Accepted Manuscript

Distributed motion misbehavior detection in teams of heterogeneous aerial robots

Simone Martini, Davide Di Baccio, Francisco Alarcón Romero, Antidio Viguria Jiménez, Lucia Pallottino, Gianluca Dini, Aníbal Ollero

PII: S0921-8890(15)00136-0
DOI: http://dx.doi.org/10.1016/j.robot.2015.06.008
Reference: ROBOT 2494

To appear in: Robotics and Autonomous Systems

Received date: 20 January 2015
Revised date: 4 June 2015
Accepted date: 15 June 2015

Please cite this article as: S. Martini, D. Di Baccio, F. Alarcón Romero, A. Viguria Jiménez, L. Pallottino, G. Dini, A. Ollero, Distributed motion misbehavior detection in teams of heterogeneous aerial robots, Robotics and Autonomous Systems (2015), http://dx.doi.org/10.1016/j.robot.2015.06.008

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.
Distributed Motion Misbehavior Detection in Teams of Heterogeneous Aerial Robots

Simone Martini*, Davide Di Baccio*, Francisco Alarcón Romero†, Antidio Viguria Jiménez†, Lucia Pallottino‡ Gianluca Dini∗‡ and Aníbal Ollero§

*Research Center “E. Piaggio,” University of Pisa, Pisa, Italy
†Fundación Andaluza para el Desarrollo Aeroespacial (FADA-CATEC), Seville, Spain
‡Dept. of Ingegneria dell’Informazione, University of Pisa, Pisa, Italy
§GRVC, University of Seville, Seville, Spain

Abstract

This paper addresses the problem of detecting possible misbehavior in a group of autonomous mobile robots, which coexist in a shared environment and interact with each other and coordinate according to a set of common interaction rules. Such rules specify what actions each robot is allowed to perform in order to interact with the other members of the group. The rules are distributed, i.e., they can be evaluated only starting from the knowledge of the individual robot and the information the robot gathers from neighboring robots. We consider misbehaving those robots which, because of either spontaneous failures or malicious tampering, do not follow the rules and whose behavior thus deviates from the nominal assigned one. The main contribution of the paper is to provide a methodology to detect such misbehavior by observing the congruence of actual behavior with the assigned rules as applied to the actual state of the system. The presented methodology is based on a consensus protocol on the events observed by robots. The methodology is fully distributed in the sense that it can be performed by individual robots based only on the local available information, it has been theoretically proven and validated with experiments involving real aerial heterogeneous robots.

I. INTRODUCTION

The availability of distributed systems gave rise in the late 80s to a profound rethinking of many decision making problems and enabled solutions that were impossible before. A similar trend is now happening in control and will soon enable a formidable number of new robotic applications. Various distributed control policies have been proposed for formation control, flocking, sensor coverage, and intelligent transportation (see e.g. [1], [2], [3]). The adoption of similar notions of decentralization and heterogeneity in Robotics is advantageous in many tasks, where a cooperation among agents with analogous or complementary capabilities is necessary to achieve a shared goal. More specifically, we are interested in distributed multi-agent systems where each agent is assigned with a possibly different private goal, but needs to coordinate its actions with other neighboring agents.

The flexibility and robustness of such distributed systems, and indeed their ability to solve complex problems, have motivated many works that have been presented in literature (see e.g., [4], [5], [6], [7], [8], [9]). Although in most cases agents are modeled as identical copies of the same prototype, this assumption is often restrictive as the different agents that form a society may be implemented by different makers, and with different technologies etc. Heterogeneity in these artificial systems is advantageous when, for example, a problem requires interaction of agents with similar skills as well as agents with complementary capabilities. Most important, heterogeneity may be introduced to model the existence of malfunctioning agents, also called intruders [10], [11]. The complexity needed to represent such behaviors can be successfully captured by hybrid models, in which a continuous-time dynamics describes the physical motion of each agent, while an event-based one describes the sequence of interactions with its neighbors.

Gianluca Dini is the corresponding author (g.dini@iet.unipi.it).
This paper addresses the problem of detecting possible misbehavior in a group of autonomous robots, which coexist in a shared environment and interact with each other while coordinating according to a set of common interaction rules. The objective is to provide robots the capability of detecting agents whose behavior deviates from the assigned one, due to spontaneous failures or malicious tampering. The objective is ambitious and indeed very difficult to be achieved without a-priori knowledge of the interaction rules, but a viable solution can be found if the hybrid models describing the behavior is known in advance. The proposed methodology is fully distributed in the sense that it can be performed by individual robots based only on the local available information. It is based on a two-step process. First, agents combines the information gathered from on-board sensors and from neighbors by using communication and compute an a-priori prediction of the set of possible trajectories that the observed agent should execute based on the cooperative rules (prediction phase). Then, the predicted trajectories are compared against the one actually executed and measured by the observed robot itself and if none results close enough, the observed robot is selected as uncooperative (verification phase). The motion misbehavior detection ability of a single local monitor (used in the verification phase) is limited by its partial visibility. Robots need hence to combine the locally available information and reach an agreement on the reputation of the observed robot. To do this, we propose a Boolean consensus protocol that differs from those provided in [12], [13], [14]. Indeed, in this paper a consensus protocol on the events (more precisely on the encoder map defined in the following) observed by robots is proposed. In contrast to the other works, the consensus was on the reconstruction of the surrounding area of the observed robot [10]. In other words, we use the consensus to reconstruct the possible presence of a robot in an area that is not visible from all observing robots while in the other approaches a consensus protocol was used to reconstruct the robot position in the area. Hence, in our approach the computational cost is limited.

Although the proposed method is general and can be applied to a wide range of applications, it has been tested with experiments involving real aerial robots where the problem of detecting an intruder is fundamental for the safety of the system.

The paper is organized as follows. We start introducing in Section II a case study example to help the reader follow the notation introduced in Section III where the hybrid model of the proposed cooperation protocol is reported. The misbehavior detection problem is formally defined in Section IV. The Boolean consensus misbehavior detection strategy is described in Section V where the convergence in a finite number of steps is formally proved. Finally, Section VI discusses a case-study and the related experimental results.

II. A Case Study Example

In order to introduce the formal definitions and concepts of the paper we first start introducing a case study example that will be used to give an intuitive idea of the formalism introduced in next sections. The example is a simplified version of the collision avoidance strategy proposed in [15] and proved to be safe for two aircraft. The example has only illustrative purposes and by no means has to be intended as a description of a realistic UAV scenario.

