ARTICLE IN PRESS
Robotics and Autonomous Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Distributed motion misbehavior detection in teams of heterogeneous aerial robots

Simone Martini [a], Davide Di Baccio [a], Francisco Alarcón Romero [b], Antidio Viguria Jiménez [b], Lucia Pallottino [a,c], Gianluca Dini [a,c,*], Aníbal Ollero [d]

[a] Research Center "E. Piaggio", University of Pisa, Pisa, Italy
[b] Fundación Andaluza para el Desarrollo Aeroespacial (FADA-CATEC), Seville, Spain
[c] Department of Ingegneria dell'Informazione, University of Pisa, Pisa, Italy
[d] GRVC, University of Seville, Seville, Spain

HIGHLIGHTS

- A distributed misbehavior 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 maneuvers.
- The algorithm has been successfully tested in a collision-avoidance scenario involving UAVs.

ARTICLE INFO

Article history:
Received 20 January 2015
Received in revised form
4 June 2015
Accepted 15 June 2015
Available online xxxx

Keywords:
Multi robot coordination
Distributed misbehavior detection
Topologies-based consensus
Rule-based collision avoidance
UAV experiments

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.

1. Introduction

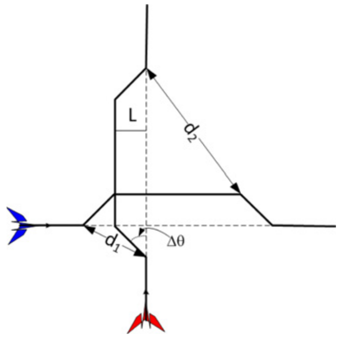
The availability of distributed systems gave rise in the late 80s to a profound rethinking of many decision making problems and enabled solutions that were impossible before. A similar trend is now happening in control and will soon enable a formidable number of new robotic applications. Various distributed control policies have been proposed for formation control, flocking, sensor coverage, and intelligent transportation (see e.g. [1–3]). 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

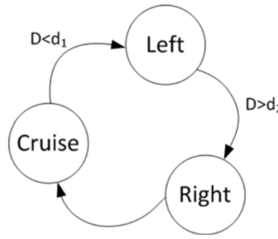
* Corresponding author at: Department of Ingegneria dell'Informazione, University of Pisa, Pisa, Italy.
E-mail address: g.dini@iet.unipi.it (G. Dini).

http://dx.doi.org/10.1016/j.robot.2015.06.008
0921-8890/

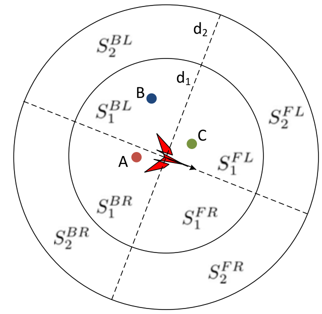
Please cite this article in press as: S. Martini, et al., Distributed motion misbehavior detection in teams of heterogeneous aerial robots, Robotics and Autonomous Systems (2015), http://dx.doi.org/10.1016/j.robot.2015.06.008



(a) Evolution of aircraft based on the collision avoidance policy.



(b) Hybrid system associated to the collision avoidance policy.



(c) Sectors associated to each aircraft.

Fig. 1. Simplified version of collision avoidance strategy proposed in [15] reported in Example 1.

many works that have been presented in literature (see e.g., [4–9]). 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.

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 [12–14]. 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 2 a case study example to help the reader follow the notation introduced in Section 3 where the hybrid model of the proposed cooperation protocol is reported. The misbehavior detection problem is formally defined in Section 4. The Boolean consensus misbehavior detection strategy is described in Section 5 where the convergence in a finite number of steps is formally proved. Finally, Section 6 discusses a case-study and the related experimental results.

2. 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 behavior 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

$$\begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta. \end{cases} \quad (1)$$

Please cite this article in press as: S. Martini, et al., 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

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_1^{FL}, S_2^{FL}), front right (S_1^{FR}, S_2^{FR}), back left (S_1^{BL}, S_2^{BL}) and back right (S_1^{BR}, S_2^{BR}) sectors. In order to monitor the behavior 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_1^{BR} of aircraft h, it can detect aircraft k only if it is at distance less than d_2 and inside sectors S_1^{BL}, S_2^{BL}, S_1^{FR}, S_2^{FR}, S_1^{BR} and S_2^{BR} but not in S_1^{FL} or S_2^{FL}. 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].

3. 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 $\mathcal{A}_1, \ldots, \mathcal{A}_n$, where $\mathcal{A}_i$ is described by a vector $\mathbf{q}_i$ in a continuous configuration space $\mathcal{Q}$. 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 $R \overset{\text{def}}{=} \{\mathsf{rule}_1, \ldots, \mathsf{rule}_m\}$ 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 $\mathcal{A}_i$ is specified by:

- A configuration vector $\mathbf{q}_i \in \mathcal{Q}$, where $\mathcal{Q}$ is a configuration space ($\mathbf{q} = (x, y) \in \mathbb{R}^2$ in the case study Example 1).
- An input vector $\mathbf{u}_i \in \mathcal{U}$, 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 Σ is the set of operating modes; ($\Sigma_i = \{Cruise, Left, Right\}$ in the case study).
- A dynamic map f_i describing how the agent's configuration is updated:

$$\dot{\mathbf{q}}_i(t) = f_i(\mathbf{q}_i(t), \mathbf{u}_i(t)) \tag{2}$$

  (for Example 1 the dynamic map is reported in (1)).
- A decoder map $\mathcal{G}_i$ describing which control values are applied in different operating modes σ_i, i.e.

$$\mathbf{u}_i(t) = \mathcal{G}_i(\mathbf{q}_i(t), \sigma_i(t_k)), \quad \text{for } t \in [t_k, t_{k+1}).$$

