Tolerating Malicious Monitors in Detecting Misbehaving Robots

Antonio Bicchi and Adriano Fagiolini
Interdepartmental Research Center "E. Piaggio"
Faculty of Engineering, Università di Pisa, Italy
{bicchi, a.fagiolini}@ing.unipi.it

Gianluca Dini and Ida Maria Savino
Dipartimento dell'Informazione
Faculty of Engineering, Università of Pisa, Italy
{gianluca.dini, ida.savino}@ing.unipi.it

Abstract — This paper considers a multi–agent system and focuses on the detection of motion misbehavior. Previous work by the authors proposed a solution, where agents act as local monitors of their neighbors and use locally sensed information as well as data received from other monitors. In this work, we consider possible failure of monitors that may send incorrect information to their neighbors due to spontaneous or even malicious malfunctioning. In this context, we propose a distributed software architecture that is able to tolerate such failures. Effectiveness of the proposed solution is shown through preliminary simulation results.

Keywords: *distributed detection, malicious monitors, monitor failure, robust consensus*

I. Introduction

We consider a set of mobile autonomous robotic agents that communicate over a wireless network and share a common environment to accomplish their tasks. In order to *safely* move within that environment, agents follow a predefined common set of decentralized motion rules. This approach allows an agent to locally decide the next maneuver according to the state of a limited set of neighboring agents in order to improve the overall scalability of the system. In such cooperative multi–agent systems, an agent that violates the motion rules may impair the overall system safety and availability. Therefore detecting *misbehaving* agents is crucial. In a decentralized approach, misbehavior detection has to be locally performed by each agent using its sensing capability. Nevertheless, limitations in sensing translates into limitations in monitoring and thus in detection [1]. In order to overtake this obstacle, agents have to cooperate by exchanging information and achieving consensus on it. In a previous work we have shown that it is possibile to detect a misbehaving agent by exchanging aggregated information with a sufficiently large subset of its neighbors [2]. This solution is effective in the case agents broadcast correct information but fails in the presence of malicious agents that deliberately broadcast false information.

In this paper we cope with the problem of detecting misbehaving agents in the presence of *malicious* agents. A malicious agent may falsely report its own position, invent a non-existing neighbor, omit an existing one, or pretend that it is elsewhere with the aim of impairing safety or causing denial of service.

We propose a distributed collaborative algorithm that allows agents with limited sensing capabilities to achieve consensus upon whether an agent misbehaves or not. Every node broadcasts its "view" of the system in order to enrich the limited sensing capabilities of the neighbors. As malicious agents are present in the system, every node has to *validate* another agent's view. Validation is based on two criteria: it must not be in contrast with the physical dynamics of the system and it must be agreed upon by a majority of agents deemed non malicious by the agents.

Security and safety are crucial challenges in multi–agent system but they have been dealt with by different research communities. The advanced control community has mainly focused on safety issues such as detecting agents that misbehave in terms of movement [1]–[3]. In contrast, the computer and communication community mainly focuses on security by protecting vehicular communications and detecting malicious vehicular traffic information [4]–[6]. As a security infringement may translate into a safety infringement, in this paper we attempt to fill the gap between the two communities and propose an algorithm that detects misbehaving agents in the presence of malicious monitors issuing fake information.

II. Problem Statement

We consider a set $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ of n mobile, autonomous robotic agents that move in a shared environment where they *safely* interact according to a finite set $\mathcal{R}$ of *decentralized cooperation rules*. At any given instant, the cooperation rules allow an agent to *locally* decide upon its next maneuver according to the configurations of its neighboring agents. We call *influence set* of agent a_h at time t, and denote it by $\mathcal{I}_h(t)$, the set of agents that determine the agent a_h's next maneuver at time t. The influence set of a node depends on the rule set $\mathcal{R}$.

An agent that moves according to the rule set $\mathcal{R}$ is *cooperative*, *misbehaving* otherwise. The movements of a misbehaving agent may arbitrarily deviate from that dictated by the rule set $\mathcal{R}$. We call *Safety Set* of agent a_h at time t, and denote it by $\mathcal{S}_h(t)$, the set of agents of which a_h wants to determine whether they are misbehaving or not in order to take adequate safety countermeasures (not discussed in this paper).

Let us suppose that an agent a_h wants to determine whether another agent $a_i \in \mathcal{S}_h(t)$ is misbehaving or not. In order to do that, a_h has to monitor both the configuration of a_i and the

configurations of all agents in a_i influence set. However, due to limited sensing capabilities, the agent a_h is able to determine the configuration of only a limited set of neighboring agents. We call *Visibility Set* of agent a_h at time t, and denote it by $\mathcal{V}_h(t)$, the set of agents of which a_h is able to determine the configuration at time t. By definition, the Visibility Set of an agent always includes the agent itself. We assume that the Visibility Set of an agent contains its Influence Set, i.e., for every agent a_i, $\mathcal{I}_i(t) \subseteq \mathcal{V}_i(t)$. Finally, we assume that visibility is a symmetric relationship. Therefore, if an agent belongs to the visibility set on another agent, then the latter belongs to the visibility set of the former.

