Interaction-Based Space Representation for Environment-Agnostic Agents

Simon Gay
Université de Lyon
CNRS, LIRIS, UMR5205
F-69622, France
+33 6 16 12 00 36
simon.gay@liris.cnrs.fr

Olivier Georgeon
Université de Lyon
CNRS, LIRIS, UMR5205
F-69622, France
+33 4 72 04 63 30
olivier.georgeon@liris.cnrs.fr

ABSTRACT
We propose a learning mechanism that allows an artificial agent to construct and exploit a representation of its surrounding space with minimal preconceptions about its environment. This representation is based on a data structure that encodes possibilities of behaviors afforded by the current context. The behaviors are modeled in the form of sequences of interactions. Over time, the agent learns to associate sequences of interactions with the presence of certain elements of the environment in certain locations in the agent's surrounding space. The agent uses this emergent relation between objects and possibilities of interactions to construct and maintain a representation of the surrounding space based on sequences of interactions. Experiments show that efficiently learning object and interaction associations requires implementing a form of curiosity as an additional motivational principle of the agent. These mechanisms open the way to implementing agents that learn to generate and exploit an awareness of their surrounding space with a minimal preconception of their environment.

Categories and Subject Descriptors
I.2.6 [**Artificial Intelligence**]: Learning. I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Intelligent agents*.

General Terms
Algorithms, Experimentation.

Keywords
Autonomous learning; Spatial awareness; Body schema; Peripersonal space; Affordances.

1. INTRODUCTION
In this paper, we address the problem of the construction of a short-range representation of the surrounding space by an artificial agent that initially ignores the fact that its environment has a spatial structure. In previous studies, we defined a design principle for artificial agents in which perception and action are kept embedded within data structures called *interaction*, rather than being separated, as it is the case in traditional modeling approaches. This design principle is intended to account for cognitive theories that suggest that perception and action are inseparable (i.e. O'Regan [14], Piaget [16]). Specifically, interactions are used to model Piaget's notion of *sensorimotor scheme*. We formalized the learning problem associated with this design principle as a special case of Partially Observable Decision Process (POMDP) [1] called Interactional Motivation Decision Process (IMDP) [7]. In an IMDP, the agent's learning is driven by predefined values associated with interactions rather than by a reward defined as a function of the state of the system. We called this new motivational principle *interaction motivation*. An interactionally motivated agent seeks to enact interactions with predefined positive values and to avoid interactions with predefined negative values. Because an IMDP agent can only obtain information about its environment through the active enaction of interactions, it implements a form of active perception.

Our implementations[1] have shown that an agent equipped with an IMDP algorithm was able to autonomously capture and exploit hierarchical sequential regularities afforded by the environment. But these works have also shown that these agents cannot learn spatial regularities. This means they are unable to notice that two different sequences can lead to the same spatial situation. They also do not recognize the persistence of objects, stopping to pursue them when they temporarily disappear from their sensory system [5]. The problem of the spatial awareness relies on the construction of a body schema, which consists of learning a model of the agent's own body and of the surrounding space that the agent can reach through interactions. Our approach relates to existing studies on body schema learning and exploitation [10,12], but differs by the agent's representation of objects and their locations in space that rests only on interactions between the agent and its environment.

More broadly, these studies investigate the hypothesis that the spatial structure does not need to be presupposed by a cognitive system. This hypothesis was formulated by theoreticians such as Poincaré [17]. Our mechanism is inspired by two observations made on monkeys. The first observation shows that certain neurons of the premotor cortex react to the global shape of presented objects, both if the monkey holds them or just looks at them [13]. This suggests that objects are recognized according to the grip movement needed to hold them. The second observation shows that neurons of the precentral cortex are associated to complex postures involving arms, torso, head and face [9]. These neurons are organized according to the final position of the hand in space. Such a mapping suggests that the prefrontal cortex encodes the surrounding space based on movements. This encoding, however, only covers the space that can be reached through direct movements, called peripersonal space. Considering the fact that a position is equivalent to the integration of a movement, we propose to extend this representation to the nearby surrounding space that a mobile agent can reach by enacting short movements.

Section 2 summarizes the IMDP formalism. Section 3 presents the new space representation mechanism. Section 4 describes the

[1] http://e-ernest.blogspot.fr/2012/03/small-loop-challenge.html

utilization of the mechanism by an artificial agent. The implementation on an artificial agent and the experiments we provided to test our mechanism are describe in section 5. Finally, the paper discusses our results and draws up recommendations for future works.

2. THE IMDP FORMALISM

Formally, an IMDP is defined as a tuple (Σ, Φ, q, ν, r) where Σ is the finite set of states σ of the environment, Φ the set of *interactions* I [7] offered by the coupling between the agent and the environment (an interaction represents an atomic sensorimotor scheme), q the distribution of probability of transition from a state σ^t to a state σ^{t+1} after attempting to enact an interaction I at step t, ν the distribution of probability to result in an enacted interaction $I_{enacted}$ after an attempt to enact an intended interaction $I_{intended}$, and r the satisfaction value function. This function differs from the reward function of standard reinforcement learning problems in that it is not a function of an environmental state σ, but of the enacted interaction. Unlike standard reinforcement learning approaches, the goal of an IMDP is not to maximize the reward value, but to learn a policy π that generates intelligent behaviors based on the intrinsic motivation provided by enacting interactions. Figure 1 illustrates this formalism.

