Robotica (2019) volume 37, pp. 691–707. © Cambridge University Press 2018
doi:10.1017/S0263574718001261

A Novel Framework for Multi-Agent Systems Using a Decentralized Strategy

Mehmet Serdar Güzel†*, Vahid Babaei Ajabshir†, Panus Nattharith‡, Emir Cem Gezer† and Serhat Can†

† *Computer Engineering Department, Ankara University, Ankara, Turkey.*
E-mails: vahid.babaei@ankara.edu.tr, emir.cem.gezer@ogrenci.ankara.edu.tr, serhat.can@ogrenci.ankara.edu.tr
‡ *Faculty of Engineering, Naresuan University, Phitsanulok 65000, Thailand.*
E-mail: panusn@nu.ac.th

(Accepted November 01, 2018. First published online: December 4, 2018)

SUMMARY
This work addresses a new framework that proposes a decentralized strategy for collective and collaborative behaviours of multi-agent systems. This framework includes a new clustering behaviour that causes agents in the swarm to agree on attending a group and allocating a leader for each group, in a decentralized and local manner. The leader of each group employs a vision-based goal detection algorithm to find and acquire the goal in a cluttered environment. As soon as the leader starts moving, each member is enabled to move in the same direction by staying coordinated with the leader and maintaining the desired formation pattern. In addition, an exploration algorithm is designed and integrated into the framework so as to allow each group to be able to explore goals in a collaborative and efficient manner. A series of comprehensive experiments are conducted in order to verify the overall performance of the proposed framework.

KEYWORDS: Multi-agent systems; Decentralized architecture; Vision for navigation; Swarm intelligence; Pattern formation.

1. Introduction

Swarm intelligence is a scientific discipline, inspired by the behaviours of social animals. This discipline incorporates the fields of swarm intelligence and collective robotics to organize and coordinate robots to complete challenging tasks within a reasonable time. Multi-agent systems are considered to be aggregations of autonomous agents, resembling the swarm intelligence field in some ways.[1,2] Consequently, both concepts will be considered together in this study. Collective behaviours of multi-agent systems can be categorized into three functional groups, namely collective decision making, navigation, and spatial organizing behaviours.[1,3,4] Aggregation (or clustering) is a vital and critical spatial organizing behaviour that allows a group of agents to get close one other, and includes interaction and collective movements.[3] Aggregation is a common behaviour and can be used for various tasks in nature, such as by ants, fish, bees, so on.[4,5] Probabilistic finite state machines (PFSMs) are a conventional and widespread approach used for aggregation behaviour, ensuring that only a single aggregate is formed at the end of the process. Each unregistered robot in the environment aims to find other robots in order to register itself with a group. When other robots are observed, the unregistered robot decides whether to join (register) or leave (stay unregistered) the aggregate in a stochastic manner.[1,6] A comprehensive review paper regarding to similar problems in swarm robotics and their applicability in various tasks can also be seen in ref. [7].

* Corresponding author. E-mail: mguzel@ankara.edu.tr

An artificial evolution approach has alternatively been described for the automatic estimation of aggregation (clustering) behaviour.[8] Flocking, the coordinated motion of animals, essentially allows species to move in a coordinated way, inspired by fish and flocks of birds.[9] In multi-agent systems, this coordinated motion behaviour, designed based on a virtual physics-based scheme, allows smoother and safer navigation for a group of robots while keeping a static or adaptable distance from one another.[5]

One of the critical studies in this area utilizes virtual heading sensors (VHS), which allow a robot to receive the relative headings of nearby robots without needing an accurate goal direction. Using the VHS system, the swarm robots gain the ability to move in a coordinated manner while evading obstacles.[10]

A further study using VHS proves that it is possible to attach some "informed" robots, which know the goal direction, to a multi-agent system so as to steer the other "non-informed" robots towards the goal direction.[11] Alternatively, another comprehensive study in the coordinated motion field allows robots to change both their angular and forward speeds with respect to the resultant force direction, without demanding an explicit orientation rule.[12] There is a huge interest on decentralized methodologies for swarm robots, especially in recent years. For instance, a recent study suggests a new decentralized navigation controller to steer a group of robots towards a desired goal. The control architecture of this system consists of two loops that one of them is responsible for crowd dynamics whereas the second loop, internal control loop, transforms velocity and acceleration parameters.[13] Besides, in an alternative study, authors propose a new method based on decentralized approach to navigate mobile robot. They claim that the proposed method ensures both collision free path and global network connectivity in large and complex environments.[14] In an interesting study, authors propose a method for formation problem of mobile robots based on a decentralized control law that robots can move a desired geometric configuration from their initial positions.[15] In another recent study, a stochastic approach is proposed to control swarms of independent robots towards a target distribution in a bounded domain.[16] Robots are only allowed to measure a scalar field that is essentially employed to deduce the desired local swarm density. Authors claim that the algorithm is scalable and applicable for large swarm systems. A probabilistic framework designed based on decentralized observable Markov decision processes was presented in ref 17. This framework aims to provide a cooperation between a group of robots to optimize a shared objective in the presence of uncertainty. Other relevant studies can be seen in refs. [18–25].

This paper proposes a decentralized architecture for the goal-based navigation problem of multi-agent systems. The hypothesis, tested in this research, is that "it is possible to develop a decentralized strategy for collective and collaborative behaviours of multi-agent systems for goal-based navigation problem". A novel system is designed and several experiments are carried out so as to test this hypothesis. The goal-based navigation problem for a multi-agent system essentially aims to approximate to the optimum strategy for matching the working environment and the tasks. Accordingly, the goal-based navigation problem for a multi-agent system can be expressed as follows:

$$\pi^* : S \rightarrow B. \tag{1}$$

Here, n robots, $R = \{r_1, r_2, \ldots, r_n\}$, m tasks, $B = \{b_1, b_2, \ldots, b_m\}$, and also each state in the working environment is defined as $s_t \in S$. The optimum strategy for the goal-based navigation problem for a multi-agent system mainly aims to complete all tasks by minimizing total navigation time while preventing collision with obstacles.

$$G = min\,\{g_1, g_2, \ldots, g_n\}. \tag{2}$$

Here g_i = time required for allocation of task b_i.

The flowchart of the overall system is presented in Fig. 1. The architecture is designed based on PFSMs and utilizes a virtual-physics-based approach to assign each robot to a group and to automatically allocate a leader for each group using a decentralized strategy. This is in contrast to the conventional aggregation behaviour of social animals, in that instead of forming a single cluster, agents are grouped according to their physical position relative to each other, and a leader is selected using a completely decentralized strategy.[18] A classification module was designed and integrated into the overall architecture which basically separates obstacles from nearby robots. Leader robots possess image recognition skills that can detect a goal using an appearance-based method and can estimate the robot's relative distance from the goal.[26] In addition, each group uses a pattern formation behaviour that allows them to keep a certain distance between group members during the navigation