In order to cope with the limited visibility, we assume that agents cooperate by periodically exchanging the configurations of agents in their own visibility sets. We call *Communication Set* of agent a_h at time t, and denote it by $\mathcal{C}_h(t)$, the set of agents with which a_h can communicate at time t. By properly dimensioning the communication set with respect to the safety set, it is possible to convey an agent enough information that integrates that locally available and allows the agent to decide upon the behaviors of agents in its safety set. Furthermore, all agents that have a given agent in their respective safety sets can reach consensus on whether that agent is misbehaving or not [2].

This approach works well under the assumption that agents report correct information. Problems arise if there are *malicious* agents that instead report false information. A malicious agent may cheat about both its configuration, e.g., by pretending being elsewhere, and the configuration of its neighbors, e.g., by inventing a non-existing neighbor, omitting to report the presence of a neighbor, or reporting a wrong configuration. A malicious node deliberately acts against the system in order to impair the system safety or cause denial of service.

In this paper we consider a system where the agents are loosely synchronized and where communication is both reliable and authenticated. Reliable communication means that each message broadcast by the agent a_h is received by all other agents belonging to the a_h Communication Set. Authenticated communication means that each received message can be attributed to its legitimate originator. Finally, we assume that a majority of agents around a given agent is not malicious. This means that the number of these agents is larger than other potentially malicious ones. This assumption is necessary to make any forward progress.

III. Agent Architecture

The architecture of an agent is depicted in Fig. 1. The *Sensing* module is composed of a set of sensors that allow each agent to measure the configuration state of all its neighbors. Then, collected configuration data are periodically broadcast through the *Communication* module. Such a module allows each agent to share information with other agents so as to improve the sensing capability. Upon receiving a message, the *Local Monitor* immediately evaluates the validity of the received configurations and attempts to detect misbehaving agents on the basis of validated ones. More in detail, upon

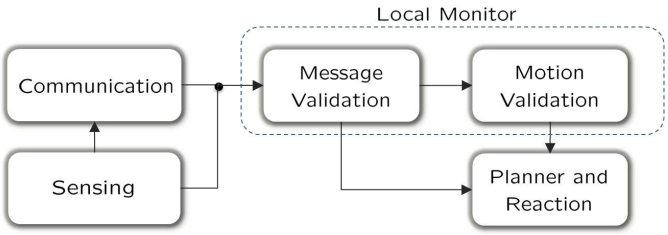

Fig. 1. The agent architecture in terms of its constituting modules and its relationships.

receiving a message the *Message Validation* module verifies the correctness of data in the incoming message to filter out invalid one, and it passes valid data to the *Motion Validation* and *Planner* module. The *Motion Validation* is responsible for detecting malfunctioning in the motion of neighboring agents, whereas the Planner module decides on the agent motion on the basis of its securely estimated neighborhood information. In case a non–cooperative agent is detected, the Motion Validation module sends an *alarm* to the Planner so that the agent performs escape maneuvers that may require e.g. to divert from the planned route. In the remainder of this section, we detail the Message Validation, the Motion Validation, and the Planner modules.

A. Message Validation Module

In order to guarantee the system safety, an agent a_h has to monitor the behavior of each agent a_i belonging to $\mathcal{S}_h(t)$ so as to be able to run adequate countermeasures in case of non–cooperative neighbors. We assume that a suitable set of emergency maneuvers are specified in the set $\mathcal{R}$ itself allowing each agent a_h to escape from a non–cooperative other agent in its safety set. To this aim, we define $\mathcal{S}_h(t)$ as the set of agents laying at time t in the circle centered at a_h position with radius ρ_S. Thus, given v_M the greatest of the agents velocities and ΔT_r the maximum time that agent a_h needs to react to the presence of a non–cooperative agent in its safety set, it is necessary that the following inequality holds $\rho_S \geq v_M \Delta T_r$.

Hence, a_h has to receive the a_i state and the state of all agents influencing a_i instantaneous motion. Let us define d_I the maximum distance between an agent and all the agents belonging to its Influence set. It is worthwhile to notice that the distance d_I strictly depends on the rule set $\mathcal{R}$. Hence, given ρ_C the radio communication range it follows that the following condition must hold:

$$\rho_C \geq \rho_S + d_I. \tag{1}$$

Each agent periodically broadcasts a message containing the state of all agents belonging to its Visibility set. Given $\tilde{D}_{max}$ the maximum error of an agent position that the rule set $\mathcal{R}$ can tolerate without compromising the system safety, each agent has to broadcast a message every ΔT_c seconds so that $\Delta T_c \leq \tilde{D}_{max}/v_M$.

