Robotica (2018) volume 36, pp. 1943–1962. © Cambridge University Press 2018
doi:10.1017/S0263574718000814

Distributed cooperative deployment of heterogeneous autonomous agents: a Pareto suboptimal approach

Giovanni Franzini* and Mario Innocenti

Department of Information Engineering, University of Pisa, Largo Lucio Lazzarino 1, Pisa 56122, Italy. E-mail: mario.innocenti@unipi.it

(Accepted July 29, 2018. First published online: August 30, 2018)

SUMMARY

The paper presents a distributed cooperative control law for autonomous deployment of a team of heterogeneous agents. Deployment problems deal with the coordination of groups of agents in order to cover one or more assigned areas of the operational space. In particular, we consider a team composed by agents with different dynamics, sensing capabilities, and resources available for the deployment. Sensing heterogeneity is addressed by means of the descriptor function framework, an abstraction that provides a set of mathematical tools for describing both agent sensing capabilities and the desired deployment. A distributed cooperative control law is then formally derived finding a suboptimal solution of a cooperative differential game, where the agents are interested in achieving the requested deployment, while optimizing the resources usage according to their dynamics. The control law effectiveness is proven by theoretical arguments, and supported by numerical simulations.

KEYWORDS: Heterogeneous autonomous agents; Cooperative control; Distributed control; Deployment problems.

1. Introduction

Deployment problems deal with the distribution of a team of agents in an environment of interest so that a desired coverage of a prescribed area is achieved.[1] Examples are the *static coverage problem*,[2] which requires the distribution of agents according to a coverage function that denotes regions with higher and lower interest; the *uniform deployment problem*,[3] where the agents must spread in the operational area so that the resources they carry are uniformly distributed (a special case of the static coverage problem); and the *target assignment problem*[4] that assigns to the team a set of static targets to cover. These types of problems arise in a wide range of applications, including, but not limited to, environment monitoring,[5] precision agriculture,[6] surveillance[7] and crowds monitoring.[8] For these problems, the use of a team composed by autonomous agents with different capabilities and characteristics is particularly appealing. As a matter of fact, a heterogeneous team can respond to different mission scenarios, since the agents can be equipped with different types of sensors and payloads.

Agents heterogeneity arises at three levels:

- *Sensing*. The area covered by the agents may differ, due to the different types of sensors equipped, and their characteristics.
- *Dynamics*. The team can be composed by different vehicles that can even operate in distinct domains (e.g. it may include both aerial and ground vehicles).
- *Resources*. The agents may not have the same amount of resources (e.g. fuel or battery charge).

Coordination algorithms for team of heterogeneous agents must cope with these three aspects, in order to achieve the requested coverage according to the agents capabilities.

* Corresponding author. E-mail: franzigi@utrc.utc.com

Sensing heterogeneity has been extensively studied in literature. Algorithms based on Voronoi tessellation, such as *power diagrams*[9] and *weighted Voronoi diagrams*,[10] are usually adopted to address static coverage problems with isotropic mobile sensors having different sensing radius.[3,11–14] The case of anisotropic sensors is considered instead in refs. [15,16], where the authors proposed a new geometric approach that overcomes Voronoi limitations. Gradient-based methods are preferred when agents' sensors have a limited field of view.[17–21] Another technique adopted for solving static coverage tasks is the area partitioning, where planning algorithms are used to divide the environment in regions and design paths the agents must follow in order to keep the assigned region under control. Partitioning techniques are generally adopted for the coordination of *unmanned aerial vehicles* (UAVs). In ref. [22], the authors propose a graph-based partitioning that determines the number of UAVs to deploy given the area to cover, the available flight time, and the time needed to prepare a vehicle. Distributed techniques for area partitioning are proposed in refs. [23–25], where a team of aerial robots with isotropic sensing with different radius and limited speed is considered. A common trend observable in the literature, related to deployment problems, is to consider agents with single-integrator kinematics. Coverage control laws for unicycles are proposed in refs. [26–28]. Agents with double-integrator kinematics are considered in refs. [29,30]. Generally, in the literature, the control law design is based on kinematic models. In ref. [31], the authors extends coverage control based on Voronoi diagrams to non-holomonic agents using dynamic models.

Resources optimization seems to be a less studied problem in the area of deployment problems. In ref. [32], the authors present a Voronoi partitioning technique for a team of single-integrator agents, that optimizes control usage. A similar problem is studied in ref. [33], where the authors considered the deployment in a constant flow environment (a river). Other modifications of the classical Voronoi diagrams were proposed in refs. [34–36] for the optimal deployment of a sensor network, in order to maximize the area covered while minimize control usage. The references considered agents with single-integrator kinematics and isotropic sensors with different radius. An extension of the previous works to linear time-invariant systems is proposed in ref. [37].

To the best of the authors' knowledge, what seems to be missing is a coordination strategy that takes in account all the three aspects of agents heterogeneity at the same time. Moreover, most of the solutions proposed in literature address only specific deployment problems. For instance, the majority of the previously cited papers considers the static coverage problem. Application of these solutions to different deployment problems, e.g. to a target assignment task, is not straightforward and may require significant modifications.

The aim of the paper is to propose a general methodology for the solution of deployment problems, which combines a well established formalism for modeling and coordinating teams of heterogeneous agents, with a cooperative control law based on game theoretic arguments.

In particular, we deal with sensing heterogeneity by means of a modeling abstraction: the *descriptor function framework*, originally proposed in ref. [38]. The framework's key idea is the use of a common formalism, the *descriptor functions*, to describe both the agent capabilities (e.g. the area covered by the agent sensors) and the desired deployment (e.g. the area to cover). In this way, it is possible to quantify the mismatch between the desired and the current deployment, and use this error measurement to coordinate the agents motion. The error can then be used for the design of the agent control laws. This feature allows the accomplishment of different tasks without modifying the underlying control architecture, as opposed to the majority of the solutions proposed in literature. The descriptor function framework capabilities of coordinating team of heterogeneous agents have been extensively proven in several papers, see e.g. [39] and [40]. Furthermore, the framework is capable of considering the human presence, and the descriptor function formalism can be extended in order to achieve human–machine cooperation.[41] The interested reader could refer to ref. [42] for further reference.

Agents cooperation is formally introduced into the descriptor function framework framing the deployment problem as a *cooperative differential game*. The agents composing the team are considered as players that aim to cooperatively reaching the desired deployment, while minimizing their consumption according to their dynamics. To achieve the objective, while coping with their limitations, agents must cooperate and coordinate their actions to avoid penalizing each other. Speaking in terms of game solutions, this can be seen as finding a *Pareto efficient solution* of the game.[43] Pareto strategies are such that any deviation does not result in lower costs for all the players. Hence, the players are interested in cooperating in order to minimize the incurred costs while avoiding to penalize each other, as opposed to Nash strategies where the players selfishly try to minimize their own costs, knowing

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

that the others do the same. Game theory formalism has already been used to address the control of multi-agent systems.[44] In particular, *potential games* were used for exploration tasks[45] and sensor coverage problems.[46,47] However, potential games, and in particular, Nash equilibria considered in the previously cited papers, do not guarantee that one agent does not penalize the others as opposed to Pareto efficient strategies.

The cooperative game, once formulated, is transformed into a classical optimal control problem with non-linear terminal cost. To provide an approximate solution of the problem, and, thus, of the cooperative differential game, we solve the associated *generalized Hamilton–Jacobi–Bellman* (GHJB) equation. The GHJB is a relaxed version of the *Hamilton–Jabobi–Bellman* (HJB) equation for solving the optimal control problem that provides a suboptimal solution, as well as the associated cost. In particular, the GHJB equation can be used to iteratively improve the performance of the control law, starting from an initial suboptimal solution.[48] In the present paper, however, we will only solve once the GHJB obtaining a suboptimal solution of the problem. The iterative method for performance improvement proposed in ref. [48] is not considered, and its application to the deployment problem is left as topic for future research. The solution of the cooperative game is a centralized strategy for the whole team. Hence, a decentralization policy is discussed in order to obtain a distributed control law that each agent can implement and compute autonomously.

The paper is structured as follows. In Section 2, we discuss the modeling of the team of autonomous agents and we introduce the main concepts at the basis of the descriptor function framework. In Section 3, the cooperative deployment game is stated and solved. A decentralization of the solution is proposed and the nature of the possible equilibria reached by the team is discussed. Section 4 presents simulations results. In particular, the proposed cooperative control law is tested for three different types of deployment: a target assignment task, a uniform deployment problem, and a static coverage task. Conclusions are drawn in Section 5.

Notation.

The field of reals is denoted with $\mathbb{R}$, and the set of the strictly positive reals is denoted with $\mathbb{R}^+ = \{x \in \mathbb{R} : x > 0\}$. The set of positive reals, including zero, is instead denoted with $\mathbb{R}_0^+ = \mathbb{R}^+ \cup \{0\}$. The agent configuration space on a topological space X is denoted with $\mathcal{C}(X)$. Vectors and matrices are denoted using the bold font, e.g. $\boldsymbol{v}$, whereas scalar values with the normal font, e.g. α. The identity matrix is denoted with $\boldsymbol{I}$, and the zero matrix with $\boldsymbol{0}$. Dimensions of these matrices are omitted when they can be easily inferred from the context, or otherwise indicated as subscripts. Positive definite and semi-definite matrices are denoted using the notation $\boldsymbol{A} > 0$ and $\boldsymbol{A} \geq 0$, respectively. The operator $\|\cdot\|_{\boldsymbol{W}}$, with $\boldsymbol{W} \geq 0$, denotes the weighted Euclidean norm of a vector, i.e. $\|\boldsymbol{v}\|_{\boldsymbol{W}} = \sqrt{\boldsymbol{v}^T \boldsymbol{W} \boldsymbol{v}}$. The operators diag $\{\cdot\}$ and ver $\{\cdot\}$ represent the block diagonal and the vertical concatenation, respectively. The operator $\otimes$ denotes the Kronecker product.

2. Team Modeling

2.1. Agents dynamics

We consider a team $\mathcal{T}$ composed by N agents. The team *operational space* is denoted with $Q \subset \mathbb{R}^n$, and may be two-dimensional or three-dimensional, i.e. $n = \{2, 3\}$. In either case, Q is assumed to be a closed and bounded set. The agent *pose*, i.e. its position and orientation in the operational space, is denoted with $\boldsymbol{p}_i \in \mathcal{C}(\mathbb{R}^n)$.

Agents are modeled by driftless control affine dynamics,

$$\begin{cases} \dot{\boldsymbol{x}}_i(t) = \boldsymbol{g}_i(t, \boldsymbol{x}_i(t))\boldsymbol{u}_i(t) \\ \boldsymbol{p}_i(t) = \boldsymbol{h}_i(\boldsymbol{x}_i(t)) \end{cases} \tag{1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^{q_i}$ is the ith agent state vector, $\boldsymbol{u}_i \in \mathbb{R}^{m_i}$ is the control vector, $\boldsymbol{g}_i : [0, \infty) \times \mathbb{R}^{q_i} \to \mathbb{R}^{q_i \times m_i}$, and $\boldsymbol{h}_i : \mathbb{R}^{q_i} \to \mathcal{C}(\mathbb{R}^n)$ is a function that maps the agent state into the pose vector $\boldsymbol{p}_i$. We assume that both $\boldsymbol{g}_i(\cdot)$ and $\boldsymbol{h}_i(\cdot)$ are continuous and differentiable functions. In addition, we assume that the agent is controllable, that is the accessibility distribution associated with the vector fields $\boldsymbol{g}_i^1(\cdot), \ldots, \boldsymbol{g}_i^{m_i}(\cdot)$, where $\boldsymbol{g}_i^j(\cdot)$ denotes the jth column of $\boldsymbol{g}_i(\cdot)$, has dimension equal to q_i. We recall that the accessibility

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

distribution of the driftless control affine dynamics (1) is given by the involutive closure of $\Delta = \text{span}\{\boldsymbol{g}_i^1(\cdot), \ldots, \boldsymbol{g}_i^{m_i}(\cdot)\}$. Further details on non-linear controllability can be found in ref. [49].