  In Example 1 we have $w(t) = \mathcal{G}_i(q_i(t), Cruise) = 0$, while $w(t) = \mathcal{G}_i(\mathbf{q}_i(t), Left)$ is a sequence of $\Delta\theta, 0, -\Delta\theta$ and 0 again. Finally, $w(t) = \mathcal{G}_i(\mathbf{q}_i(t), Right)$ is a sequence of $-\Delta\theta, 0, \Delta\theta$ and 0 again.
- A set of *topologies* $\eta_{i,1}(\mathbf{q}), \ldots, \eta_{i,\kappa_i}(\mathbf{q})$ on $\mathcal{Q}$, whose union defines the agent's neighborhood in $\mathbf{q}$, i.e. $N(\mathbf{q}_i) = \cup_{j=1}^{\kappa_i} \eta_{i,j}(\mathbf{q}_i)$. The set of neighboring agents is hence $N_i = \{\mathcal{A}_k | \mathbf{q}_k \in N(\mathbf{q}_i)\}$, while the set of neighbors' configurations is $I_i = \{\mathbf{q}_k \in \mathcal{Q} | \mathcal{A}_k \in N_i\}$, referred in the following as *influence set* of $\mathcal{A}_i$.

  Referring to the case study described in Example 1, for aircraft i in $\mathbf{q}$ we have eight topologies corresponding to the eight sectors of the detection disc centered in $\mathbf{q}$. The neighboring agents are aircraft in the detection disc, i.e. aircraft that are closer to i more than d_2.
- An event vector $\mathbf{s}_i \in \mathbb{B}^{\kappa_i}$ (whose components will be later referred to as *sub-events*) and a detection map $\mathcal{S}_i$ involving conditions over $\mathcal{Q}$ (as e.g. the presence of another agent in a specific region):

$$s_{i,j}(t) = \sum_{\mathbf{q}_k \in I_i} \mathbf{1}_{\eta_{i,j}(\mathbf{q}_i)}(\mathbf{q}_k)$$

  where $\sum$ represents the logical sum (*or*), and $\mathbf{1}_A(x)$ 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* φ_i indicating the detector condition $\mathbf{c}_i$ based on events vector $\mathbf{s}_i$:

$$\mathbf{c}_i(t_k) = \varphi_i(s_{i,1}(t_k), \ldots, s_{i,\kappa_i}(t_k));$$

  in other words, $\mathbf{c}_i(t_k)$ is a vector of logic operations of sub-events $s_{i,j}$.
- An automaton δ_i describing how the agent's current discrete state (or mode of operation) σ_i is updated based on the detector condition $\mathbf{c}_i$:

$$\sigma_i(t_{k+1}) = \delta_i(\sigma_i(t), \mathbf{c}_i(t_k)). \tag{3}$$

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*, $\Sigma \overset{\text{def}}{=} \{\mathsf{mode}_1, \ldots, \mathsf{mode}_p\}$, and the set of ν logical propositions, or *events*, $E \overset{\text{def}}{=} \{\mathsf{event}_1, \ldots, \mathsf{event}_\nu\}$. The occurrence of any of these events requires the current mode σ_i of the generic agent $\mathcal{A}_i$ to be changed. The generic event event_l measured from $\mathcal{A}_i$ can be assigned with a logical variable $c_{i,l} \in \mathbb{B}$ taking the value true if event_l has been recognized by $\mathcal{A}_i$ and false otherwise. Although event_l 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 σ_i must be updated. Hence, the *cooperation manager* can be seen as a Discrete Event System (DES) [21], and indeed an *automaton* (see Fig. 2), that receives $\mathbf{c}_l$ as input and updates its state σ_i according to rules of the forms

- $\mathsf{rule}_0 \overset{\text{def}}{=} (\sigma(t_0) = \mathsf{mode}_1)$; $\leftarrow$ start in mode_1;
- $\mathsf{rule}_j \overset{\text{def}}{=} (\textbf{if } \sigma_i(t_k) = \mathsf{mode}_l \textbf{ and } c_m(t_k) = \text{true} \textbf{ then } \sigma_i(t_{k+1}) = \mathsf{mode}_p)$.

Referring to the case study, mode_1 is *Cruise* and one of the rules is (if $\sigma_i(t_k) = \mathsf{Cruise}$ and $c_l(t_k) =$ "aircraft detected at distance less than d_1" $=$ true then $\sigma_i(t_{k+1}) = \mathsf{Left}$).

The *decoder* connects the output of 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

$$\mathbf{u}_i(t) = \mathcal{G}(\mathbf{q}_i(t), \sigma_i(t_k)), \tag{4}$$

so that the autonomous controlled system

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

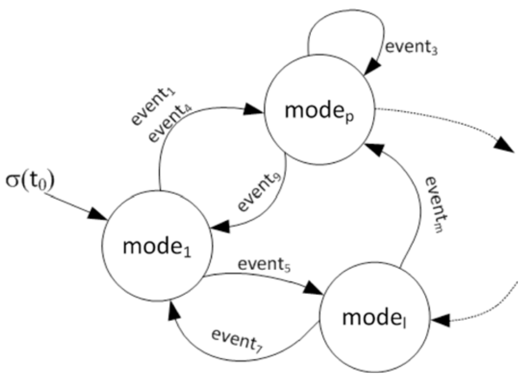


Fig. 2. Representation of dynamics δ of the automaton of the generic cooperation manager.

correctly performs the control planned for the mode $\sigma_i(t_k)$. The decoder is an application $\mathcal{G} : \mathcal{Q} \times \Sigma \rightarrow \mathcal{U}$ that returns the actuators' input during the interval $[t_k, t_{k+1})$ (up to next event). In this perspective, $\mathcal{G}$ 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 $\mathcal{A}_i$ must be able to plan its motion according to its own configuration $\mathbf{q}_i$ 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 $\mathcal{A}_i$.

