HAL
open science

Constructivist Ambient Intelligent Agent for Smart Environments

Amro Najjar, Patrick Reignier

To cite this version:

Amro Najjar, Patrick Reignier. Constructivist Ambient Intelligent Agent for Smart Environments. PerCom - IEEE International Conference on Pervasive Computing and Communications, Mar 2013, San Diego, United States. hal-00806381

HAL Id: hal-00806381

https://inria.hal.science/hal-00806381

Submitted on 2 Apr 2013

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.

Constructivist Ambient Intelligent Agent for Smart Environments

Amro Najjar*[1] and Patrick Reignier†[2]

[1]ISCOD, Henri Fayol Institute, ENS Mines Saint-Etienne , Saint-Etienne, France
[2]Université de Grenoble, Alpes,, Grenoble, France

March 30, 2013

Abstract

Building a smart home is a multi-disciplinary and challenging problem. Our goal is to build an agent that can propose context aware services to the users. High variability of users' needs and the uniqueness of every home are difficult to handle using "Classical AI". We propose an alternative approach inspired by Developmental Artificial Intelligence and Constructivism Theory. Being constructivist means that the agent builds its knowledge *in situ* through user's interactions. This continuous interaction process enables the user to customize or bring up the system to meet his personal needs. We have made a first experiment by learning schemas from a simulated two-weeks home scenario. This preliminary experiment gives us indications that Constructivism is a promising approach for ambient intelligence.

Keywords

Ambient intelligence, Machine learning, Autonomous mental development, Self organizing feature maps

1 Introduction

In the founding text of ubiquitous computing [1], Mark Weiser describes a world where computers would be quiet, seamless, proactive and would serve us without "invading" our consciousness or occupying our attention. In this *"world to come"*, our life would be full of comfort, serenity and satisfaction [2] [1].

We have just started the second decade of the 21th century, yet Weiser's world has not come, it still resides in the *"proximate future"* [3]. This observation motivated many researchers to reexamine Weiser's vision and try to rectify it so it becomes reachable. Most of the researchers attribute the unreachability of Weiser's vision to the fact that it assumes the emergence of a what R. Jose and al. in [4] call the *"techno-utopia"*. The term techno-utopia reflects the fact that the world as imagined by Weiser is too ideal, it will always be a *world to come* and will never become real.

However, many significant advances have been accomplished. As pointed by Aaarts from Philips Research [5], technology is less and less a blocking element in the development of ambient intelligence.With the recent advances in micro-electronics, wireless networks and energy management, it is now possible to deploy sensors and actuators in a "legacy home", outside a laboratory. The "ambient" term of "Ambient Intelligence" is more and more becoming a reality, even if it

*najjar@emse.fr
†Patrick.Reignier@inria.fr

does not match the initial Weiser's vision, where infrastructures would be continuous and smooth. We live instead in a world where those infrastructures are naturally messy.

One of the important problems in the context of smart environments is the mismatch between the scientists scenarios and users real expectations. There is no typical scenario. Each user has its own needs and expectations, evolving over time in an opportunistic manner. It is economically impossible to conceive and manually produce ambient intelligence applications adapted to each person [6]. An important research issue is to create a smart environment that can support deep customization of user's ambient experiences.

Machine Learning is a promising approach to try to automatically adapt smart environments behavior to user's needs. But it usually requires expert knowledge to choose and parametrize the selected algorithms. This choice is based on the available data (type of sensors for example) and of the task to be learned (reinforcement function, evaluation function). If the context changes (adding sensors, task modification), it is necessary to call again the expert to change the system accordingly. This problem is addressed by Developmental Learning Approaches.

In the next section, we will give a first definition of Developmental Learning. We will then describe constructivism, one of the psychological theory in developmental learning and how it can be implemented to solve IA problems. We will then describe Nestor, our learning agent which tries to help users to control their smart home.

2 Developmental Learning

In 1950, Alan Turing postulated that building a program that simulate an adult's mind is an overwhelming, almost impossible, task that is intractable in terms of programming time and cost. Instead, he proposed to build a program that simulate a baby's mind at birth and educate it [7]. This approach was not part of the mainstream research of what is called *classical AI*. Nevertheless, it was picked up by Gary Drescher, who in the late eighties, built an agent that discovered its micro-world in a developmental fashion [8]. Drescher's work did not receive much attention till recent years when this approach became an active research field under the title: *developmental approach to AI (DAI)*.. It as been defined by F. Gurin as "programs that learn to make their own abstraction from sensor data, learn their own commonsense knowledge (...) rather than having these things specified by human designers" [9].