Remark 1. Considering a two-dimensional operational space $Q \subset \mathbb{R}^2$, where the agent pose is given by $\boldsymbol{p}_i = [x_i, y_i, \theta_i]^T$, two common examples of vehicle kinematics that can be modeled as in (1) are:

- single integrators, with $\boldsymbol{x}_i = \boldsymbol{p}_i, \boldsymbol{u}_i \in \mathbb{R}^3$, and

$$\boldsymbol{g}_i = \boldsymbol{I}_{3\times 3}, \quad \boldsymbol{h}_i(\boldsymbol{x}_i) = \boldsymbol{x}_i \tag{2}$$

- unicycles, with $\boldsymbol{x}_i = \boldsymbol{p}_i, \boldsymbol{u}_i = [u_{i_s}, u_{i_\theta}]^T$, and

$$\boldsymbol{g}_i(\boldsymbol{x}_i) = \begin{bmatrix} \cos\theta_i & 0 \\ \sin\theta_i & 0 \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{h}_i(\boldsymbol{x}_i) = \boldsymbol{x}_i \tag{3}$$

The controllability of both vehicle kinematics (2) and (3) is straightforward to prove.

We define the team state, control, and pose vectors, respectively, as

$$\boldsymbol{x} = \text{ver}\,\{\boldsymbol{x}_i\}_{i\in\mathcal{T}} \in \mathbb{R}^q, \quad q = \sum_{i\in\mathcal{T}} q_i$$

$$\boldsymbol{u} = \text{ver}\,\{\boldsymbol{u}_i\}_{i\in\mathcal{T}} \in \mathbb{R}^m, \quad m = \sum_{i\in\mathcal{T}} m_i$$

$$\boldsymbol{p} = \text{ver}\,\{\boldsymbol{p}_i\}_{i\in\mathcal{T}} \in \mathcal{C}^N(\mathbb{R}^n)$$

The *team dynamics* are then defined as

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{g}(t, \boldsymbol{x}(t))\boldsymbol{u}(t) \\ \boldsymbol{p}(t) = \boldsymbol{h}(\boldsymbol{x}(t)) \end{cases} \tag{4}$$

where $\boldsymbol{g}\colon [0, \infty) \times \mathbb{R}^q \to \mathbb{R}^{q\times m}$ is given by

$$\boldsymbol{g}(t, \boldsymbol{x}) = \text{diag}\,\{\boldsymbol{g}_i(t, \boldsymbol{x}_i)\}_{i\in\mathcal{T}} \tag{5}$$

and the function $\boldsymbol{h}\colon \mathbb{R}^q \to \mathcal{C}^N(\mathbb{R}^n)$ is defined as

$$\boldsymbol{h}(\boldsymbol{x}) = \text{ver}\,\{\boldsymbol{h}_i(\boldsymbol{x}_i)\}_{i\in\mathcal{T}}$$

2.2. Agents capabilities: The descriptor function framework

Framework overview: The descriptor function framework is based on three main elements: the agents that form the team, the team mission, and the tasks. The team is composed by a set of heterogeneous *agents* that operate in a given environment in order to complete one or more objectives. The desired spatial distribution of the agents participating to the same objective is described by the *task*. Coordination in time and space of the various tasks is defined by the *mission*.

Agent capability of executing a task in a point of the operational space is quantified using the *agent descriptor function* (ADF), which is a function of the agent relative distance with respect to that point. In particular, the value of the ADF relative to a given task in a point of the space, describes the capacity of the agent to execute that task in that point. If an agent is not able at all of executing a particular task, the associated ADF will be equal to zero in all the operational space. Agents descriptor functions can then be viewed as the quantities of resources that the agents carry with them and deploy/use in the operational space.

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

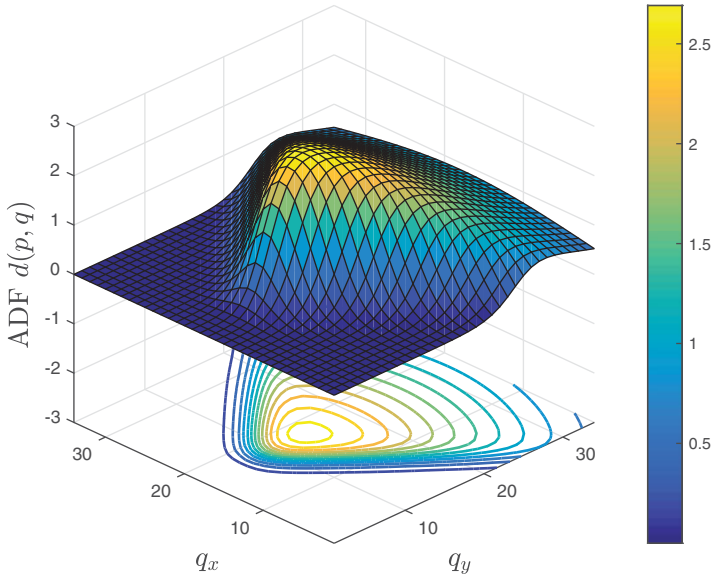

Fig. 1. Example of Gaussian ADF with limited field of view, as defined in (7), with $A = 3$, $\mathbf{\Sigma} = 3\boldsymbol{I}_{2\times 2}$, $k = 2$, and $\phi = 90^\circ$.

The desired distribution of resources in the environment is described by the *task descriptor function* (TDF). The TDF is an indication to the team on how to deploy in order to accomplish the task.

The sum of all agents' descriptor functions relative to a task represents the *current task descriptor function* (CTDF). This function describes the spatial distribution of all the resources pursuing the same objective.

The difference between the TDF and the CTDF is the *task error function* (TEF). The TEF describes the amount of resources needed, or in excess, in a given point of the operational space. Thus, it is a measure of the level of accomplishment of the task.

The TEF can be used to develop control laws capable of coordinating the agents of the team. By developing agents controllers based on the TEF, different tasks can be defined within the descriptor function framework, without the need for different control laws. One should simply define the TDFs according to the objectives of the mission the team has to accomplish. Therefore, the descriptor function framework is a potentially versatile and general tool for multi-agent system coordination and control.

In addition, the knowledge of the CTDF allows the self-organization and adaptation of the team. In particular, agents can use the CTDF to coordinate their actions in order to complete the task(s) assigned (*task level self-organization*) and to switch among the mission tasks (*mission-level self-organization*).

In the following, we will deal only with the team task level self-organization. The task considered will be the generic deployment problem, and all the descriptor functions definitions introduced in the next section will refer to this task class.

Framework elements mathematical definition: The ith agent capability with respect to the task is described by the ADF,

$$d_i(\boldsymbol{p}_i, \boldsymbol{q})\colon \mathcal{C}(\mathbb{R}^n) \times Q \to \mathbb{R}_0^+$$

The ADF indicates the quantity of resource that the agent can provide in a location $\boldsymbol{q}$ of the operative space, given its current pose. We assume that the ADF is a continuous and differentiable function over the operational space Q (see Fig. 1 as an example).

Remark 2. Consider a sensing application in $Q \subset \mathbb{R}^2$, with $\boldsymbol{q} = [q_x, q_y]^T$. A common choice for modeling the area covered by agents carrying isotropic sensors is the use of Gaussian functions

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

centered on the agent position, e.g.

$$d_{\mathrm{G}}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\mu}_i) = A_i \exp\left(-\frac{1}{2}\|\boldsymbol{R}(\theta_i)(\boldsymbol{q} - \boldsymbol{S}\boldsymbol{p}_i)\|^2_{\boldsymbol{\Sigma}_i^{-1}}\right) \tag{6}$$

with

$$\boldsymbol{R}(\theta_i) = \begin{bmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

where $\boldsymbol{\mu}_i = \{A_i, \boldsymbol{\Sigma}_i\}$ is the set of agent i ADF parameters with $\boldsymbol{\Sigma}_i \in \mathbb{R}^{2\times 2}$ denoting the symmetric positive definite covariance matrix, $\boldsymbol{R}(\cdot)$ rotates the ADF according to the current agent orientation, and $\boldsymbol{S}$ is a selection matrix that extracts the agent position components from the pose vector.

Sensors characterized by a limited field of view can be modeled using the following modified version of (6), as shown in ref. [39]:

$$d_{\mathrm{G,FOV}}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\mu}_i, \boldsymbol{\rho}_i) = d_G(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\mu}_i) f_{\mathrm{FOV}}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\rho}_i) \tag{7}$$

where the field-of-view function $f_{\mathrm{FOV}} : \mathcal{C}(\mathbb{R}^n) \times Q \to \mathbb{R}_0^+$ is defined as

$$f_{\mathrm{FOV}}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\rho}_i) = f_{\mathrm{FOV},r}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\rho}_i) f_{\mathrm{FOV},l}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\rho}_i)$$

$$f_{\mathrm{FOV},r}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\rho}_i) = \left(1 + \exp\left(-k_i\left(r_1 \cos\left(\frac{\phi_i}{2}\right) + r_2 \sin\left(\frac{\phi_i}{2}\right)\right)\right)\right)^{-1}$$

$$f_{\mathrm{FOV},l}(\boldsymbol{p}_i, \boldsymbol{q}; \boldsymbol{\rho}_i) = \left(1 + \exp\left(-k_i\left(-r_1 \cos\left(\frac{\phi_i}{2}\right) + r_2 \sin\left(\frac{\phi_i}{2}\right)\right)\right)\right)^{-1}$$

$$\begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = \boldsymbol{R}\left(\theta_i - \frac{\pi}{2}\right)(\boldsymbol{q} - \boldsymbol{S}\boldsymbol{p}_i)$$

with $\boldsymbol{\rho}_i = \{k_i, \phi_i\}$. The function $f_{\mathrm{FOV}}(\cdot)$ is equal to 1 inside the field of view of the agent, and smoothly decreases to 0 on the boundaries. The parameter $\phi \in [0, 2\pi]$ is the agent field of view, whereas k models the slope on the boundaries of the field of view, see Fig. 1.

The area covered by the agent is given by the ADF support set. In particular, we define the *support set* of a generic descriptor function as

$$\mathrm{supp}\{d_i(\boldsymbol{p}_i, \cdot)\} = \{\boldsymbol{q} \in Q : d_i(\boldsymbol{p}_i, \boldsymbol{q}) \geq 0\}$$

Remark 3. The support set of the Gaussian ADF (6) is theoretically equal to the whole operational space. However, the area effectively covered by the sensor is smaller. For this reason, for ADFs of this type, we can use the following modified definition of support set:

$$\mathrm{supp}_\epsilon\{d_i(\boldsymbol{p}_i, \cdot)\} = \{\boldsymbol{q} \in Q : d_i(\boldsymbol{p}_i, \boldsymbol{q}) \geq \epsilon\}, \quad \epsilon \in \mathbb{R}_0^+$$

where ϵ is a threshold on the sensor intensity that denotes values for which the sensor is actually covering.

