Modelling Implicit Communication in Multi-Agent Systems with Hybrid Input/Output Automata.*

Marta Capiluppi Roberto Segala
Università di Verona
Dipartimento di Informatica
Verona, Italy
marta.capiluppi@univr.it, roberto.segala@univr.it

We propose an extension of Hybrid I/O Automata (HIOAs) to model agent systems and their implicit communication through perturbation of the environment, like localization of objects or radio signals diffusion and detection. To this end we decided to specialize some variables of the HIOAs whose values are functions both of time and space. We call them *world variables*. Basically they are treated similarly to the other variables of HIOAs, but they have the function of representing the interaction of each automaton with the surrounding environment, hence they can be output, input or internal variables. Since these special variables have the role of simulating implicit communication, their dynamics are specified both in time and space, because they model the perturbations induced by the agent to the environment, and the perturbations of the environment as perceived by the agent. Parallel composition of world variables is slightly different from parallel composition of the other variables, since their signals are summed. The theory is illustrated through a simple example of agents systems.

1 Introduction

Many modern complex systems represent agents interacting to achieve a common goal, but reacting in an independent way to external stimuli, following an autonomous decision policy and coordinating using communication. When and where communication fails, the agents need to *feel* the environment reacting to its stimuli. This is the case, for example, of agents performing a *search* mission, such as UAVs [8] or autonomous underwater vehicles [5], but also of road traffic problems [14, 15] and autonomous straddle carriers in harbours [11]. These multi-agents problems have been case studies of the European Project CON4COORD (EU FP7 223844) and have motivated the modeling formalism presented in this paper. Indeed what is common to each case study is the presence of a collection of agents that communicate and coordinate to achieve a common goal. Moreover the agents move within an environment that changes dynamically and detect each other's presence not necessarily via direct communication but rather by observations of the environmental changes.

We focus on automata-based representations of hybrid systems [2, 1], adding features to a model, in order to keep as much as possible of the underlying theory. Since the motivating case studies need to satisfy some compositionality properties, we choose to start from Hybrid I/O Automata (HIOAs) of [10], for which strong results on compositionality exist. We add features to represent faithfully situations where a hybrid automaton exists within an environment and derives information about other automata by observing the environment itself, rather than by using any form of direct communication. We will call the exchange of information through observation implicit communication. Indeed groups of agents usually need to know the environment where they live and move to collect and elaborate data and react

*This research has been supported by EC-Project C4C (*Control for Coordination of Distributed Systems*) funded by the European Commission in the 7th EC framework program (Challenge ICT-2007.3.7).

M. Faella, A. Murano (Eds.): Games, Automata, Logics and Formal Verification (GandALF 2012)
EPTCS 96, 2012, pp. 1–14, doi:10.4204/EPTCS.96.1

in a coordinated way. To do this, they need to exchange stimuli with the surrounding environment, by observation and using sensors. Usually the communication between agents acting in a certain area is achieved using artificial machineries, such as supervisors or broadcasting signals. Our aim is to avoid any kind of artificial machinery to model communication between agents and their interaction with the environment, using a more *natural* method, based on the human perception, i.e. through observation of the changes in the surroundings of each agent. Moreover direct communication is not always possible, since signals are subject to noise and environmental hostilities, or sometimes it is not the best policy, because sending signals means being intercepted, not considering faults and failures in senders and receivers. Autonomy in making decisions and exchanging information based on the sensing of the reality can be used as a redundant and a faster way of communication. To achieve implicit communication, we extend the HIOAs by specializing some variables, called *world variables*. They take values both in space and time, i.e. their values are indeed functions of time and space, as occurs in diffusion equations and they represent all the information exchanged between the environment and the agents. World variables represent maps of the changes in the environment as perceived by the agents: at each point in the space and each instant of time their values show the situation that can be sensed by agents standing in that area. To this end, world variables are partitioned into input and output variables: input world variables represent the observations made by the agents sensing the environment, output world variables represent stimuli given by the agents to the environment. To keep the theory consistent with HIOAs, all the results on semantics are preserved. Moreover we introduce parallel composition using rules similar to the ones for HIOAs, i.e., automata are synchronized on common actions and shared variables, except for output world variables, whose stimuli are summed because their effect on the environment is common.

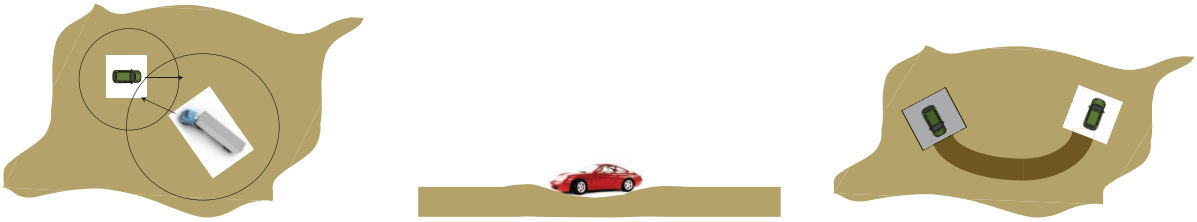
At the best of our knowledge, there are only a couple of approaches to the presented problem. One has been introduced in [6] where dynamic networks of hybrid automata are studied. The introduced programming language focuses on dynamical interfaces. Another method has been presented in [13] where a compositional interchange format (CIF) defined in terms of an interchange automaton is used as a common language to describe objects from the different models for hybrid systems existing in literature. None of these two languages is based on the idea of implicit communication coded by world variables. Our approach is a starting point to solve the problem of dynamical interfaces in a simpler way than the ones proposed. Nevertheless at the current status of our work the presented approach does not solve this problem, even though we started from it. We choose to extend HIOAs because of the underlying compositionality theory and because of the input/output distinction of the variables, which we keep in our description. Many other representations of hybrid automata could be used as basis and extended similarly looking at the main theoretical results they have been introduced for. As an example the cited hybrid automata in [1] are more focused in reachability issues, but have been studied also for decidability in [7]. As stated in [10] Hybrid Automata (HA) presented in [1] are similar to HIOAs in their combined treatment of discrete and continuous activity, but their theory does not address system decomposition issues such as external behavior, implementation relationships and composition. These issues have been addressed in [3] by using hybrid reactive modules, but they still differ from the way they are faced by HIOAs, because reactive modules still communicate via shared variables, not via shared actions. Summarizing, the choice of the HIOA model has been motivated by the fact that their application is more suited for the kind of agent systems and scenarios under study. Indeed the communication via explicit actions, similarly to discrete event automata, is used to model signals communication, while the possibility to trace an external behavior catches the interaction with the environment, which is the basic aim of extending the original formalism with world variables.

The paper is organized as follows: in Section 2 we introduce the modeling framework; in Section 3 we show and recall the main results on semantics of the proposed model; in Section 4 parallel compo-

sition of the presented automata is described, showing the main results on composability. The theory is illustrated throughout the paper with a simple example, the interested reader can find a more complex and realistic application in [11]. All the results presented in the paper use the notation of [10].

2 HIOAs with world variables

Example 1. *Consider a sandy area where two cars move, as in Fig. 1(a). We subsume an underline metric space* $\mathbb{R}^2$*. When a car takes a certain position, its pressure provokes a depression of the ground (fig. 1(b)). Hence the car changes the characteristics of the environment in a permanent way, since sand retains the shape. The other car is, then, able to* see *where a car has moved (fig. 1(c)). We aim at avoiding*



(a) A sandy area with cars. (b) Level of the ground. (c) A car moving from a point to another in the sandy area.

