Autonomous affordance construction without planning for environment-agnostic agents
Simon Gay, Alain Mille, Amélie Cordier

To cite this version:
Simon Gay, Alain Mille, Amélie Cordier. Autonomous affordance construction without planning for environment-agnostic agents. The Sixth Joint IEEE International Conference on Developmental Learning and Epigenetic Robotics (ICDL-Epirob), Sep 2016, Cergy-Pontoise, France. pp.111-116, 10.1109/DEVLRN.2016.7846800. hal-01371894

HAL Id: hal-01371894
https://hal.archives-ouvertes.fr/hal-01371894
Submitted on 26 Sep 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Autonomous affordance construction without planning for environment-agnostic agents

Simon L. Gay
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205
Villeurbanne, F-69622, France
Email: simon.gay@liris.cnrs.fr

Alain Mille
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205
Villeurbanne, F-69622, France
Email: alain.mille@liris.cnrs.fr

Amélie Cordier
Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205
Villeurbanne, F-69622, France
Email: amelie.cordier@liris.cnrs.fr

Abstract—We present an architecture for self-motivated agents to organize their behaviors according to possibilities of interactions proposed by the environment, and to modify the environment to construct new possibilities of interactions. The long-term goal is to design agents that construct their own knowledge of objects through experience, rather than exploiting pre-coded knowledge, and exploit this knowledge to generate complex behaviors that satisfy their intrinsic motivation principles. Self-motivation is defined here as a tendency, based on inborn behavioral preferences, to experiment and to respond to behavioral opportunities afforded by the environment. Over time, the agent integrates, through its experience, relations between interactions and object affording them in the form of data structures, called signatures of interaction, which encode the minimal spatial configurations affording an interaction. The agent then exploits these signatures to recognize distant possibilities of interactions (or affordances), but also incomplete affordances. These structures help the agent defining behaviors that can construct affordances from separated elements. Experiments with a simulated agent show that they learn to navigate in their environment, reaching, avoiding and constructing objects according to the valence of the interactions that they afford.

I. INTRODUCTION

In this paper, we address the problem of generating behaviors that can modify the environment in the purpose of letting affordances emerge, by an artificial agent that initially ignores elements that compose its environment and geometrical properties of its environment. Such an agent can be defined as environment-agnostic [6]. We base our work on a design principle introduced by Georgeon and Aha, called Radical Interactionism [4], that intends to account for cognitive theories suggesting that perception and action are inseparable (i.e. O’Regan [10], Piaget [12]). Specifically, interactions are used to model Piaget’s notion of sensorimotor scheme. In this approach, the agent is given a predefined set of uninterpreted interactions associated with predefined valences, and seeks to enact interactions with positive valences and avoids interactions with negative valences. This motivation principle is called interaction motivation [5], and relates to the problem of intrinsic motivation [11]. The agent perceives its environment by identifying affordances proposed by the environment rather than recognizing objects on the basis of predefined features. This approach addresses the knowledge grounding problem [8] by letting knowledge of objects arise from experience, and introduces no disconnection between agent’s experience and representation of objects.

In previous works, we implemented agents that can integrate and exploit spatial properties and elements of their environments (static [1] or dynamic [3]) discovered through experience, enabling emergence of behaviors satisfying agent’s motivational principles. The mechanisms were robust enough to be implemented on a robot [2]. However these agents can only consider affordances that are actually present in their environment. In this paper, we propose additional structures and decisional mechanisms to generate behaviors that enable constructing objects affording interactions without any preconception about objects and spatial properties of space. Such abilities relate to the problem of tool manipulation based on affordances [15]. Our approach consists in learning properties of interactions and relations that emerge between interactions, without using other structures than interactions: a tool is recognized as such because it can complete an affordance. Moreover, our goal is to make an agent able to generate behaviors satisfying its motivational principle according to offered possibilities of interaction. We tested our mechanism in an environment proposing objects that can be moved by the agent to let new affordances emerge.