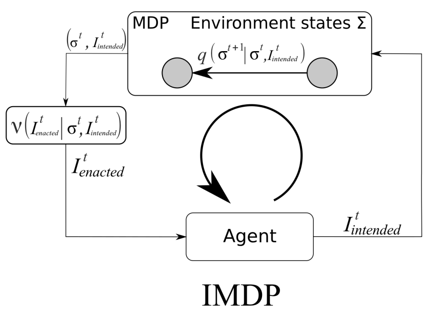


Figure 1. Diagram of an Interactional Motivation Decision Process (IMDP, adapted from [7]).

There are three major differences between an IMDP and a POMDP: (a) the cycle does not start from the environment but from the agent, making the agent's input a consequence of the agent's output rather than a premise. (b) The agent's input and output belong to the same set Φ rather than two different sets (observations and actions). (c) There is no reward defined as a function of the environment, making the agent self-motivated.

The philosophy of an IMDP is that the agent tries to enact an intended interaction, and is informed by the environment about the interaction that was enacted. If the intended interaction was effectively enacted, the enaction of this interaction is a *success,* and a *failure* otherwise. We call the *Alternative Group* A_k the set of interactions that can be enacted after an attempt to enact the interaction I_k.

3. THE SPATIO-SEQUENTIAL SYSTEM

To overcome the limitations of sequential systems, we propose to define an additional mechanism that allows an agent to organize its perception and sequential memory. Moreover, this mechanism must only be based on interactions to satisfy the sensorimotor hypothesis formalized in an IMDP.

We developed a mechanism, called Spatio-Sequential System (SSS), that helps an agent to characterize its current situation by constructing a spatial representation of its surrounding space. This representation is based on the fact that the result of the enaction of an interaction depends on a limited set of elements of the surrounding environment. These elements, called *objects*, and their locations can thus be characterized by the set of interactions that can be enacted in presence of these objects. Overall, by representing objects by the possibilities of interaction that they afford, we investigate theories of cognition that suggest that knowledge of the world arises from interaction (e.g. Gibson [8], and Hume [11]).

The agent gradually learns sequences of interactions and learns to apply them in appropriate contexts. The agent learns to predict the result of enacting a sequence of interaction and simultaneously represents the context in the form of the list of interactions predicted as success or failure. The SSS then keeps information by updating this list after each enacted interaction. This update process allows the memory to store sequences of interactions that could not be predicted, for example because the corresponding object is outside of the sensory system.

We call *decision process* the mechanism that selects the next sequence of interaction to enact. This mechanism uses the information stored by the SSS to determine the set of interactions that can be enacted in the current perceived environmental context.

This section formalizes the concepts used to implement the SSS: composite interactions, condition patterns, spatio-sequential memory and epistemic enactions. We provide details of our implementation when it can help clarify the description of these principles.

3.1 Composite Interactions

We define a *composite interaction* (C-interaction) S as a sequence of interactions of size k ∈ [1; n] with n the maximum length of C-interactions. The satisfaction value of a C-interaction is the sum of the satisfaction values of all of its k interactions. We call Ξ the set of all C-interactions. Enacting a C-interaction S of length k means sequentially enacting all of its k interactions. Note again that the intended C-interation $S_{intended}$ may differ from the enacted C-interaction $S_{enacted}$.We call decision cycle d the set of k steps during which $S_{enacted}$ is enacted. We call final interaction I_f the last interaction I_k, and path S_p the possibly empty subsequence $\langle I_1,..., I_{k-1}\rangle$ (1).

$$\begin{array}{l} S = \langle I_f \rangle \ \ if\ k=1 \\ S=\langle S_p, I_f\rangle \ /\ S_p \in \Xi \ \ else \end{array} \tag{1}$$

We expect the final interaction I_f of a C-interaction to identify the object needed to enact this interaction. We expect the path to identify the position of the object around the agent as it indicates the sequence of interactions needed to reach the object. A C-interaction thus characterizes an object in a certain localization in the surrounding space of the agent.

We define the function $\epsilon : \mathbb{N} \rightarrow \{-1;0;1\}$ that characterizes the result of the decision cycle d:

$$\epsilon(d)=\begin{cases} 1 \quad if \ \ S^d_{intended}=S^d_{enacted} \ \ (success) \\ -1 \quad if\,(S^d_{p_{intended}}=S^d_{p_{enacted}})\wedge(S^d_{intended}\neq S^d_{enacted})(\,fail) \\ 0 \quad else \ \ (abort). \end{cases} \tag{2}$$

The enaction of a C-interaction at a decision cycle d is a success if ε(d)=1, and a failure if ε(d)=-1. If ε(d)=0 (an interaction of the path has failed) then the C-interaction is said to be *aborted*. Indeed, objects are identified by the final interaction. A failure of the path only indicates that the object was unreachable.

3.2 Condition Patterns

Our learning mechanism is based on the fact that the result of the enaction of a C-interaction depends on the presence or absence of certain elements in the surrounding space. We implemented a mechanism that tries to learn, for each C-interaction S, a function $c_s(E)$ to predict the result of enacting S in a context of interactions E. We define the *interactional context* E as a vector $[e_1,\dots,e_m]$, with m=Card(Φ), $e_i \in [-1,1]$ for all $i \in [1,m]$, computed as follows:

$$\begin{gathered} I^t \ \ \textit{interaction enacted at step } t \\ I_i \ \ i^{th} \textit{ element of } \Phi, \ \ \forall i \in [\![1;m]\!]. \\ A_i \ \ \textit{Alternative group of } I_i \end{gathered}$$

$$e_i^t = \begin{cases} 1 & \textit{if} \ \ I_i = I^t \quad (\textit{confirmed interaction}) \\ -1 & \textit{if} \ (I_i \neq I^t) \wedge (I^t \in A_i) \ \ (\textit{failed interaction}) \\ 0 & \textit{else} \quad (\textit{no information}). \end{cases} \tag{3}$$

