Robotica: page 1 of 15. © Cambridge University Press 2019
doi:10.1017/S0263574719000353

Collaborative Control of Multiple Robots Using Genetic Fuzzy Systems

Anoop Sathyan*, and Ou Ma

Department of Aerospace Engineering, University of Cincinnati, Cincinnati, OH 45221, USA.
E-mail: ou.ma@uc.edu

(Accepted 3 March 2019)

SUMMARY
This paper introduces an approach of collaborative control for individual robots to collaboratively perform a common task, without the need for a centralized controller to coordinate the group. The approach is illustrated by an application example involving multiple robots performing a collaborative task to achieve a common goal. The objective of this example problem is to control multiple robots that are connected to an object through elastic cables in order to bring the object to a target position. There is no communication between the robots, and hence each robot is unaware of how the other robots are going to react at any instant. Only the information pertaining to the object and the target is available to all the robots at any instant. Genetic fuzzy system (GFS) is used to develop controller for each of the robots. The nonlinearity of fuzzy logic systems coupled with the search capability of genetic algorithms provides a tool to design controllers for such collaborative tasks. A set of training scenarios are developed to train the individual robot controllers for this task. The trained controllers are then tested on an extensive set of scenarios. This paper describes the development process of GFS controllers for dynamic case involving systems consisting of three robots. It is also shown that the GFS controllers are scalable for the more complex systems involving more than three robots.

KEYWORDS: Collaborative control; Genetic fuzzy system; Decentralized control; Intelligent systems; Machine learning; Cable robot.

1. Introduction

We investigate the applicability of the genetic fuzzy systems (GFSs), as an artificial intelligence approach, for a system of robots to accomplish a common task collaboratively. Such collaborative robotics technology can find numerous applications including multiple robots capturing a noncooperative object, transporting a large piece of hardware, cloud-based 3D mapping,[1] helping human workers in collaborative tasks,[2,3] coordinated search,[4] etc. Collaborative robots have also been used for moving a faulty robot to a target position.[5] Unlike many existing collaborative control strategies, the proposed approach assumes no centralized controller for the overall multi-robot system, and thus the involved robots are independently controlled. In other words, each robot is unaware of the state and specific future action of the partner robots although everyone in the group is aware of their common goal.

With the increase in computational capability and the advent of new and improved machine learning algorithms over the last decade, there has been an increase in the development of intelligent systems for various applications. Such intelligent systems provide significant advantages in terms of adaptability, robustness to uncertainties, and improved efficiency. Another advantage of intelligent systems is that it provides the ability to include a variety of inputs to make better decisions which in turn leads to increased efficiency in performing physical tasks.

* Corresponding author. E-mail: sathyaap@ucmail.uc.edu

Fuzzy logic system (FLS), sometimes referred to as Fuzzy Inference System (FIS), is one such intelligent system that can provide robustness and adaptability to robot controllers. Coupling this with an optimization heuristic like genetic algorithm (GA) gives a robot the ability to learn from the past experience (data). Such GFSs have been developed with much success for clustering and task planning,[6] simulated air-to-air combat,[7] aircraft conflict resolution,[8] etc. Since FLSs consist of a set of membership functions that define the inputs and a set of linguistic rules that define the relationship between the inputs and outputs, it is more interpretable compared to other machine learning techniques such as neural networks and support vector machines. Since it is trained using GA, differentiable cost functions such as integral squared error are not a requirement. So, if the mission requirement can be defined as a mathematical cost function, we do not need to have ground truth data available. GA will traverse the search space looking for the optimal set of membership functions and rulebase that minimizes the cost function, which makes it a form of reinforcement learning. Reinforcement learning is a branch of machine learning where an agent is trained to take the optimal control action to maximize a reward.

Fuzzy logic has been used for several robotics applications purely using expert knowledge without any tuning of the fuzzy logic parameters. Hagras presented an FLS[9] that improves the performance under uncertainties for the real-time control of mobile robots navigating in dynamically changing indoor and outdoor environments. This architecture required a smaller rulebase while improving the performance of the controller. Mobadersany *et al.*[10] presented a fuzzy-logic-based real-time approach for path planning to navigate a robot in an environment consisting of unknown moving obstacles. The efficiency of their approach was examined in the case of a drug-delivery nano-robot moving through blood vessels. This method was able to achieve near-optimal paths and robustness under different conditions. Seraji and Howard[11] developed FLS using expert knowledge for robot navigation on difficult terrain. FLSs have an inherent robustness[12] which makes it very appealing for a lot of real-life applications. FLSs have also been developed for online tuning of Proportional-Integral-Derivative (PID) controllers.[13, 14] Instead of using constant proportional (k_p), integral (k_i), and derivative (k_d) gains, these values are constantly updated based on the current error (e) and change in error (Δe).

Although expert knowledge can be used to build FLSs and this capability is appealing to a lot of applications, it makes sense to have a mechanism to tune the parameters of the FLS automatically. Self-tuning FLSs are very useful especially when there are many inputs and outputs and their relationships are not that straightforward or well known. As the number of inputs and outputs increases, it becomes increasingly difficult to manually tune the parameters, both membership functions and rulebase, of FLSs. Jang presented an architecture and learning procedure called ANFIS (Adaptive Network based Fuzzy Inference System)[15] that uses the backpropagation algorithm, normally employed for training neural networks, to tune the membership functions and rulebase of an FLS. This is a popular training approach that has been successfully used for a lot of estimation and prediction tasks.[16–18] Shimojima *et al.*[19] presented a supervised learning mechanism based on GA which also uses the gradient descent principle to shape the membership functions and obtain the consequents of the rules. Jain *et al.*[20] presented a tuning process using simulated annealing to optimize the parameters of the FLSs for nonlinear Single-input-single-output (SISO) and Multiple-input-multiple-output (MIMO) systems for controlling the outlet temperature of a heat exchanger as well as a nonlinear coupled tank system.

Artificial neural networks (ANN) have also been traditionally used for developing controllers. The ability to learn from data by applying the backpropagation algorithm has meant that neural networks are applied to broad range of applications. Control agents can be trained using reinforcement learning to take optimal actions at every instant to reach a final desired state. Q-learning, which is a form of reinforcement learning, has gained a lot of popularity recently in training ANNs and convolutional neural networks (CNNs) for various applications including training agents to autonomously play Atari games,[21] the development of the AlphaGo system that defeated professional human Go players,[22] etc.