Due to limited visibility of its sensors, an agent $\mathcal{A}_i$ is able to measure the configuration $\mathbf{q}_j$ of another agent $\mathcal{A}_j$ laying in its *visible region* $\mathcal{V}_i$. This region changes with time depending on the configurations of $\mathcal{A}_i$ and its neighbors, i.e. $\mathcal{V}_i(t) = \mathcal{V}(\mathbf{q}_i(t), Q(t))$. The remaining part of the configuration space, namely $\bar{\mathcal{V}}_i(t) = \mathcal{Q} \setminus \mathcal{V}_i(t)$, is the non-visible region and is composed of all those configurations that cannot be "seen" from $\mathcal{A}_i$. Note that our problem also requires the knowledge of the state (σ_j) of an agent's cooperation manager, that is unmeasurable and thus will be estimated. For simplicity, we assume that $I_i \subseteq \mathcal{V}_i$, 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 $\mathcal{H} = (\mathbf{q}_i, \sigma_i, I_i)$. The behavior of $\mathcal{A}_i$ can be written more compactly as

$$\begin{cases} (\dot{\mathbf{q}}_i(t), \sigma_i(t_{k+1})) = \mathcal{H}_i(\mathbf{q}_i(t), \sigma_i(t_k), I_i(t)), \\ (\mathbf{q}_i(0), \sigma_i(0)) = (\mathbf{q}_i^0, \sigma_i^0), \end{cases} \tag{5}$$

where $\mathcal{H}_i : \mathcal{Q} \times \Sigma_i \times \mathcal{Q}^{n_i} \rightarrow T_{\mathcal{Q}} \times \Sigma_i$ is the agent's *hybrid dynamic map* [22] and T_Q is the tangent space of Q. We will denote with $\phi_{\mathcal{H}_i}(\mathbf{q}_i(t), \sigma_i(t), I_i(t))$ the evolution of system (5).

4. 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 $\mathbf{q}_h(t)$ 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. [23–25]). 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 σ is started at the generic discrete time $t_k \stackrel{\text{def}}{=} kT$ and is run up to t_{k+1}. 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 $\bar{\mathbf{q}}_h(t)$ measured by the observer robot is close enough to the evolution of the cooperative model $\mathcal{H}$, i.e.

$$\|\bar{\mathbf{q}}_h(t) - \pi_{\mathcal{Q}}(\phi_{\mathcal{H}_i}(\bar{\mathbf{q}}_i(t_k), \bar{\sigma}_i(t_k), I_i(t)))\| \leq \epsilon, \quad \forall t,$$

where $\|\cdot\|$ is the Hausdorff distance, $\pi_{\mathcal{Q}}$ is the projector over the set $\mathcal{Q}$, and ϵ 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 behavior 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

$$\begin{aligned} I_h &= I_h \cap \mathcal{Q} = I_h \cap (\mathcal{V}_i \cup \bar{\mathcal{V}}_i) \\ &= (I_h \cap \mathcal{V}_i) \cup (I_h \cap \bar{\mathcal{V}}_i) \stackrel{\text{def}}{=} I_h^{obs} \cup I_h^{unobs}. \end{aligned}$$

Reorder the model's inputs as $I_h = (I_h^{obs}, I_h^{unobs})$, where $I_h^{obs} \stackrel{\text{def}}{=} q_{i,1}, \ldots, q_{i,\nu_i}$ is the list of configurations known to $\mathcal{A}_i$, and $I_h^{unobs} \stackrel{\text{def}}{=} q_{i,\nu_i+1}, \ldots, q_{i,n_h}$ 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 $\bar{\mathbf{q}}_h(t)$, a list $I_h^{obs}(t_k)$ of known configurations, a visibility region $\mathcal{V}_i$, and a desired accuracy ϵ, determine if there exists an input $\hat{I}_h(t_k) = (I_h^{obs}(t_k), \hat{I}_h^{unobs}(t_k))$ s.t.

$$\|\bar{\mathbf{q}}_h(t) - \pi_{\mathcal{Q}}(\phi_{\mathcal{H}_i}(\bar{\mathbf{q}}_i(t_k), \bar{\sigma}_i(t_k), \hat{I_i(t)}))\| \leq \epsilon, \quad \forall t \in [t_k, t_{k+1})$$

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.

4.1. Construction of the local monitor

The proposed approach is a two-step process: first, $\mathcal{A}_i$ computes an a-priori prediction of the set of possible trajectories that $\mathcal{A}_h$ 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 $\mathcal{A}_h$ and measured by $\mathcal{A}_i$ and if none of them results close enough, $\mathcal{A}_h$ 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 $\mathcal{A}_h$ can perform based on local information, and whose transitions $\tilde{\delta}$ are the same as in δ. The main challenge in the construction of the automaton is the estimation of an upper approximation $\tilde{\mathbf{c}}_i$ of each detector condition $\mathbf{c}_h$, 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 $\tilde{\mathcal{H}}_h$.

Please cite this article in press as: S. Martini, et al., 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

5. 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.

5.1. The consensus algorithms

Consider a network whose communication topology is represented by an undirected graph G which is composed by a set of nodes $V = \{v_1, \dots, v_n\}$ 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 $\mathcal{A}_i \in N_j$ there exists an arc from node i to node j associated to robots $\mathcal{A}_i$ and $\mathcal{A}_j$ 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 $\xi : S \times S \to S$ which is used to compute

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

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 $\mathcal{A}_h$. Consider the vector

$$\mathbf{R}_h(t_k) = \begin{bmatrix} r_h^{(1)}(t_k) \\ r_h^{(2)}(t_k) \\ \vdots \\ r_h^{(n)}(t_k) \end{bmatrix} \tag{6}$$

where $r_h^{(i)}(t_k)$ represents the reputation of agent $\mathcal{A}_i$ about agent $\mathcal{A}_h$ after t_k steps of the consensus iterative procedure. Our aim is to design a distributed consensus algorithm allowing us to have $\lim_{k\to\infty} \mathbf{R}_h(t_k) = \mathbf{1} r_h^*$, 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 ξ.

