Flocking for Multi-Robot Systems via the Null-Space-based Behavioral Control

Gianluca Antonelli Filippo Arrichiello Stefano Chiaverini

Abstract— **In this paper the flocking problem for a multi-robot system, consisting in making the robots of a team grouping together, is addressed. The flocking is achieved resorting to the Null-Space-based Behavioral (NSB) control by defining very simple behaviors for each robot of the team and by properly arranging these behaviors in priority. The NSB algorithm, making the robots using only local information, successfully achieves the flocking with or without a rendez-vous point and in eventual presence of obstacles. Extensive simulations and experiments using differential-drive mobile robots prove the effectiveness of the proposed algorithm.**

I. Introduction

In nature several biological species exist that exhibit a collective behavior implementing local interacting strategies. Examples are given by birds, ants, bees and human crowd. Among the different collective behavior, flocking is an interesting emergent attitude that fascinated researchers from several disciplines apparently far one each other: physicists, social scientist, animal psychologists, etc. The flocking problem is also an interesting control problem [16] involving the coordination of multiple agents characterized by limited sensing and communication capabilities. Flocking is strictly related to the study of self-organized networks of mobile agents and it is a specific case study of the coordination control of multiple robots that includes distribute sensing, exploration, coverage, search and rescue, etc. The work in [11] provides a significant overview about this control problem.

In 1986 Reynolds [22] published a seminal work in which a computer model for coordinating the motion of animals as bird flocks or fish schools were presented. A wide literature now exists that reports interesting results concerning the flocking problem; in [18] different solutions are investigated and their stability analysis in discussed. Based on local sensing, each agent moves according to three different terms, a gradient-based term, a consensus term and a navigational feedback term that represent different *behaviors* of each agent. The work [15] surveys recent development in modelling, analysis and design of distributed motion coordination algorithms for multiple robots systems. An aspect that strongly influences the coordination strategy is the eventual possibility for one agent to explicitly exchange information with its neighbors, this possibility poses the challenge problem of *consensus*; an overview of the information consensus is given in [21]. The work [19] investigates consensus algorithms with emphasis on robustness, time-delays and performance guarantee. In [24], the proposed decentralized controller is stable under arbitrary changes in the connected network.

Autonomous robotics has been strongly influenced by the so-called robotics paradigm of behavior-based control [7], [10] that can be described by the relationship between the 3 primitives of robotics: Sense, Plan, and Act. The behavior-based paradigm represented one of the most investigated approaches to the design of multi-robot systems. In this paper, the flocking problem is solved resorting to the behavioral approach defined NSB (Null-Space-based Behavioral control) [5]. This approach, strongly related to the kinematic control presented in [6], has been recently experimentally applied in a large number of multi-robot missions such as formation control or escort/entrap an autonomous target [3]. Most of the previously proposed missions were performed resorting to a centralized control architecture. In this paper, the first application of the NSB acting in a decentralized scenario are presented. To elaborate its motion directives, each robot applies the NSB control only referring to its local information, i.e., its position, the rendez-vous point and its instantaneous neighbors (the robots in its interaction range) positions. Extensive simulations, assuming non-holonomic agents, and experiments using a platoon of 7 differential-drive mobile robots, namely the Khepera II, prove the effectiveness of the proposed algorithm.

II. The Flocking Problem

The flocking problem has been approached by several researchers of different disciplines, for this reason the word *flocking* itself assumes slightly different meanings. We refer to it as the problem of making the robots of a team reaching particular configuration structures, namely *lattice* configurations, only using local information. In particular, the robots can sense their neighbors or exchange with them their positions' information (there is no need to exchange or measure the neighbors' velocities as in [18]), and they elaborate their motion directives on the basis of these only information. The flocking behavior, thus, emerges as the results of local behaviors of the individual robots acting simultaneously.

The local interaction among the robots can be analyzed in the framework of communication/sensing networks that depend on the displacement of the robots in the environment,

Authors are listed in alphabetical order.

G. Antonelli, F. Arrichiello and S. Chiaverini are with the Dipartimento di Automazione, Elettromagnetismo, Ingegneria dell'Informazione e Matematica Industriale, Università degli Studi di Cassino, Via G. Di Biasio 43, 03043, Cassino (FR), Italy, {antonelli,f.arrichiello,chiaverini}@unicas.it, http://webuser.unicas.it/lai/robotica.

and, in particular, on the robots relative distances and their sensing capabilities. To formally take into consideration these aspects in the paper, the basic notions on graph theory are briefly recalled in the following.