In this paper, we develop GFSs to control multiple robots that are connected to an object through cables to bring the object to any desired position within the workspace of the robots. The advantage of using FLS is that they can model nonlinear systems and are inherently robust. GA is used to train the FLS on a set of training scenarios by tuning the membership functions and the linguistic rulebase. GA is a search heuristic inspired from the process of natural selection that can perform an extensive search of a complicated n-D space, where n is the number of variables, to find a near optimal solution.[23] GA starts off with an initial set of solutions also called a population of individuals,

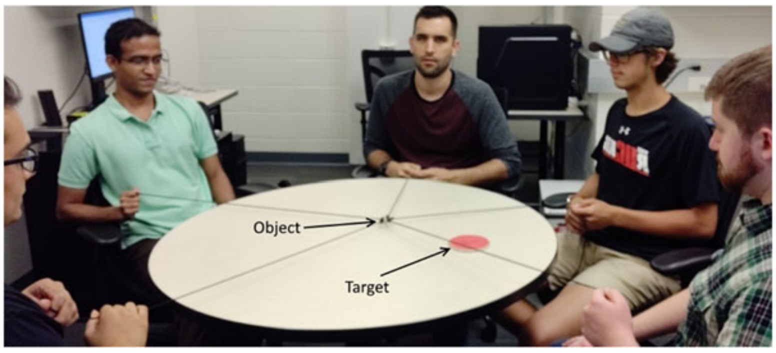

Fig. 1. A game played collaboratively by a group of people. The objective for the individuals is to collaboratively control the cables to bring the object to the target position.

the size of which is predefined for the particular optimization problem being solved. Each individual solution in a population is called a chromosome. During each generation, a set of chromosomes are selected for crossover. The chromosomes are ranked based on their fitness values (calculated using the fitness function that need to be maximized or a cost function that is minimized), and the higher ranked individuals have a greater probability of being selected for crossover. Such a selection process is known as roulette wheel selection. Once selected, the process includes doing a crossover on pairs of chromosomes to obtain two child chromosomes. Mutation is often performed on certain selected chromosomes where a randomly chosen cell of the chromosome is arbitrarily changed. Although the approach described here is widely used, there are other operators that take the place of crossover and mutation like self-crossover,[24] regrouping, migration, etc.[25] Other selection functions such as tournament selection and reward-based selection are also in use.[26,27] In some cases, the crossover and mutation operators need to be modified to make sure that these operators produce legitimate solutions.

For this research, the FLS is trained to output an action at each instant that can help with bringing the object to the desired position within a particular time-frame. Since the system uses decentralized control, this approach is expected to be more scalable especially during training process.

2. Problem Statement

The inspiration for this problem comes from a game involving people pulling on cables to collaboratively control the position of an object to bring it towards a target, as shown in Fig. 1. The people participating in this game do not talk to each other; instead they focus on the target and try to adjust their own efforts of pulling and releasing until the object reaches the target position. The strategy involved in such a game can reflect the human capabilities of learning and gaining experience with respect to decentralized control and collaboration. Inspired by this human game, we are developing decentralized control strategy and algorithms to allow individual robots to perform similar activities showing the capability of multi-robot collaboration.[28,29] Such a collaborative activity might be rather easy for humans, but quite challenging for robots because of the current limitations of robot intelligence. As a starting point, we use planar cable robots to control the position of the object.

The motion plane of the robots is assumed to be horizontal. The objective is to have the robots to work collaboratively to bring the object to an arbitrarily defined target position by pulling or retracting the elastic cables that are connected to the object. One major constraint is that each robot's controller only has information about the target and the object and does not have any knowledge about the states of the partner robots. Thus, this provides a good example to test a set of robots that can work collaboratively without the need for centralized control. To test the feasibility of the proposed intelligent approach, we trained a GFS as a controller for each individual robot to control the position of the object.

The cables are pulled or released by controlling the joint of each robot, which are attached to a spool over which the cable can wind. Depending on the rotation of each joint, the cable reeled by the joint either extends or compresses.

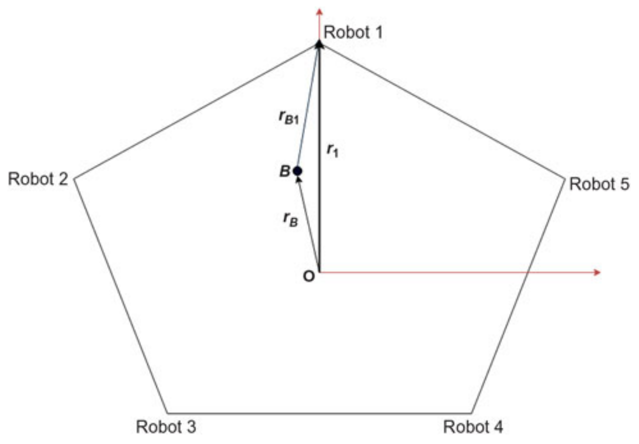

Fig. 2. Schematic showing the different vectors for a 5-robot system. This figure can be generalized for an n-robot system.

3. Multi-robot Problem

This section describes the dynamic analysis for the multi-robot problem.

Let $\boldsymbol{r_{Bi}}$ be the vector connecting the object to robot i. Let p_i be the length of the cable around the spool attached to robot i. $\boldsymbol{r_{Bi}}$ can be written in terms of the position vectors of robot i and the object, as follows.

$$\boldsymbol{r_{Bi}} = \boldsymbol{r_i} - \boldsymbol{r_B} \tag{1}$$

The total length of the cable connecting object to robot i will be $r_{Bi} + pi$. Let l_0 be the minimum length of the elastic cable and k be the spring constant of the cable. Fig. 2 shows the visual representation of the different vector relations. The tension on this cable will act along the unit vector, $\boldsymbol{\hat{r}_{Bi}}$, and is given by

$$\boldsymbol{T_i} = k(r_{Bi} + p_i - l_0)\boldsymbol{\hat{r}_{Bi}} = k[(p_i - l_0)\boldsymbol{\hat{r}_{Bi}} + \boldsymbol{r_{Bi}}] \tag{2}$$

Substituting for $\boldsymbol{r_{Bi}}$ from Eq. (1) in Eq. (3),

$$\boldsymbol{T_i} = k[(p_i - l_0)\boldsymbol{\hat{r}_{Bi}} + \boldsymbol{r_i} - \boldsymbol{r_B}] \tag{3}$$

Let the mass of the object be m. Then, by applying Newton's second law, the following relationship is obtained.

$$\sum_i \boldsymbol{T_i} = m\boldsymbol{\ddot{r}_B} \tag{4}$$

Since the robots are placed on the vertices of a regular polygon, the following relationship holds.

$$\sum_i r_i = \boldsymbol{0} \tag{5}$$

Substituting Eqs. (3) and (5) into Eq. (4) gives the following governing equation for the object under equilibrium condition. Here, n refers to the number of robots in the system.

$$k\sum_i ((p_i - l_0)\boldsymbol{\hat{r}_{Bi}}) - n\boldsymbol{r_B} = m\boldsymbol{\ddot{r}_B}$$
$$\text{where } \boldsymbol{\hat{r}_{Bi}} = \frac{\boldsymbol{r_i} - \boldsymbol{r_B}}{|\boldsymbol{r_i} - \boldsymbol{r_B}|} \tag{6}$$