Example 1: Consider two identical aircraft cruising at a given altitude with constant and equal linear velocity $v$. Aircraft can be represented by vector $(x, y, \theta) \in \mathbb{R}^2 \times S^1$. Referring to Fig. 1(a), each aircraft flies straight in Cruise mode until the other aircraft is detected at a distance closer than $d_1$. Whenever it occurs, it changes instantaneously its heading angle of amount $\Delta \theta$ and proceeds straight until a distance $L$ from nominal trajectory is reached. Then, it changes its heading of amount $-\Delta \theta$ and proceeds straight (Left mode). As soon as the other aircraft is at distance larger than $d_2$ (where $d_1 < d_2$) the aircraft changes instantaneously its heading angle of amount $-\Delta \theta$ and proceeds straight until nominal trajectory is reached. Then, it changes its heading of amount $\Delta \theta$ (Right mode) and then switches to the Cruise mode.

Let $D(t)$ be the distance between the two aircraft at time $t$, the behaviour of each aircraft is reported in Fig. 1(a) with a graphical representation of the associated hybrid model in terms of operating modes and switching conditions, see Fig. 1(b).
To describe the motion of aircraft based on those rules we may consider a configuration vector \((x, y) \in \mathbb{R}^2\) and the control input \(\theta \in \{0, \pm \Delta \theta\}\) with kinematics equations

\[
\begin{align*}
\dot{x} &= v \cos \theta \\
\dot{y} &= v \sin \theta
\end{align*}
\] (1)

To apply the described collision avoidance policy, each aircraft must be able to recognize the presence of another aircraft in a detection disc centered in its position and of radius \(d_2\). We then consider aircraft with limited sensing that are able to detect the presence of other aircraft in such a detection disc. The detection disc is then subdivided in eight sectors based on the orientation of the vehicle and the radii \(d_1\) and \(d_2\), see Fig. 1(c), i.e. the front left \((S^{FL}_1, S^{FL}_2)\), front right \((S^{FR}_1, S^{FR}_2)\), back left \((S^{BL}_1, S^{BL}_2)\) and back right \((S^{BR}_1, S^{BR}_2)\) sectors. In order to monitor the behaviour of target aircraft \(h\), we suppose that an observing aircraft \(i\), that lays in one of the sectors of \(h\), is able to detect the presence of a third aircraft \(k\) that is at distance less than \(d_2\) and that lays in one of the visible sectors. For example, if aircraft \(i\) lays in sector \(S^{BR}_1\) of aircraft \(h\), it can detect aircraft \(k\) only if it is at distance less than \(d_2\) and inside sectors \(S^{BL}_1, S^{BL}_2, S^{FR}_1, S^{FR}_2, S^{BR}_1\) and \(S^{BR}_2\) but not in \(S^{FL}_1\) or \(S^{FL}_2\). For example, referring to Fig. 1(c), an aircraft \(i\) in \(A\) can detect the aircraft \(k\) in \(B\) but not the one in \(C\).

In case of a larger number of aircraft there exist several collision avoidance policies that have been proved to guarantee safety of the system but are far too complex to be used as a simple illustrative example, see e.g. the round-about policies in [16], [17].

III. A MODEL OF COOPERATION PROTOCOLS FOR ROBOTICS AGENTS

Toward our goal of designing a distributed motion misbehavior detection system that applies to very general, heterogeneous robots, it is necessary to introduce a formalism that allows us to uniformly model a large variety of possible robots sharing sets of interaction rules.

Consider \(n\) robotic agents \(A_1, \ldots, A_n\), where \(A_i\) is described by a vector \(q_i\) in a continuous configuration space \(Q\). Such agents have their own dynamics, but need to collaborate with each other in order to accomplish a common task or to achieve possibly conflict goals. We consider systems where agents’ interaction can be described by rules that are decentralized and event-based, i.e. the cooperation actions that every agent can perform are specified according to a shared set \(R = \{\text{rule}_1, \ldots, \text{rule}_m\}\) of rules based only on locally measured events. To give an example, agents can be vehicles or robots moving in a shared environment and following common driving rules so as to avoid collisions [18], [19] as also described in the case study in the previous Section. Each vehicle determines its current maneuver based on the presence or absence of other neighboring vehicles and on its own destination. To model such cooperating
networked and distributed systems in the general case, we adopt a simplified version of the formalism introduced in [20], according to which an agent \( A_i \) is specified by:

- A configuration vector \( q_i \in Q \), where \( Q \) is a configuration space (\( q = (x, y) \in \mathbb{R}^2 \) in the case study Example 1).
- An input vector \( u_i \in \mathcal{U} \), where \( \mathcal{U} \) is a set of admissible input values; (\( \mathcal{U} = \{0, \pm \Delta \theta\} \) in the case study).
- A discrete state \( \sigma_i \in \Sigma \), where \( \Sigma \) is the set of operating modes; (\( \Sigma_i = \{\text{Cruise, Left, Right}\} \) in the case study).
- A dynamic map \( f_i \) describing how the agent’s configuration is updated:
  \[
  q_i(t) = f_i(q_i(t), u_i(t))
  \]  
  (for Example 1 the dynamic map is reported in (1)).
- A decoder map \( G_i \) describing which control values are applied in different operating modes \( \sigma_i \), i.e.
  \[
  u_i(t) = G_i(q_i(t), \sigma_i(t_k)), \quad \text{for } t \in [t_k,t_{k+1})
  \]
  In Example 1 we have \( w(t) = G_i(q_i(t), \text{Cruise}) = 0 \), while \( w(t) = G_i(q_i(t), \text{Left}) \) is a sequence of \( \Delta \theta, 0, -\Delta \theta \) and 0 again. Finally, \( w(t) = G_i(q_i(t), \text{Right}) \) is a sequence of \( -\Delta \theta, 0, \Delta \theta \) and 0 again.
- A set of topologies \( \eta_{k_1}(q), \ldots, \eta_{k_n}(q) \) on \( Q \), whose union defines the agent’s neighborhood in \( q \), i.e. \( N(q_i) = \bigcup_{j=1}^{n} \eta_{k_j}(q_i) \). The set of neighboring agents is hence \( N_i = \{A_k | q_k \in N(q_i)\} \), while the set of neighbors’ configurations is \( I_i = \{q_k \in Q | A_k \in N_i\} \), referred in the following as influence set of \( A_i \).