Inheriting the nomenclature from [18], a graph G is a pair $(\mathcal{V}, \mathcal{E})$ that consists in a set of vertexes $\mathcal{V} = \{1, 2, \ldots, n\}$ and edges $\mathcal{E} \subseteq \{(i,j) : i,j \in \mathcal{V}, j \neq i\}$. In this paper, an undirected graph will be considered, i.e., $(i,j) \in \mathcal{E} \Rightarrow (j,i) \in \mathcal{E}$. The scalar quantities $|\mathcal{V}|$ and $|\mathcal{E}|$ will be denoted as order and size of the graph, respectively. The adjacent matrix $\boldsymbol{A}$ collects the information concerning the edges such that $a_{i,j} \neq 0 \Leftrightarrow (i,j) \in \mathcal{E}$; for undirected graph it is $\boldsymbol{A} = \boldsymbol{A}^{\mathrm{T}}$. The set of neighbors of node i is defined as

$$\mathcal{N}_i = \{j \in \mathcal{V} : a_{i,j} \neq 0\} = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}\}$$

The position of each node is denoted as $\boldsymbol{p}_i \in \mathbb{R}^l$, where $l = 2$ in case of a material point moving on a surface, or $l = 3$ in the case of a material point moving in the space. The configuration of all the nodes of the graph is defined as the vector $\boldsymbol{p} \in \mathbb{R}^{ln}$ defined as

$$\boldsymbol{p} = [\, \boldsymbol{p}_1^{\mathrm{T}} \quad \boldsymbol{p}_2^{\mathrm{T}} \quad \ldots \quad \boldsymbol{p}_n^{\mathrm{T}} \,]^{\mathrm{T}}.$$

A framework, or structure, is a pair $(G, \boldsymbol{p})$ that consists in a graph and the configuration of its nodes.

A group of dynamic node/robots has equation of motion

$$\boldsymbol{v}_i = \boldsymbol{u}_i \quad \forall i \in \mathcal{V},$$

where $\boldsymbol{v}_i \in \mathbb{R}^l$ is the velocity of each node. In alternative, second order kinematic equations can be used as in [18].

If one further defines an interaction range r between two robots, it is possible to compute the neighbors of each robot as

$$\mathcal{N}_i = \{j \in \mathcal{V} : \|\boldsymbol{p}_i - \boldsymbol{p}_j\| < r\}$$

where $\|\cdot\|$ represents the Euclidean norm.

One possible model to represent the spatial order of flocks is achieved by defining a geometric structure. Among the possible choices one is given by the α-*Lattice* structures obtained by all the configurations that satisfy:

$$\|\boldsymbol{p}_i - \boldsymbol{p}_j\| = d \quad \forall j \in \mathcal{N}_i(\boldsymbol{p}) \tag{1}$$

where d is the *lattice's scale* and $\kappa = r/d$ is the *lattice's ratio*. Obviously, an α-Lattice is characterized by edges of the same length.

Configurations close the the α-Lattice are the *quasi-α-Lattice* that introduces a tolerance in the definition of eq. (1):

$$-\delta \leq \|\boldsymbol{p}_i - \boldsymbol{p}_j\| - d \leq \delta \quad \forall (i,j) \in \mathcal{E}(\boldsymbol{p}), \tag{2}$$

an example of such a structure is shown in Figure 1, obtained resorting to the algorithm presented in this paper.

A sort of metrics that measures the distance of a quasi-α-Lattice to a α-Lattice is given by an index defined as *deviation energy*:

$$E(\boldsymbol{p}) = \frac{1}{|\mathcal{E}(\boldsymbol{p})| + 1} \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \left(\|\boldsymbol{p}_i - \boldsymbol{p}_j\| - d \right)^2 \tag{3}$$

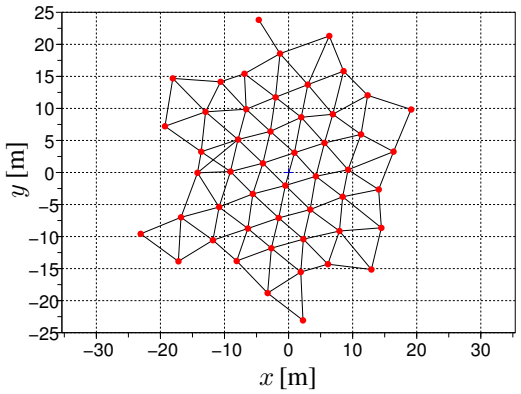

Fig. 1. Example of quasi-α-Lattice for 50 robots obtained with the algorithm presented in this paper in a 2-dimensional version.