II. FORMALIZATION OF RADICAL INTERACTIONISM

A Radical Interactionism (RI) algorithm [4] begins with a set I of primitive interactions. Each primitive interaction i is attributed a valence $v_i$ that defines the agent’s behavioral preferences. At step t, the agent selects an intended interaction $i_t$, and is informed, at the end of step t of the interaction $e_t$ that was actually enacted. The enactment is a success if $i_t = e_t$, and a failure otherwise. A RI agent learns to anticipate the results of its interactions, and tries to enact interactions with high valences.

However, it is difficult to discover spatial properties of the environment with a unique enacted interaction. We thus proposed an extension of the RI model, called Parallel Radical Interactionism (PRI) [1][3]. The PRI model differs from the RI model as it allows to experience simultaneously more than one enacted interaction as the result of an intended interaction. The intuition comes from living beings who receive multiple sensory stimuli while they are acting.
The difference is that, at the end of step $i$, an additional perception. A primary interaction thus consists of an indivisible association between an interaction and an action between an action and a perception, and a secondary interaction an indivisible association between an interaction and an additional perception. A primary interaction thus consists of a couple $i_p = (\text{action}, \text{perception})$, and a secondary interaction, a couple $i_s = (i_p, s, \text{perception})$, with $i_p$ the associated interaction of $i_s$.

Formally, the parallel RI model is similar to the RI model. The difference is that, at the end of step $t$, the agent experiences a set of enacted interactions $\{e_k\}_t$, called enacted context $E_t$. The environment is opaque and the agent can only experience it through interactions.

For example, an animal can move forward, and experience the optical flow resulting from this movement. We thus propose that the agent can experience additional stimuli, in addition to the enacted interaction. However, these stimuli cannot be considered without the movement produced by the enacted interaction. As an example, the optic flow on a retina can only carry a spatial information if it is considered with the movement that generates it. We thus propose to construct new interactions by associating an interaction and an additional stimulus. We call primary interaction an indivisible association between an action and a perception, and secondary interaction an indivisible association between an interaction and an additional perception. A primary interaction thus consists of a couple $i_p = (\text{action, perception})$, and a secondary interaction, a couple $i_s = (i_p, s, \text{perception})$, with $i_p$ the associated interaction of $i_s$.

Defining objects by learning to recognize affordances they provide is abundant in literature [9]-[14]. Signatures differs by the use of interactions, which allows implicit relations between interactions to be discovered, and recognition and localization of distant affordances in space in terms of interactions. See [1] and [3] for more details and examples of implementations.

B. Object Instances

A signature $S_i$ of an interaction $i$ characterizes a context at a certain position relative to the agent, in the form of sets of interactions $\{j_k\} \in \hat{S}_i(1)$. However, each interaction $j_k$ has its own signature, and each context $E_l = \{j_k\}$ is composed of interactions related to the same primary interaction $j$. We thus propose to backmove a signature $S_i$ through a primary interaction $j$ using the following procedure: we note $\hat{S}_i = \hat{S}_i(1)$, where $\hat{S}_i$ is an empty sequence of interactions, and construct:

$$\hat{S}_i[\sigma_0] = \bigcup_{E_l \in \hat{S}_i(1)} E_l \in \mathcal{P}(I) \setminus \{j_k \in E_l, S_j_k(E) > 0\},$$

which characterizes contexts that can afford $i$ after enacting $j$.

As this process can be repeated by considering $\sigma_{a+1} = [j, \sigma_a]$, we can backmove a signature $S_i$ by a sequence of interactions $\sigma$, to obtain a predecessor of $i$, noted $\bar{S}_i$. A predecessor characterizes a context that, if moved through the enaction of the sequence of interactions $\sigma$, affords $i$. We consider that an instance of the object affording $i$ is present at position $\sigma$ with a certitude of $S_i^*(E_l) > 0$. We thus characterize a position in space as sequences of interaction, which relates to the notion of Representative Space of Poincaré, for whom localizing an object in space means considering the movement needed to reach it [13].

C. Proto-object

We define a proto-object of an interaction $i$ as a part of the context affording $i$. We define a partial backmoved signature $S_i^\sigma$ of $i$ as a structure $S_i^\sigma \subset S_i^\sigma \subset S_i^\sigma \neq 0$. A proto-object of $i$ is detected at position $\sigma$ when $\exists S_i^\sigma \cap S_i^\sigma(E_l) > 0 \land S_i^\sigma(E_l) < 0$.

