Auton Robot (2010) 29: 331–355
DOI 10.1007/s10514-010-9198-8

Attractor dynamics approach to formation control: theory and application

Sérgio Monteiro · Estela Bicho

Received: 29 October 2008 / Accepted: 15 June 2010 / Published online: 13 July 2010

Abstract In this paper we show how *non-linear attractor dynamics* can be used as a framework to control teams of autonomous mobile robots that should navigate according to a predefined geometric formation. The environment does not need to be known a priori and may change over time. Implicit to the control architecture are some important features such as establishing and moving the formation, split and join of formations (when necessary to avoid obstacles). Formations are defined by a *formation matrix*. By manipulating this *formation matrix* it is also possible to switch formations at run time. Examples of simulation results and implementations with real robots (teams of Khepera robots and medium size mobile robots), demonstrate formation switch, static and dynamic obstacle avoidance and split and join formations without the need for any explicit coordination scheme. Robustness against environmental perturbations is intrinsically achieved because the behaviour of each robot is generated as a time series of asymptotically stable states, which contribute to the asymptotic stability of the overall control system.

Keywords Formation control · Multi-robot systems · Attractor dynamics

1 Introduction

When we have a set of moving robots that should travel according to a desired pattern (all in the same places with relation to each other), we say that we are in the presence of a formation control problem (Arai et al. 2002). Typically, each robot should be able to keep constant the distance that separates it from its neighbours, or the distance and direction to only one neighbour.

There are many tasks that can benefit from the use of a team of robots in formation. Some of the reported in the literature are: payload transportation (Johnson and Bay 1995), capturing/enclosing invaders (Yamaguchi 1999; Hsieh et al. 2008), satellite cluster formation (Bauer et al. 1997), spacecraft formation (Ren and Beard 2002), environment exploration/reconnaissance (Balch and Arkin 1998; Honary et al. 2009), agricultural and construction applications (Hao and Agrawal 2005; Kwok et al. 2007) and several marine applications (Kalantar and Zimmer 2007, 2009; Fahimi 2007). All of these applications have different control requirements. Payload transportation, for instance, needs highly rigid formations and does not allow obstacles "inside" the formation. On the other hand, in an exploration task, the formation can be more flexible, in terms of shape maintenance, and it allows obstacles inside the formation. The conclusion is that it is hard to build a control architecture that suits the needs of several tasks, specially when they have disjunct requirements.

Our purpose is to build a formation control architecture for exploration type of tasks, or team translation (move a team of robots from one location to the other, but in an orderly way). This type of tasks have some desired features: (a) the ability to stabilize a formation from any initial situation; (b) avoid obstacles (either static or dynamic); (c) split formations; (d) join formations after split; (e) switch the formation shape, either if explicitly ordered or due to some events; (f) be robust against robot failure.

There are many, and diverse, approaches to solve these problems. Some of the most relevant reported results in the

S. Monteiro (✉) · E. Bicho
Dep. Industrial Electronics, University of Minho,
4800-058 Guimares, Portugal
e-mail: sergio.monteiro@dei.uminho.pt

E. Bicho
e-mail: estela.bicho@dei.uminho.pt

literature include the use of virtual structures (Lewis and Tan 1997; Young et al. 2001), vision based approaches (Das et al. 2002; Vidal et al. 2003), behaviour-based approaches (Balch and Arkin 1998; Naffin and Sukhatme 2004), leader-follower methods (Fredslund and Matarić 2002) and graph theory (Olfati-Saber and Murray 2002). The projects reported in literature usually focus only on some of the features required above, or, have only been presented in simulation. We present a formation control architecture that subsumes the leader-follower and the behavior-based approach. More specifically we use a leader-follower strategy to build the formations, with the configuration geometry being accomplished by the chain of leaders and followers. The motor control of each robot relies on an attractor dynamics approach to behavior-based robotics, where we have formation behaviors for each leader-follower desired geometry and obstacle avoidance. At each moment only one formation behavior is active (dependent on the desired configuration) together with the obstacle avoidance behavior. Particular to our approach, we use non-linear dynamical systems theory to design and implement the behaviors. Specifically, the time course of the control variables are obtained from (constant) solutions of dynamical systems. The attractor solutions (asymptotically stable states) dominate these solutions by design. The benefit is that overt behaviour of each robot is generated as a time course of asymptotically stable states, that, therefore, contribute to the overall asymptotical stability of the complete control system and makes it robust against perturbations.

We observe the desired features and document it with results from simulations and real robot implementations. One of the important features of this work is the ability to switch formations, that allows for the initial formation stabilization independent of the initial configuration, and also triggered formation changes. The other important feature, and the most relevant contribution, is the implicit split and join formations in the presence of obstacles. This is a very important characteristic, specially in cluttered environments, which are usually the grounds of exploration tasks. Other projects deal with this issue by explicitly splitting the formations and joining afterwards, which in cluttered environments, places a lot effort at the coordination level. Usually they are not demonstrated in these environments, but with only one or two obstacles, or narrow passages.

The rest of the paper is structured as follows: Sect. 2 presents a brief overview of the related work; the attractor dynamics approach is introduced in Sect. 3; we continue to Sect. 4 where we detail our control architecture; Sect. 5 analyses some of the achieved results, both in simulation and in real implementations; we finish by presenting discussion, conclusions and future work, in Sect. 6.

2 Related work

Virtual structures are one of the solutions to the formation control problem, that was very used a few years ago. Lewis and Tan (1997) were one of the first to use it. In a simple way, the desired formation matches the geometry of the virtual structure, and to each apex of it corresponds one robot; then the virtual structure is successively moved in small steps and the robots are actuated in such way to match the position of the apex to which they belong. Beard et al. (1999) subsumed it with leader–following and behavioural approaches to a multi-vehicle coordination problem. As example, they show the application to the design of a multiple spacecraft interferometer in deep space, using a virtual structure scheme. This scheme is augmented in Young et al. (2001) by introducing formation feedback. These solutions based on virtual structures, typically implement formations of the rigid type, and typically are more centralized, and, thus, suffer from the disadvantages of these. Some of these disadvantages include extensive use of communication between the robots, poor or non-existent obstacle avoidance implementation. Another centralized approach is presented in Antonelli and Chiaverini (2006). They use a two stage controller: the first being the central coordinator and the second the local controller to each robot. Specifically, they develop task functions for several different goals (rigid formations, target escort, etc.) using inverse-kinematics. Kinematic equations are developed in Barfoot and Clark (2004) that allow a team to maintain a formation. The motion planning is local to each robot, but relies on the information of a pre-planned trajectory, that acts as reference trajectory free of obstacles. Virtual structures are also used in Do and Pan (2007) combined with path tracking. Here, the virtual structure approach is modified such that the formation shape can vary. In Cruz and Carelli (2008) rigid formations are developed, allowing obstacle avoidance, that take in consideration each robot dynamic model (which are of the unicycle type).

Opposite to the virtual structures approach, which is usually more of the centralized type because of the information about the structure, i.e. about all the robots, is the behaviour-based approach. In the later, typically, the solutions are of the decentralized type. Balch and Arkin (1998) presented such an approach where, using *motor schemas*, four formation configurations (line, column, wedge and diamond) and three types of robot references (leader referenced, unit referenced and neighbour referenced) are introduced. Fredslund and Matarić (2002) also use a behaviour-based approach. Each robot maintains the formation by assuring that its *friend sensor* (a pan only video camera) sees the leader in the desired direction, and using laser scanners to measure the distance. The drawback of this approach is that it is only able to perform certain types of formations, due to the friend sensor. A different approach is used by Hong et

al. (2001). They used fuzzy systems to control each robot speed, and fuzzy-neuro systems to implement the obstacle avoidance behaviour. Differential flat systems are utilized in Pledgie et al. (2002) to model the robots behaviour. Kim et al. (2001) use Petri-nets to model three control subtasks: formation maintenance, identification and formation generation.

Two types of controllers for follower robots navigating in a multi-robot formation are proposed in Desai et al. (2001): distance-bearing, or l–ϕ, and distance to two leaders, or l–l. Using the first, a follower tries to maintain fixed the distance and bearing to a leader, while in the second a follower tries to maintain fixed the distance to two leaders. Control laws for the followers are based on non-linear control theory, while the geometry and type of relations between robots are based on graph theory. They also define a transition matrix that governs the addition and deletion of edges to the control graph (which changes either the formation or the relations between the robots). A similar solution appears in Das et al. (2002). There, several control laws are defined: leader–follower, leader–obstacle, for three robots, etc. These controllers are then integrated in a coordination protocol, by switching among them to the proper one, given the desired formation (formulated as a control graph) and the world. Gustavi and Hu (2005) also propose two controllers for a leader–follower strategy: one for horizontal tracking (what, here, we call line formations) and other for vertical tracking (for formations other than line). They distinguish their work from the previous by explicitly assuming sensor and actuator constraints. However, they only present simulations (in an obstacle free environment). A different approach by Vidal et al. (2003), is based on omni directional vision. They use motion segmentation techniques to estimate the position and velocities of each leader and omni directional visual servoing for tracking and collision avoidance. The formation is specified in the image plane. Also in the leader–follower category, Ren and Sorensen (2008) mixes a distributed cooperative control strategy, based on consensus algorithms with an arbitrary number of group leaders in the formation. In Fahimi (2008) the l–ϕ approach is extended to 3D formations of small-sized helicopters, where the control laws are designed by using sliding-mode control.

Other types of team organization includes those that deal with artificial potentials. In Leonard and Fiorelli (2001) a different formation control approach, relying on virtual leaders and artificial potentials, is defined. To each real robot and virtual leader corresponds an artificial potential. Then, virtual leaders are placed amongst the real robots to define the formation. This framework is extended in Ogren et al. (2002) by allowing formations to be translated, rotated, expanded and contracted. Gazi (2005) integrates artificial potentials and sliding-mode control in the control strategy to drive swarms of robots, which can also be used in formations. Balch and Hybinette (2000) present a new class of potential functions, called social potentials. Here the formation is specified by choosing each robot attachment sites, that are the places that attract other robots. Similar to these attachment sites (but not related to artificial potentials) is the notion of local templates as presented in Krishnanand and Ghose (2005). They enable the robots to self-assemble into grid, line and wedge patterns. A different approach to formation definition is presented by Ge and Fua (2005). The concept of queues, instead of nodes, is introduced to build the formation. Examples of queues can be the sides of a polygon, when dealing with polygon shaped formations. These queues are user defined and can assume any form.

Sometimes, it is necessary that a team of robots divides itself into two sub-teams. This can be caused by the necessity of overtaking some obstacles, or go trough some narrow passages. Thus the importance of the ability to split and join formations. Olfati-Saber and Murray (2002) present a framework that enables formation split and join, by exploring the properties of rigid graphs. An explicit split and join (and vice-versa) maneuver is addressed in Ogren (2004). The principle is that a vehicle should only travel in a formation (rigid one) if it is suitable for it (has a common path segment with its leader in the formation, for instance) and there are no obstacles to avoid.

In the presented related work, there were several different approaches, but, typically, there was a separate coordination level (doing *decision making*, like obstacle avoidance, breaking or joining the formation, etc.) and a control level (implementing the actions decided at the coordination level). We, instead, have only one layer, that provides control signals (like path velocity and rate of change of heading direction) and also does dynamic decision making, by using the so-called *Nonlinear Attractor dynamics approach to behaviour generation*. This framework (see next section for an introduction on the basic principles) was first presented in Schöner and Dose (1992) and Schöner et al. (1995). Since then it has been demonstrated in several different environments and performing different tasks. Steinhage (1997) simulated it extensively and introduced methods for action selection. Bicho extended the approach and made it work in low-level (both computing and sensoring) vehicle platforms (Bicho and Schöner 1997; Bicho et al. 2000; Bicho 2000). The first example of cooperative navigation, by two robots, heading to a target was simulated by Large et al. (1999). To the best of our knowledge, the first approach to formation control using the attractor dynamics approach, was made by us (Monteiro and Bicho 2002; Bicho and Monteiro 2003; Monteiro et al. 2004). In parallel we have extended this approach/work to the problem of coordinating and controlling teams of autonomous mobile robots that must transport a rigid object from an initial position to a final target destination (Soares and Bicho 2002; Bicho et al. 2004; Soares et al. 2007), and to 3D formations of lighter than air airships (Bicho et al. 2006).

This paper results as a corollary and extension of our previous research in the formation control domain. The following is a new contribution: (i) Lyapunov stability analysis that have not been include before, and proves the asymptotic stability of the control laws. This is a positive feature compared to typical behavior based approaches since, usually, the control law is based only on the superposition of heuristic functions for each behavior, and mathematically proving the asymptotic stability of the control laws is difficult and sometimes impossible. (ii) We present, analyze and discuss a set of new results that have not yet been presented, namely: (a) Our first implementations completely communication free. This result is important because by design the control parameters (distance and bearing angle to leader) were chosen in a way such that the approach does not need communication (in order for the system to work properly) as long as the robots are equipped with appropriate sensors (e.g. the vision system) that provides estimates of these parameters. (b) Simulation results that show robustness against robot failure. (c) Simulation results of teams navigating in very cluttered and dynamic environment (i.e. moving obstacles). (d) Simulation results with larger teams and more complex formation shapes.

As a final note, it is important to stress that although both the Potential Field approach and the Nonlinear Attractor Dynamics approach represent obstacles and target as repellers and attractors, respectively, the two approaches are quite different. In the potential field approach the gradient of a scalar potential field is used to generate the robot's trajectory. Thus, the path is generated by the transient solutions of a dynamical system. On the other hand, in the nonlinear attractor dynamics approach the path is generated by a sequence of attractor solutions. Thus the transient solutions of the potential field approach are replaced by a sequence of attractor solutions (i.e. asymptotical stable states) of a dynamical system, that therefore contribute to the asymptotical stability of the overall control system (for a comparison of these two approaches see e.g. Fajen et al. 2003; Costa e Silva et al. 2006).

3 Basics of the nonlinear attractor dynamics approach

We start with a brief review of the basic ideas of the dynamic approach to behaviour generation (Schöner and Dose 1992; Schöner et al. 1995; Bicho and Schöner 1997; Steinhage 1997; Bicho et al. 2000; Bicho 2000): (1) *Behavioural variables* are used to describe, quantify and internally represent the state of the robot system with respect to elementary behaviours. For an autonomous mobile robot moving in the plane, the *heading direction*, ϕ_i ($0 \leq \phi_i \leq 2\pi$ rad), with respect to an arbitrary but fixed world axis, and *path velocity*, v_i, are appropriate behavioural variables; (2) Behaviour is generated by continuously providing values to these variables, which control the robot wheels. The time course of each of these variables is obtained from (constant) solutions of dynamical systems. The attractor solutions (asymptotically stable states) dominate these solutions by design. In the present system, the behavioural dynamics of heading direction, $\phi_i(t)$, and velocity, $v_i(t)$ ($i = leader,\ follower$) are differential equations

$$\dot{\phi}_i = f_i(\phi_i, parameters), \tag{1}$$

$$\dot{v}_i = g_i(v_i, parameters). \tag{2}$$

Task constraints define contributions to the vector fields. Each constraint may be modelled either as a repulsive or as an attractive force-let, which are both characterized by three parameters: (a) which value of the behavioural variable is specified? (b) how strongly attractive or repulsive the specified value is?; and (c) over which range of values of the behavioural variable a force-let acts? Thus, in isolation, each force-let creates an attractor (asymptotically stable state) or a repeller (unstable state) of the dynamics of the behavioural variables. An attractive force-let serves to attract the system to a desired value of the behavioural variable (e.g. the direction in which a target lies for the heading direction or a desired velocity value for the path velocity). A repulsive force-let is used to avoid the values of the behavioural variable that must be avoided (for example, the directions in which obstacles lie are values that the heading direction must avoid). The resultant dynamical systems are non–linear and may have multiple stable states (attractors) that change in time. By design, parameters are tuned such that the behavioural variables are very close to one attractor of the resultant dynamics most of the time, i.e. the variables follow very closely one of the moving attractors. Thus the behaviour of each robot is generated as a time series of asymptotically stable states. The fact that only attractor solutions matter can be used to design the layout of attractors and repellers using the qualitative theory of dynamical systems. Qualitative changes in the behaviours are brought about by means of bifurcations in the vector fields. Local bifurcation theory helps to design the dynamics such that these qualitative changes are automatically made under the appropriate environmental conditions (e.g. sensory information or shared information among the team of robots).

In the next section, we will build the behavioural dynamics, i.e. we derive the vector fields of (1) and (2), for each robot that generate formation control.

