Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots

Marcello Farina *, Andrea Perizzato, Riccardo Scattolini
Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy

Received 23 December 2014
Received in revised form
4 June 2015
Accepted 15 June 2015

1. Introduction

The development of algorithms for the distributed coordination and control of large scale interconnected or independent systems has received a great attention in recent years see, e.g., [1–6]. These problems are of particular interest in the robotic field, motivated by the widespread diffusion of autonomous robotic systems, which require efficient distributed control strategies for motion, coordination, and cooperation, see [7–9] and the references therein. Among the main problems of interest, it is possible to recall formation control, coverage and containment for sensing, obstacle and inter-robot collision avoidance, and border patrolling.

Distributed coordination includes a number of fundamental issues, such as *flocking* [8], *formation tracking and control* [9]. Formations can be represented in many ways, for example by means of *virtual structures* [10,11], using a *leader–follower* representation [12], or through the so-called *formation constraint function* [13]. These problems, including *obstacle and inter-robot collision avoidance*, are tackled with *potential functions*, *gradient methods* [14], linear feedback control laws [15], and *consensus* [16]. Other well-studied coordination control problems include *containment*, i.e., where a team of followers is guided by multiple leaders see, e.g., [17–19]. Also motion in unknown environments carries about challenging control issues, such as obstacle border patrolling (see, e.g., [20,21] and reference therein) while, in the context of sensor networks, one of the major issues is *coverage*, i.e., to deploy a set of sensors so as to maximize the overall sensing performance, see, e.g., [22–25] and the references therein.

* Correspondence to: Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, via Ponzio 34/5, 20133 Milan, Italy. Tel.: +39 0223993599; fax: +39 0223993412.
E-mail address: marcello.farina@polimi.it (M. Farina).

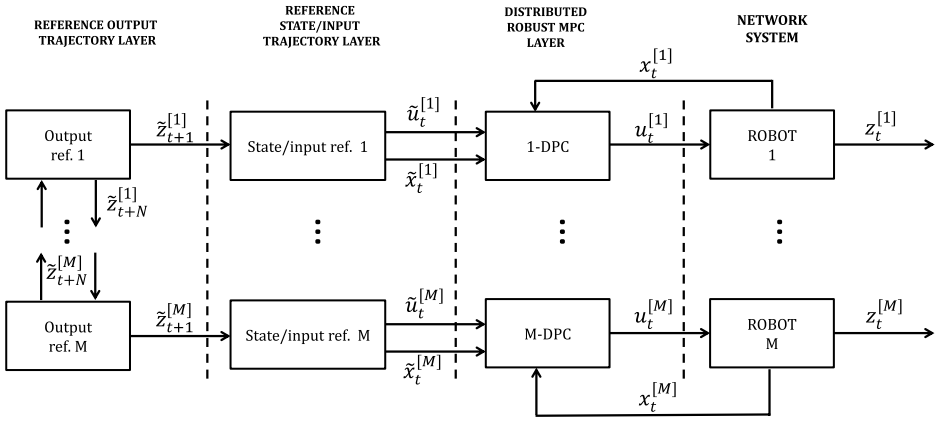

Fig. 1. The proposed multi-layer control architecture.

Despite the many results nowadays available, several issues still need to be addressed [9], if possible in a comprehensive fashion, such as the proper handling of constraints, the presence of input saturations, and the requirement of robustness. For these reasons, solutions based on robust Model Predictive Control (MPC) [3] appear to be promising. In fact constraints and limitations on the robot's dynamics and trajectory can be easily formulated as suitable constraints in the MPC optimization problem; distributed MPC solutions are currently available, see [1] and references therein, which, in a multi-robot environment, allow for the decomposition of the overall coordination control problem into smaller subproblems which can be locally solved on-board. This has significant computational and communication benefits with respect to centralized MPC solutions, and also confers flexibility and robustness to the overall system. However, the available MPC-based solutions guaranteeing robustness and collision avoidance properties (see e.g., [26,27], where notable real application experimental tests are also shown) are characterized by high dimensional and nonlinear optimization problems even in case of linear systems, which make them computationally demanding.

For all the above reasons, in this paper we propose a robust MPC approach for the solution of a number of motion and coordination problems. The method relies on the multilayer scheme shown in Fig. 1 and originally proposed in [2] for general interacting subsystems.

At the higher Reference Output Trajectory Layer of the control structure, and at any time instant t, an optimization problem is solved for any robot i to define the future reference trajectory $\{\tilde{z}_t^{[i]}, \ldots, \tilde{z}_{t+N}^{[i]}\}$ to be followed for the solution of a number of motion and coordination problems, including formation control, coverage and optimal sensing, containment control, inter-robot and obstacle collision avoidance, and motion problems in an unknown environment. At the intermediate Reference State/Input Trajectory Layer, the state and control trajectories $\{\tilde{x}_t^{[i]}, \ldots, \tilde{x}_{t+N}^{[i]}\}$ and $\{\tilde{u}_t^{[i]}, \ldots, \tilde{u}_{t+N}^{[i]}\}$ compatible with $\{\tilde{z}_t^{[i]}, \ldots, \tilde{z}_{t+N}^{[i]}\}$ are computed. Finally, at the lower Distributed Robust MPC Layer a robust DPC algorithm is solved to compute the robots' commands. Notably, at the intermediate and lower layers the algorithms are the same for all the considered problems. In the development of the method, effort has been devoted to state the optimization problems as quadratic ones, characterized by linear and computationally non-intensive constraints for collision and obstacle avoidance. This allows to cope with the usually limited on-board computational power and to enhance the reactiveness, in terms of reduction of sampling time, of the DPC algorithm.

The paper is organized as follows. In Section 2 it is shown how the unicycle robot model can be described using linear equations under a suitable feedback linearization procedure. Section 3 is the core of the paper: indeed, the obstacle and inter-robot collision avoidance problems are reformulated in terms of linear constraints, and the cost function to be minimized is customized in order to tackle different motion and coordination issues. Section 4 shortly describes the structure and the main characteristics of the DPC algorithm used at the lower layers, first proposed in [2], and here specifically tailored to the case of dynamically decoupled systems. In Section 5 we present the main theoretical result, regarding the recursive feasibility properties of the proposed control scheme. In Section 6 a sketch of the algorithm implementation is drawn and the choice of the main tuning knobs is thoroughly discussed. A number of simulation and experimental results in different scenarios, including formation control, optimal sensing, containment, border patrolling, are illustrated in Section 7. Finally, Section 8 draws some conclusions. The proofs of the main results are reported in the Appendix.

Notation. A matrix is *Schur* stable if all its eigenvalues lie in the interior of the unit circle. $\oplus$ and $\ominus$ denote the Minkowski sum and Pontryagin difference, respectively [28], while $\bigoplus_{i=1}^{M} A_i = A_1 \oplus \cdots \oplus A_M$. Where not specified, 2-norms are used, i.e., $\|\cdot\| = \|\cdot\|_2$; $\mathscr{B}_\varepsilon^{(dim)}(0) := \{x \in \mathbb{R}^{dim} : \|x\|_\infty \leq \varepsilon\}$ is a ∞-norm ball centered at the origin in the $\mathbb{R}^{dim}$ space. Given a two-dimensional vector $v = (v_x, v_y)$ on the Cartesian plane, the angle with respect to the x-axis is denoted by $\angle v$. For a discrete-time signal s_t and $a, b \in \mathbb{N}, a \leq b$, $(s_a, s_{a+1}, \ldots, s_b)$ is denoted with $s_{[a:b]}$. I_n is the $n \times n$ identity matrix.

2. The robots

2.1. Model of the unicycle robots

We consider a set of M unicycle robots, whose dynamics is

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \\ \dot{\theta} = \omega \\ \dot{v} = a \end{cases} \tag{1}$$

where v, a and ω are the linear velocity, acceleration and angular velocity, respectively, while (x, y, θ) denotes the position and orientation with respect to a fixed frame. The linear acceleration a and the angular velocity ω are the control inputs of the system.

2.2. Feedback linearization

A linear model of the robots is obtained with a feedback linearization procedure [29]. Defining $\eta_1 = x$, $\eta_2 = \dot{x}$, $\eta_3 = y$,

$\eta_4 = \dot{y}$, the dynamics resulting from (1) is

$$\dot{\eta}_1 = \eta_2 \tag{2a}$$
$$\dot{\eta}_2 = a\cos\theta - v\omega\sin\theta \tag{2b}$$
$$\dot{\eta}_3 = \eta_4 \tag{2c}$$
$$\dot{\eta}_4 = a\sin\theta + v\omega\cos\theta. \tag{2d}$$

Letting $a_x = a\cos\theta - v\omega\sin\theta$ and $a_y = a\sin\theta + v\omega\cos\theta$ we transform (2) into a set of two decoupled double integrators with inputs a_x and a_y. To recover (ω, a) from (a_x, a_y) we compute

$$\begin{bmatrix}\omega\\ a\end{bmatrix} = \frac{1}{v}\begin{bmatrix}-\sin\theta & \cos\theta\\ v\cos\theta & v\sin\theta\end{bmatrix}\begin{bmatrix}a_x\\ a_y\end{bmatrix}. \tag{3}$$

For obtaining (3) it is assumed that $v \neq 0$. This singularity point must be accounted for when designing the control law [29], as discussed in Section 6.3.

From this point on, we denote by superscript $(\cdot)^{[i]}$ the variables associated with the ith robot, $i \in \mathscr{M} = \{1, \ldots, M\}$. Being $x^{[i]} = (\eta_1^{[i]}, \ldots, \eta_4^{[i]})$ the state of the feedback-linearized model of the ith robot, its discrete-time dynamics is described by

$$x_{t+1}^{[i]} = A x_t^{[i]} + B u_t^{[i]} + w_t^{[i]} \tag{4}$$

where $A = \begin{bmatrix}1 & \tau & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & \tau\\ 0 & 0 & 0 & 1\end{bmatrix}$, $B = \begin{bmatrix}\frac{\tau^2}{2} & 0\\ \tau & 0\\ 0 & \frac{\tau^2}{2}\\ 0 & \tau\end{bmatrix}$, $u_t^{[i]} = \begin{bmatrix}a_{xt}\\ a_{yt}\end{bmatrix}$ and τ is the sampling period. The disturbance $w_t^{[i]}$ has been introduced to model uncertainties, possible discretization errors, and external disturbances. We assume that $w_t^{[i]} \in \mathbb{W}_i$, where $\mathbb{W}_i$ is a known bounded uncertainty set.

To limit the velocity and the position of the ith robot we impose that

$$x_t{}^{[i]} \in \mathbb{X}_i \tag{5}$$

where $\mathbb{X}_i$ is a convex set, for each $i \in \mathscr{M}$. The output variables are defined as the robot Cartesian coordinates, i.e.

$$z_t^{[i]} = C x_t^{[i]} \tag{6}$$

where $C = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}$. It can be readily verified that the triple (A, B, C) is reachable, observable, and does not have invariant zeros in $z = 1$.

