World Automata: a compositional approach to model implicit communication in hierarchical Hybrid Systems. *

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. The new object, called World Automaton (WA), is built in such a way to preserve as much as possible of the compositional properties of HIOAs and its underlying theory. From the formal point of view we enrich classical HIOAs with a set of *world variables* whose values are functions both of time and space. World variables are treated similarly to local variables of HIOAs, except in parallel composition, where the perturbations produced by world variables are summed. In such way, we obtain a structure able to model both agents and environments, thus inducing a hierarchy in the model and leading to the introduction of a new operator. Indeed this operator, called inplacement, is needed to represent the possibility of an object (WA) of living inside another object/environment (WA).

1 Introduction

Agents moving in an environment need to communicate to achieve coordination for a common objective. Their communication method can be *explicit*, when they broadcast or send a signal to the other agents. This forces the introduction of a mechanism of declaration of intentions, for which either each agent communicates its position at fixed time instants, or a supervisor exists, that is able to know the topology of the network of agents. They can also communicate in an *implicit* way, i.e. using their *senses* to catch perturbations of the environment due to other agents. Implicit communication may be used in case of presence of noise and environmental hostilities, that prevent direct communication, as well as in case of necessity of not being intercepted or of being subject to faults and failures affecting the sender or the receiver. Using implicit communication, the agents do not need to broadcast their position, because they can *feel* the presence of other agents, or objects in general, avoiding collision. Moreover, implicit communication is also used to catch radio (or similar) signals that involve perturbation of the environment via sound waves embedding messages. Implicit communication does not necessarily substitute direct communication, but can be used as a redundant and faster way of communication in case of immediate response to an environment perturbation.

To face the problem of implicit communication, in [2] we specialized some variables of the Hybrid I/O Automata (HIOAs) of [7] and called them *world variables*. This modification has been motivated by the case studies of the European Project CON4COORD (EU FP7 223844): agents performing a *search* mission, such as UAVs [5] or autonomous underwater vehicles [3], but also of road traffic problems [11, 12] and autonomous straddle carriers in harbours [8]. Indeed what is common to each case study

*This research has been supported by EC-Project C4C (*Control for Coordination of Distributed Systems*) and the coordination action EuroSurge (grant no. 288233) funded by the European Commission in the 7th EC framework program.

Bortolussi, Bujorianu, Pola (Eds.): HAS 2013
EPTCS 124, 2013, pp. 58–72, doi:10.4204/EPTCS.124.7

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. World variables are, hence, used to represent the changes of the environment as perceived by the agents, achieving implicit communication. The difference with the other variables of HIOAs is that world variables dynamics depend both on the time and the space, creating a sort of map of the surrounding world for the agents.

What was missing in [2] is the interaction between agents and environment. Indeed changes on the environment are caused by agents and the environment has to *refer* to the other agents these changes. By adopting the usual compositional rule to represent also the environment with the same model of agents, i.e. automata, it is possible to extend the world variables paradigm. In this paper we will then consider also the environment as a HIOA with world variables. This choice introduces a hierarchy of automata, that can be both and contemporarily agents living in an environment and environments for other agents.

We will call the extended HIOAs *World Automata* (WAs), with the double aim of stressing both the capability of representation of this framework (the world itself) and of representing the reality in the most natural way possible, without adding artificial machineries. We renamed the automata because, even if they are an extension of the HIOAs, they will need slightly different operators to prove compositionality results.

The focus of this paper will be on the hierarchical representation of WAs. Indeed, it implies the need to distinguish between the variables used by an automaton to communicate with the world in which it lives and the variables it uses to communicate with the world it creates. We could simply partition the sets of variables into variables used to communicate with the outside world and variables used to communicate with the inside world. Nevertheless, this method will hide the hierarchy of automata. On the contrary, we want to keep the hierarchy in order to be able to always retrieve automata at different levels of depth. For this reason we equip variables with levels and we assume that variables at different levels are distinct.

The introduction of levels slightly changes the parallel composition policy, since we want to compose automata without losing the original hierarchy of the components. We need to impose that variables at levels not supposed to interact are not synchronized: it suffices to use disjoint sets of world variables, renaming them when needed. Indeed parallel composition requires synchronization only at the same level. To describe the interaction of two automata at different levels we introduce a new operator called Inplacement, used to compose a WA inside another WA. Inplacement establishes the policy of communication between the environment a WA provides and the WAs living in this environment.

WAs have been used to model a real application in [8], where straddle carriers autonomously moving into a harbour have to follow some trajectories avoiding collisions. The interested reader can find in the above mentioned paper most of the modeling features described in this paper, including an example of radio communication.

At the best of our knowledge, none of the existing modeling frameworks can be used to formally represent both implicit communication and hierarchy, without flattening the representation. Some approaches have been used to write a language capable to represent dynamically changing systems. One has been introduced in [4] where dynamic networks of hybrid automata are studied. The introduced programming language focuses on dynamical interfaces. Another method has been presented in [10] 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. Nevertheless there is an ongoing effort to extend the current version of CIF to also include the language

generated by world automata, in order to have a tool for automatically implement systems where implicit communication and hierarchy is needed.

The paper is organized as follows: in Section 2 we introduce the modeling framework; in Section 3 parallel composition of WAs is described; in Section 4 the inplacement operation is introduced. The theory is illustrated throughout the paper with a very simple example.
Notice that all the results presented in the paper use the notation and follow the results of [7].

2 World automata

In [2] we extended the HIOA modeling framework of [7] by specializing some variables, called *world variables*. The main difference between world and standard 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. An automaton $\mathscr{A}$ 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 $\mathscr{A}$. Here we extend the concept of world variables with hierarchy that can be used to represent nested worlds. To preserve the hierarchy and the identity of each automaton inside a world, we introduce a level function described as $l : S \to \mathbb{N}$ and extracting the level of a variable in any set S. Basically, if we consider only variables of level 0, then we have an ordinary HIOA equipped with some input and output world variables that are used to interact with its external world. The world variables of level 1 describe the world provided by an automaton. The world variables of level 2 describe the world provided by automata of level 1 and so on. We will call the HIOA with world variables and levels: World Automaton (WA).

In the following we will assume an underlying topological space $\mathscr{M}$. Without loss of generality the reader can think at $\mathscr{M}$ as a metric space, e.g. $\mathbb{R}^3$.