4 Building robot formations

We use a leader–follower strategy to organize the robot team. Each team is composed of a *Lead robot* (Desai et

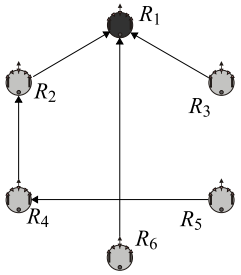


Fig. 1 Example of hexagon formation, showing the leader–follower structure. *Lead robot* is the darker one

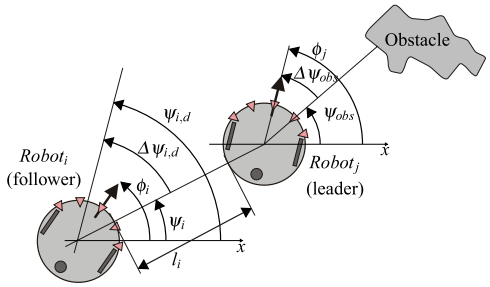


Fig. 2 Example of notation used in this paper

al. 2001) (other authors also use the name *conductor* (Fredslund and Matarić 2002)) and several (at least one) *follower* robots. The purpose of the *Lead robot* is to act as a team leader. It knows where the team final destination, or target, is and heads toward it. All the other robots, the *followers*, follow the *Lead robot*, either directly or indirectly (through another team mate, that, on its turn, follows the *Lead robot*).

Consider the example in Fig. 1, where a team with six robots is organized in a hexagon formation. Here, robot R_1, is the *Lead robot* and all the other are followers. Robots R_2, R_3 and R_6 directly follow the *Lead robot*. R_4 follows R_2 that, on its turn, follows R_1, so, it follows the *Lead robot* indirectly. One step down in the chain of leadership is robot R_5 that follows R_4. So, in our formations we have a chain of leaderships, where one robot can be a leader to another robot, while at the same time it is following another one. In this type of organizational architecture we can envisage that all robots are connected in leader–follower pairs, and we can reduce the problem of formation control at team level to a set of formations with only two robots. Here only the followers have the responsibility to maintain the formation (it is a unidirectional relationship). One of the benefits is that communication and sensorial requirements are highly reduced. If each set is in formation, i.e. if each *follower* is in the exact location regarding its *leader*, then a global formation is achieved. In the next subsections we show the behavioural dynamics to build a formation with only two robots (i.e. we derive the vector fields in (1) and (2)), and then generalize to a formation with N-robots.

4.1 Formations with two robots

When we only have two robots travelling in a formation, where the distance between them should be kept constant, one of three situation occurs: they should either travel one behind the other, or side by side or diagonally at a desired angle. To these three situations we call, respectively, column formation, line formation and oblique formation.

In order to try to avoid communication, and make the task easier for real world implementations, we will attempt to build the controllers only with information that each robot can directly (and easily) collect. Information about others behaviour, like velocity and, specially, heading direction is highly restricted. This is why heading direction controllers will not be dependent on the leader's heading.

Before we proceed we will introduce the notation used throughout this paper (Fig. 2 shows an example for clarity):

$Robot_j$: leader robot;
$Robot_i$: follower robot;
l_i: the actual distance of $Robot_i$ to its leader ($l_{i,d}$ is the desired distance);
ϕ_j: heading direction of the leader;
ϕ_i: heading direction of the follower;
ψ_{obs}: direction at which an obstacle is sensed;
$\Delta\psi_{obs}$: angular difference between the leader heading direction, ϕ_j, and the direction at which it senses obstacles, ψ_{obs};
ψ_i: direction at which the follower sees the leader;
$\Delta\psi_{i,d}$: desired angular difference between heading direction of the follower, ϕ_i, and the direction at which it sees the leader, ψ_i;
$\psi_{i,d}$: desired follower heading, i.e., the heading that keeps the follower in formation.

All the angles are measured with respect to the robot external reference frame, which is fixed, but can be arbitrarily chosen. Please note that the robots in the team do not need share the same reference frame.

Now, we continue by detailing, for each formation type, how to design the controller dynamics.

4.1.1 Column formation

$Robot_i$ is said to drive in column formation with $Robot_j$ if it drives behind it at a desired distance (see Fig. 3).

This type of formation is important in tasks involving convoying (Bom et al. 2005; Hu and Zeigler 2004). Even if the robots are organized in other formation shapes, it might be useful to switch to a column shape, prior to pass through a narrow corridor. Traffic flow is also improved by making agents organize and travel in columns (AGV delivery in a plant, or traffic in roads, for instance).

To be in column formation, the *follower* ($Robot_i$) must drive behind its *leader* ($Robot_j$), i.e. it must steer to the direction where it sees the *leader* with respect to the external

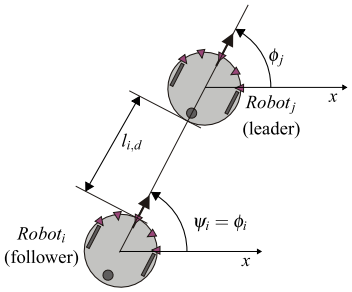


Fig. 3 Two robots in column formation. For column formation ψ_i is the desired value for ϕ_i

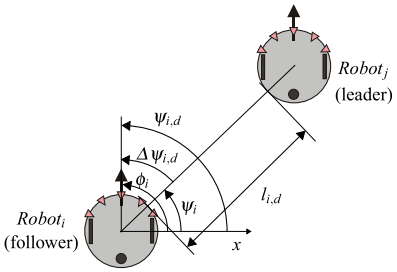


Fig. 4 Two robots in an oblique formation. $\Delta\psi_{i,d}$ is the desired relative angle between the follower ($Robot_i$) and its leader ($Robot_j$)

reference frame. This means that the desired relative angle between the leader and the follower is zero ($\Delta\psi_{i,d} = 0$) and thus the desired value for the heading direction of $Robot_i$ is ψ_i ($\psi_{i,d} = \psi_i$). A simplest dynamical system for $Robot_i$'s heading direction that generates navigation in column formation taking its *leader* as a reference point is

$$\dot{\phi}_i = f_{col,i} = -\lambda_{col} \sin(\phi_i - \psi_i) \tag{3}$$

which erects an attractor for ϕ_i directly at the direction at which the *leader* lies as seen from the current position of the *follower* (i.e. ψ_i). λ_{col} (> 0) is the strength of attraction to the attractor and corresponds to the relaxation rate to that attractor (i.e. inverse of the local relaxation time).

Besides heading direction, in order to ensure the formation stabilization and maintenance, one must also adjust path velocity. In the case of column formation, path velocity is dependent on the distance between the follower and the leader: the follower should have a lower velocity than the leader if it is closer than desired, or a larger velocity otherwise. This can be accomplished by setting the desired velocity, $v_{i,d}$, equal to

$$v_{i,d} = \begin{cases} v_j - (l_{i,d} - l_i)/T_\Delta & \text{if } l_i \geq l_{i,d} \\ -v_j - (l_{i,d} - l_i)/T_\Delta & \text{else} \end{cases} \tag{4}$$

where v_j is the reference velocity, l_i is the actual, or measured, distance between the robots and T_Δ is a parameter that smooths the robot movement, by controlling its accelerations and decelerations. The reference velocity is the leader's velocity, if available, or the *cruise velocity* (the velocity at which the leader travels when not in the presence of obstacles) that is a mission parameter and known by all robots. The dynamical system that generates the time series for the robots path velocity is a linear dynamical system, of the type

$$\dot{v}_i = g(v_i) = -\alpha_i (v_i - v_{i,d}) \tag{5}$$

that erects an attractor at the desired path velocity, $v_{i,d}$. α_i (> 0) controls the relaxation rate to that attractor.

Our decision to change the sign of v_j was because it allows eventually the follower robot to move backwards when it is too close to its leader. Please note that (4) is setting the attractor value (stable equilibrium point) for the path velocity of the follower, not directly the velocity command. Thus, although the attractor value may change discontinuously (which happens when it changes sign from positive to negative, if the distance from the follower to its leader is shorter than the desired one, or vice-versa) the values for the path velocity, v_i, will always change smoothly and continuously in time because it asymptotically converges to the attractor value, as defined by the 1st order dynamical system (see (5)). See also Appendix A.

4.1.2 *Oblique formation*

We say that $Robot_i$ drives in oblique formation with respect to $Robot_j$ when during its motion it maintains fixed (equal to a pre-defined angle $\Delta\psi_{i,d}$) the direction at which it sees $Robot_j$ (see Fig. 4).

This type of formation enables us to build formations like echelons, wedges or V-type formations that are most useful in tactical or military operations (Balch and Arkin 1998; Edwards et al. 2004), or, when in 3D, also enables efficient energy expenditure at team level (Seiler et al. 2002; Fowler and Andrea 2002; Nangia and Palmer 2007).

A dynamical system for the heading direction of $Robot_i$ that generates "oblique" formation taking $Robot_j$ as a reference point is

$$\begin{aligned} \dot{\phi}_i &= f_{oblique}(\phi_i) \\ &= f_{approach}(\phi_i) + f_{divert}(\phi_i) \end{aligned} \tag{6}$$

where each term defines an attractive force (see Fig. 5)

$$f_k(\phi_i) = -\lambda_{oblique} \lambda_k(l_i) \sin(\phi_i - \psi_k) \tag{7}$$

with $k = approach, divert$, where the first contribution, $f_{approach}$, erects an attractor at a direction

$$\psi_{approach} = \psi_i + \Delta\psi_{i,d} - \text{sign}(\Delta\psi_{i,d})\gamma \tag{8}$$

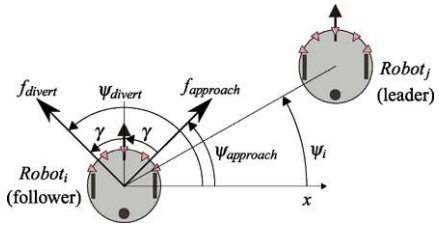


Fig. 5 Heading direction of the two forces acting on a follower, when generating the oblique formation

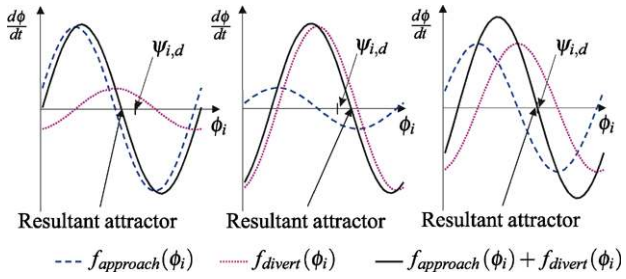


Fig. 6 This figure shows the two contributions to the "oblique" formation dynamics and their superposition for the three different physical situations: robots further than desired ($l_i > l_{i,d}$), closer than desired ($l_i < l_{i,d}$) and at the desired distance ($l_i = l_{i,d}$)

in which γ represents the separation between both attractors $f_{approach}(\phi_i)$ and $f_{divert}(\phi_i)$, or the range of values that the desired heading can take (actually the separation is 2γ).

The strength of this attractor ($\lambda_{oblique}\lambda_{approach}(l_i)$ with $\lambda_{oblique}$ fixed), increases with distance, l_i, between the two robots:

$$\lambda_{approach}(l_i) = 1/(1 + \exp(-(l_i - l_{i,d})/\mu)). \tag{9}$$

The second contribution, f_{divert}, sets an attractor at a direction pointing away from the *leader*,

$$\psi_{divert} = \psi_i + \Delta\psi_{i,d} + \text{sign}(\Delta\psi_{i,d})\gamma \tag{10}$$

with a strength ($\lambda_{oblique}\lambda_{divert}(l_i)$) that decreases with distance, l_i, between the robots,

$$\lambda_{divert}(l_i) = 1 - \lambda_{approach}(l_i). \tag{11}$$

Because these two attractive forces are overlapping only one attractor results from their superposition. The direction at which the resulting attractor emerges depends on the distance between the two robots. This is illustrated in Fig. 6.

In this example $\gamma = \pi/4$. If the distance between the two robots is larger than the desired one, then the attractive force erected at direction $\psi_{approach}$ is stronger than the attractive set at direction ψ_{divert}. Their superposition leads to an attractor at a direction still pointing toward the direction of the leader robot (see left plot). Conversely, when the distance between the two robots is smaller than the desired distance (situation depicted in the middle plot), the reverse holds, i.e.

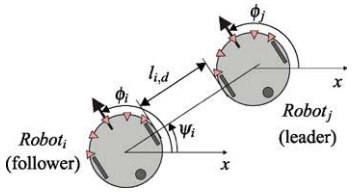


Fig. 7 Two robots in a line formation. $Robot_j$ is the *leader* of $Robot_i$ which must drive such that it sees its *leader* perpendicularly (i.e. $\psi_{i,d} = \psi_i + \pi/2$ if $Robot_i$ is to the left or $\psi_{i,d} = \psi_i - \pi/2$ if $Robot_i$ is to the right) and simultaneously keep a desired distance, $l_{i,d}$, between them

the attractive force set at direction $\psi_{approach}$ is now weaker than the attractive force at direction ψ_{divert}. The resulting oblique formation dynamics exhibits an attractor at a direction pointing away from leader's direction. In the right plot the robots are now at the desired distance. The two attractive forces have the same strength which leads to a resultant attractor at the direction $\psi_{i,d} = \psi_i + \Delta\psi_{i,d}$, which is the desired one.

Path velocity is controlled exactly in the same way as for column formation.

4.1.3 Line formation

Two robots are said to be in line formation if they drive side-by-side at a desired distance (see Fig. 7).

Line formations are useful in tasks that deal with efficient coverage of environments. Typical examples are those of outdoors exploration or reconnaissance, sweeping (Sudsang 2002) or mowing the lawn.

A behavioural dynamics for the heading direction of the follower, $Robot_i$, that generates line formation taking $Robot_j$ as a reference point is given by the same dynamical systems as for "oblique" formation. The difference is that $\Delta\psi_{i,d} = \pm\pi/2$ (depending on the desired location of the leader being on the right or the left of the follower) in (8) and (10).

Path velocity is controlled by a set of heuristic rules that give the desired value for the robot's velocity, $v_{i,d,line}$, i.e. the attractor for the velocity dynamics (see (5)) is given by Soares and Bicho (2002)

$$\begin{aligned} v_{i,d,line} = {} & DE_1 \cdot v_j(1 - |\sin(\psi_i)|) \\ & + DE_2 \cdot v_j(1 - |\cos(\psi_i)|) \\ & + AC_1 \cdot v_j(1 + K_v|\sin(\psi_i)|) \\ & + AC_2 \cdot v_j(1 + K_v|\cos(\psi_i)|) \end{aligned} \tag{12}$$

where DE_1, DE_2, AC_1 and AC_2 are mutually exclusive activation variables that reflect the relative attitude of the leader regarding the follower. They are activated by testing ϕ_i and ψ_i in the way depicted by Fig. 8 and by using the following

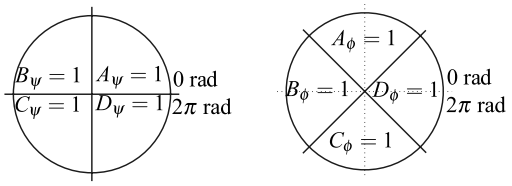


Fig. 8 Two trigonometric circles representing the activation of logical variables. *Left*: results from testing the direction at which the follower sees the *leader* ψ_i. If $0 \leq \psi_i < \pi/2$ then $A_\psi = 1$ and $B_\psi = C_\psi = D_\psi = 0$. If $\pi/2 \leq \psi_i < \pi$ then $B_\psi = 1$ and $A_\psi = C_\psi = D_\psi = 0$, and so on. *Right*: results from testing the heading direction of the follower (ϕ_i)

boolean functions.

$$DE_1 = A_\psi B_\phi + B_\psi C_\phi + A_\phi (C_\psi + D_\psi) \tag{13}$$

$$DE_2 = A_\psi C_\phi + D_\psi B_\phi + D_\phi (B_\psi + C_\psi) \tag{14}$$

$$AC_1 = A_\psi (A_\phi + D_\phi) + C_\psi (B_\phi + C_\phi) \tag{15}$$

$$AC_2 = B_\psi (A_\phi + B_\phi) + D_\psi (C_\phi + D_\phi). \tag{16}$$

When the situation requires that the follower's velocity is increased over the leader's velocity, then AC_1 or AC_2 are activated. Essentially, this happens when the follower is behind the leader. Otherwise, DE_1 or DE_2 are activated instead.

4.2 Generalizing to N-robots formations

Here we show how to extend the solutions presented in previous sections to build teams of N-robots and also present examples of maintaining particular geometric configurations.