3. Reference output trajectory generation

The fundamental feature of the algorithm for the solution of motion and coordination problems of the robots is the incremental design of their future reference trajectories guaranteeing the collision avoidance and the attainment of specific goals, such as formation control, coverage and optimal sensing, containment, patrolling. Specifically, at time t, assume that for each robot i, its reference trajectory $\tilde{z}_{[t:t+N-1]}^{[i]}$ is known over a future window of length N. Then, the value of $\tilde{z}_{t+N}^{[i]}$ is computed as the solution of the following optimization problem[1]

$$\min_{\tilde{z}_{t+N}^{[i]}} V_i^z \quad \text{subject to:} \tag{7}$$

– obstacle avoidance constraints

– inter-robot collision avoidance constraints

$$- \tilde{z}_{t+N}^{[i]} \in \tilde{z}_{t+N-1}^{[i]} \oplus \mathscr{B}_{\varepsilon_i}^{(2)}(0) \tag{8}$$

$$- \tilde{z}_{t+N}^{[i]} \in \mathbb{Z}_i \tag{9}$$

where the form of the cost function V_i^z depends on the specific problem, and will be discussed in the following paragraphs as well as the detailed definition of the obstacle and collision avoidance constraints. Constraint (8) is included to limit the speed of variation of the reference trajectory, and ε_i is a suitable tuning knob whose value influences all the parameters and sets required by the algorithm (for a discussion see the following Section 6.1). In general, large values of ε_i speed-up the robots movements, but can lead to infeasibility problems. Moreover, physical saturations can prevent the robot from achieving extreme speeds. Finally, constraint (9) is added to guarantee that no violation of the constraint set $\mathbb{X}_i$ can take place by the state trajectory of the robot. The definition of the set $\mathbb{Z}_i$ depends on the algorithms applied at the lower layers of the hierarchy, and as such will be given in the following Section 6.1 together with the method for its computation.

Constraint (8) implies (provided that the overall control in Fig. 1 is applied) the following result which shows that, if the traveling velocity – i.e., the difference of the reference position between successive time instants – is bounded, then there exists a computable time-invariant neighborhood $\mathscr{Y}_i$ of the output reference trajectory in which the robot's position is guaranteed to lie.

Proposition 1 (*The Proof is in* Appendix A)**.** *If* (8) *holds, then there exists a bounded and time-invariant set* $\mathscr{Y}_i$ *such that, for all* $t \geq 0$*:*

$$z_t{}^{[i]} \in \tilde{z}_t{}^{[i]} \oplus \mathscr{Y}_i. \quad \square \tag{10}$$

Letting

$$\delta_i = \max_{\delta z \in \mathscr{Y}_i} \|\delta z\| \tag{11}$$

be the maximum size of the uncertainty set $\mathscr{Y}_i$, in view of Proposition 1 we have that $\|\tilde{z}_{t+N}^{[i]} - z_{t+N}^{[i]}\| \leq \delta_i$. Details on the computation of $\mathscr{Y}_i$ are reported in Section 6.1.

3.1. Definition of the constraints for obstacle and inter-robot collision avoidance

In this subsection we formulate the obstacle and inter-robot collision avoidance requirements as linear constraints. Assume that there exist n^o obstacles and, for simplicity, that the hth obstacle is circular, centered at point z_h^o with radius R_h^o, although our scheme can be readily extended to obstacles with different shapes. Denoting by R_i the radius of the circle circumscribing the ith unicycle robot, in view of (10) and (11) it is easy to verify that the condition

$$\|\tilde{z}_{t+N}^{[i]} - z_h^o\| \geq R_i + R_h^o + \delta_i \tag{12}$$

guarantees that $\|z_{t+N}^{[i]} - z_h^o\| \geq R_i + R_h^o$, and hence collision between robot i and the hth obstacle are avoided, see Fig. 2. However, constraint (12) is nonlinear and non convex, and as such its inclusion into the optimization problem (7) would prevent the use of standard and fast quadratic programming techniques. Therefore, a linear approximation of (12) is used; to this end, consider a polytope $\mathscr{P}_{hi}^o$ circumscribing the circle centered at z_h^o, with radius $R_i + R_h^o + \delta_i$, defined by the set of r_{hi}^o linear inequalities $(h_k^{[o,hi]})^T(\tilde{z}_{t+N}^{[i]} - z_h^o) \leq d_{hi}^o$, with $k = 1, \ldots, r_{hi}^o$. Define $\rho_k^{[o,hi]}(t + N - 1) = (h_k^{[o,hi]})^T(\tilde{z}_{t+N-1}^{[i]} - z_h^o) - d_{hi}^o$ as the distance between the position $\tilde{z}_{t+N-1}^{[i]}$ and the kth side of the polytope defined above. Note that

[1] With some abuse of notation but for the sake of clarity and readability, we denote by $\tilde{z}_{t+N}^{[i]}$ both the argument and the result of the stated optimization problem.

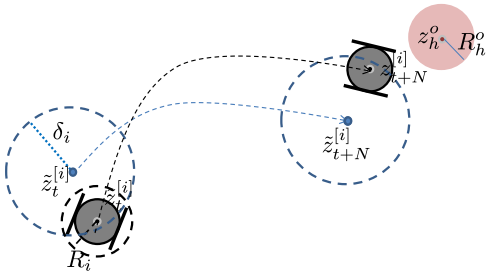

Fig. 2. Geometric illustration of condition (12). Robot: gray region; obstacle: pink region. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

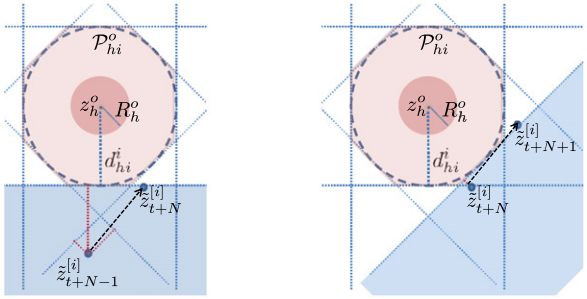

Fig. 3. Obstacle and its outer-polytopic approximation. Cyan region: area delimited by the selected linear constraint (13). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- if $\rho_k^{[o,hi]}(t+N-1) < 0$, then the points $\tilde{z}_{t+N-1}^{[i]}$ and z_h^o lie at the same side with respect to the line lying on the kth edge of the polytope;
- if $\rho_k^{[o,hi]}(t+N-1) > 0$ then, for at least one value of k, the minimum distance constraint (12) is verified.

Among all possible indices k, we select the one corresponding to the maximum value of $\rho_k^{[o,hi]}(t+N-1)$, i.e., $k_{max}^{[o,hi]}(t+N-1) = \text{argmax}_{k\in\{1,\dots,r_{hi}^o\}}\rho_k^{[o,hi]}(t+N-1)$ and, as illustrated by Fig. 3, we enforce obstacle avoidance for $\tilde{z}_{t+N}^{[i]}$ by including in the optimization problem the following linear constraint:

$$(h^{[ij]}_{k_{max}^{[o,hi]}(t+N-1)})^T(\tilde{z}_{t+N}^{[i]} - z_h^o) \geq d_{hi}^o. \tag{13}$$

The index $k_{max}^{[o,hi]}(t+N-1)$ is selected to guarantee to the robot the maximal freedom, compatibly with the presence of the obstacle, of moving towards its goal.

Concerning inter-robot collision avoidance, even if similar to the obstacle avoidance problem, it is a more challenging issue, since it involves coordination between pairs of robots moving on the plane. To solve this problem, we assume that the value $\tilde{z}_{t+N-1}^{[j]}$ is available to robot i and that $\tilde{z}_{t+N-1}^{[i]}$ is available to robot j. Collision avoidance is guaranteed if, at each instant, $\|z_{t+N}^{[i]} - z_{t+N}^{[j]}\| \geq R_i + R_j$. In turn, this condition is attained if

$$\|\tilde{z}_{t+N}^{[i]} - \tilde{z}_{t+N}^{[j]}\| \geq R_i + R_j + \delta_{ij} \tag{14}$$

where $\delta_{ij} = \max_{\delta z_i\in\mathscr{Y}_i}\|\delta z_i\| + \max_{\delta z_j\in\mathscr{Y}_j}\|\delta z_j\|$. However, requiring (14) is neither possible (since both $\tilde{z}_{t+N}^{[i]}$ and $\tilde{z}_{t+N}^{[j]}$ are optimization variables, managed by two different robots at the same time) nor sufficient for guaranteeing recursive feasibility. Therefore, as in the case of obstacle avoidance, we (out-) approximate the circle defined by (14) with the set of r_{ij} linear inequalities $(h_k^{[ij]})^T(\tilde{z}_{t+N}^{[i]} - \tilde{z}_{t+N}^{[j]}) \leq d_{ij}$, with $k = 1,\dots,r_{ij}$. We define $k_{max}^{[ij]}(t+N-1) = \text{argmax}_{k\in\{1,\dots,r_{ij}\}}(h_k^{[ij]})^T(\tilde{z}_{t+N-1}^{[i]} - \tilde{z}_{t+N-1}^{[j]}) - d_{ij}$ and $\rho_{max}^{[ij]}(t+N-1) = \max_{k\in\{1,\dots,r_{ij}\}}(h_k^{[ij]})^T(\tilde{z}_{t+N-1}^{[i]} - \tilde{z}_{t+N-1}^{[j]}) - d_{ij}$. Then, recalling that the optimization variables in (7) for robots i and j are $\tilde{z}_{t+N}^{[i]}$ and $\tilde{z}_{t+N}^{[j]}$, at time step t, the constraints to be fulfilled by the robots are

$$(h^{[ij]}_{k_{max}^{[ij]}(t+N-1)})^T(\tilde{z}_{t+N}^{[i]} - \tilde{z}_{t+N-1}^{[j]}) \geq \tilde{d}_{ij}(t+N-1) \tag{15}$$

$$(h^{[ij]}_{k_{max}^{[ij]}(t+N-1)})^T(\tilde{z}_{t+N-1}^{[i]} - \tilde{z}_{t+N}^{[j]}) \geq \tilde{d}_{ij}(t+N-1) \tag{16}$$

where $\tilde{d}_{ij}(t+N-1) = d_{ij} + \rho_{max}^{[ij]}(t+N-1)/2$. Hence, the collision avoidance problem boils down to the linear inequalities (15) and (16).

Note that the previous obstacle and collision avoidance constraints should be included in the optimization problem for any pair robot/obstacle and any pair of potentially colliding robots, correspondingly an all-to-all communication network supporting the exchange of information between the robots would be required. However, it is apparent that these constraints should be considered only among robots and obstacles which are "proximal" in a suitable metric. To this end, define the proximity radius for robot i with R_{pi}, and we say that the hth obstacle is proximal to robot i at time t if z_h^o lies in the circle centered in $\tilde{z}_{t+N-1}^{[i]}$ with radius R_{pi}. Similarly, the robot $j \in \mathscr{M}\setminus\{i\}$ is a proximal neighbor with respect to robot i at time t if $\tilde{z}_{t+N-1}^{[j]}$ lies in the circle centered in $\tilde{z}_{t+N-1}^{[i]}$ with radius R_{pi}. In view of this, for robot i at time t, we define the set of proximal obstacles and the set of proximal neighbors, i.e., $\mathscr{O}_i$ and $\mathscr{C}_i \subseteq \mathscr{M}\setminus\{i\}$, respectively, with which collision avoidance must be enforced. The cardinality of $\mathscr{O}_i$ is denoted n_i^o. For consistency, $j \in \mathscr{C}_i$ if and only if $i \in \mathscr{C}_j$. Since the robot positions is time-varying, also the sets $\mathscr{O}_i$ and $\mathscr{C}_i$ (along with their cardinality) are time varying, although this is not specifically indicated for notational simplicity. Finally remark that the communication between neighbors in $\mathscr{C}_i$ must be supported by a suitable (possibly time-varying) neighbor-to-neighbor communication network.

Finally, to allow for smooth operation and for avoiding "artificial" deadlocks, it is advisable that the number of sides of the polytopes circumscribing obstacles and robots (i.e., r_{hi} and r_{ij}, respectively) is large (e.g., greater than 10). Note that this does not impact on the computational complexity of the resulting optimization problems since, among all possible linear constraints, at each time instant only one is selected and enforced.

3.2. Definition of the cost function for motion and coordination tasks

In this section we discuss the different choices of the cost function V_i^z for the coordination problems considered, i.e. formation control, coverage and optimal sensing, containment, navigation and patrolling in an unknown environment.

3.2.1. Formation control

According to the so-called $l-\psi$ [30] formation control theory, the geometric relationship between a follower i and its leader j is expressed by a distance l_{ij}^d and an angle ψ_{ij}^d that the follower has to keep from the leader, as schematically illustrated in Fig. 4.