The sum of all the agents ADFs is the CTDF,

$$d(\boldsymbol{p}, \boldsymbol{q}) = \sum_{i \in \mathcal{T}} d_i(\boldsymbol{p}_i, \boldsymbol{q})$$

and describes how the team resources are deployed in the operative space.

The task, i.e. the desired distribution of the team resources, is described by means of the TDF,

$$d_*(t, \boldsymbol{q}) : [0, \infty) \times Q \to \mathbb{R}_0^+$$

Note that in general this function may be time-varying.

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

Remark 4. Let us consider a sensing application. In this context, the ADF models the area covered by the agent sensors, the CTDF the current sensed area, and the TDF the area the team has to cover.

The difference between the TDF and the CTDF represents the TEF,

$$e(t, \boldsymbol{p}, \boldsymbol{q}) = d_*(t, \boldsymbol{q}) - d(\boldsymbol{p}, \boldsymbol{q}) \tag{8}$$

The TEF describes the amount of resources needed, or in excess, in a given point of the operational space. Therefore, it represents the mismatch between the desired and the current team deployment.

Given the TEF, the task state of execution is quantified by the *task error index* $\xi : [0, \infty) \times \mathcal{C}^N(\mathbb{R}^n) \to \mathbb{R}_0^+$, defined as follows:

$$\xi(t, \boldsymbol{p}) = \int_Q f(e(t, \boldsymbol{p}, \boldsymbol{q}))\sigma(\boldsymbol{q})\mathrm{d}\boldsymbol{q} \tag{9}$$

where $f : \mathbb{R} \to \mathbb{R}_0^+$ is a positive definite penalty function equal to zero only when the error is zero, and $\sigma : Q \to \mathbb{R}_0^+$ is a weight function used to increase or decrease the interest in particular areas of the operational space. A possible choice for $f(\cdot)$ is the family of penalty functions

$$f(e) = \max\{0, e\}^p, \quad p = 1, 2, \ldots \tag{10}$$

that are positive and convex.

Remark 5. The error index (9) along with the penalty function (10) is commonly used for static and effective coverage problems, see e.g. [50].

Remark 6. Note that $f(\cdot)$ is a continuous positive semi-definite function over $\mathbb{R}$, and it is strictly convex over $(0, +\infty)$, along with all its derivatives $\partial^n f / \partial e^n$ for $n < p$.

3. Deployment as a Cooperative Game

3.1. Formulation of the cooperative game

Given the dynamics of the single agents and a mathematical tool for modeling their capabilities, we now develop a cooperative control law that coordinates the team to achieve a desired deployment.

Using the elements of the descriptor function framework, the desired deployment is described by a time-invariant TDF, i.e. $d_*(\boldsymbol{q})$, and the agent capabilities using ADFs.

For each agent, we define the following cost function to be minimized:

$$\mathcal{J}_i(\boldsymbol{u}_i, \boldsymbol{u}_{i^-}) = \beta\xi(\boldsymbol{p}_i(t_f), \boldsymbol{p}_{i^-}(t_f)) + \gamma v_i(\boldsymbol{p}_i(t_f), \boldsymbol{p}_{i^-}(t_f)) + \int_{t_0}^{t_f} \|\boldsymbol{u}_i(t)\|^2_{\boldsymbol{R}_i} \mathrm{d}t \tag{11}$$

where t_f is the *fixed end time*, $\beta, \gamma \in \mathbb{R}^+$, and $\boldsymbol{R}_i > 0$. The agents are interested in reaching the optimal deployment that minimizes the task error index $\xi(\cdot)$, while minimizing the control energy, i.e. the resources usage.

In (11), we also introduced the *collision avoidance* function $v_i : \mathcal{C}^N(\mathbb{R}^n) \to \mathbb{R}_0^+$, defined in the following subsection. The introduction of $v_i(\cdot)$ in the terminal cost is sufficient to guarantee that the agent i does not collide with the others during the deployment, as it will be proven in Section 3.5.

With the notation i^-, we denote the set of all agents, excluded agent i, i.e. $i^- = \{j \in \mathcal{T}/\{i\}\}$. Accordingly, $\boldsymbol{p}_{i^-}$ and $\boldsymbol{u}_{i^-}$ denote the set of position and control vectors of all the agents, except for agent i, respectively. The aim is to highlight the fact that the cost incurred by agent i is not uniquely determined by its actions, but it is also influenced by other agents decisions. To truly achieve cooperation, the agents must pursue their objective without penalizing the others.

To this end, the team deployment can be framed as a cooperative game, where the agents are players that want to achieve the desired deployment, taking also into account their control energy. In doing this, each agent must not penalize the others with its actions. Hence, we are interested in finding a

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

cooperative strategy that enables the agents coordination in order to achieve the desired deployment according to the different resources available to the agents.

Formally, given the players/agents dynamics (1) and the cost functions (11), we want to find the *Pareto efficient* solution of the game, defined as follows.

Definition 1. A set of control actions $\{\boldsymbol{u}_i^*\}_{i\in\mathcal{T}}$, is said to be *Pareto efficient* (or *Pareto optimal*), if the set of inequalities

$$\mathcal{J}_i(\boldsymbol{u}_i, \boldsymbol{u}_{i^-}) \leq \mathcal{J}_i(\boldsymbol{u}_i^*, \boldsymbol{u}_{i^-}^*), \quad i \in \mathcal{T}$$

where at least one of the inequalities is strict, does not allow for any set of solutions.[43]

Pareto optimality can be interpreted as a solution in which any change made by a single agent does not help decreasing other agents incurred cost function. Therefore, agents must cooperate in order to minimize the cost incurred without penalizing other agents. Pareto optimal strategies are thus different from a set of strategies that constitutes a Nash equilibrium. As a matter of fact, the latter represents a situation where the players act selfishly since they are interested in minimizing their own cost, knowing that the other players will do the same.

We now formulate the deployment problem as a cooperative differential game.

Problem 1. Consider the cooperative differential game played by the team of agents $\mathcal{T}$, with dynamics (1), interested in minimizing the cost functions (11). Find a set of control laws $\{\boldsymbol{u}_i^*(t)\}_{i\in\mathcal{T}}$ for $t \in [t_0, t_f]$, with t_f fixed, that constitutes a Pareto efficient solution of the game.

3.2. Collision avoidance

For each agent i, we define the following collision avoidance function:

$$v_i(\boldsymbol{p}) = \sum_{j\in i^-} l\left(\left\|\boldsymbol{S}(\boldsymbol{p}_i - \boldsymbol{p}_j)\right\|\right)$$

where $\boldsymbol{S} \in \mathbb{R}^{n\times n}$ is a selection matrix that extracts the agent position from its pose vector. The function $l\colon \mathbb{R}_0^+ \to \mathbb{R}_0^+$ is adapted from ref. [51], and it is defined as follows:

$$l(x) = \left(\min\left\{0, \frac{x^2 - R^2}{x^2 - r^2}\right\}\right)^2, \quad R > r > 0$$

The value R denotes the radius of the area where the agents can detect the presence of other agents, whereas r is the *safe distance* the agents must maintain. It is worth of note that

$$\lim_{x\to r^+} l(x) \to +\infty$$

As a consequence of this property, if $v_i(\cdot) \to +\infty$, then there exists a pair of agents i and j such that $\|\boldsymbol{S}(\boldsymbol{p}_i - \boldsymbol{p}_j)\| \to r^+$, i.e. they are getting too close. Therefore, if $v_i(\cdot)$ attains finite values during the deployment, then collisions between the agent i and the other agents do not occur.

3.3. A Pareto suboptimal strategy for static deployment

To solve Problem 1, we use the following result:

Lemma 1. *Given a set of N weighting coefficient* $\alpha_i \in (0, 1)$ *such that* $\sum_{i=1}^N \alpha_i = 1$, *if the set of control action* $\boldsymbol{u}^* = \{\boldsymbol{u}_i^*\}_{i\in\mathcal{T}}$ *is such that*

$$\boldsymbol{u}^* \in \operatorname{argmin}_{\{\boldsymbol{u}_i\}_{i\in\mathcal{T}}} \left\{\sum_{i\in\mathcal{T}} \alpha_i \mathcal{J}_i(\boldsymbol{u}_i, \boldsymbol{u}_{i^-})\right\}$$

then it is Pareto efficient.

Proof. See ref. [43], Lemma 6.1. □

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

Note that the lemma does not require the convexity of the agent cost functions $\mathcal{J}_i(\cdot)$. Furthermore, we obtain different sets of Pareto optimal strategies changing the weights α_i, i.e. the relevance of agent i.

By means of Lemma 1, we formulate a parameterized optimal control problem, equivalent to Problem 1.

Problem 2. Given the dynamics (4), find the optimal control law $\boldsymbol{u}^*(t) \in \mathbb{R}^m$ for $t \in [t_0, t_f]$ that minimizes the cost function

$$\mathcal{J}(\boldsymbol{u}) = \beta\xi(\boldsymbol{p}(t_f)) + \gamma v(\boldsymbol{p}(t_f)) + \int_{t_0}^{t_f} \|\boldsymbol{u}(t)\|_{\boldsymbol{R}}^2 \, \mathrm{d}t \tag{12}$$

with $\boldsymbol{p}(t_f) = \boldsymbol{h}(\boldsymbol{x}(t_f))$, where t_f is the fixed final time, and $\boldsymbol{x}(t_f)$ is the free endpoint.

Note that Problem 2 is a fixed-time free-endpoint optimal control problem, with non-linear terminal cost.

The following Lemma proves the equivalence of the two problems.

Lemma 2. *Problem 1 is equivalent to Problem 2, with*

$$\boldsymbol{R} = \mathrm{diag}\,\{\alpha_i \boldsymbol{R}_i\}_{i\in\mathcal{T}} \tag{13}$$

$$v(\boldsymbol{p}) = \sum_{i\in\mathcal{T}} \alpha_i v_i(\boldsymbol{p}) \tag{14}$$

where $\alpha_i \in (0, 1)$ *are N weighting coefficients such that* $\sum_{i\in\mathcal{T}} \alpha_i = 1$.

Proof. The proof is a consequence of Lemma 1, noting that the cost function (12) is equal to

$$\mathcal{J}(\boldsymbol{u}) = \sum_{i\in\mathcal{T}} \alpha_i \mathcal{J}_i(\boldsymbol{u}_i, \boldsymbol{u}_{i^-})$$

with the introduction of (13) and (14). □

We now propose a set of control laws for the agent and prove, by means of formal arguments, that it is a suboptimal solution of Problem 2, hence a set of Pareto suboptimal strategies for Problem 1.

More specifically, we will find a solution of the GHJB equation associated to the optimal control problem stated in Problem 2. The GHJB is a relaxation of the HJB equation that solves the optimal control problem, and provides a suboptimal control along with its cost. The equation can then be used to improve controller performance by means of iterative techniques.[48] Although the GHJB equation is linear and easier to solve than the HJB equation, no general solution exists, and several numerical methods were proposed over the years.[52–55] The following Lemma provides a solution for the GHJB associated to Problem 2.