A possible solution allowing us to reach an agreement consists in letting agents to share the locally estimated *encoder map* φ_h of target robot $\mathcal{A}_h$. 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 φ_h (or equivalently on the events vector $\mathbf{c}_h$ associated to agent $\mathcal{A}_h$) is the novel contribution of this paper with respect to related works [12–14].

5.2. Convergence of consensus algorithm

Given a target robot $\mathcal{A}_h$ 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 $\mathbf{c}_h$ that influence the behavior of $\mathcal{A}_h$. Such information is elaborated by each agent and flows among them through a communication network. Once an agreement on the events $\mathbf{c}_h$ is reached, the evolution of $\mathcal{A}_h$ is compared with the evolution determined by $\mathbf{c}_h$ and the hybrid dynamic map in (5). If they are not sufficiently close, $\mathcal{A}_h$ is classified as a *misbehaving agent*.

As mentioned, the information exchanged is the event vector $\mathbf{s}_h$ estimated by each robot. More formally, we consider a state vector $\mathbf{x}_i = (x_{i,1}, \dots, x_{i,\kappa_h}) \in \mathbb{B}^{1\times\kappa_h}$, that is a string of bits representing the values that observing agent $\mathcal{A}_i$ may assign to all sub-events that influence the evolution of agent $\mathcal{A}_h$.

Let $X(t) = (\mathbf{x}_1(t)^T, \dots, \mathbf{x}_n(t)^T)^T$ be the matrix in $\mathbb{B}^{n\times\kappa_h}$ that represents the network state at the time t. We assume that each agent is a dynamic node that updates its local state $\mathbf{x}_i$ 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. $\mathbf{x}_i(t+1) = F_i(X(t))$. Moreover we assume that every agent is able to produce the logical output vector $Y = (\mathbf{y}_1(t)^T, \dots, \mathbf{y}_n(t)^T)^T \in \mathbb{B}^{n\times\nu_h}$ that corresponds to the detector condition (or events vector) $\mathbf{c}_h$, estimated by the n observing robots, by using an output function D depending on the local state, i.e. $\mathbf{y}_i = D_i(X_i) = \mathbf{c}_h^{(i)} = (c_{h,1}^{(i)}, \dots, c_{h,\nu_h}^{(i)})$ is the event vector $\mathbf{c}_h$ estimated by robot $\mathcal{A}_i$.

In the most general case, the generic observing agent i may or may not be able to measure the value of the jth sub-event associated to $\mathcal{A}_h$, i.e. $s_{h,j}$. In this sense, we can conveniently introduce a *visibility matrix* $V \in \mathbb{B}^{n\times\kappa_h}$ where $V_{i,j} = 1$ if, and only if, agent $\mathcal{A}_i$ is able to measure $s_{h,j}$ and $V_{i,j} = 0$ 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:

$$\begin{cases} X(t+1) = F(X(t)) \\ \mathbf{x}_i(0) = \tilde{\mathbf{U}}^{(i)} \\ Y(t) = D(X(t)) \end{cases} \tag{7}$$

where $F : \mathbb{B}^{n\times\kappa_h} \to \mathbb{B}^{n\times\kappa_h}$, $D = Diag(D_1, \dots, D_n)$ with $D_i : \mathbb{B}^{\kappa_h} \to \mathbb{B}^{\nu_h}$, and $\tilde{\mathbf{U}}^{(i)}$ provides $\tilde{\mathbf{s}}_h^{(i)}$ that is an initial lower approximation of $\mathbf{s}_h = (s_{h,1}, \dots, s_{h,\kappa_h})$ based only on observation of the neighborhood of $\mathcal{A}_h$ operated by the ith agent. In this paper, componentwise inequalities are considered, i.e. a lower approximation is a vector whose jth component is less or equal to the jth component of $\mathbf{s}_h$.

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 $X(0) = (\tilde{\mathbf{s}}_h^{(1)T}, \dots, \tilde{\mathbf{s}}_h^{(n)T})^T$, and a visibility matrix V with non-null columns, the distributed logical consensus system*

$$\begin{cases} x_{i,j}^+ = V_{i,j}\, x_{i,j} + \neg V_{i,j} \left(\sum_{k\in N_i} x_{k,j} \right), \\ x_{i,j}(0) = \tilde{s}_{h,j}^{(i)}, \end{cases} \tag{8}$$

with $i = 1, \dots, n$ and $j = 1, \dots, \kappa_h$ converges to the consensus state $X = \mathbf{1}_n\, \mathbf{s}_h$ in a number of steps that is less than the graph diameter.

Please cite this article in press as: S. Martini, et al., 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

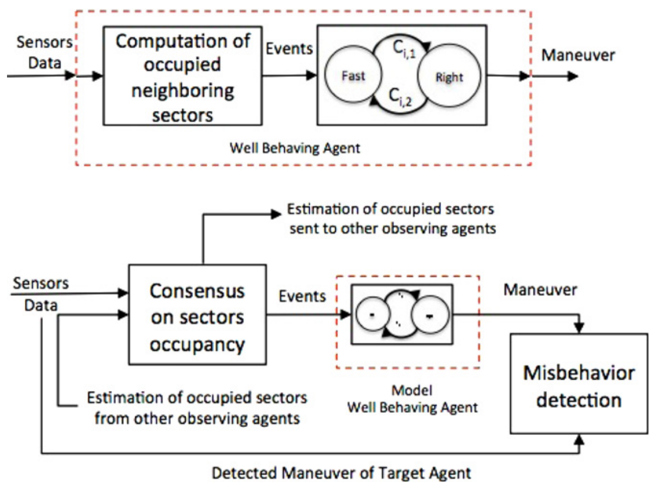


Fig. 3. Work flow of the misbehavior detection algorithm.