Teams of robots with more than two robots are built by specifying pairs of leader–follower teams and stating the particular configuration to achieve. A complete team specification is accomplished by means of a *formation matrix* as follows

$$\mathbf{S} = \begin{pmatrix} L_1 & \Delta\psi_{1,d} & l_{1,d} \\ L_2 & \Delta\psi_{2,d} & l_{2,d} \\ \cdots & \cdots & \cdots \\ L_N & \Delta\psi_{N,d} & l_{N,d} \end{pmatrix}. \tag{17}$$

This matrix codes the shape of the formation in the following way: Row i $(= 1, 2, 3, \ldots, N)$ defines the pose of robot R_i in the formation. It is a vector $\mathbf{S}_i = (L_i\ \Delta\psi_{i,d}\ l_{i,d})$, where L_i $(L_i \neq R_i)$ identifies the *leader* robot for Robot i, $\Delta\psi_{i,d}$ is the desired relative angle between Robot i and its *leader* and $l_{i,d}$ the desired distance to its *leader*.

When Robot i is the *Lead Robot* the parameters for its dynamics are $L_i = 0$, $\Delta\psi_{i,d} = 0$ and $l_{i,d}$ defines the distance at which it must stop from the target location.

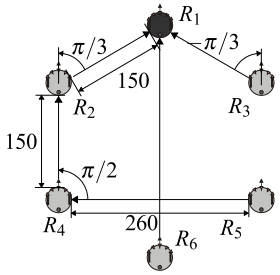


Fig. 9 Hexagon formation as determined by $\mathbf{S}_{\text{hexagon}}$

For example, one formation matrix that determines the shape of a hexagon formation (see Fig. 9) is

$$\mathbf{S}_{\text{hexagon}} = \begin{pmatrix} 0 & 0 & 150 \\ 1 & \pi/3 & 150 \\ 1 & -\pi/3 & 150 \\ 2 & 0 & 150 \\ 4 & -\pi/2 & 260 \\ 1 & 0 & 300 \end{pmatrix}. \tag{18}$$

We assume that Robot R_1 is the *Lead robot* (i.e. moves toward the target location), and that the desired distance between the robots is 150 cm. Robot R_1 is the *Lead Robot*, Robot R_2 follows R_1 on the left side and maintaining an oblique formation, Robot R_3 follows R_1 on the right side and maintaining an oblique formation. Robot R_4 follow Robot R_2 in a column formation. Robots R_5 follow Robot R_3 maintaining a line formation on the right. Robot R_6 follows R_1 in column formation. Figure 1 shows a representation of the referred hexagon pattern.

It is important to note that there are many formation matrices that generate the same geometric configuration for the formation. This is an important feature of our approach since it permits to cope with robot failure (cf. Sect. 5.1.4). Throughout this paper we assume that the shape of the formation, defined by the **S** matrix, is determined by a higher level controller, or external user, and that it can be changed at any time. Nevertheless we have started to work on this topic and already have some preliminary work (Monteiro and Bicho 2008). Other important references on the allocation of robots to places in the formation issue are Fredslund and Matarić (2002), Kostelnik et al. (2002), Brimble and Press (2003), Archibald and Frost (2007), Kaminka et al. (2008).

4.3 Behaviours integration

The behaviours described in the previous subsections are integrated together with an obstacle avoidance behaviour into a single vector field, both at the level of heading direction dynamics and path velocity dynamics.

The obstacle avoidance dynamics at the level of the heading direction is defined as (see Bicho and Schöner 1997; Bicho 2000 and Bicho et al. 2000 for more details):

$$\dot{\phi}_i = F_{obs,i} = \sum_s f_{obs,s,i}(\phi_i) \tag{19}$$

where $f_{obs,s,i}$ are repulsive "force-lets", for robot R_i, defined around each direction in which obstructions (either due to static obstacles or due to the other robots) are sensed. These are characterized by (a) the direction, $\psi_{obs,s}$, to be avoided, (b) the strength of repulsion, $\lambda_{obs,s}$, and (c) the range, $\sigma_{obs,s}$ over which repulsion acts. These repulsive force-lets can be straightforwardly erected by the distance sensors:

$$f_{obs,s,i}(\phi_i) = \lambda_{obs,s}(\phi_i - \psi_{obs,s}) \exp\left(-\frac{(\phi_i - \psi_{obs,s})^2}{2\sigma_{obs,s}^2}\right). \tag{20}$$

We integrate the behaviours by adding each contribution to the same vector field:

$$\begin{aligned}\dot{\phi}_i = &\sum_s f_{obs,s,i}(\phi_i) + \gamma_{col,i} f_{col,i}(\phi_i) \\ &+ \gamma_{oblique,i} f_{oblique,i}(\phi_i) \\ &+ \gamma_{line,i} f_{line,i}(\phi_i) + f_{stoch}\end{aligned} \tag{21}$$

where $\gamma_{col,i}$, $\gamma_{oblique,i}$ and $\gamma_{line,i}$ are mutually exclusive boolean variables that determines which configuration is desired for the formation. The vector field is augmented with a stochastic contribution, f_{stoch}, that guarantees escape from repellers, because due to a bifurcation in the vector field, it may happen that the attractor in which the system was sited becomes a repeller, and simultaneously, in simulation, models perturbations.

In terms of path velocity, a similar approach is followed. The velocity dynamics vector field is given by

$$\begin{aligned}\dot{v}_i = &\gamma_{obs,i} g_{obs,i}(v_i) + \gamma_{col,i} g_{col,i}(v_i) \\ &\gamma_{oblique,i} g_{oblique,i}(v_i) + \gamma_{line,i} g_{line,i}(v_i)\end{aligned} \tag{22}$$

where each contribution defines simply a linear dynamical system for the path velocity:

$$\dot{v}_i = g_k(v_i) = -\alpha_k(v_i - v_{i,d,k}) \tag{23}$$

with $k = \{obs, line, col, oblique\}$, that sets an attractor at the desired path velocity, $v_{i,d,k}$, with a relaxation rate controlled by α_k (> 0).

The desired path velocity, $v_{i,d,k}$, is a function of whether or not obstacles are sensed and by the requirement to keep a desired distance to the *follower* robot.

When the *followers*' heading direction is inside the repulsion range created by sensed obstructions then the obstacle avoidance term dominates (i.e. $\gamma_{obs,i} = 1$, $\gamma_{line,i} = 0$, $\gamma_{col,i} = 0$ and $\gamma_{oblique,i} = 0$) and in this case the desired path value for the path velocity is:

$$v_{i,d,obs} = d_{min}/T_{2c,obs} \tag{24}$$

which tries to stabilize a particular time to contact, $T_{2c,obs}$, with the obstacle. d_{min} is the minimum distance given by the distance sensors. Reversely, when no obstructions are sensed or the robot's heading direction is outside the repulsive effect of obstacle contributions then the particular desired value for the velocity depends on the desired configuration for the formation.

To detect whether the heading direction of the robot is inside the repulsion range, or not, created by the obstacles, we use the potential function of the obstacle avoidance dynamics, $F_{obs,i}$, as given by

$$U_{obs,i}(\phi_i) = \sum_s \lambda_{obs,s}\sigma_s^2 \left[\exp\left(-\frac{(\phi_i - \psi_s)^2}{2\sigma_s^2}\right) - \frac{1}{\sqrt{e}}\right]. \tag{25}$$

For a given heading, ϕ_i, positive values of the potential function indicate that it is inside the repulsion range of the senses obstacles. If the value is negative, then either the repulsion is very weak, or it is outside the repulsion range (see Bicho 2000 or Bicho et al. 2000 for more details).

When sensorial information changes attractors move. In order to ensure the stability of the control system, i.e., that the system always relaxes to an attractor as they shift and also to ensure that the obstacle avoidance behaviour has precedence over the maintain formation behaviour, the following hierarchy of relaxation rates has to be observed (with $k = \{col, oblique, line\}$):

$$\lambda_k \ll \alpha_k, \qquad \lambda_{obs} \ll \alpha_{obs}, \qquad \lambda_k \ll \lambda_{obs}. \tag{26}$$

4.4 Stability analysis

The analysis of stability is relatively simple. For the nonlinear dynamical system of (21), that governs the heading direction of each robot, ϕ_i, we can consider the following Lyapunov function

$$\begin{aligned}V_i(\phi_i) = &V_{obs,i}(\phi_i) + \gamma_{col,i} V_{col,i}(\phi_i) \\ &+ \gamma_{oblique,i} V_{oblique,i}(\phi_i) \\ &+ \gamma_{line,i} V_{line,i}(\phi_i) + K_{pot}\end{aligned} \tag{27}$$

where $K_{pot} \in \mathbb{R}$ and

$$V_{obs,i}(\phi_i) = \sum_s \lambda_{obs,s}\sigma_s^2 \exp\left(-\frac{(\phi_i - \psi_s)^2}{2\sigma_s^2}\right) \tag{28}$$

$$V_{col,i}(\phi_i) = -\lambda_{col}\cos(\phi_i - \psi_i) \tag{29}$$

$$\begin{aligned}V_{oblique,i}(\phi_i) = &-\lambda_{oblique}\lambda_{approach}\cos(\phi_i - \psi_{approach}) \\ &- \lambda_{oblique}\lambda_{divert}\cos(\phi_i - \psi_{divert})\end{aligned} \tag{30}$$

$$\begin{aligned}V_{line,i}(\phi_i) = &-\lambda_{line}\lambda_{approach}\cos(\phi_i - \psi_{approach}) \\ &- \lambda_{line}\lambda_{divert}\cos(\phi_i - \psi_{divert}).\end{aligned} \tag{31}$$

Choosing appropriate values for K_{pot} we may guarantee the following condition:

$$V_i(\phi_i) > 0, \quad \forall \phi_i \in [0, 2\pi[.$$

Next, since

$$\dot{V}_i(\phi_i) = \frac{dV_i(\phi_i)}{dt} = -[f_i(\phi_i)]^2 < 0, \quad \forall \phi_i \in [0, 2\pi[\setminus\{\tilde{\phi}_i\} \tag{32}$$

with

$$f_i(\phi_i) = \sum_s f_{obs,s,i}(\phi_i) + \gamma_{col,i} f_{col,i}(\phi_i) + \gamma_{oblique,i} f_{oblique,i}(\phi_i) + \gamma_{line,i} f_{line,i}(\phi_i) \tag{33}$$

where $\tilde{\phi}_i$ are the fixed points of (21), that is, are the zeros of $f_i(\phi_i)$. Thus, by the Lyapunov direct method, the dynamical system for the heading direction is asymptotically stable.

Path velocity is controlled by a linear dynamical system (see (23)). It can be easily proven, trough linear stability theory, that it is also asymptotically stable because $\frac{dg_k(v_i)}{dv_i} = -\alpha_k < 0$.

5 Results

In this section we will present a selection of important results that highlight the main characteristics of our approach. The results are presented into two separate subsections. The first contains results from computer simulations and the second contains real implementation results (with real robots).

For each experiment we will show robots' trajectories evolution, formation errors evolution and sample snapshots for the experiments with real robots. Trajectories evolution is self-explanatory: consists of an X–Y plot with the path travelled by each robot.

Some authors characterize the performance of formations by analyzing the error between pairs of robots (Fierro and Alur 2002). This is a valid metric when the controllers are designed taking in consideration the leader's heading. Because we don't use the leader's heading, analyzing the formation between pairs of robots does not capture how well the formation is performing (see Fig. 10).

To characterize the whole formation we have to take, at least, the *lead robot's* heading direction into account. Since we assume we don't have this information online, we can only compute this error after running the experiment as a way to characterize the performance of the formation. For this same reason, we cannot use the formation error as online formation feedback.

In order to analyze the formation performance we compute, for each robot, the distance error between its actual position and the expected position that it should occupy in the correct formation. For robot R_i the distance error, or position error, $\Delta_{d,i}$ is given by

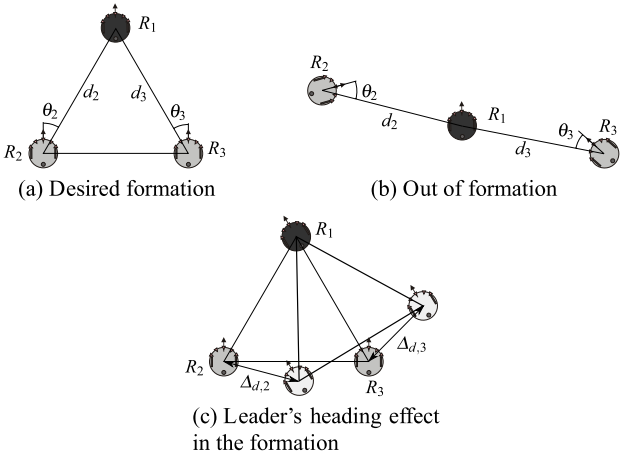


Fig. 10 (**a**) Desired pattern formation described by the distance, d_2, and bearing, θ_2, robot R_2 should keep to R_1, and the distance, d_3, and bearing, θ_3, robot R_3 should keep to R_1. (**b**) In this situation both R_2 and R_3 are at the same distances and bearings to the *Lead robot* as in (**a**), but the team is not in the desired pattern formation. (**c**) The team in the darker locations is not with zero formation error, although distance and bearing errors are zero

$$\Delta_{d,i} = \sqrt{(x_i - x_l + d_{il}\cos(\eta))^2 + (y_i - y_l + d_{il}\sin(\eta))^2}$$
$$\eta = \phi_l - \theta_{il} \tag{34}$$

where x_l, y_l, ϕ_l are the *lead robot's* x and y coordinates and heading direction, respectively. x_i, y_i, ϕ_i are the x and y coordinates and heading direction, respectively, of $Robot_i$. θ_{il} is the desired difference between the heading of $Robot_i$, if it would be in the correct position, and the direction at which it sees the *Lead robot*. d_{il} is the desired distance between $Robot_i$, if it was in the correct position, and the *Lead robot*. θ_{il} and d_{il} can be understood as the parameters $\Delta\psi_{i,d}$ and $l_{i,d}$ in a new formation matrix (see (17)) where each *follower* robot follows directly the *Lead robot*. This new formation matrix, can be calculated from the original one, to which is equal in shape. The only purpose of this new matrix is to aid in the calculation of the formation error, $\Delta_{d,i}$, because this error is *Lead robot* referenced.

In the large robot experiments (c.f. Sect. 5.3), because we don't have the Cartesian coordinates of the robots, we rewrote (34) to the situation where the distances between the robots and the angles can be measured. The position error, using the law of cosines, can be written as (see Fig. 11):

$$\Delta_{d,i} = \sqrt{l_i^2 + l_{i,d}^2 - 2 l_i l_{i,d}\cos(\Delta\psi_l)} \tag{35}$$

where l_i and $l_{i,d}$ are the actual and the desired distance between the *follower* and the *leader*, and $\Delta\psi_l = \psi_l + \Delta\psi_{i,d} - 3\pi/2$. ψ_l is the direction at which the *follower* appears, in

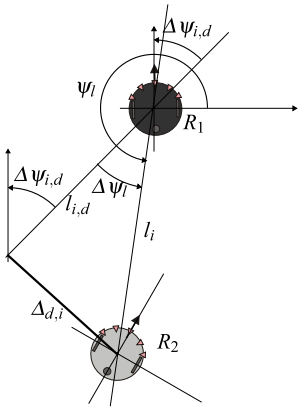


Fig. 11 Formation error. The *lead robot* is the darker robot. The error is computed in the *lead robot's* reference frame

the *leader's* reference frame. Please note that this value is not necessary while running the formation. We only use it to compute the formation error, which is done after the experiment.

If all robots have null $\Delta_{d,i}$, then the desired pattern formation is achieved. The formation error is computed as an average of the sum of the position errors of all robots:

$$\text{formation error} = \frac{1}{N-1}\sum \Delta_{d,i}. \tag{36}$$

In practice, for the analysis of error formations, we consider as an evaluation criteria, a good formation if the average $\Delta_{d,i}$ is below a certain threshold (we use 10% of the desired inter-robot distance because this also accounts for measuring errors).

5.1 Simulation results

A software simulator written in MATLAB was used to evaluate the proposed approach performance. The robotic platforms were modeled based on the following physical characteristics: a circular differential drive robot (30 cm diameter), with one caster wheel and nine infra-red distance sensors, positioned around the front of the robot. They are separated by 30°, to model each sensor field of view (which is approximately 30°). Distance sensors are simulated through an algorithm reminiscent of ray-tracing, with limited range (up to 60 cm). In the simulation the robots are represented as triplets (x_i, y_i, ϕ_i) $(i = 1, 2, \ldots, N)$, consisting of the corresponding two Cartesian coordinates and the heading direction. Cartesian coordinates are updated by a dead-reckoning rule ($\dot{x_i} = v_i \cos(\phi_i)$, $\dot{y_i} = v_i \sin(\phi_i)$) while heading direction, ϕ_i, and path velocity, v_i, are obtained from the corresponding behavioral dynamics. All dynamical equations are integrated with a forward Euler method with fixed time step, and sensory information is computed once per each cycle. The target information is defined by a goal position in space. We assume that the obstacles are shallow and hence do not interfere in the ability of a follower to acquire information about the distance and orientation to its leader.