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

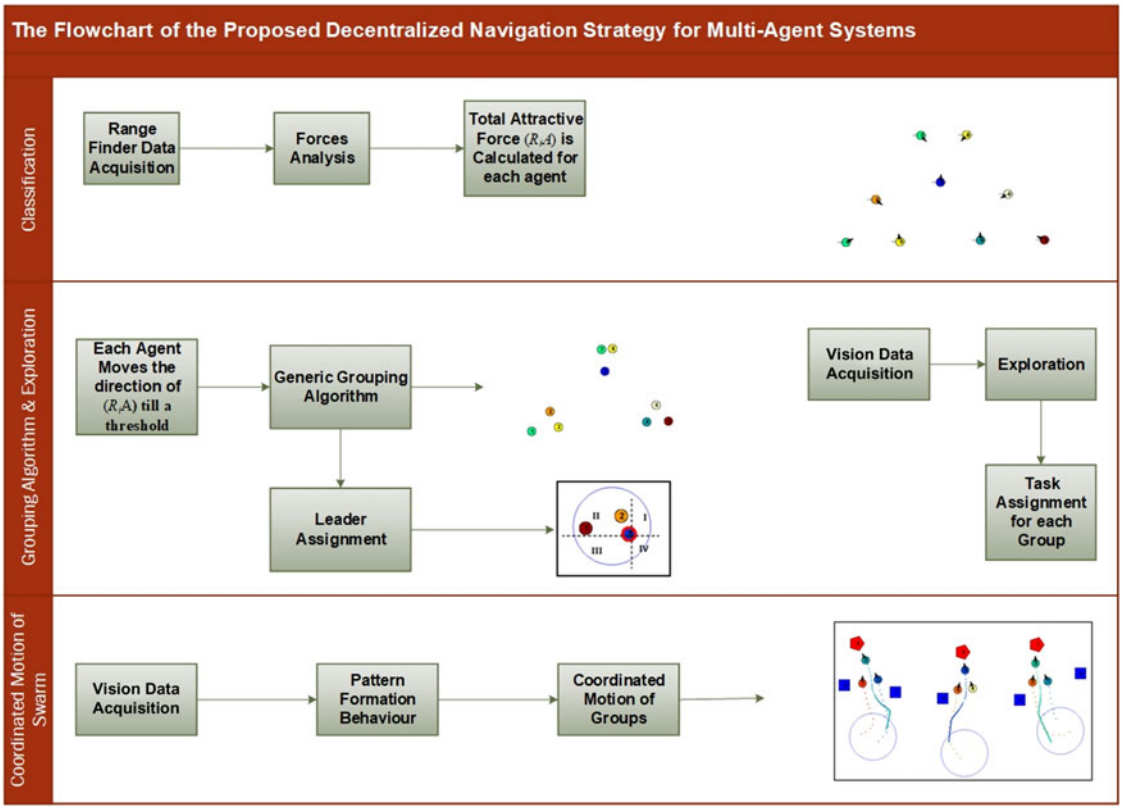

Fig. 1. Flowchart of the overall system.

process. This behaviour allows member robots to effectively navigate with a coordinated motion to a specific goal, following the leader while avoiding obstacles. One of the critical issues within this study is the optimization of force law parameters, which is essential for multi-agent systems to learn to navigate within a complex environment in a coordinated manner. The aforementioned figure essentially illustrates the layers of overall system architecture that the first layer presents the classification behaviour of the randomly located robots. Within this layer, robots perform a force analysis and initialize clustering behaviour, respectively, in a decentralized manner. Once the classification layer is completed, the grouping behaviour is activated and the robots move the direction of the total attractive force so as to form groups. This grouping procedure is completely decentralized and will be detailed in the following sections. This is followed by a new leader assignment procedure without requiring a direct communication. The details of this behaviour and the pseudo code of the clustering algorithms can be seen in Section 2. The second layer also includes an efficient exploration behaviour that divides the exploration space into sub-spaces for each group. Each group then starts exploration within its corresponding sub-space which essentially reduces the overall exploration time. Besides, this behaviour also assigns tasks to each group. The details of this exploration behaviour can be seen in Section 2.4. The final layer of the architecture proposes a pattern formation and coordinated motion behaviours based on a goal-based navigation strategy. According to which, once the formation of the groups is generated, this formation behaviour is reserved during the navigation process. The group is able to move towards the goal using a vision-based approach while avoiding obstacles and keeping the formation pattern in partially cluttered environments. This paper is organized as follows: Section 2 details the proposed collective behaviour architecture and introduces corresponding strategies and methodologies, while Section 3 discusses the implementation of the proposed system and evaluates the overall performance of the systems based on realistic scenarios. Finally, the conclusion is presented in Section 4. Part of this study was presented at the Second International Conference on Mechatronics and Robotics Engineering (ICMRE-2016).[18]

2. Collective Behaviour Architecture

This section presents some essential knowledge related to the proposed collective architecture work, including generic grouping and the leader assignment module, pattern formation, and collective

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

movement behaviours. In addition, a goal-based navigation approach using the exploration module is described. The following sub-sections will address each of the steps involved.

2.1. Force analysis and classification modules

A force analysis module mainly aims to group robots using a decentralized strategy. Accordingly, it is expected that robots are randomly located and have no direct communication. There are two assumptions in this algorithm. The first is that each robot is equipped with a standard range finder, with a 360° field of view. The second is that all robots have the same mass. Within these assumptions, the universal gravitational force (Eq. 3) applied to each robot in the swarm is calculated, in order to estimate the overall attractive force applied to each of them. The law states that a force arises between two agents is proportional to the product of their masses and inversely proportional to the square of the distance between them, as shown in Eq. (3).

$$F = G \times \frac{m_1 \times m_2}{r^2}. \tag{3}$$

Here F represents force, m_1 and m_2 are the masses of each agent, G is a constant value, and r is the distance between the centres of m_1 and m_2.

Also each robot employs the range finder data to calculate the relative position of each surrounding robot. Consequently, ith robot can estimate both the direction and magnitude of the force applied on it by surrounding robots. The distance r parameter and the direction of the applied force θ are calculated as follows:

$$r = \sqrt{(x_{r1} - x_{r2})^2 + (y_{r1} - y_{r2})^2}. \tag{4}$$

$$\theta = arctan2(y_{r1} - y_{c1}, x_{r1} - x_{c1}). \tag{5}$$

Here (x_{r1}, y_{r1}) and (x_{r2}, y_{r2}) illustrate the coordinates of the central points of two robots, respectively. The total attractive force applied to the ith robot (R_iA) in the swarm is defined as follows:

$$R_i A = \sum_{j=1}^{n} \vec{F}_j. \tag{6}$$

Arbitrarily located robots in the environment start to apply force and attract each other based on Eq. (2). Due to these attractive forces, robots start moving in the direction of the resultant force so as to approach each other; each robot remains moving until the applied attractive force exceeds a certain limit. This limit essentially determines the distance between the current robot and the closest robot applying an attractive force. With this force-based behaviour, robots get close to each other to form a group in a decentralized manner without requiring any direct communication. It should be noted again that each robot is influenced by the total attractive force within the limits of its range finder which allows robots to move the direction of the resultant vector to join a group in a reasonable acceleration. As well as, the maximum gravitational force is limited with a predefined parameter in order to avoid unexpected collisions, which essentially allows a robot to join a group in a safe and reliable manner. The details of this approach can be seen in the corresponding author's previous study.[18] However, the main drawback of this approach is that it is unable to separate obstacles from robots. To overcome this drawback, a neural network (NN)-based approach was designed and integrated into this module, which separated robots from obstacles. A range finder returns a point for each unit of angular resolution, which indicates the distance between the robot and any object that the range finder senses. The output of the range finder is first simplified by classifying the field of view into n clusters (C_n) where, for this problem, the range finder has a 360° field of view; each cluster is therefore responsible for a $\frac{360°}{n}$ field of view.

$$D = N(S), \text{ where } D = [o]^T,\ S = [C_1,\ C_2,\ C_3, \ldots,\ C_n]^T, \tag{7}$$

Here "o" refers influence of object, which is the output of the NN and is used to assign the type of sensed object. Besides, the range data cluster obtained from the laser sensor C_n is used as input layer to the proposed NN. Equation (3) essentially refers the mapping function between the object type and the range finder data output "S" based on an NN structure.