Let us consider an agent a_j belonging to the Communication set of a_h so that a_h receives the message $M_j(t)$. Since

the communication time is negligible with respect to the system time constant, the assertions received by agent a_h are considered at the time of message arrival. In case a_i belongs to a_j Visibility set, $M_j(t)$ contains $\langle q_i(t)\rangle_j$, that is the configuration state of a_i sensed by a_j. Hence, the agent a_h has to verify the validity of the assertion $\langle q_i(t)\rangle_j$ contained in the message. In case the agent a_i belongs to $\mathcal{V}_h(t)$, a_h discards the assertion $\langle q_i(t)\rangle_j$ contained in $M_j(t)$ by assuming valid its own observations. Otherwise, a_h has to verify whether $\langle q_i(t)\rangle_j$ satisfies the following conditions: 1) $\langle q_i(t)\rangle_j$ is consistent with the physical dynamics of the system (*physical condition*), and 2) $\langle q_i(t)\rangle_j$ has been confirmed by a majority of non-malicious agents (*logical condition*). In case both the conditions are satisfied, it follows that $\langle q_i(t)\rangle_h = \langle q_i(t)\rangle_j$, where $\langle q_i(t)\rangle_h$ the state of agent a_j according to agent a_h at time t.

With reference to the physical condition, let us consider $\langle q_i(t^\star)\rangle_h$ the last configuration of a_i validated by agent a_h at time $t^\star$ so that $t^\star \leq t$. Hence, the assertion $\langle q_i(t)\rangle_j$ has to satisfy the following equation:

$$\langle q_i(t)\rangle_j \in f_P(\langle q_i(t^\star)\rangle_h, (t - t^\star)) \tag{2}$$

where the function $f_P(q, \Delta\tau))$ returns the set of feasible configurations starting from q during the interval $\Delta\tau$ and according to the rule set $\mathcal{R}$.

With reference to the logical condition, let us consider the set $\mathcal{P}_h(a_i, t)$ containing all the assertions about a_i configuration state received by a_h in the interval $[t - \Delta T_c, t]$. It is worthwhile to notice that such set contains only assertions that have satisfied the physical condition.

Let m be the number of malicious agents the system must be able to tolerate. As these nodes might collude and report false information about a given configuration $q_i(t)$, then we assume that the Visibility Set of a any given node must contain at least $(m+1)$ correct agents. This assumption, together with the condition expressed in Equation 1, guarantees that $\langle q_i(t)\rangle_j$ satisfies the logical property if and only if the following condition holds:

$$f_L(\langle q_i(t)\rangle_j, \mathcal{P}_h(a_i, t)) \geq (m + 1) \tag{3}$$

where the function $f_L(q, P)$ returns the number of configuration contained in P that are compatible with q according to the rule set $\mathcal{R}$. Notice that the value of m features a security-performance trade-off. Actually, a higher value of m implies that the system tolerant to an higher number of malicious colluding agents but it requires a denser network in order to guarantee a reliable validation.

B. Planner Module

According to the set of rules $\mathcal{R}$, agents can perform at any instant t one of κ *maneuvers*, $\Sigma = \{\sigma^1, \sigma^2, \ldots, \sigma^\kappa\}$, and must change from a current maneuever to another one whenever one of a set of ν *events*, $E = \{e^1, e^2, \ldots, e^\nu\}$, depending on its influence set $\mathcal{I}_i(t)$ occurs. For the sake of clarity, consider as an example the case of n cars moving on a multi–laned highway. Such cars are supposed to have the same dynamics, and automated pilots are supposed to decide the current maneuver based on its goal, the configurations of the car and of other neighboring cars. In this example, the actions defined by $\mathcal{R}$ are accelerate, decelerate, and change to the next left or right lane. As an example, the event requiring a maneuver change that will allow a vehicle overtaking is represented by a slower car in the front and a free lane on the left. Therefore, the planner is a module that decides the trajectory of agent a_i based on the cooperation rule set $\mathcal{R}$, and valid configurations of agents in its influence set $\mathcal{I}_i(t)$ at the current time t.