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 Eq. (8) reduces to

$$x^+_{i,j} = \neg V_{i,j}\left(\sum_{k\in N_i} x_{k,j}\right) = \sum_{k\in N_i} x_{k,j}.$$

Moreover, since each column of V is non-null by hypothesis, there is at least one observing robot, say the mth, with complete visibility on the jth topology $\eta_{h,j}$, which implies $\tilde{s}^{(m)}_{h,j} = s_{h,j}$. Note that we have $\tilde{s}^{(m)}_{h,j} \geq \tilde{s}^{(i)}_{h,j}$ since $\tilde{s}^{(i)}_{h,j}$ 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 $\bar{N} \leq Diam(G) < \infty$ after which

$$x_{i,j} = \sum_{k=1}^{n} x_{k,j} = \tilde{s}^{(q)}_{h,j} = s_{h,j}.$$

If instead $V_{i,j} = 1$, agent i has complete knowledge of the jth 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 $\tilde{s}^{(i)}_{h,j} = s_{h,j}$. 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 $\mathbf{s}_h$ 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. $\varphi(\mathbf{s}_h) = \mathbf{c}_h$, 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 $r^{(i)}_h(\bar{N}) = r^*_h$ with $\bar{N} < \infty$ and $i = 1, \ldots, n$. With Theorem 1 we proved that $X(\bar{N}) = \mathbf{1}_n\, \mathbf{s}_h$. This means that $\Phi_i(\mathbf{x}_i) = \varphi(\mathbf{s}_h) = \mathbf{c}_h$ and the predictor $\tilde{\mathcal{H}}^{(i)}_h(\tilde{\mathbf{q}}^{(i)}_h, \tilde{\sigma}^{(i)}_h, \tilde{I}^{(i)}_h)$, $i = 1, \ldots, n$, (introduced in Section 4.1) is initialized with the value $\tilde{\sigma}^{(i)}_h(0)$ corresponding to the most conservative hypothesis on the activation of $\mathbf{c}_h$ which is the same for all observing agents, i.e. $\tilde{\sigma}^{(i)}_h(0) = \tilde{\sigma}^*_h(0)$, $i = 1, \ldots, n$ where $\tilde{\sigma}^*_h(0) = \tilde{\sigma}_h(0)$ in the case the estimated event vector $\tilde{\mathbf{c}}_h$ is equal to $\mathbf{c}_h$. Thus, the estimated state $\tilde{\sigma}^{(i)}_h$, $i = 1, \ldots, n$, becomes

$$\begin{aligned}\tilde{\sigma}^{(i)}_h(t_{k+1}) &= \tilde{\delta}(\tilde{\sigma}^{(i)}_h(t_k), \tilde{\mathbf{c}}^{(i)}_h(t_{k+1})) \\ &= \tilde{\delta}(\tilde{\sigma}^*_h(t_k), \tilde{\mathbf{c}}_h(t_{k+1})) = \tilde{\sigma}^*_h(t_{k+1}).\end{aligned}$$

According to this, we can compute α_r as

$$\|\bar{\mathbf{q}}_h(t) - \pi_{\mathcal{Q}}(\phi_{\tilde{\mathcal{H}}}(\bar{\mathbf{q}}_h(t)\,,{}^i\,\tilde{\sigma}^{(i)}_h(t_k), I^i_h(t)))\| = \alpha_r,$$
$$\forall t \in [t_k, t_{k+1}),\ i = 1, \ldots, n. \tag{9}$$

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. $r^{(i)}_h =$ misbehaving, $i = 1, \ldots, n$, which proves the theorem. ∎

6. Discussion

6.1. On practical applicability of the proposed approach

The proposed misbehavior detection method is based on the concept of topologies $\eta_{i,j}$ (see Section 3) 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 behavior 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 ϵ 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 6.2 a misbehavior corresponds to an aircraft that goes straight on while it is supposed to turn right. Hence the value of ϵ 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 6.2.2, the value of the threshold ϵ is chosen as 50 m.

For simplicity the workflow of the proposed method is reported in Fig. 3.

6.2. An experimental case-study

In order to practically verify the effectiveness of the theory presented in the previous sections, we consider a scenario involving several UAVs that cooperate to avoid collisions and maintain safety. In order to achieve this goal, UAVs take the same set of maneuvers, namely, (i) to accelerate up to maximum 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 6.2.1 describes the experimental setting whereas Section 6.2.2 describes the experiment and the related results.

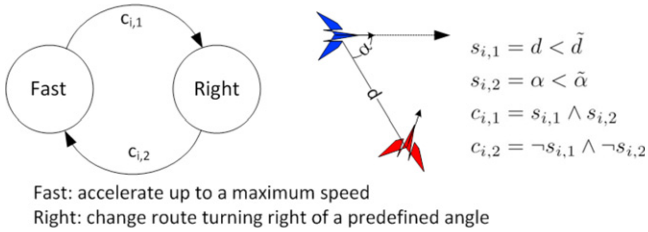


Fig. 4. Interaction rules with related automaton used in the experiments.

6.2.1. 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. 3(b). 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. 3(a)) 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. 3(a) is explicitly reported in Fig. 4 with the associated events. It has been implemented according to the model defined in Section 3. 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 a 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 kg. It has a minimum endurance of 30 min since it uses an electrical motor powered with LiPo batteries. The payload is 250–300 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.