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

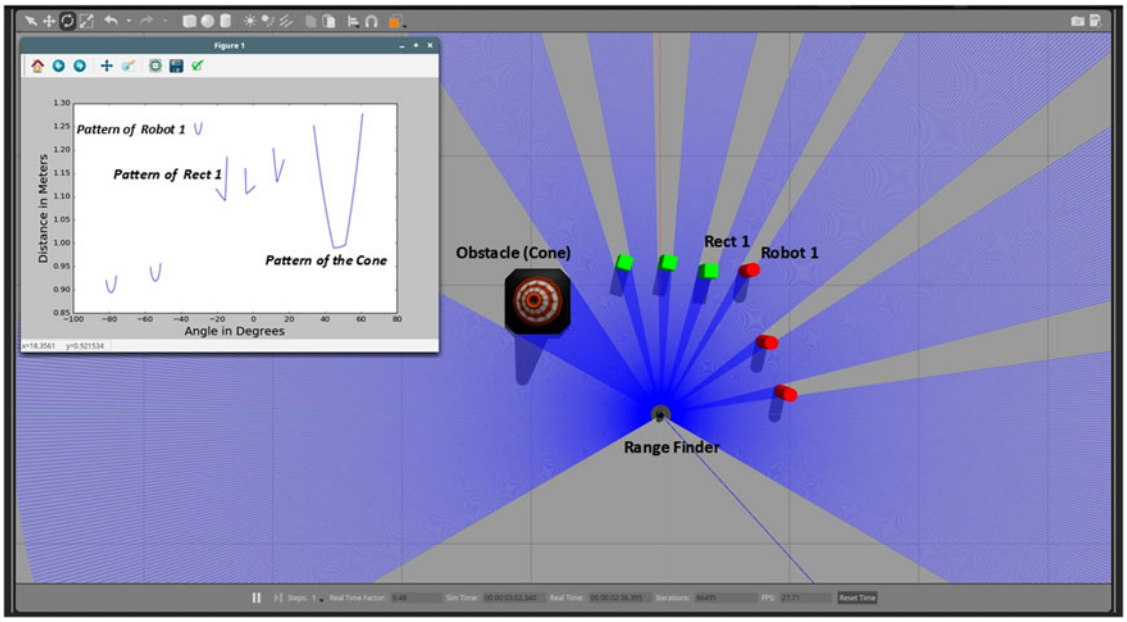

Fig. 2. Training module, output signal and training scenario using Gazebo.

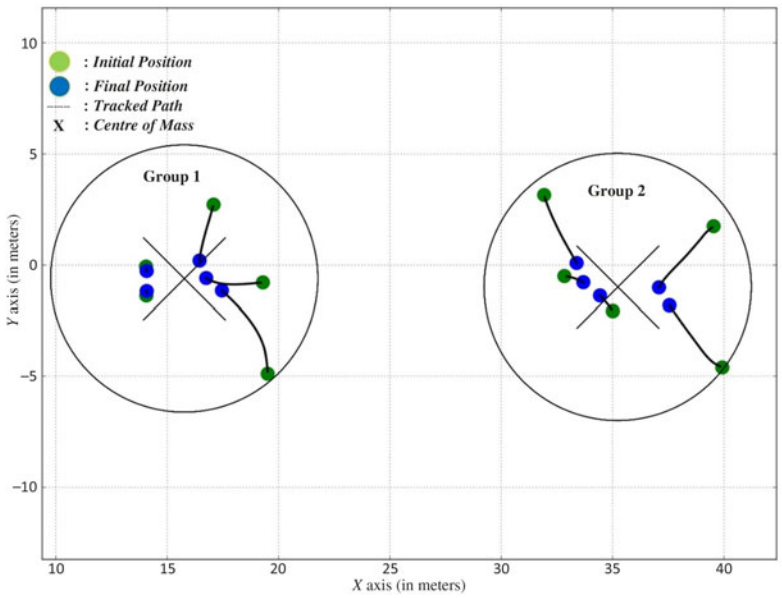

Fig. 3. Force-based clustering behaviour.

A range finder simulator module was designed and added into previously developed robotic simulator software.[27] Figure 2 illustrates an example screenshot, depicting an example training scenario.

The classification module is designed based on identifying range finder responses. As can be seen from the corresponding figure (Fig. 2), when obstacles and robots are represented with different geometrical shapes, different range finder responses are obtained. Following this, a multi-layer feed-forward NN using a backpropagation learning algorithm was employed with these data during the training phase (Fig. 3). This capability allows the robots to separate obstacles from nearby robots during the grouping and leader selection behaviours. The training samples, given in Fig. 3, especially includes three geometrical shapes, a cone (main obstacle used in experiments), a rectangle (alternative obstacle), and a circular shape, referring the robots. Each of those shapes generates a different

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

pattern, as shown in Fig. 3, and the network is trained in order to map the range finder data and the corresponding object using those patterns.

2.2. Grouping algorithm and leader assignment

Due to characteristics of the proposed clustering (grouping) strategy, a number of robots for each group are not previously defined. Instead, each robot moves the direction of the total attractive force until joining a group. For each iteration, each robot is also checked whether it joins a cluster or not by calculating the distance between the robot and its closest neighbour.

Essentially, once a robot gets close enough to any of its surrounding robot it completes the grouping behaviour and is accepted to join a group. Figure 3 illustrates an example scenario that includes two groups. According to the corresponding figure, robots move the direction of the applied total attractive force direction until getting close to one of its surrounding robots. Once this strategy runs for a while, all robots will join a group. However, if a robot cannot be sensed by range finder sensors of other robots, it will stay isolated and will move individually. For the experimental section, a time constant is set for the grouping behaviour that robots are considered to complete within this constant. This constant is estimated by considering maximum range of the range finder and average velocity of robots. The pseudo code of the decentralized clustering algorithm is also given below.

Decentralized Grouping algorithm:

Requires Randomly located robots, N robot numbers, R_i is the ith robot and R_k is the kth robot
Ensures Decentralized Clustering of robots
Procedure Clustering Behaviour
  While(Clustering time limit is not exceeded)
    for $i = 1: N$
    initialize R_iF;
      for $k = 1{:}N$
        if $i \neq k$ then
          if R_k is visible to R_i
            Calculate Force F between R_i and R_k
              $R_iF = R_iF + F$
          endif
        endif
      endfor
    endfor
    for $i = 1: N$
      Calculate the distance between R_i and its closest neighbour R_k
      if min_dist $<$ clustering threshold
        R_i has finished clustering
      else
        R_i moves the attractive force direction (R_iF)
      endif
    endfor
  EndWhile
EndProcedure

As aforementioned, each robot draws close to nearby robots due to the resultant attractive force vector. Essentially, each of them keeps moving until the resultant-applied attractive force exceeds a certain threshold value. A unique, decentralized leader assignment procedure is applied to these groups of robots. The details of this study are given in the authors' previous work.[18] The proposed decentralized algorithm states that each robot first calculates the relative position of each of the nearby robots via a force-based scheme (see Section 2.1). After this, each robot considers itself as the origin of its local coordinate system, without any outer interference, and detects other robots' locations in order to estimate in which quadrants they are positioned in the coordinate system.[18] This process allows a calculation of the positions of nearby robots and the obstacles near each robot.

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

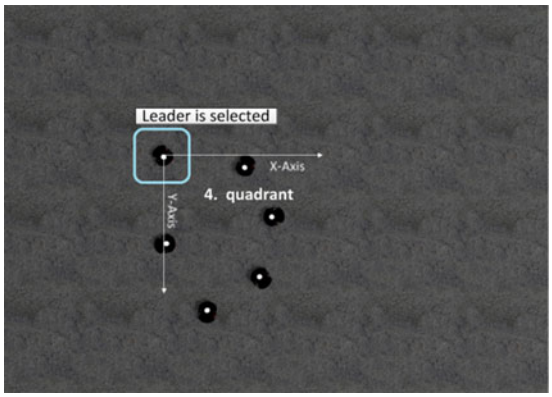

Fig. 4. Example scenario; the leader is selected (using Gazebo).

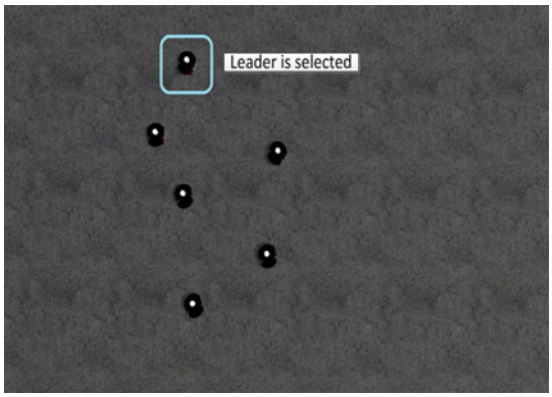

Fig. 5. Example scenario; the group moves to generate a formation pattern (using Gazebo).