Consider M robots, $i \in \mathscr{M}$, denote by $\mathscr{N}_i$ the set of robots to be followed by robot i, and by $|\mathscr{N}_i|$ its cardinality. We assume that there exists only one *formation leader*, which without loss of generality is robot 1, so that $|\mathscr{N}_1| = 0$. Moreover, for each node $i \in \mathscr{M}\setminus\{1\}$ we define one "local" leader, with respect to which the position of robot i must be defined, so that $|\mathscr{N}_i| = 1$ for all $i \in \mathscr{M}\setminus\{1\}$.

The formation leader aims to attain the final goal $z_t \to z_{goal}$. In view of this, the cost function of the formation leader is

$$V_1^z = \gamma\left\|\tilde{z}_{t+N}{}^{[1]} - \tilde{z}_{t+N-1}{}^{[1]}\right\|^2 + \left\|\tilde{z}_{t+N}{}^{[1]} - z_{goal}\right\|_T^2 \tag{17}$$

where the weights T and γ are tuning knobs satisfying $T \succ \gamma I_2 \succ 0$.

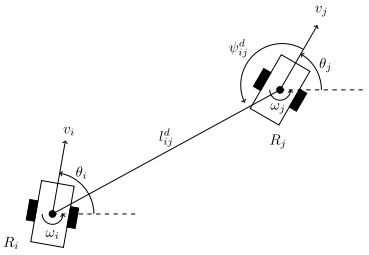

Fig. 4. $l-\psi$ formation.

Each follower $i \in \mathscr{M}\setminus\{1\}$ minimizes the following cost function, where j_{lead_i} is the index of the i's local leader, i.e., $j_{lead_i} \in \mathscr{N}_i$:

$$V_i^z = \gamma \left\| \tilde{z}_{t+N}{}^{[i]} - \tilde{z}_{t+N-1}{}^{[i]} \right\|^2 + \left\| \tilde{z}_{t+N}{}^{[i]} - z_{form}{}^{[i,j_{lead_i}]} \right\|_T^2 \tag{18}$$

where the set-points $z_{form}{}^{[i,j]}$ are defined as

$$z_{form}{}^{[i,j]} = \tilde{z}_{t+N-1}{}^{[j]} + l_{ij}^d \begin{bmatrix} \cos(\alpha_{t+N-1}^{[j]} + \psi_{ij}^d) \\ \sin(\alpha_{t+N-1}^{[j]} + \psi_{ij}^d) \end{bmatrix}$$

in which $\alpha_{t+N-1}^{[j]}$ defines the direction of the robot j and is computed as

$$\alpha_{t+N-1}^{[j]} = \angle(\tilde{z}_{t+N-1}{}^{[j]} - \tilde{z}_{t+N-2}{}^{[j]}).$$

In this way, the set-points are dynamically defined for each follower, based on the formation definition and on the position of its local leader.

3.2.2. Coverage and optimal sensing

Assuming that the robots are endowed with sensing capabilities, the aim of *coverage* is to define their optimal position in the polytopic and convex area $\mathbb{Q}$ which maximizes the overall sensing performance, under the following conditions: (i) data are generated in the regions of points $q \in \mathbb{Q}$ with some probability, defined by the probability density function $\phi : \mathbb{Q} \to \mathbb{R}_+$; (ii) the further a sensor is from the region where the data get generated, the weaker is its sensing performance. The solution here proposed is inspired by the method described in [31]; specifically, given M sensors placed at positions $\mathscr{S} = \{s_1, \ldots, s_M\}$, each designated to sense a region $W_i, i \in \mathscr{M}$, the scope of the coverage is to find $\mathscr{S}$ and a partition $\mathscr{W} = \{W_1, \ldots, W_M\}$ of $\mathbb{Q}$, that minimize the cost

$$J(\mathscr{S}, \mathscr{W}) = \sum_{i=1}^{M} \int_{W_i} \|q - s_i\|^2 \phi(q) dq.$$

On the one hand, given $\mathscr{S}$, the optimal partition of $\mathbb{Q}$ corresponds to its Voronoi partition, i.e. $\mathscr{W} = \mathscr{V}(\mathscr{S}) = \{V_1, \ldots, V_M\}$, where $V_i = \{q \in \mathbb{Q} : \|q - s_i\| \leq \|q - s_j\|, \forall j \neq i\}$. On the other hand, given a partition $\mathscr{W}$ of $\mathbb{Q}$, the optimal placement of the sensors minimizing $J(\mathscr{S}, \mathscr{W})$ corresponds to $s_i = C^i$, where C^i is the centroid of W_i. In view of this, the adopted cost function V_i^z is the following, for all $i \in \mathscr{M}$.

$$V_i^z = \gamma \left\| \tilde{z}_{t+N}{}^{[i]} - \tilde{z}_{t+N-1}{}^{[i]} \right\|^2 + \left\| \tilde{z}_{t+N}{}^{[i]} - C_{t+N-1}^i \right\|_T^2 \tag{19}$$

where $T \succ \gamma I_2 \succ 0$ and C_{t+N-1}^i is the centroid of the Voronoi set defined in $\mathbb{Q}$ and by the positions $\tilde{z}_{t+N-1}^{[j]}$, for all $j \in \mathscr{M}$. Note that the computation of C_{t+N-1}^i requires the knowledge of the vertices of $\mathbb{Q}$ and of a subset of positions $\tilde{z}_{t+N-1}^{[i]}$, i.e., those concerning the "sensing" neighbors of j, and therefore it is distributed.

The convergence (see [31]) of the previous algorithm may be very slow, and it can be convenient to use the cost functions

$$V_i^z = \gamma \left\| \tilde{z}_{t+N}{}^{[i]} - \tilde{z}_{t+N-1}{}^{[i]} \right\|^2 + \left\| \tilde{z}_{t+N}{}^{[i]} - C_{\infty}^i \right\|_T^2 \tag{20}$$

where $C^i\infty$ is computed offline through the following distributed iterative procedure:

(I) initialize s_0^i, for each $i \in \mathscr{M}$, as the position of the ith robot;
(II) for all $k \geq 0$ define C_k^i as the centroid of the ith Voronoi set defined in $\mathbb{Q}$ and by the positions $s_k^j, j \in \mathscr{M}$;
(III) define $k = k + 1$ and set $s_k^j = C_{k-1}^i$. Repeat from step (I) until, for a given threshold $\varepsilon_c > 0$, $\|s_k^j - s_{k-1}^j\| < \varepsilon_c$ for all $j \in \mathscr{M}$. Then set $C_\infty^i = C_k^i$ for all $i \in \mathscr{M}$.

3.2.3. Containment control

The containment and sensing control problem, see e.g., [17–19], consists in controlling a set of *followers* $\mathscr{N}_F$, in such a way that the followers are confined in a convex hull of the positions of a set of *leaders* $\mathscr{N}_L$. The aim of the leaders is to reach some predefined final positions. During their motion, the followers must lie in their convex hull but, as soon as the leaders reach their goal, they are bound to move towards their respective goals for carrying out sensing operations. According to [19], a robot $i \in \mathscr{N}_F$, with position $z^{[i]}$, is ε_F-contained in a set Ω if $d(z^{[i]}, \Omega) < \varepsilon_F$(with $\varepsilon_F > 0$), where $d(\cdot, \cdot)$ denotes the point-to-set distance $d(x, \Omega) = \zeta_\Omega(x) \min_{y\in\partial\Omega} \|x - y\|$, where $\zeta_\Omega(x) = 1$ if $x \in \Omega$ and $\zeta_\Omega(x) = -1$ otherwise, and $\partial\Omega$ denotes the boundary of Ω.

To guarantee that $z_{t+N}^{[i]}, i \in \mathscr{N}_F$, is ε_F-contained in the convex hull Ω_{t+N}^L of the positions $z_{t+N}^{[j]}, j \in \mathscr{N}_L$ of the leaders, we require that $\tilde{z}_{t+N}^{[i]}, i \in \mathscr{N}_F$, is $\tilde{\varepsilon}_F$-contained in the convex hull $\tilde{\Omega}_{t+N-1}^L$ of $\tilde{z}_{t+N-1}^{[j]}, j \in \mathscr{N}_L$. For a discussion on the choice of $\tilde{\varepsilon}_F$ and ε_F see Section 6.1.

Since not all the leaders transmit information to each of the followers, we guarantee that $\tilde{z}_{t+N}^{[i]}, i \in \mathscr{N}_F$, is $\tilde{\varepsilon}_F$-contained in the convex hull $\tilde{\Omega}_{t+N-1}^L$ of the positions $\tilde{z}_{t+N-1}^{[j]}, j \in \mathscr{N}_L$ by requiring that $\tilde{z}_{t+N}^{[i]}$ is $\tilde{\varepsilon}_F$-contained in the convex hull $\tilde{\Omega}_{t+N-1}^L(i)$ of the positions $\tilde{z}_{t+N-1}^{[j]}$ of a subset of leader, which can exchange information with i through a communication network. Such a subset of leaders of i is denoted by $\mathscr{N}_L(i)$.

In view of the discussion above, the goal of the leaders is to drive $z_t^{[i]} \to z_{goal}^{[i]}$, for all $i \in \mathscr{N}_L$. As such, for all $i \in \mathscr{N}_L$

$$V_i^z = \gamma \left\| \tilde{z}_{t+N}{}^{[i]} - \tilde{z}_{t+N-1}{}^{[i]} \right\|^2 + \left\| \tilde{z}_{t+N}{}^{[i]} - z_{t+N}^{*[i]} \right\|_T^2 \tag{21}$$

where $T \succ \gamma I_2 \succ 0$ and $z_{t+N}^{*[i]} = z_{goal}^{[i]}$ if $d(\tilde{z}_{t+N-1}^{[j]}, \tilde{\Omega}_{t+N-1}(j)) < \tilde{\varepsilon}_F$ for all $j \in \mathscr{N}_F$. Otherwise $\tilde{z}_{t+N}^{[i]} = \tilde{z}_{t+N-1}^{[i]}$. The latter is done when there is a least one follower which is not contained in its respective convex hull: in this case the leaders must stop moving to recover containment.

Secondly, each follower $i \in \mathscr{N}_F$ minimizes the cost function (21) where, to guarantee containment, $z_{t+N}^{*[i]} = \sum_{j\in\mathscr{N}_L(i)} k_{ij} \tilde{z}_{t+N-1}^{[j]}$ where k_{ij} are positive weights such that $\sum_{j\in\mathscr{N}_L(i)} k_{ij} = 1$. On the other hand, after the leaders have achieved their final positions, the goal of each follower switches to $z_{t+N}^{*[i]} = z_{goal}^{[i]}$, for all $i \in \mathscr{N}_F$ for sensing purposes. Note that, in turn, the positions $z_{goal}^{[i]}, i \in \mathscr{N}_F$, can be determined as in Section 3.2.2 according to optimal sensing criteria.

3.2.4. Control in an unknown environment and patrolling

We consider now the problem of navigation in an environment where only the main boundaries of the working area and both the robot and the goal positions are known. We assume that the robot is endowed with sensing capacities and must reach a given goal, but has no prior knowledge about the presence, position, size, or shape of the obstacles. More specifically, in the proximity of an obstacle, the robot is able to get a measurement of the position of a sequence of consecutive points of the obstacle's boundary, named $z_1^o, \ldots, z_{N_o}^o$. The moving robot has a visual capacity which depends

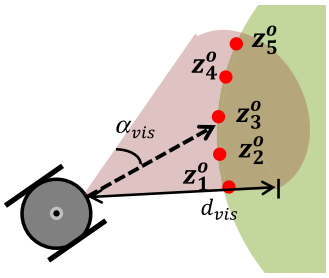

Fig. 5. Robot's visual field and measured points on the obstacle's boundary.