Definition 1 World Automaton
A World Automaton (WA) $\mathscr{A}$ is a tuple

$$((U_w, X_w, Y_w), (U_a, X_a, Y_a), (I, H, O), Q, \Theta, D, \mathscr{T}, l)$$

where

- (U_w, X_w, Y_w) *are disjoint sets of* world input, inner, and output variables, *respectively. Let W denote the set $U_w \cup X_w \cup Y_w$ of* world variables.
- (U_a, X_a, Y_a) *are disjoint sets of* automaton input, inner, and output variables, *respectively. Let U, X, Y denote the sets $U_w \cup U_a, X_w \cup X_a, Y_w \cup Y_a$ of* input, inner, and output variables, *respectively, and 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.
- $\mathscr{T}$ *is a set of trajectories on V that satisfy the following axioms*

T1 (Prefix closure)
For every $\tau \in \mathscr{T}$ and every $\tau' \leq \tau$, $\tau' \in \mathscr{T}$.

T2 (Suffix closure)
For every $\tau \in \mathscr{T}$ and every $t \in dom(\tau)$, $\tau \trianglerighteq t \in \mathscr{T}$.

T3 (Concatenation closure)
Let $\tau_0, \tau_1, \tau_2, \ldots$ be a sequence of trajectories in $\mathscr{T}$ such that, for each nonfinal index i, τ_i is closed and $\tau_i.lstate = \tau_{i+1}.fstate$. Then $\tau_0 \frown \tau_1 \frown \tau_2 \cdots \in \mathscr{T}$.

- *$l : S \to \mathbb{N}$ is the* level function *extracting the level of a variable or action in any set S.*

In this description, like in the HIOA theory, variables (U_a, X_a, Y_a) are used by the automaton to communicate with other automata living in the same world. To keep the theory consistent with previous descriptions of automata, all the X variables represent persistent characteristics of the system. For the sake of simplicity we will not use the level function in the rest of the paper, but we will make the variable level explicit by writing $v[i]$ for the variable of level i and $S[i]$ for the variables in S whose level is i. Moreover we will consider only finite depth WAs in this paper. Note that $X_w[0] = \emptyset$, due to the definition of X variables above and to the fact that level 0 is meant to be the level of the outside world, i.e. the level in which the automaton lives.

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 $type(v)$. 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 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 [7].

For a set of objects S we will write $S[i, i+1]$ to indicate the objects of S at level i and $i+1$. We will also write $S[i, .]$ for objects of S at levels i and greater (i.e. deeper).
With some minor extensions [1], the results on semantics of HIOAs are still valid for WAs, because they have been designed to follow as much as possible the HIOA theory. Hence all the results on executions, traces and simulation presented in [7] are extended to WAs.

3 Parallel Composition

In our framework, Parallel Composition models the interaction and communication of two or more agents living in the same world, i.e. of two WAs at the same level, with the environment, i.e. the world outside. We extend by comparison the notion of parallel composition introduced in [2] for HIOAs with world variables, with the treatment of levels. First, we introduce compatibility conditions to prevent undesired interactions between different levels for the WAs that have to be composed.

Definition 2 *Two WAs $\mathscr{A}_1$ and $\mathscr{A}_2$ are compatible if*

1. $V_1[1,.] \cap V_2[1,.] = \emptyset$, $A_1[1,.] \cap A_2[1,.] = \emptyset$,
2. $(U_{w1} \cup U_{w2}) \cap (Y_{w1} \cup Y_{w2}) = \emptyset$.
3. $H_1 \cap A_2 = H_2 \cap A_1 = \emptyset$,
4. $X_1 \cap V_2 = X_2 \cap V_1 = \emptyset$,
5. $O_1 \cap O_2 = \emptyset$,
6. $Y_1 \cap Y_2 = \emptyset$.

The reader may notice that conditions 2 to 6 are the compatibility conditions for HIOAs with world variables of [2]. The only difference is given by the first condition that states that all inner levels (higher than 0) are disjoint, and therefore no communication can occur at such levels. This means that generated worlds are disjoint. Since by the first condition communication may occur only at level 0, all the other properties are interesting only for level 0, even if not specified. These conditions, when not satisfied by the WAs, can be obtained by changing variables names.

Definition 3 Parallel composition
If $\mathscr{A}_1, \mathscr{A}_2$ are two compatible WAs, then their composition $\mathscr{A}_1 \| \mathscr{A}_2$ is defined as the structure $\mathscr{A} = (U_w, X_w, Y_w, U_a, X_a, Y_a, I, H, O, Q, \Theta, D, \mathscr{T}, l)$ with:

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. $Q = \{\mathbf{x} \in \mathit{vals}(X) \mid \mathbf{x} \lceil X_1 \in Q_1 \wedge \mathbf{x} \lceil X_2 \in Q_2\}$
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. $\mathscr{T} = \{\tau \mid$ *there exists* $\tau_1 \in \mathscr{T}_1, \tau_2 \in \mathscr{T}_2$ *such that*
   $\tau \downarrow (V_i \setminus (Y_{w1} \cap Y_{w2})) = \tau_i \downarrow (V_i \setminus (Y_{w1} \cap Y_{w2})), i \in \{1,2\}$
   $\tau \downarrow (Y_{w1} \cap Y_{w2}) = \tau_1 \downarrow (Y_{w1} \cap Y_{w2}) + \tau_2 \downarrow (Y_{w1} \cap Y_{w2})\}$
8. $l(v) = l_1(v)$ *if* $v \in V_1$, $l(v) = l_2(v)$ *if* $v \in V_2$.

This definition of parallel composition differs from the one of HIOAs with world variables of [2] only by the last condition: it preserves levels of the variables in composed automata $\mathscr{A}_1, \mathscr{A}_2$. The following result on composability is proved:

Proposition 1 *The composition of two WAs is a WA.*

Proof of Theorem 1 and all the results on parallel composition reported with their proofs in [2] are still valid for WAs, because the introduction of levels does not affect parallel composition, due to the compatibility conditions.

4 Inplacement operator

The main difference with the HIOA theory and the framework presented in [2] is that WAs create a hierarchy of automata. To this end a second operator is introduced that represents the interaction of a WA $\mathscr{A}_2$ inside another WA $\mathscr{A}_1$ with the world created by $\mathscr{A}_1$. The result is equivalent to compose $\mathscr{A}_2$ with the automata of level 1 of $\mathscr{A}_1$, although there are some important differences. This operation shows how the hierarchical communication works between the automata inside a world and the automaton representing their external world. Note that with this operator we do not want to describe the action of a WA *moving* inside another WA, but we want to describe the static behavior of an automaton *inside* another.

Notation: For a set S of objects we write $S\uparrow$ for the set obtained from S by increasing by 1 the level of each object.

