Available online at www.sciencedirect.com

ScienceDirect

Cognitive Systems Research 00 (2016) 1–30

Cognitive Systems Research

Autonomous construction and exploitation of a spatial memory by a self-motivated agent

Simon L. Gay[a,*], Alain Mille[a], Olivier L. Georgeon[a], Alain Dutech[b]

[a] *Université de Lyon, CNRS, LIRIS, UMR5205, F-69622, France*
[b] *Université de Lorraine, CNRS, LORIA, UMR7503, Vandœuvre-lès-Nancy, F-54506, France*

Abstract

We propose an architecture for self-motivated agents allowing them to construct their own knowledge of objects and of geometrical properties of space through interaction with their environment. Self-motivation is defined here as a tendency to experiment and to respond to behavioral opportunities afforded by the environment. Interactions have predefined valences that specify inborn behavioral preferences. The long-term goal is to design agents that construct their own knowledge of their environment through experience, rather than exploiting pre-coded knowledge. Over time, the agent learns relations between elements of the environment that afford its interactions, and its perception of these elements, in the form of data structures called signatures of interactions. These signatures allow the agent to attribute a low level semantics to elements that constitute its environment based on valences of interactions, without predefined knowledge about these elements and regardless of the number of element types. Signatures of interaction are then used to localize elements in space and to construct data structures that characterize spatial properties of space, called signatures of places and signatures of presence. Signatures of place and of presence characterize space using interactions rather than geometrical or topological properties. The agent uses these structures to maintain an egocentric representation of affordances of the surrounding environment, without any preconception about the elements that compose the environment, and without using notions of geometrical space. Experiments with simulated agents show that they learn to behave in their environment, taking into account multiple surrounding objects, reaching or avoiding objects according to the valence of the interactions that they afford.

Keywords:
affordance, learning (artificial intelligence), autonomous mental development, developmental learning, interactionism, intrinsic motivation, spatial awareness

1. Introduction

We propose a mechanism that allows an artificial agent to construct, interpret, and exploit a short-term memory of its surrounding environment, without using ontological preconception about its environment or its sensorimotor possibilities. The agent's purpose is to generate behaviors that satisfy its self-motivational principles. Such an agent can be defined as environment-agnostic (Georgeon & Sakellariou, 2012).

Utilization of a spatial memory must allow the agent to integrate the surrounding elements of its environment. In particular, the spatial memory must memorize elements that slip out of range of the agent's sensory system so the agent can keep track of them. It must also help the agent to discover spatial properties of its environment, by capturing spatial regularities offered by the environment. The spatial memory must thus construct a complete context that characterizes the agent's environment. The construction of such a spatial memory relates to the problem of space integration, which is faced by living beings as well as artificial agents.

We base our work on a design principle introduced by Georgeon & Aha (2013), called Radical Interactionism (RI). RI intends to account for cognitive theories that suggest that sensorimotor patterns of interaction are the primary bricks of cognition (e.g. Piaget (1954)). In the RI formalism, the agent is given a predefined set of *actions* $\mathcal{A}$ that the agent can perform in the environment, and a predefined set of *results* $\mathcal{R}$ that the agent gets from the environment as the result of an action. The agent's input data is called *result* rather than *observation* or *perception* because it does represent the state of the environment. In a given state of the environment, the agent may get a different result $r_t \in \mathcal{R}$ depending on the action $a_t \in \mathcal{A}$. On the contrary, if we used the terms *observation* or *perception*, the reader would expect that the agent's input data would depend on the state of the environment only. The terms *observation* and *perception*

*Corresponding author
Email address: simon.gay@liris.cnrs.fr (Simon L. Gay)

are used in many articles to mean exactly that. By using the term *result*, we wish to highlight this crucial difference from these articles.

The RI formalism defines an interaction i as a couple made of an action and a result: $i = \langle a, r \rangle \ / \ a \in \mathcal{A}, r \in \mathcal{R}, i \in I$, where I is the set of interactions, $I = \mathcal{A} \times \mathcal{R}$. Interactions represent Piagetians sensorimotor schemes and are the core elements of the RI model. An RI agent uses interactions as atomic elements.

The RI formalism defines the valence function $v : I \longrightarrow \mathbb{R}$ that associates a numerical valence with each interaction. We develop agents that seek to perform interactions that have a positive valence and to avoid performing interactions that have a negative valence. This motivational principle is called *interactional motivation* (Georgeon, Marshall, & Gay, 2012), and is related to the problem of intrinsic motivation (Oudeyer, Kaplan, & Hafner, 2007). The agent perceives its environment by identifying affordances in the environment rather than by recognizing objects on the basis of predefined features. This approach addresses the knowledge-grounding problem (Harnad, 1990) by letting knowledge of objects and spatial properties arise from experience of interaction, introducing no discontinuity between the agent's experience and the representation of objects and space.

1.1. How to integrate space management in an autonomous agent according to the RI paradigm?

Georgeon & Ritter (2011) have developed a sequential RI algorithm that allows an agent to autonomously capture and exploit sequential regularities of interaction offered by the coupling between the agent and the environment. This algorithm constructs hierarchical sequences of interactions that represent these regularities, starting from short sequences, and building increasingly longer sequences of sequences in a bottom-up fashion. However, this kind of agents had troubles organizing their behaviors in space and did not detect the persistence of objects. As an example, the agent could not detect that turning 90° left three times is equivalent to turning 90° right once. Moreover, the agent ceased to pursue objects when they slipped out of range of the agent's sensory system, even if the objects were just behind the agent.

To overcome these limitations, the challenge is to develop a mechanism that allows an artificial agent to construct, maintain and exploit a short-term internal model of the environmental context. Moreover, a second challenge is to be able to cope with objects that compose this environment in terms of interactions. All of this has to be obtained by and for generating behaviors that satisfy the agent's motivational principles.

In this context, we do not try to develop a path planning mechanism, neither are we designing a mapping algorithm. Instead, we decided to consider mechanisms inspired by simple living beings as a good starting point to take up these challenges.

1.2. Inspiration from biology

Learning to integrate and maintain a representation of the surrounding environment, and memorizing the position of elements present the environment, are vital abilities for many living beings, that help them to avoid dangerous areas or to move toward interesting objects. Considering artificial agents, such abilities help them to construct an internal model without pre-conceptions about the agent's sensorimotor properties. We drew inspiration from vertebrates, for which space is integrated in several brain areas, such as tectum (Northmore, 2011) (or colliculus in mammals), sensorimotor cortices (Graziano, Taylor, & Moore, 2002), or hippocampus (O'Keefe & Dostrovsky, 1971). These areas maintain a certain correspondence with spatial positions (Cotterill, 2001). Previc (1998) proposes a model of space divided into four areas. Each area handles an area of the surrounding space with a predefined purpose: the peripersonal space, that consists of the space the agent can directly reach; the action extrapersonal space, that consists of the space the agent can reach through movements in space; the focal extrapersonal space, which is the area around the point observed by the visual system (in living beings equipped with a *fovea*); and the ambient extrapersonal space, which takes into account the far environment, used as a reference for orientation in space.

We limit the work presented in this paper to the peripersonal and the extra-personal spaces. We believe that these two areas are necessary for rudimentary living beings to survive. Indeed, vision and space management of fish and reptiles is mainly based on the optic tectum and may not have very advanced brain structures related to space management, such as sensorimotor cortices and hippocampus, and few species have a *fovea*.

The peripersonal space can be considered as the space in which an agent can interact directly or after a short movement that can be considered as a part of the interaction. The extrapersonal space can be considered as the region of space (in egocentric reference) with which an agent can interact after a movement, limited to the region of space that the *space memory* can integrate. The extrapersonal space incorporates the peripersonal space. We can thus study the extrapersonal space without a peripersonal space integration mechanism. In a previous work, we proposed a mechanism that integrates peripersonal space (Gay & Georgeon, 2013). In this paper, we focus on mechanisms that integrate the extrapersonal space.

1.3. A spatial agnostic agent: which principles should be respected?

We have to define the set of principles we consider necessary for an environmentally agnostic agent to exhibit spatial behaviors that are adapted to an initially unknown environment. Exhibited behaviors have to respect five main principles:

Principle 1 (P1): the agent must be intrinsically motivated to ensure that its motivation principles do not rely on a direct access to environment properties or on any external influence.

Principle 2 (P2): the agent must consider objects that compose its environment by itself, based on its sensorimotor possibilities, rather than using a predefined set of object descriptors.

Principle 3 (P3): the agent must be able to integrate its surrounding environment in a form it can exploit. This property implies that the agent must be able to discover and integrate spatial properties of its environment and recognize previously defined objects in its environment.

Principle 4 (P4): the agent must be able to consider the permanence of objects, i.e. the agent must learn to build its memory in such a way that it can track object positions even when it cannot interact with them anymore.

Principle 5 (P5): the agent must exploit the structures it constructs and its internal model of its environment to generate behaviors that satisfy its motivational principles.

This paper is divided in three parts: the first part gives a state of the art of the problem of space integration and generation of behavior based on experience, and summarizes our previous work (Section 2). The second part details the mechanisms we propose to allow an artificial agent to integrate its surrounding environment without preconception about the environment (Sections 3 and 4). We use the agent employed in our experiments to illustrate principles and mechanisms of our system. The third part presents the experiments we provided to test our mechanisms (Section 5).

2. State of the art

The five principles, as defined in the previous section, are well studied in the literature. However, these properties were studied separately. The sensorimotor approach requires that these principles must be respected simultaneously to generate behaviors based on a complete sensorimotor loop.

The following state of the art lists representative works that exhibit the right properties to respect one or more principles. The different contributions are presented with corresponding studied principles (in parentheses).

2.1. Building knowledge from experience

Theories of cognition (e.g. Piaget (1954), O'Regan (2011)) assume that the knowledge of our world arises from our interaction with this world. We consider this property to be fundamental as it ensures that an agent experiences its environment through its possibilities of interaction rather than accessing different *states* of the environment, which would be completely independent from it. This section lists relevant works in the domain of the developmental approach for artificial intelligence (*Developmental AI* in short) and the emergence of the notion of object based on interactional experience with the environment.

2.1.1. Decisional learning mechanisms in Developmental AI

Oudeyer, Kaplan, & Hafner (2007) proposed a developmental learning mechanism that generates a preference for actions allowing fastest learning progress (P1). This approach helps to define priorities in the learning process. This approach respects the principle of environmental agnosticism, as the agent has no *a priori* information on its interactions and its environment, and generates behaviors that satisfy its curiosity principle based on information acquired through its interaction with its environment (P5). However, this mechanism cannot integrate spatial properties of the environment and elements that compose it, and cannot generate spatial behaviors.

Nguyen et al. (2013), and Ivaldi et al. (2014) proposed a curiosity mechanism that leads a robot to learn to recognize objects by manipulating them and observing their properties. These approaches focus on learning new knowledge and allow generation of behaviors based on a form of intrinsic motivation (P1). They are however limited to learning mechanisms based on a form of curiosity that leads an artificial agent to discover behaviors but cannot exploit them.

Blank, Kumar, Meeden, & Marshall (2005) defined a hierarchical learning mechanism where each level defines an abstraction of lower level information and learns to predict this information. The learning mechanism tries to reach environment states where information is predictable and allows fast learning. This approach autonomously generates and exploits hierarchic behaviors (P5), but is based on environment states, which infringes the principle of environmental agnosticism.

2.1.2. Object discovering and exploitation through interactions

Autonomous construction of objects is an important principle in our mechanisms. Indeed, such an ability makes the agent independent from its environment, as there is no need to define *a priori* the elements that compose the environment. This ability also allows the agent to give a meaning to elements of the environment, according to the possibilities of interactions afforded by these elements.

Maye & Engel (2011) proposed that a robot can learn to categorize and recognize objects through sequences of interactions afforded by these objects (P1). The agent thus generates internal object models based on its abilities to detect and interact with them (P2). This principle can however only discover and learn sequential properties of the environment and elements that compose it, and cannot be used to integrate spatial properties of the environment.

Hermans, Rehg, & Bobick (2011) defined a mechanism in which a robot learns to predict properties of objects according to their visual properties (such as color, texture, shape and size). Defining properties of objects according to their visual properties allows these objects to be defined in terms of interactional properties (P2). However, the elements are automatically centered in the camera image, which implies predefined preconditions about elements of the environment.

Cos-Aguilera, Cañamero, & Hayes (2004) used a Self Organizing Map (SOM) to consider object classes. The utilization of a SOM enables the agent to define objects without knowing *a priori* the number of element types in the environment (P2). The agent moves randomly in its environment: this mechanism cannot detect spatial regularities and cannot generate spatial behaviors.

Griffith, Sukhoy, Wegter, & Stoytchev (2012) proposed an approach in which a robot learns to use acoustic properties of objects and proprioceptive stimuli of arms while manipulating these objects under a sink (P2). Griffith, Sinapov, Sukhoy, & Stoytchev (2012) proposed a similar approach where the robot learns to separate containers from non-containers by manipulating them (P2). These approaches are based on the segmentation of objects according to proposed affordance, but cannot be used to generate behaviors.

Pfeifer & Scheier (1994) proposed a mechanism in which a robot learns to recognize and generate implicit object models

that can be lifted and pushed, and obstacles, according to its sensors. The generated implicit models depend of the gripper size of the robot, and its experience of its environment (P1, P2). However, this mechanism cannot recognize distant objects or integrate the surrounding environment.

Montesano & Lopes (2009) proposed a model that allows a robot to determine if an object can be grasped according to visual properties (P2). Although this model cannot generate behaviors, the emergent knowledge can be directly used by the agent.

Uğur, Doğar, Çakmak, & Şahin (2007) proposed a mechanism in which a simulated or physical robot learns to define relevant information in its perception that allows the result of its actions (a set of trajectories) to be predicted (P1). This mechanism allows the robot to navigate in a cluttered environment by avoiding or pushing obstacles (depending on their shapes) (P5). The agent constructs implicit models of elements (P2), but they cannot be used to recognize distant objects or integrate the surrounding environment.

Baleia, Santana, & Barata (2014) proposed a similar approach, but added an arm used by the robot to probe the environment in front of the robot, when the properties of the environment cannot be defined with a sufficient certitude, implementing a form of active perception (P1, P2, P5). However, as described above, this approach cannot integrate the surrounding environment.

These works study the problem of discovering and exploiting objects of the environment by an autonomous and artificial agent. However, the structures learned during object discovering cannot be used to discover and integrate spatial properties of the environment, which dramatically limits the ability to exploit these structures to generate spatial behaviors.

2.2. Space integration

Integrating surrounding space is necessary when an agent needs to localize an element of the environment in order to reach or avoid it. It is even more necessary to keep track of elements when they cannot be observed directly through interactions anymore. Integrating space thus consists in constructing a structure that can track elements in an exploitable way. The memory process has to be able to update this structure to follow the considered elements.

2.2.1. Peripersonal space

The peripersonal space consists of the close surrounding environment with which the agent can directly interact. Integrating this space helps the agent to define possible actions in its current context.

Detry et al. (2009) proposed an approach in which a robotic arm computes positions from where an object can be grasped, which allows the robot to grasp an object without any ontological preconception about this object (P2, P3). Gripper positions are however computed according to a predefined model of the arm.

Pierce & Kuipers (1997) proposed to construct the topological structure defined by an initially *uninterpreted* set of sensors, using similarities between sensor values (P1). This mechanism allows an agent to construct a map of its immediate surrounding environment, and define movements associated with its actions (P2, P3 (limited to close space)). The agent then uses this model to navigate in its environment, but the model is limited to close space (P5).

Fuke, Ogino, & Asada (2007, 2009), and Chinellato, Antonelli, Grzyb, & del Pobil (2010) proposed models that generate implicit links between positions in the visual space and positions that a robotic arm can reach, and localize tactile stimuli in space (P3). These models are not based on a Cartesian reference, but on an interactional reference (P1). This model is however limited to visual and reachable space and cannot generate behavior.

2.2.2. Extrapersonal space

Extrapersonal space is the area of the surrounding space in which an agent has to move before being able to interact with interesting elements. Integrating the extrapersonal space implies recognizing distant affordances, and being able to behave in order to reach them.

Kawamura, Koku, Wilkes, Peters, & Sekmen (2002) proposed a navigation mechanism based on an *ego-sphere*. The ego-sphere consists in projecting points of interest of the environment on a sphere centered on the agent. Points of interest are thus considered by their polar coordinates on the sphere. The agent can then navigate in its environment by comparing sets of points of interest (P3, P5). However, this mechanism does not take distance into consideration and thus does not provide a way to localize the points of interest in space. There is thus no possibility to determine how to reach them.

Lagoudakis & Maida (1999) proposed a mechanism named polar neural map to allow an agent to navigate toward a place, while avoiding obstacles in the surrounding environment. This map considers the surrounding elements in a polar representation and the orientation of the place to reach (P3, P5). However, this mechanism is based on predefined movements in space, and needs a predefined place as a goal to guide the agent.

These works study the problem of learning and exploiting a structure that represents spatial properties of the environment. Learning a structure that characterizes the peripersonal space is well studied in literature, but only a few works study the problem of integrating extrapersonal space. In the presented works, extra-personal space representation is based on a predefined structure that cannot provide information about the distance of elements in the environment. Moreover, these works do not rely on autonomously learned structures to represent elements of the environment, such as works presented in Section 2.1. Using learned structures to build a structure that characterizes peri and extrapersonal space is however required for an agnostic agent to generate behaviors adapted to its environmental context.

2.3. Sensorimotor contingencies learning

The model proposed by Uğur, Doğar, Çakmak, & Şahin (2007) is very close to our interaction signature mechanism (described in Section 4.1). These authors propose to determine

the possibilities of actions (a set of forward movements ranged from turn-sharp-right to turn-sharp-left) according to visual information. The agent learns to extract relevant visual information that can be used to determine when an action is possible, both because there is no object on the agent's path and because the object can be pushed. The agent can then navigate in a cluttered environment while avoiding or pushing obstacles without any preconception about the environment and the objects composing it. However, this mechanism cannot be used to integrate extrapersonal space: as the agent uses perceptions to define the *traversability* of the environment, it cannot exploit the learned structures to discover the spatial properties of its environment, or take distant elements into consideration.

2.4. Previous studies based on Radical Interactionism and spatial learning

Georgeon, Marshall, & Manzotti (2013) associated an RI mechanism with a spatial memory within a cognitive architecture called the Enactive Cognitive Architecture (ECA). The agent was provided with the coordinates of the enacted interactions in an egocentric referential. Knowing the spatial position of enacted interactions, the agent could detect when they overlapped. From spatial overlaps of enacted interactions, the agent could infer the presence of objects in the surrounding space that afforded these interactions. The agent learned to represent categories of objects by the set of interactions that they afforded. This model required the strong assumption that the agent received information about the position of enacted interaction. The present study aims at removing this assumption. Another limitation of this model was that it was unable to deal with objects that could be changed by the agent (e.g. a preys that disappear after being eaten). The present study also seeks to overcome this limitation.

3. The Radical Interactionism (RI) Approach

The Radical Interactionism (RI) model (Georgeon & Aha, 2013) considers the exchanges between an agent and its environment in the form of sensorimotor schemes called *interactions*, rather than separated actions and perceptions. The RI model does not require the notion of environment states or extrinsic reward, which is compliant with the principle of environmental agnosticism. The agent thus actively discovers its environment through successive interactions with the environment, and constructs its *perception* as an internal model based on interactions.

Considering interactions rather than separated actions and perceptions has the advantage of considering both the possibilities of interactions provided by the environment, related to initially unknown objects, and the initially unknown movement produced by an interaction. Indeed, an action or a perception alone cannot provide information about the agents movement because an action can fail and a perception can be observed as a consequence of several actions. Using interactions thus helps to define movements of the agent and to construct a structure to characterize a spatial environmental context.

3.1. Formalization of the RI model

The RI interaction cycle begins with the agent selecting and trying to enact an intended interaction $i_t \in I$ (as defined in Section 1). Enacting interaction $i = \langle a, r \rangle$ consists of performing action a (through activating actuators), and receiving result r (through sensors). The process of enaction of i is programmed by the designer of the agent but is ignored by the agent. At the end of the interaction cycle, the agent gets the *enacted interaction* e_t that was actually enacted. Figure 1 shows this *enaction cycle*, from the agent's point of view. The enaction of i_t is a success if $e_t = i_t$, and a failure otherwise. In a given state of the environment, the enacted interaction depends on the intended interaction. Therefore, the agents input data (the enacted interaction) does not represent the state of the environment, and thus does not constitute the agents perception. Instead, perception is an internal construction maintained by the agent according to its experience interacting with the environment. The agent selects next intended interaction during the next enaction cycle according to this internal construction.