5.1.1 *Establishing and moving the formations*

Two of the fundamental features that a team of robots should exhibit is the ability to establish a formation starting from random initial positions, and to move it from one location to another. In our approach, these features are implicit and treated simultaneously.

Figures 12 and 13 demonstrate the ability to establish and move different formations, starting from different positions, and challenged by the presence of obstacles, either static or dynamic (robots can be obstacles to other robots). The corresponding evolution of formation error is presented in Figs. 14 and 15. Formation errors show that the team converges to the desired geometric pattern, with a formation error that, although not always zero, is sufficiently small.

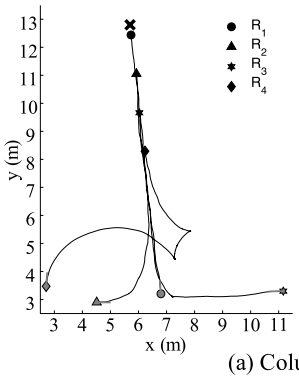
The same geometric configuration can be achieved by different inter-robot relations, represented by different formation matrices. As an example, we will focus now on a hexagon formation. Both matrices $\mathbf{S}_{\text{hex1}}$ (in (37)) and $\mathbf{S}_{\text{hex2}}$ (in (38)) specify the same pattern—an hexagon ($\mathbf{S}_{\text{hexagon}}$ in (18) is another example). In $\mathbf{S}_{\text{hex1}}$ each robot follows the one immediately in front of it (or one that is near). In $\mathbf{S}_{\text{hex2}}$ all the robots follow the same robot, which is also the team leader.

$$\mathbf{S}_{\text{hex1}} = \begin{pmatrix} 0 & 0 & 150 \\ 1 & \pi/4 & 150 \\ 1 & -\pi/4 & 150 \\ 2 & 0 & 150 \\ 3 & 0 & 150 \\ 5 & \pi/4 & 150 \end{pmatrix}. \tag{37}$$

$$\mathbf{S}_{\text{hex2}} = \begin{pmatrix} 0 & 0 & 150 \\ 1 & \pi/4 & 150 \\ 1 & -\pi/4 & 150 \\ 1 & \pi/8 & 277 \\ 1 & -\pi/8 & 277 \\ 1 & 0 & 362 \end{pmatrix}. \tag{38}$$

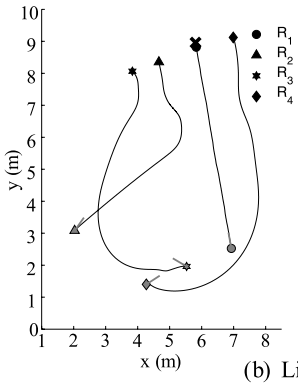
Figure 16 and Fig. 17 show the simulations of each situation, respectively. The analysis of position and formation errors is reported in Fig. 18 for $\mathbf{S}_{\text{hex1}}$ and in Fig. 19 for $\mathbf{S}_{\text{hex2}}$.

Although specifying the same geometric pattern, the path traveled by each robot can be different depending on the leader-follower relations, i.e. on the formation matrices. Finding out which formation matrix is best suited for each task is not trivial, and is out of the scope of this paper. The purpose is to highlight the importance that a proper matrix design has in the overall formation performance. Using the first matrix, $\mathbf{S}_{\text{hex1}}$, and analysing the X–Y plot in Fig. 16, it looks like the formation approaches an hexagon shape faster than when using $\mathbf{S}_{\text{hex2}}$ (compare the evolution of robot R_6, which is the one that closes the formation). It is possible to confirm this impression by comparing the respective formation errors, in Figs. 18 and 19. The average formation error



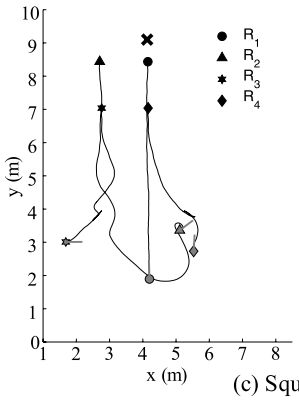
$$\mathbf{S}_{\text{column}} = \begin{pmatrix} 0 & 0 & 100 \\ 1 & 0 & 140 \\ 2 & 0 & 140 \\ 3 & 0 & 140 \end{pmatrix}$$

(a) Column



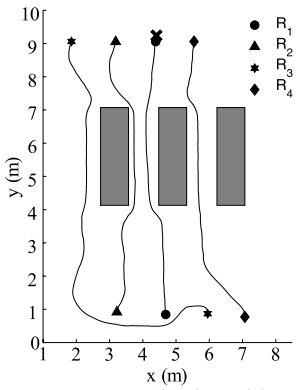
$$\mathbf{S}_{\text{line}} = \begin{pmatrix} 0 & 0 & 100 \\ 1 & \pi/2 & 120 \\ 2 & \pi/2 & 120 \\ 1 & -\pi/2 & 120 \end{pmatrix}$$

(b) Line



$$\mathbf{S}_{\text{square}} = \begin{pmatrix} 0 & 0 & 100 \\ 1 & \pi/2 & 140 \\ 3 & \pi/2 & 140 \\ 1 & 0 & 140 \end{pmatrix}$$

(c) Square



$$\mathbf{S}_{\text{line}} = \begin{pmatrix} 0 & 0 & 100 \\ 1 & \pi/2 & 120 \\ 2 & \pi/2 & 120 \\ 1 & -\pi/2 & 120 \end{pmatrix}$$

(d) Line with static obstacles

Fig. 12 Examples of multi-robot teams establishing formations. The *lead robot*, depicted by a *circle*, heads to its target (*cross mark*). The robots in the team are placed initially in random locations and, as they start to move, stabilize the desired formation

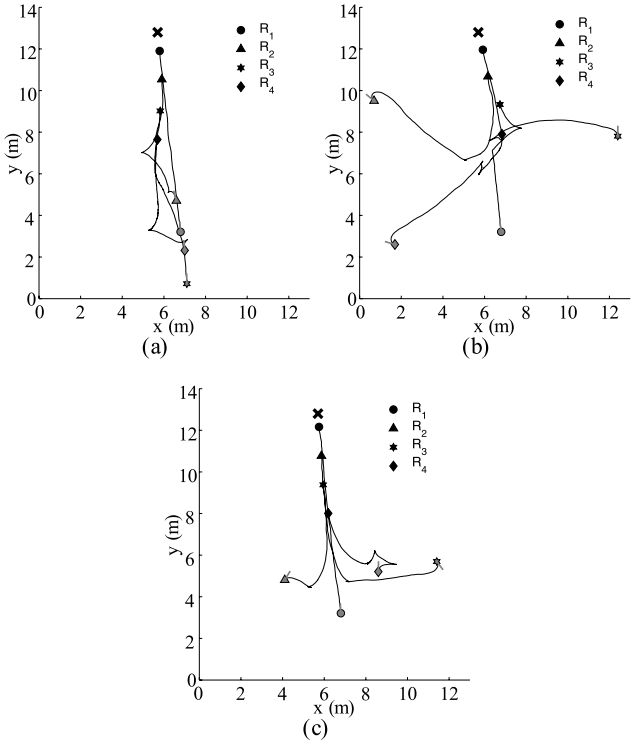


Fig. 13 Other initial locations, but establishing the formation in Fig. 12(a)

decays faster in the first situation than in the second. A closer look, specifically to each robot error, reveals that robots that are more distant to their leader's tend to have higher (and lasting) formation error.

Figure 20 shows the performance of robot R_6, in the first situation in terms of controller stability. It is possible to see that the system is well tuned, because it is able to follow the attractor evolution very closely (ϕ_i, the heading direction, is always near *Attractor 1*). When a bifurcation occurs (in this case, the appearance of a second attractor, when path obstructions are sensed) it is also fast in the decision making process of which attractor to follow.

5.1.2 *Two formations in a collision route*

Here we present an experiment where two teams, of three robots each, travel in opposite directions and in collision route. Near halfway the destination the two formations will encounter and avoid each other in order to proceed to their targets. A plot of the travelled path of each robot in the experiment is shown in Fig. 21. The formation matrix of both teams is

$$\mathbf{S}_{\text{triangle}} = \begin{pmatrix} 0 & 0 & 20 \\ 1 & \pi/4 & 150 \\ 1 & -\pi/4 & 150 \end{pmatrix}. \tag{39}$$

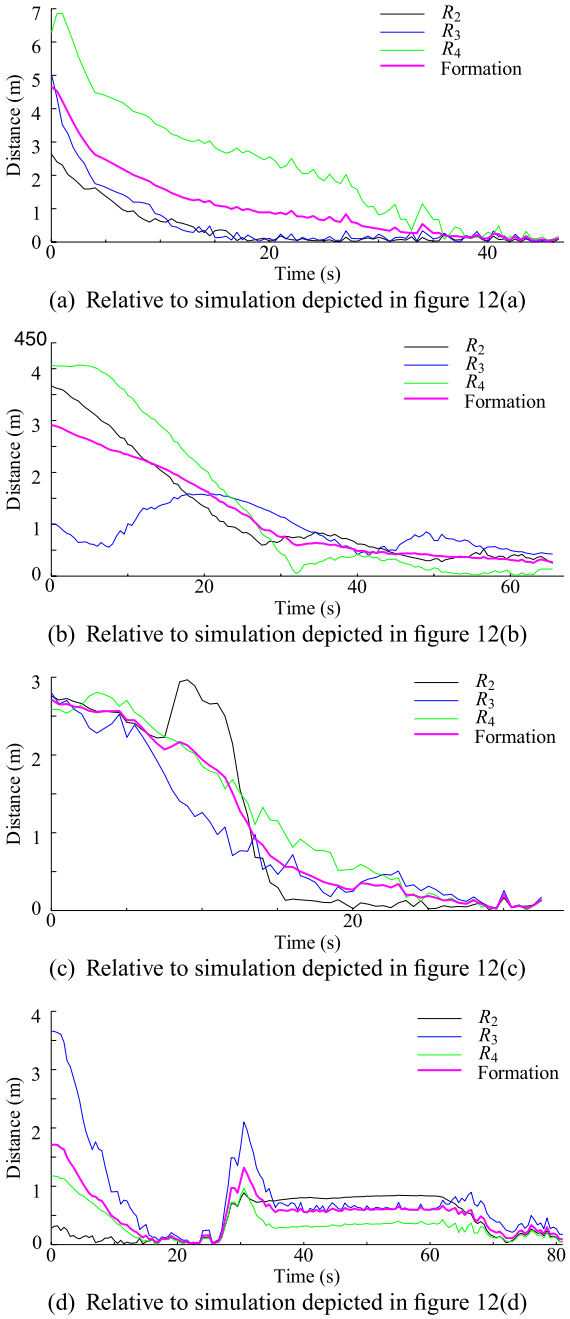


Fig. 14 Position error for each robot and formation error (*bold line*) for the examples presented in Fig. 12

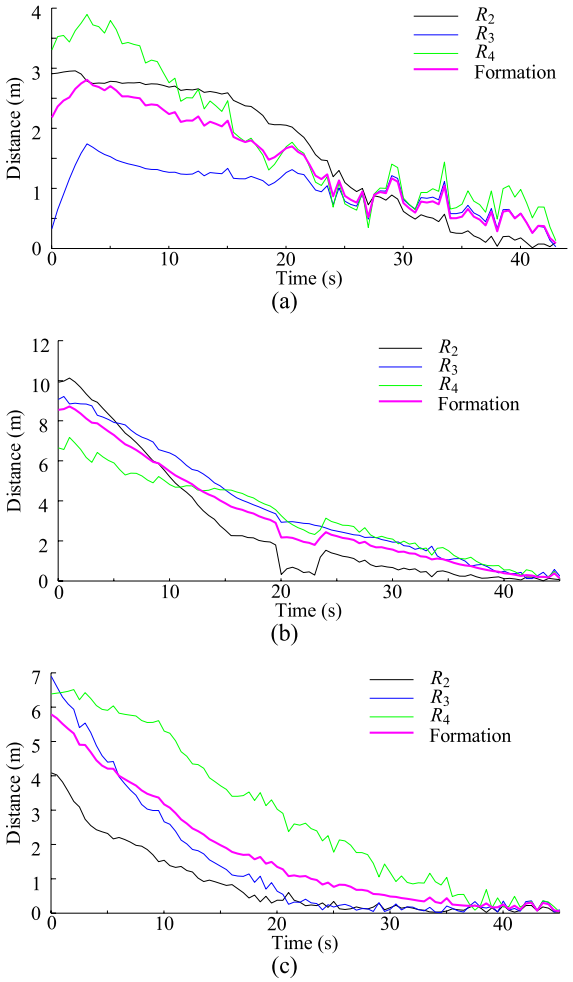


Fig. 15 Formation error for the examples presented in Fig. 13

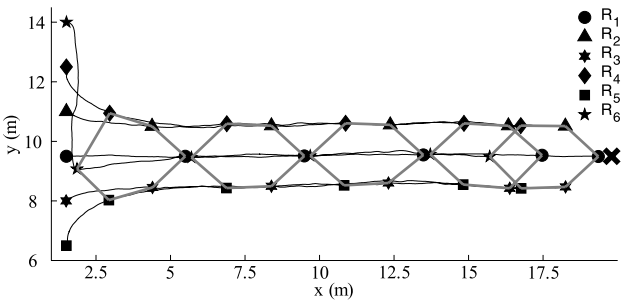


Fig. 16 Six robots establishing and moving a hexagon formation according to the hierarchical relations defined in $\mathbf{S}_{\text{hex1}}$

This example illustrates the ability to avoid dynamic obstacles, as the robots in one team serve as obstacles to the robots in the other.

In Fig. 22 the results of an analysis of the formation error evolution is plotted. In these plots it is possible to see that the robots are not in formation at start. Establishing the formation starts right at the beginning. At $t \approx 12$ s team leaders become in each other range of detection and start an avoidance maneuver (you can see this by an increase in the *followers* (robot 2) distance error. At $t \approx 16$ s the *followers* meet and have to avoid other team's *followers*. This is more dramatic for robots R_2 and R_6, that are the ones that get a larger error. It is also, clearly seen in Fig. 23, in the case of robot R_2, that due to this obstacle (robot R_5) a pitchfork bifurcation in the dynamics appears. The number of attractors in the system changed to two which reflect the two possibilities (i.e. either by the left or the right) to avoid collision with

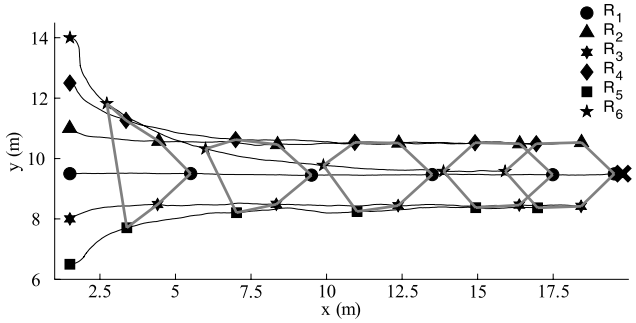


Fig. 17 Six robots establishing and moving a hexagon formation according to the hierarchical relations defined in $\mathbf{S}_{\text{hex2}}$

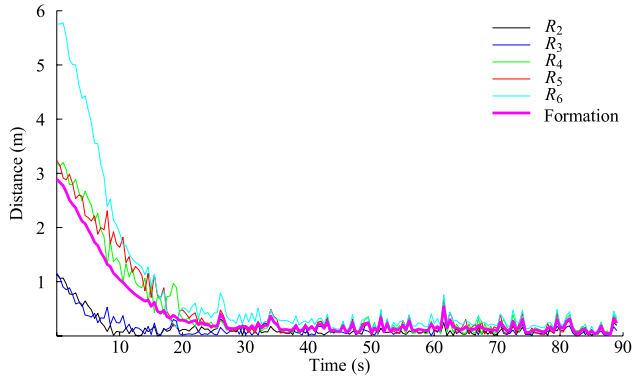


Fig. 18 Position error for each robot and formation error for the simulation presented in Fig. 16