This type of system, where agents have a physical dynamics, but interact according to *event–based* cooperation rule sets $\mathcal{R}$, can be modeled as hybrid systems [1]. For the reader convenience we recall this result in the following. Let $q_i(t) \in \mathcal{Q}$ be a vector describing the configuration of agent a_i at time t and $\mathcal{Q}$ be the corresponding configuration space. It should be clear that in this description $q_i(t)$ is a short–hand for the valid configuration of agent a_i computed on board of agent a_i itself, i.e. $q_i(t) = \langle q_i(t)\rangle_i$. Furthermore, let us denote with $\sigma_i(t) \in \Sigma$ the maneuver that agent a_i is currently executing.

We showed that $q_i(t)$ has a continuous–time dynamics

$$\dot{q}_i(t) = f(q_i(t), u_i(t)),$$

where $u_i(t) \in \mathcal{U}$ is a control input. Moreover, $u_i(t)$ is a feedback law generated by a low–level controller $g : \mathcal{Q} \times \Sigma \rightarrow \mathcal{U}$ depending on the current maneuver $\sigma_i(t)$, i.e.

$$u_i(t) = g(q_i(t), \sigma_i(t)).$$

On the contrary, $\sigma_i(t)$ has a discrete–time dynamics $\delta : \Sigma \times E \rightarrow \Sigma$ and indeed it changes value only at discrete times t_k when an event from E occurs. More precisely, we have that

$$\sigma_i(t_{k+1}) = \delta(\sigma_i(t_k), e(t_k)),$$

where $e(t_k)$ is the event occured at time t_k and requiring a maneuver change from $\sigma_i(t_k)$ to $\sigma_i(t_{k+1})$. Furthermore, event activation is detected by a static map $\mathcal{D} : \mathcal{Q} \times \mathcal{Q}^p \times Z \rightarrow E$, where p is the maximum agent number in the influence set $\mathcal{I}_i(t)$. With reference to the example mentioned above, map $\mathcal{D}$ encodes conditions such as the presence of a slower car in the front, and a free lane on the left. The event detected at time t_k is

$$e(t_k) = \mathcal{D}(q_i(t_k), N_i(t_k), \zeta_i(t_k)),$$

where $N_i(t) = (q_{i_1}(t), \ldots, q_{i_p}(t))$ is a vector impiling all valid configurations in the influence set $\mathcal{I}_i(t)$, and $\zeta_i(t) \in Z$ is a parameter that is reset at any maneuver transition. Note that $q_{i_j}(t)$ is a short–hand for $\langle q_{i_j}(t)\rangle_i$. As a whole, the dynamics of agent a_i can be described by the following hybrid model:

$$\dot{q}_i(t) = \mathcal{H}(q_i(t), q_{i_1}(t), \ldots, q_{i_p}(t)),$$

where $\mathcal{H} : \mathcal{Q} \times \mathcal{Q}^p \rightarrow \mathcal{Q}$, and $i_1, \ldots, i_p$ are the indices of agents in $\mathcal{I}_i(t)$. Under this view, we will consider $q_{i_1}(t), \ldots, q_{i_p}(t)$ as inputs of model $\mathcal{H}$ and $q_i(t)$ as its output.

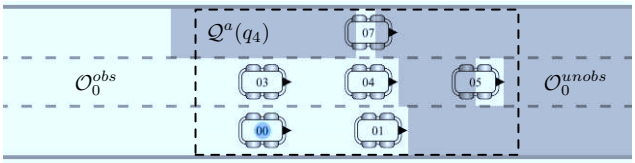

Fig. 2. Partition of the influence set $\mathcal{I}_0(t)$ of agent a_0 w.r.t. agent a_4's extended visibility $\mathcal{V}_4^*$.

C. Motion Validation Module

We consider here a generic agent a_h that is monitoring the motion of one of the agents, a_i, that are in its safety set $\mathcal{S}_h(t)$. To this aim, suppose that a measure $\langle q_i(t)\rangle_h$ of the trajectory of a_i during successive limited time intervals, $T_k = [t_k, t_{k+1}]$, for $k = 0, 1, \ldots$, is available for the observing agent a_h.

Let us denote with $\tilde{q}_i(t)$ the expected evolution of agent a_i that is the output of the hybrid model $\mathcal{H}$ receiving as inputs the configurations $\langle q_{i_1}(t)\rangle_h, \ldots, \langle q_{i_p}(t)\rangle_h$ of all agents that are in the influence set $\mathcal{I}_i(t)$ of agent a_i itself. Then, agent a_i's motion is non–cooperative if the expected motion differs from the validated one, i.e.

$$\langle q_i(t)\rangle_h \neq \tilde{q}_i(t) \;\text{ for some }\; t \in T_k\,.$$

The fact that makes this monitoring task difficult for the observing agent a_h is its partial knowledge of the influence set $\mathcal{I}_i(t)$ of agent a_i. In the example in study, some cars affecting the behavior of agent a_i may be out agent a_h's visibility set $\mathcal{V}_h$ (since they remain hidden by other cars as in Fig. 2), and no valid messages specifying this information have been received yet by other monitoring agents.