Referring to the case study described in Example 1, for aircraft \( i \) in \( q \) we have eight topologies corresponding to the eight sectors of the detection disc centered in \( q \). The neighboring agents are aircraft in the detection disc, i.e. aircraft that are closer to \( i \) more than \( d_2 \).
- An event vector \( s_i \in \mathbb{B}^{n_1} \) (whose components will be later referred to as sub-events) and a detection map \( S_i \) involving conditions over \( Q \) (as e.g. the presence of another agent in a specific region):
  \[
  s_{i,j}(t) = \sum_{q_k \in I_i} 1_{\eta_{k,j}(q_i)}(q_k)
  \]
  where \( \sum \) represents the logical sum (or), and \( 1_A(x) \) is the Indicator function of a set \( A \). Events for the case study are the presence of aircraft closer than \( d_1 \) and the absence of aircraft at distance less than \( d_2 \). In general events can be compositions of sub-events and different events may depend on the same sub-events. Hence, a vector of sub-events \( s_i \) is considered.
- A static decision map or encoder \( \varphi_i \) indicating the detector condition \( c_i \) based on events vector \( s_i \):
  \[
  c_i(t_k) = \varphi_i(s_{i,1}(t_k), \ldots, s_{i,n_1}(t_k))
  \]
  in other words, \( c_i(t_k) \) is a vector of logic operations of sub-events \( s_{i,j} \).
- An automaton \( \delta_i \) describing how the agent’s current discrete state (or mode of operation) \( \sigma_i \) is updated based on the detector condition \( c_i \):
  \[
  \sigma_i(t_{k+1}) = \delta_i(\sigma_i(t), c_i(t_k))
  \]  
  Those two last concepts applied to the case study are reported in Fig. 1(b).

It is clear that the set of rules describes the set of \( p \) operating modes, \( \Sigma = \{\text{mode}_1, \ldots, \text{mode}_p\} \), and the set of \( \nu \) logical propositions, or events, \( E = \{\text{event}_1, \ldots, \text{event}_\nu\} \). The occurrence of any of these events requires the current mode \( \sigma_i \) of the generic agent \( A_i \) to be changed. The generic event \( \text{event}_i \) measured from \( A_i \) can be assigned with a logical variable \( c_{i,l} \in \mathbb{B} \) taking the value true if \( \text{event}_i \) has been recognized by \( A_i \) and false otherwise. Although \( \text{event}_i \) depends on \( Q \), that continuously evolves with the time \( t \), it only switches from true to false or vice-versa at particular times \( t_k \), with \( k \in \mathbb{N} \), when the agents’ mode \( \sigma_i \) must be updated. Hence, the cooperation manager can be seen as a Discrete Event
Fig. 2. Representation of dynamics $\delta$ of the automaton of the generic cooperation manager.

System (DES) [21], and indeed an automaton (see Fig. 2), that receives $c_i$ as input and updates its state $\sigma_i$ according to rules of the forms

- $\text{rule}_0 \overset{\text{def}}{=} (\sigma(t_0) = \text{mode}_1); \leftarrow \text{start in mode}_1$;
- $\text{rule}_j \overset{\text{def}}{=} \text{if } \sigma_i(t_k) = \text{mode}_l \text{ and } c_m(t_k) = \text{true then } \sigma_i(t_{k+1}) = \text{mode}_p$.

Referring to the case study, mode$_1$ is Cruise and one of the rules is (if $\sigma_i(t_k) = \text{Cruise}$ and $c_l(t_k) =$ “aircraft detected at distance less than $d_1$” = true then $\sigma_i(t_{k+1}) = \text{Left}$).

The decoder connects the output of the the event-based system in (3) with the input of the time-driven system in (2). It translates or decodes the current maneuver $\sigma_i(t_k)$ into a control law, typically involving a feedback of the agent’s configuration of the form

$$ u_i(t) = G(q_i(t), \sigma_i(t_k)), $$

so that the autonomous controlled system

$$ \dot{q}_i(t) = f_i(q_i(t), G(q_i(t), \sigma_i(t_k))) = \tilde{f}_i(q_i(t), \sigma_i(t_k)),$$

correctly performs the control planned for the mode $\sigma_i(t_k)$. The decoder is an application $G : Q \times \Sigma \rightarrow U$ that returns the actuators’ input during the interval $[t_k, t_{k+1})$ (up to next event). In this perspective, $G$ acts has a converter from a discrete-valued event-driven signal to a continuous-valued time-driven one.

A second block, the encoder, realizes the reverse connection: it evaluates the logical variables $c_{i,l}$, for $l = 1, \ldots, \nu$, from current value of the system configuration $Q$. However, in decentralized scenarios, every agent $A_i$ must be able to plan its motion according to its own configuration $q_i$ and the configurations of the agents that lay in its vicinity. Hence, the encoder output will only depend on the influence set $I_i$ of the agent that instantaneously affects the behavior of $A_i$.

Due to limited visibility of its sensors, an agent $A_i$ is able to measure the configuration $q_j$ of another agent $A_j$ lying in its visible region $V_i$. This region changes with time depending on the configurations of $A_i$ and its neighbors, i.e. $V_i(t) = V(q_i(t), Q(t))$. The remaining part of the configuration space, namely $\bar{V}_i(t) = Q \setminus V_i(t)$, is the non-visible region and is composed of all those configurations that can not be “seen” from $A_i$. Note that our problem also requires the knowledge of the state $(\sigma_j)$ of an agent’s cooperation manager, that is unmeasurable and thus will be estimated. For simplicity, we assume that $I_i \subseteq V_i$, i.e. every agent is able to directly measure all information needed for planning its motion, as otherwise data received from possibly deceiving neighbors must be further validated. As a whole, the evolution of the continuous-valued time-driven dynamics of the physical system and that of the discrete-valued event-driven dynamics of the cooperation manager are entangled as it happens in a hybrid system...
\( \mathcal{H} = (q_i, \sigma_i, I_i) \). The behavior of \( \mathcal{A}_i \) can be written more compactly as

\[
\begin{align*}
(\dot{q}_i(t), \sigma_i(t_{k+1})) &= \mathcal{H}_i(q_i(t), \sigma_i(t_k), I_i(t)), \\
(\dot{q}_i(0), \sigma_i(0)) &= (q_{i0}^i, \sigma_{i0}^i),
\end{align*}
\]

(5)

where \( \mathcal{H}_i : Q \times \Sigma \times Q^{\alpha_i} \to T_Q \times \Sigma_i \) is the agent’s hybrid dynamic map \([22]\) and \( T_Q \) is the tangent space of \( Q \). We will denote with \( \phi_{\mathcal{H}_i}(q_i(t), \sigma_i(t), I_i(t)) \) the evolution of system (5).