Figure 1: Characteristics of the scenario

collisions between the two cars. This might be done by equipping each car with some tools to send signals to the other vehicle when approaching, or adding to the system a supervisor knowing at each instant of time the position of both cars. We will call this kind of communication explicit. *Another solution would be to think each car as an* intelligent *agent that* senses *the surrounding environment and is able to* understand *if the other car is too near. We call this kind of communication* implicit. *In other words each vehicle should use its sensors to catch the changes in its neighborhood and to calculate the possibility of another car to be in collision risk. The implicit communication is more* natural *to us, it does not need artificial machinery, it can be used even in case of hostile environments, where explicit communication is difficult or even impossible, but also when there is need to communicate without sending data through a network. Moreover implicit communication can be used as a redundant mean of communication, when the tools involved in explicit communication fail.*

The scenario described in Example 1 is a typical problem of coordination of agents, even though simplified to enlighten only the main challenges the designer has to face in finding a suitable model for this situation. As stated in the Introduction, we decided to use the well known framework of Hybrid I/O Automata (HIOAs) of [10] to keep the underlying composability theory, very useful in multi-agent problems.

Definition 1. Hybrid I/O Automaton (HIOA) [10]
A HIOA $\mathcal{A}$ *is a tuple* $((U,X,Y),(I,H,O),Q,\Theta,D,\mathcal{T})$ *where*

- (U,X,Y) *are disjoint sets of* input, internal, and output variables*, respectively. Let* V *denote the set* $U \cup X \cup Y$ *of* variables.
- (I,H,O) *are disjoint sets of* input, hidden, and output actions*, respectively. Let* A *denote the set* $I \cup H \cup O$ *of* actions.

- $Q \subseteq vals(X)$ *is the set of* states.
- $\Theta \subseteq Q$ *is a nonempty set of* initial states.
- $D \subseteq vals(X) \times A \times vals(X)$ *is the* discrete transition relation.
- $\mathcal{T}$ *is a set of trajectories on* V *that satisfy the following axioms*
  - **T1** (Prefix closure) *For every* $\tau \in \mathcal{T}$ *and every* $\tau' \leq \tau$, $\tau' \in \mathcal{T}$.
  - **T2** (Suffix closure) *For every* $\tau \in \mathcal{T}$ *and every* $t \in dom(\tau)$, $\tau \trianglerighteq t \in \mathcal{T}$.
  - **T3** (Concatenation closure) *Let* $\tau_0, \tau_1, \tau_2, \ldots$ *be a sequence of trajectories in* $\mathcal{T}$ *such that, for each nonfinal index* i, τ_i *is closed and* $\tau_i.lstate = \tau_{i+1}.fstate$. *Then* $\tau_0 \frown \tau_1 \frown \tau_2 \cdots \in \mathcal{T}$.

Notation: For each variable v, we assume both a *(static) type*, $type(v)$, which gives the set of values it may take on, and a *dynamic type*, $dtype(v)$, which gives the set of trajectories it may follow. A *valuation* $\mathbf{v}$ for a set of variables V is a function that associates with each variable $v \in V$ a value in $type(v)$. We write $vals(V)$ for the set of valuations for V. Let J be a left-closed interval of T (the time axis) with left endpoint equal to 0. Then a *J-trajectory* for V is a function $\tau : J \to vals(V)$, such that for each $v \in V$, $\tau \downarrow v \in dtype(v)$. A *trajectory* for V is a J-trajectory for V, for any J. Trajectory τ is a *prefix* of trajectory τ', denoted by $\tau \leq \tau'$, if τ can be obtained by restricting τ' to a subset of its domain. We define $\tau \trianglerighteq t \triangleq (\tau \lceil [t, \infty)) - t$. The concatenation $\frown$ of two trajectories is obtained by taking the union of the first trajectory and the function obtained by shifting the domain of the second trajectory until the start time agrees with the limit time of the first trajectory; the last valuation of the first trajectory, which may not be the same as the first valuation of the second trajectory, is the one that appears in the concatenation. Prefix, suffix and concatenation operations return trajectories. We define $\tau.fval$, the *first valuation* of τ, to be $\tau(0)$, and if τ is closed (J is a closed interval), we define $\tau.lval$, the *last valuation* of τ, to be $\tau(\tau.ltime)$. Given a trajectory $\tau \in \mathcal{T}$ we denote $\tau.fval \lceil X$ by $\tau.fstate$ and, if τ is closed, we denote $\tau.lval \lceil X$ by $\tau.lstate$. We write $f \lceil P$ for the restriction of function f to set P, that is, the function g with $dom(g) = dom(f) \cap P$ such that $g(c) = f(c)$ for each $c \in dom(g)$. If f is a function whose range is a set of functions and P is a set, then we write $f \downarrow P$ for the function g with $dom(g) = dom(f)$ such that $g(c) = f(c) \lceil P$ for each $c \in dom(g)$. For more detail the interested reader can refer to [10].

The reader can notice that the main difference with respect to the model introduced by [2, 1] is that locations are not explicit, indeed they are given by state variables, trajectories and transitions. Moreover transitions from one state to another do not occur by crossing guards or leaving invariants, but they occur because of actions arising (see executions definition in Section 3).

Example 2. *Imagine now to describe the scenario in example 1 using hybrid automata. To represent HIOAs we use a variant of the TIOA language [9], with some extensions for hybrid systems [12]. The HIOA of a car is reported in fig. 2. It has an output variable K representing the ground pressure provided by the car and an output variable P representing the car position. The input variables are: the level of the ground* groundlevel *as a boolean saying if the level is low (1) or high (0); the* collisionrisk *saying if another car is in collision risk (1) or not (0). A function f is defined, giving the surface of the ground occupied by the car area starting from its position p_T and its orientation angle ϕ. We can imagine that f returns a rectangle centered in p_T with orientation ϕ. The pressure variable is updated with a function z depending on the mass m and area of the car. The velocity vel of the car is 0 if collisionrisk is true. Similarly the car slows down when groundlevel is true. For the sake of simplicity we used a boolean variable to represent the ground level changes, but any other function can be used, such as more general and complex diffusion equations. Note that we need to provide the system with an* external supervisor *which, taking as input the position and pressure of each car in the area at each instant of time, calculates the collision risk and the ground level around it. Basically the supervisor needs to know each*

type Rad = $\mathbb{R}|2\pi$
hioa Car
variables
input collisionrisk: Bool, groundlevel: Bool
internal ϕ: Rad, p_T: Real^2, m: Real, vel:Real
output P: Real^2, K: Real
trajectories
$K(t) = z(m, f(\phi, p_T))$;
$$vel(t) = \begin{cases} 0 & \text{if } collisionrisk \\ 0.5 & \text{if } groundlevel \\ 1 & \text{otherwise.} \end{cases};$$
$P(t) = p_T(t)$.

Figure 2: HIOA representing a car.

car direction and position at each instant of time for calculating the possibility of a collision with other cars moving in the same area and the level of the ground along the trajectory the car is following. We do not present the design of such a supervisor because it is out of the scope of this paper. Note also that this is just a possible representation of the scenario described in example 1. We used this simple way to show the need of using some external machinery (e.g. a supervisor) to model the interaction of agents with the environment.

In example 2 we are not able to represent implicit communication without adding some artificial machinery (in this case we used an external supervisor). Since our aim is to represent the system in a more natural way, we extend HIOA modeling framework to catch this aspect. To do this we specialize some variables of the HIOA, calling them *world variables*. The name is due to the fact that we want them to represent the connection between the agents and the surrounding world. Moreover world variables represent the changes in the environment as might be perceived by the agents. Hence the set of variables V is partitioned in a set W of world variables and a set S of standard automaton variables. The set W is partitioned in sets (U_w, X_w, Y_w) of *world input, internal, and output variables*, respectively, such that: $U_w \subseteq U, X_w \subseteq X, Y_w \subseteq Y$. To avoid confusion, we will add to automaton variables the subscript a: U_a, X_a, Y_a.