6.2.2. Experimental results

With reference to the system composed of the four autonomous UAVs presented above, we denote by $\mathcal{A}_1$ (cyan) and $\mathcal{A}_4$ (blue) the virtual UAVs, $\mathcal{A}_2$ (red) the Locomove, and, finally, $\mathcal{A}_3$ (green) the Skywalker (Fig. 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 $\mathcal{A}_2$, Locomove, that, at some point, violates the interaction rules specified in Fig. 4. Therefore, in the experiment $\mathcal{A}_2$ is the target and all other UAVs monitor it.

More precisely, with reference to Fig. 7, at time $t = 5$, $\mathcal{A}_1$ and $\mathcal{A}_2$ detect a possible collision. $\mathcal{A}_1$ behaves correctly, applies the interaction rule RIGHT ($t = 6$), and turns right. On the contrary, $\mathcal{A}_2$ instead misbehaves because it keeps the FAST maneuver instead of applying the RIGHT maneuver. $\mathcal{A}_2$ 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 $t = 5$, $\mathcal{A}_2$ and $\mathcal{A}_3$ detect a possible collision too. However, they both behave correctly and perform the RIGHT maneuver ($t = 6$).

UAVs $\mathcal{A}_1$, $\mathcal{A}_3$, and $\mathcal{A}_4$ monitor $\mathcal{A}_2$ by combining the information read from the on-board sensors with the information received from the other UAVs in order to learn whether $\mathcal{A}_2$ is cooperative or not. Experiments show that $\mathcal{A}_1$, $\mathcal{A}_3$, and $\mathcal{A}_4$ correctly execute the monitor and achieve consensus on the non-cooperativeness of $\mathcal{A}_2$. Indeed, Fig. 8 shows the run of $\mathcal{A}_1$'s monitor. Runs of $\mathcal{A}_3$'s and $\mathcal{A}_4$'s monitor look like the same. The actual trajectory of $\mathcal{A}_2$ differs from the expected one. The instant in which the two trajectories diverge is the instant in which the collision with $\mathcal{A}_1$ is detected and the right maneuver should start. The correct function of the misbehavior detector is shown by the fact that $\mathcal{A}_3$ triggers an alarm for the misbehavior of $\mathcal{A}_2$ 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

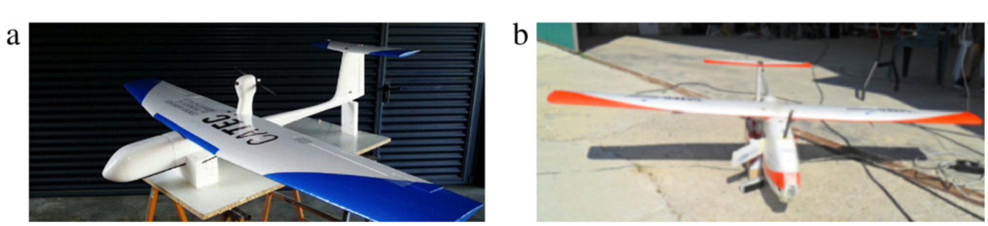


Fig. 5. The Locomove (a) and the Skywalker (b).

Please cite this article in press as: S. Martini, et al., 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

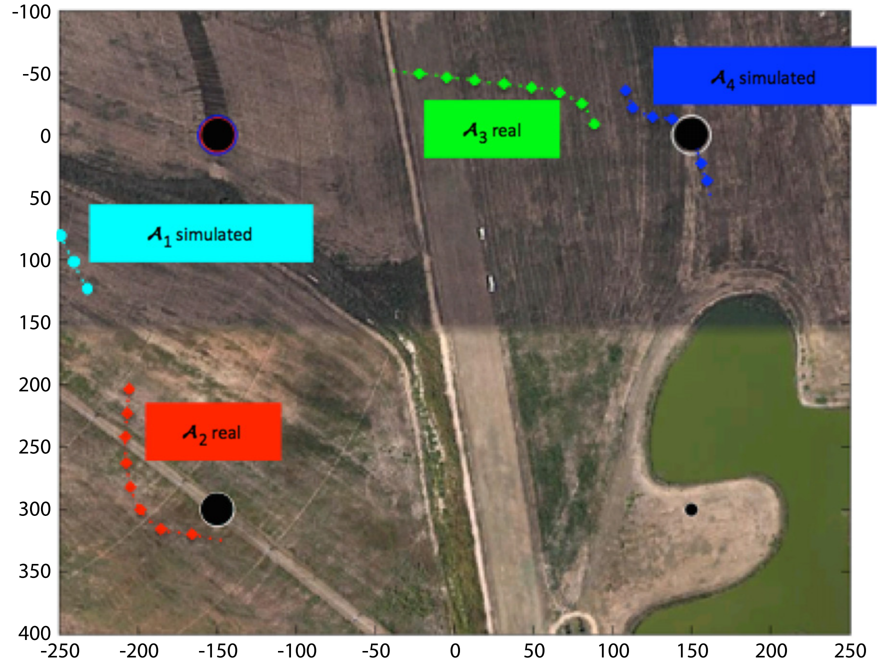
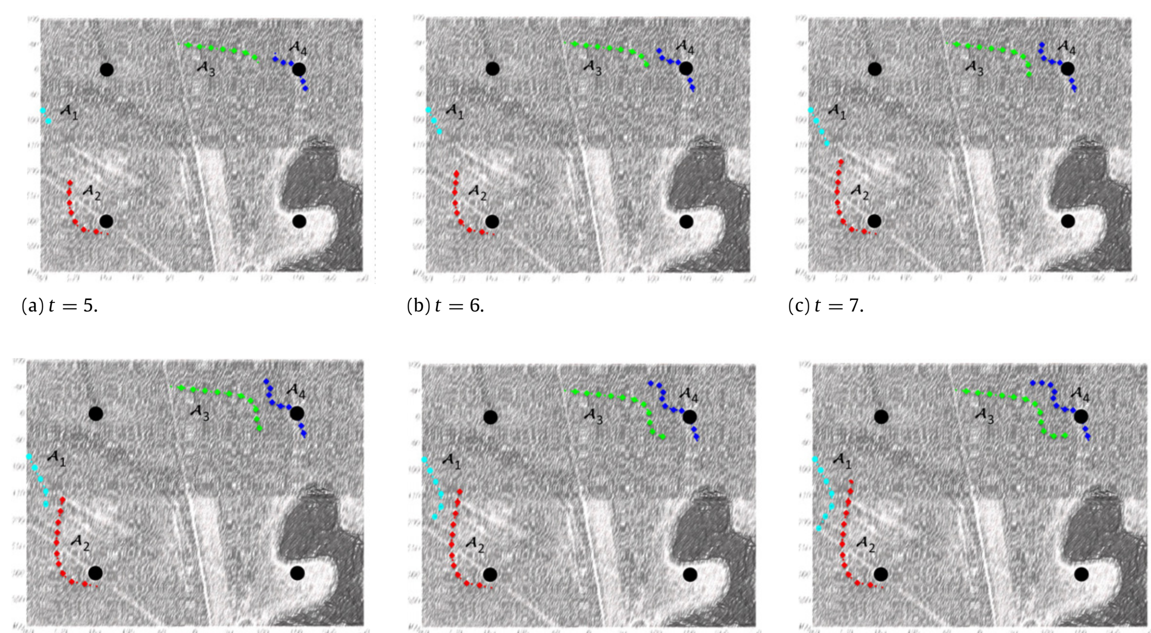