Definition 4 *An automaton $\mathscr{A}_2$ is inplace compatible with $\mathscr{A}_1$ if, letting $\mathscr{A}_3$ be $\mathscr{A}_2\uparrow$,*

1. $V_1[2,.] \cap V_3[2,.] = \emptyset$, $A_1[2,.] \cap A_3[2,.] = \emptyset$,
2. $Y_{w1} \cap Y_{w3} = \emptyset$,
3. $H_1 \cap A_3 = H_3 \cap A_1 = \emptyset$,
4. $X_1 \cap V_3 = X_3 \cap V_1 = \emptyset$.
5. $O_1 \cap O_3 = \emptyset$,
6. $Y_1 \cap Y_3 = \emptyset$.

As for parallel composition, when using the *inplacement* operator the two composing automata need to have the disjoint sets of variables at levels deeper than 1. No compatibility condition is stated for level 0 because only the outside automaton $\mathscr{A}_1$ has this level: indeed the level of $\mathscr{A}_2$ is increased to be composed with automata of level 1 of $\mathscr{A}_1$. Disjointness of variables is again achieved by renaming when necessary. Since the world variables of level 1 are used to let $\mathscr{A}_2$ communicate with the world provided by $\mathscr{A}_1$, there should be no conflicts on outputs (condition 2).

Definition 5 *The inplacement of WA $\mathscr{A}_2$ inside $\mathscr{A}_1$, denoted by $\mathscr{A}_1[\mathscr{A}_2]$, is a system $\mathscr{A} = (U_w, X_w, Y_w, U_a, X_a, Y_a, I, H, O, Q, \Theta, D, \mathscr{T}, l)$ where, letting $\mathscr{A}_3$ be $\mathscr{A}_2\uparrow$,*

1. $U_w[0,1] = U_{w1}[0,1]$, $X_w[0,1] = X_{w1}[0,1]$, $Y_w[0,1] = Y_{w1}[0,1]$
2. $U_w[2,.] = (U_{w1} \cup U_{w3})[2,.]$, $X_w[2,.] = (X_{w1} \cup X_{w3})[2,.]$, $Y_w[2,.] = (Y_{w1} \cup Y_{w3})[2,.]$
3. $Y_a = Y_{a1} \cup Y_{a3}$, $U_a = (U_{a1} \cup U_{a3}) \setminus Y_a$ *and* $X_a = X_{a1} \cup X_{a3}$
4. $O = O_1 \cup O_3$, $I = (I_1 \cup I_3) \setminus O$ *and* $H = H_1 \cup H_3$
5. $Q = \{\mathbf{x} \in vals(X) \mid \mathbf{x}\lceil X_1 \in Q_1 \wedge \mathbf{x}\lceil X_3 \in Q_3\}$
6. $\Theta = \{\mathbf{x} \in Q \mid \mathbf{x}\lceil X_1 \in \Theta_1 \wedge \mathbf{x}\lceil X_3 \in \Theta_3\}$
7. $D = \{(\mathbf{x}, a, \mathbf{x}') \mid$ *for each* $i \in \{1,3\}$
   either $a \in A_i$ *and* $\mathbf{x}\lceil X_i \xrightarrow{a} \mathbf{x}'\lceil X_i$,
   or $a \notin A_i$ *and* $\mathbf{x}\lceil X_i = \mathbf{x}'\lceil X_i\}$.
8. $\mathscr{T} = \{\tau \mid$ *there exists* $\tau_1 \in \mathscr{T}_1, \tau_3 \in \mathscr{T}_3$ *such that*
   $\tau \downarrow (V_i \setminus (U_{w1} \cap Y_{w3})) = \tau_i \downarrow (V_i \setminus (U_{w1} \cap Y_{w3}), i \in \{1,3\}$.
   $\tau_1 \downarrow (U_{w1} \cap Y_{w3}) = \tau \downarrow (U_{w1} \cap Y_{w3}) + \tau_3 \downarrow (U_{w1} \cap Y_{w3})\}$.
   For each $u \in U_{w3} \setminus V$, *each* $t \in dom(\tau_3)$, $\tau_3(u)(t) = 0$.

9. $l(v) = l_1(v)$ *if* $v \in V_1$, $l(v) = l_3(v)$ *if* $v \in V_3$.

Note that the set of world variables of levels 0 and 1 are taken only from $\mathscr{A}_1$. For level 0 this definition derives from the fact that $\mathscr{A}_3$ has no objects at level 0; for level 1 it derives from the fact that $\mathscr{A}_3$ has no internal world variables at level 1 and that the world variables of level 1 of $\mathscr{A}_3$ that are not captured by $\mathscr{A}_1$ are no longer necessary since the environment does not use them. This is also expressed by the last condition in the definition of the set of trajectories, which makes sure that the world input of $\mathscr{A}_3$ that is not captured by $\mathscr{A}_1$ is always 0. This last condition is used to avoid the risk that, composing $\mathscr{A}_1$ with other WAs, the input of its inside automata not captured by $\mathscr{A}_1$ are then captured by the other composing WAs. Indeed, local worlds of automata have to be kept separated in parallel composition. Note also that, since from level 2 on, $V_1 \cap V_3 = \emptyset$, and $\mathscr{A}_3$ has no level 0, we have that $U_{w1} \cap Y_{w3}$ might be $\neq \emptyset$ only at level 1.
The main difference with parallel composition is given by the condition on trajectories. If we consider the domain of trajectories as a group, we have that this last condition can be expressed as $\tau \downarrow (U_{w1} \cap Y_{w3}) = \tau_1 \downarrow (U_{w1} \cap Y_{w3}) - \tau_3 \downarrow (U_{w1} \cap Y_{w3})$. This statement derives from the consideration that $\mathscr{A}_1$ is supposed to have some input world variables whose values take into account the possibility to have other automata inside. The object resulting from the inplacement composition has, for the same variables, values resulting from the difference between the input signal of the outside automaton and the output signals of local automata. With the reverse reasoning $\mathscr{A}_1$ will have some input world variables whose values are the result of the sum of values of the input world variables of the object $\mathscr{A}_1[\mathscr{A}_2]$ and the values of the output world variables of $\mathscr{A}_2$ communicating with them. In an abstract way we can think of $\mathscr{A}_1[\mathscr{A}_2]$ as an object already containing other WAs, so that the local input world variables are updated decreasing their values each time another WA is composed with the already present WAs of the local world.