Thus, at each step t, e_i indicates the last result of the enaction of the i[th] interaction in Φ ; e_i = 1 means the success of the enaction, e_i = -1 means a failure. A value of 0 indicates that the interaction was not enacted. We expect c_s to indicate the prediction of enacting S in context E, with an absolute certitude of success if $c_S(E)=1$ and of failure if $c_S(E)=-1$. We propose to learn the functions c_S by defining a vector that gives the minimum conditions on the interactional context E to define the result of a C-interaction. This vector, called Condition Pattern C_S, must match the following properties:

$$\begin{aligned} \forall d \in \mathbb{N}, \ \ E^d \,.\, C_S &= 1 \ \Rightarrow \ \epsilon(d) = 1 \\ E^d \,.\, C_S &= -1 \ \Rightarrow \ \epsilon(d) = -1 \\ \epsilon(d) = 1 &\ \Rightarrow \ E^d \,.\, C_S \geq 0 \\ \epsilon(d) = -1 &\ \Rightarrow \ E^d \,.\, C_S \leq 0 \end{aligned} \tag{4}$$

The condition pattern C_S defines the minimum pattern over the interactional context E that allows the C-interaction S to be successfully enacted. Note that a condition pattern C_S can only be defined if the C-interaction S depends on a unique “object” perceived through interactions. The certitude function can then be defined as follows:

$$c_S(E) = E \,.\, C_S . \tag{5}$$

In our implementation, we generate the certitude functions using single layered neural networks, with the context vector E as input and the certitude value as output. Indeed, these structures can provide a robust pattern learning and recognition [2] over an input vector. The condition patterns are defined by the set of m weights of the corresponding network. The pattern C_S is reinforced each time the interaction S is completed (both as a success or a failure) using the delta rule (or Least Mean Square method):

$$\begin{gathered} \Delta = \epsilon(d-1) - c_S(E^{d-1}) \\ C_S[i]^d = C_S[i]^{d-1} + \alpha \times e_i^{d-1} \times \Delta \\ \forall i \in [\![1;m]\!] \ , \ \alpha \ \ \textit{learning rate}, \alpha \in [0;1]. \end{gathered} \tag{6}$$

These condition patterns define the “objects” in the point of view of the agent. The definition of objects is thus based on interactions and does not imply any internal modelization.

3.3 The Spatio-Sequential Memory

We propose an update mechanism to store recognized “objects” and track them while the agent is acting. We divide a C-interaction into a beginning interaction I_b and a final composite interaction S_f :

$$\begin{gathered} S = \langle \ I_f \ \rangle \ \textit{if} \ k = 1 \\ S = \langle I_b, S_f \rangle \ / \ S_f \in \Xi \ \ \textit{else}. \end{gathered} \tag{7}$$

The update mechanism is based on a property of the C-interactions: S and S_f are related to a same object, but they refer to different locations separated by a distance represented by the beginning interaction I_b. Thus, if a C-interaction S is predicted as a success, and then the agent enacts I_b, S can be replaced by the final C-interaction S_f. The Spatio-Sequential Memory (SSM) is then equipped with two lists, T, containing the interactions predicted as success, and F, containing the list of interaction predicted as failure. The update principle is applied after each interaction enacted by the agent. The global SSM algorithm is described in table 1. We note τ the threshold that defines the reliability of the memory with τ ∈ [0;1].

Table 1. Algorithm of the spatio-sequential memory

```
I is the last interaction enacted by the agent

  for each C-interaction S in T
      remove S from T
      if Ib = I then add Sf to T
  end for

  for each C-interaction S in F
      remove S from F
      if Ib = I then add Sf to F
  end for

  for each C-interaction S ∈ Ξ
      if cS(E) > τ then add S to T
      if cS(E) <-τ then add S to F
  end for
```

An updated C-interaction can provide, through its condition pattern, information about an element of the interactional context for which the corresponding interaction was not used in the last step (e.g. its value is 0). The fact that this C-interaction is in list T (or F) indicates that the interactional condition given by the prediction pattern is present (or absent). The agent can thus use these updated interactions to generate a completed interactional context E' = [e'₁,…, e'ₘ] as follows:

$$\begin{gathered} \forall i \ / \ S_i \in T \ \ , \ \ \forall j \ / \ S_j \in F \ \ , \\ E' = E + \sum_i C_{S_i} - \sum_j C_{S_j} . \\ e'_i \textit{ bounded by} [-1;1] \ , \ \ \forall i \in [1;m] \end{gathered} \tag{8}$$

If the interactions used to construct E' are reliable enough, this context can give the agent an improved perception of its environment. We can then add a second SSM, composed of the lists T' and F' and based on E'. This process can be repeated to complete the interactional context, but this is done at the expense of the reliability of the information.

We increase the reliability of the memory by implementing a mechanism that detects errors in condition patterns and eliminates the corresponding C-interactions from the memory. These errors can be detected by comparing the condition pattern of a C-interaction with the condition pattern of its path: enacting a C-interaction means enacting first its path, which means that a certain environmental context is favorable to the enaction of both the C-interaction and its path. Thus, if the condition patterns of a C-interaction and of its path correspond to different and incompatible environmental conditions, at least one of these patterns contains errors. We define a C-interaction S as *coherent* if it matches the following condition:

$$\begin{gathered} S=\langle I_{1,}\ldots,I_k\rangle \ \ , \ \ S_i=\langle I_{1,}\ldots,I_i\rangle \ \ , \ \ i\in[1;k] \\ coherent(S) \Leftrightarrow \exists E\in[-1;1]^m / \forall i\in[1;k], c_{S_i}(E)>0 \end{gathered} \tag{9}$$

We thus define a memory that can define elements of the environment as objects and localize them in a non-topographic representation. This memory can also track these objects while the agent is moving, which results in a form of object persistence learning.