Theorem 1. *The set of control laws*

$$\boldsymbol{u}_i^*(t, \boldsymbol{x}) = -\frac{1}{\alpha_i}\boldsymbol{R}_i^{-1}\boldsymbol{g}_i(t, \boldsymbol{x}_i)^T \frac{\partial \boldsymbol{h}_i(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i}^T \left[\beta \frac{\partial \xi(\boldsymbol{p})}{\partial \boldsymbol{p}_i} + \gamma \frac{\partial v(\boldsymbol{p})}{\partial \boldsymbol{p}_i}\right]_{\boldsymbol{p}=\boldsymbol{h}(\boldsymbol{x})}^T \tag{15}$$

for $i \in \mathcal{T}$ *is a set of Pareto suboptimal strategies for the cooperative differential game stated in Problem 1.*

Proof. The HJB equation associated to the fixed-time free-endpoint optimal control problem stated in Problem 2 is

$$\frac{\partial \mathcal{V}(t, \boldsymbol{x})}{\partial t} + \min_{\boldsymbol{u}\in\mathbb{R}^m} \mathcal{H}\left(t, \boldsymbol{x}, \frac{\partial \mathcal{V}(t, \boldsymbol{x})}{\partial \boldsymbol{x}}, \boldsymbol{u}\right) = 0$$

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

with boundary condition

$$\mathcal{V}(t_f, \boldsymbol{x}(t_f)) = \beta\xi\left(\boldsymbol{h}(\boldsymbol{x}(t_f))\right) + \gamma v(\boldsymbol{h}(\boldsymbol{x}(t_f))) \tag{16}$$

where $\mathcal{V}\colon [t_0, t_f] \times \mathbb{R}^q \to \mathbb{R}$ is the *value function*, and

$$\mathcal{H}\left(t, \boldsymbol{x}, \frac{\partial \mathcal{V}}{\partial \boldsymbol{x}}, \boldsymbol{u}\right) = \|\boldsymbol{u}\|_{\boldsymbol{R}}^2 + \frac{\partial \mathcal{V}(t, \boldsymbol{x})}{\partial \boldsymbol{x}} \boldsymbol{g}(t, \boldsymbol{x})\boldsymbol{u}$$

is the *Hamiltonian* function associated to the problem, see e.g. ref. [56]. The relative GHJB equation is defined as follows:

$$\frac{\partial \mathcal{V}(t, \boldsymbol{x})}{\partial t} + \mathcal{H}\left(t, \boldsymbol{x}, \frac{\partial \mathcal{V}(t, \boldsymbol{x})}{\partial \boldsymbol{x}}, \boldsymbol{u}\right) = 0 \tag{17}$$

with boundary condition (16), see ref. [48]. A solution of (17) is

$$\boldsymbol{u}^*(t, \boldsymbol{x}) = -\boldsymbol{R}^{-1} \boldsymbol{g}(t, \boldsymbol{x})^T \frac{\partial \mathcal{V}(t, \boldsymbol{x})}{\partial \boldsymbol{x}}^T \tag{18}$$

with value function

$$\mathcal{V}(t, \boldsymbol{x}) = \beta\xi(\boldsymbol{h}(\boldsymbol{x})) + \gamma v(\boldsymbol{h}(\boldsymbol{x})) \tag{19}$$

that satisfies the boundary condition (16).

The control law (18) is a suboptimal solution of Problem 2, and the value function (19) evaluated at $(t_0, \boldsymbol{x}(t_0))$ is the associated cost. According to Lemma 2, (18) also represents a Pareto suboptimal solution for Problem 1. In particular, noting that the Jacobian of $\boldsymbol{h}(\cdot)$ is

$$\frac{\partial \boldsymbol{h}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \operatorname{diag}\left\{\frac{\partial \boldsymbol{h}_i(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i}\right\}_{i \in \mathcal{T}}$$

and considering the definitions of $\boldsymbol{R}$, and $\boldsymbol{g}(\cdot)$, see (5) and (13), the Pareto suboptimal solution for the ith agent (15) is obtained substituting the gradient of the suboptimal value function (19) in (18), and considering the component of (18) relative to agent i. □

Remark 7. The value function (19) does not depend on time explicitly, as one may expect from having fixed the time horizon. However, the reader should bear in mind that (19) is the solution of the GHJB that represents a relaxation of the optimal control problem stated in Problem 2.

Remark 8. The control law (15) is de facto a gradient-based law. Compared to the gradient-based methodologies presented in refs. [17–21], the solution proposed in this paper capable of coping with rather general sensing capabilities (not limited to sensors with limited field of view, considered in the references) and with different types of kinematics. In addition, control usage is taken into account, a feature generally missing in gradient-based laws for coverage control.

3.4. Control law decentralization

The Pareto efficient control law (15) presents two different contributions: the task error index gradient $\partial\xi/\partial\boldsymbol{p}_i$ that steers the agent in order to minimize $\xi(\cdot)$, and the collision avoidance term based on the gradient $\partial v/\partial \boldsymbol{p}$. These two contributions can be computed, in general, using only local information.

Computation of $\partial\xi(\boldsymbol{p})/\partial\boldsymbol{p}_i$*:* Considering (8), the gradient of the task error index is equal to

$$\frac{\partial \xi(\boldsymbol{p})}{\partial \boldsymbol{p}_i} = -\int_Q \frac{\partial f(e(\boldsymbol{p}, \boldsymbol{q}))}{\partial e(\boldsymbol{p}, \boldsymbol{q})} \frac{\partial d_i(\boldsymbol{p}_i, \boldsymbol{q})}{\partial \boldsymbol{p}_i} \sigma(\boldsymbol{q}) \mathrm{d}\boldsymbol{q} \tag{20}$$

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

The argument of the integral (20) is different from zero on the set

$$Q_i(\boldsymbol{p}_i) = \left\{ \boldsymbol{q} \in Q \colon \frac{\partial d_i(\boldsymbol{p}_i, \boldsymbol{q})}{\partial \boldsymbol{p}_i} \neq \boldsymbol{0} \right\}$$

Note that, according to the assumptions made on the ADF in Section 2.2, $Q_i(\boldsymbol{p}_i)$ is a closed and bounded set, and $Q_i(\boldsymbol{p}_i) \subseteq \text{supp}\,\{d_i(\boldsymbol{p}_i, \boldsymbol{q})\}$.

Thus, (20) is equal to

$$\frac{\partial \xi(\boldsymbol{p})}{\partial \boldsymbol{p}_i} = -\int_{Q_i(\boldsymbol{p}_i)} \frac{\partial f(e(\boldsymbol{p}, \boldsymbol{q}))}{\partial e(\boldsymbol{p}, \boldsymbol{q})} \frac{\partial d_i(\boldsymbol{p}_i, \boldsymbol{q})}{\partial \boldsymbol{p}_i} \sigma(\boldsymbol{q}) \mathrm{d}\boldsymbol{q} \tag{21}$$

Computation of (21) requires the local knowledge of $e(\cdot)$, the TEF, on $Q_i(\boldsymbol{p}_i)$. Assuming that the agent i knows the TDF $d_*(\cdot)$ on $Q_i(\boldsymbol{p}_i)$, and the ADF parameters of the other agents, a possible algorithm for the decentralized computation of (21) is the following:

1. Identify the set of neighboring agents whose ADFs have a non-empty intersection with $Q_i(\boldsymbol{p}_i)$, i.e.

$$\mathcal{T}_i^{\cap}(\boldsymbol{p}_i) = \left\{ j \in \mathcal{T} \colon \text{supp}\left\{ d_j(\boldsymbol{p}_j, \boldsymbol{q}) \right\} \cap Q_i(\boldsymbol{p}_i) \neq \emptyset \right\}$$

2. Compute the TEF on $Q_i(\boldsymbol{p}_i)$ as follows:

$$e(\boldsymbol{p}, \boldsymbol{q}) = d_*(\boldsymbol{q}) - d_i(\boldsymbol{p}_i, \boldsymbol{q}) - \sum_{j \in \mathcal{T}_i^{\cap}} d_j(\boldsymbol{p}_j, \boldsymbol{q}) \tag{22}$$

3. Use (22) to compute (21).

It is reasonable to assume that the TDF is provided to the agents by an external operator or supervisor that monitors and assigns the tasks to the team. In addition, considering a team composed by agents with different characteristics, it is also acceptable that the agents have knowledge about the other agents ADFs, so that they can efficiently coordinate their actions according to their capabilities.

An alternative decentralized computation of (21) is based on the estimation of the TEF on $Q(\boldsymbol{p}_i)$. In particular, in ref. [42, Chap. 5], a TEF estimation algorithm based on dynamic consensus techniques is proposed, that does not require the knowledge of agent ADFs.

Computation of $\partial v(\boldsymbol{p})/\partial \boldsymbol{p}_i$*:* The gradient of the collision avoidance function is given by

$$\frac{\partial v(\boldsymbol{p})}{\partial \boldsymbol{p}_i} = \sum_{j \in i^-} (\alpha_i + \alpha_j) \frac{\partial l(x_{ij})}{\partial x_{ij}} \frac{\partial x_{ij}}{\partial \boldsymbol{p}_i}, \quad x_{ij} = \|\boldsymbol{S}(\boldsymbol{p}_i - \boldsymbol{p}_j)\| \tag{23}$$

where

$$\frac{\partial l(x_{ij})}{\partial x_{ij}} = \begin{cases} 0, & x_{ij} > R \mid x_{ij} < r \\ 4\frac{(R^2 - r^2)\left(x_{ij}^2 - R^2\right)}{\left(x_{ij}^2 - r^2\right)^3} x_{ij}, & R \geq x_{ij} > r \\ \text{undefined}, & x_{ij} = r \end{cases}$$

and

$$\frac{\partial x_{ij}}{\partial \boldsymbol{p}_i} = \frac{\left(\boldsymbol{p}_i - \boldsymbol{p}_j\right)^T \boldsymbol{S}^T \boldsymbol{S}}{\|\boldsymbol{S}(\boldsymbol{p}_i - \boldsymbol{p}_j)\|}$$

Therefore, the jth term of (23) is computed only when the agent j enters inside the detection radius of i, i.e. $\|\boldsymbol{S}(\boldsymbol{p}_i - \boldsymbol{p}_j)\| \leq R$.

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

3.5. Team equilibria

We now prove, by means of Lyapunov-like arguments, that under the Pareto suboptimal control law (15) the team reaches a final deployment that corresponds to a local minimum of the task error index. More specifically, we will define a positive definite function $V(\cdot)$ representative of the task execution state, which includes the error index $\xi(\cdot)$. We will prove that $V(\cdot)$ decreases monotonically with time ($\dot{V}(\cdot) \leq 0$) if the control law (15) is adopted by the agents, meaning that the coverage task is accomplished by the team. Hence, $V(\cdot)$ is a Lyapunov function, that can be used to prove also that the team converges to a deployment that represents a local minimum of $\xi(\cdot)$. Furthermore, if the agents initial configuration is *safe*, that is $\boldsymbol{p}(t_0) \in \mathcal{P}_{\text{safe}}$, where

$$\mathcal{P}_{\text{safe}} = \left\{ \boldsymbol{p} \in \mathcal{C}^N(\mathbb{R}^n) \colon \left\| \boldsymbol{S}(\boldsymbol{p}_i - \boldsymbol{p}_j) \right\| > r, \forall i, j \in \mathcal{T}, j \neq i \right\}$$

then collisions do not occur during the deployment.