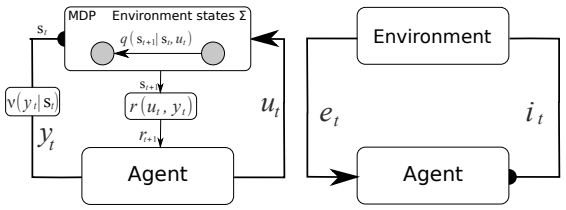


Figure 1. Comparison between POMDP (left) and RI (right). The RI cycle begins with an intended interaction i_t selected by the agent, as opposed to the POMDP cycle, which begins with the agent receiving an observation y_t. This inversion of the interaction cycle is materialized in the figure by the black circle (beginning) and the black arrowhead (end). The RI model does not refer to the notion of environmental states: for the agent, the environment is *opaque* and can only be accessed by experiencing it through interactions. The agent's motivation is intrinsic: the agent experiences the satisfaction from the enaction of an interaction rather than from an external reward based on some environmental state.

The valence function v (introduction in Section 1) defines the agents behavioral preferences, without using an extrinsic reward. An RI agent learns to anticipate the results of its interactions, tries to enact interactions with high valences, and tries to avoid enacting interactions with negative valence. As a result, to an external observer, the agent seems to like enacting interactions that have a positive valence and to dislike enacting interactions that have a negative valence.

The RI model differs from standard reinforcement learning approaches (e.g. POMDP, Åström (1965)) in that it does not aim at maximizing a reward value. Rather, the agent learns behaviors that satisfy the interactional motivation principle. The RI model does not use rewards defined as a function of the environmental state; it uses valences associated with enacted interactions. Moreover, the RI model does not refer to the environment states: for the agent, the environment is *opaque* and can only be accessed by experiencing it through interaction, which satisfies the principle of environmental agnosticism. A RI agent is evaluated through its behavior and through the structures that it learns, according to a set of criteria (Georgeon &

Sakellariou, 2012) inspired by the developmental robotics domain (Lungarella, Metta, Pfeifer, & Sandini (2003), Weng et al. (2001)).

3.2. Formalization of the Parallel RI model

To overcome the limitations of the RI model presented in Sections 1.1 and 2.4, this paper proposes an extension of the RI model, called *Parallel Radical Interactionism* (PRI). The PRI is similar to the RI in principle, but differs in that it allows the agent to simultaneously experience several enacted interactions. The intuition comes from living beings that feel multiple sensory stimuli while interacting with their environment. For example, an animal can move forward, and simultaneously experience the optical flow and the Doppler effect that result from this movement. We thus propose that the agent can get additional results, in addition to the result that belongs to the enacted interaction. In the PRI model, we distinguish between two kinds of results: *primary results* $r \in \mathcal{R}'$ and *secondary results* $r'' \in \mathcal{R}''$, with the set of all results $\mathcal{R} = \mathcal{R}' \cup \mathcal{R}''$ However, secondary results cannot be considered without the movement produced by the enacted interaction. As an example, the optic flow on the retina can only convey spatial information if it is considered with the movement that generates it. We also cannot construct new interactions by associating the result with the action that composes the enacted interaction, because an action is not sufficient to characterize the movement of the agent. This led us to distinguish between *primary interactions* and *secondary interactions*: A primary interaction $i' = \langle a, r \rangle \in I' = \mathcal{E} \times \mathcal{R}'$ is the association of an action with a primary result. Primary interactions of the PRI model are similar to interactions of the RI model. A secondary interaction $i'' = \langle i, r'' \rangle \in I'' = I' \times \mathcal{R}''$ is the association of a primary interaction with a secondary result. Secondary interactions are specific to the PRI model. They are meant to convey additional results appended to the primary interaction i'.

At enaction cycle t, the agent selects and tries to enact an intended primary or secondary interaction $i_t \in I = I' \cup I''$. The difference with the RI model is that, at the end of the enaction cycle t, the agent experiences a set of enacted interactions $E_t = \{e_k\}_t$ (Figure 2). The set of enacted interactions E_t contains one primary interaction only plus a set (possibly empty) of enacted secondary interactions associated with this primary interaction. It constitutes a representation of the current context as the agent experiences it through interactions; we thus call the set E_t the *interactional context*.

The PRI model keeps track of the success or failure of enactions in a more extended way than the RI model. In the case of a primary interaction, an interaction i is marked as *successfully enacted* at enaction cycle t when $i \in E_t$, whether it was intended or not. A primary interaction i is marked as *failed* when i is intended but not enacted (i.e. $i = i_t \wedge i \notin E_t$). This means that another interaction $j \neq i$ is enacted instead of i. If an interaction j can be enacted instead of an interaction i, and i and j are never enacted simultaneously, we can consider that j is an *alternative* of i. We define that two interactions i and j are *opposite* if i and j are mutually *alternatives*. As a consequence, if an opposite interaction j of i is enacted (i.e. $j \in E_t$), then the interaction i is marked as *failed*, even if it was not intended.

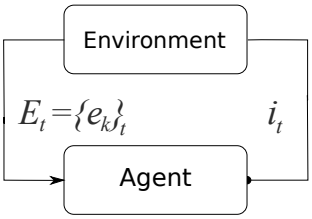


Figure 2. Diagram of the Parallel Radical Interactionism model (adapted from the RI model). At enaction cycle t, the agent tries to enact an intended interaction i_t , and gets a set of enacted interactions $E_t = \{e_k\}_t$, that constitutes the *interactional context* experienced by the agent.

All the secondary enacted interaction (elements of $E_t \cap I$) are marked as *successfully enacted* on time t. All the secondary interactions that do not belong to E_t but whose associated primary interaction belongs to E_t are marked as *failed* on time t.

3.3. Implementation of the PRI model in an artificial agent

Based on the PRI model, we designed an artificial agent that moves in a 2-dimensional environment. This environment contains several objects that afford several possibilities of interaction to the agent. The agent has five possible actions: *move forward of its length*, *turn left of* 90° , *turn right of* 90° , *turn left of* 45° , and *turn right of* 45° . The agent's sensory system generates the primary results of these actions. The *move forward* action can yield three results: 1) successfully moving forward, 2) bumping into a solid object, and 3) moving forward and eating something edible. The turn actions can yield only one primary result: successfully turning. The set of primary interactions I' thus contains the 7 interactions listed in Table 1: successfully *moving forward*, *bumping*, *eating*, *turning left of* 90°, *turning right of* 90°, *turning left of* 45°, *turning right of* 45°. We set the valence of these interactions such that the agent slightly likes moving forward, dislikes bumping, and strongly likes eating. The interactions moving forward, bumping, and eating are opposite to each other, because a failure of one produces the enaction of the other.

Table 1. List of the seven primary interactions used by the agent. The valence of each interaction is given in brackets.

move forward of one step (5)
bump in a solid element (-10)
move forward and eat something edible (50)
turn 90° left (-3)
turn 90° right (-3)
turn 45° left (-3)
turn 45° right (-3)

Additionally, the agent is equipped with a visual system that detects objects in a 180° visual field based on their color c (among {*red*, *green*, *blue*}), and provides information about the optic flow (θ, v). θ is the angle on the retina, and v is the mea-

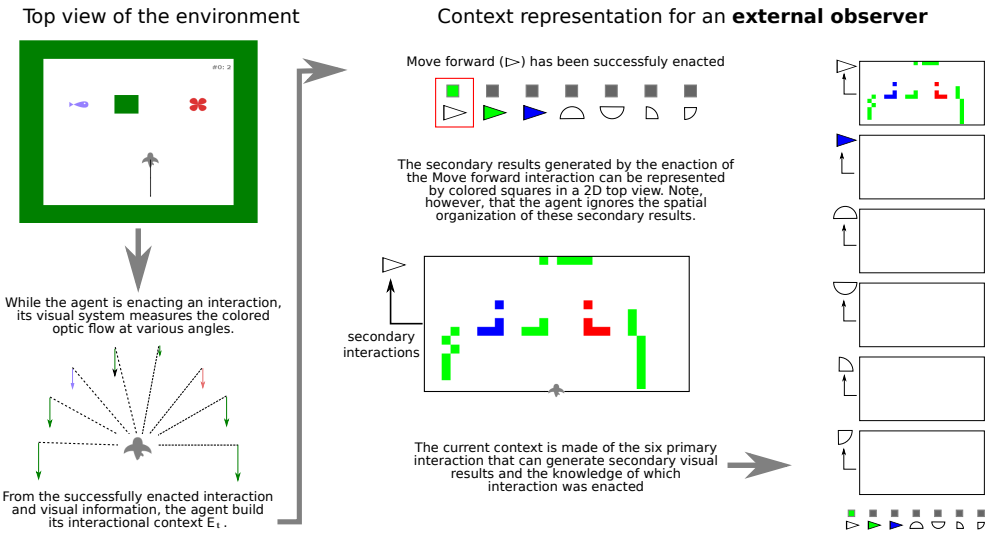


Figure 3. The environment of the agent and a representation of the interactional context for an external observer. Top left: the agent (represented as a gray shark) in its environment. A black line shows the pathway of the agent. Bottom left: the agent gets the enacted interaction and a set of secondary visual interactions. This constitutes the interactional context experienced by the agent. Center: to make the interactional context easier to read by an external observer, we mark the enacted interaction with a green square and the other interactions with dark squares. As external observers, we know the positions and colors of the objects that caused the secondary visual results. We thus display the secondary results as colored squares, using their colors and position relative to the agent. Right: Representation of the current interactional context. Top right: the additional results are attached to the enacted interaction. As there are 6 primary interactions that produce secondary interactions (*Bump* does not produce movement), there are six groups of secondary interactions. In the current context, some visual interactions associated to *move forward* (white triangle) where enacted, while the five other groups remain empty. Bottom right: the enacted interaction is marked by a green square.

sured optic flow. This visual system thus provides secondary results in the form of triples $r'' = \langle c, \theta, v \rangle$.

We did not construct primary results from visual stimuli because the measured optic flow depends on the agent's displacement. Since an action can generate different displacements whether it succeeds or not, the agent cannot learn any spatial properties based on the action only. Instead, we use visual stimuli to produce secondary results. Since an interaction provides information on the agent's displacement, each ⟨ interaction,secondary result ⟩ couple is related to a unique position in space (unknown by the agent *a priori*). Since the bumping interaction does not generate a movement in space, we do not provide the agent with secondary results while bumping. A secondary result thus consists in seeing an element of color c at a predefined (but unknown) position, while enacting an interaction other than bumping. Note that the secondary results do not convey information about the position of each result in the 2D space. Therefore, the agent must reconstruct the spatial property of the environment without presupposition about the spatial position of its sensory stimuli.

Each couple (i, r'') is related to a specific element of a certain color at a specific position in space relative to the agent. We discretize the space of couples (i, r'') to define a finite set of secondary results. Other discretizations can be used, as the agent learns to extract spatial properties that emerge from interactions *a posteriori*. The selected discretization must, however, allow the agent to observe relative movements of elements to integrate spatial properties of the environment. We propose to discretize the space of visual stimuli such that the positions associated with visual interactions define a regular grid of 15×30 positions that covers the agent's visual field. The unit of this grid is smaller than the size of the objects of the environment, so that the agent can detect relative movements of these objects. We use the same discretization for each primary interaction to make observation of emerging properties easier.

We thus define a set of 8100 secondary interactions ((15×30) positions ×3 colors ×6 interactions producing movement).

The agent moves in a 2D environment that can contain three types of elements that afford interactions. Each type of element has a specific color that makes it recognizable according to the sensorimotor possibilities of the agent:

- Preys (blue fish), that afford the interaction *eating*,
- Walls (green bricks), that afford the interaction *bumping*,
- Algae (red flowers), that have no influence on the enaction of interactions. Algae thus have the same interactional properties than an empty space. We expect the agent to learn to ignore these elements.

These three types of elements are opaque: the agent cannot see an object hidden by another one. Figure 3 shows the agent in its environment, as well as an organized representation of the interactional context.

4. A Spatial Memory Based on Interactional Experience

We present a mechanism that allows a PRI agent to integrate its surrounding environment by constructing a spatial context. The agent first learns to recognize objects that afford its interactions. Next, the agent generates data structures that characterize position of objects in terms of interactions. By categorizing and localizing objects in terms of possibilities of interaction, the agent projects valences of interactions to different regions of the surrounding space according to interactions that are assumed to

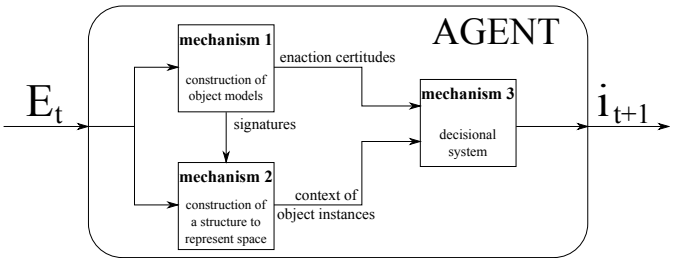


Figure 4. The agent's algorithm is divided into three inter-dependent mechanisms (Adapted from Gay, Georgeon, & Wolf (2014)). Mechanism M1 learns to recognize objects that afford interactions according to the interactional set E_t. Mechanism M2 recognizes objects in space and constructs a spatial memory in the form of data structures that characterizes spatial properties of the environment, and localizes, integrates and tracks objects defined by mechanism M1. Mechanism M3 exploits information given by the spatial memory to generate interactionally-motivated behaviors.

be afforded in these regions. Based on these values, the agent can generate behaviors that satisfy its interactional motivation in the short and medium terms.

Figure 4 presents the three sub-mechanisms that compose our agent's algorithm. Each sub-mechanism implements an elementary functionality and exploits information provided by other mechanisms. The first mechanism (Section 4.1) constructs data structures, called *Signatures of Interactions*, used to represent categories of objects. Once learned through the agent's experience, signatures of interaction allow the agent to predict the *enaction status* (success or failure) of future intended interactions based on the current context. Signatures of interactions are fundamental for the spatial memory system, because the whole mechanism relies on properties learned by these signatures. The second mechanism (Section 4.2) recognizes objects characterized by signatures of interactions in the surrounding environment, and learns to track and localize these objects in egocentric reference while the agent is moving in space. Mechanism M2 indexes *positions* in space by the sequences of interactions that allow the agent to reach these positions. It also keeps track of objects using data structures called *places*. A place is a set of *positions* that share similar interactional properties. The last mechanism (Section 4.3) exploits the two previous mechanisms to generate behaviors that can satisfy the interactional motivation of the agent.

4.1. Mechanism M1: constructing signatures of interaction

This mechanism is inspired by an experiment on the F5 area (ventral premotor cortex) of the monkey brain carried out by Murata et al. (1997). It showed that neurons in the F5 area responded to the presence of an object in front of the animal, whether the animal grasped the object or simply looked at it. The response of these neurons varied depending on the movement required to grasp the object rather than the global shape of the object. The neurons remained active until the monkey observed the absence of the object. From this experiment, we draw two hypotheses:

1) A specific context of elements of the environment can be defined and characterized by the interactions that are afforded by this context. Therefore, we do not specify objects by intrinsic features but by the interactions they afford to the agent. An *object* is thus defined as a specific context of elements that afford an interaction. This idea relates to Gibson's notion of affordances (Gibson, 1977), and, more precisely, to the formalization of affordances defined by Stoffregen (2003) and Chemero (2003), who defined an affordance as a property of the agent-environment coupling rather than a property of the agent or of the environment alone.

2) A possibility of interaction can indicate the presence of the object that affords it. The presence of objects in a given situation can thus be represented by a set of *enactable* interactions, even when the agent cannot directly detect these objects.

Mechanism M1 exploits these hypotheses by estimating the possible enaction status (success or failure) of interactions in the current context. Several implementations were described and tested by Gay & Georgeon (2013) and Gay, Georgeon, & Wolf (2014)).

Note that the agent can characterize its environment using only a limited number of interactions, regardless of the complexity of the environment. Indeed, every part of the environmental context can be characterized by the set of afforded and the set of non-afforded interactions. This representation of the environment is associated with motivational valence: objects that afford interactions that have high valences becomes attractive, while objects that afford interactions that have negative valences become repulsive. The agent thus *projects* valences of interaction onto the current environment.

4.1.1. Formalization of signatures of interactions

The agent can only perceive its environment through interactions, and thus cannot directly perceive *objects* that affords its interactions. However, we can consider that an enacted interaction e_t carries information about the presence of elements in the surrounding environment. An interactional context E_t can thus characterize a context that contains the object affording an interaction i.

Mechanism M1 constructs, for each interaction i, set(s) of interactions that can help the agent to assess the certitude of presence or absence of the object that affords i. From these sets of interactions, the agent will be able to assess the possibility of successfully enacting i. We call *signature* S_i of an interaction i a structure learned by experience that characterizes such sets of interactions. We formalize a signature S_i of an interaction i as a function (1) that gives a numerical value in the interval $[-1, 1]$ that reflects the possibility of successfully enacting i in an interactional context E.

$$S_i : \mathcal{P}(I) \rightarrow [-1; 1] \quad (1)$$

where $\mathcal{P}(I)$ denotes the partition of I (the set of subsets of I). $S_i(E) = 1$ means that the agent has the certitude that the tentative enaction of i in context E_t will succeed, $S_i(E) = -1$ means that the agent has the certitude that the tentative enaction of i will fail. Above some threshold $\mu \in [0; 1[$, the agent has some reasonable belief that i can be enacted, and below μ, a reasonable belief that it would fail. When $-\mu \leq S_i(E_t) \leq \mu$, the

agent makes no assumption about the possibility of enacting i in context E_t. The higher is μ, the more interactions are considered enactable or non-enactable, but with a less reliability. The lower μ is, the fewer interactions are considered as enactable or non-enactable, but with higher reliability.

S_i must be learned and reinforced each time i is enacted as a success or a failure (we do not consider enaction cycles where i is not enacted) to respect the following condition:

$$\lim_{t \to +\infty} S_i^t(E_{t-1}) - res(i,t) = 0 \quad (2)$$

where $res(i,t) = 1$ if i is successfully enacted at enaction cycle t and $res(i,t) = -1$ if i failed. The reinforcement of the signature of an interaction i is thus supervised, and compares certitude of success in the previous interactional context E_{t-1} with the actual enaction of i. Over time, the signature learning mechanism reinforces signatures of interaction to minimize prediction errors and provide pertinent certitudes.

The parameters that characterize S_i characterizes how the agent *perceives* (or *experiences*) the object affording the interaction i. Note that when the object affording an interaction i cannot be detected by the agent, the signature of i cannot be defined.

An implementation of this signature mechanism must respect these two properties:

- A signature must be used to predict the result of an intended interaction in the next enaction cycle.

- A signature must be reversible. Indeed, if an interaction i is considered as enactable, then the *object* affording i can be considered as present in the environment, and interactions that would have detected this object can be considered as enacted. Thus, it must be possible to define a function $\hat{S}_i$ that can provide contexts $E \in \mathcal{P}(I)$ that can afford i (given by $\hat{S}_i(1)$) and contexts that do not afford i (given by $\hat{S}_i(-1)$). The signature can then complete the current interactional context, adding information that cannot be detected by the agent in the current context.

4.1.2. An example of implementation

We propose to implement the signature mechanism M1 on our agent with a single layer neural network. This solution is pertinent for our simple environment, as each interaction is afforded by a unique element of the environment. Each interaction is attributed a formal neuron that gives the certitude of success in an interactional context E_t. We modeled the input layer of the neural network as a vector $[\epsilon_{1,t}, ..., \epsilon_{n,t}]$ where $n = Card(I)$ and $\epsilon_{k,t} = 1$ when the k^{th} interaction of I succeeded at enaction cycle t (i.e. $i_k \in E_t$) and $\epsilon_{k,t} = 0$ otherwise. The signature of an interaction i is characterized by the set of synaptic weights $[w_{i,1}, ..., w_{i,n}]$ and a bias $w_{i,n+1}$. The certitude function S_i of an interaction i is defined with a linear function of inputs, passed through an activation function that restrains the output value in the $[-1; 1]$ interval:

$$S_i(E) = g\left(\sum_{k \in [1;n]} \epsilon_k . w_{i,k} + w_{i,n+1}\right) \quad (3)$$

$$g(x) = \tanh(\frac{x}{2})$$

A signature S_i of an interaction i is reinforced each time i is enacted as a success or a failure, using the delta rule (or Least Mean Squares method) (4). We note $res(i,t) = 1$ if i was successfully enacted at enaction cycle t and $res(i,t) = -1$ if i failed. The bias $w_{i,n+1}$ is related to an input $\epsilon_{n+1,t}$ for which the value is 1 at each enaction cycle.

$$w_{i,k}^t \longleftarrow w_{i,k}^{t-1} + \eta \times \epsilon_{k,t-1} \times (res(i,t) - S_i(E_{t-1})) \quad (4)$$

$\forall k \in [1; n+1]$, η is the learning rate where $\eta \in [0; 1]$. In our experimentations, we used a constant learning rate of 0.5, which offers a good compromise between learning speed and signature stability.