The main difference between world and automaton variables is that the type of world variables is a function of time and space, not only of time as in standard automaton variables. Hence world variables values (and trajectories) will depend both on the instant of time and the position in the underlying space. Formally, if we assume an underlying topological space $\mathcal{M}$, $\mathbf{w} : (\mathcal{T} \times \mathcal{M}) \to B$ for every $w \in W$, where $\mathcal{T}$ is the time axes and B is a set. For simplicity the reader may think of $\mathcal{M}$ as a metric space, e.g. $\mathbb{R}^3$. An automaton $\mathcal{A}$ will use its world inputs U_w to receive stimuli from the world it lives in. Analogously it will use its world outputs Y_w to give stimuli to the world it lives in. Finally internal world variables X_w are used to represent the world characteristics of $\mathcal{A}$. To keep the theory consistent with previous descriptions of automata, all the X variables represent persistent characteristics of the system. We will call HIOAs with world variables HIOAWs.

Example 3. *We now represent the car in fig. 2 with a HIOAW, extending the TIOA language to include world variables. Note that world variables are always described using their trajectories in time and space, i.e. they are described for any instant of time t and any point in space p. Each car is represented by a HIOAW as in fig. 3. It has an output world variable k representing the ground pressure provided by*

the car and an output world variable representing the car color ξ. The input world variables are: the level of the ground g and its color c. Each car perceives the ground level through a boolean variable g saying if the ground is low (1) or high (0). We used the boolean representation for the sake of simplicity. Of course any other function, like diffusion equations, may be used. Each car can check the color of the ground at each point of the area by the variable c, which represents a kind of colored map of the area. We assume that the color variable ξ takes the value black for all the points inside the car area given by f and white outside. The pressure variable k is updated with a function h depending on the mass m and area of the car, associating to each point in the area of the car the value of its pressure in time, and to each point outside the area of the car a 0 value. Two actions collision, level *represent the possibility that another car is in the neighborhood and that the level of the ground in the neighborhood is low, respectively. Action* collision *activates a boolean variable* stop *if there is any black point p^* in the neighborhood of the car, which is calculated by the function q returning a circle of radius r (bigger than the semi-diagonal of the rectangle representing the area of the car) and centered in p_T, but excluding the area of the car given by function f. Action* level *activates a boolean variable* slow *if there is any point p^* in the neighborhood of the car for which the ground level variable g is true, i.e. the level of the ground is low. Hence the velocity vel of the car is 0 if* stop *is true. Similarly the car slows down when* slow *is true. All the presented equations describing the car dynamics are very simple, but the description of the motion is out of the scope of this paper. Indeed they can be substituted by any other equations. The reader can notice that in fig. 2 the position of the car is explicit in variable P, which is an output that must be collected by the supervisor at each instant of time to check where the automaton is in the space. In the HIOAW of fig. 3 the position is embedded in the world variables and does not need to be explicitly put in an automaton variable. Indeed both color and pressure world variables carry the information about the position of the automaton in the space, due to their nature.*

The reader can notice that the automaton in fig. 3 has some input world variables. Here we considered the environment as an abstract entity, modifying and being modified by the agents living in it. As in the human sensing, the agents moving in an environment can catch these modifications as changes with respect to the nominal conditions of the surrounding area and interact with them. In the same way the agents change the environment. World variables aim at representing this exchange of implicit information because they give a map of environmental changes at each point of the space and each instant of time, without need of artificial machineries such as a supervisor.

3 Semantics

Executions of HIOAWs are defined as executions of HIOAs: an *execution fragment* of a HIOAW $\mathcal{A}$ is an (A,V)-sequence $\alpha = \tau_0 a_1 \tau_1 a_2 \tau_2 \ldots$, where $a_i \in A$, $\tau_i \in \mathcal{T}$; if τ_i is not the last trajectory of α, then $\tau_i.lstate \xrightarrow{a_{i+1}} \tau_{i+1}.fstate$. An execution fragment α is defined to be an *execution* if $\alpha.fstate$ is a start state, that is, $\alpha.fstate \in \Theta$. Results on executions of HIOAs are valid also for HIOAWs.

A *trace* of an execution fragment α captures the external behavior of a HIOAW, i.e. what it is needed to identify an automaton from outside. Calling $E = I \cup O$, $Z = U \cup Y$, a trace of a HIOAW $\mathcal{A}$ is then the (E,Z)-restriction of α. All the results on traces on HIOAs are still valid and exactly stated for HIOAWs. We say that a low-level specification $\mathcal{A}$ *implements* a high-level specification $\mathcal{B}$ if any behavior of $\mathcal{A}$ is also an allowed behavior of $\mathcal{B}$.

Definition 2. *Automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are* comparable *if they have the same external interface, that is, if world and local input and output sets of variables of $\mathcal{A}_1$ are equal to the corresponding sets of $\mathcal{A}_2$ and*

type Rad = $\mathbb{R}|2\pi$
hioaw Car
world variables
input g: Bool, c: Color;
output k: Real, ξ: Color;
automaton variables
internal ϕ: Rad, p_T: Real^2, m: Real, vel:Real, r:Real, stop: Bool, slow: Bool;
actions
hidden collision, level;
transitions
hidden collision
pre $\exists p^* \in q(p_T, r, f(\phi, p_T))$ s.t. $c(t, p^*) = \text{black}$
eff stop = true;
hidden level
pre $\exists p^* \in q(p_T, r, f(\phi, p_T))$ s.t. $g(t, p^*) = \text{true}$
eff slow = true;
trajectories

$$\xi(t,p) = \begin{cases} \text{black} & \text{if } p \in f(\phi, p_T) \\ \text{white} & \text{otherwise} \end{cases};$$

$$k(t,p) = h(m, f(\phi, p_T));$$

$$vel(t) = \begin{cases} 0 & \text{if stop} \\ 0.5 & \text{if slow} \\ 1 & \text{otherwise.} \end{cases}$$

Figure 3: HIOAW representing a car.

$E_1 = E_2$ *at all levels. If* $\mathcal{A}_1$ *and* $\mathcal{A}_2$ *are comparable then we say that* $\mathcal{A}_1$ implements $\mathcal{A}_2$*, denoted by* $\mathcal{A}_1 \leq \mathcal{A}_2$*, if* $traces(\mathcal{A}_1) \subseteq traces(\mathcal{A}_2)$.

Simulation relations between HIOAWs are defined as for HIOAs in Section 4.3 of [10]. We report here the definition:

Definition 3. *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be comparable automata. A* simulation *from* $\mathcal{A}$ *to* $\mathcal{B}$ *is a relation* $R \subseteq Q_{\mathcal{A}} \times Q_{\mathcal{B}}$ *satisfying the following conditions, for all states* $\mathbf{x} \lceil Q_{\mathcal{A}} \triangleq \mathbf{x}_{\mathcal{A}}$ *and* $\mathbf{x} \lceil Q_{\mathcal{B}} \triangleq \mathbf{x}_{\mathcal{B}}$ *of* $\mathcal{A}$ *and* $\mathcal{B}$*, respectively:*

1. *If* $\mathbf{x}_{\mathcal{A}} \in \Theta_{\mathcal{A}}$ *then there exists a state* $\mathbf{x}_{\mathcal{B}} \in \Theta_{\mathcal{B}}$ *such that* $\mathbf{x}_{\mathcal{A}}\ R\ \mathbf{x}_{\mathcal{B}}$.
2. *If* $\mathbf{x}_{\mathcal{A}}\ R\ \mathbf{x}_{\mathcal{B}}$ *and* α *is an execution fragment of* $\mathcal{A}$ *consisting of one action surrounded by two point trajectories, with* $\alpha.fstate = \mathbf{x}_{\mathcal{A}}$*, then* $\mathcal{B}$ *has a closed execution fragment* β *with* $\beta.fstate = \mathbf{x}_{\mathcal{B}}$*,* $trace(\beta) = trace(\alpha)$*, and* $\alpha.lstate\ R\ \beta.lstate$.
3. *If* $\mathbf{x}_{\mathcal{A}}\ R\ \mathbf{x}_{\mathcal{B}}$ *and* α *is an execution fragment of* $\mathcal{A}$ *consisting of a single closed trajectory, with* $\alpha.fstate = \mathbf{x}_{\mathcal{A}}$*, then* $\mathcal{B}$ *has a closed execution fragment* β *with* $\beta.fstate = \mathbf{x}_{\mathcal{B}}$*,* $trace(\beta) = trace(\alpha)$*, and* $\alpha.lstate\ R\ \beta.lstate$.