D. Mobile object

We consider that an object is mobile when the same instance of this object, experienced through a set of interaction $E \in E_t$, is detected at positions $\sigma_0 \in E_t$ and $\sigma_{i}, \sigma_{m}$, with $\sigma_m$ the movement of the object relative to the agent (possibly an empty
sequence), i.e. \( \exists t \in \mathbb{N}, E \subset E_t, \sigma_0, \sigma_m / E \in \hat{S}_{t_0}^\sigma \land E \in \hat{S}_{t}^{[\sigma_0, \sigma_m]} \). This means that the agent can move toward the object by enacting sequence \( \sigma_0 \), interacting with the object \((i)\), then moving according to \( \sigma_m \) and interacting again with the object. When the object affording \( i \) is considered as mobile, the space memory gathers properties related to the manipulation of this object. Our model requires two types of properties:

- positions that allow to interact with the same mobile object (i.e. defined by the same context). Indeed, changing the position from which the agent will interact with an object does not change the distance from this object to other proto-objects. A sequence \( \sigma_1 \) is integrated as a manipulation sequence of \( i \) if \( \exists t \in \mathbb{N}, E \subset E_t, \sigma_0 / E \in \hat{S}_{t_0}^\sigma \land E \in \hat{S}_{t}^{[\sigma_0, \sigma_1]} \), with \( \sigma_1 \) not containing \( i \).

- sequences of interactions that allow to interact with the constructed object. Sequence \( \sigma_2 \) is a post-construction sequence of mobile object affording \( i \) that enables enactment of \( j \) once object is constructed if \( \exists t \in \mathbb{N}, E \subset E_t, \sigma_0 / E \in \hat{S}_{t_0}^\sigma \land E \in \hat{S}_{t}^{[\sigma_0, \sigma_2]} \).

IV. SELECTION MECHANISM

We propose three decisional mechanisms that can work in parallel to select the next intended interaction \( i_{t+1} \). Each of these mechanisms adds a utility value to the valence of interactions, which influences the selection of the next intended interaction. The two first mechanisms were introduced in previous work [1][2][3] while the third, called Construction Mechanism was developed to address the problem defined in this paper.

The exploration mechanism allows testing and reinforcing signatures when the certainty of prediction of an interaction or the reliability of the signature are low. Defining the utility value relies on the implementation of signatures. Section V-B gives rules used in the current implementation of the space memory. The exploration utility value \( u^\lambda \) is computed for each primary interaction \( i \). Utility values of secondary interactions are added to the utility values of their associated primary interaction.

The exploitation mechanism helps to generate behaviors that satisfy the agent’s motivational principles at the short and medium terms. This mechanism adds a positive utility value to interactions that enable moving closer to object instances affording interactions with high valence, and a negative value when the object instances afford interactions with low valences. The utility value is weighted by the distance of object instances so that far object instances have a lower influence. As we define positions with sequences of interaction, the distance is given by the length of sequences and the interaction that allows to move closer is the first element of sequences. The exploitation utility value \( u^\varepsilon \) is computed for each candidate (i.e. predicted as a success) primary interaction \( i \) as:

\[
u^\varepsilon = \sum_{o_j \in O^i} v_j \times f(d_{o_j})\]

Where \( O^i \) is the set of object instances that can be moved closer by enacting \( i \), \( o_j \) is an object instance affording interaction \( j \), \( v_j \) is the valence of \( j \), \( d_{o_j} \) is the distance of \( o_j \), and \( f : \mathbb{R}^+ \rightarrow [0; 1] \) is a function that characterizes object influence according to their relative distance. In our implementations, we use the function \( f : x \rightarrow e^{-\gamma \times x} \) where \( \gamma \) is a coefficient that characterizes the decreasing of object influence depending on their distance.