We can conveniently define an *extended visibility* set $\mathcal{V}_h^*$ as the region over which agent a_h has received valid information by either its own sensing module or by other agents in $\mathcal{C}_h(t)$. Then, the influence set $\mathcal{I}_i(t)$ of agent a_i can be partitioned w.r.t. a_h into a known region $\mathcal{I}_i^h(t)$ and an unknown one $\mathcal{I}_i^{\bar{h}}(t)$, i.e.

$$\mathcal{I}_i(t) = \mathcal{I}_i^h(t) \cup \mathcal{I}_i^{\bar{h}}(t)\,. \tag{4}$$

Then, we are interested in solving the following:

Problem 1: Given agent a_i's (hybrid) motion model $\mathcal{H}$, the partition of its influence set $\mathcal{I}_i(t)$ in Eq. 4 w.r.t. agent a_h's extended visibility $\mathcal{V}_h^*$, and n_o configurations $\langle q_{i_1}(t)\rangle_h, \ldots, \langle q_{i_{n_o}}(t)\rangle_h \in \mathcal{I}_i^h(t)$ of known neighbors of agent a_i, determine, if it exists, a choice of $p - n_o$ configurations $\hat{q}_{i_{n_o+1}}(t), \ldots, \hat{q}_{i_p}(t) \in \mathcal{I}_i^{\bar{h}}(t)$ such that the expected motion

$$\begin{aligned}\tilde{q}_i(t) \;=\; & \langle q_i(t_k)\rangle_h + \textstyle\int_{t_k}^{t} \mathcal{H}(\langle q_i(\tau)\rangle_h, \langle q_{i_1}(\tau)\rangle_h, \ldots, \\ & \langle q_{i_{n_o}}(\tau)\rangle_h, \hat{q}_{i_{n_o+1}}(\tau), \ldots, \hat{q}_{i_p}(\tau))\, d\tau\,,\end{aligned}$$

equals the measure one, i.e. $\tilde{q}_i(t) = \langle q_i(t)\rangle_h$ for all $t \in T_k$.

Solving this problem is in general a hard task due to the nonlinear and differential nature of the motion model $\mathcal{H}$. It basically requires that an *unknown input observer* (UIO) $\mathcal{H}^\dagger$ of the hybrid model is built. Furthermore, a direct approach for the computation of such a UIO leads to find ad–hoc solutions for specific cases. However, we showed in [1] how this can be avoided for the considered class of robotic multi–agent systems. The property that in our opinion makes our approach appealing is that all components of the proposed decentralized motion validation module can be *automatically* generated once the agent dynamics f, and the cooperation rules $\mathcal{R}$ are given. The reader may refer to our works [1], [2], [7] for a complete description of the method and can assume the existence of a procedure to build a UIO, $\mathcal{H}^\dagger$, such that

$$(\hat{q}_{i_{n_o+1}}(t), ..., \hat{q}_{i_p}(t)) = \mathcal{H}^\dagger(\langle q_i(t)\rangle_h, \langle q_{i_1}(t)\rangle_h, .., \langle q_{i_{n_o}}(t)\rangle_h) \tag{5}$$

where $\hat{q}_{i_m}(t)$ for $m = n_o + 1, \ldots, p$ are continuous sets estimating configurations of agents in $\mathcal{I}_i^{\bar{h}}(t)$ that can explain the validated motion $\langle q_i(t)\rangle_h$ of agent a_i (see Fig. 6 in the example).

By using this mechanism, agent a_h may decide on the cooperativeness $\langle b_h\rangle_i$ of agent a_i according to the following rules. If a_h has complete knowledge of agent a_i's influence set, it will be able to say it is cooperative or not. Otherwise, by observing agent a_i's motion, agent a_h is able to estimate the presence or the absence of other agents in the unknown part of a_i's influence set. As long as a choice for $\hat{q}_l$ exists, agent a_i can be considered as possibly cooperative or $\mathrm{uncertain}$ (as a matter of fact, a_i can not verify the correctness of these estimates). If no values for these estimates exist, agent a_i is considered as $\mathrm{noncooperative}$. In brief, the cooperativeness $\langle b_h\rangle_i$ of agent a_i according to agent a_h is a discrete variable taking values in the set $\mathcal{B} = \{\mathrm{cooperative}, \mathrm{noncooperative}, \mathrm{uncertain}, \mathrm{unknown}\}$. The introduction of the value "$\mathrm{unknown}$" is instrumental for the purpose of communication. Indeed, if agent a_h does not see agent a_i, but has to participate in an agreement on the value of its cooperativeness, it will initially exchange the value $\mathrm{unknown}$.