It is worth noticing that the zero is the global minimum of such an index and is achieved for α-Lattice geometries.

In this paper, the flocking problem will be solved as finding a control law $\boldsymbol{u}_i(\mathcal{N}_i)$, function of the only neighbors' positions, in order to achieve a quasi-α-Lattice structure.

III. The NSB control for multi-robot systems

In a general robot mission the accomplishment of several tasks at the same time is of interest. A possible technique to handle the tasks' composition has been proposed by [8] and [9], which consists of assigning a relative priority to the single task functions by resorting to the task-priority inverse kinematics introduced by [13] and [17] for ground-fixed redundant manipulators. Nevertheless, as discussed by [12], just in the case of conflicting tasks it is necessary to devise singularity-robust algorithms that ensure proper functioning of the inverse velocity mapping. Based on these works, this idea is developed by [6] in the framework of the singularity-robust task-priority inverse kinematics originally presented by [12]. This control approach, namely the Null-Space-based Behavioral control, has been then analyzed in the framework of behavior-based approaches for the control of a single autonomous vehicle in [5] and of multi-robot systems in [4].

In this paper, the flocking behavior emerges by applying decentralized NSB controllers on board of each robot of the team. In particular, since the robots only use local information, the NSB will be presented for the individual control of a single autonomous robots as in [5].

By defining as $\boldsymbol{\sigma} \in \mathbb{R}^m$ the generic task variable to be controlled by the i-robot, it is:

$$\boldsymbol{\sigma} = \boldsymbol{f}(\boldsymbol{p}_i, \boldsymbol{p}_j) \tag{4}$$

where $\boldsymbol{p}_i \in \mathbb{R}^l$ is the i-robot configuration and $\boldsymbol{p}_j \in \mathcal{N}_i$ are the configurations of its neighboring robots.

Considering the neighboring robots as static objects ($v_j = 0$), then, the corresponding differential relationship is:

$$\dot{\boldsymbol{\sigma}} = \frac{\partial \boldsymbol{f}(\boldsymbol{p}_i)}{\partial \boldsymbol{p}_i} \boldsymbol{v}_i = \boldsymbol{J}(\boldsymbol{p}_i) \boldsymbol{v}_i \,, \tag{5}$$

where $\boldsymbol{J} \in \mathbb{R}^{m \times l}$ is the configuration-dependent task Jacobian matrix and $\boldsymbol{v}_i \in \mathbb{R}^l$ is the robot velocity.

An effective way to generate motion references for the vehicles starting from desired values $\boldsymbol{\sigma}_d(t)$ of the task function is to act at the differential level by inverting the (locally linear) mapping (5); in fact, this problem has been widely studied in robotics (see, e.g., [23] for a tutorial). A typical requirement is to pursue minimum-norm velocity, leading to the least-squares solution:

$$\boldsymbol{u}_i \;=\; \boldsymbol{J}^{\dagger}\dot{\boldsymbol{\sigma}}_d \;=\; \boldsymbol{J}^{\mathrm{T}}\left(\boldsymbol{J}\boldsymbol{J}^{\mathrm{T}}\right)^{-1}\dot{\boldsymbol{\sigma}}_d\,. \tag{6}$$

However, discrete-time integration of the vehicle's reference velocity would result in a numerical drift of the reconstructed vehicle's position; the drift can be counteracted by a so-called Closed Loop Inverse Kinematics (CLIK) version of the algorithm, namely,

$$\boldsymbol{u}_i = \boldsymbol{J}^{\dagger}\left(\dot{\boldsymbol{\sigma}}_d + \boldsymbol{\Lambda}\widetilde{\boldsymbol{\sigma}}\right), \tag{7}$$

where $\boldsymbol{\Lambda}$ is a suitable constant positive-definite matrix of gains and $\widetilde{\boldsymbol{\sigma}}$ is the task error defined as $\widetilde{\boldsymbol{\sigma}}=\boldsymbol{\sigma}_d-\boldsymbol{\sigma}$. Thus, the Null-Space-based Behavioral control intrinsically requires a differentiable analytic expression of the tasks defined, so that it is possible to compute the required Jacobians.

