Generating Natural Behaviors using Constructivist Algorithms

Olivier L. Georgeon OGEORGEON@UNIV-CATHOLYON.FR
UR CONFLUENCE, Sciences et Humanits - UCLy, LIRIS CNRS UMR5205, Lyon, France
Paul Robertson PAULR@DOLLABS.COM
Doll inc. Lexington, MA 02421, USA
Jianyong Xue JIANYONG.XUE@LIRIS.CNRS.FR
Universit Claude Bernard Lyon 1, LIRIS CNRS UMR5205, F-69622 Villeurbanne, France

Editors: Minsky, H. and Robertson, P. and Georgeon, O. L. and Minsky, M. and Shaoul, C.

Abstract

We present a project to design interactive devices (smart displays, robots, etc.) capable of self-motivated learning through non-goal-directed interactive behaviors (e.g., curious, emotional, playful behaviors). We use and improve algorithms inspired by constructivist epistemology that we have designed previously. These algorithms incrementally learn sequential hierarchies of control loops in a bottom-up and open-ended fashion, and continuously reuse the learned higher-level control loops to generate increasingly complex behaviors that exhibit self-motivation. This project contributes to research in self-supervised learning because the learning is driven by low-level preferences that under-determine the devices future behaviors, leaving room for individuation, which, in turn, opens the way to autonomy in learning.

Keywords: Cognitive architecture, constructivist learning, enactive AI, self-motivation, control loop, constitutive autonomy.

1. Introduction

How to create a little robot that would behave like a kitten? Besides being able to solve simple problems (e.g., finding a path to a specific location, finding the appropriate sequence of actions to obtain energy, etc), it should also exhibit open-ended, exploratory, and playful behaviors. Many authors refer to the latter as non-goal directed behaviors, that is, behaviors that external observers or software developers would not interpret or conceive as directed towards a specific terminal goal. For example, the robot may be playing with fellow little robots or toys; explore its environment; follow people; perform somersaults as if it merely enjoyed it, etc. Sometimes, those behaviors may go wrong and these robots would look endearing or hilarious. If we were able to create such robots, the internet would surely be full of videos of them just as it is full of videos of kitten today.

We believe that generating some level of animal-like behavior is not out of reach with the current hardware resources at our disposal, given the appropriate software design approach. Moreover, it could be useful for reasons both practical and theoretical. The practical applications to humans are comparable to the role of domestic animals. Mankind is willing

to spend huge amounts of money to enjoy the company of domestic animals, suggesting that they fulfill many needs at various levels: social, emotional, recreational, etc. Devices capable of similar behaviors might fulfill some of these needs. As for the theoretical utility, it would help understand better the role that open-ended behaviors play in the development of intelligence. Most current research on AI is focusing on solving problems modeled a priori, and categorizing data in predefined categories, but a lot remains to discover on how to generate behaviors in the absence of a pre-modeled problem and predefined categories.

In this paper, we examine the hypothesis that devices equipped with algorithms based on constructivist design principles could generate animal-like behaviors. Constructivist principles are drawn from constructivist epistemology and applied to AI software design. The three constructivist principles we consider the most important are the following:

1) Input data is the *outcome*[1] resulting from actions, as opposed to being a direct representation of predefined features of the environment ("percepts"). Actions coupled with their expected outcomes form *control loops*. Control loops relate to Wiener (1948)'s *feedback loops* in cybernetics theory except that we do not necessarily use control loops to maintain a set point.

2) Knowledge is about possibilities of interaction rather than about the world "in itself". This principle follows Kants idea that cognitive beings do not know the *noumenal world* but only know the *phenomenal world*. In compliance with principle 1, a constructivist algorithm learns increasingly sophisticated regularities of control loops in an open-ended fashion. This learning relates to enactive AI (Froese and Ziemke, 2009) in that it occurs from enacting interactions rather than from interpreting passively-received input data supposedly representing a pre-given world.