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

Lemma 1 *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 2 *Let τ be a trajectory in V. Let $V' \subseteq V$. Then $(\tau \trianglerighteq t) \downarrow V' = (\tau \downarrow V') \trianglerighteq t$.*

Lemma 3 *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 $(\tau_0 \frown \tau_1 \frown \tau_2 \frown \ldots) \downarrow V' = (\tau_0 \downarrow V') \frown (\tau_1 \downarrow V') \frown (\tau_2 \downarrow V') \frown \ldots$.*

The proofs of these results are reported in Section 2 of [1].

Without loss of generality in the following we will assume that the domain of trajectories is a group. The results we are now going to introduce can be proved also using a monoid. We use the group to be coherent with the results of parallel composition.

Proposition 2 *The inplacement of WA $\mathscr{A}_2$ inside WA $\mathscr{A}_1$ is a WA.*

Proof: We show that $\mathscr{A}_1[\mathscr{A}_2]$ satisfies the properties of a WA. Again we let $\mathscr{A}_3$ be $\mathscr{A}_2 \uparrow$. Disjointness of the U, X, Y sets follows from disjointness of the same sets in $\mathscr{A}_1$ and $\mathscr{A}_3$ and compatibility. Similarly for the actions. Nonemptiness of starting state follows from nonemptiness of starting states of $\mathscr{A}_1$ and $\mathscr{A}_3$ and disjointness of X_1 and X_3. We verify the **T** properties of trajectories. Let C_{13} be $U_{w1} \cap Y_{w3}$.

T1 We want to prove that for every $\tau \in \mathscr{T}$ and every $\tau' \leq \tau$, $\tau' \in \mathscr{T}$. Let τ be a trajectory in $\mathscr{T}$. Let $i \in \{1,3\}$. By the definition of inplacement there exists $\tau_1 \in \mathscr{T}_1, \tau_3 \in \mathscr{T}_3$ such that $\tau \downarrow (V_i \setminus C_{13}) = \tau_i \downarrow (V_i \setminus C_{13})$, $\tau \downarrow C_{13} = \tau_1 \downarrow C_{13} - \tau_3 \downarrow C_{13}$. 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 $\tau' \downarrow (V_i \setminus C_{13}) = (\tau \lceil I) \downarrow (V_i \setminus C_{13})$. By lemma 1 $(\tau \lceil I) \downarrow (V_i \setminus C_{13}) = (\tau \downarrow (V_i \setminus C_{13})) \lceil I$. By definition of parallel composition and again by lemma 1 $(\tau \downarrow (V_i \setminus C_{13})) \lceil I = (\tau_i \downarrow (V_i \setminus C_{13})) \lceil I = (\tau_i \lceil I) \downarrow (V_i \setminus C_{13})$. Let $\tau'_1 = \tau_1 \lceil I$ and $\tau'_3 = \tau_3 \lceil I$, then

$(\tau_i \lceil I) \downarrow (V_i \setminus C_{13}) = \tau_i' \downarrow (V_i \setminus C_{13})$. Analogously, for the second statement of parallel composition of trajectories we have that $\tau' \downarrow C_{13} = (\tau \lceil I) \downarrow C_{13} = (\tau \downarrow C_{13}) \lceil I = (\tau_1 \downarrow C_{13}) \lceil I - (\tau_3 \downarrow C_{13}) \lceil I = (\tau_1 \lceil I) \downarrow C_{13} - (\tau_3 \lceil I) \downarrow C_{13} = \tau_1' \downarrow C_{13} - \tau_3' \downarrow C_{13}$. Hence $\tau' \in \mathscr{T}$.

T2 We want to prove that for every $\tau \in \mathscr{T}$ and every $t \in dom(\tau)$, $\tau \trianglerighteq t \in \mathscr{T}$. Let τ be a trajectory in $\mathscr{T}$. Let $i \in \{1,3\}$. By the definition of inplacement there exists $\tau_1 \in \mathscr{T}_1, \tau_3 \in \mathscr{T}_3$ such that $\tau \downarrow (V_i \setminus C_{13}) = \tau_i \downarrow (V_i \setminus C_{13})$, $\tau \downarrow C_{13} = \tau_1 \downarrow C_{13} - \tau_3 \downarrow C_{13}$. Hence, since $dom(\tau_1) = dom(\tau_3) = dom(\tau)$ and by lemma 2 we have that $(\tau \trianglerighteq t) \downarrow (V_i \setminus C_{13}) = (\tau \downarrow (V_i \setminus C_{13})) \trianglerighteq t = (\tau_i \downarrow (V_i \setminus C_{13})) \trianglerighteq t = (\tau_i \trianglerighteq t) \downarrow (V_i \setminus C_{13})$. Moreover $(\tau \trianglerighteq t) \downarrow C_{13} = (\tau \downarrow C_{13}) \trianglerighteq t = (\tau_1 \downarrow C_{13}) \trianglerighteq t - (\tau_3 \downarrow C_{13}) \trianglerighteq t = (\tau_1 \trianglerighteq t) \downarrow C_{13} - (\tau_3 \trianglerighteq t) \downarrow C_{13}$. By axiom **T2** applied to $\mathscr{A}_1$ and $\mathscr{A}_3$, for each $t \in dom(\tau_1)$, $\tau_1 \trianglerighteq t \in \mathscr{T}_1$ and for each $t \in dom(\tau_3)$, $\tau_3 \trianglerighteq t \in \mathscr{T}_3$. Hence $\tau \trianglerighteq t \in \mathscr{T}$.