Theorem 2. *If $\boldsymbol{p}(t_0) \in \mathcal{P}_{safe}$ and the following two relationships hold*

$$\ker \left\{ \boldsymbol{g}_i(t, \boldsymbol{x}_i)^T \frac{\partial \boldsymbol{h}_i(\boldsymbol{x}_i)}{\partial \boldsymbol{x}_i}^T \right\} = \{\boldsymbol{0}\}, \quad \forall i \in \mathcal{T} \tag{24}$$

$$\gamma = \omega\beta, \quad \omega \in \mathbb{R}_0^+ \tag{25}$$

where ω is a weighting parameter, then the team safely converges to a suboptimal deployment in the sense of the task error index (9)*, under the control law* (15)*.*

Proof. Let us introduce the positive definite function

$$V(\boldsymbol{x}) = \xi(\boldsymbol{h}(\boldsymbol{x})) + \omega v(\boldsymbol{h}(\boldsymbol{x})) \tag{26}$$

Note that $V(\cdot) \geq 0$, due to the positive semi-definiteness of $\xi(\cdot)$ and $v(\cdot)$. Applying the chain rule, the time derivative of (26) is given by

$$\dot{V} = \frac{\partial \xi}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} + \omega \frac{\partial v}{\partial \boldsymbol{x}}\dot{\boldsymbol{x}} = \sum_{i \in \mathcal{T}} \left(\frac{\partial \xi}{\partial \boldsymbol{p}_i} + \omega \frac{\partial v}{\partial \boldsymbol{p}_i} \right) \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{x}_i} \boldsymbol{g}_i \boldsymbol{u}_i \tag{27}$$

Introducing (15), we write (27) as follows:

$$\dot{V} = -\sum_{i \in \mathcal{T}} \left\| \left(\boldsymbol{I}_{2\times 2} \otimes \boldsymbol{g}_i^T \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{x}_i}^T \right) \begin{bmatrix} \partial \xi / \partial \boldsymbol{p}_i^T \\ \partial v / \partial \boldsymbol{p}_i^T \end{bmatrix} \right\|_{\boldsymbol{W}_i}^2 \tag{28}$$

where

$$\boldsymbol{W}_i = \boldsymbol{F} \otimes \frac{1}{\alpha_i} \boldsymbol{R}_i^{-1}, \quad \boldsymbol{F} = \begin{bmatrix} \beta & \frac{1}{2}(\omega\beta + \gamma) \\ \frac{1}{2}(\omega\beta + \gamma) & \omega\gamma \end{bmatrix}$$

Since $\boldsymbol{R}_i > 0$, in order to $\boldsymbol{W}_i$ being positive semi-definite, the eigenvalues of $\boldsymbol{F}$ must be greater or equal to zero. These are given by

$$\lambda_{1,2} = \frac{1}{2} \left(\beta + \omega\gamma \pm \sqrt{\left(\omega^2 + 1\right)\left(\beta^2 + \gamma^2\right)} \right)$$

and are greater or equal to zero provided that $\gamma = \omega\beta$.

If conditions (24) and (25) are verified, then $\dot{V}(\cdot) \leq 0$ and $V(\cdot)$ is a Lyapunov-type function that, under the control law (15), guarantees that the task error index monotonically decreases and the agents do not collide with each other. The latter claim is justified by the fact that a collision occurs only when $v(\cdot) \to +\infty$. But since the agents start from a safe condition and $V(\cdot)$ monotonically decreases, this cannot happen.

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

Under condition (24), the task error index decreases until the team reaches an equilibrium $\boldsymbol{x}_{\text{eq}}$, corresponding to a deployment $\boldsymbol{p}_{\text{eq}} = \boldsymbol{h}(\boldsymbol{x}_{\text{eq}})$ such that

$$\boldsymbol{p}_{\text{eq}} \in \left\{ \boldsymbol{p} \in \mathcal{C}^N(\mathbb{R}^n) \colon \frac{\partial \xi}{\partial \boldsymbol{p}_i} = \boldsymbol{0} \;\&\; \frac{\partial v}{\partial \boldsymbol{p}_i} = \boldsymbol{0}, \;\; \forall i \in \mathcal{T} \right\} \tag{29}$$

The equilibria (29) correspond to deployments where the agents are at safe distance, and a stationary point of the task error index has been reached, i.e. a local minima, maxima, or saddle point of $\xi(\cdot)$. We now study the stability of these points, under the assumption that a safe deployment has been reached (i.e. $\partial v / \partial \boldsymbol{p}_i = 0, \forall i \in \mathcal{T}$).

- *Local minima*. Denoting with $\underline{\boldsymbol{p}} = \boldsymbol{h}(\underline{\boldsymbol{x}})$ a local minima of $\xi(\cdot)$, there exists a a neighborhood of $\underline{\boldsymbol{x}}$ such that $V(\boldsymbol{x}) - V(\underline{\boldsymbol{x}}) > 0$. Since $\dot{V}(\cdot) < 0$ and its equal to zero in $\underline{\boldsymbol{x}}$, then $\underline{\boldsymbol{x}}$ is an asymptotically stable equilibrium.
- *Local maxima or saddle points*. If $\overline{\boldsymbol{x}}$ is a local maximum, then $V(\boldsymbol{x}) - V(\overline{\boldsymbol{x}}) < 0$ in a neighborhood of $\overline{\boldsymbol{x}}$, and since $\dot{V}(\cdot) < 0$, then equilibrium $\overline{\boldsymbol{x}}$ is unstable. The same applies to the saddle points of $\xi(\cdot)$.

Therefore, given the unstable nature of local maxima and saddle points, the team safely reaches a suboptimal deployment, under the Pareto suboptimal control law (15). □

Remark 9. Note that the condition (24) is satisfied by both the single integrator (2) and unicycle dynamics (3).

4. Deployment Examples

The Pareto suboptimal control law (15) is now tested on three different types of deployment: a target assignment task, a uniform deployment problem, and a static coverage task. The former is used to analyze the cooperative behavior of the control law, showing that changing the control weights the agents coordinate their motion in order to minimize their consumption and according to the different agents limitations. With the uniform deployment problem and the static coverage task, we show how the proposed cooperative control law is able to coordinate a large group of agents with different dynamics, ensuring that no collisions occur and the desired deployment is safely achieved.

4.1. Target assignment

The *target assignment problem* requires that, given K static targets and N agents, at least each target is covered by one agent if $K \leq N$, or, otherwise, N targets are covered.

In this simulation, we consider a team composed by two agents that are demanded to cover two targets. The agents have single-integrator like dynamics, and Gaussian ADFs with limited field of view defined as in (7). ADF parameters for both the agents were set to

$$A = 3, \quad \boldsymbol{\Sigma} = 3\boldsymbol{I}_{2\times 2}, \quad k = 2, \quad \phi = 90^\circ$$

Control law parameters were set to

$$\alpha_i = \frac{1}{N} = \frac{1}{2}, \quad \beta = 1, \quad \gamma = 0$$

The two targets are modeled using the Gaussian descriptor function (6), and the TDF is given by

$$d_*(\boldsymbol{q}) = d_{\text{G}}(\boldsymbol{p}_{\text{T1}}, \boldsymbol{q}; \boldsymbol{\mu}) + d_{\text{G}}(\boldsymbol{p}_{\text{T2}}, \boldsymbol{q}; \boldsymbol{\mu})$$

with $\boldsymbol{\mu} = \{A = 2, \boldsymbol{\Sigma} = 3\boldsymbol{I}_{2\times 2}\}$.

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

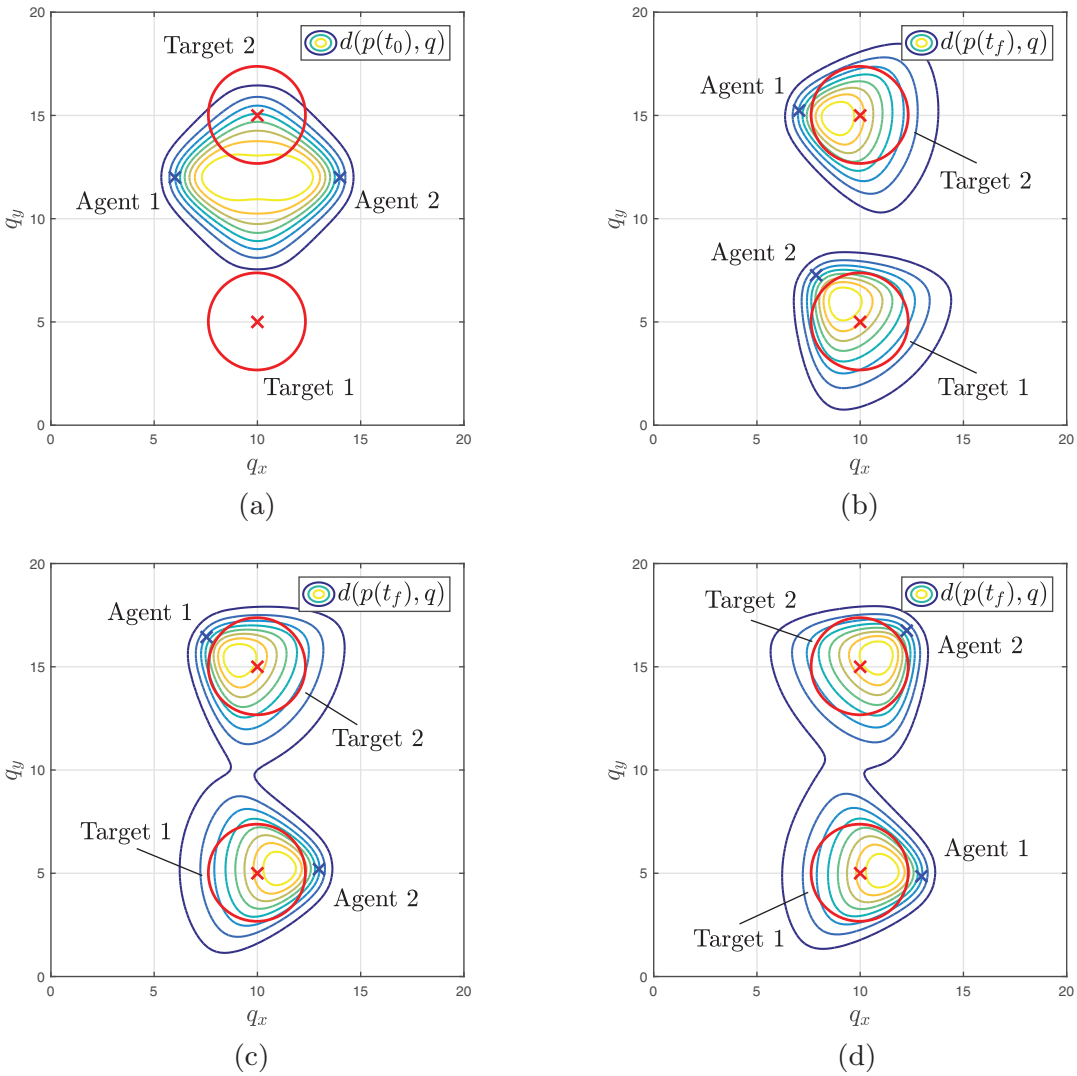

Fig. 2. Target assignment: initial and final deployments with different control weights. (a) Inital deployment. (b) $\boldsymbol{R}_1 = \boldsymbol{R}_2 = \boldsymbol{I}_{3\times 3}$. (c) $\boldsymbol{R}_1 = 10\boldsymbol{I}_{3\times 3}, \boldsymbol{R}_2 = \boldsymbol{I}_{3\times 3}$. (d) $\boldsymbol{R}_1 = \boldsymbol{I}_{3\times 3}, \boldsymbol{R}_2 = 10\boldsymbol{I}_{3\times 3}$.