Results on trace inclusion for simulation of HIOAs are valid also for HIOAWs. We also report here an important corollary on simulation relations which will be used in the rest of the paper.

Corollary 1. *Let* $\mathcal{A}$ *and* $\mathcal{B}$ *be comparable automata and let* R *be a simulation from* $\mathcal{A}$ *to* $\mathcal{B}$*. Then* $traces(\mathcal{A}) \subseteq traces(\mathcal{B})$.

3.1 Padding of executions

We introduce now the notion of *padding* of executions that will be used in the following proofs.

Definition 4. *A* padded execution *of a HIOAW $\mathcal{A}$ is an $(A \cup \{\varepsilon\}, V)-$sequence $\gamma = \tau_0 a_1 \tau_1 a_2 \tau_2 a_3 \ldots$ such that if $a_i = \varepsilon$ then $\tau_{i-1}.lstate = \tau_i.fstate$.*

Definition 5. Padding.
We call padding *of an execution α any padded execution obtained by α by extending the actions set with ε.*

For example a padded execution of an execution $\alpha = \tau_0 a_1 \tau_1 a_2 \tau_2 a_3 \ldots$ is $\gamma = \tau_0' \varepsilon \tau_0'' a_1 \tau_1 a_2 \tau_2 a_3 \ldots$, where $\tau_0' \frown \tau_0'' = \tau_0$.

Definition 6. *The restriction of a padded execution γ to a set of actions A' and a set of variables V' is the (A', V')-restriction of γ.*

Lemma 1. *Let γ be a padded execution of $\mathcal{A}$. Then there exists α, execution of $\mathcal{A}$, for which γ is a padding.*

Proof. Let A, V be the sets of actions and variables of $\mathcal{A}$, respectively. Then, by definition of restriction of padded executions and by definition of executions, $\alpha = \gamma \lceil (A, V)$ is an execution of $\mathcal{A}$. By definition of padding γ is a padding of α. □

Lemma 2. *Let α be an execution of $\mathcal{A}$ defined in (A, V) and γ a padding of α. Let $A' \subseteq A, V' \subseteq V$, then $\alpha \lceil (A', V') = \gamma \lceil (A', V')$.*

Proof. Straightforward by definition of restriction of a padded execution and of an execution and by definition of padding. □

Definition 7. *A trace of a padded execution γ is defined as $\gamma \lceil (E, Z)$.*

Lemma 3. *Let α be an execution of $\mathcal{A}$ and γ a padding of α, then $trace(\alpha) = trace(\gamma)$.*

Proof. Straightforward by lemma 2 and definition of trace of a padded execution. □

Lemma 4. *Let γ be a padding of α, execution of $\mathcal{A}$, and let γ' be a prefix of γ. Then $\gamma' \lceil (A, V)$ is a prefix of α.*

Lemma 5. *Given n executions, it is always possible to find n paddings of these executions such that all corresponding trajectories have the same length.*

4 Parallel composition

In this section we introduce parallel composition for HIOAWs. First of all some compatibility conditions have to be stated.

Definition 8. *Two HIOAWs $\mathcal{A}_1$ and $\mathcal{A}_2$ are compatible if*

1. $(U_{w1} \cup U_{w2}) \cap (Y_{w1} \cup Y_{w2}) = \emptyset$.
2. $H_1 \cap A_2 = H_2 \cap A_1 = \emptyset$,
3. $X_1 \cap V_2 = X_2 \cap V_1 = \emptyset$,
4. $O_1 \cap O_2 = \emptyset$,

5. $Y_1 \cap Y_2 = \emptyset$.

The reader may notice that conditions 2 to 5 are the classical compatibility conditions for HIOAs. The first condition states that no explicit communication between the two HIOAWs is possible via world variables. Indeed, by definition, explicit communication between HIOAWs occurs only via automaton I/O variables, whereas world variables are used for implicit communication. These conditions, when not satisfied by the HIOAWs, can be obtained by changing variables names.

Definition 9. Parallel composition
If $\mathcal{A}_1, \mathcal{A}_2$ are two compatible HIOAWs, then their composition $\mathcal{A}_1 \| \mathcal{A}_2$ is defined as the structure $\mathcal{A}$ where:

1. $U_w = U_{w1} \cup U_{w2}$, $X_w = X_{w1} \cup X_{w2}$, $Y_w = Y_{w1} \cup Y_{w2}$
2. $Y_a = Y_{a1} \cup Y_{a2}$, $X_a = X_{a1} \cup X_{a2}$, $U_a = (U_{a1} \cup U_{a2}) \setminus Y_a$
3. $O = O_1 \cup O_2$, $I = (I_1 \cup I_2) \setminus O$ *and* $H = H_1 \cup H_2$
4. $Q = \{\mathbf{x} \in vals(X) \mid \mathbf{x} \lceil X_1 \in Q_1 \wedge \mathbf{x} \lceil X_2 \in Q_2\}$
5. $\Theta = \{\mathbf{x} \in Q \mid \mathbf{x} \lceil X_1 \in \Theta_1 \wedge \mathbf{x} \lceil X_2 \in \Theta_2\}$
6. $D = \{(\mathbf{x}, a, \mathbf{x}') \mid$ *for each* $i \in \{1,2\}$ *either* $a \in A_i$ *and* $\mathbf{x} \lceil X_i \xrightarrow{a} \mathbf{x}' \lceil X_i$, *or* $a \notin A_i$ *and* $\mathbf{x} \lceil X_i = \mathbf{x}' \lceil X_i\}$.
7. $\mathcal{T} = \{\tau \mid$ *there exists* $\tau_1 \in \mathcal{T}_1, \tau_2 \in \mathcal{T}_2$ *such that* $\tau \downarrow (V_i \setminus (Y_{w1} \cap Y_{w2})) = \tau_i \downarrow (V_i \setminus (Y_{w1} \cap Y_{w2})), i \in \{1,2\}$ *and* $\tau \downarrow (Y_{w1} \cap Y_{w2}) = \tau_1 \downarrow (Y_{w1} \cap Y_{w2}) + \tau_2 \downarrow (Y_{w1} \cap Y_{w2})\}$

This definition of parallel composition is very similar to the one for HIOAs. The only two differences are given by the first and the last conditions. The first condition depends on compatibility: there is no communication between the two HIOAWs via world variables. The last condition indeed is the main difference with HIOAs composition. It might be that the two composing automata have some output world variables with the same kind of information for the external world. These output world variables will have the same name and then their intersection is not empty. For those variables it is necessary to sum the trajectories as defined in the following. We call sum any generic operator with the same characteristics of the sum in $\mathbb{R}$. In the following we will define the *sum* as an additive operator in a group. Let τ_0, τ_1 be two trajectories with the same time domain, such that: $\tau_0 : [0,t] \to (V_0 \to \mathcal{D})$ and $\tau_1 : [0,t] \to (V_1 \to \mathcal{D})$, where V_0, V_1 are sets of variables. Let $\mathcal{D}$ be a domain of values for variables in V_0, V_1 (e.g. $\mathbb{R}$), such that its structure is a (commutative) group $\mathcal{G}$, with an operator $+_G$ and an identity element called 0_G. Note that the subscript G will be omitted when it is clear from the context.