Eq. (6) is a nonlinear equation in $\boldsymbol{r_B}$ as the unit vector, $\boldsymbol{\hat{r}_{Bi}}$, has the term $|\boldsymbol{r_i} - \boldsymbol{r_B}|$ in the denominator. Eq. (6) is a 2-D vector equation pertaining to the motion of the object which is connected to the robots through the elastic cables. Thus, the objective of the system can be written as

$$\begin{aligned} &\text{minimize} \quad J = \int_0^T \text{dist}(t)dt \\ &\text{subject to} \quad k\sum_i((p_i - l_0)\hat{\boldsymbol{r}}_{\boldsymbol{Bi}}) - n\boldsymbol{r}_{\boldsymbol{B}} = m\ddot{\boldsymbol{r}}_{\boldsymbol{B}} \end{aligned} \tag{7}$$

Here, $dist(t)$ refers to the distance between the object and the target at time t. In this case, each robot has only one joint driven by a DC motor. For our simulation study, the specifications of an actual DC motor were used to set the limits for the motor torque and angular velocity and the values for the other motor parameters. Since we are designing controllers to control the voltages of the DC motors for the dynamic case, it is necessary to obtain the electro-mechanical equations corresponding to the joint. These equations are given below.

$$p_i = r_s * \theta_i \tag{8}$$

$$K_T I_i - T_i r_s - b\dot{\theta}_{Gi} = J\ddot{\theta}_{Gi} \tag{9}$$

$$L\frac{dI_i}{dt} + I_i R = V_i - K_e\dot{\theta}_{Gi} \tag{10}$$

$$I = \frac{K_T}{G_r}T_{Gi} \tag{11}$$

Eqs. (8)–(11) should be written for each robot. The joint is driven through a gear with a gear ratio of G_r. In the equations, K_T is the torque constant of the motor; K_e is the back-emf constant of the motor; T_G is the torque output to the spool; θ_G is the angular velocity of the spool; T is the cable tension force; r_s is the radius of the spool; $b\dot{\theta}_G$ is a damping torque; J is the moment of inertia of the spool or the motor load; i is the current flowing through the motor circuit; and V is the voltage applied by the controller. Eqs. (6)–(11) give the relationship between the robot voltages and the motion of the object. Thus, the objective of this example case is to develop robotic controllers that can control the position of the object by controlling the voltages of the respective driving motors.

4. GFS

Fuzzy logic, in contrast with Boolean logic, deals with degrees of truth rather than a binary True or False. There is a membership value for each value in a fuzzy set, which is more in tune with the human thought process. This gives a fuzzy boundary between two sets. This process of converting a crisp value to fuzzy membership values that represent the membership level of the crisp value to each of the fuzzy sets is called fuzzification. Fuzzification process is done by defining membership functions for each input to the FLS. Another important aspect of an FLS is the rulebase which includes a set of linguistic rules that connect the inputs to the outputs. An example for an FLS rule is *If Food is Good and Service is Good then Tip is Good.*

For a small-scale FLS, the rules can be defined using expert knowledge of the relation between inputs and outputs due to the linguistic nature of the rulebase. But, this becomes difficult for larger FLSs and FLSs where the relation between the inputs and outputs are not clearly known. It is under these scenarios where it is necessary to augment the FLS with a search heuristic such as GA to provide the FLS with the learning capability, and such systems are known as GFSs.

In order to evaluate the output of an FLS which is a crisp value, we have to evaluate each rule in the rulebase and then defuzzify the resultant aggregate solution. The aggregate solution will be a region defined under the membership function for the output variables, similar to the one shown in Fig. 3. There are different defuzzification strategies used,[30] the most popular of which is the centroid defuzzification. Centroid defuzzification gives the x-value of the centroid of the area which is generally close to the middle of the range of the output variables. The probability of centroid defuzzification to output a value close to the extremes of the output variable is very small. As can be seen from Fig. 3, largest of mean (LOM) defuzzification outputs the value that has the largest absolute value of x where the membership value is the highest.

In the case of GFS, the set of parameters to be tuned include the boundaries of the membership functions and the set of rules in the rulebase. For some applications, GA is also used to tune the shape of the membership functions, although in most cases it is safe to assume triangular and trapezoidal

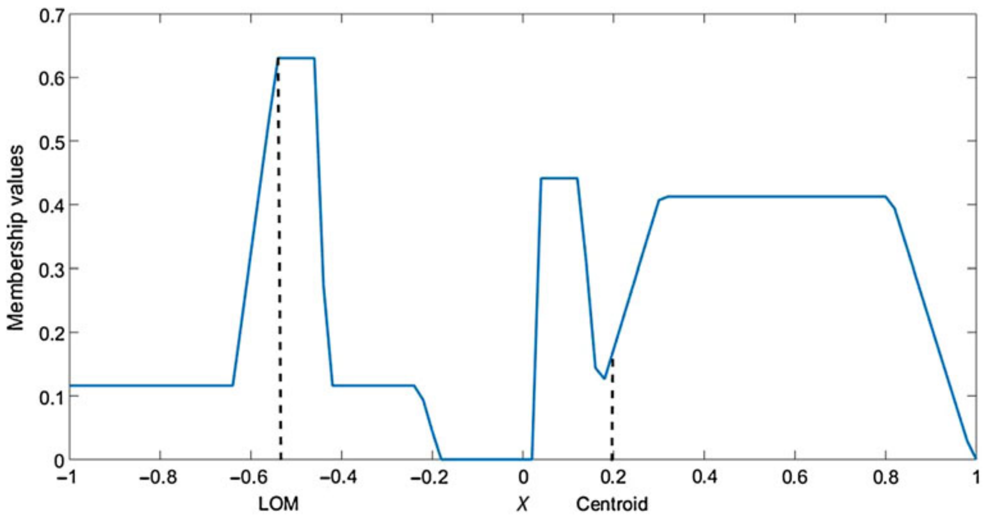

Fig. 3. Centroid and LOM defuzzification outputs.

membership functions. Intelligent systems such as GFSs are used in a wide variety of applications due to their learning and adaptive capabilities. Cordon and Herrera have presented an overview of the GFSs, showing the use of the GAs in the construction of the FLS knowledge bases.[31]

Genetic fuzzy methodology can be used to tune different types of membership functions depending on how they are defined. Triangular membership functions are defined by the x-coordinates of the three vertices. Symmetric triangular membership functions can be defined using the center of the base of the triangle and its width. Gaussian membership functions are defined using the mean (center) and standard deviation. In this paper, we only consider triangular membership functions.

5. Methodology for Training the Robots

In order to achieve the common goal, each robot needs to be trained in a collaborative environment using a set of training scenarios. These training scenarios will have different target positions for the object. By training the robots to achieve the goal of moving the object to these different target positions, the robots will learn to work collaboratively to bring the object to any target location within the polygon connecting the robots. Each robot is modeled as a GFS and they have information only pertaining to the state of the object with respect to the target location and has no information about the states of the partner robots.