3) The algorithms purpose is to generate interesting non-goal directed behaviors in an unknown environment (or, ultimately, to pursue goals that it has defined by itself), as opposed to reaching predefined goals defined by the designer under the form of particular states in a predefined problem space. In compliance with principles 1 and 2, the algorithm has no access to a representation of predefined goals and problem space.

Upon these principles, we have developed learning algorithms that go beyond cybernetics control theory by learning hierarchies of control loops in a bottom-up and open-ended fashion. In this paper, we outline a roadmap that aims to demonstrate that we could use these algorithms in a device to generate animal-like behaviors. Section 2 outlines the principle of these algorithms in reference to our previous publications. The next sections sketch a roadmap to demonstrate the capabilities of these algorithms when used in smart displays and domestic robots. Section 3 presents a short-term roadmap that will exploit our current existing algorithms. Section 4 presents a middle term roadmap that involves developing a new cognitive architecture based on the constructivist principles.

2. Constructivist learning

As introduced in Section 1, our algorithms control the devices behavior through control loops rather than perceptual data. The designer predefines a set of possible control loops by

1. We use the term *outcome* to refer to input data resulting from action in a similar way as quantum physics uses it to refer to data collected from experiment. The experiment, the outcome data, and the experimenters representation of reality are interrelated since the experimenter is a part of reality.

specifying the commands to actuators along with the outcome data expected from sensors. For example, a control loop may consist of controlling the trajectory of the touch sensor while maintaining the outcome within a certain range that corresponds to feeling an object. If the object is indeed present, the outcome achieved will comply with that expected. If the object is absent, the outcome will belong to a different range that will match another predefined control loop. The algorithm has no notion of the object in itself but uses the set of control loops that were enacted to estimate the possibility of enacting further control loops. All the predefined control loops are limited in time, either through a preset maximum duration or through an exit condition depending on the outcome. The learning occurs by recording hierarchical sequences of control loops as illustrated in Figure 1.

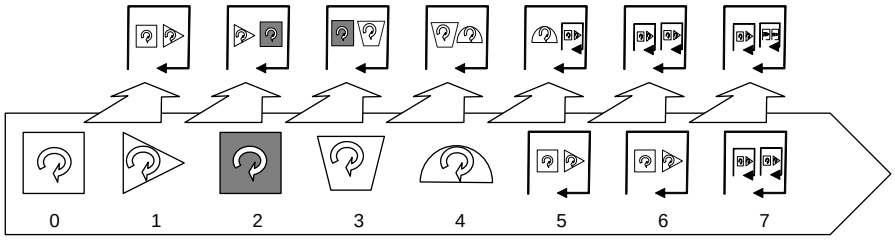

Figure 1: learning hierarchical sequences of control loops. Bottom: the shapes on Steps 0 to 4 represent predefined primitive control loops enacted by the device over time. On each step, a higher-level control loop is recorded consisting of a sequence of the last two enacted control loops. Top: the squared arrowhead loops represent the higher-level control loops learned on each step, containing the sequence of their two sub-control loops. The second-order control loop learned on Step 1 is enacted on Step 5 (lasting as long time as Step 0 plus Step 1). The third-order control loop learned on step 6 is enacted on step 7. After that, the algorithm can continue recursively learning higher-level control loops.

It is crucial that the behavior selection mechanism does not use any interpretation of the meaning of the control loops. This is compliant with constructivist principle 1 that the agent implements no presupposed interpretation of input data, and with principles 2 and 3 that the agent implements no presupposed representation of the world and of goals. This stands in sharp contrast with non-constructivist AI approaches.