In the general case, the mission for the single i-robot is composed by multiple tasks, thus, the motion directive should be elaborated by properly merging the motion directives elaborated for all the single active tasks. In detail, on the analogy of eq. (7), the single task velocity is computed as:

$$\boldsymbol{u}_k = \boldsymbol{J}_k^{\dagger}\left(\dot{\boldsymbol{\sigma}}_{k,d} + \boldsymbol{\Lambda}_k\widetilde{\boldsymbol{\sigma}}_k\right), \tag{8}$$

where the subscript k denotes the k-th task. Let us further define as

$$\boldsymbol{N}_k = \left(\boldsymbol{I} - \boldsymbol{J}_k^{\dagger}\boldsymbol{J}_k\right)$$

the null space projector of the task k. If the subscript k also denotes the degree of priority of the task with, e.g., Task 1 being the highest-priority one, in the case of 3 tasks and according to [12] or [14], then, the CLIK solution (7) is modified into

$$\boldsymbol{u}_i = \boldsymbol{u}_1 + \boldsymbol{N}_1\boldsymbol{u}_2 + \boldsymbol{N}_{12}\boldsymbol{u}_3 \tag{9}$$

where $\boldsymbol{N}_{12}$ is the null space projector obtained by stacking the Jacobians corresponding to the tasks 1 and 2. Extension to a desired number of tasks is straightforward.

As reported in [5], the NSB approach has been formulated as a knew kind of behavior-based approach that differs from the other techniques in the behavioral coordinations, that is, in the way they compose the single task outputs to build the motion command to the robots. The Null-Space-based Behavioral control, in fact, uses a priority based logic to combine multiple tasks and uses null-space projector to delete the outputs of the lower priority tasks that would conflict with the higher tasks. Thus, the Null-Space-based Behavioral control always fulfills the highest-priority task. The lower-priority tasks, on the other hand, are fulfilled only in a subspace where they do not conflict with the ones having higher priority.

Concerning the flocking problem, the NSB control approach differs from the other control approaches proposed in literature in the way the single elementary tasks are managed and combined. E.g., in the paper by [18] the single elementary task are combined following a potential approach and, from a behavior-based control point of view, implementing a sort of cooperative control strategy, that is, the outputs of the single task functions are combined as a weighted sum to elaborate the final motion reference to the robot.

IV. Flocking via the NSB approach

In this paper, the flocking problem described in Section II is solved by defining several local task functions and by applying the NSB control strategy on board of each robot. Moreover, local supervisors are in charge of dynamically selecting the active tasks and their priority orders to properly perform the individual missions.

In the following, the definitions of the task functions and the detail on the supervisor are presented.

A. Tasks' definition

Only two tasks are sufficient to generate the flocking behavior to the group of robots in presence of a rendez-vous point. An additional task is required in case of the presence of obstacles. Each of the functions is defined for each robot and relies on the neighbors' information, only:

Lattice task. This task function $\sigma_{l,ij} \in \mathbb{R}$ is aimed at keeping a constant distance (the lattice's scale) between two robots whose distance is smaller than the interaction range:

$$\sigma_{l,ij} = \left\|\boldsymbol{p}_i - \boldsymbol{p}_j\right\| \quad \text{with} \qquad \sigma_{l,ij,d} = d$$

Its Jacobian $\boldsymbol{J}_{l,ij} \in \mathbb{R}^{1\times 3}$ and Null $\boldsymbol{N}_{l,ij} \in \mathbb{R}^{3\times 3}$ matrices are defined as

$$\begin{aligned}\boldsymbol{J}_{l,ij} &= \hat{\boldsymbol{p}}_{ij}^{\mathrm{T}}\\ \boldsymbol{N}_{l,ij} &= \boldsymbol{I} - \hat{\boldsymbol{p}}_{ij}\hat{\boldsymbol{p}}_{ij}^{\mathrm{T}}\end{aligned}$$

where

$$\hat{\boldsymbol{p}}_{ij} = \frac{\boldsymbol{p}_i - \boldsymbol{p}_j}{\left\|\boldsymbol{p}_i - \boldsymbol{p}_j\right\|}$$

Moving to rendez-vous task. The task function $\boldsymbol{\sigma}_{r,i} \in \mathbb{R}^3$ is aimed at creating a connected graph. Its definition is simply given by the robot position

$$\boldsymbol{\sigma}_{r,i} = \boldsymbol{p}_i \qquad \text{with} \qquad \boldsymbol{\sigma}_{r,i,d} = \boldsymbol{p}_{\mathrm{rv}}$$

where $\boldsymbol{p}_{\mathrm{rv}} \in \mathbb{R}^3$ is the rendez-vous point. The (3×3) Jacobian is simply the Identity matrix and the Null space projector is the (3×3) null matrix. It is worth noticing that it is useless to add tasks of lower priority with respect to this one.