The construction mechanism measures variation of distance between proto-objects composing the same object that the enactment of an interaction afforded by a mobile object can produce. The utility values are computed as:

- for each detected mobile object instance \( o_{i_m} \) affording \( i_m \) including a proto-object \( i_j \) affording \( j \) (i.e. \( \exists S^j_j / S^j_j \subset \hat{S}_j \)), we define a list of couples of possible sequences \( \sigma' = [\sigma_0, \sigma_1] \) and \( \sigma'' = [\sigma_0, i_m, \sigma_m, \sigma_1] \), where \( \sigma_0 \) is the position of \( o_{i_m}, \sigma_1 \) is a manipulation sequences (which can be an empty sequence) and \( \sigma_m \) the movement of the mobile object when interacted (Section III-D). \( \sigma' \) characterizes a position of \( o_{i_m} \) before moving it, and \( \sigma'' \) a position of \( o_{i_m} \) after moving it.

- for each proto-object \( i_j \) included in the mobile object instance \( o_{i_m} \), we detect positions of complementary proto-objects \( \bar{p}_{j,k} \) of \( i_j \), defined as \( \{\bar{p}_{j,k}\}_k = o_j - p_j \). We consider proto-objects for which the position can be considered both with sequences under the form \([\sigma', \sigma_{\Delta_1}, \sigma_3]\) and \([\sigma'', \sigma_{\Delta_2}, \sigma_4]\), where \( \sigma_3 \) and \( \sigma_4 \) are post-construction sequences (possibly the same) of \( i_m \) that allow to enact \( j \). Thus, \( \sigma_{\Delta} \), characterizes the distance between proto-objects \( p_j \) and \( \bar{p}_{j,k} \) before moving the object instance \( o_{i_m} \) and \( \sigma_{\Delta} \), the distance after moving \( o_{i_m} \). We can then estimate the variation of distance between proto-objects produced by enacting \( i_m \) from position \( o_0 \) by comparing length \( l_{\Delta_1} \) of \( \sigma_{\Delta_1} \) and length \( l_{\Delta_2} \) of \( \sigma_{\Delta_2} \).

- The construction utility \( u^\varepsilon \) of each candidate primary interaction \( i \) considers the maximum utility for each interaction \( o_{i_m} \) afforded by a mobile object that can construct an object affording an interaction \( j \):

\[
u^\varepsilon = \sum_{i_m} \max_{\sigma', \sigma'', \bar{p}_{j,k}} \nu_j \times (l_{\Delta_1} - l_{\Delta_2}) \times f' (l_{\Delta_1}) \times f(d_{o_j}) \]

Where \( f' \) is a function that characterizes the influence of the distance between proto-objects and \( f \) is the same function than for (1). In our implementations, we use the function \( f' : x \rightarrow e^{-\gamma' \times x} \) where \( \gamma' \) is a coefficient that characterizes the decreasing of object influence depending on distance between proto-objects.

The mechanism then selects, among candidates \( i \), the interaction with the greatest global valence \( v^e \) defined as:

\[
v^e = v_{i_c} + \lambda \times u^\lambda + \beta \times u^\varepsilon + \delta \times u^e\]

where \( \lambda, \beta, \delta \in \mathbb{R} \) are influence coefficients of the memory. There are no separated learning and exploitation periods.

V. IMPLEMENTATION ON ARTIFICIAL AGENTS

We designed a minimalist experiment to test our mechanisms. This experiment is inspired by the sokoban game, that consists in moving boxes on specific tiles without being stucked, in a discrete 2D top-view environment. We simplified
this principle: when the agent moves a box on a specific tile, the box opens and releases a piece of food. Such an environment seems relevant to test our mechanisms as it requires to move elements in order to make objects affording high valence interaction appear. The simplicity of this environment allows to quickly obtain accurate signatures of interactions from which the agent can extract information required by the construction mechanism.