T3 We want to prove that set $\mathscr{T}$ is closed under concatenation. Let $\tau_0, \tau_1, \tau_2, \ldots$ be a sequence of trajectories in $\mathscr{T}$, 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 $i \in \{1,3\}$. By definition of inside operator for each τ_j, $\exists \tau_{1j}, \tau_{3j}$ such that $\tau_j \downarrow (V_i \setminus C_{13}) = \tau_{ij} \downarrow (V_i \setminus C_{13})$, $\tau_j \downarrow C_{13} = \tau_{1j} \downarrow C_{13} - \tau_{3j} \downarrow C_{13}$. Let τ_i be $\tau_{i0} \frown \tau_{i1} \frown \tau_{i2} \frown \ldots, i \in \{1,3\}$. Hence by lemma 3 $\tau \downarrow (V_i \setminus C_{13}) = (\tau_0 \downarrow (V_i \setminus C_{13})) \frown (\tau_1 \downarrow (V_i \setminus C_{13})) \frown (\tau_2 \downarrow (V_i \setminus C_{13})) \frown \ldots = (\tau_{i0} \downarrow (V_i \setminus C_{13})) \frown (\tau_{i1} \downarrow (V_i \setminus C_{13})) \frown (\tau_{i2} \downarrow (V_i \setminus C_{13})) \frown \ldots = (\tau_{i0} \frown \tau_{i1} \frown \tau_{i2} \frown \ldots) \downarrow (V_i \setminus C_{13}) = \tau_i \downarrow (V_i \setminus C_{13})$. Moreover $\tau \downarrow C_{13} = (\tau_0 \downarrow C_{13}) \frown (\tau_1 \downarrow C_{13}) \frown (\tau_2 \downarrow C_{13}) \frown \ldots = (\tau_{10} \downarrow C_{13} - \tau_{30} \downarrow C_{13}) \frown (\tau_{11} \downarrow C_{13} - \tau_{31} \downarrow C_{13}) \frown (\tau_{12} \downarrow C_{13} - \tau_{32} \downarrow C_{13}) \frown \ldots = ((\tau_{10} \downarrow C_{13}) \frown (\tau_{11} \downarrow C_{13}) \frown (\tau_{12} \downarrow C_{13}) \frown \ldots) - ((\tau_{30} \downarrow C_{13}) \frown (\tau_{31} \downarrow C_{13}) \frown (\tau_{32} \downarrow C_{13}) \frown \ldots) = (\tau_{10} \frown \tau_{11} \frown \tau_{12} \frown \ldots) \downarrow C_{13} - (\tau_{30} \frown \tau_{31} \frown \tau_{32} \frown \ldots) \downarrow C_{13} = \tau_1 \downarrow C_{13} - \tau_3 \downarrow C_{13}$. Hence $\tau \in \mathscr{T}$.

■

Notation: Executions of WAs are defined as executions of HIOAs: an *execution fragment* of a WA $\mathscr{A}$ is an (A,V)-sequence $\alpha = \tau_0 a_1 \tau_1 a_2 \tau_2 \ldots$, where $a_i \in A$, $\tau_i \in \mathscr{T}$; 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 WAs. A *trace* of an execution fragment α captures the external behavior of $\mathscr{A}$, 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 WA $\mathscr{A}$ is, then, the (E,Z)-restriction of α. We will call trace the (E,Z)-restriction of α at all levels, supposing that the external behavior is captured at all levels of the automaton. When needed, it is possible to restrict the behavior to a specific level i by defining the $(E[i],Z[i])$-restriction of α, called $[i]$-trace.

Definition 6 *Automata $\mathscr{A}_1$ and $\mathscr{A}_2$ are* comparable *if they have the same external interface, that is, if world and local input and output sets of variables of $\mathscr{A}_1$ are equal to the corresponding sets of $\mathscr{A}_2$ and $E_1 = E_2$ at all levels. If $\mathscr{A}_1$ and $\mathscr{A}_2$ are comparable then we say that $\mathscr{A}_1$* implements *$\mathscr{A}_2$, denoted by $\mathscr{A}_1 \leq \mathscr{A}_2$, if $traces(\mathscr{A}_1) \subseteq traces(\mathscr{A}_2)$.*

In the following we state the results analogous to lemma 9 and proposition 2 of [2] for inplacement. Note that the proofs of these results are analogous to the ones for the corresponding results in parallel composition (see also [7] and [1]).

Lemma 4 *Let $\mathscr{A} = \mathscr{A}_1[\mathscr{A}_2]$, let α be an execution fragment of $\mathscr{A}$ and let $\mathscr{A}_3$ be $\mathscr{A}_2 \uparrow$. Then $\exists \alpha_1, \alpha_3$ execution fragments of $\mathscr{A}_1$ and $\mathscr{A}_3$ respectively, such that*

1. $\alpha \lceil (A_i, V_i \setminus C_{13}) = \alpha_i \lceil (A_i, V_i \setminus C_{13}), i = 1,3$, *and*
2. $\alpha \lceil (\emptyset, C_{13}) = \alpha_1 \lceil (\emptyset, C_{13}) - \alpha_3 \lceil (\emptyset, C_{13})$,

with $C_{13} = U_{w1} \cap Y_{w3}$.

Proposition 3 *Let $\mathscr{A} = \mathscr{A}_1[\mathscr{A}_2]$, let β be a trace of $\mathscr{A}$ and let $\mathscr{A}_3$ be $\mathscr{A}_2\uparrow$. Then $\exists\beta_1,\beta_3$ traces of $\mathscr{A}_1,\mathscr{A}_3$ respectively, such that*

1. $\beta\lceil(E_i,Z_i\setminus C_{13}) = \beta_i\lceil(E_i,Z_i\setminus C_{13}), i = 1,3$ *and*
2. $\beta\lceil(\emptyset,C_{13}) = \beta_1\lceil(\emptyset,C_{13}) - \beta_3\lceil(\emptyset,C_{13})$,

with $C_{13} = U_{w1}\cap Y_{w3}$.

The following theorems state the substitutivity properties of inplacement for implementation.

Theorem 1 *Let $\mathscr{A}_1$ and $\mathscr{A}_2$ be comparable WAs with $\mathscr{A}_1 \leq \mathscr{A}_2$. Let $\mathscr{B}$ be a WA compatible with each of $\mathscr{A}_1$ and $\mathscr{A}_2$. Then $\mathscr{A}_1[\mathscr{B}]$ and $\mathscr{A}_2[\mathscr{B}]$ are comparable and $\mathscr{A}_1[\mathscr{B}] \leq \mathscr{A}_2[\mathscr{B}]$.*

Proof: Let α be an execution of $\mathscr{A}_1[\mathscr{B}]$. By lemma 4, two executions α_1,α_B exist, such that $\alpha_1 \in execs(\mathscr{A}_1)$, $\alpha_B \in execs(\mathscr{B}\uparrow)$ 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 $C_{1B} = U_{w1}\cap Y_{wB}$. By lemma 7 we can take paddings of α,α_1,α_B such that the j^{th} trajectory has the same length for all j. Let these paddings be γ,γ_1,γ_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 $\mathscr{A}_1 \leq \mathscr{A}_2$ and by compatibility, we can find an execution α_2 of $\mathscr{A}_2$ with the same trace of α_1 and a padding of α_2 following lemma 7. 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 $\mathscr{A}_2[\mathscr{B}]$ 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 $C_{2B} = U_{w2}\cap Y_{wB}$. 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 WA. Actions b_i might be different, but by construction, compatibility and lemma 6 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 influences the traces (external variables). For this reason we can state that $traces(\mathscr{A}_1[\mathscr{B}]) \subseteq traces(\mathscr{A}_2[\mathscr{B}])$, hence, by definition of implementation, $\mathscr{A}_1[\mathscr{B}] \leq \mathscr{A}_2[\mathscr{B}]$. ■