The schematic of the training process is shown in Fig. 4. GA starts off with a set of individuals for the population. Each individual is a vector that consists of parameter values for the multi-robot system. For each individual in GA, the system could be simulated to evaluate the cost function. The cost function would be representative of bringing the object to the target in the shortest time. The individuals with lower cost values have more likelihood of being selected for crossover and mutation and being chosen into the next generation. The individuals with high cost values have more likelihood of getting thrown out. This process of modifying the population of individuals continues for a predefined number of generations. During each generation, the best system of robots is also evaluated on a validation set to check for overfitting. After GA is finished, the individual with the best training and validation cost is chosen. This individual defines the trained system of collaborative robots that is capable of bringing the object to the target and can be tested on new scenarios to evaluate its effectiveness and generalization capability.

5.1. GFS controllers for the 3-robot system

GFS controllers directly control the voltage of the joint motor of each robot for the next time step. Thus, there are three GFS controllers that control the voltage of the respective DC motor in order to bring the object to the desired position on the table. Eqs. (6), (9), and (10) provide the governing equations of the dynamic system. The equations are solved using a differential equation solver for the dynamic simulation of the overall behavior of the system. The schematic of the GFS controllers is shown in Fig. 5. Each GFS controller takes in four inputs: (1) distance between the current object

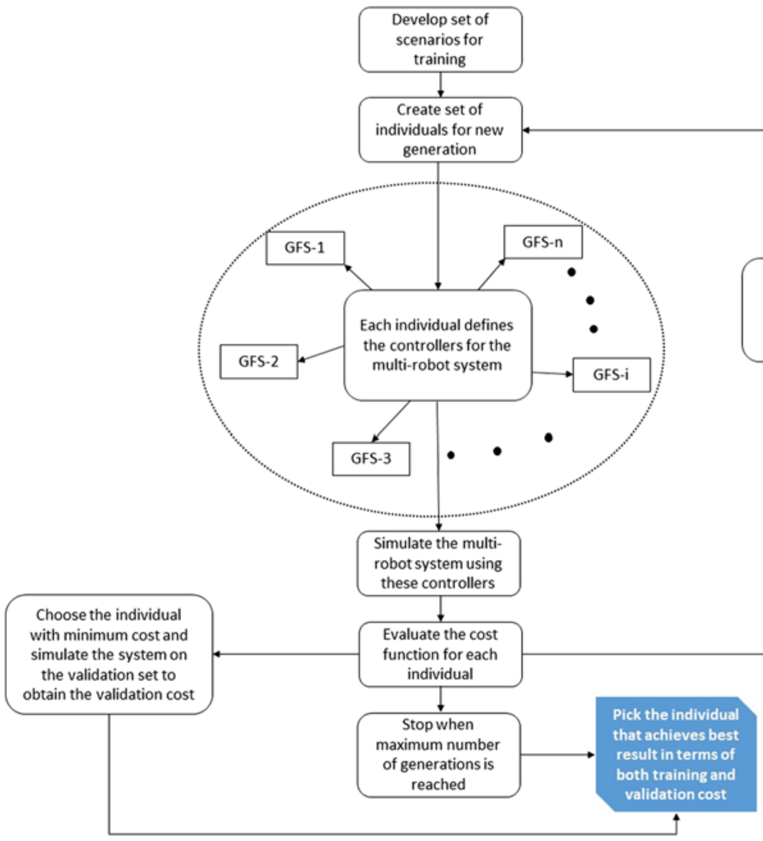
Develop set of scenarios for training
Create set of individuals for new generation
GFS-1
GFS-n
GFS-2
GFS-i
GFS-3
Each individual defines the controllers for the multi-robot system
Use crossover and mutation to modify some of the individuals in the population
Simulate the multi-robot system using these controllers
Choose the individual with minimum cost and simulate the system on the validation set to obtain the validation cost
Evaluate the cost function for each individual
Stop when maximum number of generations is reached
Pick the individual that achieves best result in terms of both training and validation cost

Fig. 4. Schematic showing the training process. The parameters of each robot are tuned using GA to minimize a cost function.

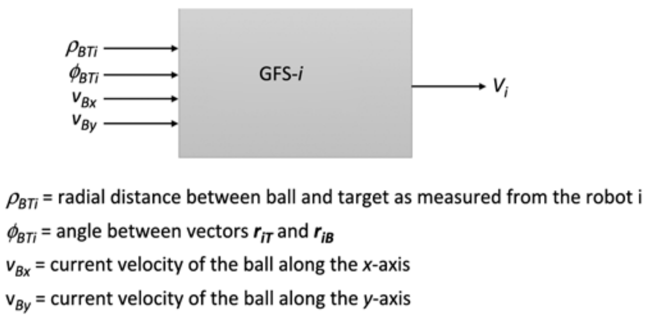
ρ_{BTi}
ϕ_{BTi}
v_{Bx}
v_{By}
GFS-i
V_i
ρ_{BTi} = radial distance between ball and target as measured from the robot i
ϕ_{BTi} = angle between vectors $\boldsymbol{r}_{iT}$ and $\boldsymbol{r}_{iB}$
v_{Bx} = current velocity of the ball along the x-axis
v_{By} = current velocity of the ball along the y-axis

Fig. 5. Schematic for the GFS controllers associated with robot i. The schematic is the same for all robots in the system.

position and the target position measured along the vector connecting the robot to the target, (2) the angle between the object-robot vector and the target-robot vector, (3) x-component of object velocity, and (4) y-component of object velocity.

In order for the controllers to bring the object to the target position, the controllers need to be trained on a set of chosen scenarios. During the training process, the FLS parameters, namely the boundaries of the membership functions as well as the rulebase, are tuned using GA. In parallel with the training, the GFS can be validated using other validation scenarios in order to spot any overfitting. Overfitting happens when the cost function on validation set increases while the training cost continues to decrease. Each scenario is run for a maximum time (T_f) of 15 s. During the training process, GA tunes the membership functions and rulebase of the FISs included in the GFS to minimize the following cost function.

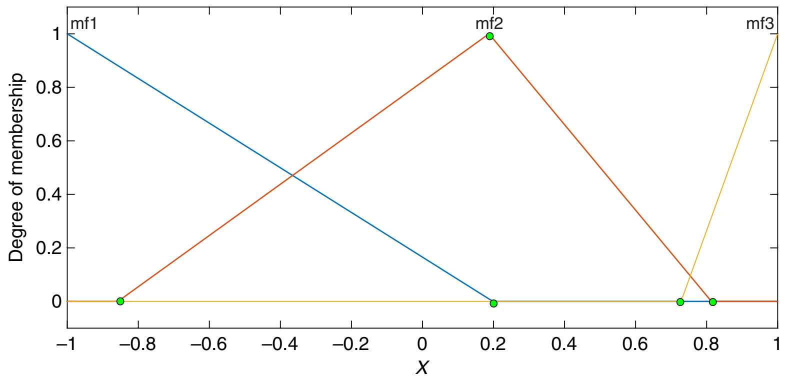

Fig. 6. Input membership functions of the GFSs for the 5-robot problem. The points tuned by GA are shown by the green dots.[29]

$$C = \int_0^{T_f} \text{dist}(t)dt + 50(T_f - t_{end}) \tag{12}$$

T_f is the maximum time and t_{end} refers to the time at which the simulation stops. Apart from reaching the maximum time, T_f, the simulation can also stop when the elastic cable length goes outside the range of 1–2 m. So, the $T_f - t_{end}$ term is used to penalize such early stoppage situations. $dist(t)$ is the distance between the object and the target at each time step.