Fig. 6. Picture of the real scenario with plotted trajectories and waypoints. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) $t = 5$.
(b) $t = 6$.
(c) $t = 7$.
(d) $t = 8$.
(e) $t = 9$.
(f) $t = 10$.

Fig. 7. The Locomove (red) is violating the interaction rules since it is not applying any collision avoidance rule. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

misbehave or the instant the misbehavior will begin. In order to keep the experiment simple, we made $\mathcal{A}_1$, $\mathcal{A}_3$, and $\mathcal{A}_4$ monitor $\mathcal{A}_2$. However, they "knew" (i) neither whether $\mathcal{A}_2$ 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 neighborhood. 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.

7. 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

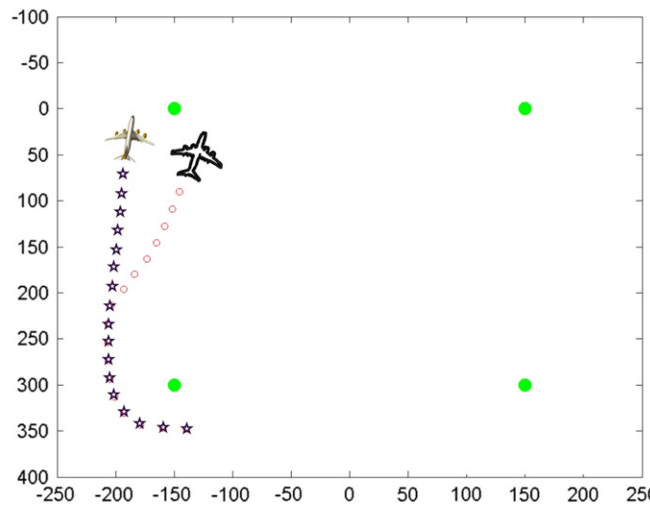


Fig. 8. Run of $\mathcal{A}_1$'s monitor when $\mathcal{A}_2$ misbehaves. Stars denote the real trajectory while circles the expected one.

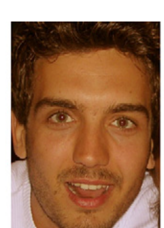
detection mechanism that we have presented gives robots the ability to monitor the behavior of neighbors and to detect robots that do not follow the assigned interaction rules, due to spontaneous failure or malicious intent. The method is fully distributed and is based on a local monitor that can be 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.