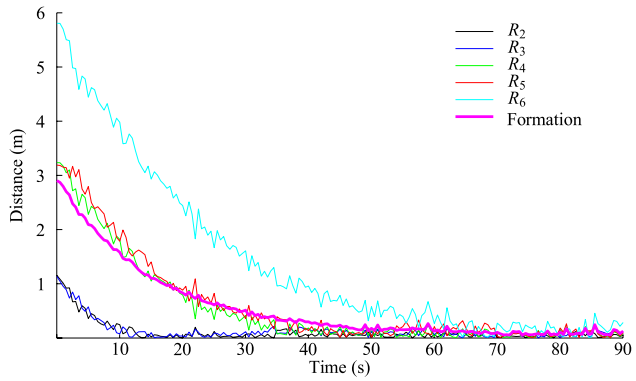


Fig. 19 Position error for each robot and formation error for the simulation presented in Fig. 17

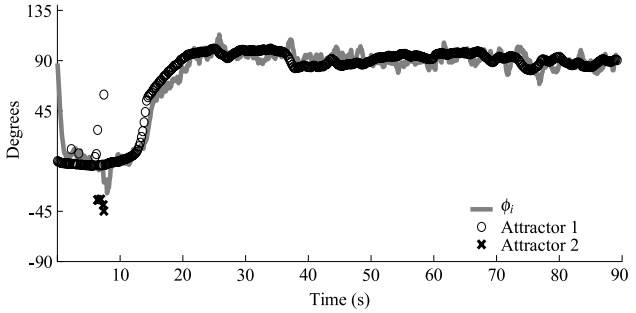


Fig. 20 Stable fixed point evolution of robot R_6, performing the formation defined in $\mathbf{S}_{\text{Hex1}}$. *Attractor 1* and *Attractor 2* are the stable fixed points. ϕ_i is the current heading direction. As can be seen ϕ_i is following one of the attractors very closely

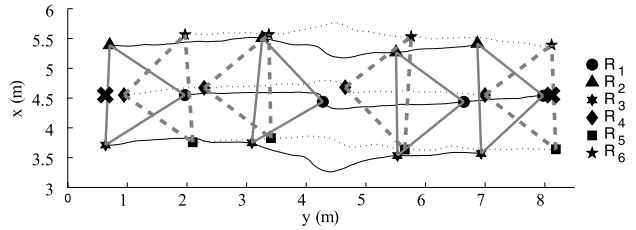


Fig. 21 A simulation with two formations heading in opposite directions and in collision route. Team A is composed by robots R_1, R_2 and R_3. R_1 is the *Lead robot* and moves from left to right, toward the target (*cross at the right*). The trajectory of each robot is depicted by the *thin solid line*. Team B is composed by robots R_4, R_5 and R_6. R_4 is the *lead robot* and moves from right to left, toward the target (*cross at the left*). The trajectory of each robot is depicted by the *thin dotted line*

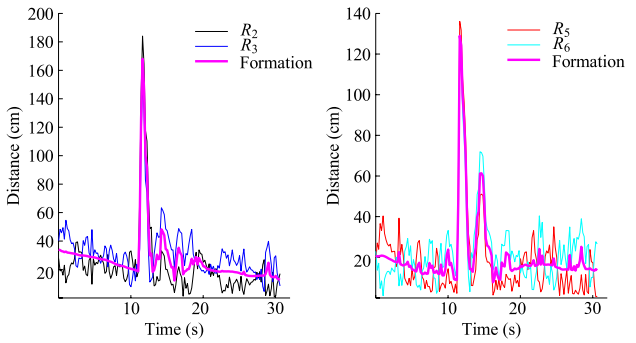


Fig. 22 Formation error analysis for the simulation in Fig. 21. (*Left plot*) Team that starts on the left, composed by R_1, R_2 and R_3; (*Right plot*) Team that starts on the left, composed by R_4, R_5 and R_6

robot R_5. Later, the obstacles are overtaken and the formation is stabilized again.

5.1.3 Navigating in a cluttered environment

The ability to maintain a formation when crossing an environment with a high density of obstacles, is a fundamental one in a framework targeted to exploration tasks. Figure 24 shows such an environment. There, a team of robots, organized according to formation matrix $\mathbf{S}_{\text{hex1}}$ (in (37)), starts on the left and aims to travel to the target on the right (marked with an **X**). In their way they find a collection of obstacles (blue rectangles and squares) which they have to traverse. As can be seen in the figure, the robots avoid collisions with obstacles, and each robot as soon as no obstruction is sensed returns to the formation. What is to emphasize here is the ability to maneuver in these environments without an explicit coordination scheme controlling these formation changes. This smooth behaviour switch is one of the key-benefits of

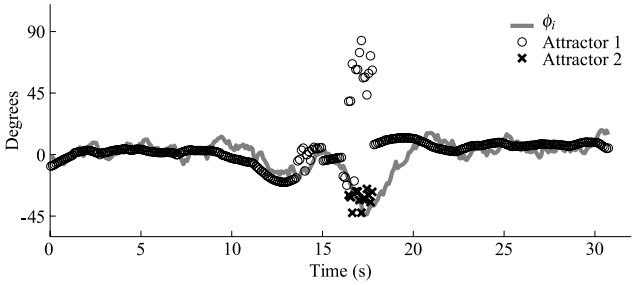


Fig. 23 Evolution of the stable fixed points (attractors) over time for robot 2, in simulation depicted in (21). It is possible to see that the system is always near an attractor and when bifurcations occur, it gets stable again (near one attractor) rather quickly

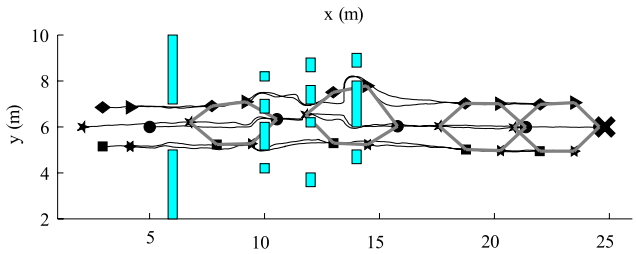


Fig. 24 Six robots navigating in a cluttered environment, while trying to preserve an hexagon formation. To make the figure clear, the robots are not represented to scale (they appear larger than the reality). The team starts on the left and heads to the target on the right. The obstacles are organized in four barriers. While avoiding obstructions the team looses formation, but it stabilizes it as soon as the obstacles are overtaken

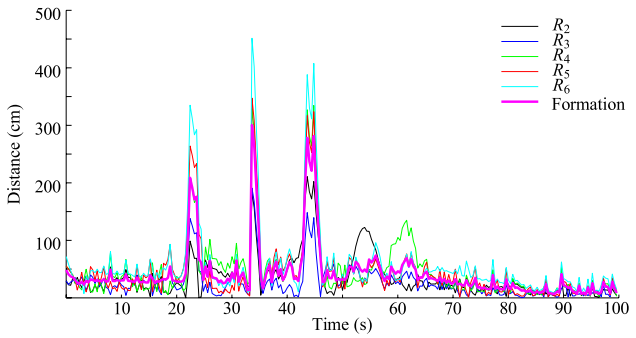
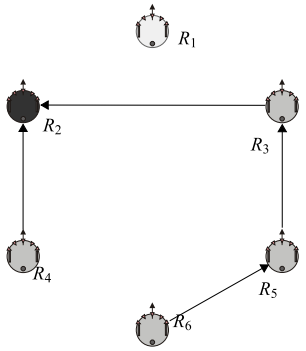


Fig. 25 Position error for each robot and formation error for the simulation of six robots navigating in a cluttered environment. The three large spikes correspond to the traversal of the three rightmost barriers of obstacles (see Fig. 24), where it is not possible to keep the correct formation configuration

the attractor dynamics approach, that on the present framework can be understood as an implicit switch-and-join of formations. It is possible to confirm the good performance of the formation in Fig. 25. There, one can see an usually low error, except for three large spikes that correspond to the traversal of the three rightmost obstacle barriers. As soon as each obstacle barrier is overtaken the formation stabilizes again and the error decreases to acceptable values.

Fig. 26 A representation of the formation defined by $\mathbf{S}_{\text{hex3}}$ in (40). R_1 had a failure and leadership was given to R_2



5.1.4 Re-establishing a formation after leader failure

Here, we test how the presented architecture reacts to a failure in one of the robots. To make the situation harder, the robot that will fail is the *lead robot*. Unless the robot that fails is an ending robot, i.e. a robot that does not lead any other robot (is at the end of the chain), it is mandatory an update to the formation matrix, such that all the inter-robot relations are updated accordingly. Currently, this update is not performed automatically.

Two scenarios are considered when updating the formation matrix. If, for instance, pursuing an hexagon formation and the team leader fails, then, the leadership is assigned to another robot. The subsequent options include either the remaining robots continue with the same shape formation (but with the place occupied by the failing robot empty) or a formation shape switch is commanded. How to proceed in this situation is dependent on the mission objectives, thus is not treated here, but see Monteiro and Bicho (2008).

In the first scenario, only the leadership is changed. The team of six robots starts in a line configuration and stabilize an hexagon formation, represented in $\mathbf{S}_{\text{hex1}}$ (37), during the first 40 seconds (see Fig. 27). At that time the leader fails and a new formation matrix is assigned:

$$\mathbf{S}_{\text{hex3}} = \begin{pmatrix} - & - & - \\ 0 & 0 & 150 \\ 2 & -\pi/2 & 212 \\ 2 & 0 & 150 \\ 3 & 0 & 150 \\ 4 & \pi/4 & 150 \end{pmatrix} \tag{40}$$

which maintains the shape of the "hexagon", except that the location occupied by the failing robot is empty (see Fig. 26).

If the formation matrix is updated as soon as the robot fails the formation reacts quite well. Actually, looking only at the error diagram (see Fig. 28) it is almost impossible to distinguish when the formation switch happens.

In the second scenario the team starts in a line configuration and stabilizes the formation $\mathbf{S}_{\text{hex1}}$. Then the *Lead robot*

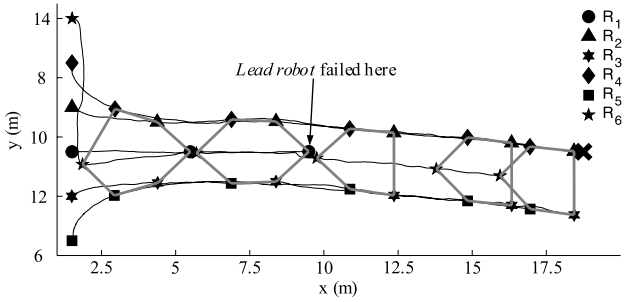


Fig. 27 Path travelled by six robots in an hexagon formation. They start in a line configuration, but around 40 s after their start (see Fig. 28), the *Lead robot* fails. Then a new *Lead robot* is assigned (R_2), and the formation matrix is updated accordingly

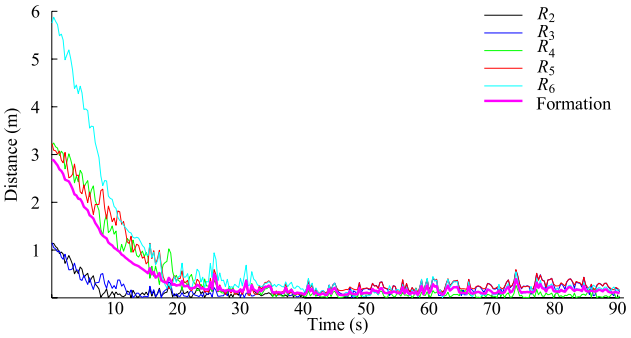


Fig. 28 Position error for each robot and formation error for the simulation presented in Fig. 27. The robots take about 15 s to 20 s to establish a good formation. The *Lead robot* fails at time 40 s. Because the formation shape is not changed after the *Lead robot* fails, the formation remains stable throughout the experiment

stops due to some malfunction. After *Lead robot* failure detection, a new formation matrix, defining a new formation pattern, is given to the team

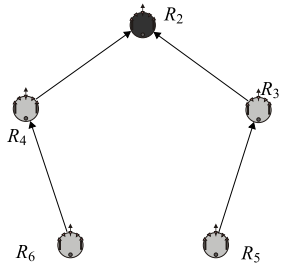
$$\mathbf{S}_{\text{penta}} = \begin{pmatrix} - & - & - \\ 0 & 0 & 150 \\ 2 & -3\pi/10 & 176 \\ 2 & 3\pi/10 & 176 \\ 3 & \pi/10 & 176 \\ 4 & -\pi/10 & 176 \end{pmatrix} \tag{41}$$

which is represented in Fig. 29 (with the travelled path in Fig. 30). Here, since there is also a change in the shape of the formation, a higher error is observed at that moment (see Fig. 31).

5.2 Implementations in Khepera robots

This control architecture has been implemented first in a team of three Khepera I robots. These are small sized robots (about 6 cm diameter) equipped with six infra-red distance sensors (from 2 to 5.5 cm range) and have as processing unit a Motorola 68000. This processing unit is responsible for making all computations necessary to determine

Fig. 29 A representation of the formation defined by $\mathbf{S}_{\text{penta}}$ in (41)



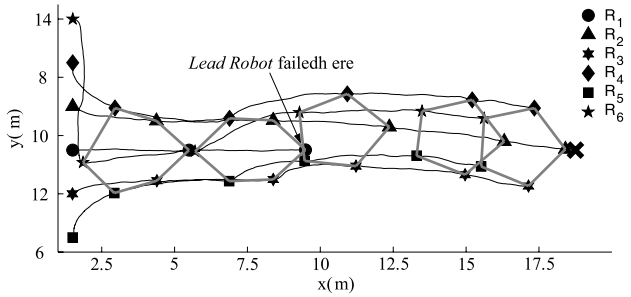


Fig. 30 Path traveled by six robots in an hexagon formation. After *Lead robot* failure, the remaining robots switch to a pentagon formation, with a new *Lead robot*

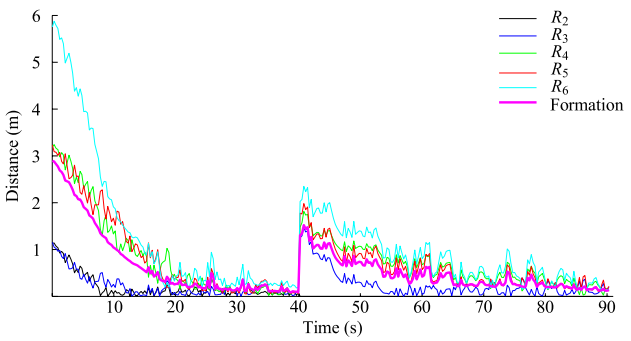


Fig. 31 Position error for each robot and formation error for the simulation presented in Fig. 30. By loosing R_1, a formation switch is commanded at $t = 40$ s, thus causing an increase in the error at that time

its (the Khepera) behaviour. There is no off-board processing. In these experiments one external computer was used to centralize the information regarding the formation, making the interface with an user more easy. Its purpose was to allow a user to input the desired geometric formation, construct the corresponding formation matrix, and then communicate to each robot its desired pose within the formation. While actually running the experiments there was no intervention from this computer. Since our Kheperas are not equipped with vision they can't locate their team-mates and have to rely on communicated information, thus using a global coordinate system. As a turnaround to not being able to detect other robots (except via IR and treated as obstructions), when the starting order is given, the team leader starts, then, to broadcast to its followers its actual position. We use a lower-layer that provides the behavioral dynamics

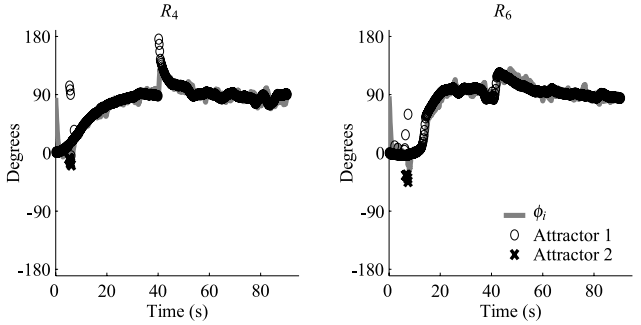


Fig. 32 A plot of the evolution of the stable fixed points (attractors) for some of the robots. It is possible to see that the architecture is quite stable, and even when formation switches occur, internally, it also stabilizes fast

with estimates of distance, $l_{i,j} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, and direction to the follower, $\psi_{i,j} = \text{atan2}(y_j - y_i, x_j - x_i)$. Where (x_i, y_i) and (x_j, y_j) are the robot Cartesian coordinates, of the follower and leader respectively, updated by a dead-reckoning rule ($\dot{x_i} = v_i \cos(\phi_i)$ and $\dot{y_i} = v_i \sin(\phi_i)$) and with the leader ($j$ index) communicating its position to the follower (i index). Heading direction, ϕ_i, and path velocity, v_i, are obtained from the corresponding behavioural dynamics. All dynamical equations are integrated with a forward Euler method with time step equal to the actual computation time. Sensory information and leader's position are updated once per each cycle. The target information is defined by a goal position in space (i.e. (x_{tar}, y_{tar})) using the global coordinate system.