Obstacle avoidance task. Obstacle avoidance for autonomous robots is a mandatory task and, resorting to the NSB approach, has been deeply discussed in previous papers such as, e.g., [6], [5]. Not surprisingly,

the obstacle avoidance task function is formally equal to the lattice task:

$$\sigma_{o,i} = \|\boldsymbol{p}_i - \boldsymbol{p}_o\| \qquad \text{with} \qquad \sigma_{o,i,d} = d_o$$

and

$$\begin{cases} \boldsymbol{J}_{o,i} &= \hat{\boldsymbol{p}}_{io}^{\mathrm{T}} \\ \boldsymbol{N}_{o,i} &= \boldsymbol{I} - \hat{\boldsymbol{p}}_{io}\hat{\boldsymbol{p}}_{io}^{\mathrm{T}} \end{cases} \quad \text{with} \quad \hat{\boldsymbol{p}}_{io} = \frac{\boldsymbol{p}_i - \boldsymbol{p}_o}{\|\boldsymbol{p}_i - \boldsymbol{p}_o\|}$$

where $\boldsymbol{p}_o$ is the position of the obstacle.

It is worth noticing that the task functions represent a sort of elementary behaviors for each robot. In this sense, a task function can be used several times if needed. As an example, an agent can implement the lattice task with respect to several different agents laying in its interaction range. More insights on the number and kind of tasks to be used is given in next Subsection.

B. Supervisor

The supervisor is a higher level function that, for each robot, is in charge of selecting the priority of the active tasks. According to the Degree of Freedoms (DOFs) considerations given, e.g., by [5], it is convenient to properly take into account the dimension of each task and to avoid requiring the fulfillment of an overall dimension larger than the available DOFs.

Each of the robot is aware only of the other robots inside its interaction area, among them, a list containing robots $\in \mathcal{N}_i$ sorted by their distance with respect to i is considered being $\boldsymbol{k}_i(1)$ the closest.

By referring to a 3-dimensional case, each robot computes the desired velocities corresponding to

- Lattice task with respect to the robot $\boldsymbol{k}_i(1)$ (if at least one robot $\in \mathcal{N}_i$);
- Lattice task with respect to the robot $\boldsymbol{k}_i(2)$ (if at least two robots $\in \mathcal{N}_i$);
- Lattice task with respect to the robot $\boldsymbol{k}_i(3)$ (if at least three robots $\in \mathcal{N}_i$);
- Moving to rendez-vous task (if present in the mission specification);
- Obstacle avoidance task (if $\boldsymbol{p}_o \in \mathcal{N}_i$)

that need to be properly arranged in a priority. A trivial situation arises when the flocking is required without a rendez-vous point and the set $\mathcal{N}_i$ is empty, in such a case the robot obviously stays still.

Let us first consider the case of the absence of obstacles in the set $\mathcal{N}_i$, in this case the supervisor computes the Lattice tasks assigning the higher priority to the closest robot. Since the Lattice task is monodimensional, if at least three robots belong to $\mathcal{N}_i$ the moving-to-rendez-vous task is discarded; otherwise, it is added as the lowest in priority. It is worth noticing that, even if more than three robots belong to $\mathcal{N}_i$, for the presented approach it is sufficient to consider only the closest three and not all of them as proposed by [18].

Let us now consider one obstacle in the interaction range of the robot, the supervisor firstly computes the desired velocity without the obstacle, it then checks if the robot would approach the obstacle or go away from it. In the latter situation nothing is changed with respect to the non-obstacle case. If, on the other side, the robot would approach the obstacle, then, the obstacle-avoidance task is selected as primary task, all the tasks are correspondingly lowered in priority and the last one is eventually removed if the sum of the tasks' dimension is larger than three.