Considering this implementation, a signature S_i is characterized by a list of weights $\{w_{i,k}\}_{k \in [0;n+1]}$, that summarizes contexts that afford i or prevent enaction of i. We thus propose that, in this implementation, $\hat{S}_i(1) = \{w_{i,k}\}_{k \in [0;n+1]}$ and $\hat{S}_i(-1) = \{-w_{i,k}\}_{k \in [0;n+1]}$.

For visualization by an external observer (like us), we propose to display summarized context $\hat{S}_i(1)$ in a way that makes them easy to read, shown in Figure 5 (note that the agent does not need to topographically organize signature weights). First, as we know, as an external observer, the associated primary interaction and the characterized color of visual interactions, we propose to gather weights according to the associated primary interaction and color of the secondary interaction they are connected to. We also know the position in space characterizing each secondary interaction: each group of weights can be organized to match positions of their connected secondary interactions. This organization allows observation of spatial properties of objects, such as position according to the agent and size. Finally, we overlap groups of weights for which the connected secondary interactions are associated to the same primary in-

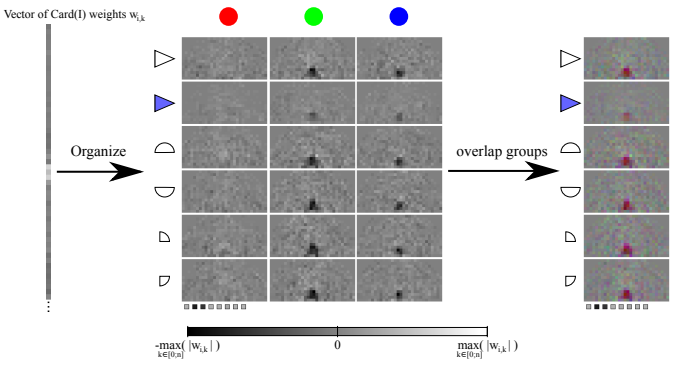


Figure 5. Representation of signatures for an external observer (here, the signature of interaction *move forward* of a trained agent). Signature weights are organized in a similar way as for the interactional context (Figure 3). Left: a signature consists of a vector of weights. We propose to gather weights related to visual interactions that are related to a same color stimulus and associated with a same primary interaction, and organize each group to match positions characterized by visual interactions (middle). Weights related to primary interactions and the bias are displayed separately with eight squares, in the order given in Table 1. As we have three colors, we overlap groups related to different colors using the three channels of a RGB image (right). Here, an external observer can observe that the interaction *move forward* is afforded by the absence of green and blue elements in front of the agent: some weights related to interactions *seeing a green* and *seeing a blue* element are strongly negative.

teraction: as there are three primary colors, we can display the three overlapping groups using the three color channel of a RGB image. Of course, each color group is displayed using the channel of the same color. We can thus observe the color properties of the object defined by signatures.

4.2. Mechanism M2: construction of a structure to characterize space

This mechanism is designed to generate a structure, we call *space memory*, which can give relevant and exploitable information about the environmental context of the agent. Our previous work (Gay, Georgeon, & Wolf, 2014) have shown that only two types of information are required to localize and consider an object in space: the interactions that allow to move closest to this object, and an estimation of its distance. Implementation of a hard-coded but imprecise space memory that generates this limited information on a robot has shown that the decisional mechanism was robust enough to enable the robot to generate behaviors satisfying its interactional motivation. Thus, the mechanism we presented in this section is designed to generate these two types of information.

The mechanism that constructs and maintains this structure is divided into two sub-mechanisms (Figure 6): the first mechanism extracts spatial regularities from signatures of interactions and detects objects in the area of space covered by interactions. We call this space *Observable Space*, as an object in this area can be detected through enacted interactions. The second mechanism exploits spatial regularities to maintain the position of previously detected objects in the surrounding space, we call *Global Space*. The global space consists of the space, in egocentric reference, that the agent can integrate. Global space includes observable space.

4.2.1. Detecting objects in observable space

This mechanism exploits the relations between interactions, discovered through their signatures. Indeed, a signature S_i generates a link between an interaction i and sets of interactions $\{j_k\}_l \in \hat{S}_i(x), x \in \{1; -1\}$ that allows to determine the enactability of i. However, interactions j_k of these sets may have their own signatures S_{j_k}, and each context $E_l = \{j_k\}_l$ contains a unique primary interaction j_0 and a set of secondary interactions $\{j_k\}_{k\neq 0}$ associated to j_0. By considering signatures of interactions $j_k \in E_l$, we can characterize an environmental context that, if *moved* by enacting j_0, will afford i. We can thus *backmove* a signature S_i through a primary interaction j_0 using the following procedure: we note $\hat{S}_i^{\sigma_0} = \hat{S}_i(1)$, where σ_0 is an empty sequence of interactions, and construct:

$$\hat{S}_i^{[j_0,\sigma_0]} = \bigcup_{\forall E_l \in \hat{S}_i^{\sigma_0} / j_0 \in E_l} \{E \in \mathcal{P}(I) \,/\, \forall j_k \in E_l, S_{j_k}(E) > 0\} \quad (5)$$

Interactional contexts of $\hat{S}_i^{[j_0,\sigma_0]}$ characterize environmental contexts that can afford i after enacting j_0. By applying successively (5), it is possible to *backmove* a signature S_i through an increasing sequence of interaction σ. We call *predecessor* S_i^{σ} a signature S_i *backmoved* through a sequence of interactions σ. A predecessor characterizes a context that, if *moved* through the enaction of the sequence of interactions σ, affords i.

By using the predecessor S_i^{σ} of S_i, it is possible to estimate the prediction of enaction result of an interaction after enacting a sequence of interactions σ (regardless of the enactability of σ). We then consider that an *instance* of the object affording i is present at a position that can be reached by enacting the sequence σ, with a certitude given by $S_i^{\sigma}(E_t)$. A certitude of 1 indicates that an instance of the object affording i is present at position characterized by σ with an absolute certitude, and a certitude of -1 indicates that the instance is absent with an absolute certitude. Note that this mechanism is not a path planning algorithm: the sequence σ is considered regardless of its enactability.

We thus define a way to assimilate a position in the observable space as a sequence of interactions. An instance of an object affording an interaction i, localized at a *position* σ, consists of a context that, when moved by a transformation σ, affords the interaction i. This definition of positions relates to the notion of *Representative Space* of Poincaré (1902), for whom localizing an object in space means considering the movement needed to reach it.

We implemented the signature predecessor mechanism on our agent. Considering our implementation of signatures based on formal neurons, the predecessor of a signature of an interaction is computed as follows: we note $\hat{S}_{i,j_0}^{\sigma} \subset \hat{S}_i^{\sigma}$ the subset of weights of a signature S_i or a predecessor S_i^{σ} that are connected to the primary interaction j_0 or a secondary interaction j_k

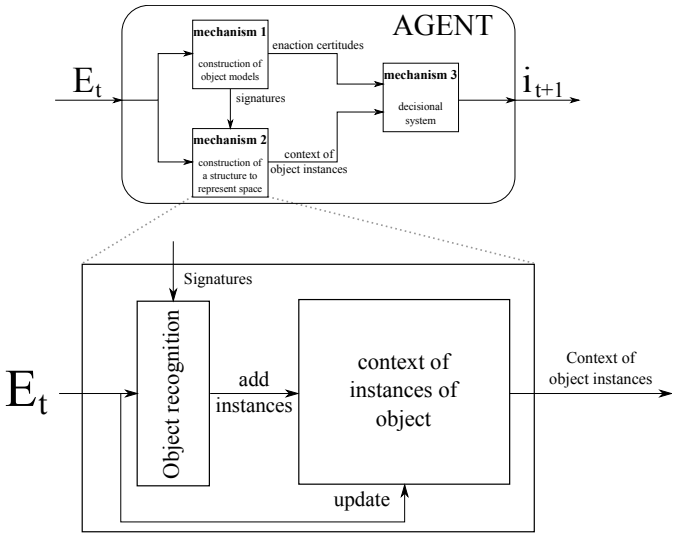


Figure 6. Construction mechanism for a structure that characterizes the environmental context. This mechanism is divided into two sub-mechanisms: the first mechanism recognizes and localizes distant objects in the observable space, using the last interactional context E_t and signatures of interactions. The second sub-mechanism then stores detected objects, and maintains positions of stored objects in egocentric reference at each enaction cycle. This second sub-mechanism characterizes positions of objects in the global space. Stored objects can be compared with detected objects to possibly recognize and update them. The object context consists of a list of objects localized by the interaction allowing to move closest and their distances, and can be used by the decisional system (M3).

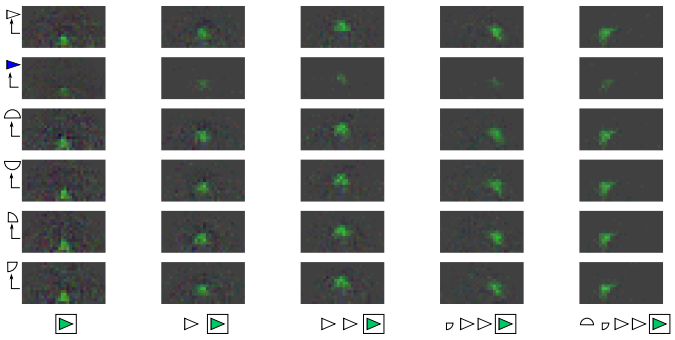


Figure 7. From left to right: signature of interaction *bump*, predecessors of the signature of interaction *bump* by an increasing sequence of interactions (below the signature predecessors). The green blob that represents the object affording *bump* is moved through the observable space when an interaction is added to the sequence characterizing the object instance position. However, we do not observe significant changes in color, shape or size of the object, which allows detection of objects at different positions.

associated to j_0. The predecessor $S_i^{[j_0,\sigma]}$ is computed by adding signatures of interactions j_k associated with j_0, weighted by the corresponding weights $w_{i,k}^{\sigma} \in \hat{S}_{i,j_0}^{\sigma}$, and normalized according to the greatest weight of $\hat{S}_{i,j}^{\sigma}$:

$$\hat{S}_i^{[j_0,\sigma]} = \sum_{w_{i,k}^{\sigma} \in \hat{S}_{i,j_0}^{\sigma}} \frac{w_{i,k}^{\sigma}}{\max\limits_{w_{i,k}^{\sigma} \in \hat{S}_{i,j_0}^{\sigma}} (w_{i,k}^{\sigma})} \times \hat{S}_{j_k}(1) \qquad (6)$$

In this implementation, we only consider weights for which the absolute value is greater than a threshold to reduce the number of weights to compute. The predecessor $S_i^{[j,\sigma]}$ is considered as non-exploitable if a certain amount of signature of interactions designated by $S_{i,j}^{\sigma}$ are not defined. This means that the object is partially or totally out of the observable space and cannot be detected through the path $[j, \sigma]$. Figure 7 shows an example of sequence of signature predecessors by an increasing sequence of interactions, with signatures learned by a trained agent (we do not threshold weights in this example).

We also propose to reduce the number of possible transformations σ by detecting redundant *positions*: when two sequences σ_1 and σ_2 of different lengths generate two similar predecessors of the same signature of interaction (i.e. they define similar contexts $\hat{S}_i^{\sigma_1} \simeq \hat{S}_i^{\sigma_2}$), it is then possible to remove the longest sequence. This detection ensures that sequences are always the shortest, for every position of space. Redundancy can vary from one interaction to another: as an example, if a signature characterizes an object with no specified orientation, two sequences of interactions that lead to the same instance but from different orientations may be considered as redundant, which is not the case with an object that needs a certain orientation to afford an interaction.

4.2.2. Notion of Places

It is possible to store object instances and their positions in egocentric reference by using sequences defined by the object detection mechanism described in Section 4.2.1. However, if the agent does not enact this sequence of interactions, or if the sequence is not enactable in the current environment, the object instance is lost from agent's memory. We thus need to define a structure that can learn to store and update positions of object instances, regardless of enacted interactions.

As we defined positions with sequences of interactions (Section 4.2.1), we can obtain the two required types of information: first, the distance, given by the minimum number of interactions needed to reach the object instance. Distance is thus defined in terms of number of interactions rather than a geometrical distance. Second, the first interaction of a sequence gives the interaction making it possible to move closest, as the sequence is considered as the shortest path to reach the instance.

We do not need to consider two instances of the same object separately when they share the same characteristics. We thus propose to define a *place* as the list of positions (or sequence of interactions) that are characterized by the same interaction and the same distance. A *place* thus consists of a list of sequences of interactions that have the same length and begin with the same interactions. An object instance is considered as *present* in a place l if at least one instance of this object is detected at a position that composes this place, i.e. $\exists \sigma \in l \;/\; S_i^{\sigma}(E_t) > \mu$.

We can thus define a context that characterizes the content of the observable space by listing the interactions that can be enacted in each place. This context defines interactions that allow the agent to move toward distant affordances.

4.2.3. Composite places

A context of objects based on places is limited to the observable space as it consists of positions that can be experienced through interactions. We propose the following principle, illustrated in Figure 8, to characterize the presence of an object instance in the global space:

We consider an object instance in the non-observable space. If the agent uses an interaction i that produces a movement bringing the object instance in a place l of the observable space, then we consider that before enacting i, it was possible to characterize the presence of the object instance by considering the place l *backmoved* by the interaction i. We thus propose the notion of *composite place* to define such a moved primitive place: we note $[i_1, ..., i_n; l\rangle$, where $[i_1, ..., i_n]$ is a sequence of interactions we call the *path* of the composite place, and l is the *final*

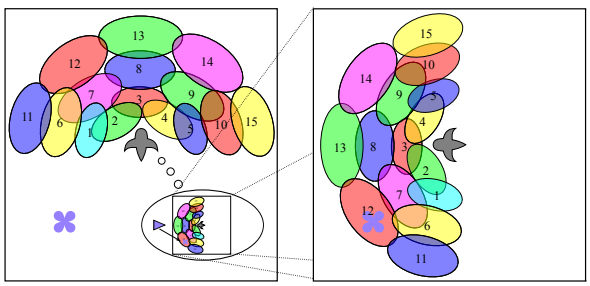


Figure 8. Principle of composite places. Left: an edible alga is present in the non-observable space (places are randomly defined for illustration). If the agent turns left, then the element can be detected in place 12. It was thus possible to initially characterize the presence and the position of this element by considering the place 12 displaced by the interaction *turn left*. We call *composite place* a place preceded by an interaction or a sequence of interactions.

place of the composite place. A composite place thus consists of a set of positions (the final place), preceded by a sequence of interactions (the path) that can *backmove* the final place in the non-observable space.

4.2.4. Signatures of places

To use a composite place, we first need to determine the positions of the observable space (if any) that belong to the area defined by this composite place. Note that this process is not applied to primitive places as their positions are defined by construction. To determine whether a position belongs to a composite place, we propose the following principle: the agent detects positions of object instances in its environment, as an example, an instance of the object affording the interaction i at position σ. Then, it enacts the path of a composite place l. If an instance of the object affording i is then detected in the final place of l, then the initial position σ of the object instance may be included in the area characterized by the composite place l. Of course, the instance is not necessarily the same. It is thus preferable to focus on instances of the less represented object of the context. Figure 9 illustrates this principle.

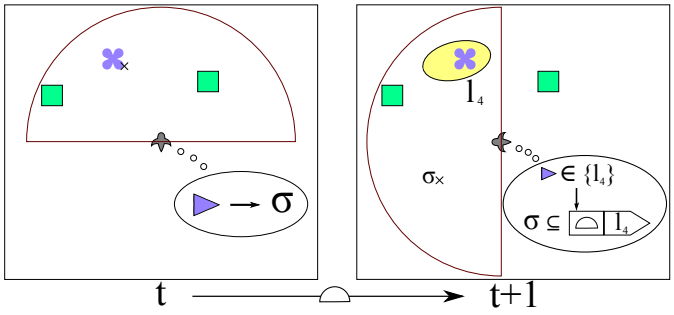


Figure 9. Principle of signatures of places. At enaction cycle t (left), the agent detects an instance of the object that affords the interaction *eat*, localized at position σ. The agent then *turns* 90° *left*. An instance of the object that affords the interaction *eat* is detected in the place l_4 (yellow area). Thus, the composite place [*turn* 90° *left*][l_4⟩ may contain the position σ. Note that the agent cannot determine whether or not the two detected instances are the same.

Learning to define places that belong to a composite place thus helps predict observation of an object instance in the final place according to a context observed before enacting the path of the composite place. We propose a principle similar to the interaction signature principle.

We define a *signature of place* S_l of a composite place l as a structure that characterizes the positions σ in space considered as included in l. For each interaction i, we define the *context of position* $E^{\pi}_{t,i}$ of the object affording i as the list of every position σ of space in which an instance of the object affording i is detected: $E^{\pi}_{t,i} = \{\ \sigma\ /\ S^{\sigma}_{i}(E_t) > \mu\}$. As with signatures of interaction, signatures of places can be formalized as a function

$$S_l : \mathcal{P}(\Sigma) \rightarrow [-1; 1] \qquad (7)$$

where Σ is the set of possible sequences of interactions σ and $\mathcal{P}(\Sigma)$ denotes the partition of Σ (the set of subsets of Σ). Function S_l characterizes the certitude of presence of an instance of an object affording an interaction i in the *final place* of a composite place l, after enaction of the *path* of l, according to a context of place $E^{\pi}_{t,i}$, with an absolute certitude of presence when $S_l(E^{\pi}_{t,i}) = 1$ and an absolute certitude of absence when $S_l(E^{\pi}_{t,i}) = -1$.

The place signature learning process is similar to the interaction signature learning process. It consists in reinforcing signatures of places, by analyzing contexts of places, to define pertinent certitude of presence of instances in a place l. The learning process is applied to a signature S_l when the *path* of the composite place l is successfully enacted. The reinforcement of the signature S_l of a place l compares certitudes of presence in a place in the initial context of position $E^{\pi}_{t-n,i}$, where n is the length of the *path* of l, and the actual presence of an instance of the object affording i in the place l. In the same way as the interaction signature learning mechanism, the place signature learning mechanism reinforces signatures to minimize prediction errors and provide pertinent certitudes.

4.2.5. Storage of object instances

According to our definition of objects, an object instance is considered as present at every position in space where the corresponding interaction is enactable. However, certain elements can afford an interaction from multiple positions. It is thus necessary to limit the stored instances of a certain object instance to the most relevant ones. As an example, walls can be bumped from an infinity of positions, but the most relevant position is the position for that the agent is most likely to reach in the short term, which corresponds to the closest point of the wall. We thus define the *pertinence* of an object instance according to the certitude of success and the distance of this instance: $\frac{S^{\sigma}_{i}(E_t)}{d(\sigma)}$, where σ is the position of the instance, and $d(\sigma)$ the distance of the instance, which corresponds to the length of the sequence σ.

We need to limit the number of stored object instances, by limiting the stored instances to the most pertinent instances. We propose a procedure that first selects the most relevant object instance contained in each place (primitive or composite), and then eliminates instances until each remaining instance is covered by at least one place that only covers this instance. This procedure ensures that, for each stored instance, there is at least one place that can characterize the position of this place and separate it from other instances of the same object.

Selected instances are then stored by the space memory in the form of a list of places $\{l_k\}$ containing all the places (primitive or composite) that contain the position of this instance. The position of the instance is then characterized by the intersection of places of this list. As composite places can characterize areas in the non-observable space, such a list can characterize the position of an object instance in the global space.