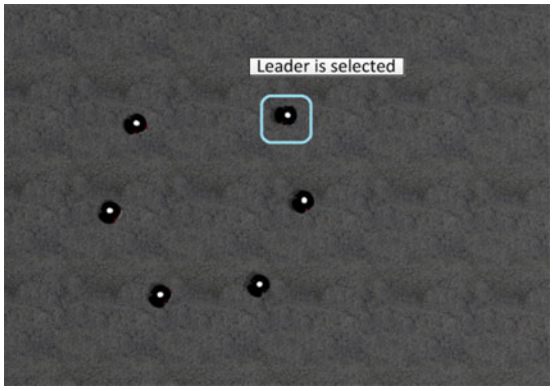

Fig. 6. Example scenario; a pattern has almost been formed by the group (using Gazebo).

A robot is only assigned as the leader of its group when it detects the maximum number of member robots located in the same quadrant. This procedure provides flexibility for the group and separates the leader from the rest of its group. The only exception to this algorithm occurs when all the robots are located along the same line. In order to prevent this situation, an assumption is made that robots cannot initially be located along the same line. An example scenario regarding grouping and leader selection is illustrated in Figs. 4–6. Each robot registered to a group runs the developed algorithm to assign a leader for the group, without requiring any direct communication or using a central decision mechanism, as illustrated in Fig. 4. The decision mechanism of the algorithm for the given scenario reveals that only the robot shown in the upper left-hand part of Fig. 4 is able to sense other robots in the same quadrant (fourth quadrant).

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

Each robot analyses the range finder data individually in order to reveal which is the leader of the group, and the locations of the leader and other group members. An early version of this algorithm was designed and tested for groups including at most three robots.[18] In the current study, the grouping size is increased gradually in order to reveal the limits and overall performance of the algorithm. The results show that the algorithm can handle at most nine robots in each group in a decentralized manner. There is rarely more than one robot fulfilling the requirements to be registered as the leader, especially as the maximum group size is increased. In these circumstances, one of these leaders is selected as the only leader of the group in a stochastic manner (randomly). Figures 5 and 6 illustrate an example of how a group generates a pattern formation; the details are discussed in the next section.

2.3. Pattern formation algorithm

Pattern formation is a critical and challenging research field, especially in areas related to swarm robotics and multi-agent systems. This field essentially aims to form a complex system by evaluating the interaction and cooperation between robots.[27] A strong pattern formation algorithm is designed and implemented in this study, facilitating the overall success of the system. Part of this section also appeared in the authors' previous study.[13] Pattern formation can be considered as a self-organization behaviour that utilizes local interaction between the members of the system instead of external sources. In the proposed system, the pattern formation algorithm is executed after the grouping and leader selection steps are completed; the algorithm allows the swarm to perform a circle formation behaviour autonomously, regardless of the size of the swarm. Based on this algorithm, the leader robot first estimates the location of all member robots using the range finder data, and then calculates the position of the centre of gravity (COG) of the group. Subsequently, the leader generates an artificial vector $\overline{CL}$ from its position to the COG in order to estimate the exact position of the member robots (R_1–R_n) relative to the COG. This, in essence, allows the leader to generate a vector for each member robot with respect to the COG. As soon as the artificial vectors ($\overline{CR1}$, $\overline{CR2}$, . . . , $\overline{CRN}$) of the member robots have been obtained, the leader sorts the member robots according to the angle they make with the COG, using the $\overline{CL}$ vector as the start point and moving in a counter-clockwise direction. Afterwards, the leader conveys the critical parameters to each member, including the *position of the COG*, *formation role number (FRN), formation radius*, which determines the size of the circular pattern, and *number of robots (n)*. This allows member robots to move towards a potential position and to generate a circular pattern according to Eqs. (4) and (5):

$$pos_x = c_x + f_r * \cos\left(\left(\frac{2\pi}{n}\right) \times frn + c_{diff}\right). \tag{8}$$

$$pos_y = c_y + f_r * \cos\left(\left(\frac{2\pi}{n}\right) \times frn + c_{diff}\right). \tag{9}$$

Here n the number of robots, f_{rn} formation role number, c position of the COG, $f_{r:}$ formation radius value, *and* c_{diff} difference between the robot and the x-axis. Once pos_x and pos_y values are calculated by each robot using the abovementioned equations, each robot moves the corresponding position with respect to the COG point based on the potential field algorithm. This algorithm allows each robot to reach its formation position while preventing collision. The potential filed method is an efficient and powerful approach for local navigation problem. According to which, the attractive potential is calculated for the goal position while a repulsive potential is also calculated for each of the obstacles located in the environment. Overall, the scalar potential field value "P" can be defined as follows:

$$P = P_{att} + P_{rep}. \tag{10}$$

Here, P_{att} and P_{rep} indicate attractive and repulsive forces, respectively.

The vector field of forces F (q) is given as follows:

$$F_q = -\nabla P_{att} + \nabla P_{rep}. \tag{11}$$

The details of the potential field algorithm can be seen in refs. [13, 14]. The proposed system essentially has a local and global obstacle avoidance strategy based on the potential field algorithm and formation behaviour. As previously mentioned, each robot possesses the conventional potential field algorithm that once any of the group member encounters an obstacle, which is a repulsive force,

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

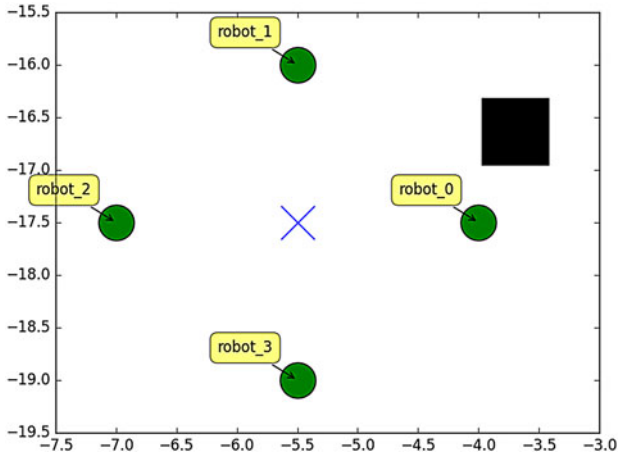

Fig. 7. Robot 0 encounters an obstacle group.

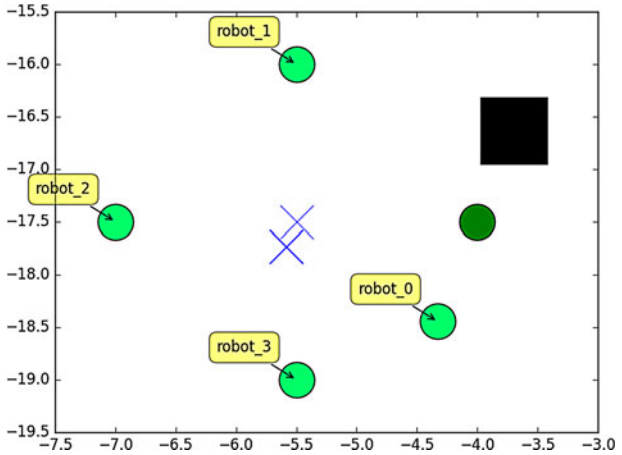

Fig. 8. Robot 0 avoids the obstacle and COG is shifted.

it performs a manoeuvre to defeat the obstacle. This local avoidance behaviour also triggers the global obstacle avoidance manoeuvre. For instance, once a robot performs an avoidance manoeuvre, it may be lead from its formation role position to any other position. This manoeuvre both breaks the formation of the swarm and shifts the COG point. Due to this change, other group members must be relocated in an adaptive manner so as to keep the formation. This adaptive formation behaviour may enable the robots to navigate along the obstacle-free path. An example scenario including this avoidance behaviour is illustrated in Figs. 7–9. Especially, when the leader robot encounters an obstacle and prevents collision by employing its local obstacle avoidance behaviour, member robots are relocated adaptively that probably decreases the probability of those robots to encounter the same obstacle or nearby obstacles. The pattern formation algorithm is designed completely dynamically, so that even if a robot is forced to make a sharp manoeuvre due to the sudden appearance of an unexpected obstacle, which will probably disrupt the shape of the formation pattern, the formation pattern is reorganized and regenerates the circular formation. This is because the status of the pattern is checked at short intervals by the leader.