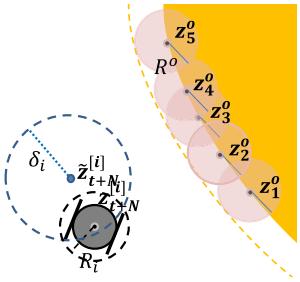

Fig. 6. Geometric illustration of border patrolling. Robot: gray region; obstacle: yellow region; pink regions: circles around the visible points on the obstacle's boundary. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of two main parameters, see also Fig. 5: the maximum distance d_{vis}, and the maximum visual angle α_{vis}.

If there is no obstacle in its visual range, the robot moves towards its goal, as previously discussed. As the obstacle appears in the its visual field, the robot carries out an obstacle border patrolling [20,21], until the object is no more in the space between the robot and the obstacle. At this point, the robot steers itself towards the goal.

The main rationale of our solution to the border patrolling problem consists in satisfying minimum-distance constraints with respect to the each point z_h^o but, at the same time, minimizing the deviation between the minimal distance allowed from the point z_h^o itself, for each $h = 1, \ldots, N_o$. This is done by a suitable definition of slack variables μ_h, $h = 1, \ldots, N_o$ for each point in the visual range. The ith robot minimizes the following cost function

$$V_i^z = \gamma \left\| \tilde{z}_{t+N}{}^{[i]} - \tilde{z}_{t+N-1}{}^{[i]} \right\|^2 + \sum_{h=1}^{N_o} \lambda^h \mu_h^2 \tag{22}$$

where $\gamma > 0$, $\lambda > 1$ are tuning parameters and, under the following constraints, for all $h = 1, \ldots, N_o$:

$$(h_{k_{max}^{[o,hi]}(t+N-1)}^{[o,hi]})^T (\tilde{z}_{t+N}^{[i]} - z_h^o) = d_{hi}^o + \mu_h \tag{23a}$$

$$\delta_h \geq 0 \tag{23b}$$

where $d_{hi}^o = R_i + \delta_i + R^o$, see Fig. 6, and where R^o is the minimum distance allowed from the point z_h^o. Similarly to Section 3.1, $k_{max}^{[o,hi]}(t+N-1) = \text{argmax}_{k \in \{1,\ldots,r_{hi}^o\}} (h_k^{[o,hi]})^T (\tilde{z}_{t+N-1}^{[i]} - z_h^o) - d_{hi}^o$ identifies a linear constraint, among the ones defining a region outside the polytope whose sides are d_{hi}^o-distant from z_h^o, with respect to which point $\tilde{z}_{t+N-1}^{[i]}$ is at maximum distance.

In a few words, the minimization of $\sum_{h=1}^{N_o} \lambda^h \mu_h^2$ under (23), forces the center of the robot i, i.e., $z_{t+N}^{[i]}$, to keep a minimum distance $R_i + R^o$ from the points z_h^o, $h = 1, \ldots, N_o$ but, at the same time, aims to reduce the distance of $z_{t+N}^{[i]}$ from the whole sequence $z_1^o, \ldots, z_{N_o}^o$. Also, in view of the fact that $\lambda > 1$, the robot is forced to minimize the distance μ_{N_o} to the point $z_{N_o}^o$ with higher priority: this induces the direction of the robot while circumnavigating the obstacle.

Specific solutions have been developed for dealing with acute angles, bottlenecks (e.g., between different but proximal objects), and for optimal switching strategies between the states *go towards goal* and *border patrolling*. For further details, see [32].

4. Reference state/input trajectory and distributed robust MPC layers

Once the reference output trajectory layer $\tilde{z}_{[t:t+N-1]}^{[i]}$ has been defined and incrementally updated as discussed in the previous section, compatible trajectories $\tilde{x}_{[t:t+N-1]}{}^{[i]}$ and $\tilde{u}_{[t:t+N-1]}{}^{[i]}$ for the states and inputs must be available to apply the final distributed MPC algorithm computing the control variables $u_t^{[i]}$ to be effectively used. To this end, consider the following dynamic system

$$\begin{bmatrix} \tilde{x}_{t+1}^{[i]} \\ \tilde{e}_{t+1}^{[i]} \end{bmatrix} = \begin{bmatrix} A & 0 \\ -C & I_2 \end{bmatrix} \begin{bmatrix} \tilde{x}_t^{[i]} \\ \tilde{e}_t^{[i]} \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \tilde{u}_t^{[i]} + \begin{bmatrix} 0 \\ I_2 \end{bmatrix} \tilde{z}_{t+1}{}^{[i]} \tag{24}$$

where the new state variable $\tilde{e}_{t+1}^{[i]}$ is the integral of the tracking error $\tilde{z}_{t+1}{}^{[i]} - C\tilde{x}_t^{[i]}$. We consider this form of tracking error to induce an anticipating action in the control scheme and to make it more reactive to time-changing references. In view of the reachability of the pair (A, B) and the absence of invariant zeros in $z = 1$ of model (4), for system (24) it is possible to compute the control law

$$\tilde{u}_t^{[i]} = \tilde{K}_x \tilde{x}_t^{[i]} + \tilde{K}_e \tilde{e}_t^{[i]} \tag{25}$$

where the gain $\tilde{K} = \begin{bmatrix} \tilde{K}_x & \tilde{K}_e \end{bmatrix}$ can be designed with any stabilizing algorithm, such as LQ or pole placement control.

Finally, at the Distributed Robust MPC Layer the control variables are computed to keep the real states $x^{[i]}$ and inputs $u^{[i]}$ as close as possible to their reference values $\tilde{x}^{[i]}$ and $\tilde{u}^{[i]}$, while respecting the constraints (5). To this end, define the nominal model

$$\hat{x}_{t+1}^{[i]} = A\hat{x}_t^{[i]} + B\hat{u}_t^{[i]} \tag{26}$$

and, as more thoroughly discussed in [2], the MPC optimization problem to be solved at any time instant by the ith robot is

$$\begin{aligned} \min_{\hat{x}_t^{[i]}, \hat{u}_{t:t+N-1}^{[i]}} & \sum_{h=0}^{N-1} \|\hat{x}_{t+h}^{[i]} - \tilde{x}_{t+h}^{[i]}\|_Q^2 + \|\hat{u}_{t+h}^{[i]} - \tilde{u}_{t+h}^{[i]}\|_R^2 \\ & + \|\hat{x}_{t+N}^{[i]} - \tilde{x}_{t+N}^{[i]}\|_P^2 \\ & \hat{x}_{t+1}^{[i]} = A\hat{x}_t^{[i]} + B\hat{u}_t^{[i]} \\ & \hat{x}_{t+h}^{[i]} \in \hat{\mathbb{X}}_i, \quad \forall h = 1, \ldots, N-1 \\ & x_t^{[i]} - \hat{x}_t^{[i]} \in \mathscr{E}_i \\ & C(\hat{x}_{t+h}^{[i]} - \tilde{x}_{t+h}^{[i]}) \in \Delta_i^z, \quad \forall h = 1, \ldots, N-1 \\ & \hat{x}_{t+N}^{[i]} - \tilde{x}_{t+N}^{[i]} \in \kappa_i \mathscr{E}_i \end{aligned} \tag{27}$$

where $\kappa_i > 0$ is a tuning parameter and the choice of the sets $\mathscr{E}_i$ and $\Delta_i^z \subseteq \mathbb{R}^2$ is discussed in Section 6.1. Moreover

$$\hat{\mathbb{X}}_i = \mathbb{X}_i \ominus \mathscr{E}_i. \tag{28}$$

The symmetric matrices $Q \geq 0$ and $R > 0$, weighting the state and the control error, respectively, are free design parameters, as in LQ control, while P is assumed to fulfill the Lyapunov equation

$$(A + BK)^T P (A + BK) - P = -(Q + K^T R K) \tag{29}$$

where K is selected in such a way that $(A + BK)$ is *Schur* stable. The solution to (27) is $(\hat{x}_{t|t}^{[i]}, \hat{u}_{[t:t+N-1]|t}^{[i]})$ and the actual control action to be applied to the robot is

$$u_t^{[i]} = \hat{u}_{t|t}^{[i]} + K(x_t^{[i]} - \hat{x}_{t|t}^{[i]}). \tag{30}$$

5. Main results

The following proposition proves the soundness of the scheme proposed in this paper. Here, for brevity, we only cope with obstacles of the type described in Section 3.1; however, similar arguments can be applied to the case of unknown obstacles, by treating the object's points in the robot visual range as center of suitable circular obstacles.

Proposition 2. *Assume that, at time step t, the optimization problem (7) is feasible and has solution $\tilde{z}^{[i]}_{t+N}$ for all $i \in \mathcal{M}$ such that* (a) *(13) is verified for all $h \in \mathcal{O}_i$;* (b) *(15) is verified for all $j \in \mathcal{C}_i$,* (c) $\tilde{z}^{[i]}_{t+N} \in \tilde{z}^{[i]}_{t+N-1} \oplus \mathcal{B}^{(2)}_{\varepsilon_i}(0)$. *Then:*

(i) *at time step t, collisions are prevented both with proximal obstacles $h \in \mathcal{O}_i$ and with proximal robots $j \in \mathcal{C}_i$ for all $i \in \mathcal{M}$;*
(ii) *at time step $t+1$ the optimization problem (7) is feasible for all $i \in \mathcal{M}$.* □

The proof of Proposition 2 is given in Appendix B.

Some final remarks are due. The present scheme guarantees recursive feasibility of the optimization problem in view of Proposition 2 and, as discussed in [2], convergence to the goal provided that deadlock solutions are not met. Strategies for overcoming deadlocks can be developed along the lines of [33].

It is also worth remarking that the linear obstacle and inter-robot collision avoidance constraints have little impact on the complexity of the optimization problem (7) of each robot. In fact, the number of constraints in (7) does not depend on the number of inequalities used to define the polytopes, but just on the number of objects to be avoided.

Finally note that, in a real application context, many disturbances, e.g., model uncertainties, discretization errors, external disturbances, and position/velocity measurement errors, may affect the system. In this paper robustness with respect to undesired disturbances has been conferred by explicitly considering the additive bounded noise terms $w^{[i]}_t$ in (4).

6. Design and implementation issues

6.1. Tuning and computation of the algorithm's parameters

In this section we discuss the different tuning knobs of the proposed scheme and how, based on their definition, the matrices and sets, required for the algorithm implementation, can be computed.

(1) **Inputs**: the system matrices A, B, C, the bounds $\mathbb{W}_i$ (see Eq. (4)), the operational constraints $\mathbb{X}_i$, and the geometric constants (i.e., the radius R_i of the circle circumscribing the ith robot) are given.
(2) **Control gains**: the matrices $\tilde{K}_e$ and $\tilde{K}_x$ must be defined in such a way that

$$\mathscr{F} = \begin{bmatrix} A & 0 \\ -C & I_2 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} \begin{bmatrix} \tilde{K}_x & \tilde{K}_e \end{bmatrix} \tag{31}$$

is Schur stable. Similarly, we define gain K in such a way that $A+BK$ is Schur stable. The set $\mathscr{E}_i$ is defined the robust positively invariant (RPI) set

$$\mathscr{E}_i = \bigoplus_{h=0}^{\infty} (A+BK)^h \mathbb{W}_i. \tag{32}$$

Methods for computing approximations of (32) are discussed in [28].
(3) **Tuning knobs**: the main tuning knobs are
- for each robot $i = 1, \ldots, M$, $\varepsilon_i > 0$, $\kappa_i > 0$, and the set Δ^z_i in (27). The latter must be defined as a (e.g., polytopic) set containing the origin in its interior.
- the weighting matrices Q and R in the cost function used in the robust MPC problem (27).
- the weights γ and T, verifying $T > \gamma I > 0$, for the definition of the cost function V^z_i in (17)–(22). In (22), also parameter $\lambda > 1$ must be defined.
- in the containment control problem discussed in Section 3.2.3, the containment level $\tilde{\varepsilon}_F > 0$ must be defined.