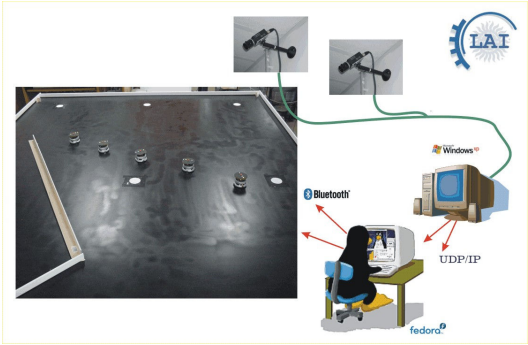

Fig. 2. Sketch of the multi-robot set-up available at the LAI (Laboratorio di Automazione Industriale) of the Università degli Studi di Cassino.

For seek of clarity, let us imagine a situation where the robot i has only one robot j inside the interaction range, its supervisor would then consider:

priority	task	dimension
1	lattice between i and j	1
2	rendez-vous	3

if, on the other side, several robots and the obstacle are inside the interaction range the supervisor would output:

priority	task	dim.
1	obstacle avoidance between i and $\boldsymbol{p}_o$	1
2	lattice between i and $\boldsymbol{k}_i(1)$	1
3	lattice between i and $\boldsymbol{k}_i(2)$	1

Is is worth noticing that in [18], as well as in this paper, each robot needs to know the position of other robots present in a set $\mathcal{N}_i$ that is simply a sphere around it, this is reasonable for robotic systems but not for biological flock mainly characterized by directional sensing such as, e.g., the eye-based vision. Future research might consider anisotropic sets $\mathcal{N}_i$ and proper tasks that take it into account.

V. Experiments and Simulations

In the following, the experimental set-up (Subsection V-A) and the results of the flocking execution (Subsection V-B) are reported. Moreover, a video of one of the experiments accompanies this paper, additional videos concerning other simulations and experiments can be downloaded from the laboratory website (address in the affiliation).

A. Experimental set-up

For these experiments, several Khepera II mobile robots manufactured by K-team [1] available at the LAI (Laboratorio di Automazione Industriale) of the Università degli Studi di Cassino are used. Those are differential-drive mobile

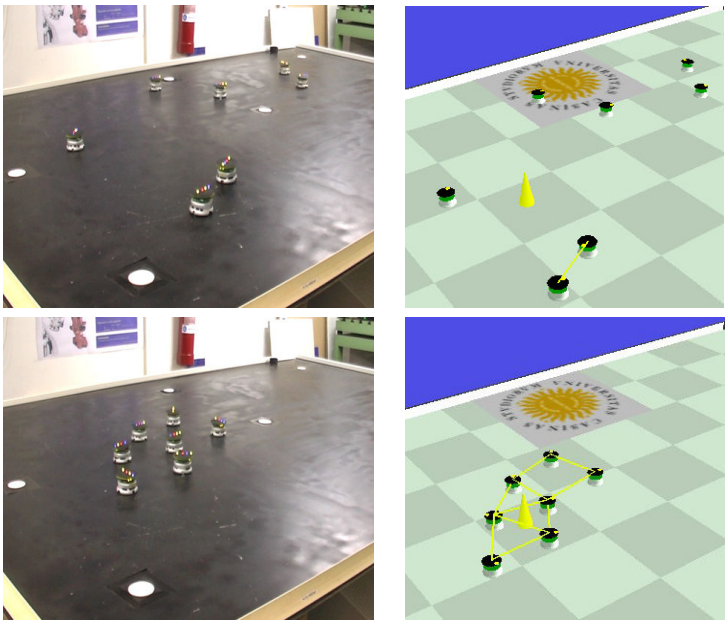

Fig. 3. Snapshots of the initial and final configurations of an experiment where the robots flock around a fixed rendez-vous point.

robots (with a unicycle-like kinematics) with an approximate dimension of $8\,\text{cm}$ of diameter. Each of them can communicate trough a Bluetooth module with a remote Linux-based PC where the NSB has been implemented. To allow the needed absolute position measurements we have developed a vision-based system using two CCD cameras, a Matrox Meteor-II frame grabber [2] and self-developed C++ image-processing functions. The acquired images are 1024×768 RGB bitmaps. The measurement error has an upper bound of $\approx 0.5\,\text{cm}$ and $\approx 1\,\text{deg}$. The remote PC receives the position measurements at a sampling time of $100\,\text{ms}$ and elaborates the NSB control for each of the robot, i.e., several controllers, independent one each other, are implemented on the remote PC; each of the controllers have access only to the corresponding agent's position and to the distances from the neighbors. In this way a decentralized controller is implemented by *filtering* the non accessible information for each robot. Once the NSB outputs the desired linear velocities for each robot, an heading controller is implemented [20] to obtain the wheels' desired velocities. The remote PC sends to each vehicle (trough the Bluetooth module) the wheels' desired velocities with a sampling time of $T = 80\,\text{ms}$. The wheels' controller (on board of each robot) is a PID developed by the manufacturer. A saturation of $40\,\text{cm/s}$ and $100\,\text{deg/s}$ has been introduced for the linear and angular velocities, respectively. Moreover, the encoders resolution is such that a quantization of $\approx 0.8\,\text{cm/s}$ and $\approx 9\,\text{deg/s}$ are experienced.