Definition 10. *The sum of τ_0, τ_1 is defined as:*

$$(\tau_0 + \tau_1)(t)(v) = \begin{cases} \tau_i(t)(v) & \text{if } v \in V_i \setminus V_{1-i} \\ \tau_0(t)(v) + \tau_1(t)(v) & \text{if } v \in V_0 \cap V_1 \end{cases}$$

For the sake of simplicity in the following we will consider the operation of sum in $\mathbb{R}$. But all the results presented in this paper are still valid using any other operator with the same mathematical characteristics.

We report here some lemmas on trajectories that will be used in the following proof.

Lemma 6. *Let τ be a trajectory in V. Let $I \subseteq dom(\tau)$ and $V' \subseteq V$. Then $(\tau \lceil I) \downarrow V' = (\tau \downarrow V') \lceil I$.*

Lemma 7. *Let τ be a trajectory in V. Let $V' \subseteq V$. Then $(\tau \trianglerighteq t) \downarrow V' = (\tau \downarrow V') \trianglerighteq t$.*

Lemma 8. *Let τ be a trajectory in V such that $\tau = \tau_0 \frown \tau_1 \frown \tau_2 \frown \ldots$. Let $V' \subseteq V$. Then $(\tau_0 \frown \tau_1 \frown \tau_2 \frown \ldots) \downarrow V' = (\tau_0 \downarrow V') \frown (\tau_1 \downarrow V') \frown (\tau_2 \downarrow V') \frown \ldots$.*

Proposition 1. *The composition of two HIOAWs is a HIOAW.*

Proof. We show that $\mathcal{A}_1 \| \mathcal{A}_2$ satisfies the properties of a HIOAW. Disjointness of the U, X, Y sets follows from disjointness of the same sets in $\mathcal{A}_1$ and $\mathcal{A}_2$ and compatibility. Similarly for the actions. Nonemptiness of starting state follows from nonemptiness of starting states of $\mathcal{A}_1$ and $\mathcal{A}_2$ and disjointness of X_1 and X_2. We verify the **T** properties of trajectories (see definition 1). Let C_{12} be $Y_{w1} \cap Y_{w2}$.

T1 We want to prove that for every $\tau \in \mathcal{T}$ and every $\tau' \leq \tau$, $\tau' \in \mathcal{T}$. Let τ be a trajectory in $\mathcal{T}$. Let $i \in \{1,2\}$. By the definition of parallel composition there exists $\tau_1 \in \mathcal{T}_1, \tau_2 \in \mathcal{T}_2$ such that $\tau \downarrow (V_i \setminus C_{12}) = \tau_i \downarrow (V_i \setminus C_{12})$, and $\tau \downarrow C_{12} = \tau_1 \downarrow C_{12} + \tau_2 \downarrow C_{12}$. Let $\tau' \leq \tau$. By definition of prefix we have that $\tau' = \tau \lceil I$ with $I = dom(\tau') \subseteq dom(\tau)$. Hence we can state that $\tau' \downarrow (V_i \setminus C_{12}) = (\tau \lceil I) \downarrow (V_i \setminus C_{12})$. By lemma 6 $(\tau \lceil I) \downarrow (V_i \setminus C_{12}) = (\tau \downarrow (V_i \setminus C_{12})) \lceil I$. By definition of parallel composition and again by lemma 6 $(\tau \downarrow (V_i \setminus C_{12})) \lceil I = (\tau_i \downarrow (V_i \setminus C_{12})) \lceil I = (\tau_i \lceil I) \downarrow (V_i \setminus C_{12})$. Let $\tau'_1 = \tau_1 \lceil I$ and $\tau'_2 = \tau_2 \lceil I$, then $(\tau_i \lceil I) \downarrow (V_i \setminus C_{12}) = \tau'_i \downarrow (V_i \setminus C_{12})$. Analogously, for the second statement of parallel composition of trajectories we have that $\tau' \downarrow C_{12} = (\tau \lceil I) \downarrow C_{12} = (\tau \downarrow C_{12}) \lceil I = (\tau_1 \downarrow C_{12}) \lceil I + (\tau_2 \downarrow C_{12}) \lceil I = (\tau_1 \lceil I) \downarrow C_{12} + (\tau_2 \lceil I) \downarrow C_{12} = \tau'_1 \downarrow C_{12} + \tau'_2 \downarrow C_{12}$. Hence $\tau' \in \mathcal{T}$.

T2 We want to prove that for every $\tau \in \mathcal{T}$ and every $t \in dom(\tau)$, $\tau \trianglerighteq t \in \mathcal{T}$. Let τ be a trajectory in $\mathcal{T}$. Let $i \in \{1,2\}$. By the definition of parallel composition there exists $\tau_1 \in \mathcal{T}_1, \tau_2 \in \mathcal{T}_2$ such that $\tau \downarrow (V_i \setminus C_{12}) = \tau_i \downarrow (V_i \setminus C_{12})$, and $\tau \downarrow C_{12} = \tau_1 \downarrow C_{12} + \tau_2 \downarrow C_{12}$. Hence, since $dom(\tau_1) = dom(\tau_2) = dom(\tau)$, by lemma 7 we have that $(\tau \trianglerighteq t) \downarrow (V_i - C_{12}) = (\tau \downarrow (V_i - C_{12})) \trianglerighteq t = (\tau_i \downarrow (V_i - C_{12})) \trianglerighteq t = (\tau_i \trianglerighteq t) \downarrow (V_i - C_{12})$. Moreover $(\tau \trianglerighteq t) \downarrow C_{12} = (\tau \downarrow C_{12}) \trianglerighteq t = (\tau_1 \downarrow C_{12}) \trianglerighteq t + (\tau_2 \downarrow C_{12}) \trianglerighteq t = (\tau_1 \trianglerighteq t) \downarrow C_{12} + (\tau_2 \trianglerighteq t) \downarrow C_{12}$. Recall that by the properties of trajectories $\tau \trianglerighteq t$ is still a trajectory. Hence $\tau \trianglerighteq t \in \mathcal{T}$.

T3 We want to prove that set $\mathcal{T}$ is closed under concatenation. Let $\tau_0, \tau_1, \tau_2, \ldots$ be a sequence of trajectories in $\mathcal{T}$, such that, for each nonfinal index j, τ_j is closed and $\tau_j.lstate = \tau_{j+1}.fstate$. Let τ be $\tau_0 \frown \tau_1 \frown \tau_2 \frown \ldots$. Let $i \in \{1,2\}$. By definition of parallel composition for each τ_j, $\exists \tau_{1j}, \tau_{2j}$ such that $\tau_j \downarrow (V_i \setminus C_{12}) = \tau_{ij} \downarrow (V_i \setminus C_{12})$, and $\tau_j \downarrow C_{12} = \tau_{1j} \downarrow C_{12} + \tau_{2j} \downarrow C_{12}$. Let τ_i be $\tau_{i0} \frown \tau_{i1} \frown \tau_{i2} \frown \ldots$. Hence by lemma 8 $\tau \downarrow (V_i \setminus C_{12}) = (\tau_0 \downarrow (V_i \setminus C_{12})) \frown (\tau_1 \downarrow (V_i \setminus C_{12})) \frown (\tau_2 \downarrow (V_i \setminus C_{12})) \frown \ldots = (\tau_{i0} \downarrow (V_i \setminus C_{12})) \frown (\tau_{i1} \downarrow (V_i \setminus C_{12})) \frown (\tau_{i2} \downarrow (V_i \setminus C_{12})) \frown \ldots = (\tau_{i0} \frown \tau_{i1} \frown \tau_{i2} \frown \ldots) \downarrow (V_i \setminus C_{12}) = \tau_i \downarrow (V_i \setminus C_{12})$. Moreover $\tau \downarrow C_{12} = (\tau_0 \downarrow C_{12}) \frown (\tau_1 \downarrow C_{12}) \frown (\tau_2 \downarrow C_{12}) \frown \ldots = (\tau_{10} \downarrow C_{12} + \tau_{20} \downarrow C_{12}) \frown (\tau_{11} \downarrow C_{12} + \tau_{21} \downarrow C_{12}) \frown (\tau_{12} \downarrow C_{12} + \tau_{22} \downarrow C_{12}) \frown \ldots = ((\tau_{10} \downarrow C_{12}) \frown (\tau_{11} \downarrow C_{12}) \frown (\tau_{12} \downarrow C_{12}) \frown \ldots) + ((\tau_{20} \downarrow C_{12}) \frown (\tau_{21} \downarrow C_{12}) \frown (\tau_{22} \downarrow C_{12}) \frown \ldots) = (\tau_{10} \frown \tau_{11} \frown \tau_{12} \frown \ldots) \downarrow C_{12} + (\tau_{20} \frown \tau_{21} \frown \tau_{22} \frown \ldots) \downarrow C_{12} = \tau_1 \downarrow C_{12} + \tau_2 \downarrow C_{12}$. Hence $\tau \in \mathcal{T}$.