2.4. Exploration module

The exploration of a large space within a reasonable period of time is a challenging problem for mobile robot-based navigation systems, including swarm robots and multi-agent systems, especially when the goals are unknown to the swarm and vision is used as the main sensor for detecting goals. To reduce the overall exploration time, a simple but efficient exploration algorithm is designed for the proposed system. The algorithm basically divides the exploration space into sub-spaces for

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

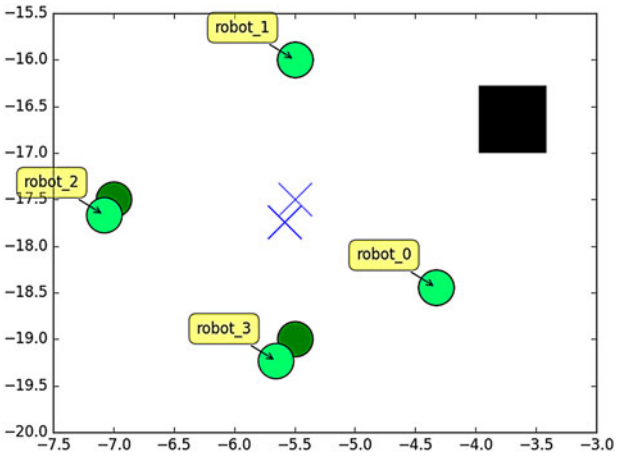

Fig. 9. Other robots are repositioned according to the new COG point.

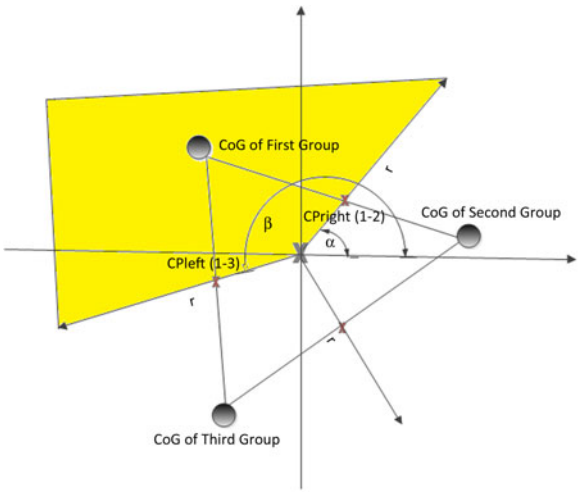

Fig. 10. Exploration module example for three groups, illustrating the first group search space (yellow area).

each group. Each group then starts exploration within its corresponding sub-space. Thus, a collaborative behaviour is provided by the swarm groups both to detect goals in efficient manner and also to complete tasks in a shorter time span. Each group communicates with the others via the leaders. The corresponding algorithm is designed for three or more groups of swarm robots. Based on the exploration algorithm, the COG point of each group is first calculated. Next, an artificial vector is generated from the position of the COG to the leader robot for each group. Following this, each leader learns the ID of the left- and right-hand neighbouring groups. Each robot calculates the partial COG points considering its left- and right-hand neighbours. Equation (6) illustrates this partial COG calculation for the group and its left-hand neighbour; the same equation is applied to calculate the right-hand neighbour, as shown in Fig. 10.

$$partial_{left^x} = \frac{\left(COG_{left^x} + COG_x\right)}{2}, partial_{left^y} = \frac{\left(COG_{left^y} + COG_y\right)}{2}. \tag{12}$$

Here, COG_{left^x} is x-coordinate of the left neighbour, and COG_{left^y} is y-coordinate of the left neighbour.

The leader (the active group) draws two artificial vectors from the positions of the COG of all groups to the calculated partial COG points, namely $\overrightarrow{CP_r}$ and $\overrightarrow{CP_l}$. Following this, α and β angles are calculated, indicating the angles of the $\overrightarrow{CP_r}$ and $\overrightarrow{CP_l}$ vectors, respectively, with the x-axis (see Fig. 6). Figure 6 illustrates an example scenario in which there are three individual swarm groups; the position of the COG of each group is illustrated with a circle. The large cross ("X") illustrates

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

the position of the COG obtained from all groups, while the small crosses ("×") illustrate the partial COG points obtained from the neighbouring groups. The corresponding figure also illustrates the search space for the first group (yellow area for the first robot group). Consequently, for each group a random search position is calculated based on Eq. (13):

$$pos_x = COG_x + \cos(rand\,(\alpha, \beta)\; x\; (rand\,(0, r)), \quad pos_y = COG_y + \sin(rand\,(\alpha, \beta)\; x\; (rand\,(0, r)). \tag{13}$$

Here, COG_x refers x-coordinate of the position of the COG, and COG_y refers y-coordinate of the position of the COG. As well as, α is the angle of $\overrightarrow{CP_r}$ with respect to the x-axis, β is the angle of $\overrightarrow{CP_l}$ with respect to the x-axis, and r refers the maximum allowed search radius.

2.5. Vision for goal detection

An appearance-based approach is employed to detect the unknown goals in this study. The vision module of the system detects the goal using an appearance-based method and estimates the robot's relative distance from and orientation to the goal. The vision-based goal detection algorithm has previously been discussed in ref. [26], and the pseudo-code is shown below.

Vision-based goal detection algorithm:

Requires Red Green Blue (RGB) image, scale of the object, pixels numbers and focal length
Ensures Distance and orientation to the goal
Procedure Detect the goal
    Load image and initialize parameters
    Convert the image from RGB to Hue, Saturation, Value (HSV) color space
    Pre-processing for image enhancement (smoothing)
    Apply morphological operators to refine image (dilate and erode operators)
    while_completed
        Read image pixel by pixel
    Detect and extract contours from the image
        Select the largest contour
        Calculate the rectangle with the largest contour
end_while
Calculate the distance, d, (Eq. 14) and the orientation angle, θ, (Eq. 15)
EndProcedure

$$Distance\ to\ goal = \frac{focal\ \ length\,(mm)\ *\ image\ \ height\,(\text{pixels})}{object\ \ height\,(pixels)\ *\ sensor\ \ height\,(\text{mm})}. \tag{14}$$

$$\theta = a\tan\left(\frac{y_t - y_c}{x_t - x_c}\right). \tag{15}$$

Here, y_t is the height of the contour centre in pixels, y_c is initial height of the camera centre in pixels, x_t is width of the contour centre in pixels, and x_c is the initial width of the camera centre in pixels.

The algorithm is initialized by converting the image from RGB color space to HSV. This is because HSV images can tolerate variations in lighting and noise better than RGB images. A Gaussian-based smoothing filter is also applied to reduce noise effects, and a series of morphological operators are applied afterwards in order to improve the overall image quality (dilate and erode). The gradients of the enhanced image are then calculated, which is followed by employing a connected component labelling algorithm (CCL).[14] In essence, the CCL algorithm groups image pixels according to their pixel connectivity and is utilized to extract contours from binary images. It is considered that the contour with the largest area must be related to the goal. The approximate height and width parameters are calculated from the rectangular area surrounding the goal.

Once the goal is detected, Eqs. (8) and (9) are then employed to estimate both the distance to the goal and the orientation of the robot with respect to the goal. Figure 11 illustrates an example scenario, illustrating the monocular-based goal detection system using Gazebo.