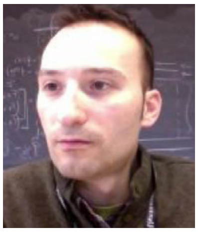
References

[1] R. Olfati-Saber, Flocking for multi-agent dynamic systems: algorithms and theory, IEEE Trans. Automat. Control 51 (3) (2006) 401–420.
[2] A. Danesi, A. Fagiolini, I. Savino, L. Pallottino, R. Schiavi, G. Dini, A. Bicchi, A scalable platform for safe and secure decentralized traffic management of multi-agent mobile system, in: ACM Proc. Real-World Wireless Sensor Network, 2006.
[3] B. McQueen, J. McQueen, Intelligent Transportation Systems Architectures, Artech House Publishers, 1999.
[4] C. Kube, E. Bonabeau, Cooperative transport by ants and robots, Robot. Auton. Syst. 30 (1–2) (2000) 85–102.
[5] L. Pallottino, V. Scordio, A. Bicchi, E. Frazzoli, Decentralized cooperative policy for conflict resolution in multivehicle systems, IEEE Trans. Robot. 23 (6) (2007) 1170–1183.
[6] F. Bullo, J. Cortés, S. Martinez, Distributed Control of Robotic Networks, Princeton University Press, 2007.
[7] R. Olfati-Saber, J. Fax, R. Murray, Consensus and cooperation in networked multi-agent systems, Proc. IEEE 95 (1) (2007) 215.
[8] A. Jadbabaie, J. Lin, A. Morse, Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Automat. Control 48 (6) (2003) 988–1001.
[9] J. Fax, R. Murray, Information flow and cooperative control of vehicle formations, IEEE Trans. Automat. Control 49 (9) (2004) 1465–1476.
[10] A. Fagiolini, M. Pellinacci, G. Valenti, G. Dini, A. Bicchi, Consensus-based distributed intrusion detection for multi-robot systems, in: Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on, 2008, pp. 120–127.
[11] F. Pasqualetti, A. Bicchi, F. Bullo, Distributed intrusion detection for secure consensus computations, in: Proc. 46th IEEE Conf. on Decision and Control, New Orleans, LA, USA, 2007, pp. 5594–5599.
[12] A. Fagiolini, E. Visibelli, A. Bicchi, Logical Consensus for Distributed Network Agreement, in: IEEE Conf. on Decision and Control, 2008, pp. 5250–5255.
[13] A. Fagiolini, S. Martini, D. Di Baccio, A. Bicchi, A self-routing protocol for distributed consensus on logical information, in: IEEE/RSJ International Conference on Intelligent Robots and Systems 2010, in: IROS 2010, 2010.
[14] S. Martini, D. Di Baccio, A. Fagiolini, A. Bicchi, Robust network agreement on logical information, in: 18th IFAC World Congress, IFAC 2011, 2011.
[15] C. Tomlin, G. Pappas, S. Sastry, Conflict resolution for air traffic management: a study in multiagent hybrid systems, IEEE Trans. Automat. Control 43 (4) (1998) 509–521.
[16] J. Kosecka, C. Tomlin, G. Pappas, S. Sastry, Generation of conflict resolution manoeuvres for air traffic management, in: Intelligent Robots and Systems, 1997. IROS'97, Proceedings of the 1997 IEEE/RSJ International Conference on, vol. 3, 1997, pp. 1598–1603.
[17] L. Pallottino, V. Scordio, A. Bicchi, E. Frazzoli, Decentralized cooperative policy for conflict resolution in multivehicle systems, IEEE Trans. Robot. 23 (6) (2007) 1170–1183.
[18] S. Kato, S. Nishiyama, J. Takeno, Coordinating mobile robots by applying traffic rules, in: Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, 1992.
[19] A. Bicchi, A. Danesi, G. Dini, S. La Porta, L. Pallottino, I.M. Savino, R. Schiavi, Heterogeneous wireless multirobot system, IEEE Robot. Autom. Mag. 15 (1) (2008) 62–70.
[20] A. Bicchi, A. Fagiolini, L. Pallottino, Towards a society of robots: Behaviors, misbehaviors, and security, IEEE Robot. Autom. Mag. 17 (4) (2010) 26–36.
[21] C.G. Cassandras, S. Lafortune, Introduction to Discrete Event Systems, Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
[22] J. Lygeros, Lecture notes on hybrid systems, in: Notes for an ENSIETA Workshop, Citeseer, 2004.
[23] S. Pearson, B. Balacheff, Trusted Computing Platforms: TCPA Technology in Context, Prentice Hall PTR, 2003.
[24] B. Chen, R. Morris, Certifying program execution with secure processors, in: USENIX HotOS Workshop, 2003, pp. 133–138.
[25] R. Gallo, H. Kawakami, R. Dahab, Fortuna-a framework for the design and development of hardware-based secure systems, J. Syst. Softw. (2013).
[26] L. Schenato, G. Gamba, A distributed consensus protocol for clock synchronization inwireless sensor network, in: IEEE Conf. on Decision and Control, 2007.
[27] A. Giua, C. Seatzu, Fault detection for discrete event systems using petri nets with unobservable transitions, in: Proc. IEEE Conference on Decision and Control and European Control Conference, 2005, pp. 6323–6328.
[28] F. Basile, P. Chiacchio, G. De Tommasi, Improving on-line fault diagnosis for discrete event systems using time, 2007, pp. 26–32.
[29] Y. Ru, M. Cabasino, A. Giua, C. Hadjicostis, Supervisor synthesis for discrete event systems with arbitrary forbidden state specifications, in: IEEE Conf. on Decision and Control, 2008, pp. 1048–1053.
[30] S. Martini, A. Fagiolini, G. Zichittella, M. Egerstedt, A. Bicchi, Decentralized classification in societies of autonomous and heterogenous robots, in: ICRA. Proceedings of the IEEE International Conference on Robotics and Automation, 2011.
[31] R. Olfati-Saber, J. Fax, R. Murray, Consensus and cooperation in networked multi-agent systems, Proc. IEEE 95 (1) (2007) 215–233.
[32] R. Olfati-Saber, R.N. Murray, Consensus problems in networks of agents with switching topology and time–delays, IEEE Trans. Automat. Control (2004).
[33] J. Cortès, Distributed algorithms for reaching consensus on general functions, Automatica 44 (3) (2008) 726–737.
[34] A. Bicchi, A. Fagiolini, G. Dini, I.M. Savino, Tolerating, malicious monitors in detecting misbehaving robots, in: Safety, Security and Rescue Robotics, 2008. SSRR 2008. IEEE International Workshop on, IEEE, 2008, pp. 109–114.

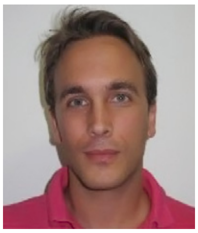


Simone Martini received his Master and Ph.D. Degree in "Automation and Robotics Engineering" in 2008 and 2012, respectively, from the University of Pisa. Since 2012 he has been PostDoc at Research Center "E. Piaggio" of the University of Pisa. His interests are in the field of distributed algorithms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

Please cite this article in press as: S. Martini, et al., 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



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 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 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 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 received his Electronics Engineering Degree from the University of Pisa (IT), in 1990 and his Ph.D. 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 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).

Please cite this article in press as: S. Martini, et al., 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