Lemma 5 *Let $\mathscr{B}_1$ and $\mathscr{B}_2$ be comparable WAs with $\mathscr{B}_1 \leq \mathscr{B}_2$. Then $(\mathscr{B}_1\uparrow) \leq (\mathscr{B}_2\uparrow)$.*

Proof: Proved by the definitions of executions, traces and $\uparrow$ operator. ■

Theorem 2 *Let $\mathscr{B}_1$ and $\mathscr{B}_2$ be comparable WAs with $\mathscr{B}_1 \leq \mathscr{B}_2$. Let $\mathscr{A}$ be a WA compatible with each of $\mathscr{B}_1$ and $\mathscr{B}_2$. Then $\mathscr{A}[\mathscr{B}_1]$ and $\mathscr{A}[\mathscr{B}_2]$ are comparable and $\mathscr{A}[\mathscr{B}_1] \leq \mathscr{A}[\mathscr{B}_2]$.*

Proof: Let α be an execution of $\mathscr{A}[\mathscr{B}_1]$. By lemma 4, two executions α_A,α_1 exist, such that $\alpha_A \in execs(\mathscr{A})$, $\alpha_1 \in execs(\mathscr{B}_1\uparrow)$ and: $\alpha\lceil(A_A,V_A\setminus C_{A1}) = \alpha_A\lceil(A_A,V_A\setminus C_{A1})$, $\alpha\lceil(A_1,V_1\setminus C_{A1}) = \alpha_1\lceil(A_1,V_1\setminus C_{A1})$, $\alpha\lceil(\emptyset,C_{A1}) = \alpha_A\lceil(\emptyset,C_{A1}) - \alpha_1\lceil(\emptyset,C_{A1})$, with $C_{A1} = U_{wA}\cap Y_{w1}$. By lemma 7 we can take paddings of α,α_A,α_1 such that the j^{th} trajectory has the same length for all j. Let these paddings be γ,γ_A,γ_1 respectively with $\gamma = \tau_0 a_1\tau_1 a_2\tau_2 a_3\ldots$, $\gamma_A = \tau_{0A}a'_1\tau_{1A}a'_2\tau_{2A}a'_3\ldots$ and $\gamma_1 = \tau_{01}a''_1\tau_{11}a''_2\tau_{21}a''_3\ldots$. Since $\mathscr{B}_1 \leq \mathscr{B}_2$ (hence $(\mathscr{B}_1\uparrow) \leq (\mathscr{B}_2\uparrow)$ by lemma 5) and by compatibility, we can find an execution α_2 of $\mathscr{B}_2$ with the same trace of α_1 and a padding of α_2 following lemma 7. 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 $\mathscr{A}[\mathscr{B}_2]$ obtained by γ_A and γ_2 will be $\gamma' = \tau'_0 b_1\tau'_1 b_2\tau'_2 b_3\ldots$, where $\tau'_j\downarrow(V_A\setminus C_{A2}) = \tau_{jA}\downarrow(V_A\setminus C_{A2})$, $\tau'_j\downarrow(V_2\setminus C_{A2}) = \tau_{j2}\downarrow(V_2\setminus C_{A2})$, $\tau'_j\downarrow C_{A2} = \tau_{jA}\downarrow C_{A2} - \tau_{j2}\downarrow C_{A2}\}$, where $C_{A2} = U_{wA}\cap Y_{w2}$. 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 WA. Actions b_i might be different, but by construction, compatibility and lemma 6 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 influences the traces (external variables). For this reason we can state that $traces(\mathscr{A}[\mathscr{B}_1]) \subseteq traces(\mathscr{A}[\mathscr{B}_2])$, hence, by definition of implementation, $\mathscr{A}[\mathscr{B}_1] \leq \mathscr{A}[\mathscr{B}_2]$. ■

Notation for proofs: The interested reader can find all the definitions and notation in [2, 1].

Definition 7 *A* padded execution *of a WA $\mathscr{A}$ 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 8 Padding.
We call padding *of an execution α any padded execution obtained by α by extending the actions set with ε.*

Lemma 6 *Let α be an execution of $\mathscr{A}$ and γ a padding of α, then $trace(\alpha) = trace(\gamma)$.*

Lemma 7 *Given n executions, it is always possible to find n paddings of these executions such that all corresponding trajectories have the same length.*

5 Case Study

In this section we use the above introduced theory to describe a scenario based on the work in [5]. We consider an environment, later called field. On this field there are M targets, of which $M_k \subset M$ are known. Over the field n UAVs (Unmanned Aerial Vehicles) fly, which are supposed to detect all the targets and engage them. We are not interested in the cooperative search policy, which is described in [5]. We want to prove, using this example, that our formal model is reliable in describing the characteristics of such scenario, preserving some results of composability. The WAs reported here are not the only ones that can be used to represent the same scenario of course, we used them because they are the most straightforward to design. To represent WAs we use a variant of the TIOA language (see [6]), with some extensions for hybrid systems ([9]) and some new tools for 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. We assume an underlying metric space $\mathbb{R}^2$, where points are indicated by $p \in \mathbb{R}^2$. Obviously p is given by a couple of coordinates (x,y). We do not use $\mathbb{R}^3$ because we are interested in the position of the targets on the field and in the projection of the position of the UAVs on the field, not in their altitude. We also assume that the area occupied by the field is the entire space $\mathbb{R}^2$.

Since the description of the field characteristics is very dependent from the nature of the targets inside it, we start by describing targets represented in fig. 1. Usually what determines the interaction with the ground is the class of the target. For example if we are in an environment surveillance scenario, some classes might be: fire, flood, steam, etc. A very general parameter might be the color: if a UAV is flying over a certain field, which we suppose to be green, and we see a black spot, then we conclude that something wrong is happening, hence a black target might be there.

The local internal variables of a Target are: position p_T, orientation ϕ, a Fail signal used by the target to *delete* itself when engaged. The world input is compression k, given by the ground when the target is engaged. For example, if the target is a fire and the UAV is pouring some water, the field will react by getting colder and change the pressure in the meanwhile, until the fire is put off. We will generally call compression the reaction of the ground to the engage signal. The target world output is its color ξ. Each target has an ID given by its parameter IDT and a class given by its color T_C, which is still a parameter. A function f is defined for targets, giving the surface of the field occupied by the target area. We do not specify this function because it is irrelevant to the scope of this paper, but it is calculated starting from the target position p_T and its orientation angle ϕ. The target has an internal action *delete* occurring when the field compression reaches a certain value k_{max}. Its effect is to put the Fail signal on and then change the color of the target location to the color of the field, which we assume to be green. Note that the Target has two states: in one it is alive and its Fail signal is false, in the other one it is deleted and its Fail signal is true. The switching between the states occurs when the action *delete* arises.