Figure 2 illustrates the mechanism of control loop selection: on Step 4, the human designer can implement different action selection criteria depending on the kind of behaviors she wishes the device to exhibit. For example, if the algorithm selects an action associated with a control loop that has the least been tried in a given situation, then we expect the device to behave as if it were curious to try new things. If the algorithm selects an action associated with the control loop that has the highest probability to result in the most anticipated control loop, then we expect the device to appear to prefer being in the flow (Steels, 2004). The designer can also associate a predefined numerical valence with each predefined control loop, and implement criteria to select actions associated with control

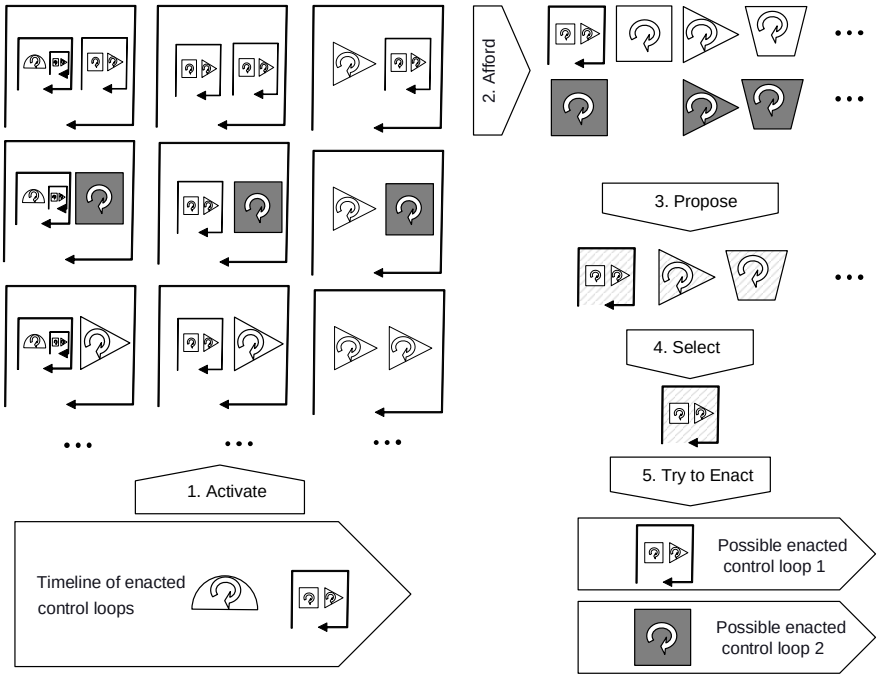

Figure 2: control loop selection. Step 1: previously-learned higher-level control loops are activated in memory when their left-hand element matches the control loops that have just been enacted by the device. The figure shows nine higher-level control loops activated by the last two enacted control loops, the last enacted control loop alone, and the right-hand part of the last enacted control loop alone. Step 2: the right-hand parts of the activated higher-level control loops are considered afforded in the current context. The figure shows seven afforded control loops. Step 3: afforded control loops are aggregated into possible actions. The figure shows 3 proposed actions represented as crosshatched shapes. Step 4: an action is selected from the set of proposed action upon criteria that depend on the kind of behavior the designer wants the device to exhibit. Step 5: The device tries to enact a control loop associated with the selected action, which may result in different enacted control loops depending on the outcome received from the environment. The figure shows two possible control loops that the algorithm can expect. Only one of them is actually enacted depending on the situation of the device in its environment. The cycle will recommence on the basis of the last enacted control loop.

loops that have the highest valence. The device will appear to enjoy interactions that have a positive valence and to dislike interactions that have a negative valence—a motivational drive we have called interactional motivation (Georgeon et al., 2012).

For a more precise description of this algorithm, we refer the reader to our previous papers (Georgeon and Riegler, 2019). We have reported preliminary demonstrations in simulated environments in videos (e.g., https://youtu.be/t1RO5S4mBEY) showing interesting behaviors in a noisy setting. The next section presents how we will use these algorithms in interactive devices to demonstrate natural behaviors.