(4) **Compute** $\mathscr{Y}_i$ as follows.
- Compute $\bar{\mathbb{W}}_i = -\mathscr{F}(I_6 - \mathscr{F})^{-1} \mathscr{G} \mathscr{B}^{(2)}_{\varepsilon_i}(0)$ where

$$\mathscr{G} = \begin{bmatrix} 0 \\ I_2 \end{bmatrix}. \tag{33}$$

- Compute the RPI set $\Delta^X_i = \bigoplus_{h=0}^{\infty} \mathscr{F}^h \bar{\mathbb{W}}_i$.
- Compute $\mathscr{Z}_i = C\mathscr{E}_i \oplus \Delta^z_i$.
- Compute $\mathscr{Y}_i = C \begin{bmatrix} I_4 & 0 \end{bmatrix} \Delta^X_i \oplus \mathscr{Z}_i$.

(5) **Compute** $\hat{\mathbb{X}}_i$ according to (28).
(6) **Compute** $\mathbb{Z}_i$, used in (9), according to the following set inclusion

$$\begin{bmatrix} I_4 & 0 \end{bmatrix} \left((I_6 - \mathscr{F})^{-1} \mathscr{G} \mathbb{Z}_i \oplus \Delta^X_i \right) \oplus \kappa_i \mathscr{E}_i \subseteq \hat{\mathbb{X}}_i. \tag{34}$$

Concerning the choice of the main tuning knobs the following general considerations hold. First, the choice of Δ^z_i and ε_i is characterized by a trade-off. On the one hand, it is desirable to increase ε_i as much as possible (e.g., compatibly with the possible speed saturations) in order to enhance the robot allowed speed; also, a small size of Δ^z_i can have the effect of limiting the robustness of the control scheme by allowing only small deviations of the nominal state trajectory $\hat{x}^{[i]}_t$ with respect to the reference one $\tilde{x}^{[i]}_t$. On the other hand, the bigger ε_i and Δ^z_i, the bigger $\mathscr{Y}_i$ (and consequently δ_i), which implies that the size of the uncertainty set where the robot position is guaranteed to lie may increase. This has the undesirable effect of increasing the level of confidence required in the generation of the output reference trajectory; in the limit case, the inclusion (34) may not have a solution. Concerning this point, it is important to mention the fact that (34) admits a solution under the condition that ε_i is also compatible with (i.e., smaller than) the velocity saturation levels, i.e., the constraints on the velocities in the x and y directions, included in the tightened constraint set $\hat{\mathbb{X}}_i$.

In both the formation and containment control problems, it is suggested to allow the followers to move with greater speed than the formation leaders, to foster the attainment of the formation and the containment, respectively, also in transient conditions. This is achieved by setting $\varepsilon_i > \varepsilon_j$ for all $i \notin \mathcal{N}_L$ and $j \in \mathcal{N}_L$.

Regarding the containment control problem, it is important to remark that, the value of ε_F must be compatible with the fact that, in view of (8), $\|\tilde{z}^{[j]}_{t+N} - \tilde{z}^{[j]}_{t+N-1}\| \leq \sqrt{2}\varepsilon_j$ and that, in view of (10) and (11), $\|z^{[i]}_{t+N} - \tilde{z}^{[i]}_{t+N}\| \leq \delta_i$, therefore the parameter ε_F defining the containment performance is equal to $\varepsilon_F = \tilde{\varepsilon}_F + \delta_i + \max_{j \in \mathcal{N}_L} \sqrt{2}\varepsilon_j$. In conclusion, to enhance the performances of the containment scheme here proposed (which is equivalent to limit ε_F), it may be necessary to reduce the allowed speed of each robot (i.e., parameter ε_i in (8)) which, in turn, has the effect of reducing δ_i.

In the cost function V^z_i (e.g., in (17)–(21)) the values of γ and T trade between smoothness of the trajectory $\tilde{z}^{[i]}_t$ (in case γ takes relevant values) and speed of convergence to the goal (in case $T \gg \gamma I$). However, note that the latter objective is subject to a saturation due to constraint (8).

Finally, in the definition of the cost function (22) for border patrolling, as remarked in Section 3.2.4, relevant values of parameter $\lambda > 1$ increase the speed towards the last point $z^o_{N_o}$ in the visual range, reducing the smoothness of the traveled trajectory.

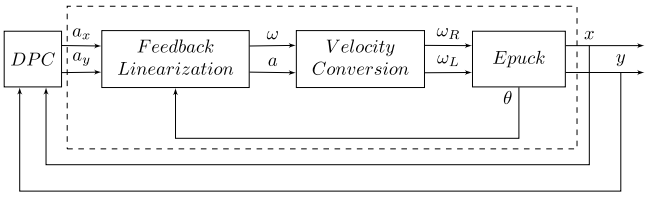

Fig. 7. Realization scheme.

6.2. Algorithm initialization

In view of the fact, at each time instant t, the output reference trajectory $\tilde{z}^{[i]}_{[t:t+N-1]}$ is assumed to be known, initialization is fundamental for our scheme. In this section we assume that the agents, at time $t = 0$, are not moving. Indeed, agent i is at initial position $z^{[i]}_o$. The following initialization procedure can be adopted.

(1) **Set** $\tilde{z}^{[i]}_{[0:(N-1)]} = \{z^{[i]}_o, \dots, z^{[i]}_o\}$.

(2) **Compute** $\tilde{\chi}^{[i]}_o = \begin{pmatrix} \tilde{x}^{[i]}_o \\ \tilde{e}^{[i]}_o \end{pmatrix} = (I_6 - \mathscr{F})^{-1}\mathscr{G} z^{[i]}_o$, see (31), (33).

(3) **Set** $\tilde{x}^{[i]}_{[0:(N-1)]} = \{\tilde{x}^{[i]}_o, \dots, \tilde{x}^{[i]}_o\}$.

6.3. Online implementation

The online implementation scheme for each robot, see Fig. 7, involves two different control loops, acting at different timescales.

6.3.1. External Control loop

The *external loop* is implemented according to Algorithm 1.

Algorithm 1 DPC-Control

At each time step $t > 0$:

Ia) Update the set of proximal obstacles $\mathscr{O}_i$
Ib) Update the set of proximal neighbors $\mathscr{C}_i$
Ic) Receive information from neighboring robots (see below).
Id) Solve problem (7), compute $\tilde{z}^{[i]}_{t+N}$, and update $\tilde{z}^{[i]}_{[t:(t+N-1)]}$
II) Update $\tilde{x}^{[i]}_{[t:(t+N-1)]}$ using system (25)
III) Solve problem (27) and compute $u^{[i]}_t$ as in (30)

Concerning step (**Ic**), note that the needed communication framework among robots is minimal. Indeed, only neighbor-to-neighbor communication is involved, and only when strictly required for the specific motion/coordination problem to be tackled. More specifically:

- For collision avoidance each robot must communicate predicted reference positions with all its proximal neighbors, i.e., $\tilde{z}^{[j]}_{t+N-1}$ for all $j \in \mathscr{C}_i$.
- For formation control, each robot $i \in \mathscr{M} \setminus \{1\}$ must receive the predicted reference position by it local leader(s), i.e., $\tilde{z}^{[j]}_{t+N-1}$, $j \in \mathscr{N}_i$.
- For coverage control, each robot must receive the predicted reference positions of its proximal neighbors, analogously to collision avoidance.
- For containment control purposes, each robot must receive the predicted reference positions of its local leaders, i.e., $\tilde{z}^{[j]}_{t+N-1}$, $j \in \mathscr{N}_L(i)$.

6.3.2. Internal control loop

The *internal loop*, implemented on-board, performs the feedback linearization procedure described in Section 2.2. The variables v and ω are derived from the linear system inputs a_x and a_y using (3). More specifically, the following recursive procedure can be used to compute v and ω, iterated – for better precision – with a small sampling period $\tau_{in} \ll \tau$:

1. at iteration $k = 0$, v is available from the estimator of the linear state-space model (computed as $\sqrt{\eta_2^2 + \eta_4^2}$) and θ is available directly from the measurement device;
2. while no new values of v and θ are available from the external loop, repeat

$$\omega^+ = (-a_x \sin\theta + a_y \cos\theta)/v \tag{35a}$$

$$v^+ = v + \tau_{in}(a_x \cos\theta + a_y \sin\theta) \tag{35b}$$

$$\theta^+ = \theta + \tau_{in}\omega \tag{35c}$$

where the superscript $^+$ is used to denote the values of the state variables at the subsequent iteration step.

The problems related to the singularity point $v = 0$ can be numerically circumvented by setting $v = \text{sign}(v)v_{min}$ in (35) when the absolute velocity falls below a given threshold, i.e., when $|v| < v_{min}$.

For completeness, note that the real inputs of the e-puck robots are the angular velocities ω_R and ω_L of the right and of the left wheels, respectively. We recover ω_R and ω_L from v and ω as follows:

$$\begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} = \frac{1}{R}\begin{bmatrix} 1 & L_{axle}/2 \\ 1 & -L_{axle}/2 \end{bmatrix}\begin{bmatrix} v \\ \omega \end{bmatrix} \tag{36}$$

where L_{axle} is the robot axle length and R is the wheel radius.

7. Application

Some of the previously proposed algorithms have been tested in simulation to consider a large number of robots, while others have been used to control a fleet of three robots in a real laboratory environment.

7.1. Description of the test bed and parameter tuning

The real robot demonstrations have been performed using three e-puck robots [34] (with axle length $L_{axle} = 5.2$ cm and wheel radius $R = 1.05$ cm) moving on a flat worktable (115×66 cm^2 wide) observed by a color camera (Microsoft LiveCam VX-800, resolution 640×364 px^2). According to the dimensions of the working area and the limitations on the wheel velocities, $\mathbb{X}_i$ is defined as (dimension in cm and cm/s, respectively):

$$\mathbb{X}_i = \begin{cases} 0 \le \eta_1^{[i]} \le 115 \\ -10 \le \eta_2^{[i]} \le 10 \\ 0 \le \eta_3^{[i]} \le 66 \\ -10 \le \eta_4^{[i]} \le 10. \end{cases}$$

Regarding the *external control loop*, each robot receives its position $z^{[i]}_t$ from the color camera. A state estimator (i.e., a suitable Kalman predictor) is then used to compute the estimates of the state variable $x^{[i]}_t$. The control action $u^{[i]}_t$ is computed according to the DPC control scheme, and is transmitted to the robots through a Bluetooth channel. The state estimator and the DPC algorithm are implemented in MATLAB. The time required for image capturing and processing is approximately 0.2 s while the pure computational effort requires about 0.1 s. Consistently with this, the sampling time of the outer control loop is about $\tau = 0.5$ s.

Regarding the *internal control loop*, the sampling period is set to $\tau_{in} = 0.01$ s.