The Field WA is represented in fig. 2. The world internal variables of Field are color C and compression K. The world input variables are: color of the target ξ and the engage signal e from UAVs. The world output variables refer compression k and color c. The Field has only one state in which its color is a replica of the color variable given by the targets, its compression increases when the engage signal is activated and its outputs are used to communicate with targets (compression) and UAVs (color).

UAVs have a more complex structure represented in fig. 3. Each UAV is specified by an ID (IDU) and a color (T_U) associated with the kind of target it can engage. Note that not all UAVs can engage all targets, but only the ones with the same color. Again we are not going into the detail of levels greater than 0. UAVs local internal variables are: position p_U, speed v, heading angle ψ, performing task tk, actual assignment assign, position given by the sensor footprint s_{fp}. The UAV can perform one of the following tasks: search (S), confirm (C), engage (E). The search task is the default one: the UAV moves following a constant trajectory and always searching for targets. When it sees with a certain probability a target (color different from green) it updates a map of targets, and its task goes to confirm. If the target is of the same color of the one given by T_U, then the UAV can compete for engage it. If it wins the competition its task moves to engage. Similarly for the assignments: a free assignment means that the UAV has not been associated to any target, i.e. it has not been elected to engage a target, but it is not even competing to engage a target; a competing assignment means that a UAV is competing with other UAVs to engage a certain target; a committed assignment means that the UAV has been elected to engage the target. The sensor footprint gives the position in which the UAV is looking for a target. The local input variables are: probability Pin_t of presence of a target broadcasted by the UAVs, probability Pin_ϕ of orientation of a target broadcasted by the UAVs, cost_{in} array of costs for reaching a target given by the other competing UAVs. The local output variables are: probability P_t of presence of a target, probability P_ϕ of orientation of a target, cost of reaching the target for which the UAV is competing; map of target presence probability for each position m_p, map of task for each position m_t, map of target kind for each position m_k, map of target orientation estimate for each position m_ϕ, map of target assignment m_a. All the maps are broadcasted and updates using a function *update* which puts a value on a specific position of the map. We are not going into the detail neither of the map model, nor of the update function. The UAV world input is the color of the field c, its world output is the engage signal e.

type Rad = $\mathbb{R}|2\pi$
type Color = {green, χ_1, χ_2, χ_3}

worldautomaton Target (IDT: Real,T_C: Color)

world variables LEVEL 0
  input k: Real
  output ξ: Color
local variables LEVEL 0
  internal ϕ: Rad, p_T: Real^2,
    Fail: Bool := false
actions
  internal *delete*
transitions
  internal *delete*
  pre $\exists p \in f(\phi, p_T)$ s.t. $k(t,p) \geq k_{max}$
  eff Fail = true
trajectories
  $\xi(t,p) = \begin{cases} T_C & \text{if } p \in f(\phi,p_T) \wedge \neg\text{Fail} \\ \text{green} & \text{otherwise} \end{cases}$

Figure 1: Target world automaton.

We now want to put a Target inside Field. We verify that the inplace compatibility is valid. First of

worldautomaton Field

world variables LEVEL 1
internal C: Color, K: Real := 0
input ξ: Real, e: Bool
output k:Real, c: Color
trajectories
$C(t,p) = \xi(t,p)$;
$\dot{K}(t,p) = e(t,p)?1:0$;
$c(t,p) = C(t,p)$;
$k(t,p) = K(t,p)$.

Figure 2: Field world automaton.

type Task = $\{S,C,E\}$
type Assignment = {free, competing, committed}

worldautomaton UAV (IDU: Real, T_U: Color)

world variables LEVEL 0
input c: Color,
output e: Bool := false
local variables LEVEL 0
internal p_U: Real^2, v: Real^2, ψ: Rad,
tk: Task := S,
assign: Assignment := free,
s_{fp}: Real^2
input Pin_t: $\{0,1\}$, Pin_ϕ: $\{0,1\}$, cost$_{in}$: array of Real
output P_t: $\{0,1\}$, P_ϕ: $\{0,1\}$, cost: Real,
m_p: Map, m_t: Map, m_k: Map,
m_ϕ: Map, m_a: Map
actions
internal *search*, *confirm*, *engage*, *compete*, *commit*
transitions
internal *search*
pre $P_t(s_{fp},t) \leq p_s \vee \exists i$ s.t. cost$(t) \geq$ cost$^i_{in}(t)$
$\vee\ c(t,s_{fp}) =$ green
eff $tk = S$; assign = free;
internal *confirm*
pre $P_t(s_{fp},t) > p_s \wedge c(t,s_{fp}) \neq$ green
eff $tk = C$;
internal *compete*
pre $c(t,s_{fp}) = T_U$
eff assign = competing;
internal *commit*
pre cost$(t) =$ min$_i$ cost$^i_{in}(t)$
eff assign = committed;
internal *engage*
pre $P_t(s_{fp},t) > p_e$
eff $tk = E$;
trajectories
$p_U(t) = v(t)$;
$\dot{\psi} \leq \eta$;
$\dot{v} = 0$;
$e(t,p) = (tk = E)$? true: false;
$P_\phi(s_{fp},t) = g(P_\phi(s_{fp},t^-), Pin_\phi(s_{fp},t), c(t,s_{fp}), \psi(t))$;
$m_\phi(s_{fp},t) = update(P_\phi(s_{fp},t))$;
$P_t(s_{fp},t) = h(P_t(s_{fp},t^-), Pin_t(s_{fp},t), c(t,s_{fp}))$;
$m_p(s_{fp},t) = update(P_t(s_{fp},t))$;
cost$(t) = r(|P_U - s_{fp}|)$;
$m_k(s_{fp},t) = update(c(t,s_{fp}))$;
$m_t(s_{fp},t) = update$(task);
$m_a(s_{fp},t) =update$(assign);

Figure 3: UAV world automaton.

all we lift the level of Target (and hence of all its variables and actions) to 1. We did not define variables at levels greater than 1, hence the first condition for compatibility is valid. The second condition is on world outputs: $Y_F = \{k, c\}$, $Y_T = \{\xi\}$ implies that $Y_T \cap Y_F = \emptyset$. Since Field has no actions, the third condition is also verified. Field and Target have no output variables and no output actions, so the last two conditions for compatibility are also verified. The inplacement of Target in Field is the object described in fig. 4. The reader can notice that condition 9 of Def. 5 is respected, since world variable $\xi \in U_{w1} \cap Y_{w3}$.