3. Six-month roadmap

On a six-month term, we will develop two demonstrations in parallel: a robotic demonstration and an interactive display (smartphone, tablet, etc.) demonstration. Later, we will reunite these two demonstrations by installing an android interactive display on the robot. For the robotic demonstration, we have chosen Turtlebot (https://www.turtlebot.com/) because it is a fairly widespread robot that runs on an open source operating system (Ubuntu) with the standard Robotic Operating System (ROS), and offers significant flexibility for configuration and evolution. For the interactive display demonstration, we will develop an app using a multiplatform framework such as Flutter in order to run on android and iOS devices (Figure 3).

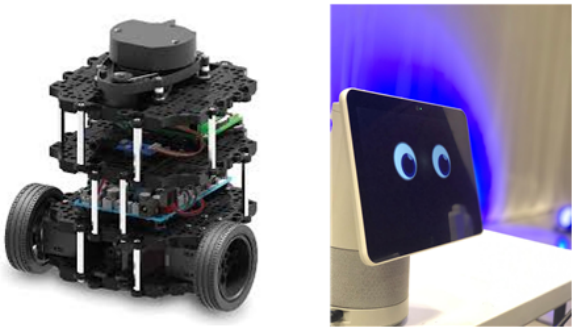

Figure 3: the turtlebot and the smart display running the Hoomy app by Hoomano.

For each of the two demos, the roadmap is split into four tasks:
- Implement a set of primitive control loops.
- Implement the interface with our constructivist algorithm.
- Implement the interface with our visualization and behavior analysis software tool.
- Run, record, and report the demonstrations.

The constructivist algorithm and the behavior visualization and analysis software tool will be available on a server. The robot and the smart display will interface with this server through a REST API.

Table 1 presents an initial list of primitive control loops to implement in the robot. Table 2 presents an initial list of actions and outcomes that can be associated in various combinations to form primitive control loops in the smart display.

Name	Description
Move forward	Move forward for a certain distance and stop if an obstacle is encountered
Spin	Spin in place of a certain angle and stop if facing a standalone object
Snort	Move in place (forward/backward, or left/right) and detect a sign of the user.
Shoot	Move forward and detect contact with an object through touch sensor.

Table 1: example primitive control loops implemented in the robot.

Action performed by the device	Outcome detected by the device
Play a sound	The screen is being stared at
Display an image	A button is being pressed
Vibrate	The device is being shaken
Show an animation	Some words are heard

Table 2: example categories of actions and outcomes implemented in the smart display. In each category, various primitive actions and outcomes can be implemented: various sounds, images, vibration patterns, animations, buttons, words recognized, etc.

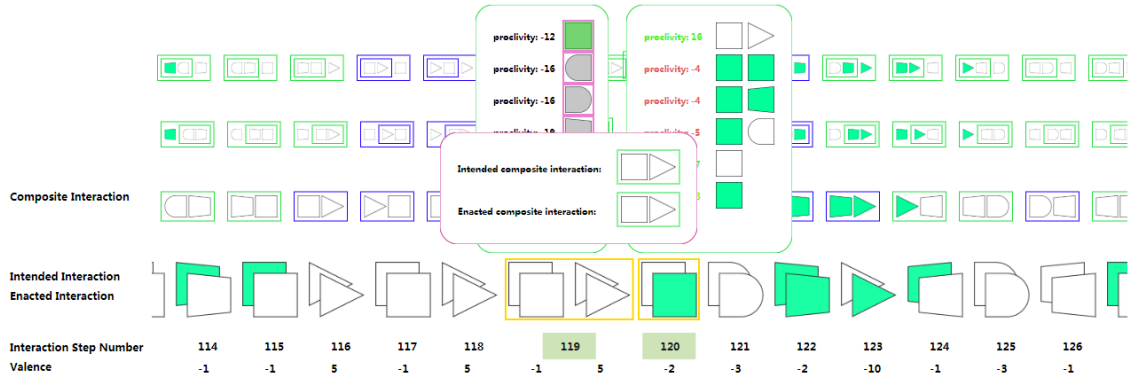

Figure 4: screenshot of the behavior visualization and analysis tool. A second order control loop is enacted on step 119. By clicking on it, the user opens a tip window showing the intended and enacted control loops (in this example, they are the same), and other tip windows showing various information helping understand why this control loop was selected at this step.