5.2. *GFS controllers for the 5-robot system*

In order to test the scalability of the approach, GFS controllers were developed for the 5-robot system. The 5-robot problem is much more complicated to control as compared to the 3-robot case. The robots use the same schematic shown in Fig. 5. The output from the GFS is the voltage that need to be applied at the current time step.

During the training process, GA tunes the membership functions and rulebase of the 5-FISs included in the GFS to minimize the cost function given by Eq. (12). The maximum time, T_f, for the 5-robot case is set to 20 s.

In this case, the membership functions are modeled as triangles and GA tunes five membership function parameters for each input variable, as shown in Fig. 6. The x-values of the five green dots are tuned using GA for each input variable. Only one parameter of the two extreme membership functions is tuned. On the other hand, the output variable is defined using five triangular membership functions and each vertex is tuned using GA. This means that GA tunes 15 parameters of the output membership functions for each robot. This provides more learning capacity for the controllers. Also, since the controllers have four inputs and each input variable is defined using three membership functions, there will be $3^4 = 81$ rules in the rulebase for each FIS. Thus, a total of 580 parameters are tuned by GA for the five FISs.

6. Results

6.1. *3-robot system*

The GFSs control the voltages of the three robots to bring the object to the desired target position. LOM defuzzification is used by each GFS. The cable can stretch from 1 to 2 m within which the spring force kx is valid. Outside of this range, the dynamic equation of motion defined in Eq. (6) is invalid. The simulation stops whenever the length of any of the cables goes beyond 2 m. During the training process, such scenarios are penalized in the cost function. The motor is connected to a spool of radius 12.5 mm through a gear with a gear ratio of 2.4 that makes the maximum torque output to the spool to be $2.4T_M$ and the maximum angular velocity to be $\omega_M/2.4$.

The object's path towards the target, the distance *v/s* iterations plot, and the torques of each motor are shown in Figs. 7 and 8. The controller torques for robots 1 and 3 act like a bang-bang controller whereas the torque of robot 2 is relatively smaller. This means that in most cases, robot 2 makes the

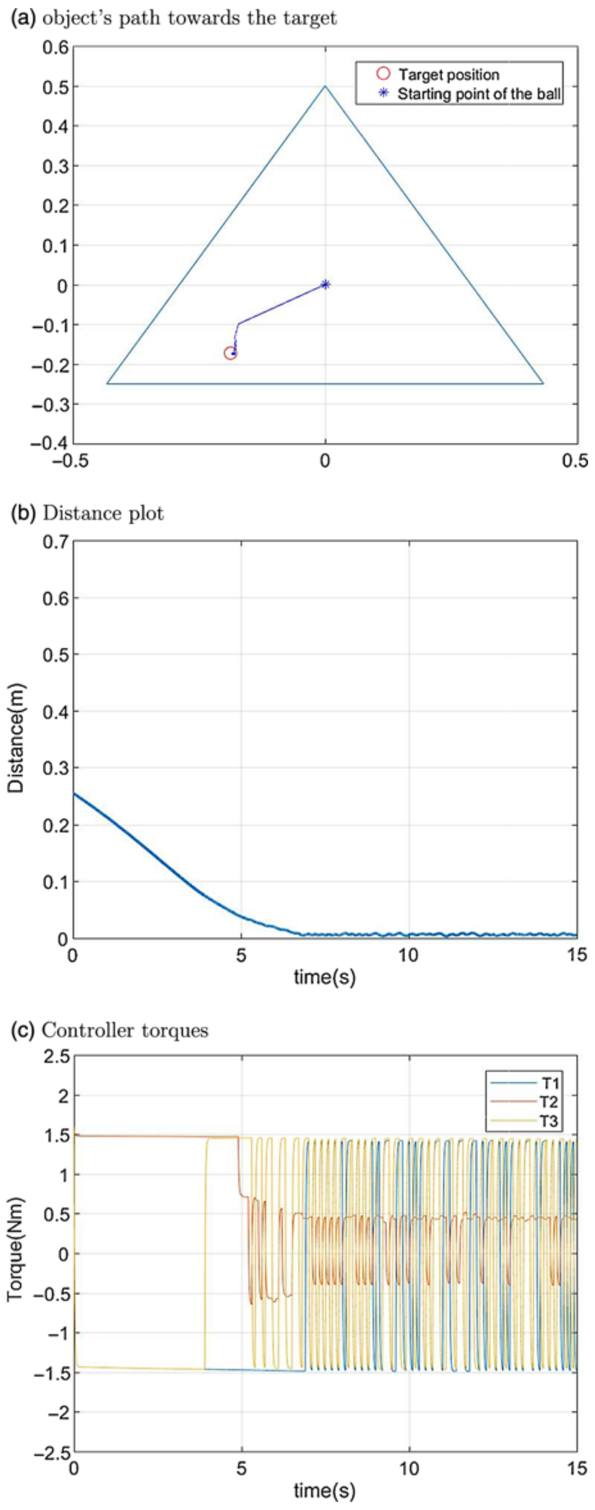

Fig. 7. 3-robot system: Scenario 1.

minor adjustments that are needed to bring the object closer to the target. In fact, the robots were able to bring the object to settle within 2 cm of the target position for all of the 100 scenarios tested.

6.2. *5-robot system*

Once the FISs are trained for the 5-robot case, the system was tested for 100 different target locations within the workspace of the robots. Two of these scenarios are shown in Figs. 9 and 10. The