□

Example 4. *Consider again example 1. Suppose to have a HIOAW representing a car as in fig. 3 in the sandy area, called $\mathcal{B}_1$. Another car represented by $\mathcal{B}_2$ (again of type represented in fig. 3) enters the sandy area. We want to compose the two cars. Variables and actions of $\mathcal{B}_1$ are labelled by the subscript 1, the ones of $\mathcal{B}_2$ by the subscript 2. The obtained HIOAW $\mathcal{B}_1 \| \mathcal{B}_2$ is represented in fig. 4. Notice that the effect of the output world variables are summed: the ground pressure k of $\mathcal{B}_1 \| \mathcal{B}_2$ represents a sort of a map of the values taken by the pressures given by the cars in the considered area, the same for the color ξ. Indeed here we did not make any constraints of two cars being at the same point at a time, because the model can take into account also collisions.*

We now show that simulation relation and trace inclusion are preserved by composition. The main difficulty compared to the analogous results in [10] is that output world variables sum their effects. This means that it is not possible anymore to project executions of a composite system to obtain executions

of the components. Rather we have to show that for each execution of the composite system there are executions of the components that can be pasted together.

hioaw $\mathcal{B}_1 \| \mathcal{B}_2$
world variables
input g: Bool, c: Color;
output k: Real, ξ: Color;
automaton variables
internal ϕ_1: Rad, p_{T1}: Real^2, m_1: Real, vel_1:Real, r_1:Real, ϕ_2: Rad, p_{T2}: Real^2, m_2: Real, vel_2:Real, r_2:Real, stop$_1$: Bool, stop$_2$: Bool, slow$_1$: Bool, slow$_2$: Bool;
actions
hidden collision$_1$, collision$_2$, level$_1$, level$_2$;
transitions
hidden collision$_1$
pre $\exists p^* \in q(p_{T1}, r_1)$ s.t. $c(t, p^*) = \text{black}$
eff stop$_1$ = true;
hidden collision$_2$
pre $\exists p^* \in q(p_{T2}, r_2)$ s.t. $c(t, p^*) = \text{black}$
eff stop$_2$ = true;
hidden level$_1$
pre $\exists p^* \in q(p_{T1}, r_1)$ s.t. $g(t, p^*) = \text{true}$
eff slow$_1$ = true;
hidden level$_2$
pre $\exists p^* \in q(p_{T2}, r_2)$ s.t. $g(t, p^*) = \text{true}$
eff slow$_2$ = true;
trajectories

$$\xi(t,p) = \begin{cases} \text{black} & \text{if } p \in f(\phi_1, p_{T1}) \vee p \in f(\phi_2, p_{T2}) \\ \text{white} & \text{otherwise} \end{cases};$$

$$k(t,p) = h(m_1, f(\phi_1, p_{T1})) + h(m_2, f(\phi_2, p_{T2}));$$

$$vel_1(t) = \begin{cases} 0 & \text{if stop}_1 \\ 0.5 & \text{if slow}_1 \\ 1 & \text{otherwise.} \end{cases}$$

$$vel_2(t) = \begin{cases} 0 & \text{if stop}_2 \\ 0.5 & \text{if slow}_2 \\ 1 & \text{otherwise.} \end{cases}$$

Figure 4: HIOAW representing parallel composition of $\mathcal{B}_1$ and $\mathcal{B}_2$.

Lemma 9. *Let $\mathcal{A} = \mathcal{A}_1 \parallel \mathcal{A}_2$ and let α be an execution fragment of $\mathcal{A}$. Then $\exists \alpha_1, \alpha_2$ execution fragments of $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively, such that*

1. $\alpha \lceil (A_i, V_i \setminus C_{12}) = \alpha_i \lceil (A_i, V_i \setminus C_{12}), i = 1,2$, *and*
2. $\alpha \lceil (\emptyset, C_{12}) = \alpha_1 \lceil (\emptyset, C_{12}) + \alpha_2 \lceil (\emptyset, C_{12})$,

with $C_{12} = (Y_{w1} \cap Y_{w2})$.

Proof. Let $\alpha = \tau_0 a_1 \tau_1 a_2 \tau_2 a_3 \ldots \in frags_{\mathcal{A}}$. By definition of parallel composition, since each $\tau_j \in \mathcal{T}$ there exists $\tau_{j1} \in \mathcal{T}_1, \tau_{j2} \in \mathcal{T}_2$ such that: $\tau_j \downarrow (V_1 \setminus C_{12}) = \tau_{j1} \downarrow (V_1 \setminus C_{12})$, $\tau_j \downarrow (V_2 \setminus C_{12}) = \tau_{j2} \downarrow (V_2 \setminus C_{12})$

and $\tau_j \downarrow C_{12} = \tau_{j1} \downarrow C_{12} + \tau_{j2} \downarrow C_{12}$. Hence by definition of padding we can build two padded executions $\gamma_1 = \tau_{01} a'_1 \tau_{11} a'_2 \tau_{21} a'_3 \ldots, \gamma_2 = \tau_{02} a''_1 \tau_{12} a''_2 \tau_{22} a''_3 \ldots$ of $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively, where

$$a'_j = \begin{cases} a_j & \text{if } a_j \in A_1 \\ \varepsilon & \text{otherwise} \end{cases} \qquad a''_j = \begin{cases} a_j & \text{if } a_j \in A_2 \\ \varepsilon & \text{otherwise} \end{cases}$$

Let $i \in \{1,2\}$. By construction of γ_i it is $\alpha \lceil (A_i, V_i \setminus C_{12}) = \gamma_i \lceil (A_i, V_i \setminus C_{12})$. By lemma 1 it is possible to define the execution α_i of $\mathcal{A}_i$ for which γ_i is a padding. Then by lemma 2 it holds: $\gamma_i \lceil (A_i, V_i \setminus C_{12}) = \alpha_i \lceil (A_i, V_i \setminus C_{12})$ which proves point 1 of this lemma. Moreover, since the projection of an execution on an empty set of action gives a trajectory, we have that $\alpha \lceil (\emptyset, C_{12}) = \gamma_1 \lceil (\emptyset, C_{12}) + \gamma_2 \lceil (\emptyset, C_{12}) = \alpha_1 \lceil (\emptyset, C_{12}) + \alpha_2 \lceil (\emptyset, C_{12})$. By definition 10 the last statement proves point 2 of this lemma. □

The following lemma from HIOAs applies directly to HIOAWs. The proof is reported in [10].