DAI approaches are mainly unsupervised, unscheduled and task-less [10]. They rely on three concepts : **abstraction**; filtering the overwhelming, mostly irrelevant sensor input data and extract useful environment regularities [10], **anticipation**; using these extracted patterns to predict the environment behavior and act in advance in order to compensate for the inherent delays and move from random behavior to purposeful one [11], and **self-motivation**. Self motivation is the factor that should drive the agent discovery of the environment regularities. Without it, the agent discovery of the environment fall into randomness [10]. Hence, in our environment the agent will be motivated by the need to satisfy the user and thus user satisfaction is its ultimate goal.

As a consequence of abstraction and anticipation guided by motivation, the agent organizes *hierarchically* its representation of the world and its behavior repertoire. The lowest behavioral level contains innate behaviors or reflexes, then "each subsequent level combines an abstraction mechanism and an anticipation mechanism to adapt to the environment based on experience" [11]

3 Constructivism

One of the first theory that has been used in developmental AI is developmental psychology and more particularly, constructivism. Constructivism is the theory of knowledge that argues that humans generate knowledge and meaning from an interaction between their experiences and ideas.

The Swiss psychologist Jean Piaget claimed that infants have knowledge of schemas through which they process, comprehend and react to their environments. Those schemas are divided into three categories: sensorimotor schemas (coupling observations and actions), propositional schemas (classifying observations) and operational schemas (describing systems and their inner working).

Sensorimotor and propositional schemas are constructed in newborns immediately, whereas operational schemas come in a later stage of development [12]. The developmental process of schemas is carried using two mechanisms: assimilation and accommodation. Assimilation describes how humans perceive and adapt to new information, fitting them into pre-existing cognitive schemas to make sense of them. Accommodation, unlike assimilation, is the process of restructuring pre-existing schemas to handle this new information [13].

Assimilation might lead to a success or a failure; when the infant actual experience is aligned to his internal knowledge representation, this produces a success. On the other hand, when the internal representation is too simple, performing the schema leads to a failure. When a schema fails many times, the infant discovers that the environment contradicts her schemas and therefore, the infant attempts to "accommodate" the environment. Accommodation is the process by which failure leads to learning. When reality does not meet our expectations of the world any more, our reaction is to modify our expectations and reform our internal representation. The infant learns from her experience: knowledge is grounded on behavior.

Learned schemas are inherently hierarchical. Complex schemas are created by integrating low level schemas based on regularities observed in the environment.

4 Constructivism as an AI approach

The first computational model of Piaget schemas (the sensorimotors only) has been proposed by Dresher in his PhD thesis [8]. The schemas operated an artificial baby in a 7*7 grid world. The baby had an eye, a mouth and a hand that allowed him to grab objects in the environment when they are in his reach. A schema was defined as a triple (Context, Action, Result). A schema captures a regularity about the world. A regularity predicted by a schema is not true all the time. This is reflected by a ratio associated with each schema indicating its reliability. A classic example of a Drescher schema:

$$\begin{cases} Context = HandInfrontOfMouth \\ Action = HandBackwords \\ Result = HandToutchingMouth \end{cases} \tag{1}$$

At the beginning, the system has no initial knowledge of the environment behavior. The system innately "knows" how to perform a set of *primitive actions*. When turned on for the first time, it associates a *bare schema* to each action [8]. A bare schema is a schema with empty context and empty result indicating that the agent does not know when this action should be triggered (context) and what will be its consequences (result).

4.1 Marginal Attribution

Marginal attribution is the process that transforms bare schemas to concrete schemas with context and result. Bare schemas are also augmented with what Drescher calls *extended context* and *extended result*:

- **Extended result:** the positive and negative transition for every known item[1]. It is the percentage that this item is respectively *on* or *off* when the schema (the action) is performed.
- **Extended context:** it is also associated to every known item and indicates the probability that the schema succeeds when the item is on and when it is off. A schema succeeds if the result is correctly predicted by the result of its action.

[1] A context element

When schemas are executed, the agent perceives their contexts and result. When transition correlation crosses a threshold, a new schema is created out of the original bare one with that item on the result part. When a context item makes a schema more reliable, a new schema is created with that item on the context part.