3.4 Epistemic Enaction System

As introduced in Section 1, an IMDP agent actively perceives its environment through interactions rather than passively receiving perceptual data. This active perception principle implies that the agent may sometimes enact interactions for the purpose of acquiring information about its current situation rather than for the purpose of immediately satisfying its interactional motivation. We call *epistemic enaction* the enaction of a C-interaction intended for a perception purpose, as opposed to the enaction of a C-interaction intended for the satisfaction value it can provide. The term epistemic refers to Gatti's *Sensorial Epistemic Actions* [4], that is defined as an action "*in which the cognitive agent structures her own sensorial action in order to receive from the environment a feedback structured sensation that carries information*".

An epistemic enaction may be needed if the intended C-interaction cannot be predicted in the current environmental context E, i.e. the absolute value of $c_s(E')$ is lower than a certain threshold. We implemented a mechanism, called Epistemic Enaction System (EES), to propose such an epistemic enaction. For an intended C-interaction $S_{intended}$, the EES proposes to enact a C-interaction $S_{epistemic}$ that can provide an interactional context in which the prediction of success or failure of $S_{intended}$ can be defined with a high degree of certitude. The EES learns the information provided by each C-interaction. We integrate the provided information of a C-interaction S by measuring the average improved context E' observed when it is successfully enacted. For each C-interaction S, we thus define a vector $P_S = [p_{S\_1},...,p_{S\_m}]$, called the prediction pattern P_S , that characterizes the average environmental context observed when this C-interaction is enacted, and thus, the provided information.

In our implementation, we define a prediction pattern P_S as a weighted average of every context vector E' observed when the C-interaction S is successfully enacted, with high weights for recent contexts (10). We thus reduce the influence of old observations as they can be out of date due to condition patterns learning.

$$p_{S_i}^{d+1} = \begin{cases} \dfrac{(\beta \times p_{S_i}^d + e'^d_i)}{(\beta+1)} & if\ e_i{}'^d \neq 0 \\ p_{S_i}^{d+1} & else \end{cases} \tag{10}$$
$$\forall i\in[\![1;m]\!],\ \beta \ \ discount\ factor,\ \ \beta>0.$$

The mechanism then uses the prediction patterns to select the most suitable interaction $S_{epistemic}$ to provide the needed information to define the certitude of the intended C-interaction $S_{intended}$. The aim of an epistemic enaction is to provide an interactional context in which an intended C-interaction can be predicted (as a success or a failure) with a high certitude. As the prediction pattern gives the average interactional context observed when the epistemic interaction is enacted, we can define the suitability of an epistemic interaction as the absolute value of $c_{S_{intended}}(P_{S_{epistemic}})$. We called the selected C-interaction of the EES the *epistemic interaction*.

4. UTILIZATION OF THE SPATIO-SEQUENTIAL SYSTEM

The SSS is an extension of the sequential decision process described in section I. However, the SSS can be self-sufficient if the range of the memory can cover the space covered by the sensory system of the agent. We tested two different approaches in exploiting the SSS on such an agent. The first approach is based on the satisfaction values of interactions. The agent selects an interaction according to its satisfaction value, based on the hypothesis that the condition patterns, and thus the predictions of success, are correct. The condition patterns are only corrected when errors are observed. The second approach adds an additional intrinsic motivational mechanism that implements a form of curiosity that drives the agent to test and validate its hypothesis.

4.1 Greedy Agent

We qualify an agent that uses the first approach as *greedy* if its behavior is guided only by the satisfaction of its interactional motivation. This approach consists of listing every interaction considered as enactable according to the SSS. We define an interaction as *enactable* in a decision cycle d if the interaction is predicted as a success and if the path is recognized as *valid* (11) according to the SSS. The SSS thus act like a filter that simplifies the selection process by eliminating irrelevant elements.

$$\begin{gathered} S=\langle I_{1,}..,I_k\rangle \ \ , \ \ S_p=\langle I_{1,}..,I_{k-1}\rangle \\ S_i=\langle I_{1,}..,I_i\rangle \ \ , \ \ i\in[1;k] \\ Valid(S_p,d)=\forall i\in[1;k-1],\ \ S_i\in T'^d \end{gathered} \tag{11}$$

Note that, due to the initial lack of reliability, the threshold that characterizes a prediction of success must be initially negative to allow an untested C-interaction S to be selected. This value can then be increased with the reliability of the interactions.

The agent's decision process then uses the list of enactable interactions to select the next C-interaction. This policy consists of selecting the C-interaction with the highest satisfaction value. Note that an interaction that does not imply movement can be repeated indefinitely. We thus implemented a mechanism that compares condition and prediction patterns to detect and avoid such C-interactions. If the prediction of the selected C-interaction cannot be defined in the current interactional context, the EES selects an epistemic interaction that can provide the missing information. Once the epistemic interaction is enacted, and if the selected C-interaction is still enactable, the system enacts it. Otherwise, the selection process is repeated.

4.2 Curious Agent

The second approach is similar to the first approach, but adds an additional mechanism that drives the agent to test the validity of its condition patterns. This mechanism implements a form of curiosity because the agent selects interactions for the purpose of testing them in unknown situations.