Agents initial poses and targets descriptor functions position and orientation vectors are, respectively,

$$\boldsymbol{p}_1(t_0) = \begin{bmatrix} 6 \\ 12 \\ 3\pi/2 \end{bmatrix}, \quad \boldsymbol{p}_2(t_0) = \begin{bmatrix} 14 \\ 12 \\ \pi/2 \end{bmatrix}, \quad \boldsymbol{p}_{\mathrm{T1}} = \begin{bmatrix} 10 \\ 5 \\ 0 \end{bmatrix}, \quad \boldsymbol{p}_{\mathrm{T2}} = \begin{bmatrix} 10 \\ 15 \\ 0 \end{bmatrix}$$

The selected penalty function for the evaluation of the task error index (9) is given by

$$f(e) = \max\{0, e\}^2 \tag{30}$$

The aim of this simulation is to analyze the cooperative nature of the proposed agent control law. More specifically, we expect that, by increasing the control weight of one agent, this will move toward the nearest target, whereas the other agent, that has a lower penalization on control, will move to the farthest target. Note, in fact, that Target 2 is closer to the agents than Target 1. To gain better insights into the agent behavior, we turned off the collision avoidance capability.

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

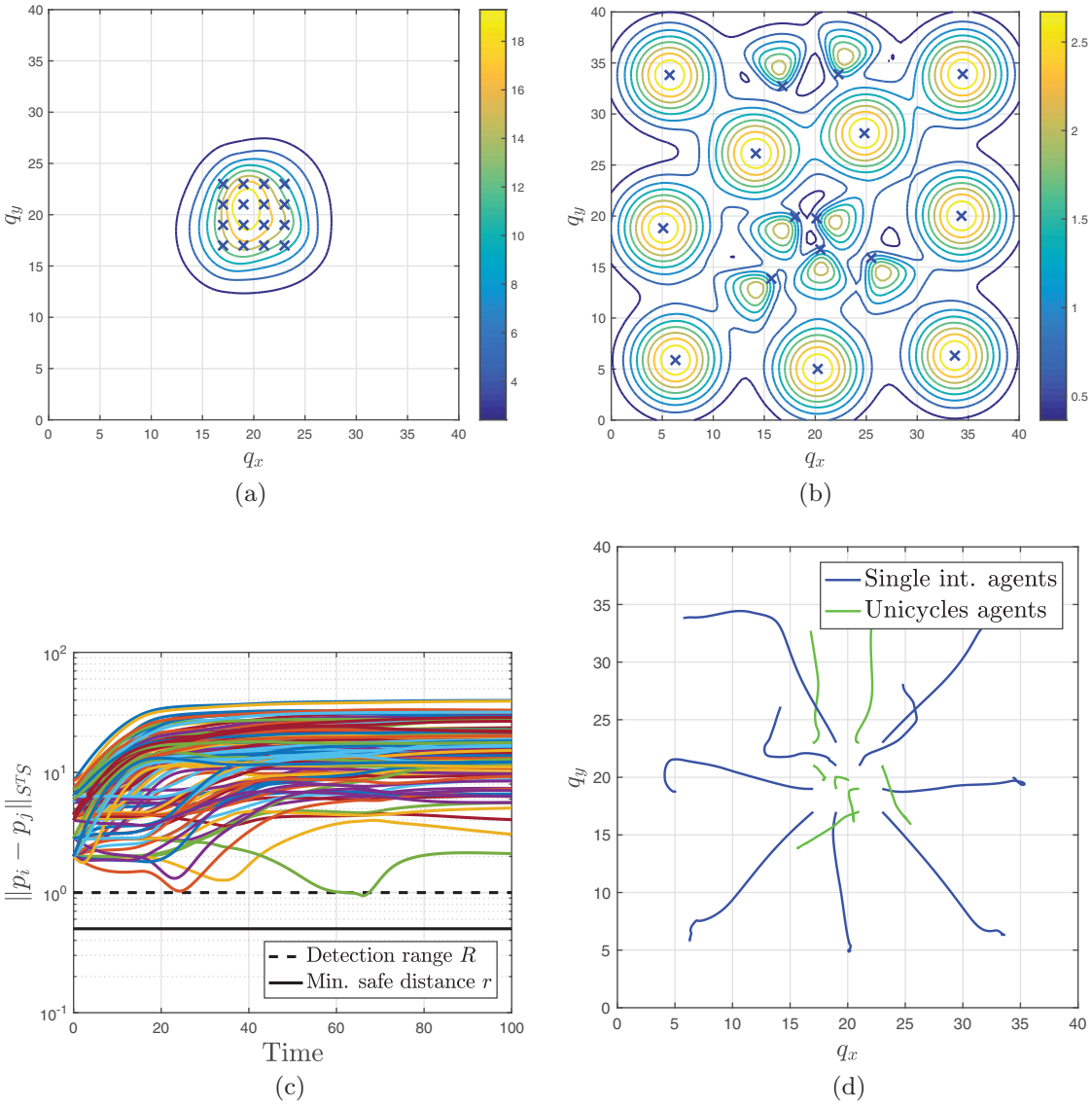

Fig. 3. Uniform deployment: results for $\boldsymbol{R}_i = \boldsymbol{I}_{m_i \times m_i}$, $\forall i \in \mathcal{T}$. (a) Initial deployment. (b) Final deployment. (c) Inter-agent distances evolution. (d) Agents trajectories.

Figure 2 shows the different cooperative behaviors that emerge by changing agents' control weights $\boldsymbol{R}_i$. The team initial deployment is shown in Fig. 2a. When the agents have the same control weights, they reach the final deployment illustrated in Fig. 2b. In Fig. 2c, we increased the control cost for Agent 1, and, as we can see, the agent moved to the closest target, whereas Agent 2, which has a lower penalization on control, moved to Target 2. Hence, agents cooperation allowed to perform the task while taking in account the different control capabilities of the agents. In Fig. 2c, we repeated the test, but this time penalizing agent 2. A perfectly symmetric behavior emerged.

Table I summarizes the tests results. The normalized index $\Delta u_i\%$ quantifies agents control usage, and is defined as

$$\Delta u_i\% = \frac{\Delta u_i}{\sum_{j\in\mathcal{T}} \Delta u_j}, \quad \Delta u_i = \int_{t_0}^{t_f} \|\boldsymbol{u}_i(t)\| \mathrm{d}t$$

The results confirm the above observations. It should be noted that the task error index final value $\xi(\boldsymbol{p}(t_f))$ is the same for all the tests, despite of the different choices of $\boldsymbol{R}_i$.

The reason of the behavior observed in these tests lies in the fact that the inverse of the control weight $\boldsymbol{R}_i$ acts as a gain for the control law (15). The agents with high values of $\boldsymbol{R}_i$, i.e. with high control

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

Table I. Task assignment, test results.

Control weights				
$\boldsymbol{R}_1$	$\boldsymbol{R}_2$	$\Delta u_1\%$	$\Delta u_2\%$	$\xi(\boldsymbol{p}(t_f))$
$\boldsymbol{I}_{2\times 2}$	$\boldsymbol{I}_{2\times 2}$	0.495	0.505	264
$10\boldsymbol{I}_{2\times 2}$	$\boldsymbol{I}_{2\times 2}$	0.054	0.956	263
$\boldsymbol{I}_{2\times 2}$	$10\boldsymbol{I}_{2\times 2}$	0.950	0.050	263

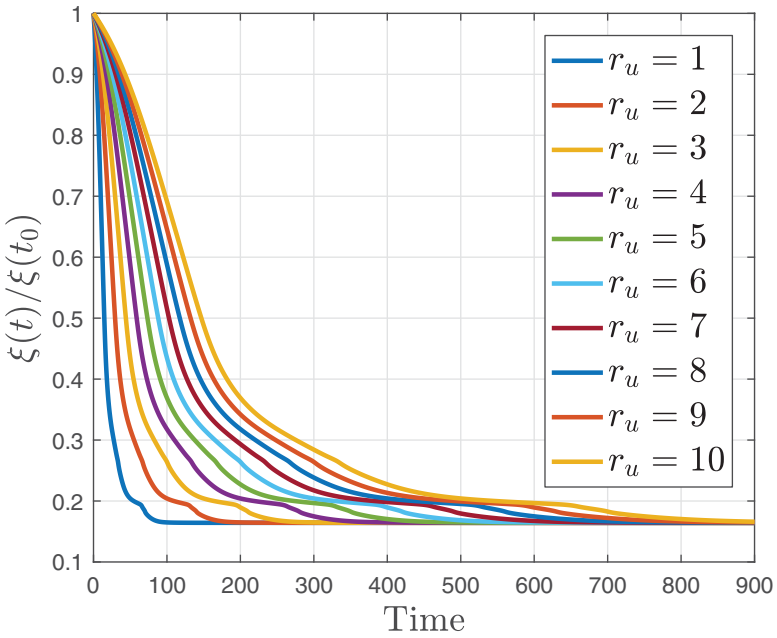

Fig. 4. Normalized task error index vs. control weight ($\boldsymbol{R}_i = r_u \boldsymbol{I}_{m_i \times m_i}, \forall i \in \mathcal{T}$).

penalization, sense only the closest targets. Hence, they move toward them. The agents with lower control penalization will be attracted by the farthest targets, since the agents with higher penalization already covered the closest targets.

4.2. Uniform deployment

The *uniform deployment problem* deals with the deployment of a team of agents so that a given area of interest is uniformly covered.