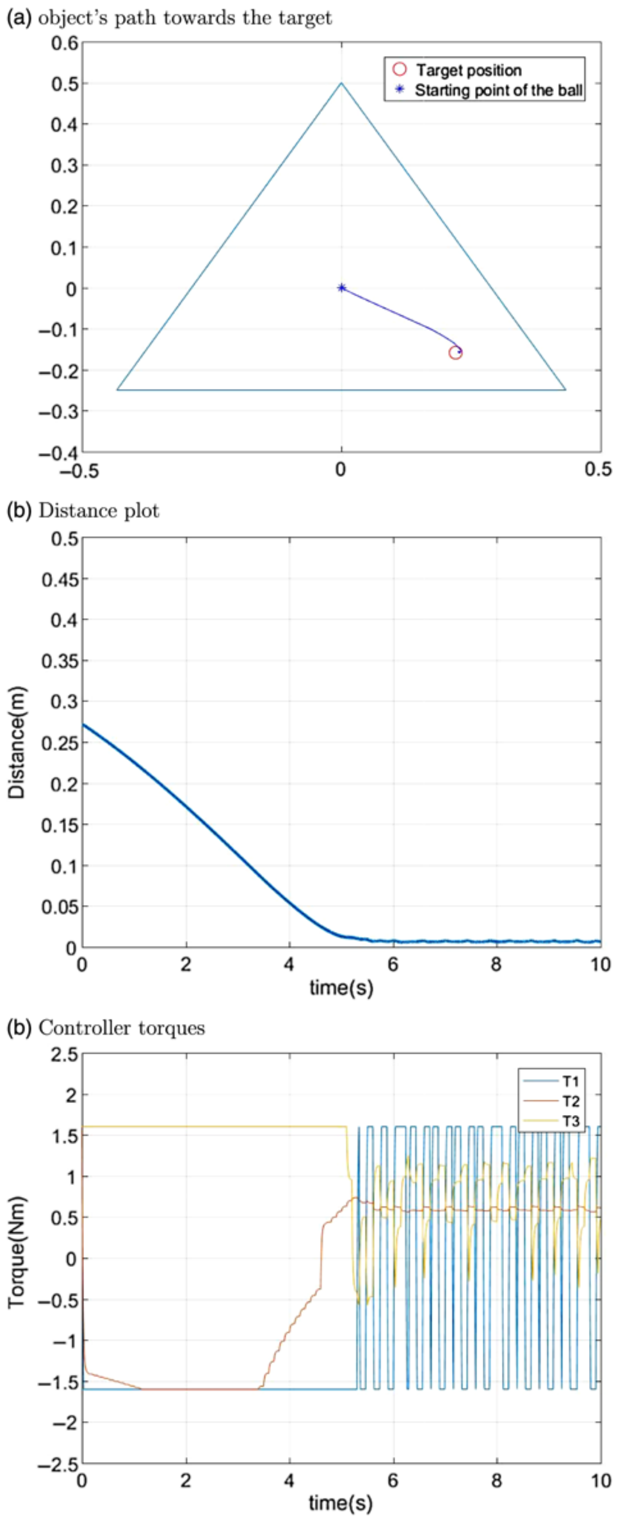

Fig. 8. 3-robot system: Scenario 2.

controllers are able to bring the object to within 2 cm of the target within a time-frame of 20 s. The starting torque is set to 0.59 Nm so that the system starts off from equilibrium. This is unlike the 3-robot scenarios where the three robots started off with the maximum torques of 1.667 Nm. We noticed that the centroid defuzzification works better for the 5-robot problem. The centroid defuzzification takes into account the output from each rule in the rulebase, whereas LOM defuzzification, in

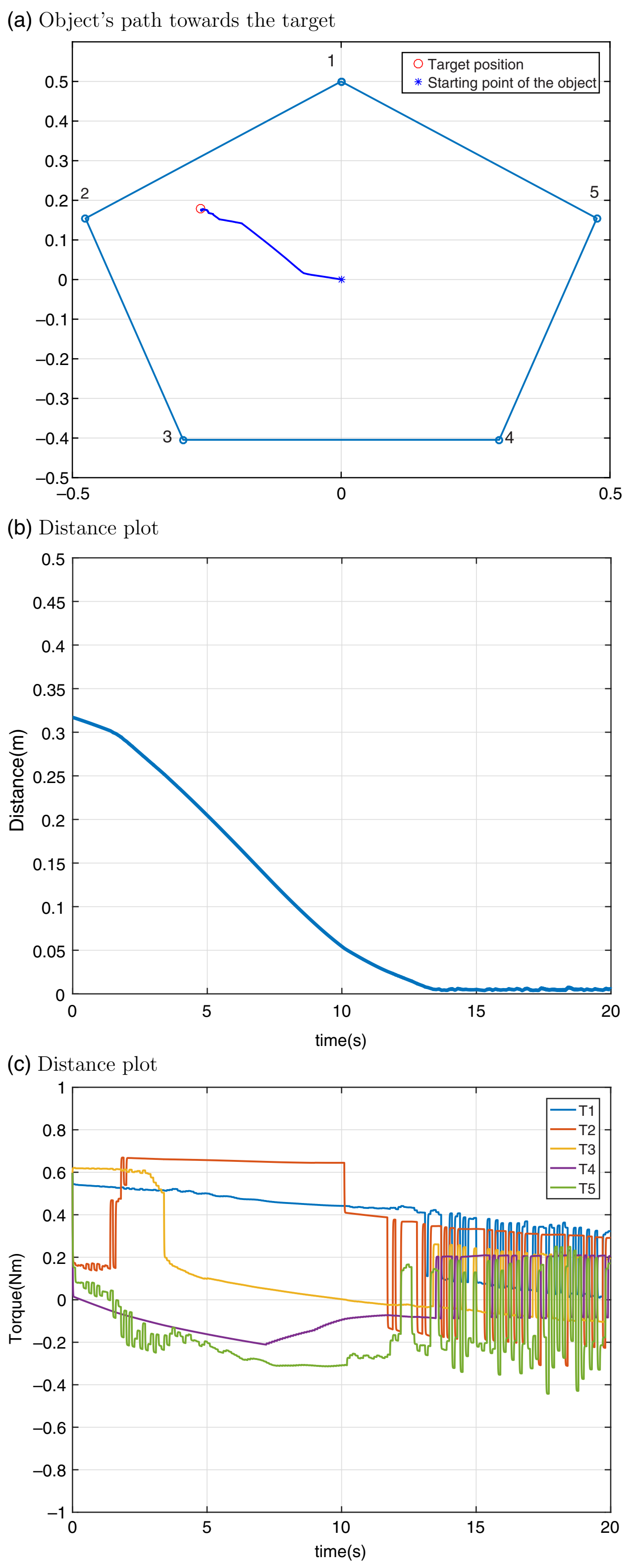

Fig. 9. 5-robot system: Scenario 1.

most cases, only uses the rule with the highest firing strength. If the simulations are run for a larger time-frame, it is noticed that the object could drift outside the target region for many of the scenarios. Increasing the time-frame of the simulations during the training scenarios might be able to solve this