IV. MISBEHAVIOR AND LOCAL DETECTION

Since our goal is to detect misbehaving agents, we first need to define how a misbehavior may manifest i.e. how a behavior may deviate from the nominal one. The first assumption is that an agent \( \mathcal{A}_h \) may execute trajectories \( q_h(t) \) that do not comply with the common interaction rules, but the information it exchanges with its neighbors is always correct. This can be guaranteed by the use of emerging trusted computing platforms (see e.g. \([23]\), \([24]\), \([25]\)). Secondly, we consider the fact that the cooperation manager of a robot is implemented as a control task that runs periodically and that is scheduled every \( T \) seconds. This means that a mode \( \sigma \) is started at the generic discrete time \( kT \) and is run up to \( (k+1)T \). Then, we assume that the local monitor of each robot and all the robots cooperation manager are synchronized, which can be obtained by means of the distributed solution proposed in \([26]\) for example. The section presents the architecture of a monitor that can detect such misbehavior, by using only information available to the agent. For the sake of clarity we refer to the robot \( \mathcal{A}_i \) with an on-board monitor as the observer robot and to \( \mathcal{A}_h \) as the target robot.

To begin with, consider that, if \( I_h \) is completely visible from \( \mathcal{A}_i \), it is sufficient to verify that the trajectory \( q_h(t) \) measured by the observer robot is close enough to the evolution of the cooperative model \( \mathcal{H} \), i.e.

\[
\|q_h(t) - \pi_Q(\phi_{\mathcal{H}_i}(q_i(t), \sigma_i(t_k), I_i(t)))\| \leq \epsilon, \quad \forall t,
\]

where \( \|\cdot\| \) is the Hausdorff distance, \( \pi_Q \) is the projector over the set \( Q \), and \( \epsilon \) is an accuracy based on the quality of available sensors.

Nonetheless, it typically holds that \( I_h \not\subseteq \mathcal{V}_i \), which makes it impossible to directly apply this simple solution. In other words, it may occur that a robot that influences the behaviour of \( \mathcal{A}_h \) is not visible by \( \mathcal{A}_i \). For example, referring to Fig. 1(c) \( B \) is visible by \( A \) while \( C \) is not. Hence, the influence region is partitioned as

\[
I_h = I_h \cap Q = I_h \cap (\mathcal{V}_i \cup \mathcal{V}_i) = (I_h \cap \mathcal{V}_i) \cup (I_h \cap \mathcal{V}_i) = I_h^{\text{obs}} \cup I_h^{\text{unobs}}.
\]

Reorder the model’s inputs as \( I_h = (I_h^{\text{obs}}, I_h^{\text{unobs}}) \), where \( I_h^{\text{obs}} = q_{i1}, \ldots, q_{i,\nu_i} \) is the list of configurations known to \( \mathcal{A}_i \), and \( I_h^{\text{unobs}} = q_{i,\nu_i+1}, \ldots, q_{i,n_h} \) is the list of remaining configurations that are unknown to \( i \).

Misbehavior of agent \( \mathcal{A}_h \) during the period \( [t_k, t_{k+1}] \) can be found by solving the following

Problem 1 (Decentralized Intrusion Detection): Given a trajectory \( q_h(t) \), a list \( I_h^{\text{unobs}}(t_k) \) of known configurations, a visibility region \( \mathcal{V}_i \), and a desired accuracy \( \epsilon \), determine if there exists an input \( I_h(t_k) = (I_h^{\text{obs}}(t_k), I_h^{\text{unobs}}(t_k)) \) s.t.

\[
\|q_h(t) - \pi_Q(\phi_{\mathcal{H}_i}(q_i(t), \sigma_i(t_k), I_i(t)))\| \leq \epsilon, \quad \forall t \in [t_k, t_{k+1}].
\]

where \( \hat{I}_h \) represents an “unobservable explanation” whose notion was introduced in DES \([27]\) and used in \([28]\), \([29]\). In other words, the problem is to determine if, given the available information, there exist unobserved conditions that influence the target robot and justify its motion based on the predefined rules.
A. Construction of the Local Monitor

The proposed approach is a two-step process: first, $A_i$ computes an a-priori prediction of the set of possible trajectories that $A_h$ can execute according to the cooperative model $\mathcal{H}_h$ and the partially known influence region (prediction phase); then, the predicted trajectories are compared to the one actually executed by $A_h$ and measured by $A_i$ and if none of them results close enough, $A_h$ is selected as uncooperative (verification phase).

The prediction phase involves constructing a predictor $\tilde{\mathcal{H}}_h$ that encodes all the observer’s uncertainty. The model is composed of a nondeterministic automaton whose state $\tilde{\sigma}_h \in \Sigma_h$ represents the set of operating modes that $A_h$ can perform based on local information, and whose transitions $\tilde{\delta}$ are the same as in $\delta$. The main challenge in the construction of the automaton is the estimation of an upper approximation $\tilde{c}_i$ of each detector condition $c_h$, that is achieved through the results that can be found in [30] and that are omitted for the sake of space. We hence suppose that each monitor is able to construct a predictor $\tilde{\mathcal{H}}_h$.

V. CONSENSUS FOR MISBEHAVIOR DETECTION

The motion misbehavior detection ability of a single local monitor is limited by its partial visibility. In this section we show how agents can combine the information and reach an agreement on the reputation of other agents through communication so as to cooperatively react against intruders.

We assume that agents can communicate via one-hop links in order to reduce their detection uncertainty and “converge” to a unique network decision. In this respect we need to introduce concepts involving procedures and algorithms aiming to reach an agreement in networks.

A. The Consensus Algorithms

Consider a network whose communication topology is represented by an undirected graph $G$ which is composed by a set of nodes $V = \{v_1, \ldots, v_n\}$ linked by edges s.t. an edge $e_{i,j}$ means that the node $v_i$ is able to communicate with node $v_j$. In our case for each agent $A_i \in N_j$ there exists an arc from node $i$ to node $j$ associated to robots $A_i$ and $A_j$ respectively.