This second approach thus consists in learning and testing C-interactions while they are considered as reliable. We define an interaction as *reliable* if its number of correct consecutive predictions of success, and of failure, are greater than a predefined threshold. Note that the higher the threshold value, the higher the certitude of having a correct prediction pattern. But on the other hand, the number of tests needed increases. This selection mechanism selects a non-reliable sequence with a valid path in order to maximize the learning rate, according to the actual information stored in the SSS and the number of correct consecutive predictions. The selection process does not take the satisfaction value of interaction into consideration. This mechanism is thus related to the problem of active learning [15].

The selection process is based on the number of correct consecutive number of successes and of failures. The C-interactions are tested alternatively in a situation of success and of failure prediction. This alternation prevents the learning mechanism from repeating consecutively a test on a C-interaction in the same environmental context. Note that in certain cases, the final interaction I_f is always possible after enacting the path S_p. For these C-interactions, the number of failures will remain zero. We then define an interaction as *always true* if the number of consecutive predictions of success is greater than a predefined threshold and if the C-interaction has never failed. An always true C-interaction is considered as reliable.

This mechanism returns at most one candidate C-interaction. If no C-interaction can be learned in the current environmental context, the mechanism selects a C-interaction according to the selection process of the greedy approach. Thus, while the number of reliable C-interactions increases, the curiosity approach is progressively replaced by the greedy approach. Note that the agent does not become definitively greedy, as the agent can continue to use the curiosity approach if new sequences of interactions are observed.

5. IMPLEMENTATION IN AN ARTIFICIAL AGENT

This mechanism was tested on a simple agent evolving in a 2-dimensions discrete and static environment. This environment was implemented from Cohen's Vacuum Environment [3]. Both the environment and the agent were implemented in Java. The agent has twelve possible interactions, listed in table 2.

Table 2. List of interactions. The satisfaction value of each interaction is given in parentheses.

I_1 =	(-1) touch an empty space on left side
I_2 =	(-2) touch a wall on left side
I_3 =	(-1) touch an empty space on front side
I_4 =	(-2) touch a wall on front side
I_5 =	(-1) touch an empty space on right side
I_6 =	(-2) touch a wall on right side
I_7 =	(5) move forward one grid cell
I_8 =	(-20) hit a wall
I_9 =	(-3) turn left (90°) in front of an empty space
I_{10}=	(-5) turn left (90°) in front of a wall
I_{11}=	(-3) turn right (90°) in front of an empty space
I_{12}=	(-5) turn right (90°) in front of a wall

We reduced the complexity of the problem by limiting the context vector E to the six *touch* interactions. This simplification is reasonable as these interactions are the only ones that can be considered as purely perceptive. The agent can thus detect two types of objects, Empty Spaces and Walls, in three distinct locations, Front, Left and Right. Note that the agent has no *a priori* preconceptions about the number of objects and locations.

We implemented the SSS as shown in Figure 2. This memory is composed of three SSMs representing three levels of reliability. The interactional context E is used for condition pattern learning, as the information provided is defined with an absolute certitude. The second SSM is used for prediction pattern learning, and the third SSM is used to define the enactability of C-interactions.

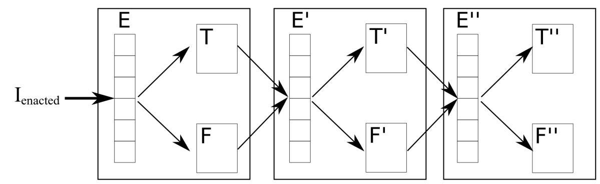


Figure 2. The agent's spatio-sequential system. The system is composed of three successive SSMs. Each SSM is composed of an interactional context vector of size 6 (respectively E, E' and E'') and the lists of C-interactions predicted as true (respectively T, T', T'') and false (F, F', F'') according to the corresponding interactional context. The interactional context of a SSM is completed by C-interactions stored in the lists of the previous SSM.

We tested our mechanisms in four different environments to measure the influence of the environmental structure on the learning process and the final behavior of the agent. These environments are variants of *the small loop environment* proposed in [6]. They are shaped in a loop to offer sequential and spatial regularities for the agent to discover and exploit. Environment 1 has five right turns and one left turn (clockwise) while Environment 2 has 6 right turns and 2 left turns (Figure 3). Environments 3 and 4 are mirror versions of the environments 1 and 2. The memory was tested with a range (maximum lenght of C-interactions) of 2 and 3.

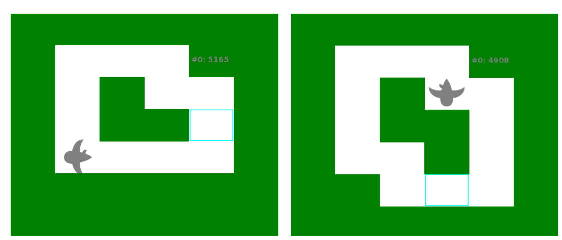


Figure 3. Left: Environment 1. Right: Environment 2. The blue outlined squares show the position of the additional wall blocks.

The experiments were conducted as follows: the agent is started in its environment. Once the behavior stops evolving, we add a wall block to close the loop of the environment and add a new environmental configuration (a dead end). While the agent is learning, the behavior and the condition patterns are analyzed.

We propose to observe the emergence of discriminated elements by analyzing the condition patterns of C-interactions. We can especially observe the location specialization and the object specialization of sequences. We propose to observe the object persistence by comparing the C-interaction stored in the SSMs and the improved interactional context E'' with the interactional context E.

