. Jeng, and Y.-C. Chang, "An interpretable fuzzy system learned through online rule generation and multiobjective aco with a mobile robot control application," *IEEE transactions on cybernetics*, vol. 46, no. 12, pp. 2706–2718, 2015.

[18] F. Aghaeipoor and M. M. Javidi, "Mokbl+moms: An interpretable multi-objective evolutionary fuzzy system for learning high-dimensional regression data," *Information Sciences*, vol. 496, pp. 1–24, Sept. 2019.

[19] M. I. Rey, M. Galende, M. J. Fuente, and G. I. Sainz-Palmero, "Multi-objective based fuzzy rule based systems (frbss) for trade-off improvement in accuracy and interpretability: A rule relevance point of view," *Knowledge-Based Systems*, vol. 127, pp. 67–84, July 2017.

[20] W. Zhang and Y. Zhang, "Behavior switch for drl-based robot navigation," *2019 IEEE 15th International Conference on Control and Automation (ICCA)*, pp. 284–288, 2019.

[21] Z. Wang, C. Cheng, P. Blunsom, A. Markham, L. Xie, N. Trigoni, Y. Miao, and W. Sen, "Learning with stochastic guidance for robot navigation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 166–176, January 2020.

[22] C.-F. Juang and C.-T. Lin, "An online self-constructing neural fuzzy inference network and its applications," *IEEE Transactions on Fuzzy Systems*, vol. 6, no. 1, pp. 12–32, 1998.

[23] Y.-C. Chang, A. Dostovalova, C.-T. Lin, and J. Kim, "Intelligent multirobot navigation and arrival-time control using a scalable pso-optimized hierarchical controller," *Frontiers in Artificial Intelligence*, vol. 3, August 2020.