Given a graph $G$, a consensus algorithm is an iterative interaction rule that specifies how each node $v_i$ updates its estimates of the generic information $s \in S$ shared among neighbors based on any received value $v_j$, i.e. it specifies the function $\xi : S \times S \rightarrow S$ which is used to compute

$$s_i^+ = \xi(s_i, s_j), \quad \text{for } i, j = 1, \ldots, n.$$ 

If the iteration of each node converges toward a common value, a consensus is reached. Typical consensus algorithms available from the literature assume that exchanged data are represented by real numbers [31], [32] and is typically combined according to a weighted average rule. More general cases may require even a nonlinear combination [33], that is still not applicable in our case in which we have $n$ uncertain measures.

In more general cases the quantities of interest could be possibly non-convex sets, intervals, or logical values. Motivated by this fact, we need to involve a more general class of consensus algorithms so as to permit agents sharing locally collected information and eventually “converge” to a unique network decision. Referring to our scenario, nodes are robots that are monitoring a common neighbor and that are supposed to communicate as in $G$ in order to reach an agreement on the reputation of the observed robot $A_h$. Consider the vector

$$\mathbf{R}_h(t_k) = \begin{bmatrix} r_h^{(1)}(t_k) \\ r_h^{(2)}(t_k) \\ \vdots \\ r_h^{(n)}(t_k) \end{bmatrix}$$

(6)
where $r_h^{(i)}(t_k)$ represents the reputation of agent $A_i$ about agent $A_h$ after $t_k$ steps of the consensus iterative procedure. Our aim is to design a distributed consensus algorithm allowing us to have $\lim_{k\to\infty} R_h(t_k) = 1 r_h^*$, where $r_h^*$ is the centralized reputation vector defined as the vector that would be constructed by a monitor collecting all initial measures and combining them according to $\xi$.

A possible solution allowing us to reach an agreement consists in letting agents to share the locally estimated encoder map $\varphi_h$ of target robot $A_h$. In other words, we propose a solution where agents share any information that is directly measured or reconstructed by inspecting its neighborhood through logical consensus [12]. After having established an agreement for the value of the encoder map for a generic agent, they will use the same decision rule and hence decide for the same classification vector. The proposed idea will be formalized and proved in following section. It is worth noting that, the proposed consensus on the encoder map $\varphi_h$ (or equivalently on the events vector $c_h$ associated to agent $A_h$) is the novel contribution of this paper with respect to related works [12], [13], [14].

B. Convergence of Consensus Algorithm

Given a target robot $A_h$ and $n$ observing agents, the goal of the proposed intrusion detection problem is hence to let the observing agents to exchange locally available information on the events $c_h$ that influence the behaviour of $A_h$. Such information is elaborated by each agent and flows among them through a communication network. Once an agreement on the events $c_h$ is reached, the evolution of $A_h$ is compared with the evolution determined by $c_h$ and the hybrid dynamic map in (5). If they are not sufficiently close, $A_h$ is classified as a misbehaving agent.

As mentioned, the information exchanged is the event vector $s_h$ estimated by each robot. More formally, we consider a state vector $x_i = (x_{i1}, \ldots, x_{in_h}) \in \mathbb{R}^{1 \times n_h}$, that is a string of bits representing the values that observing agent $A_i$ may assign to all sub-events that influence the evolution of agent $A_h$.

Let $X(t) = (x_1(t)^T, \ldots, x_n(t)^T)^T$ be the matrix in $\mathbb{R}^{n \times n_h}$ that represents the network state at the time $t$. We assume that each agent is a dynamic node that updates its local state $x_i$ through a distributed logical update function $F$ that depends on its state, on the state of its neighbors and on the observed inputs which is used to initialize the value of the state, i.e. $x_i(t+1) = F_i(X(t))$. Moreover we assume that every agent is able to produce the logical output vector $Y = (y_1(t)^T, \ldots, y_n(t)^T)^T \in \mathbb{R}^{n \times n_h}$ that corresponds to the detector condition (or events vector) $c_h$, estimated by the $n$ observing robots, by using an output function $D$ depending on the local state, i.e. $y_i = D_i(X_i) = c_h(t) = (c_{h1}, \ldots, c_{hn})$ is the event vector $c_h$ estimated by robot $A_i$.

In the most general case, the generic observing agent $i$ may or may not be able to measure the value of the $j$-th sub-event associated to $A_h$, i.e. $s_{hj}$. In this sense, we can conveniently introduce a visibility matrix $V \in \mathbb{R}^{n \times n_h}$ where $V_{ij} = 1$ if, and only if, agent $A_i$ is able to measure $s_{hj}$ and $V_{ij} = 0$ otherwise. Moreover, each agent is able to communicate only with a subset of other agents. Therefore, to effectively accomplish the given decision task, we need that such an information flows from one agent to another, consistently with available communication paths.

Hence the system can be described by the logical functions:

\[
\begin{align*}
X(t+1) &= F(X(t)) \\
\hat{X}_i(0) &= \tilde{U}(i) \\
Y(t) &= D(X(t))
\end{align*}
\]  

(7)

where $F : \mathbb{R}^{n \times n_h} \to \mathbb{R}^{n \times n_h}$, $D = \text{Diag}(D_1, \ldots, D_n)$ with $D_i : \mathbb{R}^{n_h} \to \mathbb{R}^{n_h}$, and $\tilde{U}(i)$ provides $\hat{s}_h^{(i)}$ that is an initial lower approximation of $s_h = (s_{h1}, \ldots, s_{hn})$ based only on observation of the neighborhood of $A_h$ operated by the $i$-th agent. In this paper, component-wise inequalities are considered, i.e. a lower approximation is a vector whose $j$-th component is less or equal to the $j$-th component of $s_h$.

We can now introduce the consensus algorithm, i.e. the logical function $F$, and state the following result
**Theorem 1:** Given a connected communication graph $G$, $n$ initial estimates $X(0) = (\tilde{s}_h^{(1)T}, \ldots, \tilde{s}_h^{(n)T})^T$, and a visibility matrix $V$ with non-null columns, the distributed logical consensus system

\[
\begin{align*}
x_{i,j}^+ &= V_{i,j} x_{i,j} - \sum_{k \in N_i} x_{k,j} , \\
x_{i,j}(0) &= \tilde{s}_h^{(i)} ,
\end{align*}
\]

with $i = 1, \ldots, n$ and $j = 1, \ldots, \kappa_h$ converges to the consensus state $X = 1_n s_h$ in a number of steps that is less than the graph diameter.