IV. Example

A. An Automated Highway

Consider n mobile agents that are traveling along a highway with different maximum speed and different final positions. Agents are supposed to cooperate according to the common driving rules in order to avoid collisions. Informally, the rule set $\mathcal{R}$ is the following:

- $R1$) proceed at the maximum speed along the rightmost free lane when possible (fast maneuver);
- $R2$) if a slower vehicle proceeds in front on the same lane, then overtake the vehicle if the next lane on the left is free (left maneuver), or reduce the speed (slow maneuver) otherwise;
- $R3$) as soon as the next lane on the right becomes free, change to that lane (right maneuver);
- $R4$) overtaking any vehicle on the right is forbidden.

The generic agent chooses one of these maneuvers based on events on its neighborhood. With reference to Fig. 3, agent

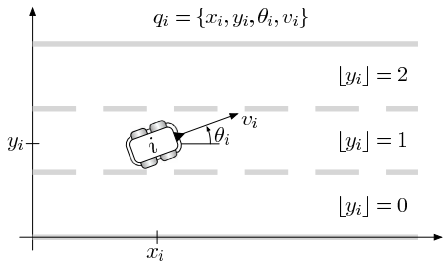

Fig. 3. A 2–lane automated highway with a set of common individual driving rules.

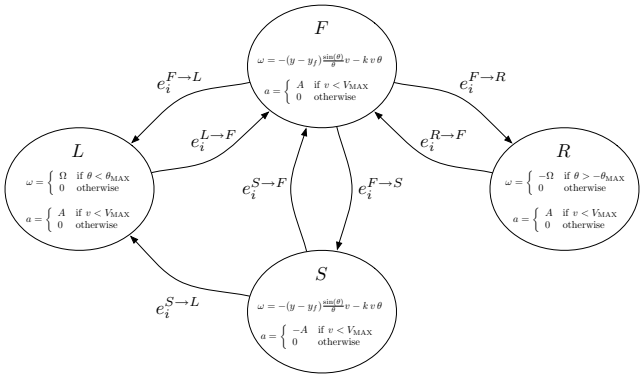

Fig. 4. Discrete dynamics δ of the automaton, and low–level feedback control g ensuring that the plant f behaves according to the rule set $\mathcal{R}$.

a_i's configuration is $q_i(t) = (x_i(t), y_i(t), \theta_i(t), v_i(t))$ and has the continuous–time unicycle–like dynamics f:

$$\begin{cases} \dot{x}_i(t) = v_i(t)\,\cos(\theta_i(t))\,, \\ \dot{y}_i(t) = v_i(t)\,\sin(\theta_i(t))\,, \\ \dot{\theta}_i(t) = \omega_i(t)\,, \\ \dot{v}_i(t) = a_i(t)\,, \end{cases}$$

where $a_i(t)$ and $\omega_i(t)$ are linear acceleration and angular velocities, respectively. According to the set $\mathcal{R}$, the maneuver $\sigma_i(t)$ of the i–th robot may take value on the set $\Sigma = \{\text{fast}, \text{left}, \text{right}, \text{slow}\}$ and has the discrete dynamics δ of the automaton in Fig. 4, where the low–level feedback controller g ensures that the current maneuver σ_i is performed. The generic event $e_i^{\sigma^1\to\sigma^2}$ of Fig. 4 is described in [2].

B. Tolerating malicious agents

As stated above, the Message Validation module is responsible for validating all assertions that an agent receives. Let us consider the agent a_h that has to validate $\langle q_i(t)\rangle_j$ contained in $M_j(t)$. According to the considered cooperation rule set $\mathcal{R}$, the physical and logical conditions can be specified as follows.

As to the physical condition, let us consider whether a_h has recently validated the configuration of a_i or not. In the first case, let $\langle q_i(t^\star)\rangle_h$ be the last configuration of a_i validated by agent a_h at time $t^\star$ so that $t^\star \le t$. Thus, function f_P (Eq. 2) defines the sector centered at $(x_i(t^\star), y_i(t^\star))$ with radius $v_i(t^\star)(t - t^\star)$ and angle $\theta_i(t^\star)$. In the second case, we assume that a_i is a new member of the influence set of an agent laying on the border of the radio communication range. Hence, function f_P defines the ring centered at $(x_h(t), y_h(t))$ with radii ρ_C and $\rho_C + d_I$.