The relation between C-interactions and space is displayed as follows: we represent the surrounding space of the agent with three squares corresponding to the left, front and right cells of the environment the agent can perceive through *touch* interactions. C-interactions are represented by points, for whose position is determined by the absolute values of its condition pattern. We display a link between interactions and their final interaction. This link represents the "movement" of objects around the agent while the path is enacting.

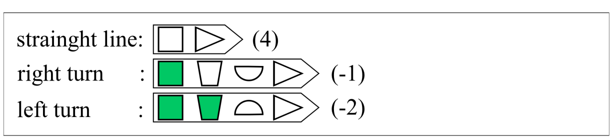
We represent the relation between interactions and objects with a second set of points. For each pair of points, we generate an attractive force with a value corresponding to the scalar product of condition patterns of corresponding C-interactions. Thus, points of C-interactions with similar pattern are attracted, and repulsed if the patterns are opposite. A strong attractive force is applied between points with a same final interaction, as they are related to a same object by definition. Points of C-interactions with a final interaction I_f related to an empty place (as described in table II) are represented with light gray crosses, points related to walls are represented with green discs. Note that the agent does not have any access to these graphs.

The object persistence is observed by analyzing the interactional contexts E, E' and E'', and the C-interactions stored in the memory lists. As described in section 3.3, the object persistence consists of information stored by the SSMs that cannot be found in the interactional context E. We represent the six elements of the interactional contexts and condition patterns with three squares, as interactions with a same movement are mutually incompatible. We thus only display interactions related to empty space. A white square means the corresponding interaction is a success (and thus a failure of the opposite interaction) or the corresponding value of a prediction pattern is positive; a black square means the interaction has failed or a negative value of the condition pattern.

5.1 Results of the Greedy Approach

We first tested the direct utilization approach with a memory range of 2. In the four environments, we observed, after fewer than 400 steps, a very efficient behavior summarized in table 3.

Table 3: behavior observed with the direct utilization approach with a memory range of 2



This behavior is based on an error in condition patterns: as the *turn* interactions are used only in turn configurations, the agent can associate both *turn* interactions with a single side (empty space on left side for turn left and wall on left side for turn right). This hypothesis is confirmed by the pattern repartition (Figure 4.a): the agent does not take its left side into consideration. These patterns are incorrect as they would fail in a straight line configuration, but as the agent only uses the corresponding interactions in turns, we can consider them as correct for their current utilization.

We then add the wall block. The agent adapts its behavior in 500 to 700 steps, depending on the environment. We can observe that the agent has learned that certain C-interactions are related to its left side (Figure 4.b). However, a large part of C-interactions still

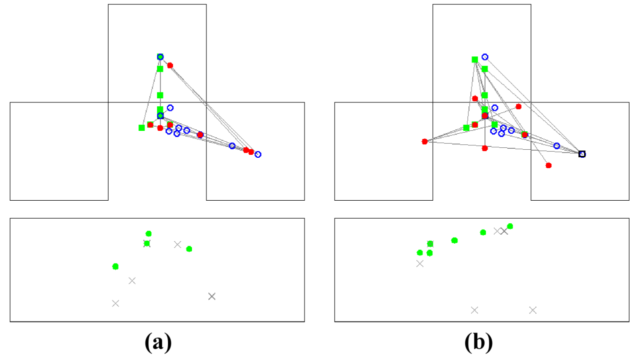
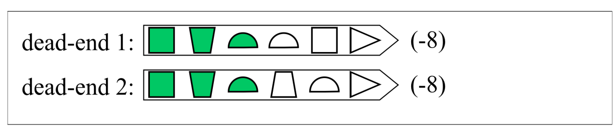


Figure 4. interaction specialization before (left) and after (right) adding a wall block in the environment (first environment, memory range of 2). Top: the location specialization of C-interactions: each C-interaction is represented by a point whose position is determined by absolute values of its condition pattern. The color and shape of the points represent the real location of the C-interaction: red disc for left, green square for front and blue circle for right, black square if the C-interaction depends on a location that is outside of the sensory system. Grey lines connect C-interactions with their final interaction. Bottom: the object specialization: points with a same final interaction are grouped together, points with similar condition patterns are attracted. C-interactions with a final interaction related to an empty space are light gray crosses, and green discs for a wall. These graphs show that the direct utilization approach does not allow the agent to discriminate environment location nor objects. The (a) graph shows that before the wall block is added, the agent does not take its left side into consideration.

contains errors. We did not observe changes in known configurations. We however observed two different behaviors in the dead-end configuration, depending on the environment (table 4). These two sequences use a touch interaction to probe the element that was originally behind the agent.

Table 4: behaviors observed in the dead-end configuration.



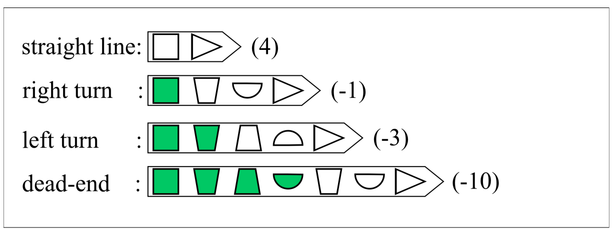
Despite this efficient behavior, the learned condition patterns are not reliable enough to discriminate objects and locations. The agent thus cannot use the C-interactions to characterize its surrounding environment. The test with a memory range of 3 showed another limit of this approach: because of the accumulation of errors, the final behavior remains inefficient in some environments: the agent continues to bump into walls or stay trapped in a part of the environment. These observations show that the greedy approach does not allows the agent to specialize the C-interactions with pertinent information. This suggests that the agent needs to validate the condition patterns before exploiting them.