Lemma 10. *Let $\mathcal{A} = \mathcal{A}_1 \| \mathcal{A}_2$, and let α be an execution fragment of $\mathcal{A}$. Then, for $i = 1,2$, $trace(\alpha) \lceil (E_i, Z_i) = trace(\alpha \lceil (A_i, V_i))$.*

The following proposition relates the set of traces of a composite automaton to the sets of traces of the component automata.

Proposition 2. *Let $\mathcal{A} = \mathcal{A}_1 \| \mathcal{A}_2$ and β a trace of $\mathcal{A}$. Then $\exists \beta_1, \beta_2$ traces of $\mathcal{A}_1, \mathcal{A}_2$ respectively, such that*

1. $\beta \lceil (E_i, Z_i \setminus C_{12}) = \beta_i \lceil (E_i, Z_i \setminus C_{12}), i = 1,2$ *and*
2. $\beta \lceil (\emptyset, C_{12}) = \beta_1 \lceil (\emptyset, C_{12}) + \beta_2 \lceil (\emptyset, C_{12})$,

with $C_{12} = (Y_{w1} \cap Y_{w2})$.

Proof. Let β be a trace of $\mathcal{A}$. By definition of trace $\exists \alpha \in execs(\mathcal{A})$ such that $\beta = trace(\alpha)$. By Lemma 9, $\exists \alpha_1, \alpha_2$ execution fragments of $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively, such that $\alpha \lceil (A_i, V_i \setminus C_{12}) = \alpha_i \lceil (A_i, V_i \setminus C_{12}), i = 1,2$, and $\alpha \lceil (\emptyset, C_{12}) = \alpha_1 \lceil (\emptyset, C_{12}) + \alpha_2 \lceil (\emptyset, C_{12})$. Let $\beta_1 = trace(\alpha_1)$ and $\beta_2 = trace(\alpha_2)$. We want to prove that $\beta \lceil (E_i, Z_i \setminus C_{12}) = \beta_i \lceil (E_i, Z_i \setminus C_{12}), i = 1,2$ and $\beta \lceil (\emptyset, C_{12}) = \beta_1 \lceil (\emptyset, C_{12}) + \beta_2 \lceil (\emptyset, C_{12})$. By Lemma 10, $\beta \lceil (E_i, Z_i) = trace(\alpha \lceil (A_i, V_i))$. Moreover by the properties of projection of executions, we have that $\beta \lceil (E_i, Z_i \setminus C_{12}) = trace(\alpha \lceil (A_i, V_i \setminus C_{12}))$ and $\beta \lceil (\emptyset, C_{12}) = trace(\alpha \lceil (\emptyset, C_{12}))$. Furthermore by the properties of projection of executions we have that $(\alpha_i \lceil (A_i, V_i \setminus C_{12})) \lceil (E_i, Z_i \setminus C_{12}) = \alpha_i \lceil (A_i \cap E_i, (V_i \cap Z_i) \setminus C_{12})$. Since by definition $A_i \cap E_i = E_i$ and $V_i \cap Z_i = Z_i$, we obtain $\alpha_i \lceil (A_i \cap E_i, (V_i \cap Z_i) \setminus C_{12}) = trace(\alpha) \lceil (E_i, Z_i \setminus C_{12})$. Similarly for projections on C_{12}. □

The next two theorems prove the results on substitutivity for implementation and simulation relations.

Theorem 1. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be comparable HIOAWs with $\mathcal{A}_1 \leq \mathcal{A}_2$. Let $\mathcal{B}$ be a HIOAW compatible with each of $\mathcal{A}_1$ and $\mathcal{A}_2$. Then $\mathcal{A}_1 \| \mathcal{B}$ and $\mathcal{A}_2 \| \mathcal{B}$ are comparable and $\mathcal{A}_1 \| \mathcal{B} \leq \mathcal{A}_2 \| \mathcal{B}$.*

Proof. Let α be an execution of $\mathcal{A}_1 \parallel \mathcal{B}$. By lemma 9, two executions α_1, α_B exist, such that $\alpha_1 \in execs(\mathcal{A}_1)$, $\alpha_B \in execs(\mathcal{B})$ and: $\alpha \lceil (A_1, V_1 \setminus C_{1B}) = \alpha_1 \lceil (A_1, V_1 \setminus C_{1B})$, $\alpha \lceil (A_B, V_B \setminus C_{1B}) = \alpha_B \lceil (A_B, V_B \setminus C_{1B})$, $\alpha \lceil (\emptyset, C_{1B}) = \alpha_1 \lceil (\emptyset, C_{1B}) + \alpha_B \lceil (\emptyset, C_{1B})$, with $C_{1B} = Y_{w1} \cap Y_{wB}$. By lemma 5 we can take paddings of $\alpha, \alpha_1, \alpha_B$ such that the j^{th} trajectory has the same length for all j. Let these paddings be $\gamma, \gamma_1, \gamma_B$ respectively with $\gamma = \tau_0 a_1 \tau_1 a_2 \tau_2 a_3 \ldots$, $\gamma_1 = \tau_{01} a'_1 \tau_{11} a'_2 \tau_{21} a'_3 \ldots$ and $\gamma_B = \tau_{0B} a''_1 \tau_{1B} a''_2 \tau_{2B} a''_3 \ldots$. Since $\mathcal{A}_1 \leq \mathcal{A}_2$ and by compatibility, we can find an execution α_2 of $\mathcal{A}_2$ with the same trace of α_1 and a padding of α_2 following lemma 5. We write $\gamma_2 = \tau_{02} a'''_1 \tau_{12} a'''_2 \tau_{22} a'''_3 \ldots$. By the definition of composition the execution of $\mathcal{A}_2 \parallel \mathcal{B}$ obtained by γ_2 and γ_B will be $\gamma' = \tau'_0 b_1 \tau'_1 b_2 \tau'_2 b_3 \ldots$, where $\tau'_j \downarrow (V_2 \setminus C_{2B}) = \tau_{j2} \downarrow (V_2 \setminus C_{2B})$, $\tau'_j \downarrow (V_B \setminus C_{2B}) = \tau_{jB} \downarrow (V_B \setminus C_{2B})$, $\tau'_j \downarrow C_{2B} = \tau_{j2} \downarrow C_{2B} + \tau_{jB} \downarrow C_{2B}$, where $C_{2B} = Y_{w2} \cap Y_{wB}$. This

is valid even if the trajectories in the padded executions have not the length of the original trajectories, by definition of prefix of a trajectory and prefix closure of trajectories in a HIOAW. Actions b_i might be different, but by construction, compatibility and lemma 3 we have that γ' has the same trace of γ hence of α. Indeed the (padded) executions can differ only in their internal variables (state), but they do not influence the traces (external variables). For this reason we can state that $traces(\mathcal{A}_1 \parallel \mathcal{B}) \subseteq traces(\mathcal{A}_2 \parallel \mathcal{B})$, hence, by definition 2 of implementation, $\mathcal{A}_1 \| \mathcal{B} \leq \mathcal{A}_2 \| \mathcal{B}$. □

Corollary 2. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be compatible HIOAWs, and let R be a simulation relation between $\mathcal{A}_1$ and $\mathcal{A}_2$. Let $\mathcal{B}$ be a HIOAW compatible with each of $\mathcal{A}_1$ and $\mathcal{A}_2$. Then $\mathcal{A}_1 \| \mathcal{B} \leq \mathcal{A}_2 \| \mathcal{B}$.*

Proof. Since $\mathcal{A}_1 \; R \; \mathcal{A}_2$, by corollary 1, $traces(\mathcal{A}_1) \subseteq traces(\mathcal{A}_2)$. By definition 2 of implementation, $\mathcal{A}_1 \leq \mathcal{A}_2$. By theorem 1 this implies that for any $\mathcal{B}$, $\mathcal{A}_1 \| \mathcal{B} \leq \mathcal{A}_2 \| \mathcal{B}$. □

Theorem 2. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be compatible HIOAWs, and let R be a simulation relation between $\mathcal{A}_1$ and $\mathcal{A}_2$. Let $\mathcal{B}$ be a HIOAW compatible with each of $\mathcal{A}_1$ and $\mathcal{A}_2$. Then $\exists \; R'$ such that $(\mathcal{A}_1 \| \mathcal{B}) \; R' \; (\mathcal{A}_2 \| \mathcal{B})$.*

Proof. Let R' be a relation between $\mathcal{A}_1 \| \mathcal{B}$ and $\mathcal{A}_2 \| \mathcal{B}$ such that for each $x_1 \in Q_1$, $x_2 \in Q_2$, $x_B, x'_B \in Q_B$, $(x_1, x_B) \; R' \; (x_2, x'_B)$ iff $(x_1 \; R \; x_2) \wedge (x_B = x'_B)$. We prove that R' is a simulation relation by proving that R' satisfies each point of definition 3.