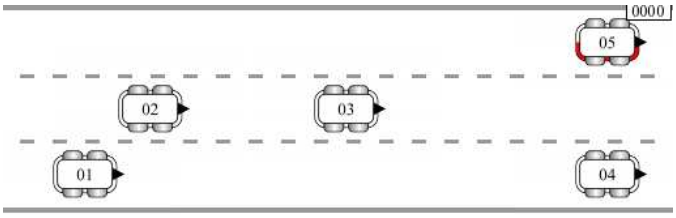

Fig. 5. Simulation run where agent a_3 is non–cooperative since it keeps traveling on the second lane, even though the lane on the right is free.

Agent	Visibility set
a_1	$\mathcal{V}_1(t) = \{a_1, a_2, a_3, a_4\}$
a_2	$\mathcal{V}_2(t) = \{a_1, a_2, a_3\}$
a_3	$\mathcal{V}_3(t) = \{a_2, a_3, a_4, a_6\}$
a_4	$\mathcal{V}_4(t) = \{a_1, a_3, a_4, a_5\}$
a_5	$\mathcal{V}_5(t) = \{a_3, a_4, a_5\}$

TABLE I
VISIBILITY SETS

As to the logical condition, function f_L (Eq. 3) returns the number of configurations $\langle q_i(t')\rangle_k$ belonging to $\mathcal{P}_h(a_i, t)$ so that $||\langle q_i(t')\rangle_k, \langle q_i(t)\rangle_j|| \le (t - t')v_M$, where $||\cdot||$ is the Euclidean distance. This condition takes into account configurations contained in the messages are sensed at different times.

With reference to the automated highway, consider the case depicted in Fig. 5, where agent a_3 is trying to damage the system by violating rule R3 and moving on the second lane whereas the lane on its right is free. Fig. 6 shows the information on the influence set of agent a_3 that every agent is able to estimate using only local sensing as in Eq. 5. Such estimates are possibly non–convex regions where the presence of agents is required (when reported in red in the figure) or is excluded (when reported in green). The figure also reveals that none of them is individually able to detect the misbehavior. Then, agent a_3 will send a false assertion that tends to justify its non–cooperative behavior. In particular, agent a_3 is trying to leverage on the partial visibility of the following agent a_2, and indeed it claims the existence of an another agent a_6 in front of it. Clearly, on the basis of agent a_3's message, agent a_2 will determine that agent a_3's behavior is cooperative. This example shows the necessity of a mechanism that passes valid messages and discards the others. Table I reports the visibility sets of every agent and in particular shows the malicious data produced by agent a_3.

To overcome local sensing limitation, all agents share their estimated information. Let us focus on how agent a_2 updates its local view based on received messages. Table II shows the message validation mechanism performed by agent a_2 whenever it receives a message. In the example, given the low density of the network, we consider $m = 1$. The first and second column shows message reception order at a_2, the third column specifies the set of validated agent configurations,