B. Experimental and simulative results

All the following experiments were performed with a team of 7 robots with the following parameters: lattice's scale $d = 25\,\text{cm}$, interaction range $r = 30\,\text{cm}$, lattice's ratio $\kappa = \frac{r}{d} = 1.2$, lattice task gain $\lambda_l = 0.3$, rendez-vous task gain $\lambda_r = 0.5$. Moreover, the initial agent configuration was always casual.

Figures 3 reports two snapshots corresponding to the initial and final configuration of an experiments run with 7 robots. Despite the presence of non-holonomicity, dynamics, communication delays, etc. it can be noticed that the flocking behavior is successfully achieved. Moreover, in the right side snapshots of figure 3, the acquired data are graphically elaborated to better appreciate the robots' connections. In particular, the yellow cone represent the position of the rendez-vous point and the lines correspond to edges. This experiment is also shown in the accompanying video.

In order to appreciate the algorithm performance with multi-robot systems, extensive simulations have been run considering a wide range of possible situations, presence/absence of the (eventually moving) rendez-vous point, presence/absence of the obstacle, 2-dimensional/3-dimensional motion, (non)holonomic vehicles, increasing number of agents, etc. Due to the limited space, it is not possible to report all the simulation results, only some numerical results with 30 non-holonomic agents running at $100\,\text{ms}$ are reported. Notice that the simulations run with the same code as the experiment, this allows to debug the experimental code. Figure 4 reports the initial and final configurations for one sample of the simulations run. It can be noticed that the algorithm's performance is invariant with the number of agents; being totally decentralized, moreover, the computational complexity for each agent does not change with the group dimension while the overall computational complexity is obviously proportional to the number of agents. Finally, it has been observed that the agents reach a steady-state after a transient that is related to the group size due to the dynamic interaction among the agents themselves.

As a conclusive plot, figure 5 reports the final configu-

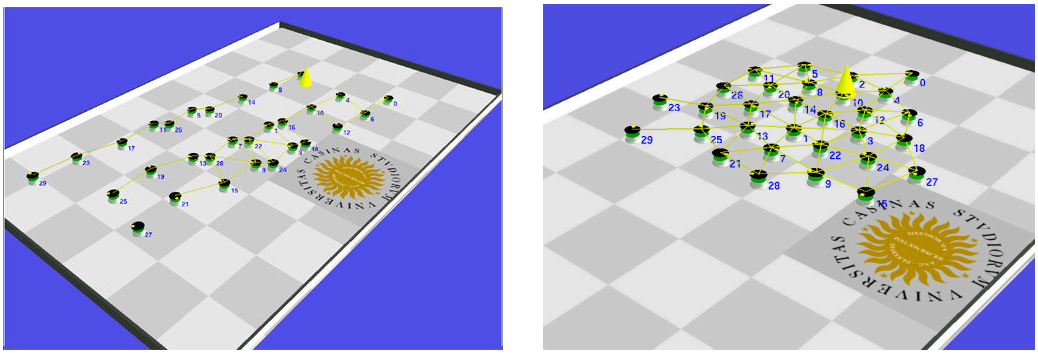

Fig. 4. Graphical representation of the initial and final configurations for a numerical simulation with 30 non-holonomic agents.

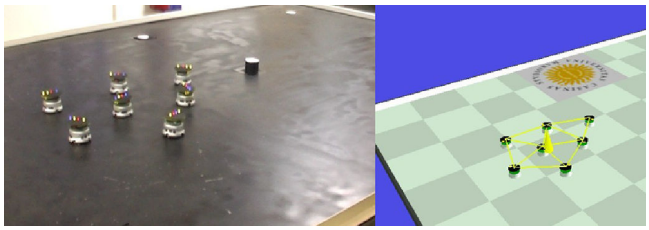

Fig. 5. Final configuration for a second experiment; the left frame shows a snapshot while the right frame the corresponding graphical elaboration.

ration for an additional experiment; the left frame shows a snapshot while the right frame the corresponding graphical elaboration.