4.2.6. Object tracking based on composite place

We use the sequential aspect of composite places to keep the localization of object instances in the space memory. Indeed, when the first interaction of a composite place is enacted, this interaction can be removed from the path of the composite place. The object can then be considered as present in a new composite place composed of the previous composite place mi-

nus the first interaction. For example, if the position of an object instance is characterized by a composite place $[i_1, i_2..., i_n; l\rangle$, and if the agent enacts the interaction i_1, then the instance position can be characterized by the composite place $[i_2, ..., i_n; l\rangle$. This method allows an object instance to be followed in the global space. However, this method is limited by the length of composite interactions. We thus propose an additional mechanism that learns to define connections between places.

4.2.7. Signature of presence

Signatures of places cannot characterize the position of an object instance in the non-observable space as signatures of places are based on positions of the observable space. To overcome this limitation, we propose that the intersection of the set of places characterizing the presence and position of an object instance defines a small area of space that can be assimilated to a position in the global space. Just as for place signatures, it is possible to determine and learn such sets of places defining positions that are included in the area characterized by a composite place.

To determine whether a set of places defines a position that belongs to a composite place, we propose the following principle: the agent first considers object instances stored in its space memory. The position of each object instance is characterized by a list of places, for which the intersection defines the most probable position of the object instance. As an example, the position of an instance of the object affording the interaction i is stored by the space memory with a set of places $\{l_k\}$. Then, the agent enacts the path of a composite place l. If an instance of the object affording i is then detected in the final place of l, then the initial set of places $\{l_k\}$ may define a position that is included in the area characterized by the composite place l. Figure 10 illustrates this principle:

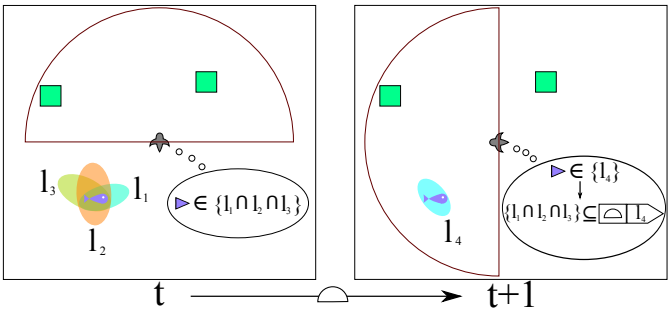


Figure 10. Principle of signatures of presence. At enaction cycle t (left), the space memory contains an instance of the object affording the interaction *eat*, for which the position is characterized by places l_1,l_2,l_3. The agent then *turns* 90° *left*. An instance of the object affording the interaction *eat* is detected in the place l_4. Thus, the composite place [*turn* 90° left][$l_4\rangle$ may contain the context of place $\{l_1 \cap l_2 \cap l_3\}$. Note that the agent cannot determine whether or not the two detected instances are the same.

We define the *context of place* $E^{\lambda}_{t,\omega}$ of an object instance ω as the list of places $\{l_k\}$ that characterize the position of ω. We define the *signature of presence* of a place l as a structure that characterizes contexts of place that define positions included in the area characterized by l. As with signatures of places, a *signature of presence* S^p_l of a place l can be formalized as a function that characterizes the certitude of presence of an object instance ω in the final place of a composite place l after enacting the path of l, according to a context of place $E^{\lambda}_{t,\omega}$ that characterizes the position of this instance, with an absolute certitude of presence when $S^p_l(E^{\lambda}_{t,\omega}) = 1$ and an absolute certitude of absence when $S^p_l(E^{\lambda}_{t,\omega}) = -1$.

The presence signature learning process is similar to the other signature learning processes. It consists in reinforcing signatures of presence, by analyzing contexts of place, to define pertinent certitude of presence of an instance ω in a place l. The signature of presence of a place l is reinforced when the path of l is successfully enacted. The reinforcement of the signature of presence of a place l compares certitudes of presence according to the initial presence context $E^{\lambda}_{t-n,\omega}$, where n is the length of the path of l, and the real presence or absence of the object instance ω in the place l. Just like the two previous signature learning mechanisms, the presence signature learning mechanism must reinforce presence signatures to minimize prediction errors and provide relevant certitudes.

While *place signatures* are used to evoke places that can characterize the position of an object instance detected in the observable space, *presence signatures* can evoke places that can characterize the position, in the global space, of an object instance stored in space memory. The principle of presence signature allows the position of an object instance to be tracked for long sequences of interactions, by continuously adding composite places to the context of place of an object instance, and is only limited by the certitude and the precision of signatures of presence.

4.2.8. Recognition of stored instances

As the agent can observe the same object instance more than once, we propose a mechanism that recognizes and updates stored instances that may correspond to instances detected at the last enaction cycle. This mechanism compares the expected positions of an object instance with the positions where new instances of the same object are detected.

As stored instance position is characterized by a list of places, we can use these places, and especially their signatures of places, to define the expected positions of an object instance. The certitude at each position is defined by the number of places of the list for which the signature contains this position. Then, for each detected instance, we find the stored instance with the greatest certitude at the position of the detected instance (when the certitude is not null). The detected instance then replaces the stored instance.

4.2.9. Illustration of the principle of the space memory mechanism

The following situation was observed during our experiments, and illustrates how the space memory can exploit presence signatures to track an object escaping from its sensory system. We begin with the configuration displayed in Figure 11. In this configuration, there is a prey on the left of the agent (we call *A-prey*) and a prey on its right (we call *B-prey*). During this experimental run, we observed that the agent showed

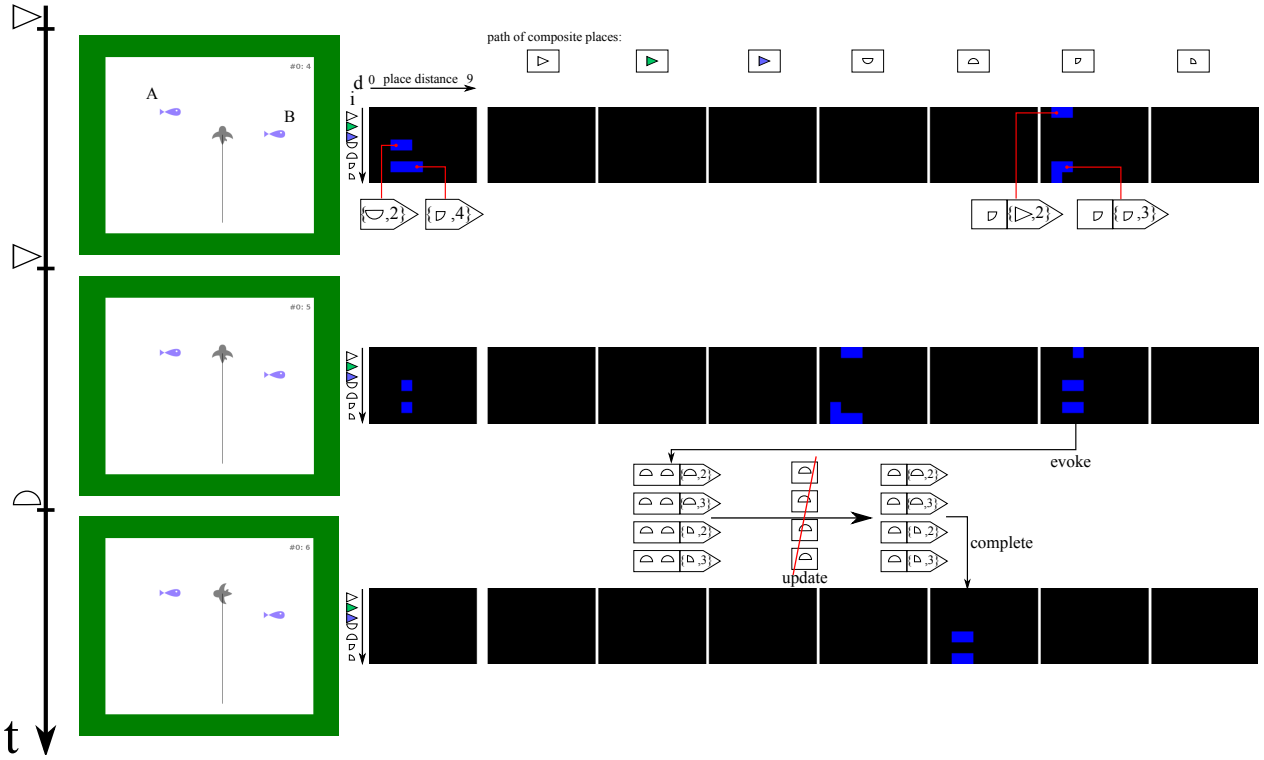


Figure 11. Update of the estimated position of an object with presence signatures. From left to right: the enacted primary interactions at each enaction cycle, the agent in its environment, and the list of places that characterize the position of B-prey. The list of places is organized in the form of a context that gives the state of all places (places that are in the list of places that characterizes the position of the prey are blue, other places are black), to be compared with presence signatures (see Section 5.3.3). This context is organized as follows: primitive places are organized according to the interaction and distance that characterize them (left rectangle). Composite places are organized in a similar way, according to their final path, and separated according to their path (seven right rectangles). A blue square means the object position is characterized by the corresponding place. In the first step, position of B-prey is characterized by ten places (we give some examples of places). In this configuration, we exploit the fact that the agent prefers preys on its left side when the difference in distances is small, due to asymmetries in its signatures. The agent moves forward and turns left to reach A-prey. At this time, the places that characterize the position of B-prey are removed, as none of them begin with *turn* 90° *left*. However, signatures of presence evoke a set of four composite places (presented in Figure 24) that begin with *turn* 90° *left*. The estimated position of B-prey is then updated by removing the first element of the path of these four places, as this element was already enacted by the agent. At the end of the enaction cycle, the position of B-prey is characterized by a set of four composite places with a path that begins with *turning* 90° *left*. An interesting point is that the position of B-prey was initially characterized by *turning* 90° *right*, while now it is characterized by *turning* 90° *left*.

a preference for preys that are on its left side when the difference in distance is small. This preference is probably due to asymmetries in its signatures. We exploit here this characteristic to observe how the space memory encodes the position of B-prey when it disappears from the visual field of the agent. Attracted by the A-prey, the agent moves forward. Then, B-prey is not longer visible, and its position is characterized by a set of places. These places indicate that B-prey can be reached by *turning* 90° *right* and *turning* 45° *right*. This context is compatible with presence signatures of four composite places (see Section 5.3.3 and Figure 24 for their signatures). These four places can thus be added to the list of places that characterize the position of B-prey. The agent then enacts *turn* 90° *left* to reach A-prey. The places that characterize the position of B-prey are removed, as they do not begin with *turn* 90° *right* interaction. However, the four composite places evoked in the previous enaction cycle begin with this interaction. These places are then updated. The position of B-prey is now characterized with four places which indicate that B-prey can be reached by *turning* 90° *left*. Thus, B-prey moved (in egocentric reference) from an area that can be reached by turning right to an area that can be reached by turning left: presence signatures thus link areas of the global space in terms of interaction.

4.3. Mechanism M3: Decisional Mechanism

The agent's decisional mechanism is divided into four submechanisms, as shown in Figure 12. Three of these mechanisms are defined to learn the three type of signatures. These mechanisms are similar and their learning process are implemented in the same way. The fourth mechanism exploits the space memory to generate behaviors satisfying the interactional motivation of the agent.

4.3.1. Learning mechanisms

A previous work (Gay & Georgeon, 2013) has showed that an agent equipped with our signature mechanism must be equipped with a mechanism dedicated to learn and test signatures of interaction. Indeed, when the agent constructs its signatures based on coincidences observed with a behavior generated by the exploitation mechanism, it may be locked in an irrelevant behavior that does not allow it to find out incoherences in signatures.

The agent is equipped with three mechanisms dedicated to learn, respectively, signatures of interactions, signatures of places and signatures of presence. As each type of signature depends on the previous types, each mechanism takes priority over the next mechanisms.

As the signature learning process is similar for all signatures, the three mechanisms are based on the same principle. The first mechanism (here, the signature learning mechanism) first computes the certitude of its structures, and selects the structure with the lowest certitude (in absolute value). Indeed, the structure that has the lowest certitude (in absolute value) can be considered as the least predictable structure in the current context, and thus the structure for which the signature may learn the most if tested. If this certitude is lower than a certain threshold, we call *learning threshold*, then the mechanism proposes this structure and the agent tries to enact it. If no structure can be elected, the next mechanism repeats this procedure. The process is repeated until a learning mechanism proposes a structure to test or until all the mechanisms have been tested. In this second case, the agent uses the exploitation mechanism, described in the next section (4.3.2).

In the case of the mechanism dedicated to the interaction signature learning process, we need to consider that a secondary interaction can only succeed or fail when its associated interaction is successfully enacted: a secondary interaction is thus a candidate when its associated interaction is considered as enactable with a high certitude (we used a threshold of 0.8). We also need to consider interactions for which the object cannot be observed, and thus for which the result cannot be predicted.

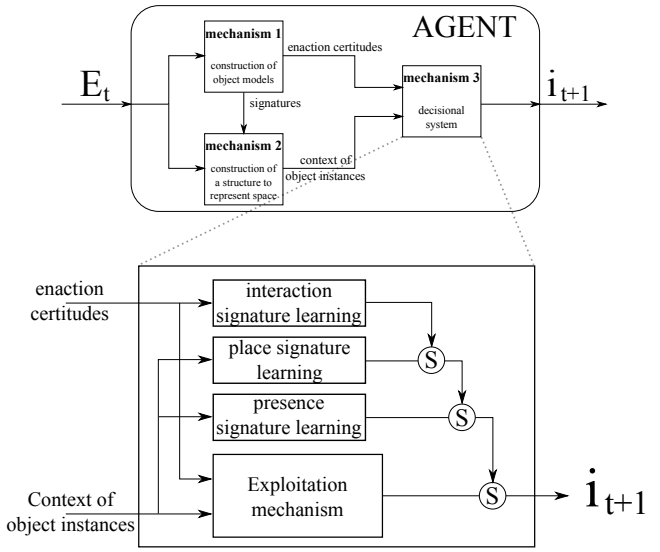


Figure 12. Decisional mechanism. This mechanism is composed of three learning mechanisms that test and learn (in the following order of priority) signatures of interaction, signatures of places and signatures of presence. When a signature is considered as unreliable in the current context, the mechanism proposes to test the associated interaction or place. When none of the learning mechanisms propose an interaction or a place, the exploitation mechanism determines the interaction that best satisfies the interactional motivation of the agent in the short and medium terms, according to the list of enactable interactions (given by mechanism M1) and the context of object instances (given by mechanism M2).

A secondary interaction can be selected if, after at least five consecutive correct predictions, the absolute value of the certitude is greater than a threshold (set to 0.2). Indeed, weights of the signature of an interaction afforded by an object that cannot be detected are defined according to coincidences, and thus are expected to have a low value.

The mechanisms dedicated to place and presence signature learning processes can also propose a composite place for which a part of the path was already enacted. In this case, the mechanism proposes the sequence of remaining interactions, when the certitude of the composite place is low in absolute value in a context of position or a context of place at enaction cycle $t - n$ (where n is the number of interactions of the path that are already enacted)

Note that we do not define a learning and an exploitation period: when the signatures become reliable, the learning mechanisms are less used. However, the agent retains its learning abilities during all its *life*, and learning mechanisms can be used in case of environmental changes.

4.3.2. Exploitation mechanism

Exploitation is based on the mechanism presented in Gay, Georgeon, & Wolf (2014) and adapted to be used with an agnostic space memory based on places. This mechanism consisted in measuring variations in distance of object instances produced by a candidate interaction. The variation in distance is weighted by the valence of interactions afforded by object instances, and by the distance of the instance. Thus, an instance of an object affording an interaction with a positive valence, that can be moved closer by a candidate interaction i_c, adds a positive utility value to i_c, and an instance of an object affording an interaction with a negative valence adds a negative utility value. The utility value depends on the distance of instances, so that close instances have a greater influence on the decision. Experiments with a hard-coded space memory showed that this mechanism was robust enough to be implemented on a robot, despite the low precision of the memory.

The utilization of places simplifies the exploitation mechanism: indeed, each place carries the two types of information required, i.e. the distance and the interaction(s) making it possible to move closest the most to the object instance:

- primitive places contain the distance and the interaction by definition.

- The distance of a composite place is the sum of the distance defined by its final place and the length of its path. The interaction making it possible to move closest the most is the first interaction of the path.

However, the path is not necessarily the shortest sequence to reach an object. We thus need to estimate the distance of an object to eliminate irrelevant paths and thus only take interactions that allow to move closer to the object instances into account. We propose using the certitude of places: the places that characterize the position of an instance are ordered by increasing distance. For each distance, we select the place with the greatest certitude, noted as $c_{d,max}$. We then select the first local maximum of certitude, noted as d_{max}, defined by

$$d_{max} = \min_{d \in \mathbb{N}}(d) / \{c_{d,max} > c_{d-1,max} \wedge c_{d,max} \geq c_{d+1,max}\} \quad (8)$$

Distance is then estimated as a sum of distances of places for which the distance $d_l \in [d_max - \epsilon; d_max + \epsilon]$ (we used $\epsilon = 1$ in our implementations), weighted by the certitudes of presence c_l of these places.

We consider that the variation in distance produced by a candidate interaction is always -1 interaction, as the distance of an instance is defined in terms of number of interactions. We note ω an instance of the object affording i_ω stored in memory M, and R_ω the list of interactions making it possible to move closer to instance ω. The utility value u_{i_c} given to each candidate interaction i_c is given by:

$$u_{i_c} = \sum_{\omega \in M} \begin{cases} f(d_\omega) \times \nu(i_\omega) & if \;\; i_c \in R_\omega \\ 0 \quad else \end{cases} \quad (9)$$

Where f is a function that reduces the influence of object instance according to their distance, and $\nu(i_\omega)$ is the valence of i_c. In our implementation, we used the function $e^{-\gamma d}$, where γ is the *object coefficient* that characterizes the influence variation according to the distance of an object instance. With a low object coefficient, distant objects may have a greater influence than close objects. With a high value, the agent may not consider distant objects. Figure 22 shows how the object coefficient can affect the agent's behavior.

We need to consider that an object instance at a distance of 1 must not be taken into consideration in the utility value, as the agent can directly interact with it. Indeed, if the agent interacts with an object, the object may be modified by the interaction and vanish from the space memory. This disappearance could be considered as a positive or negative event, according to the valence of the afforded interaction. However, this event is irrelevant as the agent can enact the afforded interaction and experience the valence of the interaction.

The variation of global proximity is then used to compute the global satisfaction value of a candidate interaction:

$$\nu'(i) = \nu(i) + \beta \times u_i \quad (10)$$

where β is the *influence coefficient* of the space memory. This coefficient characterizes the influence of variation produced by the enaction of a candidate interaction on the environment (long-term decision) with respect to the satisfaction value of this interaction (short-term decision). With a low coefficient, the agent will not consider distant object instances. With a high coefficient, the agent will not consider the satisfaction value of its interactions. In the implementations proposed in this paper, the coefficient β is predefined. Defining a variable coefficient that depends on agent internal states is an open question that we intend to study in future work.

The mechanism then selects the candidate interaction with the greatest global satisfaction value. The decision thus considers short term satisfaction (through interaction valence) and future satisfaction (through variation in distance of object instances).

5. Experiments

We tested our mechanisms on the agent described in Sections 3 and 4. We propose to test the different mechanisms separately to observe their properties more precisely. When a whole sensorimotor loop is needed, we implement simplified or hard-coded versions of the other mechanisms based on observations of these mechanisms. The versions of the agent and the corresponding mechanisms are displayed in Figure 13. Testing mechanisms separately makes the analysis of learned structures and emergent behaviors easier to understand and interpret. The downside of this approach is that we cannot observe side-effect that can result from a simultaneous utilization of the different mechanisms. Conclusion section discusses this possibility.

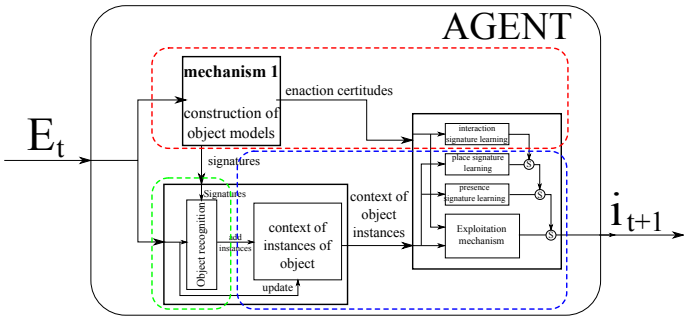


Figure 13. The successive experiments test the different sub-mechanisms of the agent's decisional system. The first experiment tests the mechanism M1 with the interaction signature learning process (red). The second experiment focuses on the properties of the distant object detection sub-mechanism (green). The last experiments use structures obtained in previous experiments and simplified versions of other mechanisms based on properties observed in previous experiments to investigate properties of the space memory and emergent behaviors produced by the exploitation mechanism (blue).