These agents have a predefined list of six primary interactions, listed below (valences are in parentheses):
- ▶️ move forward by one step (1)
- ▶️ push a mobile element (5)
- ▶️ bump in a solid element (-5)
- ▶️ eat something edible (200)
- ◐ turn right by 90° (-1)
- ◐ turn left by 90° (-1)

We add a set of secondary interactions provided by the agent's visual system, that can detect colors among {red, green, blue, yellow}, and measure distances. Like a *sokoban* player, the agent can perceive its whole environment (although the visual system is ego-centric rather than geo-centric). This simplification dispenses the use of structures learning object permanence [1] that would require a long learning period while not being related to the problems addressed in this paper. Visual interactions consist in seeing a red, green, blue or yellow element while enacting a primary interaction, at a certain position of space. We discretize the visual field as a regular grid of \(25 \times 25\) positions centered on the agents that matches the grid of the environment. These interactions have a predefined valence of zero. We thus define \(6 \times 4 \times 25 \times 25 = 15000\) possible secondary interactions.

### A. Environment

The environment is designed to afford spatial regularities that the agent can discover through its interactions. We defined four types of elements characterized by a color that makes them recognizable with visual interactions:
- wall (green), affording *bump*.
- box (yellow), affording *push*. If a box is in front of a wall preventing it from being pushed, it affords *bump*.
- food (blue), affording *eat*. When the agent eats a piece of food, it becomes a tile.
- tiles (red), affording *move forward*, as well as empty spaces. However, boxes and tiles can be *combined* to get a piece of food.

The environment’s contents can be edited during the experimental run. Figure 2 gives an example of environment.

### B. Implementation of signatures

We propose a new implementation of signatures of interactions that is more efficient in discrete environment than an implementation based on formal neurons [2][3] (although it is less robust), as it enables obtaining reliable and accurate signatures after a shorter period. This implementation is based on the construction of minimal contexts that characterize objects affording interactions. We define a context \(c\) as a set of interactions that can be observed simultaneously. \(c\) is considered as *active* when \(c \subset E_t\). Each interaction is attributed two nodes, the first predicting a success and the second predicting a failure. When the prediction of a node is observed as wrong, the signature integrates the context \(E_t\) as a context that would have inhibited the node. When adding the new context, the signature mechanism first looks for existing contexts \(c_k\) for which \(E_t \cap c_k \neq \emptyset\). If such contexts exist, then the context for which \(\text{Card}(E_t \cap c_k)\) is maximum is removed and replaced by \(c'_k = E_t \cap c_k\). This principle ensures that contexts are quickly reduced to the minimum context affording or forbidding an interaction (basically 3 to 8 tests). While each node collects contexts that inhibit its prediction, it produces less errors. We propose that when a node gives 100 consecutive correct predictions, it can inhibit the other node. If the node predicting a success is inhibited, the signature can be interpreted as “interaction will fail except if a context inhibiting node predicting failure is active”, and vice-versa. In case of complex objects, a context \(c\) can also make errors in prediction. When a context makes an error, it constructs contexts that can inhibit it. Contexts are considered as wrong and removed after 10 errors.

We define the reliability of a context \(c\) in interactional context \(E_t\) as the number of consecutive correct predictions (bounded by 10). The reliability of a signature is defined as the maximum number of consecutive correct predictions among the two nodes, added with the minimum reliability among contexts \(c_k\) for which \(E_t \cap c_k \neq \emptyset\).

### VI. Experiments and Observations

As the construction mechanism is very CPU consuming, we first tested the exploration and exploitation mechanisms to obtain accurate signatures. We then tested the exploitation and construction mechanisms, using signatures obtained in the first part of the experiment. Testing the construction mechanism separately is not problematic as this mechanism is not functional until a large amount of signatures become reliable. Results described in this section use the following coefficients: \(\lambda = 1, \beta = 1, \delta = 2, \gamma = 0.5, \gamma' = 0.001\).

#### A. Signature Learning

We let the agent move in its environment and observe evolution of signatures. Signatures of primary interactions stabilize after 4000 to 8000 simulation steps, depending on interactional opportunities offered by the environment. These
signatures successfully integrated contexts that afford primary interactions, as shown in Figures 3 and 4.

Signatures of visual interactions show an interesting structure: they designate contexts containing visual interactions related to seeing an element of the same color at a position that characterizes the movement produced by the associated primary interaction, characterizing spatial properties of the environment (Figure 4). Signature of seeing a blue element in front of the agent while pushing is also interesting: it designates a context composed of a yellow box in front of the agent and a red tile behind the box. Indeed, enacting push in this context will construct a food element. This signature thus integrated a way to construct the affordance of eat.