1. Since for each $x_1 \in \Theta_1$, $x_2 \in \Theta_2$, $x_1 \; R \; x_2$, then by definition of R', for each initial state (x_1, x_B) of $\mathcal{A}_1 \| \mathcal{B}$ and each initial state (x_2, x_B) of $\mathcal{A}_2 \| \mathcal{B}$, $(x_1, x_B) \; R' \; (x_2, x_B)$, with $x_B \in \Theta_B$.
2. Let α be an execution fragment of $\mathcal{A}_1 \| \mathcal{B}$ consisting of one action surrounded by two point trajectories, with $\alpha.fstate = (x_1, x_B)$. Let $\alpha.lstate = (x'_1, x'_B)$. Since $x_1 \; R \; x_2$ then $\exists x'_2 \in Q_2$ such that $x'_1 \; R \; x'_2$. Since $\mathcal{A}_1 \; R \; \mathcal{A}_2$, by corollary 2, $\mathcal{A}_1 \| \mathcal{B} \leq \mathcal{A}_2 \| \mathcal{B}$. Then there exists β execution fragment of $\mathcal{A}_2 \| \mathcal{B}$ with the same trace of α and $\beta.fstate = (x_2, x_B)$. By definition of parallel composition there exists an action bringing the state to (x'_2, x'_B). Hence by definition of R' we have that $(x'_1, x'_B) \; R' \; (x'_2, x'_B)$.
3. Let α be and execution fragment of $\mathcal{A}_1 \| \mathcal{B}$ such that $\alpha = \tau \in \mathcal{T}$ closed and with $\alpha.fstate = (x_1, x_B)$. Let β be an execution fragment of $\mathcal{A}_2 \| \mathcal{B}$ such that $\beta.fstate = (x_2, x_B)$. Let $\alpha.lstate = (x'_1, x'_B)$ and $\beta.lstate = (x'_2, x'_B)$. Since $x_1 \; R \; x_2$, by definition of R' it is $(x_1, x_B) \; R' \; (x_2, x_B)$. By corollary 2, there exists an execution fragment β of $\mathcal{A}_2 \| \mathcal{B}$ with the same trace of α.

□

5 Conclusions

In this paper we have proposed an extension of the Hybrid I/O Automaton model of [10] to provide a natural representation of the fact that objects move in a world that they can observe and modify. We started from the analysis of the case studies of the C4C project, representing agents that move in a dynamical environment and have to achieve a goal by coordination. Besides the classical signals that automata send to each other either via discrete communication events or shared continuous variables, we specialized some variables of HIOAs to let them communicate implicitly by affecting their surrounding world and observing the effects on the worlds of the activity of other automata. This mechanism for interaction turns out to be adequate for compositional analysis, which is one of the main features of HIOAs that we wanted to keep in an extended model. Indeed we introduced the notion of parallel composition, and proved compositionality results. The natural extension of this formalism to model environment has been reported in [4], leading to a hierarchical representation of automata and introducing

the ability of composing them *vertically* into nested worlds. We presented in this paper a toy example to show the application of the theory, but a more complex and reality-based application can be found in [11]. The simulation tools are under study. Future research directions include the ability to describe scenarios where automata are created and destroyed and where communication links change dynamically.

References

[1] R. Alur, C. Courcoubetis, T. Henziger & P. Ho (1993): *Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems*. Lecture Notes in Computer Science 736, pp. 209–229, doi:10.1007/3-540-57318-6_30.

[2] R. Alur & D. Dill (1990): *Automata For Modeling Real-Time Systems*. Lecture Notes in Computer Science 443, pp. 322–335, doi:10.1007/BFb0032042.

[3] R. Alur & T.A. Henzinger (1997): *Modularity for timed and hybrid systems*. In: Ninth International Conference on Concurrency Theory. Lecture Notes in Computer Science 1243, Springer, pp. 74–88, doi:10.1007/3-540-63141-0_6.

[4] M. Capiluppi & R. Segala (2012): *Hybrid automata with worlds: A compositional approach to modeling objects that move in a complex environment*. Technical Report RR 87/2012, Department of Computer Science, University of Verona.

[5] J. B. De Sousa, K. H. Johansson, A. Speranzon & J. Silva (2005): *A control architecture for multiple submarines in coordinated search missions*. In: 16th IFAC World Congress on Automatic Control.

[6] A. Deshpande, A. Gollu & L. Semenzato (1998): *The SHIFT Programming Language for Dynamic Networks of Hybrid Automata*. IEEE Transactions on automatic control 43(4), doi:10.1109/9.664163.

[7] T.A. Henzinger, P.W. Kopke, A. Puri & P Varaiya (1998): *What's decidable about hybrid automata?* Journal of Computer and System Sciences 57, pp. 94–124, doi:10.1006/jcss.1998.1581.

[8] Y. Jin, Y. Liao & M.M. Polycarpou (2006): *Balancing search and target response in cooperative unmanned aerial vehicle (UAV) teams*. IEEE Transactions on systems, man, and cybernetics 38/3.

[9] D.K Kaynar, N. Lynch, R. Segala & F. Vaandrager (2006): *The Theory of Timed I/O Automata*. Synthesis Lectures on Computer Science doi:10.2200/S00006ED1V01Y200508CSL001.

[10] N. Lynch, R. Segala & F. Vaandrager (2003): *Hybrid I/O automata*. Information and Computation 185, pp. 105–157, doi:10.1016/S0890-5401(03)00067-1.

[11] E.N. Marinica, M. Capiluppi, J.A. Rogge, R. Segala & R.K. Boel (2012): *Distributed collision avoidance for autonomous vehicles: world automata representation*. In: 4th IFAC Conference on Analysis and Design of Hybrid Systems (ADHS).

[12] S. Mitra, Y. Wang, N. Lynch & E. Feron (2003): *Safety verification of model helicopter controller using hybrid input/output automata*. In O. Maler & A. Pnueli, editors: Hybrid Systems: Computation and Control. Lecture Notes in Computer Science 2623, Springer-Verlag, Berlin, pp. 343–358, doi:10.1007/3-540-36580-X_26.

[13] C. Sonntag, R.R.H. Schiffelers, D.A. van Beek, J.E. Rooda & S. Engell (2009): *Modeling and Simulation using the Compositional Interchange Format for Hybrid Systems*. In: MATHMOD 2009 - 6th Vienna International Conference on Mathematical Modelling. pp. 640–650.

[14] P. Varaiya (1993): *Smart Cars on Smart Roads: Problems of Control*. IEEE Transactions on automatic control 38/2, pp. 195–207, doi:10.1109/9.250509.

[15] J.L.M. Vrancken, J.H. van Schuppen, M.S. Soares & F. Ottenhof (2009): *A hierarchical model and implementation architecture for road traffic control*. In: 2009 IEEE International Conference on Systems, Man and Cybernetics. doi:10.1109/ICSMC.2009.5346841.