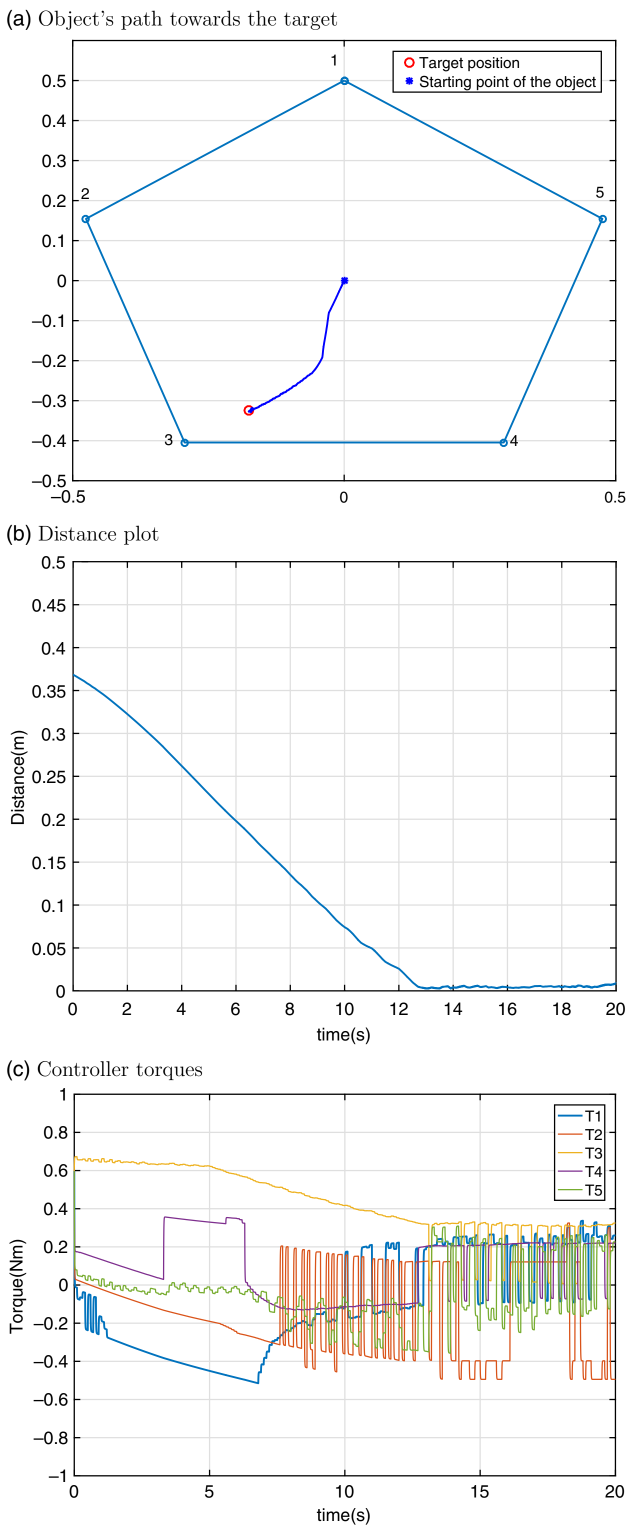

Fig. 10. 5-robot system: Scenario 2.

issue. But, nevertheless, the robots bring the object to the target for 85% of the scenarios tested and stays within 2 cm of the target for approximately 10 s before drifting apart.

Since the object is connected to five elastic cables, there are more vibrational effects on the object for the 5-robot problem. Moreover, the system uses decentralized control where each robot is unaware of the actions of the partner robots. This makes the problem more complex as the number of robots

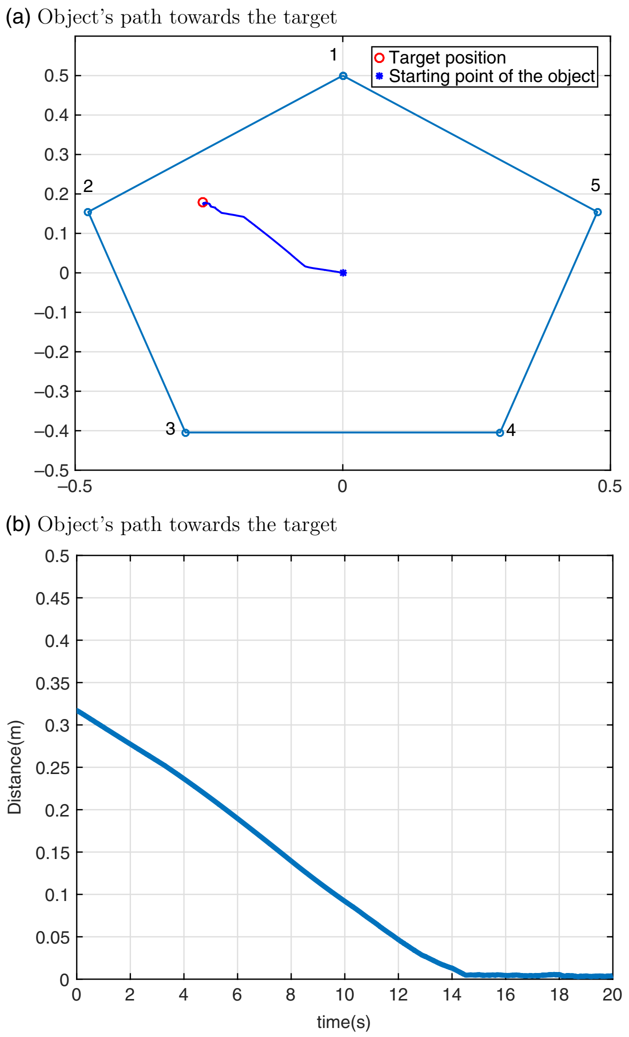

Fig. 11. 5-robot system: Fault occurred in robot 2 during Scenario 1. The remaining four robots were able to bring the object to the target.

increases. This is further complicated by the fact that the cables cannot stretch beyond 2 m. As a result, of the 100 test scenarios, there are some scenarios where the robots are unable to bring the object exactly to the target location. This is still being investigated as we are trying to improve the performance of the controllers.

The robots were also tested on a few scenarios where one of the robots failed. The failed robot was assumed to have no active control capability. It was seen that the other robots were still capable of bringing the object to the target position. Figure 11 shows one scenario where robot 2 failed. This means that the DC motor voltage for this robot is zero ($V_2 = 0$). For this scenario, the target position is the same as that of Scenario 1 (Fig. 9). It can be seen that the failure of robot 2 affects the path followed by the object, although the robots were able to bring the object to the target. This shows that the other four robots were still able to find a way to bring the object to the target in spite of robot 2 behaving like a passive system. It is possible that robot failure would be more fatal for target positions that are closer to the boundaries. Such scenarios will be investigated in more detail in a separate future research effort regarding the fault-tolerance topic.

7. Conclusions and Future Work

This paper presented a decentralized approach for a group of independent robots for collaboratively achieving a common goal. The robots were trained using the GFS methodology. Through the training process performed using GA, the robots were able to come up with an effective strategy to bring the object to the target position. It was shown that once the system was trained, the goal could be quickly achieved for all the scenarios tested in the case of the 3-robot problem.

The same approach was used for training and testing for systems involving five robots. The robots were able to bring the object to the target position for most of the cases. The lack of a central controller and the inter-robot communication reduces any overhead requirement. Such a methodology also ensures that even if one of the robots were to malfunction, the system will not collapse entirely and should still be able to perform even though fewer robots are contributing to the common goal. This will be especially true for problems with larger number of robots where the liability of individual robots will be low. In all the cases where the robots were able to successfully achieve the common goal, the system settled very fast (<15 s). The controllers were able to perform efficiently even though there were several constraints considered in the system such as the maximum length of the cable and the limited degrees of freedom of the robots.