In the experiments shown below, we set $\varepsilon_i = 1.5$ cm in case i is a follower and $\varepsilon_L = 1$ cm in case i is the formation leader in the

Please cite this article in press as: M. Farina, et al., Application of distributed predictive control to motion and coordination problems for unicycle autonomous robots, Robotics and Autonomous Systems (2015), http://dx.doi.org/10.1016/j.robot.2015.06.007

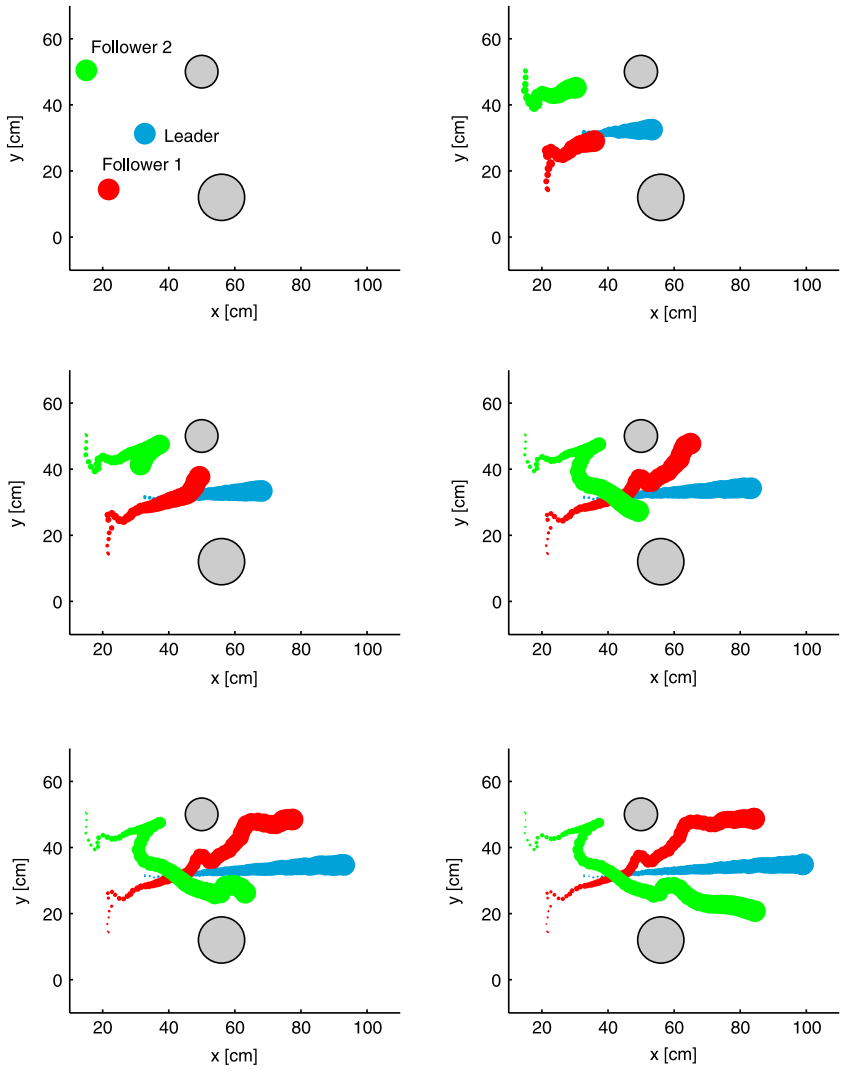

Fig. 8. Plots of the robot center trajectories of Experiment 1. Formation leader: blue dots, follower 1: red dots, follower 2: green dots. The gray-filled circles represent the obstacles. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

formation control example, or in case $i \in \mathcal{N}_L$ in the containment control example. Note that, recalling (8), $\varepsilon_L < \varepsilon_i$ makes the followers faster than the respective leaders: this is required for the followers to be able to attain the required formation while the leaders are still in motion.

We set $\mathbb{W}_i = \mathscr{B}^{(4)}_{0,1}(0)$, $\Delta^z_i = \mathscr{B}^{(2)}_{1,5}(0)$. Matrix K is the gain of an LQ regulator with $Q = I_4$ and $R = 200I_2$. We also set $\kappa_i = 1$.

7.2. Formation control: real experiment

In Experiment 1 (see Fig. 8) the three robots are initially placed at positions $(32.80, 31.28)$, $(21.82, 14.40)$, and $(15.10, 50.50)$, where all coordinates are expressed in cm. Two obstacles have been introduced: O_1 at $(62, 12)$ with radius 7 cm and O_2 at $(50, 50)$ with radius 5 cm. The desired formation is described by the following relationships:

$L \leftrightarrow F_1 : l^d_{L,F_1} = 20$ cm, $\quad \psi^d_{L,F_1} = 135°$

$L \leftrightarrow F_2 : l^d_{L,F_2} = 20$ cm, $\quad \psi^d_{L,F_2} = -135°$

resulting in a triangular shape.

7.3. Formation control: medium-scale simulation test

Experiment 2 has been performed in simulation in order to test the performance of the algorithm in a larger-scale set-up. The system configuration consists of one leader and 15 followers to be arranged in a circular fashion around the leader. Fig. 9 shows the robot positions at different time instants. As expected, when the robots reach the constraints, the formation gets modified to allow for obstacle avoidance. Furthermore, as the physical constraints become inactive (i.e., after passing in the bottleneck-shaped throat), the formation is restored.

7.4. Optimal sensing

In this section we perform a test concerning coverage and optimal sensing. The adopted cost function is the one reported in (20), where C^i_∞ have been computed using the procedure described in Section 3.2.2.

Since minimizing (20) corresponds to the problem of moving a set of robots towards their respective goals, which is a simplified version of the formation control previously-described, we do not describe complete test, but only the evolution of the results of the iterative procedure described in Section 3.2.2. In Fig. 10 we show the evolution of the Voronoi partitions of the set Q during the first 17 iterations of the procedure, while in Fig. 11 we show the Voronoi partitions of the working area when convergence is achieved.

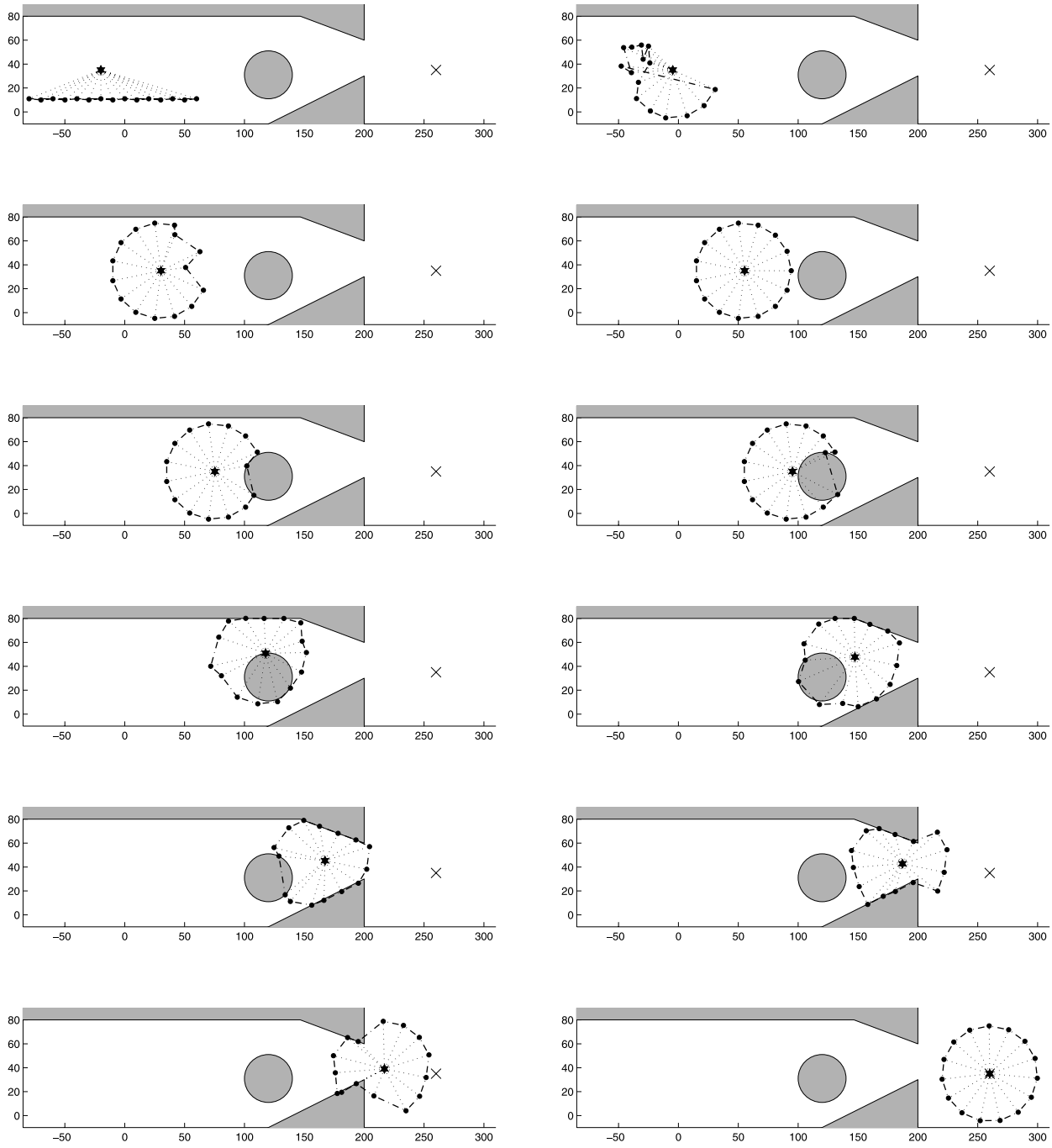

Fig. 9. Robot positions in Experiment 2.

7.5. Containment control: real experiment

In this section we consider six robots, where $\mathcal{N}_L = 1, 2, 3, 4$ and $\mathcal{N}_F = 5, 6$. For practical reasons, the leaders are just virtually implemented, while the followers are real robots. We set $\mathcal{N}_L(1) = \{1, 3, 4\}$ and $\mathcal{N}_L(2) = \{1, 2, 3\}$ and $k_{ij} = \frac{1}{3}$ for all i, j. In Fig. 12 we show the result of the experiment, where the goal of the leaders is twofold: first move towards the vertices of the region $\mathcal{D}_1$; secondly, after the followers reach the respective sensing positions (see again Fig. 12), move towards the vertices of region $\mathcal{D}_2$. The followers, when the leaders are in motion, keep their position in the prescribed convex hull of their respective leaders while, after the leaders reach their goal positions, move towards the sensing coordinates.

7.6. Exploration and border patrolling: real experiment

In this section we consider, for simplicity, one e-puck robot and three obstacles. If we consider the obstacle configuration in Fig. 13, it is easy to see that deadlocks (i.e., local minima) can be encountered when applying pure collision avoidance strategies (as in Section 3.1) with respect to both the circular obstacles. To avoid this, the border patrolling method described in Section 3.2.4 has been successfully adopted.