worldautomaton Field[Target(IDT,T_C)]

world variables LEVEL 1
  internal C, $K := 0$
  input ξ, e
  output k, c
local variables LEVEL 1
  internal ϕ, p_T, Fail := false
actions
  internal *delete*
transitions
  internal *delete*
  pre $\exists p \in f(\phi, p_T)$ s.t. $k(t,p) \geq k_{max}$
  eff Fail = true
trajectories
  $C(t,p) = \neg\text{Fail}?\xi(t,p) : \text{green}$
  $\dot{K}(t,p) = e(t,p)?1 : 0;$
  $c(t,p) = C(t,p);$
  $k(t,p) = K(t,p).$

Figure 4: Field[Target]

worldautomaton FalseField

world variables LEVEL 1
  internal fC: Color, fK: Real := 0
  input e: Bool
  output c: Color, k: Real
local variables LEVEL 1
  internal θ: Rad, fp: Real, Del: Bool := false, χ: Color
actions
  internal *change*
transitions
  internal *change*
  pre $\exists p \in f(\theta, fp)$ s.t. $k(t,p) \geq k_{max}$
  eff Del = true
trajectories
  $fC(t,p) = (p \in f(\theta, fp) \wedge \neg\text{Del})?\chi : \text{green};$
  $f\dot{K}(t,p) = e(t,p)?1 : 0;$
  $c(t,p) = fC(t,p);$
  $k(t,p) = fK(t,p).$

Figure 5: FalseField

We now define system FalseField as in fig. 5. We want to prove that FalseField is equivalent to Field[Target(IDT,T_C)], i.e. a bisimulation exists between the two automata. Consider a state of Field[Target(IDT,T_C)] in which Fail=false ($k(t,p) < k_{max}, \forall p \in \mathbb{R}^2$), hence $C(t,p) = \xi(t,p)$, i.e. $C(t,p) = T_C, p \in f(\phi, p_T)$ and green everywhere else. Also in this state $e(t,p) = \text{false}, \forall p \in \mathbb{R}^2$, hence $\dot{K}(t,p) = 0$. The bisimilarity relation is the identity for all internal variables (both world and local). The corresponding state of FalseField is given by Del=false, hence $fC(t,p) = \chi, \forall p \in f(\theta, fp)$, where θ has the same value of ϕ, fp has the same value of p_T and χ has the same value of T_C. Since $e(t,p) = \text{false}, \forall p \in \mathbb{R}^2$,

then $f\dot{K}(t,p) = 0$. Hence if the color of Target is χ_2, i.e. $T_C = \chi = \chi_2$, the output variables of both WAs will be $c = \chi_2$ and $k =$ CONST. Now if $e(t,p) = 1$ for some p, then $\dot{K}(t,p) = 1$, i.e. K starts increasing, and so does k. After a certain time t it reaches the value of k_{max} and the action *delete* occurs. Hence in the new state Fail=true, i.e. $C(t,p) =$ green for $p \in f(\phi, p_T)$. Similarly, for FalseField, $e(t,p) = 1$ implies that $f\dot{K}(t,p) = 1$, i.e. fK starts increasing, and so does k. When k reaches k_{max}, action *change* occurs, and Del=true. In this new status $fC(t,p) =$ green for $p \in f(\theta, fp)$. In both cases $c =$ green.

What we want to prove now is that (Field[Target])[UAV] is equivalent to FalseField[UAV]. For lack of space these two automata are not represented here, but the reader can easily follow the inplacement procedure to design them. We start from the following state in Field[Target][UAV] at time t^*: Fail=false, $c(t*, p*) = T_U = \chi_2$ for a certain position $p* \in \mathbb{R}^2$, $s_{fp} = p*$, $tk = C$, assign=committed. The corresponding state of FalseField[UAV] is: Del=false, $c(t*, p*) = T_U = \chi_2$, $s_{fp} = p*$, $tk = C$, assign=committed. Hence equivalence by identity is preserved. At next instant of time, $t*^+$, $P_t(p*, t*^+) > p_e$, and action *engage* occurs with the effect of letting $tk = E$. As a consequence $e(t*^+, p*) =$ true. In (Field[Target])[UAV], this implies that $\dot{K}(t*^+, p*) = 1$. Output variable k will start increasing, following internal variable K until it reaches value k_{max} and activating action *delete* with the consequence of variable Fail becoming true and world output c becoming green. Analogously in FalseField[UAV] $f\dot{K}(t*^+, p*) = 1$ and output k starts increasing until it reaches value k_{max} such activating action *change* with the consequence of variable Del becoming true and world output c becoming green. The above execution represents the case in which a UAV i is associated with a certain target j having the same color, i.e. $T_C(j) = T_U(i)$. The UAV sees where the color is and activates the engage signal to delete target j, that after a certain time fails and sets the color in its location to green.

6 Concluding Remarks

In this paper we proposed an extension of the model introduced in [2] to provide an explicit and natural representation of the fact that objects move in a world that they can observe and modify. Besides the classical signals that automata send to each other via discrete communication events or shared continuous variables, the automata we introduced, called World Automata (WAs), can communicate implicitly by affecting their surrounding world and observing the effects on the world 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 the extended model. Moreover we represented hierarchical nested worlds, by introducing a notion of levels of variables. Indeed we extended the notion of parallel composition of [2] to WAs, keeping compositionality results. We introduced another operator, called inplacement, to represent the hierarchical composition of WAs. We presented in this paper an example to show the effectiveness of the theory, but another reality-based application can be found in [8]. 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. One approach for dynamic communication that we find promising is to associate each state and output action with an open set of the underlying space $\mathscr{M}$ and interact only with automata that lie in such open set, which we can call neighborhood. The natural extension to this problem is the ability of WAs to achieve selective communication, by choosing if they want to broadcast information to every other agent in the neighborhoods or only to some of them.

References

[1] 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.

[2] M. Capiluppi & R. Segala (2012): *Modelling Implicit Communication in Multi-Agent Systems with Hybrid Input/Output Automata*. In: *Third International Symposium on Games, Automata, Logics and Formal Verification (GandALF 2012)*. Electronic Proceedings in Theoretical Computer Science (EPTCS), doi:10.4204/EPTCS.96.1.

[3] 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*. doi:10.3182/20050703-6-CZ-1902.01960.

[4] 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.

[5] 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, doi:10.1109/TSMCB.2005.861881.

[6] 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.

[7] 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.

[8] 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)*. doi:10.3182/20120606-3-NL-3011.00035.

[9] 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.

[10] 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.

[11] 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.

[12] 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.