Figure 4 shows a screenshot of the behavior visualization and analysis tool. In our experience, this kind of tool proved indispensable to design and debug the algorithm, and to report the results of experiments.

4. Five-year roadmap

The constructivist algorithm presented in Section 2 implements sequential/temporal behavior control, but, to generate more sophisticated behaviors, we need not only to control behaviors in time but also in the 3D space.

Towards this goal, the five-year roadmap involves four tasks:

-Defining advanced control loops involving 3D spatial control.

-Extending the constructivist algorithm into a constructivist cognitive architecture capable of simulating and organizing control loops in the 3D space.

-Improving the behavior visualization and analysis tool to display behaviors in time and space.

-Run, record and report the demonstrations.

4.1. Defining advanced control loops

We will program additional primitive control loops in the robot using more displays and sensors: leds, speaker, touch sensors, microphones, etc. We will also assemble the smart display with the robot. This will allow more sophisticated user-robot interactions. We will generate more complex outcomes through advanced algorithms such as image, sound, and gesture recognition.

We will exploit the possibilities of the Inertial Measurement Unit (IMU) available in the Turtlebot and in most smart displays to control the control loops in the 3D space.

4.2. Improving the cognitive architecture

We will extend the constructivist algorithm into a constructivist cognitive architecture to make it capable of organizing behavior in the 3D space. Figure 5 shows the structure of this cognitive architecture.

Some may argue that implementing the presupposition of the 3D spatial structure of the world in the cognitive architecture contradicts the constructivist principle 2. We, however, believe that this is acceptable for two reasons. Firstly, this presupposition is consistent with what we know about animal and human brain, namely that it incorporates many spatial memory structures (superior colliculus, hippocampus, etc. e.g., Gross and Graziano (1995)). Secondly, this presupposition is not restrictive as long as we seek to control devices in the 3D world.

The five-year project will work on improving the interactions between the different modules of the cognitive architecture. In particular, spatial memory will help deal with persistence of objects, and will allow a form of reflexivity consisting in simulating different possible spatial interactions before selecting one. We will build upon our previous work on a 2D cognitive architecture (Georgeon et al., 2013), which already produced preliminary results in a simple simulated environment shown in video https://youtu.be/LjOck5ts_2g.