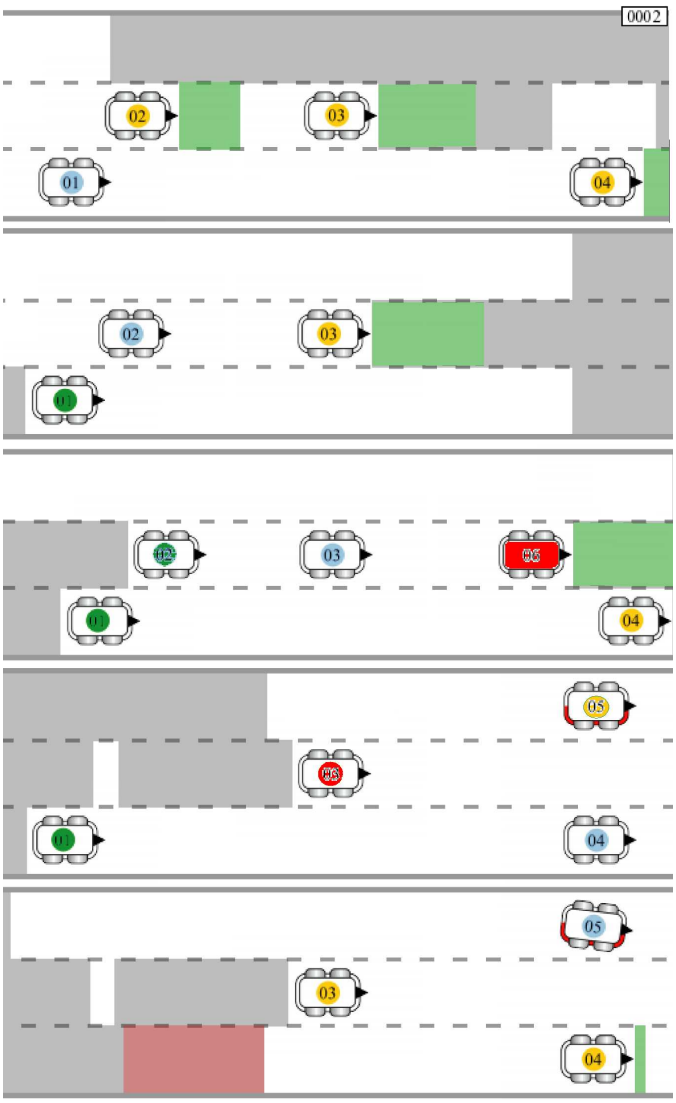

Fig. 6. Initial results of the motion validation module at agents a_1, a_2, a_4, and a_5 that are acting as monitors of their neighbor a_3 and are trying to explain the its motion behavior. The five pictures reports the monitors' views at different times, where red and green colors indicate regions where the presence of an agent is required or is excluded, respectively. Agent a_3 sends an assertion containing a false agent a_6 that tends to justify its non–cooperative behavior.

and the fourth column agents whose configuration has to be validated. At the step 4, a_2 validates a_5's position, whereas at step 5 it does not validate the position of a_6. As shown in the table, agent a_6 is correctly not handed to the Planner module. The same procedure is run at every agent and reveals that the proposed mechanism is able to tolerate the malicious agent a_3.

V. Conclusion

The problem of detecting misbehavior in multi–agent systems is considered. A solution where agents act as local monitors of their neighbors and use locally sensed information as well as data received from other monitors is presented. The proposed solution is robust to possible failure of some monitors sending incorrect information. Future work will consider the development of emergency strategies by using which such robotic systems can react to the presence of malicious agents.

Step	Message	Valid agents	Uncertain agents
1	M_2	$\{a_1, a_2, a_3\}$	$\{\}$
2	M_1	$\{a_1, a_2, a_3\}$	$\{a_4\}$
3	M_4	$\{a_1, a_2, a_3, a_4\}$	$\{a_5\}$
4	M_3	$\{a_1, a_2, a_3, a_4\}$	$\{a_5, a_6\}$
5	M_5	$\{a_1, a_2, a_3, a_4, a_5\}$	$\{a_6\}$

TABLE II

Run of the message validation mechanism at agent a_2.

VI. Acknowledgment

Authors wish to thank undergraduate student Francesco Babboni for his useful help. This work has been partially supported by the European Commission under FP7 with contract IST 224428 (2008) "CHAT - Control of Heterogeneous Automation Systems: Technologies for scalability, reconfigurability and security", by CONET, the "Cooperating Objects Network of Excellence", funded by the European Commission under FP7 with contract number FP7-2007-2-224053, by HYCON, the Network of Excellence on "Hybrid Control: Taming Heterogeneity and Complexity of Networked Embedded Systems", funded by the European Commission under FP6 with contract number IST-2004-511368, and by Research Project 2007 funded by Cassa di Risparmio di Livorno, Lucca e Pisa.

References

[1] A. Fagiolini, G. Valenti, L. Pallottino, G. Dini, and A. Bicchi, "Decentralized Intrusion Detection For Secure Cooperative Multi–Agent Systems," *IEEE International Conference on Decision and Control*, 2007.

[2] A. Fagiolini, M. Pellinacci, G. Valenti, G. Dini, and A. Bicchi, "Consensus-based distributed intrusion detection for multi-robot systems," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008)*, Pasadena, CA, USA, 19–23 May 2008, pp. 120–127.

[3] F. Pasqualetti, A. Bicchi, and F. Bullo, "Distributed intrusion detection for secure consensus computations," in *proceedings of the 46th IEEE International Conference on Decision and Control*, New Orleans, LA, USA, 12–14 December 2007, pp. 5594–5599.

[4] P. Golle, D. Greene, and J. Staddon, "Detecting and correcting malicious data in vanets," in *Proceedings of the first ACM workshop on Vehicular Ad-Hoc Networks*, Philadelfia, Pennsylvania, 1 October 2004, pp. 29–37.

[5] J.-P. Hubaux, S. Capkun, and jun Luo, "The security and privacy of smart vehicles," *IEEE Security and Privacy Magazine*, vol. 2, no. 3, pp. 49–55, May-June 2004.

[6] M. Raya, P. Papadimitratos, and J.-P. Hubaux, "Securing vehicular communications," *IEEE Wireless Communications*, vol. 13, no. 5, pp. 8–15, October 2006.

[7] A. Fagiolini, G. Valenti, L. Pallottino, G. Dini, and A. Bicchi, "Local Monitor Implementation for Decentralized Intrusion Detection in Secure Multi–Agent Systems," *IEEE Conference on Automation, Science, and Engineering*, 2007.