In this simulation, we consider a team of 16 agents, with different dynamics and sensing capabilities. In particular, the team is composed by nine single-integrator agents with Gaussian ADFs (6) with

$$A = 3, \quad \boldsymbol{\Sigma} = 3\boldsymbol{I}_{2\times 2}$$

and seven unicycles with dynamics (3) and Gaussian ADFs with field of view (7) with

$$A = 3, \quad \boldsymbol{\Sigma} = 3\boldsymbol{I}_{2\times 2}, \quad k = 2, \quad \phi = 90^\circ$$

Control law parameter were set as follows for all the agents:

$$\alpha_i = \frac{1}{N} = \frac{1}{16}, \quad \beta = 1, \quad \boldsymbol{R}_i = \boldsymbol{I}_{m_i \times m_i}$$

The collision avoidance parameters were set to

$$\gamma = 1, \quad r = 0.5, \quad R = 1$$

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

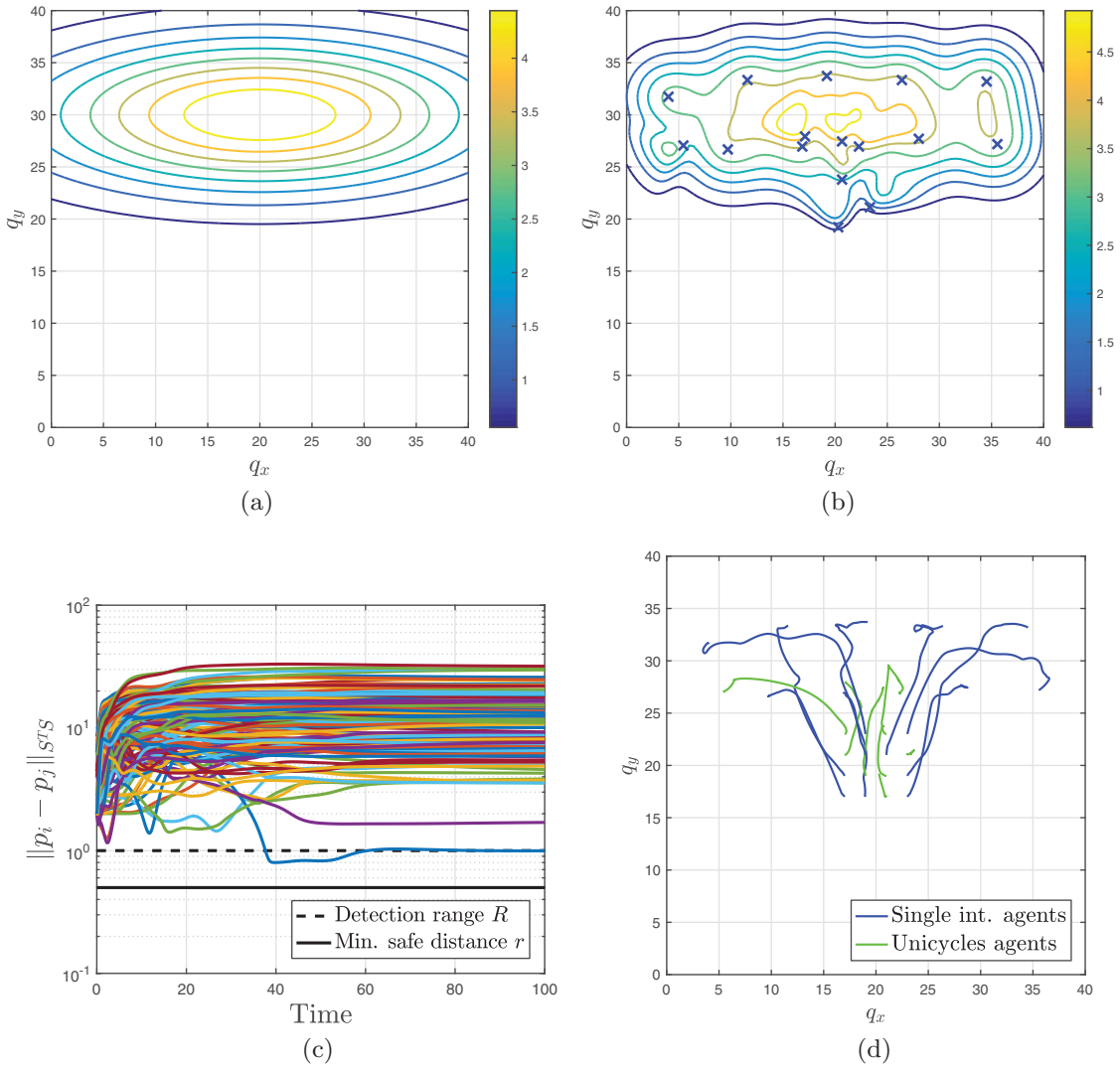

Fig. 5. Example of static coverage. (a) Task descriptor function. (b) Final deployment. (c) Inter-agent distances evolution. (d) Agents trajectories.

The desired TDF is

$$d_*(\boldsymbol{q}) = 1, \quad \forall \boldsymbol{q} \in Q$$

i.e. the team is demanded to uniformly cover the whole operational space, a 40×40 square. For this simulation, we chose the penalty function (30) as for the previous test.

The CTDFs relative to the team initial and final deployments are shown in Fig. 3a and 3b, respectively. As can be seen, the team successfully covered all the operational area. During the deployment, the agents maintained the minimum safe distance and no collisions occurred, see Fig. 3c and the agents trajectories shown in Fig. 3d.

In Fig. 4, we show how increasing the weights $\boldsymbol{R}_i$ to all the agents, influences the convergence rate of the team to the desired deployment. As the control weight increases, the convergence is slower, as one can expect looking at (28), and reaching the final deployment requires more time. Note that the task error index final value is always the same, as $\boldsymbol{R}_i$ only influences the convergence rate.

4.3. Static coverage

The *static coverage* problem is similar to the uniform deployment, except that there are areas of the operational space with higher and lower interest. A target coverage function is assigned to

https://doi.org/10.1017/S0263574718000814 Published online by Cambridge University Press

the agents, that are demanded to spread over the region of interest according to the coverage level requested.

We consider the same team of agents used for the uniform deployment test, with initial deployment as in Fig. 3a. Control law parameters are also the same.

The desired TDF is shown in Fig. 5a, and is defined as follows:

$$d_*(\boldsymbol{q}) = d_{\mathrm{G}}(\boldsymbol{p}_{\mathrm{SC}}, \boldsymbol{q}; \boldsymbol{\mu}), \quad \boldsymbol{p}_{\mathrm{SC}} = \begin{bmatrix} 20 \\ 30 \\ 0 \end{bmatrix}, \quad \boldsymbol{\mu} = \left\{ A = 5, \boldsymbol{\Sigma} = \begin{bmatrix} 15 & 0 \\ 0 & 5 \end{bmatrix} \right\}$$

The CTDF relative to the final team deployment is shown in Fig. 5b. As can be seen, the agents successfully accomplished the task and reached the desired deployment. Figure 5c shows the inter-agent distances during the deployment and proves that no collision occurred. The agents trajectories are shown in Fig. 5d.

5. Conclusions

In this paper, we address the problem of deploying a team of heterogeneous agents formulated as a cooperative differential game. In particular, we propose a Pareto suboptimal solution of the game, solving an equivalent fixed-time free-endpoint optimal control problem by means of the generalized HJB equation and the related theory. Agent heterogeneity was considered in terms of different sensing patterns, dynamics, and resources available for the deployment. The descriptor function framework was used in order to cope with the sensing heterogeneity of the agents, by means of simple yet effective modeling tools and formalism. Under this framework, we showed how three different types of deployment problems, viz. target assignment, uniform deployment, and static coverage, can be readily solved, and how the use of cooperative strategies can take in account the different agents resources available for the deployment. Lyapunov theory was used to formally prove the convergence of the team to the desired deployment under the control law proposed. The Pareto suboptimal strategies found can be implemented using only local data. More specifically, the control law can be distributed assuming that each agent knows the position and orientation of the agents within its sensing pattern, as well as their sensing capabilities.

References

1. F. Bullo, J. Cortés and S. Martínez, *Distributed Control of Robotic Networks* (Princeton University Press, Princeton, New Jersey, USA, 2009).
2. J. Cortés, S. Martínez, T. Karatas and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Trans. Robot. Autom.* **20**(2), 243–255 (2004). https://doi.org/10.1109/TRA.2004.824698.
3. X. Wang, S. Han, Y. Wu and X. Wang, "Coverage and energy consumption control in mobile heterogeneous wireless sensor networks," *IEEE Trans. Autom. Control* **58**(4), 975–988 (2013). https://doi.org/10.1109/TAC.2012.2225511.
4. S. Carpin, T. H. Chung and B. M. Sadler, "Theoretical Foundations of High-Speed Robot Team Deployment," *Proceedings of the IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany (2013) pp. 2033–2040. https://doi.org/10.1109/ICRA.2013.6630849.
5. M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robot. Autom. Mag.* **19**(1), 24–39 (2012). https://doi.org/10.1109/MRA.2011.2181683.
6. D. Albani, D. Nardi and V. Trianni, "Field Coverage and Weed Mapping by UAV Swarms," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems* (Sep. 2017) pp. 4319–4325. https://doi.org/10.1109/IROS.2017.8206296.
7. Z.-J. Wang and W. Li, "A solution to cooperative area coverage surveillance for a swarm of MAVs," *Int. J. Adv. Robot. Syst.* **10**(12) (2013) pp. 398(1)–398(8). https://doi.org/10.5772/56801.
8. A. Saeed, A. Abdelkader, M. Khan, A. Neishaboori, K. A. Harras and A. Mohamed, "Argus: Realistic Target Coverage by Drones," *Proceedings of the 16th ACM/IEEE International Conference on Information Processing in Sensor Networks*, Pittsburgh, PA, USA (2017) pp. 155–166. http://doi.acm.org/10.1145/3055031.3055078.
9. F. Aurenhammer, "Power diagrams: Properties, algorithms and applications," *SIAM J. Comput.* **16**(1), 78–96 (1987). https://doi.org/10.1137/0216006.
10. I. Z. Emiris and M. I. Karavelas, "The predicates of the apollonius diagram: Algorithmic analysis and implementation," *Comput. Geom.* **33**, 18–57 (2006). https://doi.org/10.1016/j.comgeo.2004.02.006.
11. L. C. A. Pimenta, V. Kumar, R. C. Mesquita and G. A. S. Pereira, "Sensing and Coverage for a Network of Heterogeneous Robots," *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico (Dec. 2008) pp. 3947–3952. https://doi.org/10.1109/CDC.2008.4739194.