VI. Conclusions

In recent years, the Null-Space-based Behavioral control approach has been applied for a wide range of robotic systems. In case of a multi-agent systems, e.g., problems such as formation control, escorting a target, or reconfiguration as a mobile ad-hoc network, have been successfully achieved. In this paper, the flocking problem of a group of dynamic agents has been addressed and solved with the NSB framework by defining very simple task functions. Flock in presence of a common rendez-vous point and/or in presence of obstacles has been discussed and verified by experiments performed with a team of 7 Khepera II mobile robots.

References

[1] http://www.k-team.com/. K-Team.

[2] *Matrox Imaging Library User Guide*. Matrox Electronic Systems Ltd, Canada, 2001.

[3] G. Antonelli, F. Arrichiello, and S. Chiaverini. The entrapment/escorting mission for a multi-robot system: Theory and experiments. In *Proceedings 2007 IEEE International Conference on Advanced Intelligent Mechatronics*, Zurich, CH, Sept. 2007.

[4] G. Antonelli, F. Arrichiello, and S. Chiaverini. An experimental study of the entrapment/escorting mission for a multi-robot system. *IEEE Robotics and Automation Magazine (RAM). Special Issues on Design, Control, and Applications of Real-World Multi-Robot Systems*, 15(1):22–29, March 2008.

[5] G. Antonelli, F. Arrichiello, and S. Chiaverini. The Null-Space-based Behavioral control for autonomous robotic systems. *Journal of Intelligent Service Robotics*, 1(1):27–39, online March 2007, printed January 2008.

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

[7] R.C. Arkin. Motor schema based mobile robot navigation. *The International Journal of Robotics Research*, 8(4):92–112, 1989.

[8] B.E. Bishop. On the use of redundant manipulator techniques for control of platoons of cooperating robotic vehicles. *IEEE Transactions on Systems, Man and Cybernetics*, 33(5):608–615, Sept. 2003.

[9] B.E. Bishop and D.J. Stilwell. On the application of redundant manipulator techniques to the control of platoons of autonomous vehicles. In *Proceedings 2001 IEEE International Conference on Control Applications*, pages 823–828, México City, MX, Sept. 2001.

[10] R.A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2:14–23, 1986.

[11] Y. Cao, A.S. Fukunaga, and A.B. Kanhg. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4:7–27, 1997.

[12] S. Chiaverini. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398–410, 1997.

[13] A.A. Maciejewski. Numerical filtering for the operation of robotic manipulators through kinematically singular configurations. *Journal of Robotic Systems*, 5(6):527–552, 1988.

[14] N. Mansard and F. Chaumette. Task sequencing for high-level sensor-based control. *IEEE Transactions on Robotics and Automation*, 23(1):60–72, 2007.

[15] S. Martinez, J. Cortes, and F. Bullo. Motion coordination with distributed information. *IEEE Control Systems Magazine*, 27(4):75–88, Aug. 2007.

[16] M.J. Mataric. Issues and approaches in the design of collective autonomous agents. *Robotics and Autonomous Systems*, 16(2):321–331, 1995.

[17] Y. Nakamura, H. Hanafusa, and T. Yoshikawa. Task-priority based redundancy control of robot manipulators. *The International Journal Robotics Research*, 6(2):3–15, 1987.

[18] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, March 2006.

[19] R. Olfati-Saber, J.A. Fax, and R.M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, Jan. 2007.

[20] G. Oriolo, A. De Luca, and M. Vendittelli. WMR control via dynamic feedback linearization: design, implementation, and experimental validation. *IEEE Transactions on Control Systems Technology*, 10(6):835–852, 2002.

[21] W. Ren, R.W. Beard, and E.M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, Apr. 2007.

[22] C. Reynolds. Flocks, herd and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34, 1987.

[23] B. Siciliano. Kinematic control of redundant robot manipulators: A tutorial. *Journal of Intelligent Robotic Systems*, 3:201–212, 1990.

[24] H. Tanner, A. Jadbabaie, and G.J. Pappas. Flocking in Fixed and Switching Networks. *IEEE Transactions on Automatic Control*, 52(5):863–868, May 2007.