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

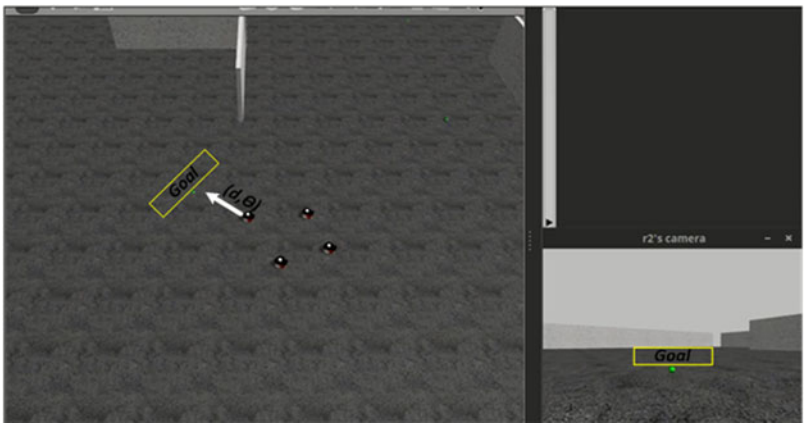

Fig. 11. A scenario illustrating the vision-based goal detection system using Gazebo.

3. Experimental Section

This experimental section has two parts; the first addresses the implementation details of the proposed navigation system and corresponding evaluation parameters, while the second part illustrates the experimental results.

3.1. Implementation of the system

The proposed navigation framework designed for multi-agent systems was extensively tested in a comprehensive and realistic simulation environment in order to verify its applicability. The system was implemented using the Gazebo 3D simulation environment supported by the ROS platform under Linux Ubuntu (14.04) (see Fig. 8). The overall system was installed on a machine with a 2.2-GHz Intel Core i7 processor with 8-GB RAM. Several different evaluation parameters were utilized either to assess the performance of overall system or the performance success of individual behaviours, such as pattern formation and exploration modules. The following evaluation parameters are defined and used to verify the performance of the proposed navigation system.

Total navigation time (t_s): This parameter shows the navigation duration in seconds. Lower values indicate faster navigation.

Total travel distance (d_m): This parameter indicates the distance travelled by a robot from its starting position to complete the task. Lower values indicate more optimized solutions.

Change in angular velocity ($\Delta\Omega$): This parameter indicates the average change in angular velocity (in deg/s^2) during the robot's navigation. Lower values show a more reliable angular velocity and smoother movement of the robot, which is shown as follows:

$$\triangle\Omega = \frac{1}{n-1}\sum_{i=2}^{n}\frac{|\Omega(i)-\Omega(i-1)|}{l_c(i-1)}. \tag{16}$$

Here, $\Omega(i)$ *is* the angular velocity at the ith cycle and $l_c(i-1)$ and the length of the decision cycle.

Precision (p_v): This indicates the fraction of properly retrieved instances, including the total number of true and false matches obtained from an experiment, which is given as follows:

$$p_v = \frac{correct\ matches}{correct\ matches + false\ match}. \tag{17}$$

3.2. Experimental results

As mentioned above, due to the modular structure of the proposed navigation framework, the essential modules entailing different behaviours were tested. Four different test scenarios were designed to evaluate the performance of the system in the experiments, each of which was conceived with an increased level of complexity. These scenarios are described below:

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

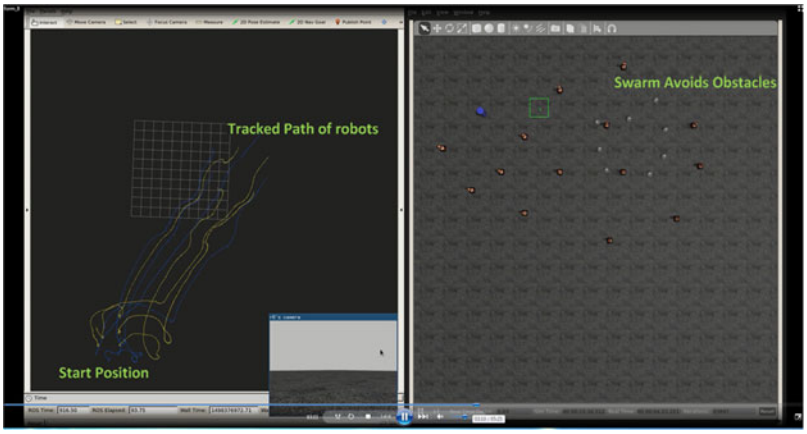

Fig. 12. Pattern formation behaviour of robots avoiding obstacles: (a) tracked paths of all robots; (b) simulation environment for S_1.

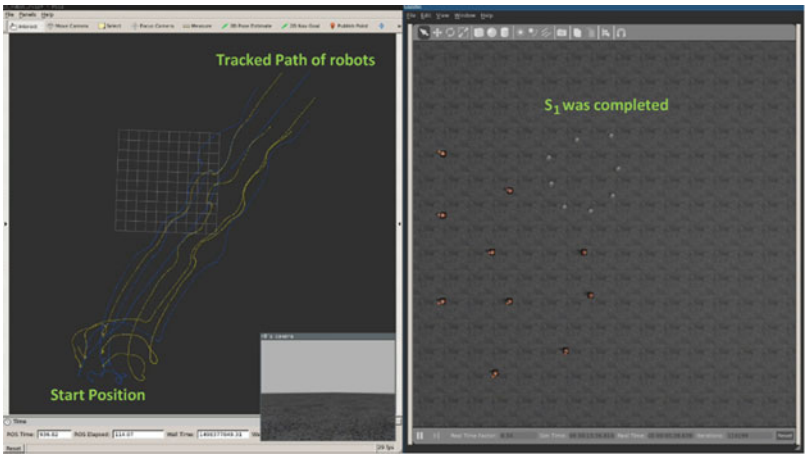

Fig. 13. Pattern formation behaviour of robots completing the goal: (a) tracked paths of all robots; (b) simulation environment for S_1.

3.2.1. Scenario 1 (S1). This scenario illustrates the experimental results of pattern formation behaviour with respect to varying numbers of robot. An example of a pattern formation sub-scenario with eight robots is shown in Figs. 12 and 13. The results show that the proposed algorithm achieves good formation manoeuvres, even in complex environments. The most important point for this scenario is that as the number of robots increases, the formation radius value should be increased to give the expected better formation behaviour. For the given simulation environment and configurations, the maximum allowed number of robots in a group is eight. Part of this section was also presented in the authors' previous work.[14] Figures 12 and 13 illustrate a sub-scenario for circular pattern formation behaviour of a swarm with eight robots. Figure 12 illustrates an instance from a scenario in which robots are both trying to defeat obstacles and to stay in a circular formation pattern; Fig. 13 indicates the final position of the swarm and the path followed during the navigation process. The average performance measurement for robots using the aforementioned evaluation parameters for S_1 is illustrated in Table I. Each sub-scenario test was repeated five times due to the randomized parameters of the simulation environment, and the average was determined for each robot for each performance parameter.

The results indicate the proportional relationship between the swarm size and the behaviours of swarm robots based on the parameters of distance travelled (d_m), navigation time (t_s), and change in angular velocity $(\Delta\Omega)$. When the number of robots in a group increases, the performance of each robot generally decreases gradually. In addition, no collisions were recorded during the experiments.

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

Table I. Performance measures for S_1.

No. of robots	t_s (s)	d_m (m)	$\Delta\Omega$ (deg/s^2)	Collision
4	102.5	47.45	15.87	No
5	103.6	47.74	24.57	No
6	103.7	49.51	18.28	No
7	100.3	47.23	22.34	No
8	108.3	49.31	18.85	No

Table II. Performance measures for S_2.