The first experiment tests the interaction signature construction mechanism (M1). We implemented the interaction signature mechanism M1 and the interaction signature learning submechanism of the decisional system M3 to generate a complete sensorimotor loop. As the interaction learning process is the first learning stage of the agent, we do not need the space memory and the exploitation mechanism. This experiment is designed to observe properties of signatures of interactions.

The second experiment tests the object recognition submechanism of the construction mechanism of a structure to characterize space (M2). We needed to test this sub-mechanism separately as it is the most CPU consuming. As the object recognition mechanism is not based on a learning process, we only test object detection in several environmental contexts rather than observing an emergent behavior. We used the signatures of interactions learned in the previous experiment.

The last experiments test the space memory (sub-mechanism of M2) and the decisional system (M3). We used a simplified interaction signature mechanism and object detection submechanism, based on observations of the first two experiments. These experiments investigate the learned structures and emergent behaviors by observing how the agent interacts with its environment in different environmental contexts.

5.1. Signature of interactions learning

This experiment tests the interaction signature learning process, by analyzing properties that emerge from signatures of interactions. Although this experiment was described in a previous work (Gay, Georgeon, & Wolf, 2014), we give in this paper more results to highlight properties that emerge from signatures.

We equipped the agent with the mechanism dedicated to the interaction signature learning process described in Section 4.3.1, and let the agent learns signatures of its interactions. The signatures of primary interactions emerge and stabilize after nearly 4000 to 5000 enaction cycles, and signatures of visual interactions begin to emerge after 8000 enaction cycles. While signatures of visual interactions begin to stabilize, elements are progressively removed from the environment to let the agent experience visual interactions related to distant positions. Figure 14 gives some examples of signatures of interaction obtained after 20 000 enaction cycles. We can observe that interaction *move forward* is related to the absence of green and blue objects (mid-red blob) in front of the agent, *bump* is related to a green object in front of the agent and *eat* is related to a blue object in front of the agent. We can also note that the size of blobs is equivalent to the size of the agent. Thus, these signatures can define certain spatial properties of the agent's body schema. We can observe that *move forward* and *eat* are related to the interaction *bump* with a negative weight, while *bump* is related to itself with a positive value. Indeed, when the agent bumps into a wall, the wall stays in front of the agent, which makes the *bump* interaction possible again. Signatures can thus integrate non-visual information, and associate multi-modal information that characterizes the same object. Signatures of *turn* interactions (not represented on the figure), that cannot fail, are strongly related to the bias of the formal neuron that composes the signature, indicating that the result of *turn* interactions does not depend on the context.

Signatures of visual interactions show that an interaction related to a certain position p and a color c is related to an element with the same color c, located at a position p' that corresponds to the position p moved by the transformation produced by the primary associated interaction: a translation for visual interactions associated with *move forward* and a rotation for the visual interactions related to *turn* interactions. Figure 15 shows these transformations, by drawing a link between the position associated with a visual interaction and the barycenter of the positions of the higher weights of its signature, which allows observation of the geometrical information defined by signatures of interactions. This difference of position shows that spatial transformations produced by interaction can be defined in the observable space through signatures of interactions. Table 2 summarizes average geometrical transformations obtained for each group

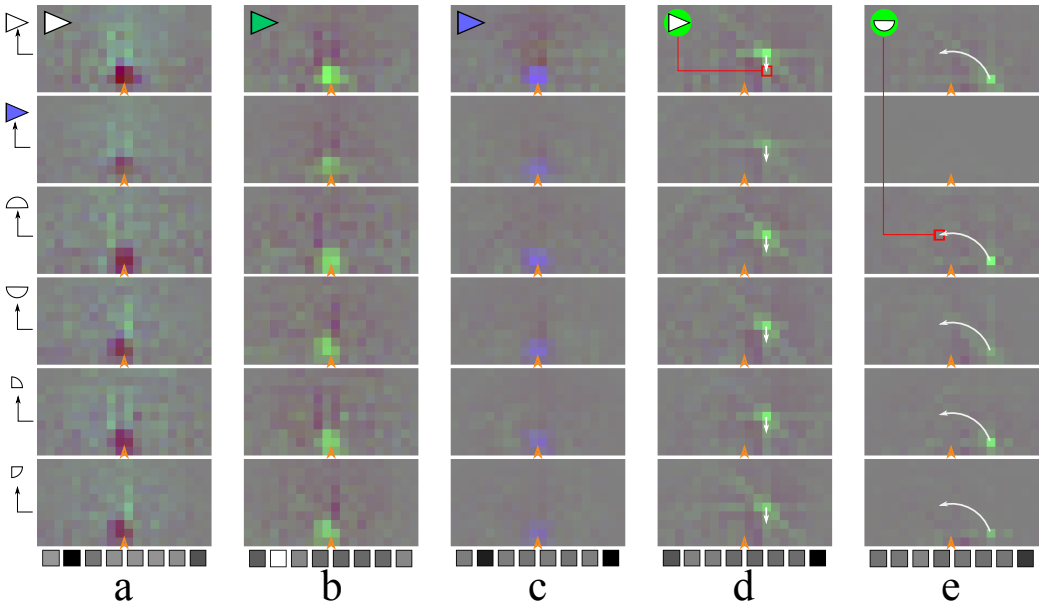


Figure 14. Signatures of interaction of *move forward* (a), *bump* (b) and *eat* (c), *seeing green at position given by the red square while moving forward* (d) and *seeing green at the position given by the red square while turning* 90° *right* (e). Interactions displayed on the left show the associated primary interaction of each group of secondary interactions. The position of the agent (according to positions of visual interactions) is given by an orange arrow. Weights of each signature are normalized according to the greatest weight (in absolute value) of the signature. On each color channel, a value of 1 means a normalized value of 1 and a value of 0 means a normalized value of -1. In the case of primary interactions, a white square means a normalized weight of 1 and a black square, a normalized weight of -1. We can observe that the signature of *move forward* relates to the absence of green and blue in front of the agent (mid-red blob), *bump* is afforded by a green object in front of the agent and *eat* is afforded by a blue object if in front of the agent. *Bump* is also related to itself, which means that *bump* can be enacted repeatedly. The secondary interactions are related to an element of the same color. The position of this element corresponds to the position of the interaction, moved by the transformation produced by the associated primary interaction (i.e. a translation and a rotation). As an example, if the agent has enacted, at the previous enaction cycle, the interaction *move forward* and a visual interaction consisting in seeing green at the right of the agent, then the interaction *seeing green at the position given by the red square while turning* 90° *right* (e) may be successfully enacted. The signature (e) also indicates that this interaction can be considered as successfully enactable if the agent has, as an example, turned left and seen green on its right. These offsets show that signatures can define the geometrical properties of observable space.

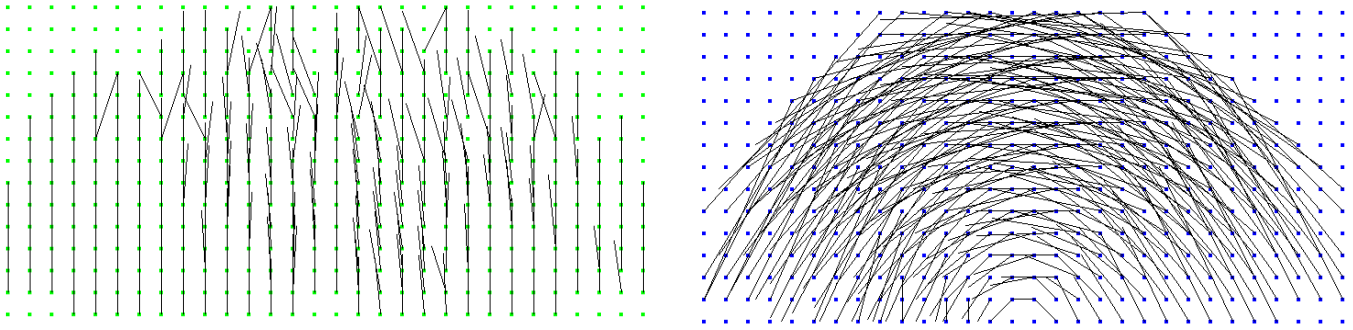


Figure 15. Transformations produced by enacting primary interactions. We display these movements by drawing a link between interactions and the barycenter of their signatures (we only take highest weights, in absolute value, into consideration). Left: visual interactions related to *seeing green while moving forward*. Right: visual interactions related to *seeing blue while turning* 45° *left*. The translation and the rotation are clearly recognizable. Note that the agent cannot access this representation, as it ignores positions associated to visual interactions.

Table 2. Average geometrical transformations discovered through signature of interactions, for each group of visual interaction: translations on x and y axis, rotation on z axis, as an external observer can observe. Distance unit is defined according to the distance covered by enacting *move forward*. Transformation are very close to real transformations (translation of 1 unit on y axis and rotations of 90° and 45°). We do not compute the average transformation for visual interactions associated with *eat* as too few of them are considered as reliable.

associated interaction	green	blue	red
forward	−0.033; 1.003; −0.15°	−0.030; 1.005; 0.13°	−0.031; 1.012; −0.12°
eat			
left 90°	0.235; −0.075; 89.43°	0.190; −0.159; 89.87°	0.154; −0.100; 89.35°
right 90°	−0.284; −0.086; −89.12°	−0.191; −0.154; −89.93°	−0.189; −0.125; −89.64°
left 45°	0.168; −0.013; 44.39°	0.141; −0.043; 44.77°	0.150; −0.033; 44.73°
right 45°	−0.195; −0.004; −43.84°	−0.137; −0.034; −44.84°	−0.134; −0.033; −45.01°

of visual interactions.

Another interesting observation is that the weight patterns are similar for each group of visual interactions (except for interactions associated with *eat*, as this interaction is less frequently enacted than the others, and because *eat* needs a prey in front of the agent, which makes some of the visual interactions impossible to experience). This means that the signature of an interaction i allows to cluster interactions enabling detection of the object affording i, even though these interactions cannot be enacted simultaneously. For example, enacting *move forward and seeing a green element in p* is equivalent to enacting *turn* 90° *left and seeing a green element* in the same position p (the agent initially ignore that it is the same position) because the enaction of one of these two interactions makes possible the enaction of another interaction *move forward and seeing a green element in p'*. This property is observed at each position p of the observable space.

In Figure 16, we represent visual interactions using points. When a signature designates two visual interactions with weights of the same sign, the points corresponding to these two visual interactions are gathered. To help an external observer to identify signatures, we set positions of visual interactions associated with the interaction *move forward* to match their real positions in space. Then, we draw links between interactions re-

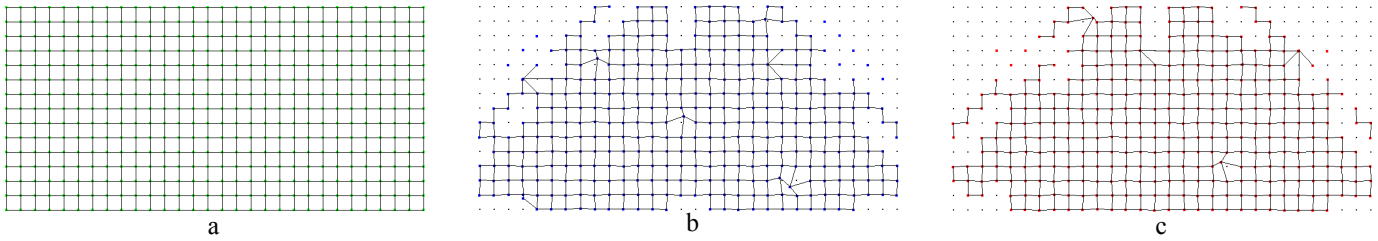


Figure 16. Relations between signatures. When a signature designates two interactions with weights of the same sign, these two interactions are gathered. To help an external observer to identify interactions, we set positions of interactions associated with *move forward* as a reference (a). Neighboring positions are linked by a line to define a regular mesh. Groups of interactions are represented separately (although they are overlapping). We display the structure obtained with visual interactions related to *seeing blue while turning* 90° *left* (b) and *seeing red while turning* 45° *right* (c). The reference mesh is recognizable: interactions related to the same position are clustered. The average error among reliable signatures is 0.034 unit of the reference grid. We can observe that the meshes are limited to positions that the agent can experience in its environment: other positions are too far to be experienced in our test environment.

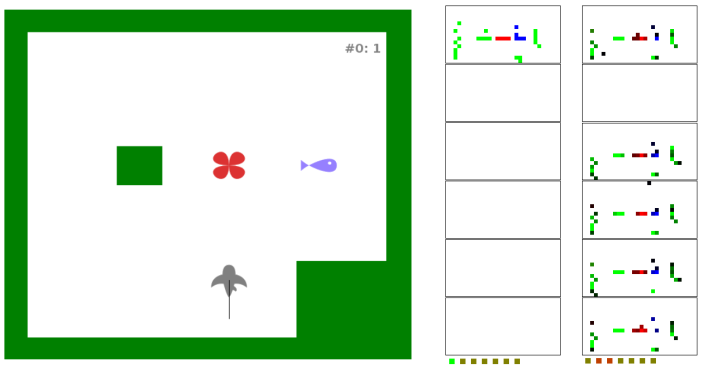


Figure 17. Completed context. Interactions that are considered as enactable are used to complete the environmental context E_t (left context) by applying their signatures. In this example, we can observe in the completed context (right context) that the pattern of enacted interactions is similar for each group of visual interaction (except for interactions associated with *eat*, which are not reliable enough). The color shade indicates the certitude of predicted interaction: the lighter the color, the greater the certitude. The completed context (right context) can be used to define the enactability of an interaction with better accuracy.

lated to neighbor positions. Visual interactions related to *move forward* thus constitute a reference grid. As can be observed in Figure 16b and c, this reference grid appears in each group of visual interactions, indicating that visual interactions associated with the same position are clustered. It is thus not necessary to define *a priori* the positions of visual interactions to define interactions that characterize the presence of the same element, which can be useful on a real optic flow system, where the measured relative speed of the same object depends on the movement of the agent.

The ability to cluster interactions that characterize the presence of the same element can be used to complete the interactional context E_t by adding interactions that would be enacted if their intention have led to the same environmental context. The fact that two interactions associated with different primary interactions can be considered as related to the same element makes it possible to consider the simultaneous enaction of these interactions. Indeed, if an interaction is considered as enactable, the element affording it can be considered as present. Then, every interaction that allows this element to be detected can be considered as enacted. Figure 17 gives an example of a completed interactional context.

Signatures of interaction can thus be used to define enactability of interaction, and then to define the consequences of geometrical transformations produced by interactions in the observable context. This second property is used by the distant object recognition mechanism to recognize distant elements in the observable space.

5.2. Recognition and localization of distant objects

We add the object recognition mechanism to our agent, and reuse signatures obtained with the experiment described in Section 5.1. This mechanism has two functions: first, it computes *predecessors* of signatures according to a set of sequence of interactions. Then, it compares these predecessors to the current interactional context to detect object instances.

An instance of the object that affords an interaction i is considered as present at a position σ if $S_i^\sigma(E_t) > \mu$ (where $\mu = 0.9$), where S_i^σ is the predecessor of S_i through sequence of interaction σ, and if the opposite interactions $\{j_k\}$ of i are considered as failures at this position (i.e. $S_{j_k}^\sigma(E_t) < -\mu$). This condition allows removal of positions σ for which the afforded interaction is ambiguous.

We propose to remove what we call redundant positions to reduce the number of positions to consider. Two positions are considered as redundant when one of these positions is longer than the other one and if a certain proportion (80%) of weights considered as relevant (absolute value higher than 5 and higher than 1/3 of the highest weight of the signature) of the signature predecessor of the longest distance are included in the predecessor signature associated with the shortest position. Then, it is possible to remove the longest position. This principle ensures that the position of an object instance is defined by the shortest sequence of interactions.

We then define the most relevant positions to remove positions that lead to the same element of the environment. We consider that two positions leads to the same element if their predecessor signatures share a certain proportion of weights considered as relevant (20%). It is then possible to bundle together positions that are related to the same element of the context when this element covers a large surface of the environment. The positions of such *position bundles* are characterized by the distance and the first interaction of the sequences that compose them. For each bundle B, we define the most relevant transformation σ, defined by (11), to find the distance of the detected object instance.

$$\max_{\sigma \in B}(S_i^\sigma(E_t) \times \varphi^{d(\sigma)}) \tag{11}$$

Table 3. List of defined object instances in the configuration displayed in Figure 18. From left to right, the afforded interaction, the interactions making it possible to move closer to the object instance, the distance of the instance, the shortest and most certain places, and the certitude of success of the afforded interaction. The first instance has a null distance as the interaction *move forward* can be immediately enacted.

afforded interaction	interactions	dist.	main positions
		0	
		3	
		4	
	{, }	6	
	{, }	3	
		4	
		5	

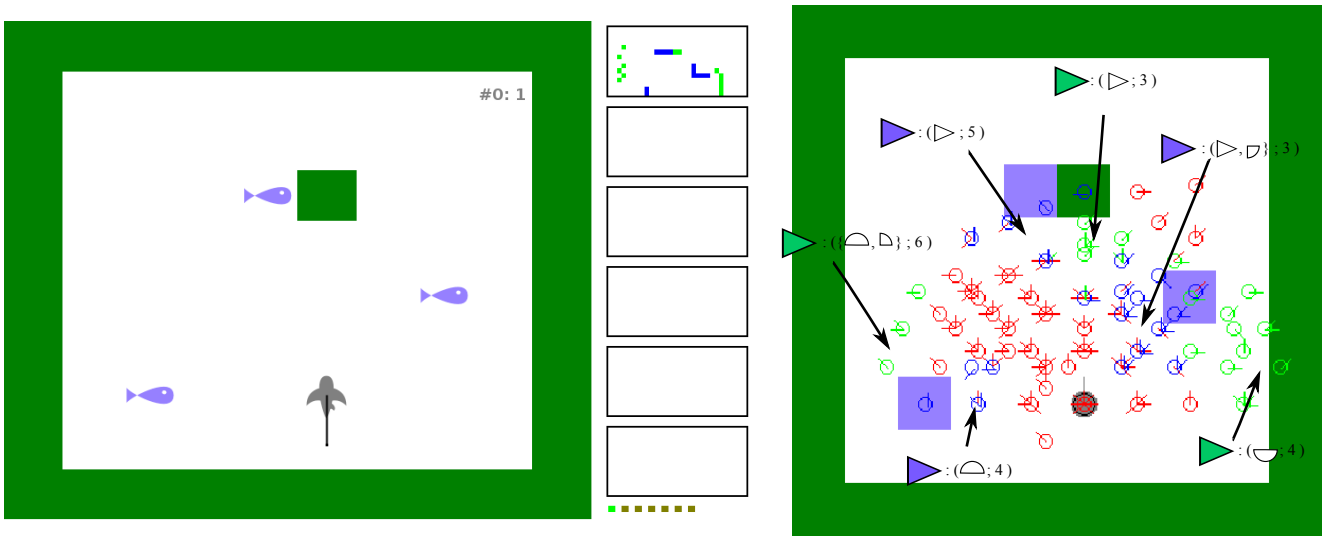


Figure 18. Object instance detection. From left to right: the agent in its environment, the environmental context (in organized form), and the position where objects are detected, with a sequence limited to 7 interactions, after removing redundant positions. The positions on this representation are computed according to transformation produced by interactions of the corresponding sequences (the agent cannot access this representation). The environmental context shows that the agent succeeded to *move forward* (green square at the bottom) and experienced several secondary visual interactions (top group). The positions in which an instance is detected are displayed with circles, and a line shows the orientation: red for instances of the object affording *move forward*, green for *bump* and blue for *eat*. The positions and orientations of objects in the right representation are determined according to the theoretical movements produced by interaction (the agent cannot access this representation). In this configuration, the instance detection mechanism found one instance of *move forward* (that can be directly enacted), three instances of the object affording *bump* and three instances of the object affording *eat*. Arrows show the approximate positions of these object instances, designated in the form *afforded interaction, (interaction(s) making it possible to move closest most, distance)*

where $d(\sigma)$ is the distance defined by the length of sequence σ, and $\varphi \in [0;1]$ a coefficient that balances the influence of the distance and of the certitude. As variations in certitudes according to distance are very small, we use a high value (0.99) in our experiments.