Computation time per each cycle is greatly dependent on the desired formation that the robot is performing. Thus, if one robot is performing a column formation, in these robots, its computation time is typically between 40 ms and 50 ms per cycle. This time increases to values between 65 ms to 85 ms in the cases of either oblique or line formation. This means that, in principle, when doing column formation the observed results, in terms of dead reckoning, should be more precise than for the other two formation behaviours. It also means that, although the parameters are tuned such that the relaxation rates are adapted automatically as a function of the computation cycle, it is hard to find a compromise for these parameters because of the range that the computation cycle can take. The only cause of this iteration time variability is the use of trigonometric functions and exponentials in a processor that does not support them. Thus, they have to be implemented by software, with higher times per operation. In implementations in robots equipped with processors with more computing power, this problem is negligible. We prove this with the implementations in our larger robots, where the cycle time is almost invariable (cf. Sect. 5.3).

In the next subsections we show two of the conducted experiments with Khepera robots and an analysis of the results.

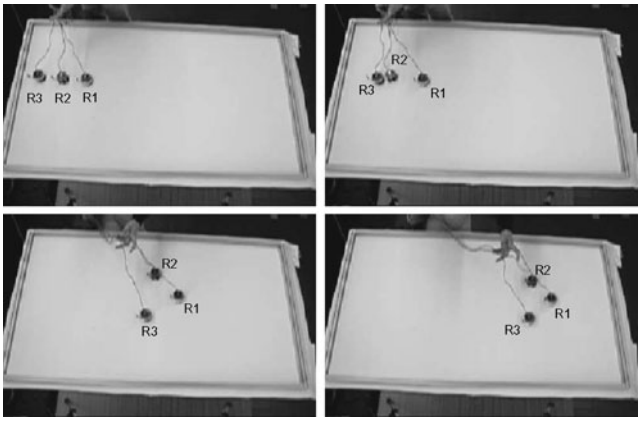


Fig. 33 Video snapshots of three Kheperas switching from a column to a triangle formation. *Up left*: shows the robots starting position, which is in column with 150 mm separation from each other. *Up right*: at $t = 2$ s the leader, robot R_1 is moving toward the goal and the followers try to position themselves. Because robot R_3 is moving faster, almost hits R_2. *Down left*: at $t = 16$ s the team is almost in formation, only the distances are slightly larger than desired. *Down right*: at $t = 19$ s the team is now in formation

5.2.1 *Three robots switching from a column formation to a triangle formation*

Figure 33 shows a sequence of video snapshots of three Kheperas switching from a column to a triangle formation. This can be seen in the upper left panel. This example serves to show the implicit ability to stabilize a desired formation. They start in a line formation with the formation matrix given by

$$\mathbf{S}_{\text{column}} = \begin{pmatrix} 0 & 0 & 20 \\ 1 & 0 & 125 \\ 1 & 0 & 125 \end{pmatrix}. \tag{42}$$

Upon start the formation matrix is changed to $\mathbf{S}_{\text{triangle}}$ in (43) (with distances in mm).

$$\mathbf{S}_{\text{triangle}} = \begin{pmatrix} 0 & 0 & 20 \\ 1 & \pi/4 & 150 \\ 1 & -\pi/4 & 150 \end{pmatrix}. \tag{43}$$

In Fig. 34 the results of an analysis of the formation error evolution is plotted. This plot is shown for the two followers. As expected, as time evolves, the position error comes closer to zero, meaning that the robots are closer to the desired formation.

Robot R_2 attractors evolution is depicted in Fig. 35. There, we can see that this robot over-turned between $t \approx 6$ s and $t \approx 14$ s, because the actual heading direction is larger than the desired one (current stable fixed point). This is main-ly caused by high variance in the cycle-time that our implementation in these robots have. For larger cycle times, the relaxation rates decrease (see Appendix B), thus the con-

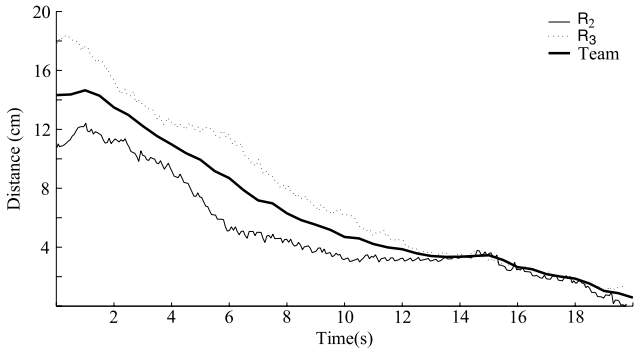


Fig. 34 Position and formation error analysis for the experiment with three Kheperas switching from column to triangle formation

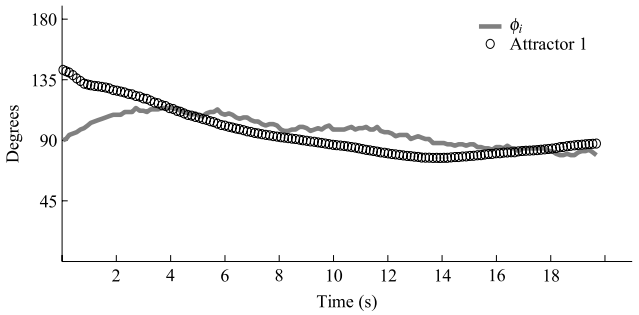


Fig. 35 Evolution of the stable fixed points, for robot R_2. At all times, only one attractor exists

trol variable takes longer to converge to the attractor state as it moves.

5.2.2 *Three robots in a line formation*

Figure 36 shows a sequence of video snapshots of three Kheperas driving in a line formation and avoiding an obstacle. The formation matrix is given by

$$\mathbf{S}_{\text{line}} = \begin{pmatrix} 0 & 0 & 20 \\ 1 & \pi/2 & 150 \\ 1 & -\pi/2 & 150 \end{pmatrix}. \tag{44}$$

In Fig. 37 the results of an analysis of the formation evolution are plotted. Here only robot R_3 senses the obstacle. This example illustrates the ability of the control architecture to integrate other behaviours besides formation control (in this case, obstacle avoidance). Regarding the formation control behaviour, it exhibits its implicit capacity to split and join formations. Split formation starts when obstacles are detected. At that time another vector field has to be accounted for: the one regarding obstacle avoidance. This vector field places an unstable fixed point at the direction where the obstacle lies, causing this way the robot to go around the obstacle. When the obstacle is overtaken, the resultant vector field is only governed by the formation behaviour,

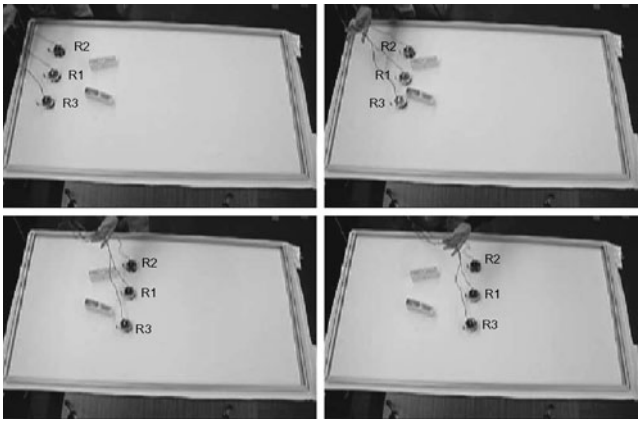


Fig. 36 Video snapshots of three Kheperas moving in line formation. *Up left*: shows the robots starting position. The *Lead robot* is robot R_1. *Up right*: at $t = 4$ s the robots approach the obstacles. Robot R_3 does not have space to pass without leaving formation, thus will have to avoid the obstacle. *Down left*: at $t = 10$ s, after overtaking the obstacle the robot R_3 starts to rejoin the formation. *Down right*: at $t = 18$ s the robots are again almost in formation

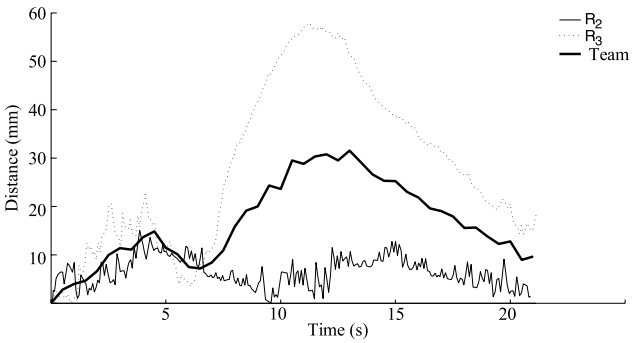


Fig. 37 Error analysis for the experiment with three Kheperas stabilizing a line formation, with obstacles

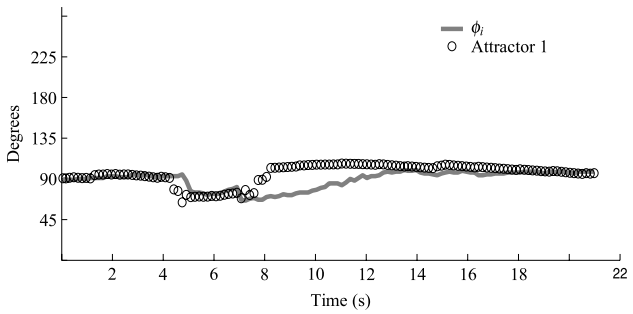


Fig. 38 Evolution of the stable fixed points of robot R_3. At all times, only one attractor exists

causing the formation to stabilize to the one specified by the formation matrix, i.e. a formation join is performed.

Figure 38 shows the attractor evolution for robot R_3. For the same reasons of the previous experiment, the robot dynamical system takes longer to stabilize to the attractor, although it is able to successfully maneuver around the box.

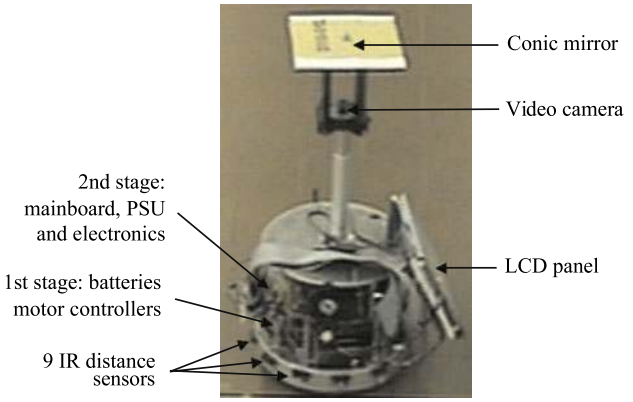


Fig. 39 The *Gusmao* robot

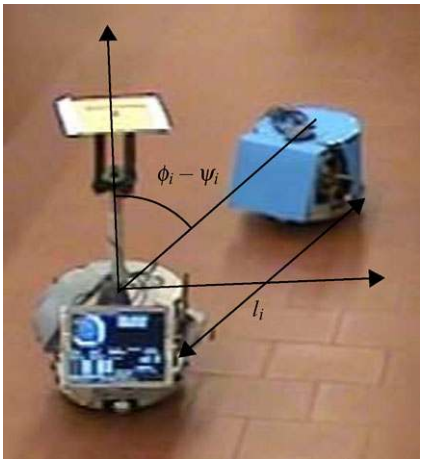


Fig. 40 A photo of the two robots in formation. The robot, in the *top-right corner* of the photo, is the *leader*. The other robot is the *follower*. The *follower* estimates the distance, l_i, and relative direction to the *leader*, $\phi_i - \psi_i$ using the vision system

5.3 Implementations in robots equipped with vision

Here we present some preliminary tests that confirm the good results observed in simulation and in previous Khepera implementations.

The robots used in these experiments were developed in-house (see Fig. 39). They are differential drive robots, kinematically equivalent to the Kheperas, with 30 cm diameter and height. On top, each robot has an omni-directional vision system (AISVision from the Fraunhoffer Institute), that allows it to locate other robots in its neighbourhood (as used by Das et al. 2002; Vidal et al. 2003). It also has 9 distance to obstacle sensors, based on the infra-red principle (Sharp GP2D12). In terms of computing, it is equipped with a single-board computer in a *mini-itx* form factor, with a 1 GHz Intel compatible (VIA C3) processor. The image processing was based on color blob extraction. The size and location of the blob, followed by an ad-hoc calibration, allowed us to compute an estimate of the distance and relative direction of the leader. All this in real time (up to 26 frames/second) and with a maximum error of 5% in direction and 10% in distance in our arena.

Since we only have two robots available, we will show only two robots formations, (see Fig. 40). Here no dead-reckoning, nor explicit communication are used. The output of the vision system is an estimate of the distance between the *follower* and its *leader*, l_i, and the angle $\phi_i - \psi_i$, that can be directly used in the equations that shape the heading direction dynamics vector field ((3) for column and (7) for line and oblique). Since the *follower* locates its team mate by using a vision system, no global reference frame is necessary, thus no dead-reckoning is also necessary. These are two major benefits over the Khepera implementations (or other implementations based on dead-reckoning), where errors in the integration of the coordinates and heading direction cause an increasing degradation of the observed results with time. Here, the only error is introduced by the estimation method, associated with the vision system, but is constant all the time.

We present the results of two experiments: the first with a column formation, and the other with an oblique formation. Since no communication is used, the follower does not know its leader velocity, v_j, which is necessary in (4). One way to overcome this problem is to consider v_j as the desired cruise velocity of the leader, which can be set before the experiment and is known to all robots.

In the first experiment (see Fig. 41), the desired configuration is a column formation at 100 cm ($l_{i,d} = 100$ cm and $\Delta\psi_{i,d} = 0$ deg). The robots start side-by-side, in a line configuration. They are inside a square area with walls. The *leader* moves in front while avoiding collisions with the walls. This translates into a semi-circular path of the *leader*.

Figure 42 shows the formation error, as defined by (35). When the *follower* starts, it quickly reduces the formation error. Near the 18th sec the estimated error is between 10 cm and 20 cm. At this time the *leader* turns to avoid a wall, thus causing an increase in the formation error. The *follower* takes about 15 secs to stabilize the formation again. The second large spike in the formation error, is, again, caused by an abrupt alteration in the *leader's* heading when avoiding a wall. At the 70th sec, the *leader* is stopped. The *follower* does not stop completely, but moves back and forward in the close vicinity of the desired distance. During the time the leader is stopped the formation error variates slightly, but does not increase, being nearly stationary. As soon as the *leader* is restarted the *follower* also restarts (remember that there is no explicit communication between the robots). Near the 100th sec, the *leader* is caught in a corner, and turns back. To accommodate the formation, the *follower* starts moving backwards. 20 secs later the *follower* can start moving forward and, again, stabilizes the formation In terms of dynamical system stability, represented in Fig. 43, the sys-

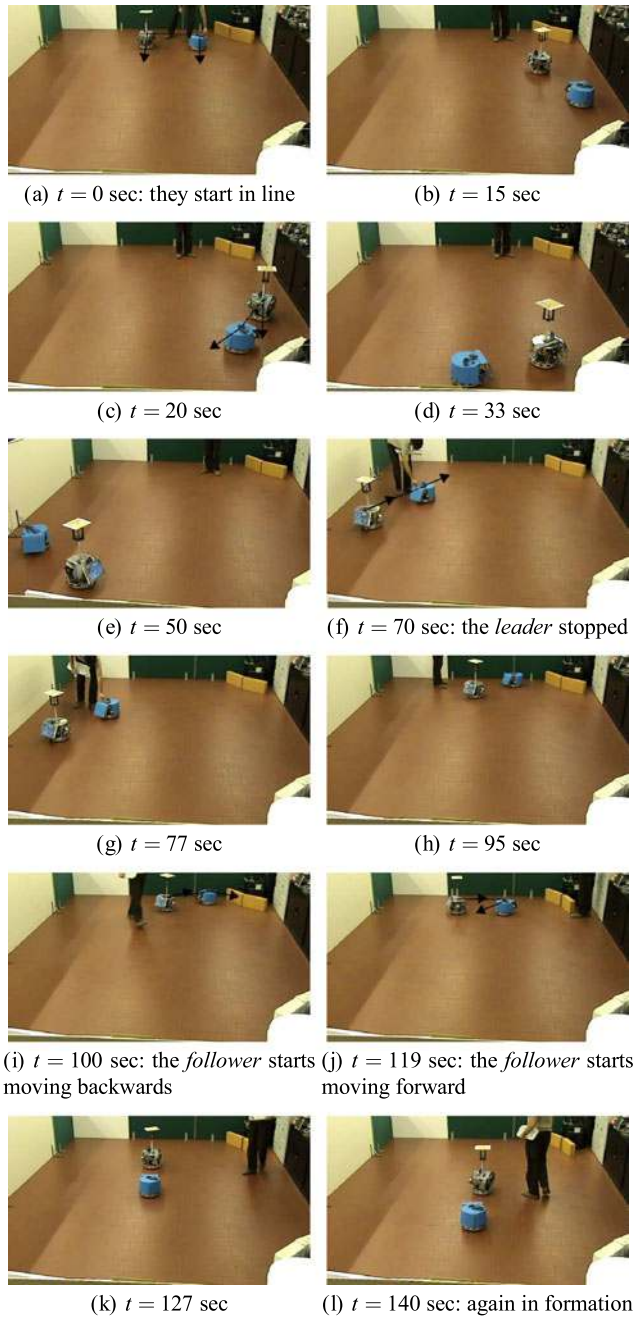


Fig. 41 (Color online) Snapshots of the column formation stabilization experiment (parameters are $\Delta\psi_{i,d} = 0$ deg and $l_{i,d} = 100$ cm). The blue robot is the leader. Arrows indicating each robot's heading direction were added to the snapshots, for clarity purposes