G_{no}: 3	t_s (s) "*Total*"	t_s (s) "*Explore*"	d_m (m) "*Total*"	$\Delta\Omega$ (deg/s^2)
R_{no}: 4	28.01	12.99	12.7	44.69
R_{no}: 4	16.64	0.68	9.1	29.22
R_{no}: 3	26.4	14.32	11.87	22.45

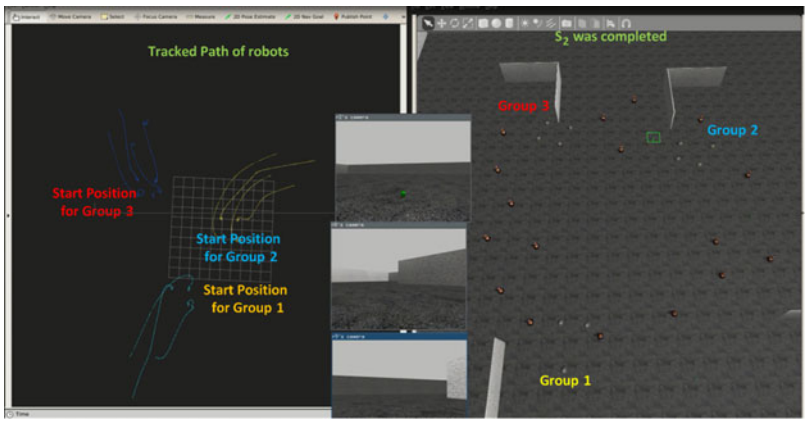

Fig. 14. Exploration behaviour of robots searching for and navigating to the goal: (a) tracked paths of all robots; (b) simulation environment for S_2.

3.2.2. Scenario 2 (S2). This scenario demonstrates the exploration behaviour of the robots and the navigation process of the robots. Figure 14 illustrates a sub-scenario where three groups of robots with varying numbers of members start exploring the goal using the goal detection algorithm (see Section 2.5). Each leader detects the goal using a simple appearance-based approach, while member robots, on the other hand, move in the direction of the leader to retain their formation.

Due to the complexity and vagueness of this scenario, member robots especially need to make sharp manoeuvres in order to retain their formation, which results in changes in the angular velocity ($\triangle\Omega$) parameter. Table II illustrates the performance measures of this sub-scenario, which is shown in Fig. 14. The term $\boldsymbol{G_{no}}$ refers to the total number of groups, while $\boldsymbol{R_{no}}$ denotes the number of robots in each corresponding group. Unlike the previous scenario, exploration time (meaning the overall time consumed in the goal-seeking problem) is also considered as a parameter. When the goal can be seen by the leader of a group, or is close to it, the total time consumed by the exploration behaviour becomes trivial, as shown in the third column of Table III.

3.2.3. Scenario 3 (S3). The third scenario involves a complex environment for swarm robots that includes a long corridor and a significant number of obstacles obscuring the robots' path, as shown in Fig. 15. The first group successfully retains the circular formation while passing the corridor safely and finally attains the goal. This group successfully passes along the corridor in a smooth manner. The width of corridor is set to 8 m in this scenario, and the formation radius is also set to a larger value of 1.8 m. This is because during the pattern formation behaviour, robots may consider other members as obstacles, which results in unexpected manoeuvres and robots ultimately departing from

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

Table III. Performance measures for S_3.

S_3	Conf.*	t_s (s)	d_m (m)	$\Delta\Omega$ (deg/s^2)
S_{31}	G_{no}: 1 R_{no}: 4	71.06	33.21	17.18
	G_{no}: 2 R_{no}: 4	68.12	32.7	13.75
	G_{no}: 3 R_{no}: 3	71.48	33.39	10.37
S_{32}	G_{no}: 1 R_{no}: 5	71.2	33.59	22.4
	G_{no}: 2 R_{no}: 4	68.08	31.7	12.9
	G_{no}: 3 R_{no}: 3	71.44	33.33	9.96
S_{33}	G_{no}: 1 R_{no}: 6	71.24	34.03	29.6
	G_{no}: 2 R_{no}: 4	68.37	31.67	13.07
	G_{no}: 3 R_{no}: 3	71.52	33.09	9.88

*Conf. = Configuration

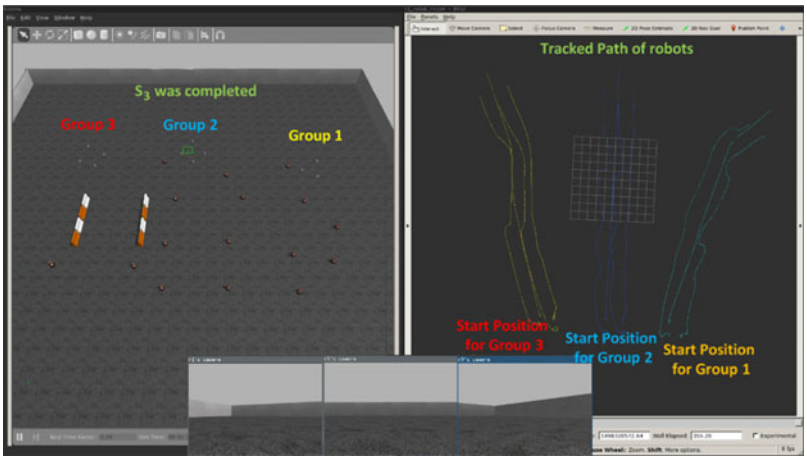

Fig. 15. Navigation behaviour of robots for a complex scenario: (a) tracked paths of all robots; (b) simulation environment for S_3.

the group. It has been proved that the proposed architecture is able to steer robots safely towards unknown goals in a collaborative and decentralized manner. Table III illustrates the average performance evaluation results of swarm robots for S_3, involving three sub-scenarios. The term S_{31} refers to the first sub-scenario, G_{no} denotes the group number, and R_{no} represents the number of robots in the corresponding group. Depending on the increase in robot numbers, the swarm architectures show a minor increase in the parameters of total distance travelled and navigation time. The first group passing along the corridor aims to retain the formation and avoid collisions along the corridor. This behaviour causes more manoeuvres in order to retain the formation safely but also results in more changes in angular velocity, which means that the robot group follows a less smooth path.

3.2.4. Scenario 4 (S4). The fourth scenario also involves a complex environment including a significant number of obstacles obscuring the robots' path, as illustrated in Fig. 16. This scenario essentially illustrates a simple maze that the robot groups are aiming to pass the maze in a collaborative and decentralized manner. The first group starts its navigation, successfully defeats obstacles, and finally reaches its goal position. The second and the third groups move towards each other while trying to attain their goals. Both groups first pass each other in a smooth manner by avoiding any collision to each other. Afterwards, both handle the obstacles located into their paths successfully by retaining the triangular formation. In conclusion, all groups achieve to complete their tasks by avoiding obstacles and keeping the triangular formation pattern. Table IV demonstrates the average performance evaluation results of swarm robots for S_4. While the term, G_{no} denotes the group number, R_{no} represents the number of robots in the corresponding group. The first group achieves to pass the maze in a reasonable amount of time while keeping the formation and avoiding obstacles. However, the second and third groups try to attain their goals which are located at farther locations. Owing to this

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

Table IV. Performance measures for S_4.

G_{no}: 3	t_s (s) *"Total"*	d_m (m) *"Total"*	$\Delta\Omega$ (deg/s^2)
R_{no}: 3	131	58.9	22.69
R_{no}: 3	184	81.2	59.22
R_{no}: 3	223	102.7	67.45

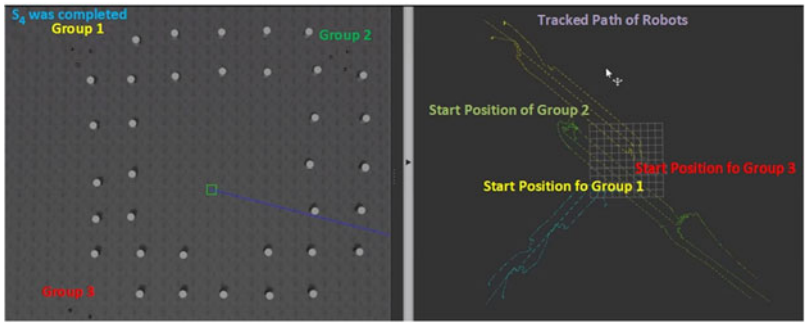

Fig. 16. Navigation behaviour of robots for a complex scenario: (a) tracked paths of all robots; (b) simulation environment for S_4.

distance, both groups reach their goal positions by travelling more distance with a greater change in their angular velocity as expected and shown in Table IV. This scenario illustrates both the local and global obstacle avoidance skills of the proposed system. According to which, a robot can both individually defeat obstacles and also group's COG-based global obstacle avoidance behaviour allows the swarm to overcome obstacles by performing smoother manoeuvre even in complex and partially cluttered environments.