**Proof:** To prove the proposition, consider factorizing the update rule as follows. If $V_{i,j} = 0$, agent $i$ is unable to autonomously compute $s_{h,j}$. In this case equation (8) reduces to

\[x_{i,j}^+ = -V_{i,j} \sum_{k \in N_i} x_{k,j} = \sum_{k \in N_i} x_{k,j} .\]

Moreover, since each column of $V$ is non-null by hypothesis, there is at least one observing robot, say the $m$-th, with complete visibility on the $j$-th topology $\eta_{h,j}$, which implies $\overline{s}_{h,j}^{(m)} = s_{h,j}$. Note that we have $s_{h,j}^{(m)} \geq \overline{s}_{h,j}^{(i)}$ since $\overline{s}_{h,j}^{(i)}$ is a lower approximation of $s_{h,j}$. Since $G$ is connected, the real value $s_{h,j}$ is propagated from agent $m$ to the rest of the network, which implies that there exists a time $\bar{N} \leq \text{Diam}(G) < \infty$ after which

\[x_{i,j} = \sum_{k=1}^{n} x_{k,j} = s_{h,j} .\]

If instead $V_{i,j} = 1$, agent $i$ has complete knowledge of the $j$-th topology $\eta_{h,j}$ and its update rule (8) specializes to

\[x_{i,j}^+ = V_{i,j} x_{i,j} = x_{i,j} ,\]

and its initial estimate is $\overline{s}_{h,j}^{(i)} = s_{h,j}$. It trivially holds that $x_{i,j} = s_{h,j}$, which proves the theorem. □

It is worth noting that the hypothesis on the column of $V$ corresponds to the fact that each topology of $\mathcal{A}_h$ is visible by at least one of the observing agents.

Once a consensus has been reached on the event vector $s_h$ we still need to prove that the output of system (7) solves the intrusion detection problem and allows to identify if the target robot follows or not the predefined rules.

**Theorem 2:** Given a connected communication graph $G$, a visibility matrix $V$ with non-null columns, an output function $\Phi = (\varphi, \ldots, \varphi)$ s.t. $\varphi(s_h) = c_h$, the distributed logical consensus system (7), where $F$ is given by (8), solves Problem 1.

**Proof:** To prove the theorem we need to prove that the vector (6) is such that $r_h^{(i)}(\bar{N}) = r_h^*$ with $\bar{N} < \infty$ and $i = 1, \ldots, n$. With Theorem 1 we proved that $X(\bar{N}) = 1_n s_h$. This means that $\Phi_t(x_i) = \varphi(s_h) = c_h$ and the predictor $\tilde{\eta}_h^{(i)}(q_h, \overline{s}_h^{(i)}, \overline{I}_h^{(i)})$, $i = 1, \ldots, n$, (introduced in Section IV-A) is initialized with the value $\tilde{\sigma}_h^{(i)}(0)$ corresponding to the most conservative hypothesis on the activation of $c_h$ which is the same for all observing agents, i.e. $\tilde{\sigma}_h^{(i)}(0) = \tilde{\sigma}_h^*(0)$, $i = 1, \ldots, n$ where $\tilde{\sigma}_h^*(0) = \tilde{\sigma}_h(0)$ in the case the estimated event vector $\tilde{e}_h$ is equal to $c_h$. Thus, the estimated state $\tilde{\sigma}_h^{(i)}$, $i = 1, \ldots, n$, becomes

\[
\tilde{\sigma}_h^{(i)}(t_{k+1}) = \tilde{\delta}(\tilde{\sigma}_h^{(i)}(t_k), \tilde{e}_h(t_{k+1})) .
\]

According to this, we can compute $\alpha_r$ as

\[\|q_h(t) - \pi_{\mathcal{Q}}(\phi_{\mathcal{R}}(q_h(t), t^{(i)}_h(t_k)), r_h^{(i)}(t_k)) \| = \alpha_r , \quad \forall t \in [t_k, t_{k+1}], i = 1, \ldots, n .\]

If $\alpha_r > \varepsilon$ the trajectory of $\mathcal{A}_h$ is not compatible with the nominal one and hence it is considered as misbehaving, i.e. $r_h^{(i)} = \text{misbehaving}$, $i = 1, \ldots, n$, which proves the theorem. □
VI. DISCUSSION

A. On practical applicability of the proposed approach

The proposed misbehavior detection method is based on the concept of topologies \( \eta_{i,j} \) (see Section III) and events consist in the presence, or absence, of agents in such topologies. In real experiments it is hence sufficient to provide agents with on board sensors that are able to detect the presence of other agents in a region whose size will depend on the sensors range. In other approaches to detect misbehaving agents, the consensus is based on the value of the position of the target robot. As the validity of position values is strictly related to the quality of the on board sensors, the result of the consensus can be jeopardized in case of sensors performance decay. In contrast, in our case the result of the consensus protocol is an agreement among observing robots on the presence, or absence, of other robots in the target robot neighborhood. It follows that proposed approach is hence more robust with respect to sensor performance than position-based consensus.

Once the agreement is achieved, each observing robot has a complete knowledge of the events that influence the behaviour of the target robot. A comparison of the expected trajectory with the one really pursued by the target robot is finally computed. The discrepancy threshold \( \epsilon \) used to detect a misbehaving robot is a design parameter that depends on the sensor accuracy. For example, in the case study example reported in Section VI-B a misbehavior corresponds to an aircraft that goes straight on while it is supposed to turn right. Hence the value of \( \epsilon \) can be sufficiently large to allow small discrepancies due to unmodeled disturbances and noises such as the wind. For example, in the experiment described in Section VI-B2, the value of the threshold \( \epsilon \) is chosen as 50m.

For simplicity the workflow of the proposed method is reported in Fig. 3
speed (FAST); and ii) to change route to the right with a predefined angle (RIGHT) upon detecting a possible collision with another UAV (Fig. 4). The goal of the experiment is to validate the proposed approach by creating a situation where a UAV misbehaves and experimentally checking whether it is correctly detected. This scenario was integral part of the “Highly Automated Airfield” scenario of the EU Project PLANET (www.planet-ict.eu). The rest of the Section is organized as follows. Section VI-B1 describes the experimental setting whereas Section VI-B2 describes the experiment and the related results.

1) Experimental setting: We consider a system composed of four UAVs, two of which are *real* and the remaining two are *virtual*, i.e., they are UAVs whose behavior is simulated by a Simulink Model. We also assume that UAVs have the same model of dynamics and controllers and that they can communicate in one-hop.