Marginal Attribution Example

let's say that the infant knows how to click on the "switch on" button of a TV remote controller. Initially the infant knows nothing about the effects of this action and its context. But after some attempts, the infant deduces that this action switches on TV. Rewriting the previous sentence using schema terms, we can say that the infant added the result *+TVSwitchedOn* to the action part of the so called switch tv schema. The next question the infant should answer is "why TV is not being switched on every-time I click in the button?". After more attempts, the infant will discover (randomly or through imitation) that in order for the remote controller to work, it should be pointing towards the TV (context).

The artificial infant perceives the world via a set of primitive binary sensory items that reflect the low-level concepts of the environment. When the artificial baby attempts to *assimilate* the environment using only primitive schemas, it faces a reliability problem; the world is too complex to be handled using only primitive schemas[2]. Thus, the baby reacts by restructuring its knowledge. It integrates its primitive items to form *synthetic items*; higher-level items that are used to create more reliable higher-level schemas.

Drescher schema mechanism has been criticized for its computational inefficiency and its incapacity to scale outside simple grid-world [14]. To overcome those limitations, Chaput has proposed the Constructivist Learning Architecture (CLA) [12]. CLA is based on the information processing approach [15] (a neo-Piagetan formalism). To solve the computational intractability of Dreshers' approach, Chaput is first creating an abstraction of the high dimensional sensor space using Self Organizing Maps (SOM) [16] as an unsupervised clustering approach. a SOM projects high dimensional data on a 2D neural map. Each neuron maintains a prototypical input representation of the corresponding cluster. SOMs are sensitive to input patterns frequency: a more frequent pattern will be represented with a greater accuracy. It preserves also topology: two similar prototypes will correspond to close neurons.

5 Nestor

We will present in this section our constructivist approach for a smart environment assistant: Nestor.

5.1 Environmental setup

We need to capture the user activities in smart environments for several days. Before deploying our system on a real house, we have developed a smart home simulator over Siafu[3]. In this simulated home, we have sensors (light sensors, sound sensors, movement sensors etc.) and actuators. Actuators control shutters (raising the light level in a room), the TV sound etc.

A human user, Bob, lives in this smart home composed of three bedrooms, a living room, a dinning room, a kitchen, a bathroom and a garage. Nestor's role is to assist him by proposing/doing services associated to the current context.

Bob behaves according to a pre-defined scenario. The scenario describes daily activities (for week days and week-end). Random variations are automatically generated to add data variability.

[2] A primitive schemas is a one that is composed of primitive sensory item as a context, an action and a primitive item as a result.

[3] http://siafusimulator.sourceforge.net/

5.2 Overview of our experiment scenario

Nestors knowledge of the environment (the house and Bob), is continuously evolving to keep track of the changing behavioral regularities of Bob. Nevertheless, when deployed for the first time, Nestor who has no initial knowledge, should have the opportunity to observe Bobs activities in the smart home in order to capture useful patterns. we denote this process as "the education phase". Then in the next phase, the" working phase", Based on the user regularities captured in the education phase Nestor engages the user in active participation and hence, user-system dialog is initiated.

5.3 Our Method

5.3.1 Education Phase

The purpose of the education phase is to help Nestor to extract patterns of user's interaction with the smart home. The recorded sensors data are used to train a learning algorithm based on Chaput's CLASM architecture (Constructivist Learning Architecture Schema Mechanism) [12]. A SOM is associated to each available action. An observed action is every action that can be: *performed* by the user, *detected* by Nestor, and *performed* by Nestor using one of its actuator. When an action triggered by the user is detected at time t, the values of all the sensors at time t, t+1, t+2 and t+3 (to capture the action's effect on the environment) are clustered by the associated SOM. Each cluster is a context prototype for the action. After two weeks, SOM is harvested: best prototypes are converted into schemas (context, action, result).

5.3.2 Working Phase

Initial schemas have been learned: Nestor will now start interacting with the user. When the context of an initial schema matches the current context, Nestor proposes its associated action to the user. If the user accepts and if the schema produces the expected results, the schema reliability is increased. Otherwise, it is decreased. If a schema fails, Nestor will then ask the user if he wishes to do this action in another close context. This other context is selected from a topologically close prototype on the action's SOM. For instance, in the morning if the user rejects Nestor proposition "Switch TV on" Nestor will ask "Bob, do you want to watch tv in the evening?".

If two lower-level schemas satisfying some criteria reach a reliability threshold, Nestor integrates them into a higher-level schema. Higher-level schemas are built in a pairwise fashion [17]; when the context of a schema matches the result of a previous schema, the two schemas are combined in one higher-level composite schema. If a higher level schema fails, Nestor reacts by falling back to a lower-level (graceful degradation).