Research is being done on how to improve the efficiency of the robots for the 5-robot problem and how this approach can be used for problems involving more number of robots. As the number of robots increases, the collaborative strategy needed to achieve the goal becomes more complicated. Nevertheless, the success of this approach of the robot from the 3-robot to the 5-robot case means that the proposed GFS approach is scalable for different problem sizes. However, the potential benefits and limitations associated with the scalability are the ongoing investigation topics for the research. We plan on validating this approach on actual hardware. We will also study the fault tolerance aspect, such as the one shown in Fig. 11, in more detail in the future.

Acknowledgments
The authors acknowledge many constructive discussions with Professor Kelly Cohen of the Department of Aerospace Engineering, University of Cincinnati.

References
1. G. Mohanarajah, V. Usenko, M. Singh, R. D'Andrea and M. Waibel, "Cloud-based collaborative 3d mapping in real-time with low-cost robots," *IEEE Trans. Autom. Sci. Eng.* **12**(2), 423–431 (2015).
2. J. Realmuto, R. B. Warrier and S. Devasia, "Data-inferred personalized human-robot models for iterative collaborative output tracking," *J. Intell. Robot. Syst.* **91**(2), 1–17 (2018).
3. U. Kartoun, H. Stern and Y. Edan, "A human-robot collaborative reinforcement learning algorithm," *J. Intell. Robot. Syst.* **60**(2), 217–239 (2010).
4. A. Baranzadeh and A. V. Savkin, "A distributed control algorithm for area search by a multi-robot team," *Robotica* **35**(6), 1452–1472 (2017).
5. G. Eoh, J. S. Choi and B. H. Lee, "Faulty robot rescue by multi-robot cooperation," *Robotica* **31**(8), 1239–1249 (2013).
6. A. Sathyan, N. D. Ernest and K. Cohen, "An efficient genetic fuzzy approach to UAV swarm routing," *Unmann. Syst.* **4**(02), 117–127 (2016).
7. N. Ernest, D. Carroll, C. Schumacher, M. Clark, K. Cohen and G. Lee, "Genetic fuzzy based artificial intelligence for unmanned combat aerial vehicle control in simulated air combat missions," *J. Def. Manag.* **6**(144), 2167–0374 (2016).
8. A. Sathyan, N. Ernest, L. Lavigne, F. Cazaurang, M. Kumar and K. Cohen, "A Genetic Fuzzy Logic Based Approach to Solving the Aircraft Conflict Resolution Problem," **In**: *AIAA Information Systems-AIAA Infotech@Aerospace* (2017) pp. 1751 .
9. H. A. Hagras, "A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots," *IEEE Trans. Fuzzy Syst.* **12**(4), 524–539 (2004).
10. P. Mobadersany, S. Khanmohammadi and S. Ghaemi, "A fuzzy multi-stage path-planning method for a robot in a dynamic environment with unknown moving obstacles," *Robotica* **33**(9), 1869–1885 (2015).
11. H. Seraji and A. Howard, "Behavior-based robot navigation on challenging terrain: A fuzzy logic approach," *IEEE Trans. Robot. Autom.* **18**(3), 308–321 (2002).
12. A. Saffiotti, "The uses of fuzzy logic in autonomous robot navigation," *Soft Comput.* **1**(4), 180–197 (1997).
13. S.-Z. He, S. Tan, F.-L. Xu and P.-Z. Wang, "Fuzzy self-tuning of PID controllers," *Fuzzy Sets Syst.* **56**(1), 37–46 (1993).
14. R. K. Mudi and N. R. Pal, "A robust self-tuning scheme for PI-and PD-type fuzzy controllers," *IEEE Trans. Fuzzy Syst.* **7**(1), 2–16 (1999).
15. J.-S. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Trans. Syst. Man Cybern.* **23**(3), 665–685 (1993).
16. R. Singh, A. Kainthola and T. Singh, "Estimation of elastic constant of rocks using an ANFIS approach," *Appl. Soft Comput.* **12**(1), 40–45 (2012).
17. S. R. Khuntia and S. Panda, "Simulation study for automatic generation control of a multi-area power system by ANFIS approach," *Appl. Soft Comput.* **12**(1), 333–341 (2012).
18. P. Melin, J. Soto, O. Castillo and J. Soria, "A new approach for time series prediction using ensembles of ANFIS models," *Expert Syst. Appl.* **39**(3), 3494–3506 (2012).

19. K. Shimojima, T. Fukuda and Y. Hasegawa, “Self-tuning fuzzy modeling with adaptive membership function, rules, and hierarchical structure based on genetic algorithm,” *Fuzzy Sets Syst.* **71**(3), 295–309 (1995).
20. R. Jain, N. Sivakumaran and T. K. Radhakrishnan, “Design of self tuning fuzzy controllers for nonlinear systems,” *Expert Syst. Appl.* **38**(4), 4466–4476 (2011).
21. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature* **518**(7540), 529–533 (2015).
22. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel and D. Hassabis, “Mastering the game of Go with deep neural networks and tree search,” *Nature* **529**(7587), 484–489 (2016).
23. D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning* (Addison Wesley, Boston, MA, 1989).
24. N. Ernest and K. Cohen, “Fuzzy Clustering Based Genetic Algorithm for the Multi-Depot Polygon Visiting dubzins Multiple Traveling Salesman Problem,” **In**: *Infotech@Aerospace 2012* (2012), Garden Grove, CA, p. 2562.
25. R. Akbari and K. Ziarati, “A multilevel evolutionary algorithm for optimizing numerical functions,” *Int. J. Ind. Eng. Comput.*, **2**(2), 419–430 (2011).
26. B. L. Miller and D. E. Goldberg, “Genetic algorithms, tournament selection, and the effects of noise,” *Complex Syst.* **9**(3), 193–212 (1995).
27. K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Trans. Evol. Comput.* **6**(2), 182–197 (2002).
28. A. Sathyan and O. Ma, “Collaborative Control of Multiple Robots Using Genetic Fuzzy Systems Approach,” **In**: *ASME 2018 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers (2018) p. V001T03A002.
29. A. Sathyan, O. Ma and K. Cohen, “Intelligent Approach for Collaborative Space Robot Systems,” **In**: *2018 AIAA SPACE and Astronautics Forum and Exposition* (2018), Orlando, FL, p. 5119.
30. R. R. Yager and L. A. Zadeh, *An Introduction to Fuzzy Logic Applications in Intelligent Systems*, vol. 165 (Springer Science & Business Media, New York, 2012).
31. O. Cordón and O. F. Herrera, *A general study on genetic fuzzy systems* (John Wiley and Sons, Hoboken, NJ, USA, 1993) p. 1–25.