While signatures of interactions becomes reliable, influence of the exploration mechanism decreases, while influence of exploitation mechanism increases as the agent becomes able to detect distant object instances. The agent thus moves toward objects affording interactions with high valence (food and boxes), but neglects red tiles.

B. Construction of affordances

We tested the agent equipped with the exploitation and construction mechanism, using signatures obtained previously (Section VI-A) after 42000 decision cycles. We observed the behavior of the agent in several environmental configurations where a box and a tile are present. When the box and the tile are close enough to make influence of construction mechanism greater than other mechanisms (eq. 3) (typically less than 4-5 grid units with current parameters), the agent moves around the box and pushes it toward the tile, changing direction when needed, until the box is in front of the tile. Then, the agent pushes the box to construct the food and eats it. Otherwise, the agent only pushes the box, as push has a positive valence. Figure 5 lists discovered properties of mobile object affording push: the same object instance can be interacted from two additional positions, after pushing a box toward a green wall, the interaction bump can be immediately enacted, and after pushing a box near a tile, three configurations appear, according to the relative position of the tile.

Figure 7 lists object instances and proto-objects that have the greatest influence on the behavior, at each step, in the environmental configuration described in Figure 6. We observe that, at step 4, the exploitation mechanism proposes turn left to interact with the box. When the agent is only driven by the exploitation mechanism, as in the case of previous agents [1][3], the agent effectively turns left and pushes the box (Figure 6a). However, the construction mechanism detects an interesting configuration: after enacting sequence $[\triangleright \triangleright \triangleright \triangleright]$, the proto-objects that enable construction of eat affordance are separated by sequence $[\leftarrow \leftarrow \leftarrow \leftarrow]$ of length 5. After pushing the box (here, sequence $[\triangleright]$), the two object instances are separated by sequence $[\leftarrow \leftarrow \leftarrow \leftarrow \leftarrow]$, which is shorter (length=4). Pushing the object instance from position $[\triangleright \triangleright \triangleright]$ is thus interesting as it can gather proto-objects affording eat. The construction mechanism thus propose move forward with a high utility, related to the high valence of eat.

Configurations given in Figure 8 show that the agent considers every elements of its environment: when a wall is added, the agent interacts with the box from another position, as the
an object instance. This model is thus similar to the exploitation mechanism developed in previous works [1], but that is applied as agents introduced in previous work [2][3]. At step 4, the agent turns left and interact with the object affording push, as push has a positive valence. b), c) and d) the agent is equipped with both exploitation and construction mechanisms. We observe that the agent goes behind the box and push it toward the tile. At step 10, the agent changes its position to not overpass the tile, and complete pushing the box to construct the affordance of eat (step 17). Object instances and proto-object influencing the behavior of the agent are listed in Figure 7.

The agent can thus construct the object affording eat using another sequence of interactions.

VII. CONCLUSION

This work proposes a model that enable an agent to modify its environment to construct possibilities of interaction that are not present at first. Our implementation in artificial agent shows how this model can extract and integrate object properties from signatures of interactions and its own experience of the environment, and exploit these properties to construct objects affording interactions with high valence and thus satisfy agent’s motivational principles.

This model does not rely on plan construction, but increase attractiveness of object instances with which interacting will change the distance between elements that compose the needed object. This model is thus similar to the exploitation mechanism developed in previous works [1], but that is applied between two proto-objects rather than between the agent and an object instance.

In future works, we will implement this model on more sophisticated RI implementations, such as agent in continuous and dynamic environment, or even on robots, and using variable valences and coefficients that rely on internal states of the agent (such as hunger or tiredness), which can instantly change attractiveness of objects and global behavior of the agent. We also intend to extend the model: in the model presented in this paper, we only consider object instances and proto-objects that are accessible. A possible extension of this model could consider a non-enactable path leading to an object instance as an object that may be constructed. A agent equipped with such a model could construct or modify a path that lead to an object, or construct an object that require several steps to be constructed, which constitutes the bases of an abstract construction plan.

REFERENCES