8. Conclusions

The multi-level DPC scheme proposed in this paper is characterized by some features which make it of interest in the framework

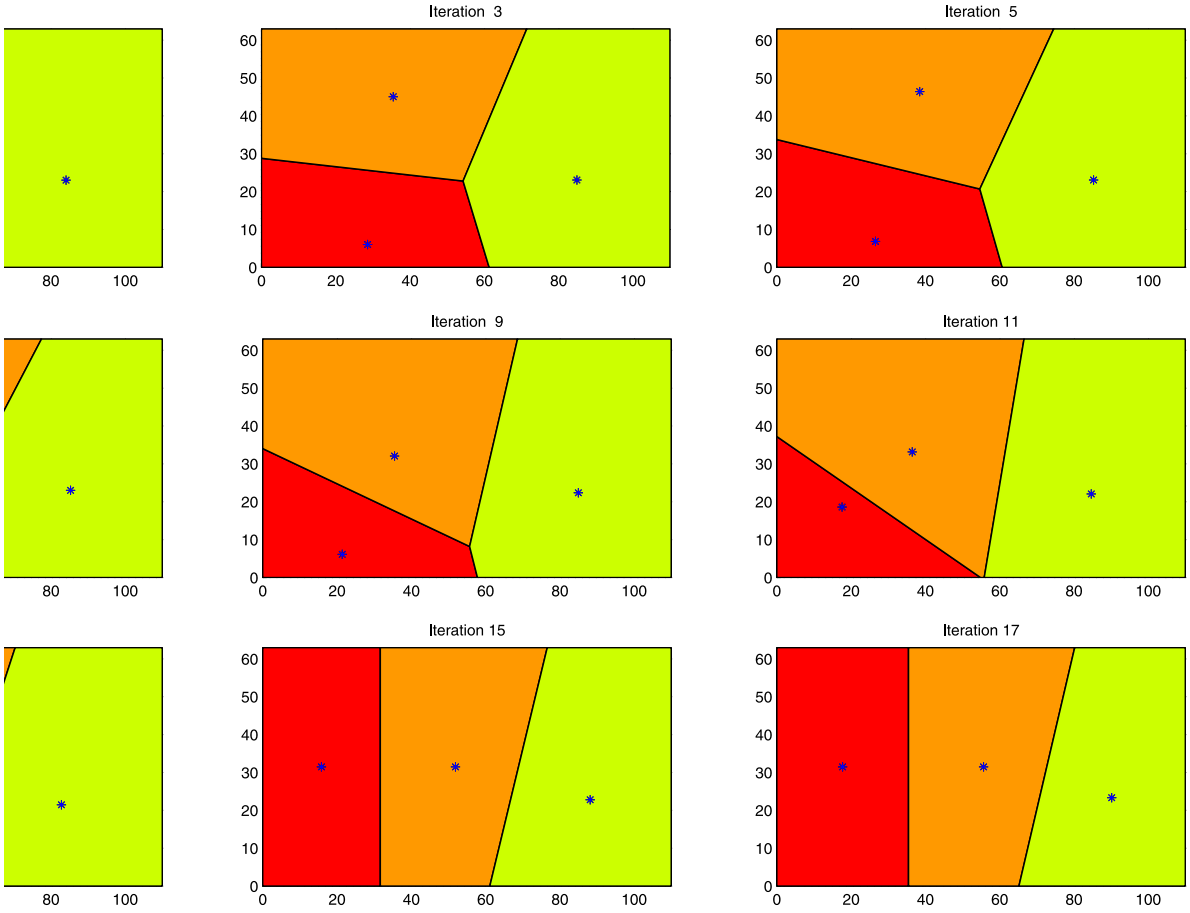

Fig. 10. Evolution of the Voronoi partition of the working area.

of motion and coordination of mobile robots: (i) it is very flexible, so that different goals and behaviors can be obtained by simply incorporating suitable terms in the optimization problem at the output reference trajectory generation level; (ii) it is robust, since it is based on a robust MPC method; (iii) the related computational burden is limited, since standard quadratic, small-size, and scalable optimization problems with linear constraints must be solved online; (iv) it requires limited neighbor-to-neighbor communication among on-board controllers. Future developments will consist in considering a stochastic framework, where uncertain probabilistic inputs can represent noises or describe robots' interactions in a less conservative way than the one here adopted.

Acknowledgments

The authors are indebted with Luca Giulioni and Marco Vergani for fruitful discussions.

Appendix A. Proof of Proposition 1 and definition of the sets $\mathscr{Y}_i$

The proof of Proposition 1 is constructive, and is based on the overall control algorithm described in Sections 3 and 4. In the following analysis the superscript $(\cdot)^{[i]}$ and the subscripts $(\cdot)_i$ introduced in Section 4 will be dropped to simplify the notation.

The first step is to find the maximum error between the output reference trajectory $\tilde{z}_t$ defined by the first layer and the corresponding output of the reference trajectory for the state $C\tilde{x}_t$. In fact, they are asymptotically equal only in case of $\tilde{z}_t$ constant, because of the integrator in the second layer. Such layer consists in Eq. (24), which can be written in compact form as

$$\tilde{\chi}_{t+1} = \mathscr{F}\tilde{\chi}_t + \mathscr{G}\tilde{z}_{t+1} \tag{37}$$

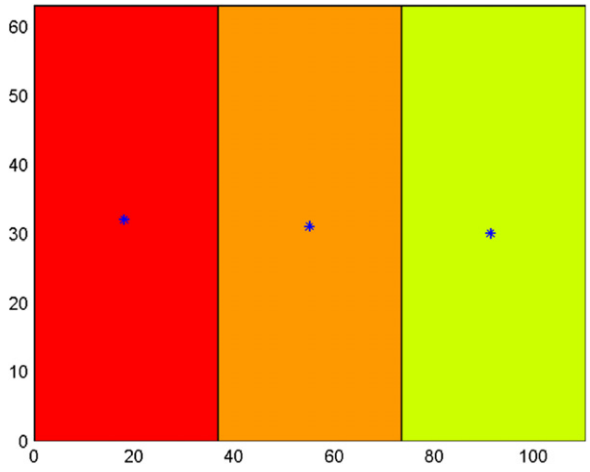

Fig. 11. Stationary Voronoi partition of the working area.

where $\tilde{\chi}_t = \begin{bmatrix}\tilde{x}_t & \tilde{e}_t\end{bmatrix}^T$. Let $\tilde{\chi}_t^{SS}$ be the steady state value of the enlarged state $\tilde{\chi}_t$ when the input is constant and equal to $\tilde{z}_t$:

$$\tilde{\chi}_t^{SS} = \mathscr{F}\tilde{\chi}_t^{SS} + \mathscr{G}\tilde{z}_t. \tag{38}$$

Therefore, we want to find a relationship between $\tilde{\chi}_t$ and the corresponding steady state value $\tilde{\chi}_t^{SS}$. Consider the following difference:

$$\tilde{\chi}_{t+1}^{SS} - \tilde{\chi}_t^{SS} = \mathscr{F}(\tilde{\chi}_{t+1}^{SS} - \tilde{\chi}_t^{SS}) + \mathscr{G}(\tilde{z}_{t+1} - \tilde{z}_t)$$

which leads to

$$\tilde{\chi}_{t+1}^{SS} - \tilde{\chi}_t^{SS} = (I_6 - \mathscr{F})^{-1}\mathscr{G}(\tilde{z}_{t+1} - \tilde{z}_t). \tag{39}$$

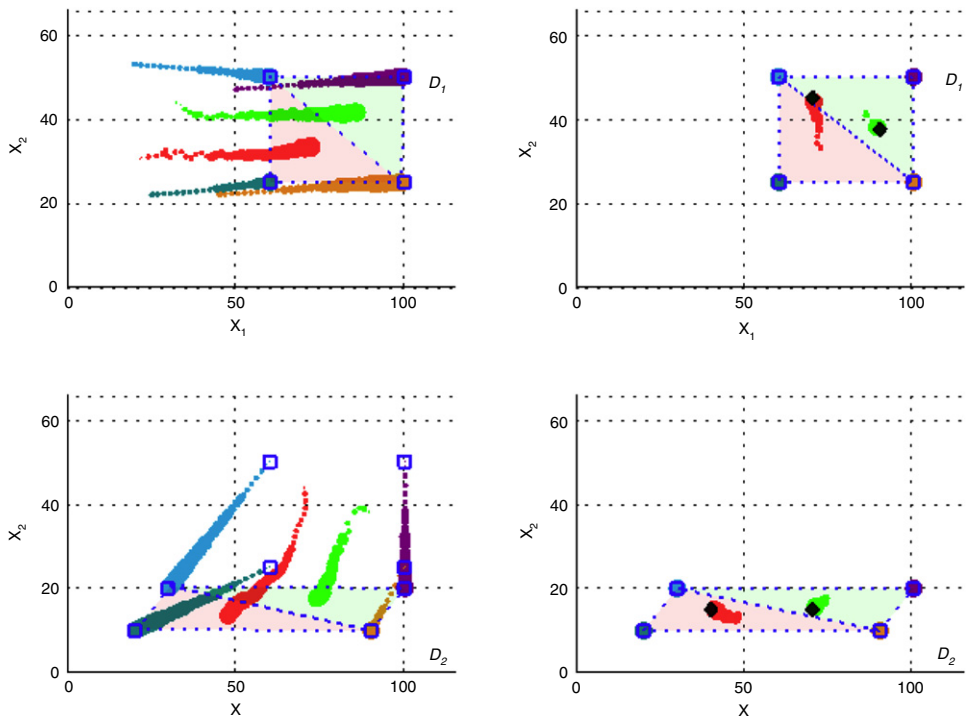

Fig. 12. Containment control problem. Trajectories of the centers of robots 1 (blue dots), 2 (purple dots), 3 (brown dots), 4 (dark green dots), 5 (red dots), 6 (green dots). Blue squares denote the vertices of the regions $\mathscr{D}_1$ and $\mathscr{D}_2$; the light red region denotes $\Omega_L(1)$, while the light red region denotes $\Omega_L(2)$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

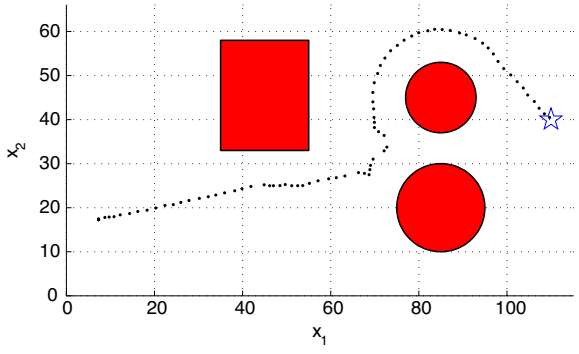

Fig. 13. Trajectory of the center of the robot. Blue star: goal. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We can now compute the dynamic equation of the error between the state $\tilde{\chi}_t$ and its corresponding steady state value $\tilde{\chi}_t^{SS}$:

$$\begin{aligned}\tilde{\chi}_{t+1} - \tilde{\chi}_{t+1}^{SS} &= \mathscr{F}(\tilde{\chi}_t - \tilde{\chi}_{t+1}^{SS}) \\ &= \mathscr{F}(\tilde{\chi}_t - \tilde{\chi}_t^{SS}) - \mathscr{F}(\tilde{\chi}_{t+1}^{SS} - \tilde{\chi}_t^{SS}) \\ &= \mathscr{F}(\tilde{\chi}_t - \tilde{\chi}_t^{SS}) + \tilde{w}_t \end{aligned} \tag{40}$$

where $\tilde{w}_t$ can be seen as a disturbance acting on the error. Recalling Eqs. (39), the domain of $\tilde{w}_t$ can be easily found:

$$\begin{aligned}\tilde{w}_t &= -\mathscr{F}(\tilde{\chi}_{t+1}^{SS} - \tilde{\chi}_t^{SS}) \\ &= -\mathscr{F}(I_6 - \mathscr{F})^{-1}\mathscr{G}(\tilde{z}_{t+1} - \tilde{z}_t) \in \tilde{\mathbb{W}}.\end{aligned}$$

Therefore, in view of (40) and according to [2], since $\mathscr{F}$ is *Schur* stable, there exists a robust positive invariant set (RPI) for the error $\tilde{\chi}_t - \tilde{\chi}_t^{SS}$, i.e., Δ^χ Therefore, by definition of RPI set and recalling the definition of $\tilde{\chi}_t$, we can write:

$$\tilde{\chi}_t \in \tilde{\chi}_t^{SS} \oplus \Delta^\chi \Rightarrow \tilde{x}_t \in \tilde{x}_t^{SS} \oplus \begin{bmatrix} I_4 & 0 \end{bmatrix} \Delta^\chi.$$

Moreover, since $C\tilde{x}_t^{SS} = \tilde{z}_t$, by simple operations we get

$$C\tilde{x}_t \in \tilde{z}_t \oplus C \begin{bmatrix} I_4 & 0 \end{bmatrix} \Delta^\chi \tag{41}$$

resulting in a relationship between the output reference trajectory and the output variables associated with the reference trajectory for the state, i.e. between $\tilde{z}_t$ and $C\tilde{x}_t$.