We tested this mechanism in several environmental configurations where elements are placed around the agent. The memory range is limited to a distance of 7 interactions.

In Figure 18, we set 4 elements around the agent, in addition to the surrounding border. We can observe the positions in which object instances are recognized, after removing redundant positions.

After filtering bundles, the mechanism found an object affording *move forward* that corresponds to the large empty area in front of the agent, three instances of the object affording *bump*, a first one at a distance of three interactions, that can be moved closer by *moving forward* (wall square in front of the agent), a second one at a distance of 4 interactions, that can be moved closer by *turning* 90° *right* (right border) and a third one at a distance of 6 interactions, that can be moved closer by *turning* 90° *left* and by *turning* 45° *left* (left border). The mechanism also detects three instances of objects that afford *eat*: the first one at a distance of 3 interactions, that can be moved closer by *moving forward* and by *turning* 45° *right*, the second one at a distance of 4 interactions, that can be moved closer by *turning* 90° *left*, and the third one at a distance of 5 interactions, that can be moved closer by *moving forward* (Table 3). This context of object instance actually characterizes the environmental context of the agent.

We also tested the precision of the mechanism by comparing

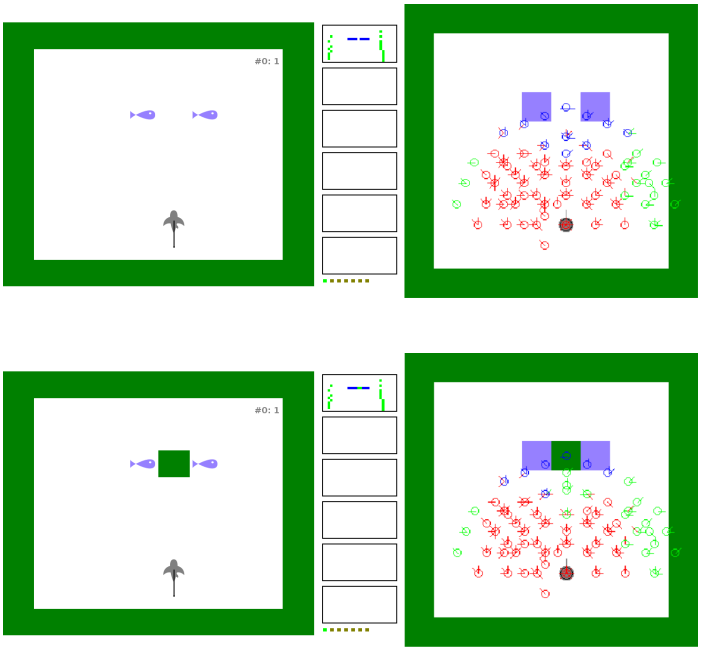


Figure 19. Precision of the object instance segmentation according to surrounding elements. Top: the agent considers the two preys as a unique instance of the object affording *eat*. Indeed, the two preys are at the same distance from the agent and can be moved closer by enacting *move forward*. Bottom: if a wall is set between the two preys, the agent considers the two preys separately and with different distances. The presence of the wall invalidates several positions for the preys, which separates the two preys. The difference in distances indicates that the agent has a preferred side, for which signature predecessors are more reliable.

Table 4. List of defined object instances in the configuration displayed in Figure 19 (top). The agent considers the two preys as a unique object affording *eat*, "in front of it" (reachable by enacting *move forward*.

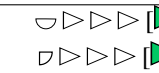
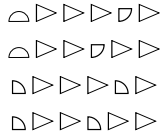
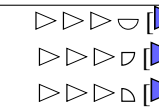
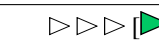
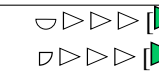
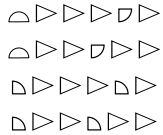
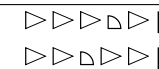
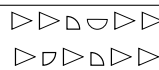
afforded interaction	interactions	dist.	main positions
		0	
	{,}	4	
	{,}	6	
		4	

Table 5. List of defined object instances in the configuration displayed in Figure 19 (bottom). The agent now considers preys as two distinct objects. The presence of the wall block invalidates the enaction of interaction *eat* at positions between the two preys, which divides remaining positions into two distinct groups of positions.

afforded interaction	interactions	dist.	main positions
		0	
		3	
	{,}	4	
	{,}	6	
		5	
		6	

the detected object instances in configurations where elements of the environment are spatially close. In one of these configurations, displayed in Figure 19, we set two preys in front of the agent, separated by an empty space. The agent apparently considers these two elements to be the same object, which is normal as these elements are at the same distance and can be moved closer with the same interaction (*move forward*). However, if a wall is set between the two preys, the agent considers the preys separately: the wall generates a strong separation between the two preys, by invalidating some object instances near the middle of the two preys.

The object segmentation mechanism thus takes surrounding elements of an object into consideration. We can also note that the two preys are not considered to be at the same distance. This shows that positions on a certain side are better known by the agent, and thus more accurate. The experience the agent has with its environment generates a form of individuation of the agent.

5.3. Construction and exploitation of a space memory

We test the space memory mechanism (M2) and the exploitation mechanism (M3) on an agent equipped with a simplified version of the signature mechanism (M1) and object detection mechanism. These simplified mechanisms are based on observations from the previous experiments (Sections 5.1 and 5.2) and define the same properties.

In this section, we first focus on the structure learned by the agent (signatures of place and signatures of presence) and on spatial properties that emerge from these structures. Then, we observe emergent behaviors generated by the decisional system, by analyzing how the agent interacts with elements of its environment, and the influence of the space memory on the agent's behavior.

5.3.1. Agent simplifications

We propose a simplified version of the signature mechanism. A first simplification is based on the observation that signatures can cluster interactions that are related to the same object, as shown in Section 5.1. We thus propose to merge visual interactions that are related to the same position and the same color, regardless of their associated primary interaction. For example, *move forward and see a green element at position* p_1 is equivalent *to turn* 90° *left and see a green element at* p_1. This simplification significantly reduces the number of interactions and increases their frequency of enaction. The downside is that we cannot define the result of visual interactions or select them as an intended interaction, as they can be enacted as a consequence of more than one primary interaction. However, visual interactions do not influence the decision mechanism as their valences are zero. It is thus not necessary to define their signatures. We increase the resolution of the visual field to define a grid of 50×100 visual positions p, which allows more accurate observation of the properties of objects defined by the agent.

This simplified signature mechanism was tested with a hard-coded space memory in a previous work (Gay, Georgeon, & Wolf, 2014). The signatures of primitive interaction stabilize after only 2000 enaction cycles. Figure 20 shows these signatures which we will use in the next experiments. The objects defined by these signatures are similar to the objects defined by the signatures obtained with the signature mechanism tested in Section 5.1: *move forward* is related to the absence of wall and prey in front of the agent (mid-red blob), *bump* is related to a green object in front of the agent and *eat* is related to a blue object in front of the agent. The higher resolution allows more accurate observation of the objects and, in particular the shape and constitution of these objects. It appears that the object affording *eat* is not only related to a blue object, but also to the absence of green element on both sides of the blue blob (mid-red blobs): indeed, a wall corner next to a prey can prevent the agent from reaching the prey. The objects defined by the agent are thus related to its experience of its environment and can differ from the objects defined by an external observer.

A second simplification consists in using geometrical translations rather than sequences of interactional transformations to define positions in space. This simplification allows a continuity between positions (and thus associated transformations),

Figure 20. Signatures of primary interactions obtained with the simplified interactional system after 2000 enaction cycles. From left to right, the signatures of *move forward*, *bump*, *eat*, *turn* 90° *right*, *turn* 90° *left*, *turn* 45° *right*, *turn* 90° *left*. Signatures are displayed in a way similar than to the complete interactional system, except that there is only one group of three overlapping color groups of visual interactions per signature, instead of six. These signatures are similar to signatures obtained with the previous agent. Signatures of *turn* interactions are strongly related to the bias of the formal neuron as these interactions cannot fail. The higher resolution shows that the signature of *eat* is not only related to a blue element, but also to the absence of green elements (purple circle arc).

thus allowing intuitive observation of signatures of places and simplifying the object instance recognition mechanism. As the objects have no specific orientation in our environment, we do not use rotation to limit the transformation set to a bounded interval of $\mathbb{Z}^2$. We define a set of transformation, for which each *position* (x, y) is equivalent to the shortest sequence of interactions σ allowing this position to be reached. We define the correspondence between positions and sequences of interactions using signatures of the three primary interactions with a non empty signature (*move forward*, *bump* and *eat*): we generate predecessors of signatures and find the shortest sequence of interaction that allows each position (x, y) to be reached. Note that we used the theoretical movements produced by interactions. As the position and shape of the object designated by each signature are similar, we obtain a similar correspondence. We select the correspondence obtained with the signature of *move forward*

We then use this correspondence between positions of space and sequences of interactions to define primitive places. We consider that *move forward* and *eat* generate the same movement. We propose to define distances using geometrical distances. This ensures that a maximum number of places are contiguous, thus simplifying place signature analysis. The set of primitive places we used to test the space memory is displayed in Figure 21.

In the following experiments, the agent begins with the set of signatures of primary interactions and the set of places defined in this section.

5.3.2. Space memory specifications

The signatures of places and of presence are implemented in a similar way as signatures of interactions, based on formal neurons. The input layer of a neuron implementing a place signature is organized as vectors $[e^{\pi}_{i,1}, ..., e^{\pi}_{i,m_\pi}]$, where m_π is the number of observable positions, where $e^{\pi}_{i,k} = 1$ when an instance of the object affording i is detected in the k^{th} position in the list of positions and $e^{\pi}_{i,k} = 0$ otherwise. The certitude of presence in a place is defined with a linear function of inputs, passed through an activation function (12). The signature S_l of a place l is characterized by the set of synaptic weights $[w_{l,1}, ..., w_{l,m_\pi}]$ and the bias $w_{l,m_\pi+1}$ of the associated formal neuron.

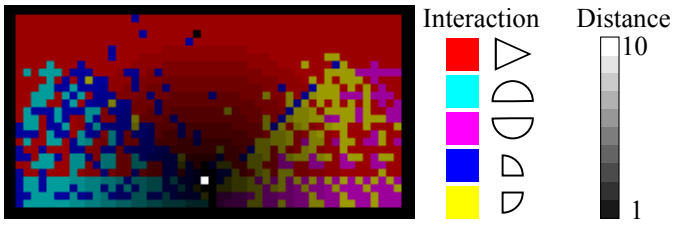


Figure 21. Set of places defined according to the signature of *move forward*. The white point shows the neutral position (no transformation). Colors show the interaction making it possible to move closest most. The color gradient shows the distance, from 1 to 10. A distance unit nearly correspond to the distance covered by enacting the interaction *move forward*.

The input layer of a neuron implementing a presence signature is organized, for each object instance ω, as vectors $[e^{\lambda}_{\omega,1}, ..., e^{\lambda}_{\omega,m_\lambda}]$, where m_λ is the number of places (primitive and composite), where $e^{\lambda}_{\omega,k} = 1$ when the object instance ω is characterized by the k^{th} place in the list of places and $e^{\lambda}_{\omega,k} = 0$ otherwise. The certitude of presence is defined with a linear function of inputs, passed through an activation function (13). The signature of presence of a place l is characterized by the set of synaptic weights $[w^p_{l,1}, ..., w^p_{l,m_\lambda}]$ and the bias $w^p_{l,m_\lambda+1}$ of the associated formal neuron.

$$S_l(E^{\pi}_i) = g\left(\sum_{k\in[1;m_\pi]} e^{\pi}_{i,k}.w_{l,k} + w_{l,m_\pi+1}\right) \quad (12)$$

$$S^p_l(E^{\lambda}_\omega) = g\left(\sum_{k\in[1;m_\lambda]} e^{\lambda}_{\omega,k}.w^p_{l,k} + w^p_{l,n+1}\right) \quad (13)$$

$$g(x) = \tanh(\frac{x}{2})$$

Signatures of places and of presence are reinforced by applying the delta rule (4), with a constant learning rate of 0.05.

The global satisfaction of a candidate interaction cannot take an object instance into consideration when the agent can interact with this instance in the next enaction cycle. We thus do not consider object instances for which the estimated distance is less than 1. As the system is not accurate, we increase this limit to 2 to ensure that a close object instance that may be interacted in the next enaction cycle will not be taken into account.

We use the following coefficients, as they provide a good compromise between the influence of close elements and the influence of distant elements:

- object coefficient $\gamma = 1$
- influence coefficient of the space memory $\beta = 15$

A modification of these coefficients does not affect the functioning of the decisional mechanism. However, it influences the emergent behaviors of the agent, as it modifies the attractiveness of object instances according to their distances. With a

low space memory coefficient, the valences of interactions have a greater influence than object instances: the agent thus selects the interaction *move forward* more often because the valence of this interaction becomes higher than the utility values defined by object instances. The object coefficient influences the attractiveness of object instances according to their distance: with a high value, distant object instances have little influence on the decision, while a low value makes the agent strongly influenced by distant object instances. Figure 22 shows an example where a wall block is closed to a prey: with a low object coefficient, the distance of an object instance has little influence on its attractiveness. In this situation, the agent is strongly attracted by the prey, but also repulsed by surrounding wall blocks, which prevents the agent from moving toward the prey. With a higher object influence, the agent moves toward the prey, as it is attractive, but remains at a reasonable distance from wall blocks. The variation of influence of object instances according to their distances enables the agent to move toward the prey without being repulsed by surrounding walls that are further away. With a high object coefficient, object instances have little influence on the behavior of the agent, which turns toward the prey only when the prey is very close to the agent.

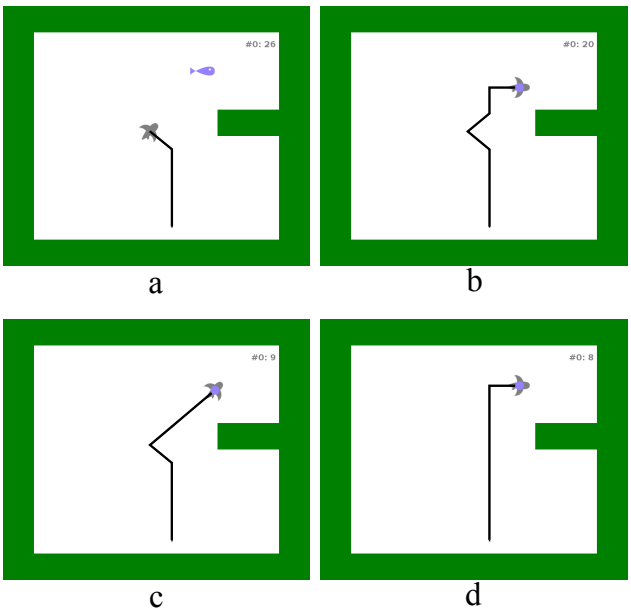


Figure 22. Influence of the object coefficient on agent behavior. a) with a low coefficient ($\gamma \in]0; 0.5]$), the distance of an object instance has a small influence on its attractiveness. In this situation, walls are strongly repulsive (although they are further than the prey), thus preventing the agent from moving forward the prey. b) and c): with a higher object coefficient (0.6 for b) and between 0.7 and 1.1 for c)), the difference in influence between the prey and surrounding walls make the agent move toward the prey until it reaches it. We can observe that the agent remains at a reasonable distance from the wall. d): with a high coefficient, the object instances have a very small influence on agent behavior: the agent is not repulsed by the wall and turns toward the prey only when the latter is very close to the agent

.

We limit the length of the path of composite places to a maximum of 2, which is the minimum length allowing updating of a position of the global space. Indeed, with a visual field of 180° and a rotation of 90°, the space memory can characterize every position of its global space with composite places with a path of length 1. With a path of length 2, it is possible to evoke a composite place l which can result in a composite place with a path of length 1 after an update of the path of l. As places with a path of length 1 are sufficient to characterize the global space, we limit the context of presence E^{λ} to primitive places and composite places with a path of length 1.

Each object instance is characterized in the space memory by two lists of places. The first list gives places that characterize the position of the instance with a high reliability. This list is updated by updating the path of composite places. Thus, this list can characterize the position of an instance for a maximum of two enaction cycles, but with a high reliability. The first list is used for the presence signature learning process. The second list gives places that characterize the position of an object instance with a lower reliability, and is updated both by updating the path of composite places and by evoking places through presence signature. This list characterizes the position of an instance for long sequences of enaction cycles, until the object is observed again or if there are no known places to evoke. The exploitation decision mechanism uses the first list as long as this list is not empty. It then uses the second list.

5.3.3. Learning structures of the space memory

We first let the agent move in its environment, driven by the place signature learning mechanism. After 50 000 enaction cycles, most of the signatures of places are stabilized. Figure 23 shows a sample of place signatures. Signature weights are organized topographically to match the position associated with each weight. A white pixel indicates a positive weight and a black pixel a negative value. A gray pixel indicates a value of 0. We can easily recognize the areas that belong to each place. We can also observe that signatures of composite places define areas that extend into the non-observable space, especially with composite places with a path composed of a *turn* interaction. This extension of places into the non-observable space shows that composite places can be used to define the presence of object instances in the global space.

Once signatures of place have emerged and stabilized, we let the agent learn presence signatures by activating the presence signature learning mechanism. The exploitation mechanism is also activated but is useless at this time as a large amount of signatures (both place and presence signatures) are unreliable. Note that autonomous activation of the presence signature learning process is still an open question. Learning reliable presence signatures takes nearly 50 000 additional enaction cycles.

The presence signatures show how places can be connected when they are related to the same area. Figure 24 shows some examples of presence signatures. These signatures show that the four displayed places can be associated with place [*turn* 45° *right*][{*turn* 90° *right*;2}⟩ and [*turn* 45° *right*][{*turn* 45° *right*;2}⟩. The similarity between these signatures indicated that these four composite places characterize similar areas in space. These signatures explain the space memory update observed in Figure 11.

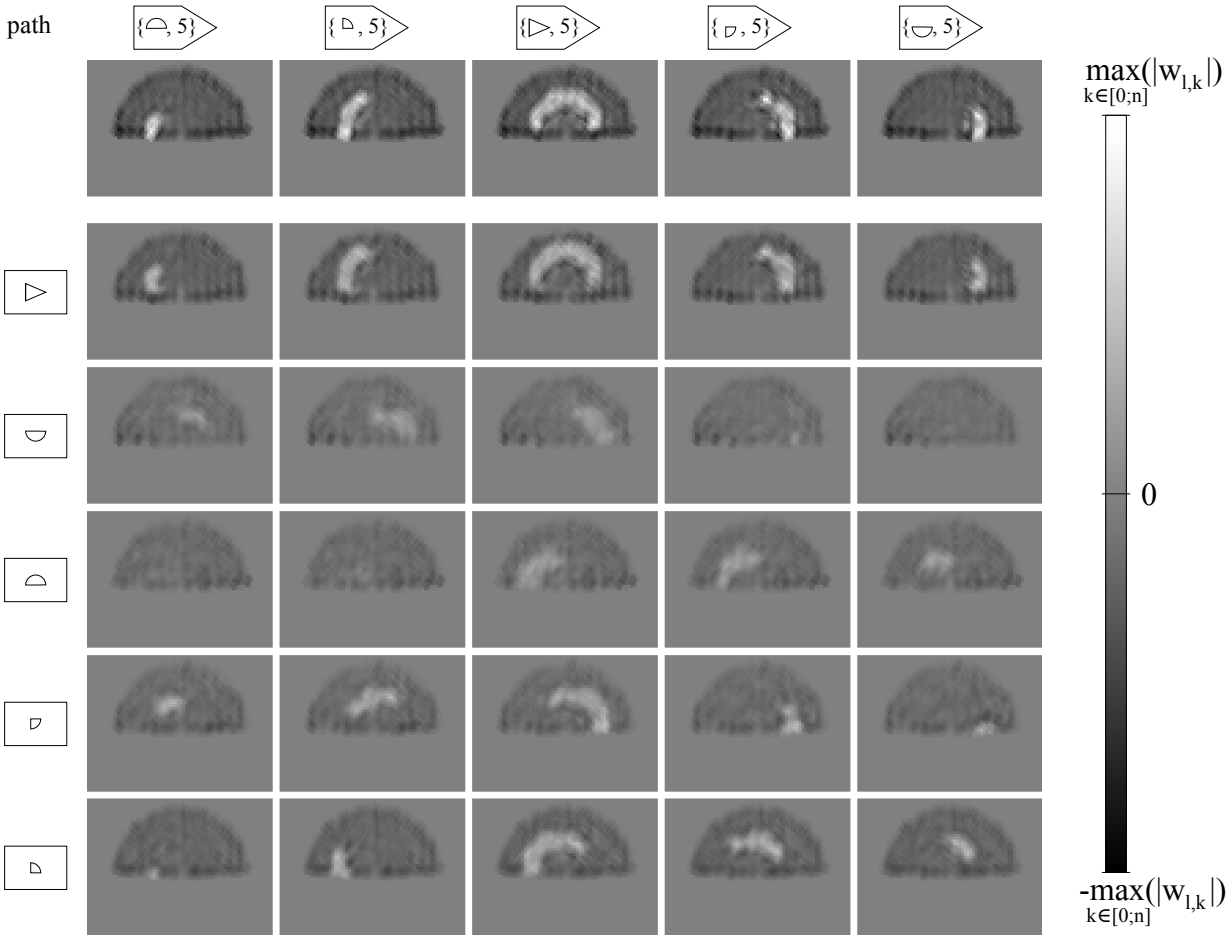


Figure 23. Sample of place signatures after 50 000 enaction cycles. The first row shows the place signatures of primitive places characterized by a distance of 5 (this distance was selected because these signatures are the easiest to read). The weights of each signature are organized to match the position of the corresponding position. The neutral translation, that corresponds to the position of the agent, corresponds to the center of the signature. White areas show positions that are considered as part of the area covered by the corresponding place, while black areas show positions that are considered as out of the area. The dark half-circle in signatures corresponds to the area covered by the visual field of the agent (the depth of the visual field is limited by the length of the environment). The area defined for each place is clearly defined. The next rows show the place signatures of composite places composed of the above primitive place and a path composed of the interactions displayed on the left. We can recognize the white areas of signatures of primitive places, which are moved according to the translation produced by the path (rotation and translation). We can guess that some of these areas extend into the non-observable space (especially with the signatures of [*turn* 90° *left*][{*move forward*;5}⟩ and [*turn* 90° *right*][{*move forward*;5}⟩). These composite places are thus suited for characterizing the presence of an object in the non-observable space.