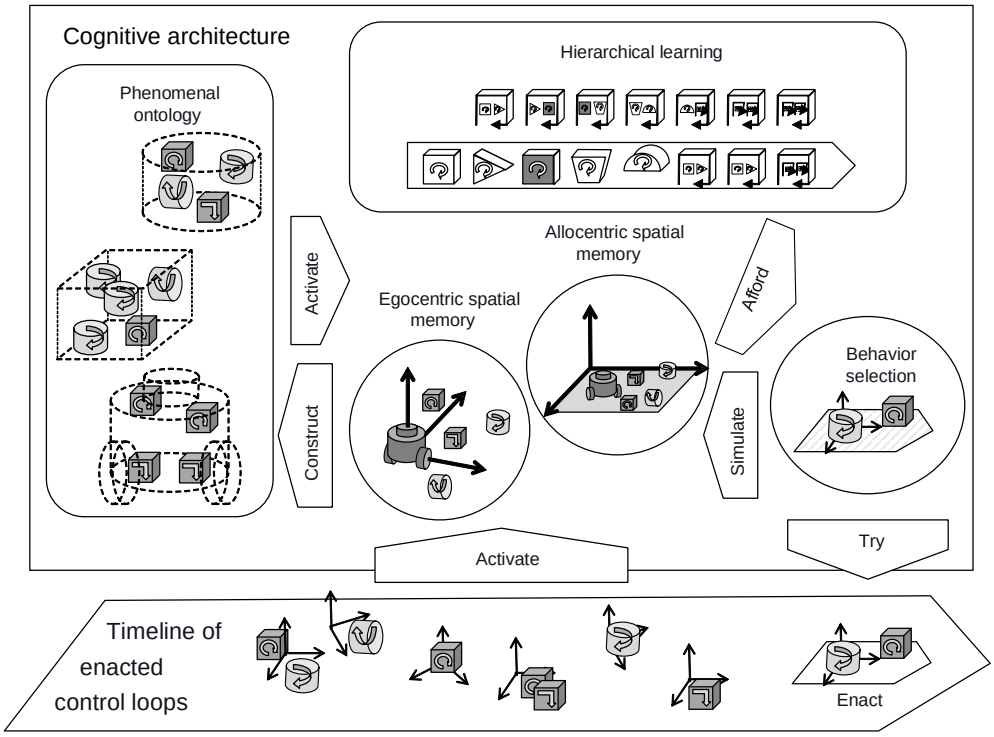

Figure 5: constructivist cognitive architecture to generate natural behaviors in 3D space. control loops are represented as 3D blocks to highlight the fact that they involve control in the 3D space. The cognitive architecture incorporates the hierarchical learning component presented in Section 2 (top right). It also incorporates different kinds of spatial memory (egocentric and allocentric) used to localize and track the position of control loops (center). The phenomenal ontology (top left) is a memory of categories of phenomena (objects as they appear through interaction) defined by the control loops that they afford. Among other types of phenomena, it contains a learned representation of the robot and its own kind.

5. Conclusion

In previous studies, we have proposed constructivist design principles to create AI algorithms, and we have implemented a constructivist-learning algorithm based on these principles. The motivation for this work was theoretical: we wanted to examine and demonstrate the implication of constructivist epistemology to artificial intelligence. In particular, we argued that this algorithm allows constitutive autonomy through self-programming, and individuation—features that many authors consider a requirement for cognition (Georgeon and Riegler, 2019). We have run experiments in simple simulated environments showing some interesting behaviors, albeit rudimentary.

In this paper, we propose the hypothesis that these constructivist principles and algorithms could be used in a robot or a smart display to generate natural interactive behaviors in the real world. We outline a roadmap to create demonstrations to validate this hypothesis.

If we can confirm this hypothesis, we expect to open the way to commercial products in the domain of entertainment and hi-tech. These products will have interesting behavioral traits such as curiosity, emotional behaviors, co-adaptation with their user. This will make them somewhat similar to animals; they will be able to generate empathy from their users and will fuel societal and ethical debates related to the status of devices that will appear increasingly sentient as we make progress in open-ended artificial intelligence.

Acknowledgments

This paper greatly benefitted from discussions with Xavier Basset from Hoomano. Olivier Georgeon acknowledges financial support by ANR under contract ANR-11-DPBS-0001.

References

Tom Froese and Tom Ziemke. Enactive artificial intelligence: Investigating the systemic organization of life and mind. *Artificial Intelligence*, 173(3–4):466–500, 2009.

Olivier L Georgeon and Alexander Riegler. Cash only: Constitutive autonomy through motorsensory self-programming. *Cognitive Systems Research*, 58:366–374, 2019.

Olivier L Georgeon, James B Marshall, and Simon Gay. Interactional motivation in artificial systems: Between extrinsic and intrinsic motivation. In *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL)*, pages 1–2. IEEE, 2012.

Olivier L Georgeon, James B Marshall, and Riccardo Manzotti. Eca: An enactivist cognitive architecture based on sensorimotor modeling. *Biologically Inspired Cognitive Architectures*, 6:46–57, 2013.

Charles G Gross and Michael SA Graziano. Review: Multiple representations of space in the brain. *The Neuroscientist*, 1(1):43–50, 1995.

Luc Steels. The autotelic principle. In *Embodied artificial intelligence*, pages 231–242. Springer, 2004.

Norbert Wiener. *Cybernetics or Control and Communication in the Animal and the Machine*. Technology Press, 1948.