5.2 Results of the curiosity approach

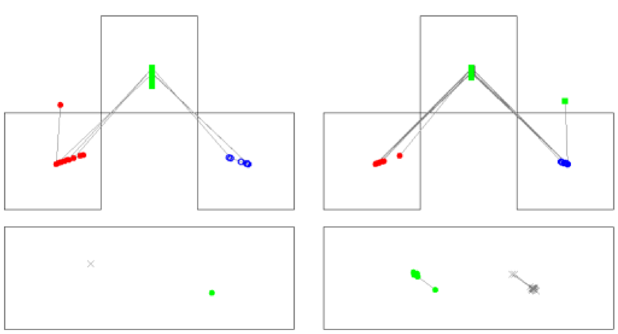
The learning approach was first tested with a memory range of 2. The agent reached a stable behavior after 3700 to 4200 steps, depending on the environment, and learned the dead-end configuration in fewer than 1000 steps after the additional wall block is added. The final behavior is summarized in Table 5. Compared to the direct utilization approach, we can observe an additional touch interaction in the left turn configuration. The reason is that the agent has successfully associated the turn left interaction with its left side. The agent thus uses an epistemic interaction to probe the element on its left side.

We then tested the mechanism with a memory range of 3. Of course, the learning process becomes slower as the number of possible interactions increases: 19000 to 22000 steps are needed to reach the final behavior, and the about-turn configuration needs 3000 to 12000 additional steps. The final behavior is basically the same as with a memory range of 2. However, we observed some variations in the first and second environments for the turn-about configuration. This variation is due to an error in a condition pattern that cannot be corrected once the agent reaches permanent behavior. However, this error has no negative effect on the behavior: as observed with the greedy approach, the condition patterns containing errors are correct for their current utilization. These variations suggest that the environment still has an influence on the learning process of the memory, and thus, on the behavior.

Table 5: final behavior of the agent with the learning approach



The location and object discrimination is visible on Figure 5: even if some errors remain, a large majority of C-interactions are specialized to the correct locations. With a range of 2, more than 90% of the reliable C-interactions are specialized in the correct location in every environment, and more than 70% with a range of 3. We do not observe errors in the object discrimination: the C-interactions are aggregated into two groups, corresponding to the two objects (empty space and wall) of the environment. The agent has thus defined the notions of "left", "front", "right", "empty space" and "wall" based on its interactions.



(a) **(b)**

Figure 5. Location and object specialization for a range of 2 (a) and 3 (b) in environment 2, after behavior stabilization (respectively steps 4500 and 28000). We only display points of reliable and coherent C-interactions. Top: respectively 92.2% and 79.5% of reliable C-interactions are specialized to their correct location (center of squares). Bottom: the interactions are perfectly grouped according to the object on which they depend. This graph shows that the agent has discriminated the two objects in the environment (walls and empty cells).

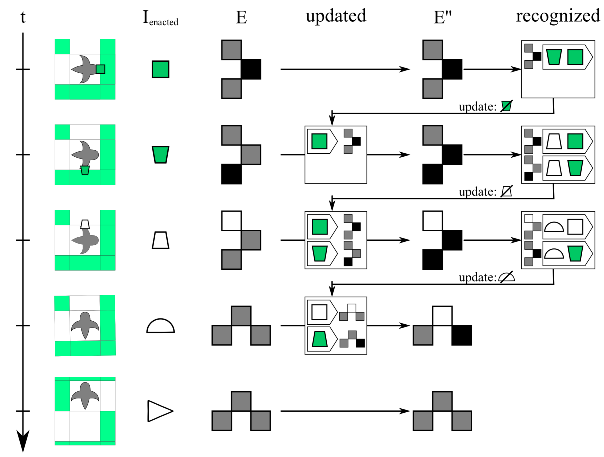


Figure 6. Object persistence in memory. Top to bottom: the five steps of the sequence used to turn left. From left to right, we display the agent in its environment, the enacted interaction, the interactional context E, updated elements of the memory T'' with their condition patterns, the completed interactional context E'', and recognized C-interactions. The three squares of interactional contexts and patterns represents the *touch left*, *front* and *right* interactions, white color represents a *touch empty space* interaction and black color a *touch wall* interaction. Only relevant C-interactions stored in the memory are listed. The improved interactional context E'' contains information that is not provided by the original context E. This information allows the agent to move forward without probing its front side on the last step.

Figure 6 shows an example of sequence of interactions where the improved context E'' contains information that is not found in the initial context E. This example shows how the memory keeps track of objects while the agent is touching its environment or rotating. We can observe that in the third step, the six elements of the completed interactional context E'' are non-zero. The agent can thus simultaneously "perceive" the enaction of interactions , , and the failure of interactions , , . This information allows the agent to characterize its current situation. In step 4, the memory has tracked the object (empty space) that was initially on the agent's left side. The object is now in front on the agent, which allows it to move forward without probing the environment.