The simulator of virtual UAVs actually consists in two Simulink models, the one for the simulation of the dynamics of the virtual UAVs and the other for the simulation of the monitors (that is used also for real UAVs). The dynamic model of the UAV is not reported for the sake of simplicity whereas a conceptual scheme of the monitoring simulation model is reported in Fig. 3b. Each monitor consists in two parts. The first combines the information gathered from both on-board sensors and neighboring UAVs (by using communication) and applies the proposed consensus protocol to detect the presence/absence of aircraft around the monitored one (“Consensus on sectors occupancy”). The second part is the rule-based agent model (Fig. 3a) that is used for the comparison of the real trajectory performed by the target and the expected one (“Misbehavior detection”).

The rule-based agent model in Fig. 3a is explicitly reported in Fig. 4 with the associated events. It has been implemented according to the model defined in Section III. In particular, the event-driven dynamics implements simple interaction rules to avoid collisions among aircraft while executing the assigned task. Indeed, each aircraft follows the assigned flight plan, and if a collision is detected it changes its route to right with a predefined angle following the rules reported in Fig. 4. It is worth noting that the minimum distance that triggers a collision alarm is chosen so that, if the behavior is correct, aircraft are allowed to avoid collisions in any conditions.

As to the real UAVs, the “Locomove” (Fig. 5(a)) has been designed, developed, and built in FADA-CATEC. It is sufficiently lightweight and has an adequate size so that it can be hand launched. The maximum take-off weight is $5.5$ kg. A minimum endurance of 40 min is achieved and an electrical motor powered with LiPo batteries is used as power plant. The payload is $500 - 600$ g which allows the user to implement different sensors for a wide range of applications. In full autonomous mode, the aircraft is able to land in flat grounds with wide clearance to obstacles. The UAV lands over its belly, which has reinforcement to avoid any damage in the fuselage. The aircraft uses a ground based realtime barometric pressure corrections server to precisely measure the altitude over the ground level at the landing area. For the touchdown phase, the aircraft also uses a sonar range finder.
The “Skywalker” (Fig. 5(b)) is instead a commercial product. It is lightweight and it has an adequate size such that it can be hand launched. The maximum take-off weight is $3 \text{ kg}$. It has a minimum endurance of 30 min since it uses an electrical motor powered with LiPo batteries. The payload is $250 - 300 \text{ g}$ which allows the user to put on-board sensors for different kinds of applications.

Both the Skywalker and the Locomove in fully autonomous mode fly under the control of an autopilot which has been developed by CATEC. CATEC’s autopilot has been designed to serve as a generic prototyping platform of GNC algorithms allowing the adoption of a model based design approach with rapid prototyping capability. It provides a wide enough diversity of interfaces to be able to connect different sensors and payloads and implements appropriate safety mechanisms in order to assure safe operations. This autopilot has been used in PLANET project to develop guidance, navigation and control algorithms and allows the integration of Simulink models, so developing and testing different algorithms is easy and fast.
2) Experimental Results: With reference to the system composed of the four autonomous UAVs presented above, we denote by $A_1$ (cyan) and $A_4$ (blue) the virtual UAVs, $A_2$ (red) the Locomove, and, finally, $A_3$ (green) the Skywalker (Figure 6). UAVs fly over way-points on the ground that are represented by black points.

The experiment consists in creating a misbehavior situation and checking if it is correctly detected by monitors. The situation consists in having UAV $A_2$, Locomove, that, at some point, violates the interaction rules specified in Figure 4. Therefore, in the experiment $A_2$ is the target and all other UAVs monitor it.

More precisely, with reference to Figure 7, at time $t = 5$, $A_1$ and $A_2$ detect a possible collision. $A_1$ behaves correctly, applies the interaction rule RIGHT ($t = 6$), and turns right. On the contrary, $A_2$ instead misbehaves because it keeps the FAST maneuver instead of applying the RIGHT maneuver. $A_2$ is clearly uncooperative and this is highlighted by the fact that its (actual) trajectory is different from the one that would be expected from the interaction rules. It is worth noting that at time $t = 5$, $A_2$ and $A_3$ detect a possible collision too. However, they both behave correctly and perform the RIGHT maneuver ($t = 6$).

UAVs $A_1$, $A_3$, and $A_4$ monitor $A_2$ by combining the information read from the on-board sensors with the information received from the other UAVs in order to learn whether $A_2$ is cooperative or not. Experiments show that $A_1$, $A_3$, and $A_4$ correctly execute the monitor and achieve consensus on the non-cooperativeness of $A_2$. Indeed, Figure 8 shows the run of $A_1$’s monitor. Runs of $A_3$’s and $A_4$’s monitor look like the same. The actual trajectory of $A_2$ differs from the expected one. The instant in which the two trajectories diverge is the instant in which the collision with $A_1$ is detected and the right maneuver should start. The correct function of the misbehavior detector is shown by the fact that $A_3$ triggers an alarm for the misbehavior of $A_2$ allowing the system to take adequate countermeasures.

As a final consideration, the proposed methodology makes it possible to detect dynamic misbehavior, in other terms, it does not require to know in advance either which agent will misbehave or the instant the misbehavior will begin. In order to keep the experiment simple, we made $A_1$, $A_3$, and $A_4$ monitor $A_2$. However, they “knew” i) neither whether $A_2$ was starting misbehaving at some point, ii) nor the instant of misbehavior began. The ability of managing dynamic misbehavior requires the ability for a UAV to
continuously monitor its neighbourhood. Furthermore, it requires a proper management of the consensus protocol. The proposed consensus-based algorithm converges to an agreement with a speed that depends on the connectivity of the communication graph [31]. In case of high velocity of convergence, the protocol can be re-initiated each time an agreement is reached in order to continuously monitor the target robot and detect possible dynamic misbehavior. Otherwise, the observing agents must have sufficiently computational power to handle parallel executions of the consensus protocol instantiated at different initial time. Notice that in the experiment we assumed UAVs to be one-hop away from one another so featuring the highest speed of convergence.

VII. CONCLUSION AND FUTURE WORK

In this paper we have presented a method for designing distributed algorithms for detecting misbehavior in systems composed of a group of autonomous cooperative objects. The detection mechanism that we have presented gives robots the ability to monitor the behavior of neighbors and to detect robots that do not follow the assigned interaction rules, due to spontaneous failure or malicious intent. The method is fully distributed and is based on a local monitor that can be systematically built once the interaction rules are specified. The method is general and can be applied to a wide range of applications. It has been tested with simple experiments involving real UAVs: results have been encouraging and motivate future research on this topic aiming at using our method to monitor real systems. Furthermore, starting from the previous experience on work [14], [34], future developments will consider the Byzantine Generals disagreement problem for the consensus approach in order to add the necessary redundancy in sensors and make the detection protocol robust their arbitrary failure.