5.4. Emergence of behaviors

The following experiments test the space memory mechanism and the exploitation decision mechanism. We thus deactivate the signature learning mechanisms so that they cannot interfere with the exploitation decision system. A first set of experiments shows how the agent interacts with elements of the environment, and thus shows how the agent *interprets* elements of its environment. A second set of experiments shows how the space memory influence the agent's behavior and guides it to elements of the environment even when they are masked after their detection.

5.4.1. Interaction with elements of the environment

We first observe how the agent considers elements of its environment and interacts with them. We exploit here the fact that the agent is attracted by preys. Indeed, the agent recognizes them as objects affording the interaction *eat*, which has a very high valence. We can thus force the agent to take another object of its environment into consideration, by placing another element the path leading the agent to the prey. We first used an alga, then a wall.

We first let the agent discover the prey, then we hide it with algae (Figure 25). It appears that the agent moves through algae as if they were not present. Indeed, algae have the same interactional properties as an empty space as they have no influence on the enaction of *move forward*, which is confirmed by signatures of primary interactions that do not integrate visual interactions related to seeing a red element: weights related to visual interactions that consist in *seeing a red element* have a low value. Thus, the agent considers that algae are (interactionally) the same objects as empty spaces. The fact that the agent considers algae as empty spaces shows that it takes affordances into consideration rather that elements of its environment. We can also observe that the agent is still attracted by

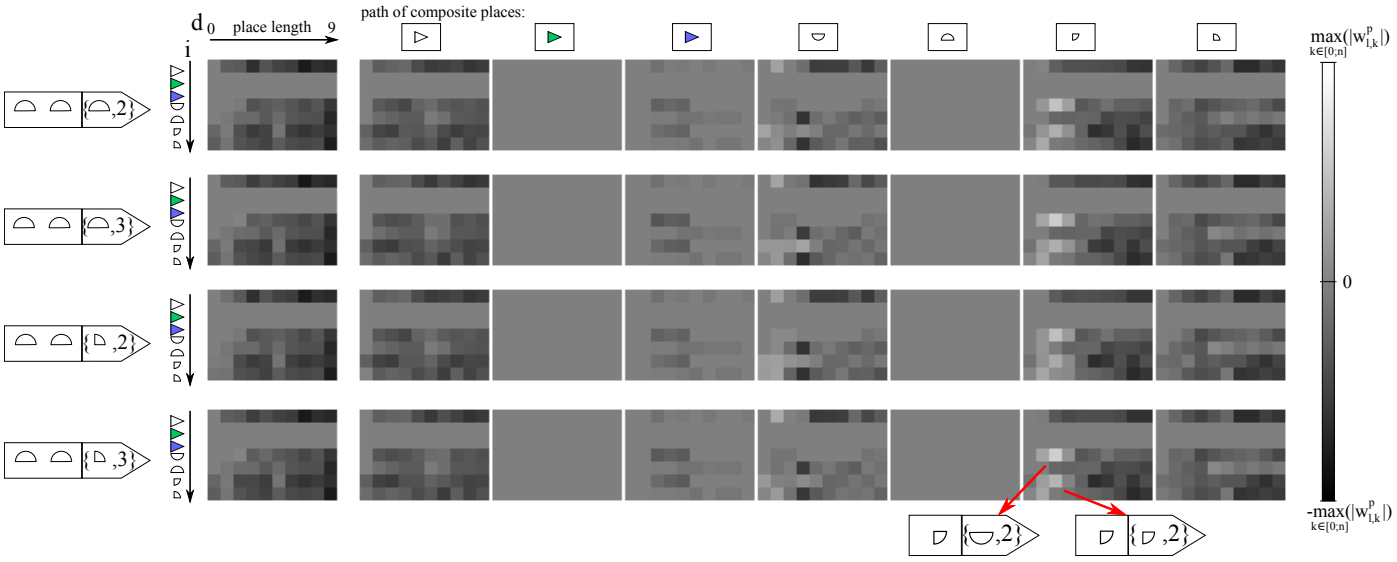


Figure 24. Sample of presence signatures. Signature weights are organized as follows: weights related to primitive places are organized in the left group according to the interaction and the distance that characterize them (left rectangle). Weights related to composite places are organized in a similar way, each group shows the weights related to composite places composed of the given path (seven right rectangles). The context is limited to primitive places and composite places with a path of length 1, as these places are sufficient to characterize the global space. White weights show the set of places that characterizes the area associated with the composite place (left), and black weights, the set of places that do not. These signatures look similar as the four displayed composite places are related to close areas. These signatures show that the four displayed composite places are mainly related to places [*turn* 45° *right*][{*turn* 90° *right*;2}⟩ and [*turn* 45° *right*][{*turn* 45° *right*;2}⟩. This means that if the position of an object instance is characterized by the two places displayed under signatures, then the four composite places may also be used to characterize the position of this object instance.

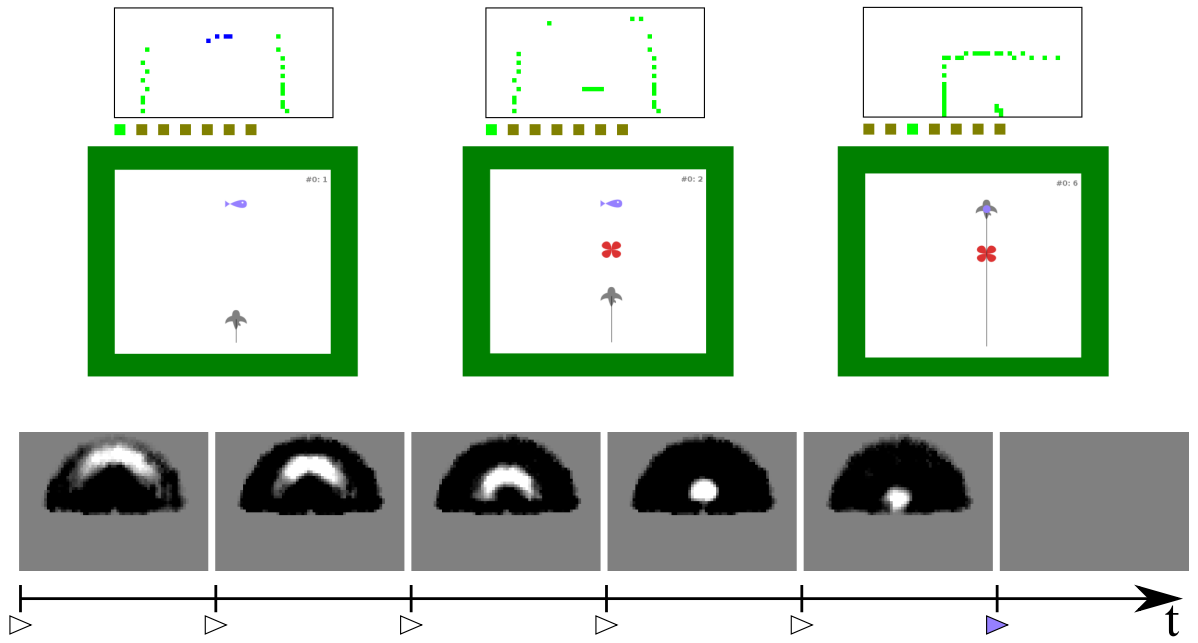


Figure 25. Behavior of the agent in presence of an alga. Top: the interactional context. We first let the agent discover a prey (left), detected as a positive object because it affords an interaction with a high valence. We set an alga between the agent and the prey, on the agent's path (middle). We observe that the agent moves through algae as if they were not present: algae become interactionally equivalent of empty spaces, as they both afford *move forward*. Bottom: the estimated position of the prey in space memory, obtained by adding place signatures of places that characterize the position of the instance affording *eat*. The prey disappears from memory when eaten: the space memory thus integrates the fact that a prey disappears from the environment when the agent eats it.

the prey, which confirms that the prey is stored in memory.

We then repeat this experiment with a wall block (Figure 26). This time, the agent turns to avoid the wall block when its influence becomes greater than the prey. The agent then turns around the wall, and when the interaction *move forward* does not make the wall come closer, the agent continues to move toward the prey (right). This behavior illustrates the principle of the space memory: during the first enaction cycles, the relative difference in distance between the wall block and the prey is small. As the prey affords an interaction with a higher valence

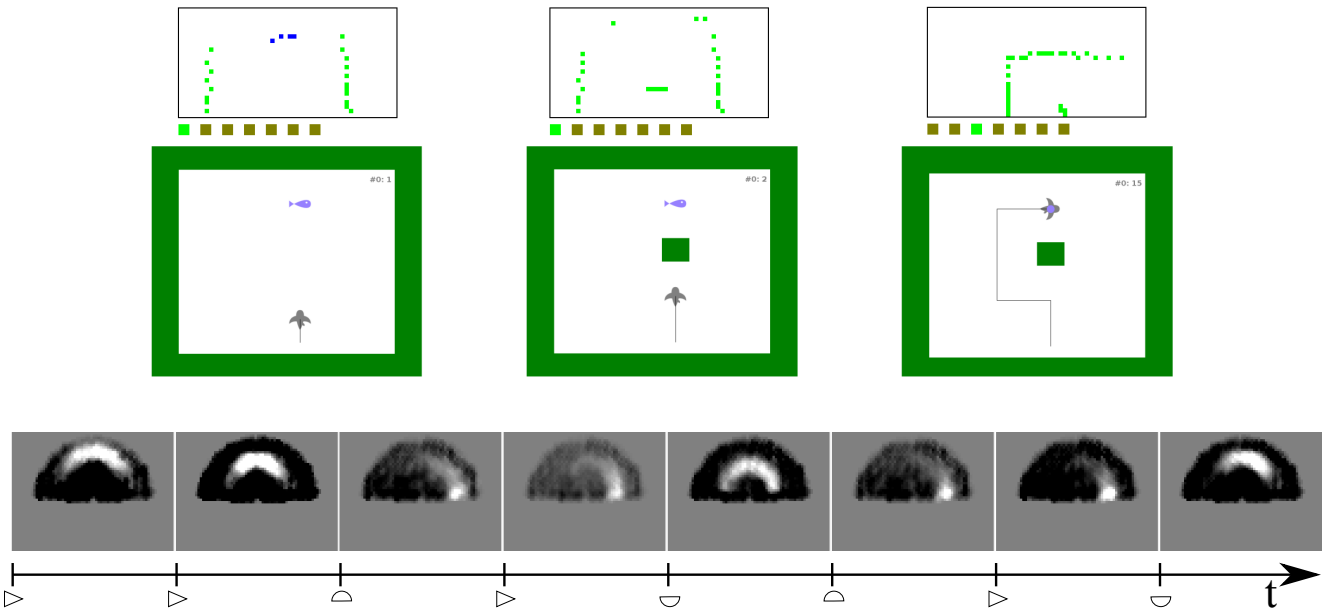


Figure 26. Behavior of the agent in presence of a wall. Top: the interactional context. We first let the agent discover a prey (left). We then set a wall between the agent and the prey, on the agent's path (middle). The agent turns to get away from the wall (right): this element is considered as negative as it affords the interaction *bump*. The agent then gets around the wall, and move toward the prey. Bottom: the estimated position of the prey according to the space memory. The timeline displays the estimated position until the agent can observe the prey again (enaction cycles 1 to 8).

(in absolute value) than the wall block, the agent is more attracted by the prey than repulsed by the wall. While the agent is approaching, the influence of the wall becomes higher than the prey, which makes the agent turn, as *turn* interactions allows it to escape from the wall. Once the agent has bypassed the wall block, the interaction *move forward* has a positive utility value, as it does not make it possible to move closer to the wall anymore. The agent thus moves forward again and reaches the prey. This avoidance behavior shows that wall blocks are now considered as negative objects. Indeed, the agent recognizes walls as objects that afford the interaction *bump*, which has a negative valence.

In these two experiments, we observe that when the agent eats a prey, this prey disappears from its memory: the agent thus integrates the fact that eating a prey makes it disappear. However, in certain configurations (see Figure 27), the agent may believe that another prey can be present beside the position of the eaten prey (the dark blob at the positions near the agent shows that there is no prey in this area). This occurs because the agent experienced several times the situation where two preys are adjacent, and considered as a unique object instance: when the agent eats one of these preys (generally the left prey as the agent has a preference for its left side), the other prey remains near the agent (generally, on its right side). We also observed that if we remove the prey before the agent reaches it, the agent, that cannot observe the absence of an object, generates an erratic behavior, wandering near the estimated position of the missing prey. Indeed, the space memory indicated the presence of a prey in front of the agent, that may disappear if the agent enacts *move forward* while the interaction *eat* is not considered as enactable. This behavior stops when the prey vanishes from the space memory.

5.4.2. *Influence of the space memory*

This second set of experiments shows how object instances, stored in the space memory and not visible anymore, can influence the agent's behavior. We exploit the fact that the algae are considered as empty spaces by the agent for hiding elements in the observable space. Thus, it is possible to observe the estimated position of objects stored in memory with the place signatures of places that characterize the position of this object. In this experiment, we propose to compare the behavior of the agent in a reference context, containing a prey, and in a second context where a second element is added. The comparison of behaviors in these two contexts allows to observe the influence of the second object on the decisional mechanism. The elements are masked by algae after being detected by the agent, to ensure that the agent's behavior is only influenced by the object context given by the space memory.

To better visualize the estimated position of object instances, we propose to display the estimated certitude of presence $\hat{c}_{p,\omega}$ of an object instance ω in a position p of the observable space by adding, weight by weight, place signatures of places that characterize the position of ω. The result values of each position are then normalized according to the highest value (in absolute value):

$$\hat{c}_{p,\omega} = \frac{\sum_{l \in \omega} w_{l,\omega}}{\max_{p}(|\hat{c}_{p,\omega}|)} \quad (14)$$

These certitude values are organized topographically according to the related position in space. A white pixel indicates that the instance may be present at the corresponding position with a high certitude, while a black pixel indicates that the instance is absent from the corresponding position with a high certitude.

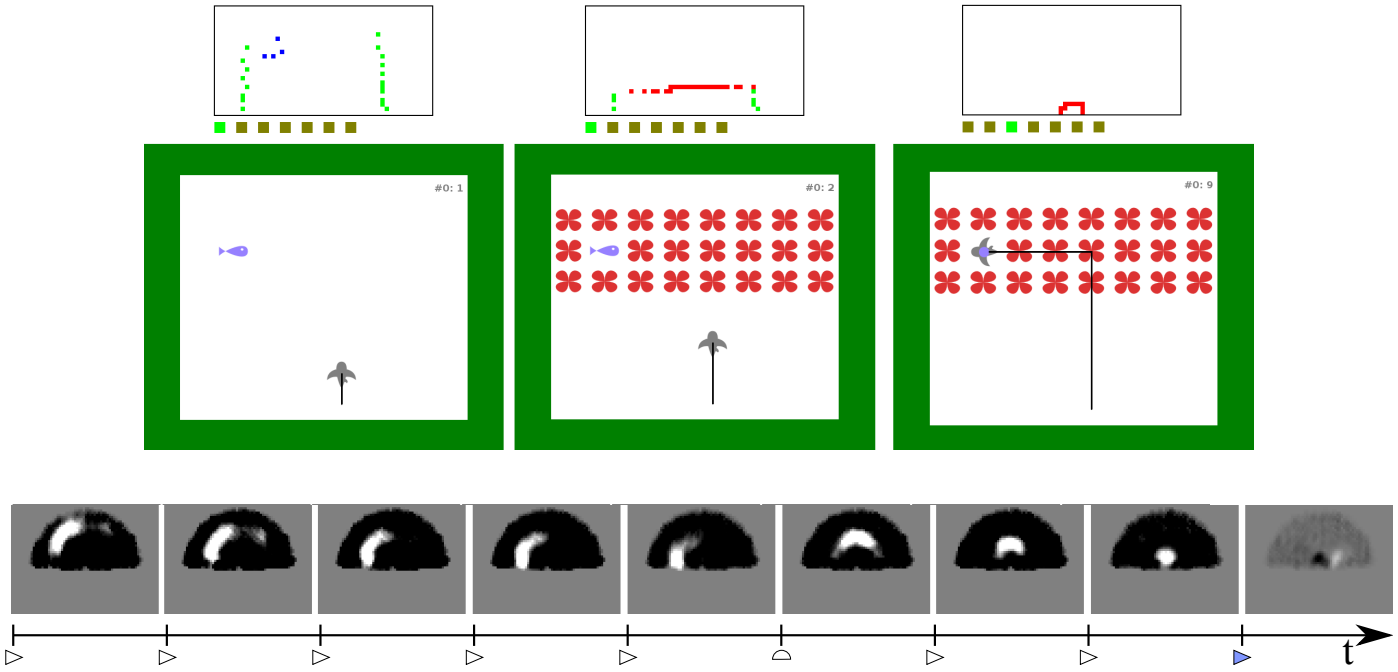


Figure 27. We let the agent discover the prey (left), then we hide the environment with algae (middle). The agent moves through algae with an efficient path (only one rotation) and reaches the prey (right). Bottom: the estimated position of the prey at each enaction cycle. The estimated position is obtained by adding place signatures of places that characterize the position of the object. The white blobs show the positions in which the object instance is likely to be. We can observe, on the last enaction cycle, that the agent *believes* that another prey is present on its right side (while the position of the eaten prey is considered to be empty). This means that during signature learning, the agent experiences several times the situation where two preys are adjacent.

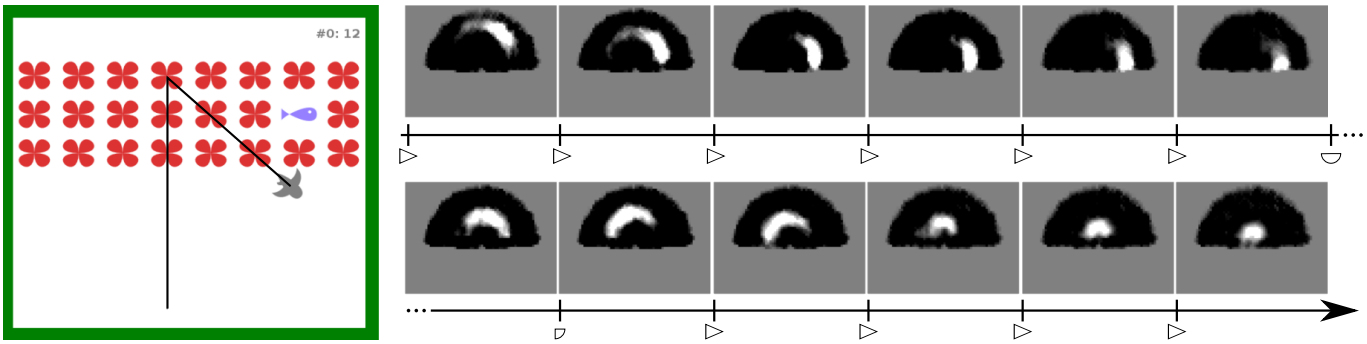


Figure 28. Agent in a symmetrical context: when the agent tries to reach the prey, it makes an error in estimation of the prey position. The agent then misses the prey, with an error smaller than a agent's step length. Left: the estimated position of the object instance affording *eat* according to the space memory. The agent is thus more fluent on its left side, due to asymmetries in its signatures of place and presence.