The object persistence allowed by the SSS is not limited to the completed context E'': Figure 7 shows an example of a sequence of interactions where the SSS gives information about an object that is lost by the agent's sensory system. In this example, the SSS indicates that there is a wall behind the agent. This C-interaction cannot be used to complete the interactional context as its condition pattern is empty, but informs the agent about an object

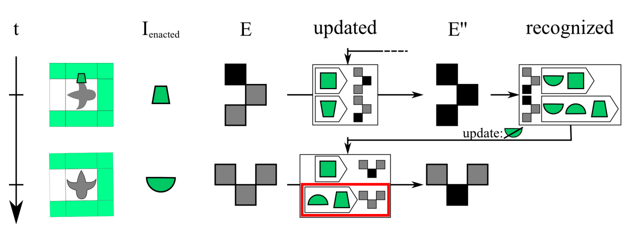


Figure 7. In the second step, the memory contains a C-interaction that is not related to an element of the sensory system of the agent, but to a wall behind the agent: . The memory thus allows the agent to be aware and to reach an element that is outside of its sensory system.

that is outside of the agent's sensory system. The memory can thus increase the agent's perception field. Note that the SSS does not only indicate the presence of the object, but also the sequence of interaction needed to reach it, in the form of a C-interaction.

These observations show that the SSS mechanism allows an agent to give a meaning to its interactions. Our agent is thus able to recognize, localize and track objects that it has defined according to its own viewpoint. The observed object persistence shows that the agent has construct an "awareness" of its surrounding space.

This second approach shows that a form of curiosity motivation must be implemented to allow an agent to associate pertinent information with its interactions. Once this information is reliable enough, the intrinsic motivation based on satisfaction values of interactions can provide efficient behavior and a correct utilization of the SSS abilities.

6. DISCUSSION AND CONCLUSION

We propose an approach to implement a non-topographic representation of the peripersonal space based on interactions. Our implementation shows that such a representation can simultaneously be learned and used by an artificial agent.

This memory segments, recognizes, localizes and tracks objects of the surrounding environment without any ontological preconception about these objects. We believe that this mechanism sheds some light on the question of how an agent can construct its own perception and knowledge of its environment. The structure of the memory based on interactions allows the agent to exploit information with a limited computational cost, because the agent directly uses the stored interactions to generate behavior. This suggests that this memory can generate a set of reflex behaviors related to space.

These results are however limited by the simplicity of our test system. These results are nevertheless encouraging as they show that an agent equipped with a rudimentary sensory apparatus can construct a representation of the perceived environment without presupposing the spatial nature of its environment. In future work, we intend to address the problem of implementing our mechanisms in more complex possibilities of interactions, especially agents evolving in continuous environments as opposed to a grid. We believe that a continuous set of interactions can improve the agent learning skills in such an environment by allowing parallel learning and pattern overlapping of similar interactions.

7. ACKNOWLEDGMENTS

This work received the support the Agence Nationale de la Recherche (ANR) contract ANR-10-PDOC-007-01.

8. REFERENCES

[1] Aström, K., 1965. "Optimal control of Markov processes with incomplete state information", Journal of Mathematical Analysis and Applications (10), pp. 174-205.

[2] Bishop C. M. 1995. Neural networks for pattern recognition, Clarendon Press, Oxford.

[3] Cohen M. A. 2005. “Teaching agent programming using custom environments and Jess”. AISB Quarterly, 120(Spring), 4.

[4] Gatti A. 2005. “On Distinguishing Sensorial and Eliciting Epistemic Actions and on the Relationship between Perceptive Structure of Body and Cognitive Processes”, in XXVII Annual Conference of the Cognitive Science Society (CogSci2005).

[5] Gay S., Georgeon O. L. and Kim J. W. 2012. “Implementing a spatial awareness in an environment-agnostic agent”, BRIMS2012 Proc., pp. 62-69.

[6] Georgeon O. L. and Marshall J. B. 2012. “The Small Loop Problem: A challenge for artificial emergent cognition”. In proceedings of BICA2012, Annual Conference on Biologically Inspired Cognitive Architectures. Palermo, Italy, pp. 137-144.

[7] Georgeon O. L., Marshall J. B. and Gay S. 2012. “Interactional motivation in artificial systems: between extrinsic and intrinsic motivation”, IEEE Conference on Development and Learning / EpiRob 2012, San Diego.

[8] Gibson J. J. 1977. The theory of affordances. In R. Shaw & J. Bransford (Eds.), Perceiving, acting, and knowing: Toward an ecological psychology (pp. 67-82). Hillsdale, NJ: Erlbaum.

[9] Graziano M., Taylor C. and Moore T. 2002. “Complex movements evoked by microstimulation of precentral cortex”, in Neuron, vol. 34, pp. 841-851.

[10] Hoffmann M., Marques H. G., Arieta A. H., Sumioka H., Lungarella M. and Pfeifer R. 2010. “Body schema in robotics: a review”, IEEE Transactions on Autonomous Mental Development, vol.2.

[11] Hume D. 1739, ed. 2000. Treatise of human nature, Oxford: David Fate Norton & Mary J. Norton.

[12] Lee M., Law J., Shaw P. and Sheldon M. 2012. “An infant inspired model of reaching for a humanoid robot”, 2012 IEEE Joint Conference on Development and learning - epigenetics and robotics.

[13] Murata A., Fadiga L., Fogassi L., Gallese V., Raos V. and Rizzolatti G. 1997. “Object representation in the ventral premotor cortex (area F5) of the Monkey”, J Neurophysiol 78:2226-2230.

[14] O'Regan J. K. 2011. Why Red Doesn't Sound Like a Bell: Understanding the feel of consciousness, Oxford.

[15] Oudeyer P.-Y., Kaplan F. and Hafner V. V. 2007. “Intrinsic Motivation Systems for Autonomous Mental Development”, IEEE Transactions on Evolutionary Computation, 11(2), pp. 265-286.

[16] Piaget J. 1937/1954. The construction of reality in the child. New York: Basic Books.

[17] Poincaré H. 1905. Une architecture du son - Cahiers n°1/2, Ed. L'Harmattan, 2005, p. 322.