4. Conclusions

This study introduces a new framework for the collective and collaborative behaviour of multi-agent systems in particularly cluttered environments. This framework is mainly designed based on a novel decentralized strategy, relying on vision as the primary sensor for goal detection, with range finder sensors for grouping, pattern formation, and obstacle avoidance behaviours. The grouping behaviour is designed based on a force-based strategy, allowing robots to get close to each other and to select an appropriate leader to steer the group towards specific locations. Once the grouping is completed, each group generates a formation pattern, which is completely dynamic and allows robots to defeat obstacles and reorganize their circular formation pattern quickly. The framework also proposes a global goal-based exploration algorithm that divides the exploration space dynamically into sub-spaces; this aims to prevent collisions between groups of robots during exploration behaviour and allows robots to detect goals in a shorter time interval. A series of comprehensive experiments, including different configurations of robots and challenging environments, were conducted. The results verified the efficiency and reliability of the proposed framework. Future development will focus on integrating the proposed framework using low-cost physical robots in order to assess the performance and robustness of the proposed method in real physical operation conditions.

Acknowledgments

This work is supported by the Scientific and Technological Research Council of Turkey (Tubitak Project No. 114E648).

References

1. M. Brambilla, E. Ferrante, M. Birattari and M. Dorigo, "Swarm robotics: A review from the swarm engineering," *Swarm Intell.* **7**(1), 1–41 (2013).
2. M. Chamanbaz, D. Mateo, B. M. Zoss, G. Tokic, E. Wilhelm, R. Bouffanais and D. K. P. Yue, "Swarm-enabling technology for multi-robot systems," *Front. Rob. AI* **4**, 12 (2017).

https://doi.org/10.1017/S0263574718001261 Published online by Cambridge University Press

3. P. Flocchini, G. Prencipe, S. Nicola and P. Widmayer, "Arbitrary pattern formation by asynchronous, anonymous, oblivious robots," *Theor. Comput. Sci.* **407**(1–3), 412–447 (2008).
4. S. Nouyan, A. Campo and M. Dorigo, "Path formation in a robot swarm: self-organized strategies to find your way home," *Swarm Intell*. **2**(1), 1–23 (2008).
5. O. Soysal, E. Bahçeci and E. Şahin, "Aggregation in swarm robotic systems: Evolution," *Turk. J. Electr. Eng*. **15**(2), 199–225 (2005).
6. O. Soysal and E. Bahçeci, "Probabilistic aggregation strategies in swarm robotic systems," *Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005*, Pasadena, CA, USA (2005) pp. 325–332.
7. J. C. Barca and A. Sekercioglu, "Swarm robotics reviewed," *Robotica* **31**(3), 345–359 (2012).
8. O. Soysal and E. Şahin, "A Macroscopic Model for Self-organized Aggregation in Swarm Robotic Systems," **In**: Swarm Robotics (Springer, Berlin, Heidelberg, 2007) pp. 27–42.
9. T. Balch and M. Hybinette, "Social Potentials for Scalable Multi-robot Formations," *IEEE International Conference on Robotics and Automation, 2000. Proceedings. ICRA'00* (IEEE Press, Piscataway, 2000) pp. 73–80.
10. A. E. Turgut, H. Çelikkanat, H. Gökçe and Ş. Erol, "Self-organized flocking inmobile robot swarms," *Swarm Intell*. **2**(2), 97–120 (2008).
11. H. Çelikkanat and Ş. Erol, "Steering self-organized robot flocks through externally guided individuals," *Neural Comput. Appl.* **19**(6), 849–865 (2010).
12. E. Ferrante, A. E. Turgut, C. Huepe, A. Stranieri, C. Pinciroli and M. Dorigo, "Self-organized flocking with a mobile robot swarm: a novel motion control method," *Adapt. Behav*. **20**(6), 460–477 (2012).
13. M. S. Guzel, E. C. Gezer, V. B. Ajabshir and E. Bostancı, "An Adaptive Pattern Formation Approach for Swarm Robots," *2017 4th International Conference on Electrical and Electronic Engineering (ICEEE)* (Ankara University, Ankara, TR, 2017) pp. 194–198.
14. C. Ton, Z. Kan and S. S. Mehta, "Obstacle avoidance control of a human-in-the-loop mobile robot system using harmonic potential fields," *Robotica* **36**(4), 463–483 (2018).
15. A. Rodriguez-Angeles and C. L. Vazquez, "Bio-inspired decentralized autonomous robot mobile navigation control for multi agent systems," *Kybernetika* **54**(1), 135–154 (2018).
16. Y. Z. Wang, D. W. Wang and E. Mihankhah, "Navigation of Multiple Robots in Unknown Environments Using a New Decentralized Navigation Function," *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV),* Phuket, Thailand (IEEE Press, 2016).
17. V. Nazarzehi and A. V. Savkin, "Decentralized Navigation of Nonholonomic Robots for 3D Formation Building," *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, Bali, Indonesia (IEEE Press, 2014) pp. 2133–2137.
18. M. Guzel and H. Kayakökü, "A Collective Behavior Framework for Multi-agent Systems," *Mechatronics and Robotics Engineering for Advanced and Intelligent Manufacturing. Lecture Notes in Mechanical Engineering* (Springer, Cham, 2016) pp. 61–71.
19. A. Stranieri, E. Ferrante, A. E. Turgut, V. Trianni, C. Pinciroli, M. Birattari and M. Dorigo, "Self-organized Flocking with a Heterogeneous Mobile Robot Swarm," *Advances in Artificial Life, ECAL* (MIT Press, Cambridge, MA, 2011) pp. 789–796.
20. H. Li, C. Feng, H. Ehrhard, Y. Shen, B. Cobos, F. Zhang, K. Elamvazhuthi, S. Berman, M. Haberland and A. L. Bertozzi, "Decentralized Stochastic Control of Robotic Swarm Density: Theory, Simulation, and Experiment," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada (IEEE Press, 2017) pp. 4341–4347.
21. C. Amato, G. Konidaris, G. Cruz, C. A. Maynor, J. P. How and L. P. Kaelbling, "Planning for Decentralized Control of Multiple Robots under Uncertainty," *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, USA (IEEE Press, 2015) pp. 1241–1248.
22. R. Maeda, T. Endo and F. Matsuno, "Decentralized navigation for heterogeneous swarm robots with limited field of view," *IEEE Robot. Autom. Lett.* **2**(2), 904–911 (2017).
23. B. Xin, G. Q. Gao, Y. L. Ding, Y.G. Zhu and H. Fang, "Distributed Multi-robot Motion Planning for Cooperative Multi-area Coverage," *2017 13th IEEE International Conference on Control & Automation (ICCA)*, Ohrid, Macedonia, (2017) pp. 361–366.
24. G. Roussos and K. J. Kyriakopoulos, "Decentralized and Prioritized Navigation and Collision Avoidance for Multiple Mobile Robots," **In**: *Distributed Autonomous Robotic Systems. Springer Tracts in Advanced Robotics* (A. Martinoli et al., eds.), vol. **83** (Springer, Berlin, Heidelberg, 2013).
25. A. C. Jiménez, V. García-Díaz and S. Bolaños, "A decentralized framework for multi-agent robotic systems," *Sensors* **18**(2), 417 (2018).
26. P. Nattharith and M. S. Guzel, "Machine vision and fuzzy logic-based navigation control of a goal-oriented mobile robot," *Adapt. Behav*. **24**(3), 168–180 (2016).
27. M. Kasuya, N. Ito, N. Inuzuka and K. Wada, "A pattern formation algorithm for a set of autonomous distributed robots with agreement on orientation along one axis," *Syst. Comput. Jpn.* **37**(10), 747–757 (2006).