ACKNOWLEDGMENTS

This work has been supported by the European Commission within the Integrated Project PLANET, “PLAtform for the deployment and operation of heterogeneous NETworked cooperating objects” (grant
no. FP7-2010-257649 PLANET); the Italian Ministry of Education, University and Research within the PRIN project TENACE, “Protecting National Critical Infrastructures from Cyber Threats” (Grant no. 20103P34XC_008); and, the Tuscany Region within the project PITAGORA, “Innovative technologies and processes for Airport Management” under the POR CReO 2007-2013 Framework. Finally we are indebted with reviewers whose comments helped us to greatly improve the paper.

REFERENCES


Simone Martini
Simone Martini received his Master and PhD Degree in “Automation and Robotics Engineering” in 2008 and 2012, respectively, from the University of Pisa. Since 2012 he has bee PostDoc at Research Center “E. Piaggio” of the University of Pisa. His interests are in the field of distributed algorithms for robust and secure network agreement.

Davide Di Baccio
Davide Di Baccio received his Master in “Automation and Robotics Engineering” from the University of Pisa in 2009. Since then to 2014 he has been Research Fellow at Research Center “E. Piaggio”. Since May 2014, he has been working for Evidence S.r.l., Pisa (IT). His interests are in the field of control and distributed robotics.

Francisco Alarcón Romero
Francisco Alarcón Romero received both his Master Degree in Telecommunication Engineering and E.T.S. Degree in Computer Engineering from the University of Seville, Spain, in 2008 and 2013, respectively. From 2007 to 2009, he was Research Fellow at the University of Seville and the Andalusian Association for Research and Industrial Cooperation (AICIA). Since 2009 he has been working for the Center for Advanced Aerospace Technologies (CATEC), Seville, Spain. His interests are in the field of avionics and unmanned aerial systems.

Antidio Viguria
Antidio Viguria received the “Engineering” degree in telecommunication from the University of Seville, Spain, in 2004. From 2004 to 2006 he was working as a researcher at the University of Seville. From 2006 to 2008, he was a Fulbright scholar and a graduate student at the Georgia Institute of Technology. Since 2008 he is with the Center for Advanced Aerospace Technologies (CATEC), Seville, Spain, where he is Head of Avionics and Unmanned Systems Department. His main research interests are in multi-robot coordination and cooperation and robot architectures.

Lucia Pallottino
Lucia Pallottino received the “Laurea” degree in Mathematics from the University of Pisa in 1996, and the Ph.D. degree in Robotics and Industrial Automation degree from the University of Pisa in 2002. She has been Visiting Scholar in the Laboratory for Information and Decision Systems at MIT, Cambridge, MA and Visiting Researcher in the Mechanical and Aerospace Engineering Department at UCLA, Los Angeles, CA. She joined the Faculty of Engineering in the University of Pisa as an Assistant Professor in 2007. Her main research interests within Robotics are in optimal motion planning and control, multi-agent systems and cooperating objects.

Gianluca Dini
Gianluca Dini received his Electronics Engineering Degree from the University of Pisa (IT), in 1990 and his PhD in Computer Engineering from Scuola Superiore S.Anna, Pisa (IT), in 1995. In 1993 he was research fellow at the Department of Computer Science of the University of Twente (NL). In 1999 he was adjunct professor at the University of Siena. From 1993 to 2000 he was assistant professor at the University of Pisa where is now Associate Professor. In 2014, he got the National Scientific Qualification to function as Full Professor in Computer Science. He has been (co)author of 100+ papers in international journal and conferences. He has been
principal investigator in quite a few projects funded by the European Union, the Italian Government, and private companies. His research interests are in the field of distributed systems, operating systems, and cyber-security.

**Aníbal Ollero**

Aníbal Ollero is the head of the GRVC group and Full Professor at the University of Seville. He received his Electrical Engineering degree (1976) and the Doctor Engineer degree (1980) with doctoral award from the University of Seville. He has been full professor at the Spanish Universities of Santiago and Malaga and researcher at LAAS-CNRS (France) and Carnegie Mellon University (USA). He has led or participated in more than 110 R&D projects, including 17 projects funded by the European Commission (Information and Communication Technologies, Information Society Technologies, Energy Environment and Sustainable Development, ESPRIT, Telematics Applications, Environment and Climate, CRAFT-BRITE) and other projects funded by NASA, the Spanish National Research Program, the Regional Research Program, and many institutions and companies. He has been scientific and technical coordinator of FP5 COMETS, coordinator of FP6 AWARE, coordinator of FP7 ARCAS and EC-SAFEMOBIL and Associated Coordinator of FP7 PLANET and CONET projects. Professor Ollero has about 440 publications including papers in journals (more than 100), book chapters (28), and conference proceedings. He is author or co-author of seven books including a book on computer control “Premio Mundo Electrónico” (Spanish Award), a book on robotics, the book “Intelligent Mobile Robot Navigation” (Springer, 2005), a book on “Teleoperation and Telerobotics” (Pearson–Prentice Hall, 2006) and a book on autonomous and distributed systems for applications on vehicles and natural environments (2008). Furthermore, he is editor or co-editor of 10 books including “Multiple Heterogeneous Unmanned Aerial Vehicles” (Springer, 2007), with nine chapters, being co-author of six, and “Cooperating Objects and Wireless Sensor Networks” (Hermes, 2007), with six Chapters being co-author of three. He is associated editor of the Journal of Field Robotics (Wiley), editor at large (Europe) of the Journal of Intelligent and Robotics Systems (Springer) and the Revista Iberoamericana de Automática e Informática. He has been editor of Control Engineering Practice (1993–2005) and IEEE Transactions on Systems Man and Cybernetics (1995–2008).
Simone Martini

Davide Di Baccio
Francisco Alarcón Romero

Antidio Viguria
Lucia Pallottino

Gianluca Dini
Aníbal Ollero
Highlights

A distributed misbehaviour detection algorithm is proposed

The algorithm detects autonomous robots that violate agreed interaction rules

The algorithm is based on topologies-based consensus

The algorithm detects anomalies in high-level manoeuvres

The algorithm has been successfully tested in a collision-avoidance scenario involving UAVs