12. N. Bartolini, T. Calamoneri, T. F. La Porta and S. Silvestri, "Autonomous deployment of heterogeneous mobile sensors," *IEEE Trans. Mobile Comput.* **10**(6), 753–766 (2011). https://doi.org/10.1109/TMC.2010.192.
13. M. Thanou, Y. Stergiopoulos and A. Tzes, "Distributed Coverage Mobile Heterogeneous Networks in Non-Convex Environments," *Proceedings of the 21st Mediterranean Conference on Control and Automation*, Platanias-Chania, Crete, Greece (Jun. 2013) pp. 956–962. https://doi.org/10.1109/MED.2013.6608837.
14. B. Boardman, T. Harden and S. Martínez, "Limited Range Spatial Load Balancing for Multiple Robots," *Proceedings of the American Control Conference*, Seattle, WA, USA (May 2017) pp. 2285–2290. https://doi.org/10.23919/ACC.2017.7963293.
15. Y. Stergiopoulos and A. Tzes, "Autonomous Deployment of Heterogeneous Mobile Agents with Arbitrarily Anisotropic Sensing Patterns," *Proceedings of the 20th Mediterranean Conference on Control and Automation*, Barcellona, Spain (Jul. 2012) pp. 1585–1590. https://doi.org/10.1109/MED.2012.6265865.
16. Y. Stergiopoulos and A. Tzes, "Cooperative Positioning-Orientation Control of Mobile Heterogeneous Anisotropic Sensor Networks for Area Coverage," *Proceedings of the IEEE International Conference on Robotics and Automation*, Hong Kong, China (May 2014) pp. 1106–1111. https://doi.org/10.1109/ICRA.2014.6906992.
17. A. Gusrialdi, T. Hatanaka and M. Fujita, "Coverage Control for Mobile Networks with Limited-Range Anisotropic Sensors," *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico (Dec. 2008) pp. 4263–4268. https://doi.org/10.1109/CDC.2008.4739007.
18. K. Laventall and J. Cortés, "Coverage Control by Robotic Networks with Limited-Range Anisotropic Sensory," *Proceedings of the American Control Conference*, Seattle, WA, USA (Jun. 2008) pp. 2666–2671. https://doi.org/10.1109/ACC.2008.4586895.
19. B. Hexsel, N. Chakraborty and K. Sycara, "Coverage Control for Mobile Anisotropic Sensor Networks," *Proceedings of the IEEE International Conference on Robotics and Automation*, Shanghai, China (May 2011) pp. 2878–2885. https://doi.org/10.1109/ICRA.2011.5980370.
20. B. Hexsel, N. Chakraborty and K. Sycara, "Distributed coverage control for mobile anisotropic sensor networks," Technical Report CMU-RI-TR-13-01, Robotics Institute, Pittsburgh, PA, USA (Jan. 2013).
21. Y. Kantaros, M. Thanou and A. Tzes, "Visibility-Oriented Coverage Control of Mobile Robotic Networks on Non-Convex Regions," *Proceedings of the IEEE International Conference on Robotics and Automation*, Hong Kong, China (Sep. 2014) pp. 1126–1131. https://doi.org/10.1109/ICRA.2014.6906995.
22. G. S. C. Avellar, G. A. S. Pereira, L. C. A. Pimenta and P. Iscold, "Multi-UAV routing for area coverage and remote sensing with minimum time," *Sensors* **15**(11), 27783–27803 (2015). http://dx.doi.org/10.3390/s151127783.
23. J. J. Acevedo, B. C. Arrue, I. Maza and A. Ollero, "Distributed approach for coverage and patrolling missions with a team of heterogeneous aerial robots under communication constraints," *Int. J. Adv. Robot. Syst.* **10**(1), (2013) pp. 28(1)–28(13). https://doi.org/10.5772/52765.
24. J. J. Acevedo, B. C. Arrue, I. Maza and A. Ollero, "A Decentralized Algorithm for Area Surveillance Missions Using a Team of Aerial Robots with Different Sensing Capabilities," *Proceedings of the IEEE International Conference on Robotics and Automation*, Hong Kong, China (May 2014) pp. 4735–4740. https://doi.org/10.1109/ICRA.2014.6907552.
25. J. J. Acevedo, B. C. Arrue, I. Maza and A. Ollero, "A distributed algorithm for area partitioning in grid-shape and vector-shape configurations with multiple aerial robots," *J. Intell. Robot. Syst.* **84**(1), 543–557 (2016). https://doi.org/10.1007/s10846-015-0272-5.
26. J. Enright, K. Savla and E. Frazzoli, "Coverage Control of Nonholomonic Agents," *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, Mexico (Dec. 2008) pp. 4250–4256. https://doi.org/10.1109/CDC.2008.4739379.
27. G. Mathew and A. Surana, "A Static Coverage Algorithm for Locational Optimization," *Proceedings of the 51st IEEE Conference on Decision and Control*, Maui, HI, USA (Dec. 2012) pp. 806–811. https://doi.org/10.1109/CDC.2012.6426561.
28. J. M. Luna, R. Fierro, C. T. Abdallah and J. Wood, "An adaptive coverage control for deployment of nonholonomic mobile sensor networks over time-varying sensory functions," *Asian J. Control* **15**(4), 988–1000 (2013). http://dx.doi.org/10.1002/asjc.636.
29. F. Sharifi, A. Chamseddine, H. Mahboubi, Y. Zhang and A. G. Aghdam, "A distributed deployment strategy for a network of cooperative autonomous vehicles," *IEEE Trans. Control Syst. Technol.* **23**(2), 737–745 (2015). https://doi.org/10.1109/TCST.2014.2341658.
30. F. Fabiani, D. Fenucci, T. Fabbri and A Caiti, "A distributed, passivity-based control of autonomous mobile sensors in an underwater acoustic network," *IFAC-PapersOnLine* **49**(23), 367–372 (2016). https://doi.org/10.1016/j.ifacol.2016.10.432.
31. R. A. Razak, S. Sukumar and H. Chung, "Decentralized adaptive coverage control of non-holonomic mobile robots," *IFAC-PapersOnLine* **49**(18), 410–415 (2016). https://doi.org/10.1016/j.ifacol.2016.10.200.
32. A. Kwok and S. Martínez, "Deployment algorithms for a power-constrained mobile sensor network," *Int. J. Robust Nonlinear Control* **20**(7), 745–763 (2010). http://dx.doi.org/10.1002/rnc.1464.
33. Y. Ru and S. Martínez, "Coverage control in constant flow environments based on a mixed energy-time metric," *Automatica* **49**(9), 2632–2640 (2013). https://doi.org/10.1016/j.automatica.2013.05.024.

34. Y. Song, B. Wang, Z. Shi, K. R. Pattipati and S. Gupta, "Distributed algorithms for energy-efficient even self-deployment in mobile sensor networks," *IEEE Trans. Mobile Comput.* **13**(5), 1035–1047 (2014). https://doi.org/10.1109/TMC.2013.46.
35. M. Moarref and L. Rodrigues, "An optimal control approach to decentralized energy-efficient coverage problems," *IFAC Proc. Vol.* **47**(3), 6038–6043 (2014). Proc. 19th IFAC World Congress, https://doi.org/10.3182/20140824-6-ZA-1003.01625.
36. M. T. Nguyen, L. Rodrigues, C. S. Maniu and S. Olaru, "Discretized Optimal Control Approach for Dynamic Multi-Agent Decentralized Coverage," *Proceedings of the IEEE International Symposium on Intelligent Control*, Buenos Aires, Argentina (Sep. 2016) pp. 335–340. https://doi.org/10.1109/ISIC.2016.7579984.
37. M. T. Nguyen, C. Stoica Maniu and S. Olaru, "Optimization-based control for multi-agent deployment via dynamic Voronoi partition," *IFAC-PapersOnLine* **50**(1), 1828–1833 (2017). https://doi.org/10.1016/j.ifacol.2017.08.185.
38. M. Niccolini, M. Innocenti and L. Pollini, "Near Optimal Swarm Deployment Using Descriptor Functions," *Proceedings of the IEEE International Conference on Robotics and Automation*, Anchorage, AK, USA (May 2010) pp. 4952–4957. https://doi.org/10.1109/ROBOT.2010.5509984.
39. A. Ferrari Braga, M. Innocenti and L. Pollini, "Multi-Agent Coordination with Arbitrarily Shaped Descriptor Function," *Proceedings of the AIAA Guidance, Navigation and Control Conference*, Boston, MA, USA (Aug. 2013). https://doi.org/10.2514/6.2013-4996.
40. M. Niccolini, L. Pollini and M. Innocenti, "Cooperative control for multiple autonomous vehicles using descriptor functions," *J. Sensor Actuator Netw.* **3**(1), 26–43 (2014). http://dx.doi.org/10.3390/jsan3010026.
41. G. Franzini, S. Aringhieri, T. Fabbri, M. Razzanelli, L. Pollini and M. Innocenti, "Human-Machine Interface for Multi-Agent Systems Management using the Descriptor Function Framework," **In**: *Modelling and Simulation for Autonomous Systems, 3rd International Workshop, MESAS 2016*, Rome, Italy, June 15–16. 2016, Revised Selected Papers (J. Hodicky, eds.) (Springer International Publishing, 2016) pp. 25–39. https://doi.org/10.1007/978-3-319-47605-6_3.
42. M. Niccolini, Swarm Abstractions for Distributed Estimation and Control *Ph.D. Thesis* (Pisa: University of of Pisa, Jul. 2011).
43. J. Engwerda, *LQ Dynamic Optimization and Differential Games* (John Wiley & Sons Ltd, Chichester, West Sussex, England, 2005).
44. Y. Shoam and K. Leyton-Brown, *Multiagent Systems – Algorithmic, Game-Theoretic, and Logical Foundations* (Cambridge University Press, New York, USA, 2009).
45. A. Caiti, T. Fabbri, D. Fenucci and A. Munafò, "Potential Games and AUVs Cooperation: First Results from the THESAURUS Project," *Proceedings of the MTS/IEEE OCEANS*, Bergen, Norway (2013). https://doi.org/10.1109/OCEANS-Bergen.2013.6608165.
46. J. R. Marden, G. Arslan and J. S. Shamma, "Cooperative control and potential games," *IEEE Trans. Syst., Man, Cybern. - Part B: Cybern.* **39**(6), 1393–1407 (2009). https://doi.org/10.1109/TSMCB.2009.2017273.
47. H.-B. Dürr, M. S. Stanković and K. H. Johansson, "Distributed Positioning of Autonomous Mobile Sensors with Application to Coverage Control," *Proceedings of the American Control Conference*, San Francisco, CA, USA (Jun. 2008) pp. 4822–4827. https://doi.org/10.1109/ACC.2011.5991324.
48. G. N. Saridis and C.-S. G. Lee, "An approximation theory of optimal control for trainable manipulators," *IEEE Trans. Syst. Man, Cybern.* **9**(3), 152–159 (1979). https://doi.org/10.1109/TSMC.1979.4310171.
49. A. Isidori, *Nonlinear Control Systems*, 3rd ed. (Springer-Verlag, London, 1995).
50. I. I. Hussein and D. M. Stipanović, "Effective coverage control for mobile sensor networks with guaranteed collision avoidance," *IEEE Trans. Control Syst. Technol.* **15**(4), 642–657 (2007). https://doi.org/10.1109/TCST.2007.899155.
51. D. M. Stipanović, P. F. Hokayem, M. W. Spong and D. D. Šiljak, "Cooperative avoidance control for multiagent systems," *J. Dyn. Syst. Meas. Control* **129**, 699–707 (2007). https://doi.org/10.1115/1.2764510.
52. R. W. Beard, G. N. Saridis and J. T. Wen, "Galerkin approximations of the Generalized Hamilton–Jacobi–Bellman equation," *Automatica* **33**(12), 2159–2177 (1997). https://doi.org/10.1016/S0005-1098(97)00128-3.
53. R. W. Beard, G. N. Saridis and J. T. Wen, "Approximate solutions to the time-invariant Hamilton–Jacobi–Bellman equation," *J. Optim. Theory Appl.* **96**(3), 589–626 (1998). https://doi.org/10.1023/A:1022664528457.
54. C. Park and P. Tsiotras, "Approximations to Optimal Feedback Control Using a Successive Wavelet Collocation Algorithm," *Proceedings of the American Control Conference*, Denver, CO, USA (Jun. 2003) pp. 1950–1955. https://doi.org/10.1109/ACC.2003.1243359.
55. Z. Chen and S. Jagannathan, "Generalized Hamilton–Jacobi–Bellman formulation-based neural network control of affine nonlinear discrete-time systems," *IEEE Trans. Neural Netw.* **19**(1), 90–106 (2008). https://doi.org/10.1109/TNN.2007.900227.
56. M. Athans and P. L. Falb, *Optimal Control: An Introduction to the Theory and Its Applications* (McGraw-Hill, New York, USA, 1966).