A gray color indicates that the certitude at this position cannot be defined, because the position is out of the observable space experienced by the agent or because the object instance vanished from the space memory. Thus, the lighter the position, the higher the certitude of presence in this position. Obviously, the lightest positions are at the intersection of the largest amount of places.

We begin with the reference context displayed in Figure 27: we place a prey on the left part of the agent's visual field. We then let the agent enact an interaction (here, *move forward*) to allow it to discover its environment. We then mask the environment with algae. The agent generates an efficient behavior consisting in moving forward until the prey is aligned, then turning 90° left and finally moving forward to reach the prey. We observe that the space memory is precise enough to estimate the moment when the agent has to turn. The estimation of prey position shows that the place update mechanism is able to track the invisible prey and give a correct estimation of this instance.

In a symmetrical situation, we observe that the agent misses the prey (Figure 28): the agent enacts an additional *move forward* interaction before turning. The agent then turns right and tries to align toward the prey. However, due to the lack of precision in the space memory, the agent cannot reach the prey, although the error was smaller than a step length. This observation shows that there are asymmetries in the signatures of place and presence and in the learning process. These asymmetries are mainly due to the order of interactions in the interactional set I that makes a bias in the signature learning mechanism. The

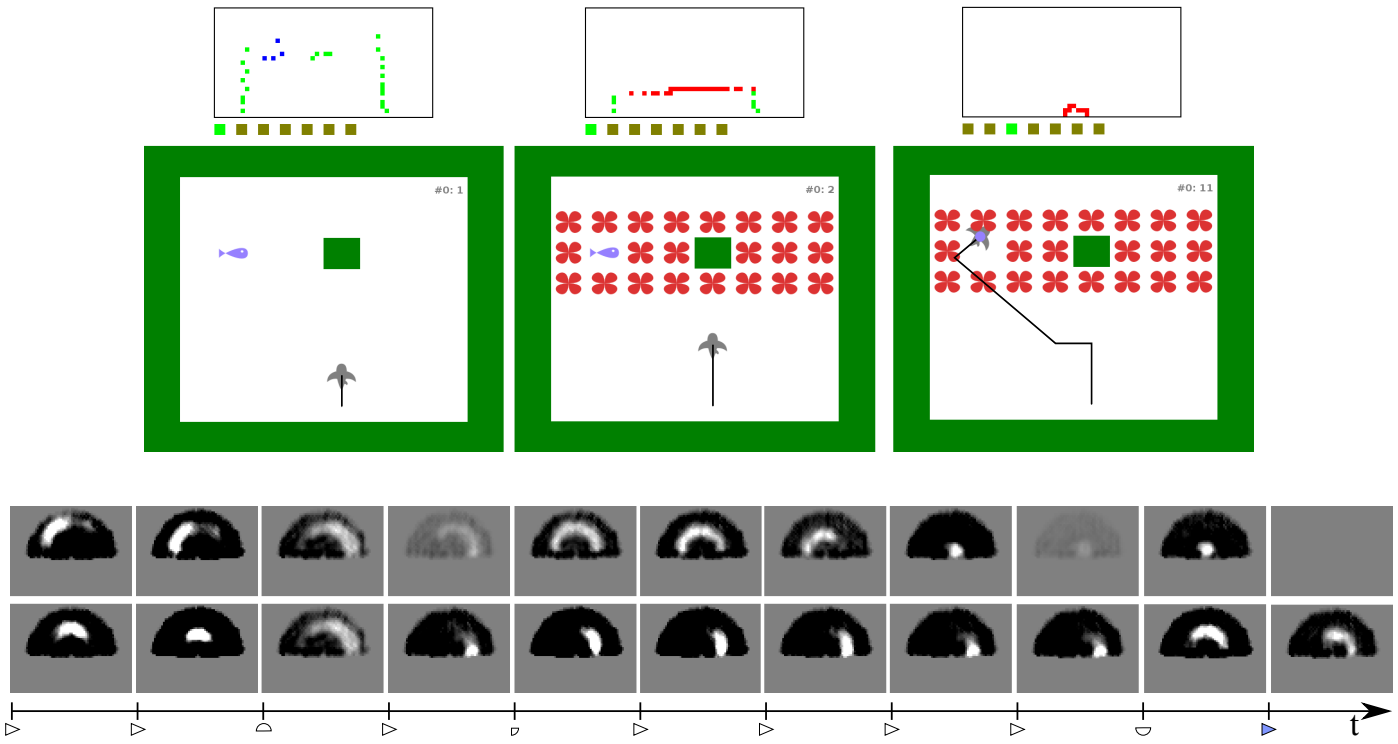


Figure 29. We add a wall on the path previously observed. The agent begins in the same conditions as before. Bottom: estimated position of the prey and the wall in space memory. The top timeline shows the estimated position of the prey. The bottom timeline shows the estimation of the position of the wall block. We can observe that the agent uses another path to reach the prey. This experiment shows that the agent takes every object stored in its space memory to adapt its behavior. We can also note that the prey disappears from the space memory.

agent is thus more *familiar* with its left side: its experience of its environment affects the reliability of its environment internal model and influences its behavior.

For the second part of the experiment, we add a wall block on the path used by the agent in the first part of the experiment (Figure 29). The agent starts at the same position and with the same signatures as before. We let the agent enact an interaction to discover its environment, and mask the environment with algae. We then observe that the agent is influenced by the wall present in the space memory: the agent uses another path that avoids the wall. It begins by turning left, moves forward to pass the wall block, turns to align toward the prey and finally moves toward the prey.

The agent turns after only two enaction cycles, which shows that the behavior is not a simple modification of the previous behavior for bypassing a wall on the initial path, but is a different behavior that includes the wall presence right from the start. This variation in behavior indicates that the agent is influenced by every element of its memory, as it takes both the prey and the wall into consideration.

6. Conclusion

We proposed a mechanism that allows an artificial agent to discover and integrate properties of its environment and that exploits this emerging knowledge to generate behaviors that satisfy a form of intrinsic motivation called *interactional motivation*, with the least possible predefined ontology about its environment and its sensorimotor possibilities. The agent defines a low-level semantics for its environment based on the valence of its interactions.

An agent equipped with this mechanism can construct a spatial memory that integrates spatial properties of its environment and generates a context that characterizes elements of the environment escaping the agent's sensory system. The precision of this spatial memory, which depends on the interactional possibilities of the agent, is low but sufficient to help the agent characterize its situation and generate behaviors that satisfy its interactional motivation.

The experiments demonstrate that the mechanism satisfies the five principles defined in Section 1.3:

- The agent is intrinsically motivated as its decisions are based on the principle of interactional motivation. This principle implies that no information about the environment is needed: the agent does not use the notion of environmental states or the notion of extrinsic reward.

- The agent defines its own internal object models through signatures of interactions. Objects defined by the agent are characterized by properties of elements of the environment that afford interactions as experienced by the agent. The agent can thus directly exploit these models as they consist of interactional properties. The objects defined by an agent can differ from objects that an external observer with different possibilities of interaction could define. Thus, where we, as external observers, defined empty spaces and algae, the agent defined a unique object that can be crossed. The agent also considers

long border walls and wall blocks as a unique object that affords the interaction *bump*.

- The agent can integrate surrounding space and its spatial properties without using the notion of geometrical space: object instances are recognized by *backmoving* signatures of interactions according to a sequence of interactions. Signatures of place and signatures of presence characterize spatial properties of the environment in terms of interactions. Thus, the agent constructs its own notion of space based on interactions.

- Signatures of place and signatures of presence are used to update the estimated position of object instances stored in the space memory. The space memory can then track object instances even when the agent can no longer perceive them, which constitutes a form of object permanence.

- The space memory generates a context of object instances that can be used by the agent to select intended interactions in order to maximize its interactional motivation principle in the short and medium term, by considering interactions that can be enacted in a near future, characterized by object instances. The decisional mechanisms presented in this paper are rudimentary but generate behaviors adapted to the agent's motivational principles.

The different components of the global mechanism were tested separately. Testing the interaction signature mechanism (M1) and the interaction signature learning mechanism first is possible and pertinent as the interaction signature learning process constitutes the first stage of the agent's learning process. Testing the object recognition mechanism alone is also possible as this mechanism does not rely on a learning process. We tested the place and presence signature learning processes with a partially hard-coded and simplified version of the interaction signature learning and object recognition mechanism. These hard-coded mechanisms implement properties observed with their agnostic equivalent mechanism, which allows to investigate the construction of a space memory in controlled conditions. However, we cannot observe any side-effects that can emerge from simultaneous learning of interaction, place and presence signatures.

We tested our mechanisms in a simplified environment, on agents equipped with limited sensorimotor possibilities. We simplified our experimental system for practical and technical reasons: first, we needed to reduce the complexity of our mechanisms to observe more precisely properties emerging from the learning process. As an example, with a 3D environment, it would be difficult to observe and interpret signatures of interactions, which would be displayed as 3D structures. We limited the number of colors to 3 to display signatures with RGB images rather than multiple columns (as we do in Figure 5, before overlapping columns). However, the environment is still more complex than the interactional possibilities of the agent: there are objects with the same interactional properties (such as algae and empty spaces) and elements of different sizes (walls), which allow us to observe how an agent equipped with a RI decisional system can interpret an environment that exceeds its sensorimotor possibilities.

6.1. Perspectives

In future work, we intend to extend this decisional system to more complex agents, to enable them to interact with more complex environments using improved possibilities of interaction:

- agents in dynamic environments, by simultaneously using spatial and sequential (Georgeon & Ritter, 2011) interactional mechanisms. Further works (Gay, Hassas, 2015) have shown that coupling the space memory with sequential mechanisms allows integration of dynamic properties of the environment, and thus, prediction of movements and tracking of mobile elements.

- simultaneous enaction of intended interactions: living beings move by using multiple muscles simultaneously. Modifying the Radical Interactionism model to integrate this property will help to generate more complex behaviors.

- inter-agent interactions: the current version of our agent cannot interact with another agent. Defining interactions that allow simple communication between agents will help form groups of agents and let low level social behavior emerge.

Our implementations used predefined and constant interaction valences to investigate our mechanisms in consistent conditions. We intend to study the possibilities of using valences that can depend on agent's internal states, such as hunger or tiredness: as an example, the interaction *eat* can have a high valence when the agent is starving and a low or negative valence when the agent is satisfied. As the object semantics defined by the agent is based on valences of interaction, rather than an object property that needs to be discovered, elements became immediately attractive or repulsive depending on the agent's internal states.

Similarly, the influence coefficient of the space memory and the object influence coefficient can depend on internal states: an agent in a critical state may generate short term behaviors based on close objects and immediately enactable interactions, while a satisfied agent can afford to consider distant possibilities of interactions and long-term behaviors.

The current object detection mechanism eliminates ambiguous objects (see Section 5.2). A curiosity mechanism can be added to make ambiguous objects attractive, thus leading the agent to interact with these objects and categorize them with other interactions.

References

Åström K. J. (1965). Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, vol. 10, no. 1, pp. 174-205.

Baleia J., Santana P., & Barata J. (2014). Self-Supervised Learning of Depth-Based Navigation Affordances from Haptic Cues. In *Proceeding of IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 146-151.

Blank D., Kumar D., Meeden L., & Marshall J. B. (2005). Bringing up robot: Fundamental mechanisms for creating a self-motivated, self-organizing architecture. *Cybernetics and Systems*, vol. 36, no. 2, pp. 125-150.

Chemero A. (2003). An outline of a theory of affordances. *Ecological Psychology*, vol. 15, no. 2, pp. 181-195.

Chinellato E., Antonelli M., Grzyb B. J., & del Pobil A. P. (2010). Implicit mapping of the peripersonal space by gazing and reaching. *IEEE Transactions on Autonomous Mental Development*.

Cos-Aguilera I., Cañamero L., & Hayes G. (2004). Using a SOFM to learn Object Affordances. In *Proceedings of the 5th Workshop of Physical Agents (WAF'04)*, pp. 139-150.

Cotterill R. M. (2001). Cooperation of the basal ganglia, cerebellum, sensory cerebrum and hippocampus: possible implications for cognition, consciousness, intelligence and creativity. *Progress in Neurobiology*, vol. 64, no. 1, pp. 1-33.

Crook P. A. & Hayes G. (2003). Learning in a state of confusion: Perceptual aliasing in grid world navigation. In *Proceedings of Towards Intelligent Mobile Robots, 4th British Conference on (Mobile) Robotics*, vol. 4.

Detry R., Baseski E., Popovic M., Touati Y., Kruger N., Kroemer O., Peters J., & Piater J. H. (2009). Learning object-specific grasp affordance densities. *Proceeding of IEEE 8th International Conference on Development and Learning*, pp. 1-7.

Fuke S., Ogino M., & Asada M. (2007). Body Image Constructed from Motor and Tactile Images with Visual Information. *International Journal of Humanoid Robotics*, vol. 4, no. 2, pp. 347-364.

Fuke S., Ogino M., & Asada M. (2009). Acquisition of the head-centered peripersonal spatial representation found in VIP neuron. *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 2, pp. 131-140.

Gay S. L. & Georgeon O. L. (2013). Interaction-Based Space Representation for EnvironmentAgnostic Agents. In *proceedings of ALA2013, workshop at AAMAS*, pp. 38-45.

Gay S. L., Georgeon O. L., & Wolf C. (2014). Autonomous object modeling based on affordances for spatial organization of behavior. *IEEE Fourth Joint International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EPIROB)*.

Gay S. L., Hassas S. (2015). Autonomous object modeling based on affordances in a dynamic environment. In *Annual International Conference on Biologically Inspired Cognitive Architecture (BICA)*.

Georgeon O. L. & Sakellariou I. (2012). Designing Environment-Agnostic Agents. In *proceedings of ALA2012, Adaptive Learning Agents workshop, at AAMAS 2012, 11th International Conference on Autonomous Agents and Multiagent Systems*, pp. 25-32.

Georgeon O. L., Marshall J. B., & Gay S. L. (2012). Interactional Motivation in Artificial Systems: Between Extrinsic and Intrinsic Motivation. In *Proceedings of the 2nd Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics (EPIROB)*, pp. 1-2.

Georgeon O. L., Marshall J. B., & Manzotti R. (2013). ECA: An enactivist cognitive architecture based on sensorimotor modeling. *Biologically Inspired Cognitive Architectures*, vol. 6, pp. 46-57.

Georgeon O. L. & Aha D. W. (2013). The Radical Interactionism Conceptual Commitment. *Journal of Artificial General Intelligence*, vol. 4, no. 2, pp. 31-36.

Georgeon O. L. & Ritter F. E. (2011). An intrinsically-motivated schema mechanism to model and simulate emergent cognition. *Cognitive Systems Research*, vol. 15-16, pp. 73-92.

Gibson J. J. (1977). The theory of affordances. In *Perceiving, Acting, and Knowing: Toward an Ecological Psychology*, Eds. Robert Shaw and John Bransford, ISBN-13: 9780470990148.

Graziano M. S. A., Taylor C. S. R., & Moore T. (2002). Complex movements evoked by microstimulation of precentral cortex. *Neuron*, vol. 34, no. 5, pp. 841-851.

Griffith S., Sinapov J., Sukhoy V., & Stoytchev A. (2012). A Behavior-Grounded Approach to Forming Object Categories: Separating Containers From Non-containers. *IEEE T. Autonomous Mental Development*, vol. 4, no. 1, pp. 54-69.

Griffith S., Sukhoy V., Wegter T., & Stoytchev A. (2012). Object Categorization in the Sink: Learning Behavior-Grounded Object Categories with Water. In *Proceedings of the 2012 ICRA Workshop on Semantic Perception, Mapping, and Exploration (SPME)*.

Harnad S. (1990). The symbol grounding problem. *Physica* D(42), pp. 335-346.

Hermans T., Rehg J. M., & Bobick A. F. (2011). Affordance Prediction via Learned Object Attributes. *International Conference on Robotics and Automation (ICRA): Workshop on Semantic Perception, Mapping, and Exploration*.

Hume D. (1739). Treatise of human nature. Oxford: David Fate Norton & Mary J. Norton, 1739, ed. 2000.

Ivaldi S., Nguyen S.-M., Lyubova N., Droniou A., Padois V., Filliat D., Oudeyer P.-Y., & Sigaud O. (2014). Object learning through active exploration.*IEEE Transactions on Autonomous Mental Development*, vol. 6, no. 1, pp. 56-72.

Kawamura K., Koku A.B., Wilkes D.M., Peters II R.A., & Sekmen A. (2002). Toward egocentric navigation. *International Journal of Robotics and Automation*, vol. 17, no. 4, pp. 135-145.

Lagoudakis M. G. & Maida A. S. (1999). Robot navigation with a polar neural map. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference*, pp. 965-.

Lungarella M., Metta G., Pfeifer R., & Sandini G. (2003). Developmental robotics: a survey. *Connection Science*, vol. 15, no. 4, pp. 151-190.

Maye A. & Engel A. K. (2011). A discrete computational model of sensorimotor contingencies for object perception and control of behavior. *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3810-3815.

Montesano L. & Lopes M. (2009). Learning grasping affordances from local visual descriptors. *IEEE 8TH International Conference on Development and Learning*, pp. 1-6.

Murata A., Fadiga L., Fogassi L., Gallese V., Raos V., & Rizzolatti G. (1997). Object Representation in the Ventral Premotor Cortex (Area F5) of the Monkey. *Journal of Neurophysiology*, vol. 78, no. 4, pp. 2226-2230.

Nguyen S.-M., Ivaldi S., Lyubova N., Droniou A., Gérardeaux-Viret D., Filliat D., Padois V., Sigaud O., & Oudeyer P.-Y. (2013). Learning to recognize objects through curiosity-driven manipulation with the iCub humanoid robot. *IEEE Third Joint International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pp. 1-8.

Northmore D. P. M. (2011). The Optic Tectum. In: *Encyclopedia of Fish Physiology: from Genome to Environment*, Academic Press, ISBN-13: 9780123745453.

O'Keefe J. & Dostrovsky J. (1971). The hippocampus as a spatial map: preliminary evidence from unit activity in the freely moving rat. *Brain Research*, vol. 34, no. 1, pp. 171-175.

O'Regan J. K. (2011). Why Red Doesnt Sound Like a Bell: Understanding the feel of consciousness. Oxford University Press USA, ISBN-13: 9780199775224.

Oudeyer P.-Y. , Kaplan F., & Hafner V. V. (2007). Intrinsic Motivation Systems for Autonomous Mental Development. *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 2, pp. 265-286.

Pfeifer R. & Scheier C. (1994). From Perception to Action: The Right Direction?. In *Proceedings of From Perception to Action Conference*, pp. 1-11.

Piaget J. (1954). The construction of reality in the child. Basic Books, New York, ISBN-13: 978-0465014071.

Pierce D. & Kuipers B. (1997). Map learning with uninterpreted sensors and effectors. *Artificial Intelligence*, vol. 92, no. 1-2, pp. 169-227.

Poincaré H. (1902). La Science et l'Hypothèse, ed. Flammarion.

Previc F. H. (1998). The neuropsychology of 3-D space. *Psychological Bulletin*, vol. 124, no. 2, pp. 123-164.

Stoffregen T. A. (2003). Affordances as properties of the animal environment system. *Ecological Psychology*, vol. 15, no. 2, pp. 115-134.

Uğur E., Doğar M. R., Çakmak M., & Şahin E. (2007). The learning and use of traversability affordance using range images on a mobile robot. In *Proceeding of IEEE International Conference on Robotics and Automation*, pp. 1721-1726.

Weng J., McClelland J., Pentland A., Sporns O., Stockman I., Sur M., & Thelen E. (2001). Artificial intelligence - Autonomous mental development by robots and animals. *Science*, vol. 291, no. 5504. pp. 599-600.