Whenever a schema is created (high-level or low-level), Nestor asks Bob to name it. This name is used in user-system discussion. This discussion is a part of the "knowledge co-construction".

5.4 Results

Our agent Nestor was tested in the simulation environment described in section 5.1. During the education phase that lasted two simulation weeks, Nestor received a big amount of data from its sensors (more than 500K entries). Nevertheless, It was able to perform the required *abstraction* and harvest useful user behavior regularities from them. For instance a simplified harvested schema is: *In the evening of week-days regardless of all value of the lights sensors, propose to turn on tv.* Using these harvested schemas, Nestor was able to *anticipate* the user's behavior. Obviously, the usefulness and pertinence of these schemas need to be evaluated. However, this necessitates undertaking subjective tests where human participants need to be involved and is considered as a necessary step in our future research.

6 Conclusion

Creating a *classical AI system for smart environment* is an overwhelming task that is described by [18] to be an *AI-complete* problem. To build a "*classic*" system fed by the developer ontologies, we need to create a generic, complete, consistent and accurate model of the environment. But every home is different and each human user is unique. We want to build an agent able to respond to the dynamic fast-changing nature of the smart homes, to exhibit graceful degradation properties when confronted with unexpected environmental changes (fallback mechanism on constructivism approaches), and robust to noisy sensor input and unclear user's intentions.

We are currently working on a constructivist approach to build this agent. Although, Nestor is still basic and this work is in its preliminary stage, we think that this approach is promising, specially in homes where the agent that satisfies the user is the one that is "brought up" (co-constructed) by the user himself.

References

[1] M. Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[2] Y. Rogers, "Moving on from weisers vision of calm computing: Engaging ubicomp experiences," *UbiComp 2006: Ubiquitous Computing*, p. 404421, 2006.

[3] G. Bell and P. Dourish, "Yesterdays tomorrows: notes on ubiquitous computings dominant vision," *Personal and Ubiquitous Computing*, vol. 11, no. 2, pp. 133–143, Nov. 2006.

[4] R. Jos, H. Rodrigues, and N. Otero, "Ambient intelligence: Beyond the inspiring vision," *Journal of Universal Computer Science*, vol. 16, no. 12, p. 14801499, 2010.

[5] E. Aarts and B. de Ruyter, "New research perspectives on ambient intelligence," *Journal of Ambient Intelligence and Smart Environments*, vol. 1, no. 1, pp. 5–14, 2009.

[6] M. van Doorn, E. van Loenen, and A. P. de Vries, "Deconstructing ambient intelligence into ambient narratives: the intelligent shop window," in *Proceedings of the 1st international conference on Ambient media and systems*, ser. Ambi-Sys '08, ICST, Brussels, Belgium, 2008.

[7] A. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460, 1950.

[8] G. L. Drescher, *Made-up minds: a constructivist approach to artificial intelligence.* Cambridge, MA, USA: MIT Press, 1991.

[9] F. Guerin, "Learning like baby: A survey of AI approaches," *The Knowledge Engineering Review, accepted, to appear*, 2009.

[10] J. Schmidhuber, "Formal theory of creativity, fun, and intrinsic motivation (1990–2010)," *Autonomous Mental Development, IEEE Transactions on*, vol. 2, no. 3, pp. 230–247, 2010.

[11] D. Blank, D. Kumar, L. Meeden, and J. Marshall, "Bringing up robot: Fundamental mechanisms for creating a self-motivated, self-organizing architecture," *Cybernetics and Systems*, vol. 36, no. 2, pp. 125–150, 2005.

[12] H. Chaput, *The constructivist learning architecture: A model of cognitive development for robust autonomous robots.* Citeseer.

[13] Wikipedia, "Piaget's theory of cognitive development — wikipedia, the free encyclopedia," 2011.

[14] C. Witkowski, "Schemes for learning and behaviour: A new expectancy model," 1997.

[15] L. Cohen, "An information-processing approach to infant perception and cognition." 1998.

[16] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[17] O. Georgeon, J. Morgan, and F. Ritter, "An algorithm for self-motivated hierarchical sequence learning," in *Proceedings of the International Conference on Cognitive Modeling. Philadelphia, PA. ICCM-164*, 2010, pp. 73–78.

[18] L. Leahu, P. Sengers, and M. Mateas, "Interactionist AI and the promise of ubicomp, or, how to put your box in the world without putting the world in your box," *Proc. Ubicomp'08*, p. 134143.