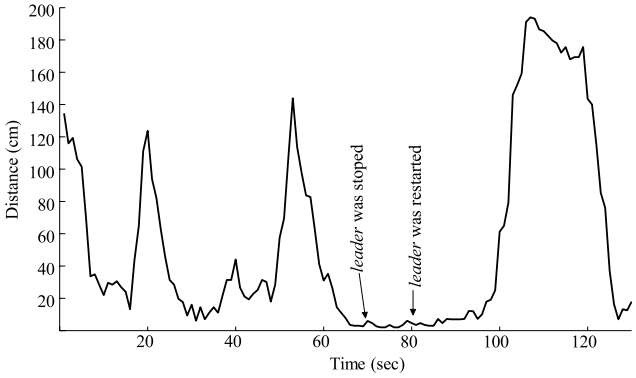


Fig. 42 Position error evolution of a team of two robots navigating in a column formation (with snapshots in Fig. 41). Approximately at $t = 100$ sec the *leader* approaches a corner, turns and heads in the *follower's* direction. With the purpose of keeping the inter-robot distance, the *follower* moves backwards (until, approximately, $t = 119$ sec). During this time, the *leader* and *follower* are pointing in (almost) opposite directions, which causes a large position error

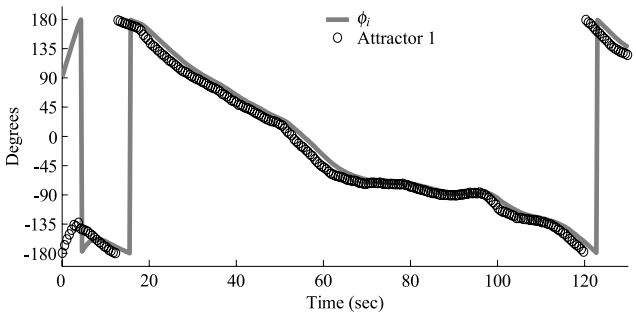


Fig. 43 Fixed point evolution of the *follower* robot, when stabilizing a column formation. The environment is obstacle free, except for the walls

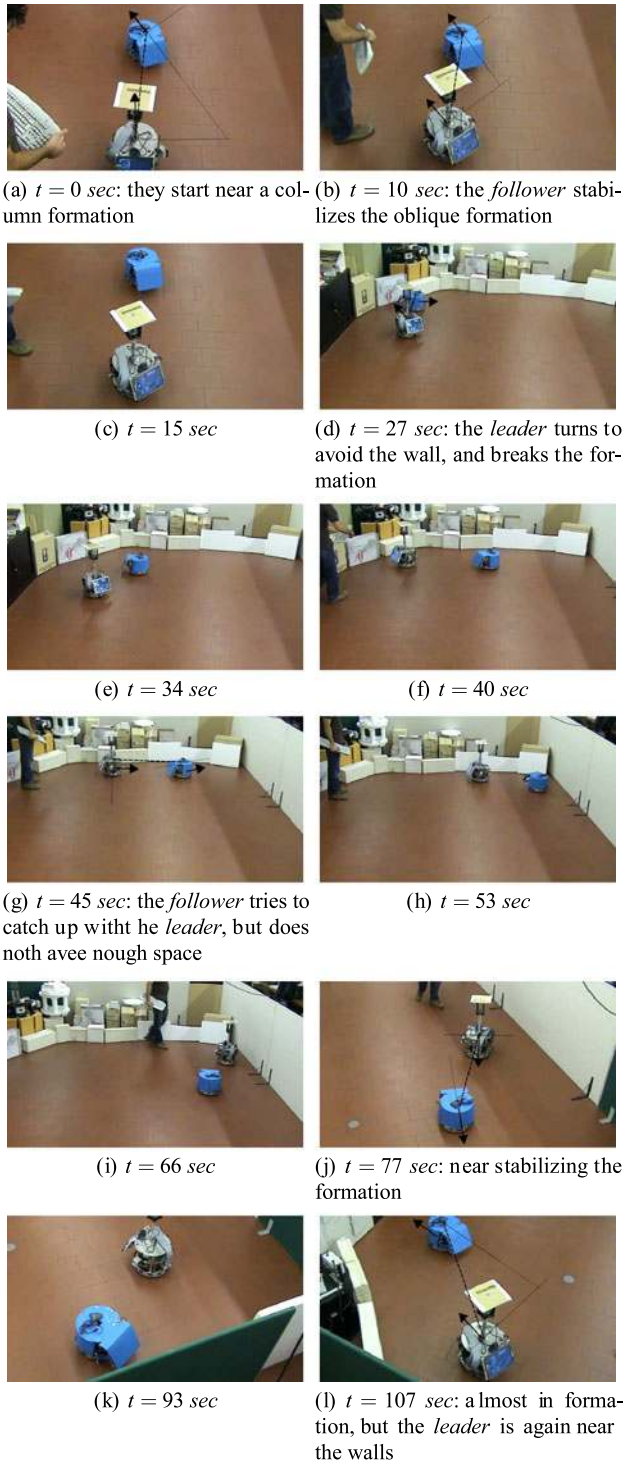
tem is well tuned, because heading direction is able to track closely the moving attractor of the dynamics.

In the second experiment, the *follower* has to stabilize an oblique formation, at 45 degrees and 100 cm from the *leader* (see Fig. 44). They start out of formation and the follower is initiated first. During the first 10 seconds of the experiment, the *leader* remains stopped in the same place. During this time, the follower moves in order to stabilize the desired configuration. When the formation leader starts to move, the formation error increases, because the follower takes some time to catch up with the leader. Near the 20th sec, the formation error already stopped to increase, but then, the *leader* was faced with a wall and had to turn, thus breaking the formation, and increasing even further the error. The same happens at the 50th and 90th sec (approximately). The control system shows to be stable, as depicted in Fig. 46. The second attractor in the dynamics appears when the *follower* also detects the walls.

6 Discussion

We have shown how non-linear attractor dynamics can be used as a framework to generate controllers that allow a team of N-robots to navigate according to a prescribed geometric formation while doing obstacle avoidance. The environment is not known a-priori and it can change over time.

Three controllers have been developed for a *leader–follower strategy*. These allow one robot (*follower*) to maintain



(a) $t = 0$ *sec*: they start near a column formation

(b) $t = 10$ *sec*: the *follower* stabilizes the oblique formation

(c) $t = 15$ *sec*

(d) $t = 27$ *sec*: the *leader* turns to avoid the wall, and breaks the formation

(e) $t = 34$ *sec*

(f) $t = 40$ *sec*

(g) $t = 45$ *sec*: the *follower* tries to catch up witht he *leader*, but does noth avee nough space

(h) $t = 53$ *sec*

(i) $t = 66$ *sec*

(j) $t = 77$ *sec*: near stabilizing the formation

(k) $t = 93$ *sec*

(l) $t = 107$ *sec*: almost in formation, but the *leader* is again near the walls

Fig. 44 (Color online) Snapshots of the oblique formation stabilization experiment (parameters are $\Delta\psi_{i,d} = 45$ deg and $l_{i,d} = 100$ cm). The blue robot is the leader. Arrows were added to each snapshot to indicate each robot's heading direction

a desired pose regarding another robot (*leader*). They allow three different particular situations: column formations (*follower* behind the *leader*), line formations (*follower* side-by-side with the *leader*) and oblique formations (otherwise).

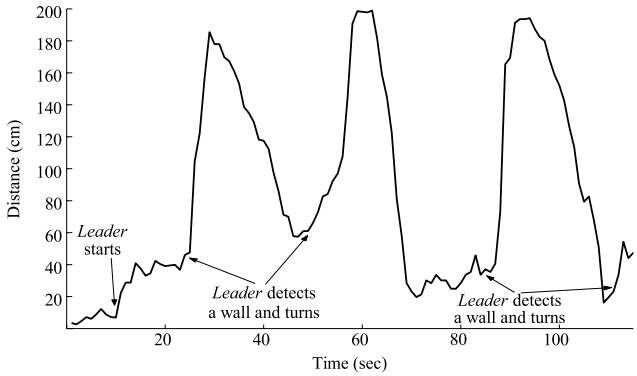


Fig. 45 Position error evolution when a *follower* navigates in an oblique formation with its *leader* (with snapshots in Fig. 44). During the first 10 seconds the *leader* remains in the same spot. Between $t = 27$ and $t = 32$ sec the follower moves backwards, because the distance to the *leader* is smaller than the desired one (this was caused by the *leader's* wall avoidance maneuver, that caused it to approach the *follower*)

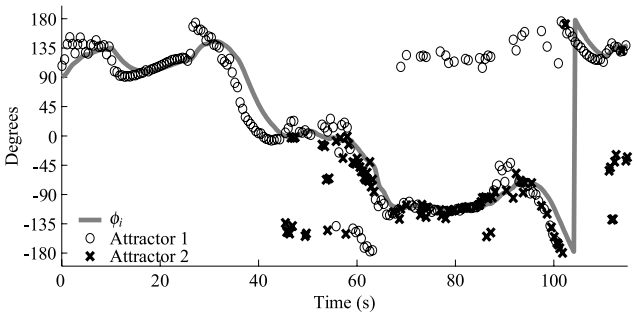


Fig. 46 Fixed point evolution of the *follower* robot when stabilizing an oblique formation (with snapshots in Fig. 44). During the first 10 seconds the *leader* remains in the same spot

The formation pattern is aggregated in a matrix, the *formation matrix*, that contains the information for each robot in the team: its *leader*, the distance and orientation to it. By manipulating this matrix it is possible to change the desired formation.

Perhaps the most closely related work to ours, in terms of team organization, is the one reported in Desai et al. (2001). They also use leader–follower strategy, but with two types of controllers (feedback controllers), that control either the position and orientation of the robot to a leader, or the position relative to two robots. In terms of team structure they use the concepts of transition matrix and control graphs, explicitly switching formations in the presence of obstacles. In our approach, the formations are flexible in the presence of obstacles, i.e., the formation will adapt itself and maneuver between the obstacles without explicitly switching formations (i.e. without changing the formation matrix). Another fundamental difference is that while they specify the formation under the leader's reference frame, and taking in consideration the leader's heading, we do so under the fol-

lower's reference frame, without using the leader's heading (which is difficult to estimate based on on-board sensors) in our controllers specification.

Because we assume the *lead robot's* heading direction is not known at runtime we cannot compute the formation error while running the experiments. Thus we cannot include it as feedback to improve the quality of the formation. A consequence is that only when all the robots have the same heading direction, which only happens when the leader is moving in a linear path, the formation error can really converge to zero. However, even when that is not the case the geometric shape exhibited by the formation approaches closely the desired one, except for the formation orientation. Another consequence, of not knowing the heading direction of the leader/lead, is that the team of robots is not able to perform maneuvers that absolutely require the knowledge of the leader's heading. Such a maneuver is e.g. the formation acquisition problem (Arai et al. 2002): the lead robot stands still in an assigned position and the remaining robots have to position themselves in specific positions (the vertices of the geometric configuration that defines the formation) and with a specific orientation (i.e. the orientation of the lead robot). To the best of our knowledge this is a common problem to all approaches that do not rely on the leader's heading direction. Certainly, if we would have used the lead's heading direction we could have written control laws that would generate formation acquisition.

Several simulation results have been presented, that demonstrate the features of our controllers: the ability to establish and move a given formation; the ability to change the formation shape at runtime by an explicit change in the formation matrix; the implicit ability to split and join formations in the presence of obstructions due either to static or moving obstacles, thus reducing the coordination effort. These are inherent to the framework, that is there no need to trigger these features explicitly. It is important to strength that some of the results are presented in simulation only because we are limited in practice by the number of robots we had available (only 3 Kheperas and 2 large robots).

We have also presented implementation results in teams of Khepera robots performing a line formation and switching from column to triangle formation. Although we have presented our results with only three robots, this framework scales naturally to teams with more robots without extra computation costs (the computation cost per robot only depends on the controller it is implementing and not on the number of the robots on the team). Due to the limited sensorial resources of the Kheperas we had to rely on a dead-reckoning mechanism so that each robot locates itself (it is necessary a common reference frame) and radio-link to communicate the *leader* position to *followers*. A degradation of the results is visible which limits the extent of the experiments. This is mainly because of the difficulty to align all the robots in the same reference frame combined with positioning by dead-reckoning. We have shown that this problem can be overcome if one uses robots equipped with vision. In the larger robots the omnidirectional vision system provides directly estimates of the bearing angle, $\phi_i - \psi_i$, and distance. Thus neither communication nor dead-reckoning is necessary. In fact each robot does not need to know its pose (location and heading direction) neither the pose of other team mates.

Going from a simulation to a real implementation is very straightforward. Only some fine-tuning was necessary, specially on parameters that adjust path velocity and that deal with sensor measures (because of the sensor range).

We have shown that even if the formation matrix is chang-ed at runtime (e.g. due to a robot failure) the control system is able to stabilize the new formation shape. One remaining issue, relates to the automatic generation/change of the formation matrix, or, more specifically, given a desired formation shape that is dependent on the task (for terrain sweep, the shape could be a line; for moving a team, the shape could be a grid, or a column; etc.) and an assigned *Lead robot*, how to distribute the remaining robots in that formation (Fredslund and Matarić 2002; Brimble and Press 2003)? We have some preliminary work done (Monteiro and Bicho 2008) using the concept of cost function minimization to assign robots to locations in the formation, and taking into account the maximum allowable distance from a follower to a leader (dependant on the follower visibility range). This approach is valid both when the formation is explicitly triggered (establishing the initial formation at start, or changing the formation at runtime) or the change is due to an alteration of the number of robots in the formation (a decrease caused by the failure of robots, or an increase by adding more robots to the formation). An additional criterion to build the formation matrix could be to consider the shortest possible chains of leader–follower robots, because in principle this would lead to a reduction of the propagated formation error. One should also consider each robot sensor constraints, when building the formation matrix, in order to ensure that a follower is able to see its *leader*.

How to deal with problems of occlusion and missing sensory information (e.g. *follower* temporarily does not see its *leader*) is another important topic that needs to be addressed (see e.g. Renaud et al. 2004; Asama et al. 2009). In our approach these problems can be overcome in two distinct ways: (1) via re-definition of the formation matrix whenever necessary, and taking into account robots visibility; (2) by endowing the robots with cognitive capabilities (e.g. detection, memory, forgetting, predictive perception and anticipation), which can be generated using dynamic neural fields that represent information about the direction at which the leader robot may lie (Bicho et al. 2000; Erlhagen and Bicho 2006). It is important to strength that

this will not imply changing the controllers, only a dynamic neural field that allows to estimate and memorize the direction, ψ_i, at which the *leader* is needs to be introduced. This will be the focus of our work in the near future.

Acknowledgements This work was supported through COOP-DYN (POSI/SRI/38051/2001), financed by the Portuguese Foundation for Science and Technology (FCT) and project fp6–IST2 EU—project JAST—Joint Action Science and Technology (project number 003747). We thank Miguel Vaz and Nzoji Hipolito for their help in the implementations with real robots. We would like also to thank the anonymous reviewers for their insightful comments which helped improving the paper.