In order to relate the output reference trajectory to the actual output of the system, the third level of the control architecture has to be analyzed. As described in Section 4, the robust input u_t is obtained through an optimization problem based on the nominal system (26). We define the error

$$\varepsilon_t = x_t - \hat{x}_t$$

whose dynamics is described by

$$\varepsilon_{t+1} = (A + BK)\varepsilon_t + w_t$$

where the disturbance $w_t \in \mathbb{W}$ has been previously defined in Eq. (4). Therefore, since the matrix $(A + BK)$ is *Schur* stable, there exists [2] a RPI set for such error $\mathscr{E}$, defined in (32), which leads to the following constraint:

$$x_t \in \hat{x}_t \oplus \mathscr{E}. \tag{42}$$

As discussed in [2], the optimization problem is also subject to the following constraint:

$$C\hat{x}_t \in C\tilde{x}_t \oplus \Delta^z. \tag{43}$$

Combining the constraints (42) and (43) gives

$$Cx_t \in C\tilde{x}_t \oplus C\mathscr{E} \oplus \Delta^z \in C\tilde{x}_t \oplus \mathscr{Z}.$$

Finally, by applying the constraint (41) we get

$$z_t = Cx_t \in \tilde{z}_t \oplus \mathscr{Y}.$$

Appendix B. Proof of Proposition 2

Concerning result (i), since (13) is verified, and recalling that it is set $\tilde{z}_{t+N}^{[i]} = \bar{z}_{t+N|t}^{[i]}$, it holds that, for all $h \in \mathscr{O}_i$, $R_i + R_h^o + \delta_i \leq$

$\|\tilde{z}^{[i]}_{t+N} - z^o_h\| \le \|z^{[i]}_{t+N} - z^o_h\| + \|z^{[i]}_{t+N} - \tilde{z}^{[i]}_{t+N}\| \le \|z^{[i]}_{t+N} - z^o_h\| + \delta_i$, which implies that $\|z^{[i]}_{t+N} - z^o_h\| \ge R_i + R^o_h$, and hence collision prevention.

Consider now a proximal robot $j \in \mathscr{C}_i$. From (15)–(16)

$$
\begin{aligned}
&(h^{[ij]}_{k^{[ij]}_{max}(t+N)})^T(\bar{z}^{[i]}_{t+N} - \tilde{z}^{[j]}_{t+N}) \ge (h^{[ij]}_{k^{[ij]}_{max}(t+N-1)})^T(\bar{z}^{[i]}_{t+N} - \tilde{z}^{[j]}_{t+N}) \\
&= (h^{[ij]}_{k^{[ij]}_{max}(t+N-1)})^T(\bar{z}^{[i]}_{t+N} - \tilde{z}^{[j]}_{t+N-1}) + (h^{[ij]}_{k^{[ij]}_{max}(t+N-1)})^T(\tilde{z}^{[i]}_{t+N-1} \\
&\quad - \tilde{z}^{[j]}_{t+N}) - (h^{[ij]}_{k^{[ij]}_{max}(t+N-1)})^T(\tilde{z}^{[i]}_{t+N-1} - \tilde{z}^{[j]}_{t+N-1}) \\
&\ge 2(d_{ij} + \rho^{[ij]}_{max}(t+N-1)/2) \\
&\quad - (d_{ij} + \rho^{[ij]}_{max}(t+N-1)) = d_{ij}. \qquad (44)
\end{aligned}
$$

This guarantees that $\rho^{[ij]}_{max}(t+N) \ge 0$, and hence that $R_i + R_j + \delta_{ij} \le \|\tilde{z}^{[i]}_{t+N} - \tilde{z}^{[j]}_{t+N}\| \le \|z^{[i]}_{t+N} - z^{[i]}_{t+N}\| + \delta_{ij}$, which implies that $\|z^{[i]}_{t+N} - z^{[j]}_{t+N}\| \ge R_i + R_j$, and hence collision prevention.

Concerning result (ii), to guarantee feasibility of the optimization problem (7) at time $t+1$ it is sufficient to define a solution $\bar{z}^{[i]}_{t+N+1}$ fulfilling all the constraints. Let us set $\bar{z}^{[i]}_{t+N+1} = \tilde{z}^{[i]}_{t+N}$. First note that, by definition, $\bar{z}^{[i]}_{t+N+1} \in \tilde{z}^{[i]}_{t+N} \oplus \mathscr{B}^{(2)}_{\varepsilon_i}(0)$. Furthermore, note that $(h^{[ij]}_{k^{[o,hi]}_{max}(t+N)})^T(\bar{z}^{[i]}_{t+N+1} - z^o_h) = (h^{[ij]}_{k^{[o,hi]}_{max}(t+N)})^T(\tilde{z}^{[i]}_{t+N} - z^o_h) \ge (h^{[ij]}_{k^{[o,hi]}_{max}(t+N-1)})^T(\tilde{z}^{[i]}_{t+N} - z^o_h) \ge d^o_{hi}$, which proves that (13) is also verified at time step $t+1$. Regarding (15) $(h^{[ij]}_{k^{[ij]}_{max}(t+N)})^T(\bar{z}^{[i]}_{t+N+1} - \tilde{z}^{[j]}_{t+N}) = (h^{[ij]}_{k^{[ij]}_{max}(t+N)})^T(\tilde{z}^{[i]}_{t+N} - \tilde{z}^{[j]}_{t+N}) = d_{ij} + \rho^{[ij]}_{max}(t+N) \ge d_{ij} + \rho^{[ij]}_{max}(t+N)/2$ since $\rho^{[ij]}_{max}(t+N) \ge 0$ in view of (44). □

References

[1] R. Negenborn, J.M. Meastre (Eds.), Distributed MPC Made Easy, Springer, 2014.
[2] M. Farina, G. Betti, L. Giulioni, R. Scattolini, An approach to distributed predictive control for tracking-theory and applications, IEEE Trans. Control Syst. Technol. 22 (4) (2014) 1558–1566.
[3] J.B. Rawlings, D.Q. Mayne, Model Predictive Control: Theory and Design, Nob Hill Publishing, 2009.
[4] J. Liu, D. Muñoz de la Peña, P.D. Christofides, Distributed model predictive control of nonlinear process systems, AIChE J. 55 (5) (2009) 1171–1184.
[5] J. Liu, D. Muñoz de la Peña, P.D. Christofides, Distributed model predictive control of nonlinear systems subject to asynchronous and delayed measurements, Automatica 46 (1) (2010) 52–61.
[6] J. Liu, X. Chen, D. Muñoz de la Peña, P.D. Christofides, Sequential and iterative architectures for distributed model predictive control of nonlinear process systems, AIChE J. 56 (8) (2010) 2137–2149.
[7] R.M. Murray, Recent research in cooperative control of multivehicle systems, ASME J. Dyn. Syst. Meas. Control 129 (2007) 571–583.
[8] R. Olfati-Saber, Flocking in multi-agent dynamic systems: algorithms and theory, IEEE Trans. Automat. Control 51 (3) (2006) 401–420.
[9] Y. Cao, W. Yu, W. Ren, G. Chen, An overview of recent progress in the study of distributed multi-agent coordination, IEEE Trans. Ind. Inf. 9 (1) (2013) 427–438.
[10] M.A. Lewis, K.-H. Tan, High precision formation control of mobile robots using virtual structures, Auton. Robots 4 (4) (1997) 387–403.
[11] A. Sadowska, T. van den Broek, H. Huijberts, N. van de Wouw, D. Kostic, H. Nijmeijer, A virtual structure approach to formation control of unicycle mobile robots using mutual coupling, Internat. J. Control 84 (11) (2011) 1886–1902.
[12] Y.-Q. Chen, Z. Wang, Formation control: a review and a new consideration, in: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005. IROS 2005, 2005, pp. 3181–3186.
[13] M. Egerstedt, X. Hu, Formation constrained multi-agent control, IEEE Trans. Robot. Automat. 17 (6) (2001) 947–951.
[14] P. Ögren, E. Fiorelli, N.E. Leonard, Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment, IEEE Trans. Automat. Control 49 (8) (2004) 1292–1302.
[15] H.G. Tanner, G.J. Pappas, V. Kumar, Leader-to-formation stability, IEEE Trans. Robot. Autom. 20 (2003) 443–455.
[16] Z. Li, X. Lui, W. Ren, L. Xie, Distributed tracking control for linear multiagent systems with a leader of bounded unknown input, IEEE Trans. Automat. Control 58 (2) (2013) 518–523.
[17] Y. Cao, W. Ren, M. Egerstedt, Distributed containment control with multiple stationary or dynamic leaders in fixed and switching directed networks, Automatica 48 (8) (2012) 1586–1597.
[18] Y. Cao, D. Stuart, W. Ren, Z. Meng, Distributed containment control for multiple autonomous vehicles with double-integrator dynamics: Algorithms and experiments, IEEE Trans. Control Syst. Technol. 19 (4) (2011) 929–938.
[19] L. Galbusera, G. Ferrari-Trecate, R. Scattolini, A hybrid model predictive control scheme for containment and distributed sensing in multi-agent systems, Systems Control Lett. 62 (5) (2013) 413–419.
[20] A.S. Matveev, H. Teimoori, A.V. Savkin, A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance, Automatica 47 (3) (2011) 515–524.
[21] A.S. Matveev, C. Wang, A.V. Savkin, Real-time navigation of mobile robots in problems of border patrolling and avoiding collisions with moving and deforming obstacles, Robot. Auton. Syst. 60 (6) (2012) 769–788.
[22] Y. Ru, S. Martínez, Coverage control in constant flow environments based on a mixed energy metric, Automatica 49 (9) (2013) 2632–2640.
[23] A. Ghosh, S.K. Das, Coverage and connectivity issues in wireless sensor networks: A survey, Pervasive Mob. Comput. 4 (3) (2008) 303–334.
[24] B. Wang, H.B. Lim, D. Ma, A survey of movement strategies for improving network coverage in wireless sensor networks, Comput. Commun. 32 (13–14) (2009) 1427–1436.
[25] M. Cardei, J. Wu, Energy-efficient coverage problems in wireless ad-hoc sensor networks, Comput. Commun. 29 (4) (2006) 413–420.
[26] P. Trodden, A. Richards, Multi-vehicle cooperative search using distributed model predictive control, in: AIAA Guidance, Navigation and Control Conference and Exhibit, 2008.
[27] Y. Kuwata, A. Richards, T. Schouwenaars, J.P. How, Distributed robust receding horizon control for multivehicle guidance, IEEE Trans. Control Syst. Technol. 15 (4) (2007) 627–641.
[28] S.V. Raković, E.C. Kerrigan, K.I. Kouramas, D.Q. Mayne, Invariant approximations of the minimal robust positively invariant set, IEEE Trans. Automat. Control 50 (3) (2005) 406–410.
[29] G. Oriolo, A. De Luca, M. Vendittelli, WMR control via dynamic feedback linearization: design, implementation, and experimental validation, IEEE Trans. Control Syst. Technol. 10 (6) (2002) 835–852.
[30] R. Fierro, A.K. Das, V. Kumar, J.P. Ostrowski, Hybrid control of formation of robots, in: International Conference on Robotics and Automation, 2001, pp. 157–162.
[31] J. Cortes, S. Martinez, T. Karatas, F. Bullo, Coverage control for mobile sensing networks, IEEE Trans. Robot. Automat. 20 (2) (2004) 243–255.
[32] A. Perizzato, Navigazione e coordinamento di agenti mobili mediante tecniche di controllo predittivo distribuito (Master's thesis), Politecnico di Milano, 2012–2013, (in Italian).
[33] F. Tedesco, A. Casavola, E. Garone, Distributed command governor strategies for constrained coordination of multi-agent networked systems, in: Proceedings of the American Control Conference, ACC, 2012, 2012, pp. 6005–6010.
[34] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, A. Martinoli, The e-puck, a robot designed for education in engineering, in: Proceedings of the Conference on Autonomous Robot Systems and Competitions, vol. 1, No. 1, 2009, pp. 59–65.