Appendix A: Robot kinematics

The path velocity, v, and angular velocity, ω, of our platforms are controlled by setting the linear speeds of the two driving wheels as follows ($l = left$, $r = right$):

$$v_r = v + \frac{D_{wheels}}{2}\omega, \tag{45}$$

$$v_l = v - \frac{D_{wheels}}{2}\omega \tag{46}$$

where D_{wheels} is the distance between the two driving wheels. $\omega = \frac{d\phi}{dt}$ is obtained directly from the behavioral dynamics for the heading direction, (21), and v results from integrating (23), by following a Euler method, i.e., $v = v + dt.g(v)$, with dt being the time step.

Appendix B: Parameters used in the experiments

Parameter	Simulation	Kheperas	Large robots
β_1:	20	$1/(4dt)$	$1/(5dt)$
β_2:	20	25	25
λ_{col}:	1.5	$1/(40dt)$	$1/(20dt)$
λ_{lin}:	1.5	$1/(40dt)$	$1/(20dt)$
λ_{obl}:	1.5	$1/(40dt)$	$1/(20dt)$
T_Δ:	4 sec	$7dt$	6
α_{obs}:		$4.5dt$	dt
α_{col}:		$4.5dt$	$5dt$
α_{lin}:		$4.5dt$	$5dt$
α_{obl}:		$4.5dt$	$5dt$
γ:	$\pi/4$	$\pi/4$	$\pi/4$
μ:	15	100	10
K_v:	6	4	–
$T_{2c,obs}$:	4 sec	$115dt$	$100dt$
dt :	20 ms	min = 40 ms max = 80 ms	min = 51 ms max = 55 ms

References

Antonelli, G., & Chiaverini, S. (2006). Kinematic control of platoons of autonomous vehicles. *IEEE Transactions on Robotics*, *22*(6), 1285–1292.

Arai, T., Pagello, E., & Parker, L. (2002). Editorial: Advances in multi-robot systems. *IEEE Transactions on Robotics and Automation*, *18*(5), 655–661.

Archibald, J. K., & Frost, R. L. (2007). A decentralized approach to multi-robot formation initialization. *International Journal of Robotics and Automation*, *22*(4), 304–312.

Asama, H., Kurokawa, H., Ota, J., & Sekiyama, K. (2009). Cooperative object tracking with mobile robotic sensor network. *Distributed Autonomous Robotic Systems*, *8*, 51–62.

Balch, T., & Arkin, R. C. (1998). Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, *14*(6), 926–939.

Balch, T., & Hybinette, M. (2000). Social potentials for scalable multi-robot formations. In *Proc. of the IEEE intl. conf. on robotics and automation* (Vol. 1, pp. 73–80).

Barfoot, T., & Clark, C. (2004). Motion planning for formations of mobile robots. *Robotics and Autonomous Systems*, *46*, 65–78.

Bauer, F., Bristow, J., Hartman, K., Quinn, D., & How, J. (1997). Satellite formation flying using an innovative autonomous control system (autocon) environment. In *AIAA guidance, navigation and control conf.* (pp. 657–666).

Beard, R., Lawton, J., & Hadaegh, F. (1999). A feedback architecture for formation control. In *Proc. of the 1999 American control conf.* (Vol. 6, pp. 4087–4091). San Diego, USA.

Bicho, E. (2000). *Dynamic approach to behavior-based robotics: design, specification, analysis, simulation and implementation*. Aachen: Shaker Verlag.

Bicho, E., & Monteiro, S. (2003). Formation control for multiple mobile robots: a non-linear attractor dynamics approach. In *2003 IEEE/RSJ intl. conf. on intelligent robots and systems* (pp. 2016–2022). Las Vegas, NV.

Bicho, E., & Schöner, G. (1997). The dynamic approach to autonomous robotics demonstrated on a low-level vehicle platform. *Robotics and Autonomous Systems*, *21*, 23–35.

Bicho, E., Mallet, P., & Schöner, G. (2000). Target representation on an autonomous vehicle with low-level sensors. *The International Journal of Robotics Research*, *19*(5), 424–447.

Bicho, E., Louro, L., & Erlhagen, W. (2004). Coordinated transportation with minimal explicit communication between robots. In *5th IFAC symposium on intelligent autonomous vehicles (IAV 2004)*, Lisbon, Portugal.

Bicho, E., Moreira, A., Diegues, S., Carvalheira, M., & Monteiro, S. (2006). Airship formation control. In *3rd intl. conf. on informatics in control, automation and robotics, in workshop multi-agent robotic systems (MARS 2006)* (pp. 22–33). Setubal, Portugal.

Bom, J., Thuilot, B., Marmoiton, F., & Martinet, P. (2005). A global control strategy for urban vehicles platooning relying on nonlinear decoupling laws. In *Proc. of the IEEE/RSJ intl. conf. on intelligent robots and systems* (pp. 1995–2000). Edmonton, Alberta, Canada.

Brimble, R., & Press, J. (2003). Minimal cost robot formations. In *Proc. of the 11th intl. conf. on advanced robotics* (pp. 1487–1495). Coimbra, Portugal.

Costa e Silva, E., Bicho, E., & Erlhagen, W. (2006). The potential field method and the nonlinear attractor dynamics approach: what are the differences? In *Controlo 2006—7th Portuguese conference on automatic control*, Lisboa, Portugal.

Cruz, C., & Carelli, R. (2008). Dynamic model based formation control and obstacle avoidance of multi-robot systems. *Robotica*, *26*, 345–356.

Das, A., Fierro, R., Kumar, V., Ostrowski, J. P., Spletzer, J., & Taylor, C. J. (2002). A vision-based formation control framework. *IEEE Transactions on Robotics and Automation*, *18*(5), 813–825.

Desai, J., Ostrowski, J., & Kumar, V. (2001). Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, *17*(6), 905–908.

Do, K. D., & Pan, J. (2007). Nonlinear formation control of unicycle-type mobile robots. *Robotics and Autonomous Systems*, *55*, 191–204.

Edwards, D., Bean, T., Odell, D., & Anderson, M. (2004). A leader–follower algorithm for multiple AUV formations. In *Autonomous underwater vehicles* (pp. 40–46).

Erlhagen, W., & Bicho, E. (2006). The dynamic neural field approach to cognitive robotics. *Journal of Neural Engineering*, *3*, 36–54.

Fahimi, F. (2007). Sliding-mode formation control for underactuated surface vessels. *IEEE Transactions on Robotics*, *23*(3), 617–622.

Fahimi, F. (2008). Full formation control for autonomous helicopter groups. *Robotica*, *26*(2), 143–156.

Fajen, B. R., Warren, W. H., Temizer, S., & Kaelbling, L. P. (2003). A dynamical model of visually-guided steering, obstacle avoidance, and route selection. *International Journal of Computer Vision*, *54*(1/2/3), 13–24.

Fierro, R., Alur, R., Das, A., Esposito, J., Grudic, G., Hur, Y., Kumar, V., Lee, I., Ostrowski, J., Pappas, G., Southall, B., Spletzer, J., & Taylor, C. (2002). A framework and architecture for multi-robot coordination. *The International Journal of Robotics Research*, *21*(10–11), 977–995.

Fowler, J., & Andrea, R. (2002). Distributed control of close formation flight. In *Proc. of the 41st IEEE conf. on decision and control* (pp. 2972–2977). Las Vegas, USA.

Fredslund, J., & Matarić, M. (2002). A general local algorithm for robot formations. *IEEE Transactions on Robotics and Automation*, *18*(5), 837–846, special issue on Multirobot Systems.

Gazi, V. (2005). Swarm aggregations using artificial potentials and sliding-mode control. *IEEE Transactions on Robotics*, *21*(6), 1208–1214.

Ge, S., & Fua, C. (2005). Queues and artificial potentials trenches for multirobot formations. *IEEE Transactions on Robotics*, *21*(4), 646–656.

Gustavi, T., & Hu, X. (2005). Formation control for mobile robots with limited sensor information. In *Proc. IEEE intl. conf. robotics and automation* (pp. 1791–1796). Barcelona, Spain.

Hao, Y., & Agrawal, S. (2005). Formation planning and control of UGVs with trailers. *Autonomous Robots*, *19*, 257–270.

Honary, E., McQuade, F., Ward, R., Woodrow, I, Shaw, A., Barnes, D., & Fyfe, M. (2009). Robotic experiments with cooperative aerobots and underwater swarms. *Robotica*, *27*(1), 37–49.

Hong, S., Shin, S., & Ahn, D. (2001). Formation control based on artificial intelligence for multi-agent coordination. In *Proc. of the intl. symposium on industrial electronics* (pp. 429–734). Busan, Korea.

Hsieh, A., Kumar, V., & Chaimowicz, L. (2008). Decentralized controllers for shape generation with robotic swarms. *Robotica*, *26*(05), 691–701.

Hu, X., & Zeigler, B. (2004). Model continuity to support software development for distributed robotic systems: a team formation example. *Journal of Intelligent and Robotic Systems*, *39*, 71–87.

Johnson, P., & Bay, J. (1995). Distributed control of simulated autonomous mobile robot collectives in payload transportation. *Autonomous Robots*, *2*(1), 43–64.

Kalantar, S., & Zimmer, U. (2007). Distributed shape control of homogeneous swarms of autonomous underwater vehicles. *Autonomous Robots*, *22*(1), 37–53.

Kalantar, S., & Zimmer, U. (2009). Optima localization by vehicle formations imitating the Nelder-Mead simplex algorithm. *Autonomous Robots*, *27*(3), 239–260.

Kaminka, G., Schechter-Glick, R., & Sadov, V. (2008). Using sensor morphology for multirobot formations. *IEEE Transactions on Robotics*, *24*(2), 271–282.

Kim, G., Lee, D., & Lee, K. (2001). Formation approach for mobile robots with inaccurate sensor information. In *Proc. of the intl. conf. on control, automation and systems* (pp. 805–808). Cheju National University, Jeju, Korea.

Kostelnik, P., Samulka, M., & Janosik, M. (2002). Scalable multi-robot formations using local sensing and communication. In *Proc. of the third intl. workshop on robot motion and control* (pp. 319–324).

Krishnanand, K., & Ghose, D. (2005). Formations of minimalist mobile robots using local-templates and spatially distributed interactions. *Robotics and Autonomous Systems*, *53*, 194–213.

Kwok, N. M., Ha, Q. P., & Fang, G. (2007). Motion coordination for construction vehicles using swarm intelligence. *International Journal of Advanced Robotic Systems*, *4*(4), 469–476.

Large, E., Christensen, H., & Bajcy, R. (1999). Scaling the dynamic approach to path planning and control: Competition amoung behavioral constraints. *The International Journal of Robotics Research*, *18*(1), 37–58.

Leonard, N., & Fiorelli, E. (2001). Virtual leaders, artificial potentials and coordinated control of groups. In *Proc. of the 40th conf. on decision and control* (pp. 2968–2973).

Lewis, M. A., & Tan, K. (1997). High precision formation control of mobile robots using virtual structures. *Autonomous Robots*, *4*, 387–403.

Monteiro, S., & Bicho, E. (2002). A dynamical systems approach to behavior-based formation control. In *Proc. IEEE intl. conf. robotics and automation* (pp. 2606–2611).

Monteiro, S., & Bicho, E. (2008). Robot formations: Robots allocation and leader-follower pairs. In *Proc. IEEE international conference on robotics and automation ICRA 2008* (pp. 3769–3775).

Monteiro, S., Vaz, M., & Bicho, E. (2004). Attractor dynamics generates robot formations: from theory to implementation. In *Proc. IEEE intl. conf. on robotics and automation* (pp. 2582–2586). New Orleans, LA.

Naffin, D. J., & Sukhatme, G. S. (2004). Negotiated formations. In *Proceedings of the eighth conference on intelligent autonomous systems* (pp. 181–190). Amsterdam, The Netherlands.

Nangia, R., & Palmer, M. (2007). Formation flying of a commercial aircraft—assessment using a new approach—wing span load and camber control. In *Proc. of the 45th AIAA aerospace sciences meeting and exhibit*, Reno, USA.

Ogren, P. (2004). Split and join of vehicle formations doing obstacle avoidance. In *Proc. of the 2004 IEEE intl. conf. on robotics and automation* (pp. 1951–1955). New Orleans, LA.

Ogren, P., Fiorelli, E., & Leonard, N. (2002). Formations with a mission: stable coordination of vehicle group maneuvers. In *Proc. symposium on mathematical theory of networks and systems*.

Olfati-Saber, R., & Murray, R. (2002). Graph rigidity and distributed formation stabilization of multi-vehicle systems. In *Proc. of the 41st conf. on decision and control* (Vol. 3, pp. 2965–2971). Las Vegas, NV.

Pledgie, S., Hao, Y., Pereira, A., Agrawal, S., & Murphey, R. (2002). Groups of unmanned vehicles: differential flatness, trajectory planning and control. In *Proc. of the 2002 IEEE intl. conf. on robotics and automation* (pp. 3461–3466). Washington, DC.

Ren, W., & Beard, R. (2002). Virtual structure based spacecraft formation control with formation feedback. In *AIAA guidance, navigation, and control conference* (pp. 2002–4963). Monterey, CA.

Ren, W., & Sorensen, N. (2008). Distributed coordination architecture for multi-robot formation control. *Robotics and Autonomous Systems*, *56*, 324–333.

Renaud, P., Cervera, E., & Martinet, P. (2004). Towards a reliable vision-based mobile robot formation control. In *Proc. of the IEEE/RSJ intl. conf. on intelligent robots and systems* (pp. 3176–3181). Sendai, Japan.

Schöner, G., & Dose, M. (1992). A dynamical systems approach to task-level system integration used to plan and control autonomous vehicle motion. *Robotics and Autonomous Systems*, *10*, 253–267.

Schöner, G., Dose, M., & Engels, C. (1995). Dynamics of behavior: Theory and applications for autonomous robot architectures. *Robotics and Autonomous Systems*, *16*, 213–245.

Seiler, P., Pant, A., & Hedrick, K. (2002). Analysis of bird formations. In *Proc. of the 41st IEEE conf. on decision and control* (pp. 118–123). Las Vegas, USA.

Soares, R., & Bicho, E. (2002). Using attractor dynamics to generate decentralized motion control of two mobile robots transporting a long object in coordination. In *Proc. of the workshop on cooperative robotics, in IROS, 2002: 2002 IEEE/RSJ intl. conf. on intelligent robots and systems*. EPFL Lausanne, Switzerland.

Soares, R., Bicho, E., Machado, T., & Erlhagen, W. (2007). Object transportation by multiple mobile robots controlled by attractor dynamics: theory and implementation. In *Proc. of the IEEE/RSJ intl. conf. on intelligent robots and systems* (pp. 937–944). San Diego, CA.

Steinhage, A. (1997). *Dynamical systems for the generation of navigation behavior*. PhD thesis, Ruhr-Universitat Bochum, Germany.

Sudsang, A. (2002). Sweeping the floor: moving multiple objects with multiple disc-shaped robots. In *Proc. of the IEEE/RSJ intl. conf. on intelligent robots and systems* (pp. 2825–2830). Lausanne, Switzerland.

Vidal, R., Shakernia, O., & Sastry, S. (2003). Formation control of nonholonomic mobile robots with omnidirectional visual servoing and motion segmentation. In *IEEE conf. on robotics and automation* (pp. 584–589). Taipei, Taiwan.

Yamaguchi, H. (1999). A cooperative hunting behavior by mobile-robot troops. *The International Journal of Robotics Research*, *18*(8), 931–940.

Young, B., Beard, R., & Kelsey, J. (2001). A control scheme for improving multi-vehicle formation maneuvers. In *Proc. of the American control conf.* (pp. 704–709). Arlington, VA.

Sérgio Monteiro received is Ph.D. in Automation and Control from the University of Minho in 2007, with a thesis in the formation control of robot teams. Currently he serves as Assistant Professor at the Department of Industrial Electronics at University of Minho, where he also integrates the Autonomous Robotics & Dynamical Systems group. His research interests are focused on multi-robot systems.

Estela Bicho is Professor at the Department of Industrial Electronics at University of Minho, Portugal, where she is responsible for courses in Non-linear Dynamical Systems, Control and Robotics and heads the research team on Autonomous Robotics & Dynamical systems. She is also a staff member of the MIT|Portugal Joint Program on Advanced Studies in Bioengineering. She obtained the Ph.D. degree in Robotics, Automation and Control, in 1999, from the University of Minho. Her Ph.D. work received the honour price from Portuguese-IBM (1999). Between 1995–99 she was a member of the 'Equipe de Dynamique', lead by Prof. Gregor Schöner, at the Centre de Recherche en Neurosciences Cognitives at CNRS in Marseille, France. Her research concentrates on the use of dynamical systems for the design and implementation of cognitive control architectures for single and multi-robot systems. She has been PI in several nationally and EU funded research project in robotics. For more information see: http://dei-s1.dei.uminho.pt/pessoas/estela/.