From biological to numerical experiments in systemic neuroscience: a simulation platform

Nicolas Denoyelle, Maxime Carrere, Florian Pouget, Thierry Viéville, Frédéric Alexandre

From biological to numerical experiments in systemic neuroscience: a simulation platform

Nicolas Denoyelle[12], Maxime Carrere[123], Florian Pouget[12], Thierry Viéville[1] and Frédéric Alexandre[123]

Key words: Simulation, Computational Neuroscience, Virtual Reality.

Abstract Studying and modeling the brain as a whole is a real challenge. For such systemic models (in contrast to models of one brain area or aspect), there is a real need for new tools designed to perform complex numerical experiments, beyond usual tools distributed in the computer science and neuroscience communities. Here, we describe an effective solution, freely available on line and already in use, to validate such models of the brain functions. We explain why this is the best choice, as a complement to robotic setup, and what are the general requirements for such a benchmarking platform. In this experimental setup, the brainy-bot implementing the model to study is embedded in a simplified but realistic controlled environment. From visual, tactile and olfactory input, to body, arm and eye motor command, in addition to vital interoceptive cues, complex survival behaviors can be experimented. We also discuss here algorithmic high-level cognitive modules, making the job of building biologically plausible bots easier. The key point is to possibly alternate the use of symbolic representation and of complementary and usual neural coding. As a consequence, algorithmic principles have to be considered at higher abstract level, beyond a given data representation, which is an interesting challenge.

1 Introduction

Computational neuroscience is often presented as a way to better understand the complex relations between structures and functions in the brain. Particularly, these relations are complex because they are not symmetrical: one stucture participates to several functions and functions are distributed among many structures. This is an important limitation against developping a model of a structure in isolation, which is often the case in computational neuroscience. This is not sufficient to emulate one behavioral function, but participates only to studying some of its properties, with the risk of neglecting the key influence of another structure on this function.

[1] Inria Bordeaux Sud-Ouest, 200 Avenue de la Vieille Tour, 33405 Talence, France [2]LaBRI, Université de Bordeaux, Institut Polytechnique de Bordeaux, CNRS, UMR 5800, Talence, France [3]Institut des Maladies Neurodégénératives, Université de Bordeaux, CNRS, UMR 5293, Bordeaux, France e-mail: Frederic.Alexandre@inria.fr

Consequently, for modeling studies interested in integrative and behavioral neuroscience and in the emulation of behavioral functions, this analysis is a plea for designing brain models including many brain structures. In addition, in the framework of studying behavioral functions, the brain must also be considered as a complex system in interaction with the body and the environment. Two important consequences can be drawn. The first consequence is related to the model itself. Additional modules must be considered, to allow for the sensation and processing of signals from the environment (exteroception) and also from the body (interoception). Designing such a network of brain structures and modules at the interface with the outer and inner world includes not only understanding how each subsystem (visual, motor, emotional, etc.) works but also how these subsystems interact as a whole, to yield emerging behaviors, i.e. effects that result from interactions between subsystems. The second consequence is related to the use of this complex system. Studying and validating functional models of brain structures at a macroscopic behavioral scale cannot be performed with restrained artificial static paradigms but requires experiments in complex environments, with realistic sensory-motor tasks to be performed, including high-level interactive behaviors (e.g. survival strategy in the presence of prey/predators) and long-term protocols (since both statistical studies and biologically-plausible learning mechanisms require long epochs). Such paradigms are to be related to biological experiments conducted on animals. These statements are not only characterizing brain models working in interaction with their environment, they also give strong requirements on the tools that must be designed to simulate these models and to experiment them.

Designing such tools is also an excellent way to address at the same time the two main objectives of such brain models at the macroscopic scale. One one hand, they are intended to serve neuroscientists as a new platform of experimentation, on which they can apply their classical protocols of observation and analysis of animals at the behavioral as well as electrophysiological levels. It is consequently important that neuroscientists can observe the inner activity of the models, as they use to do for example with electrodes (but we can imagine that this observation in digital models might be more easy than in the real brain). It is also important that they can define classical behavioral protocols like they do in animals (eg. fear conditioning) in order to observe the resulting behavior and the corresponding brain activation. Defining such protocols implies that the structure of the external world (e.g., maze, food magazine) as well as its intrinsic rules (eg. tone followed by an electric shock) should be easy to design.

On the other hand, these tools are also intended to serve computer scientists as a way to design artificial autonomous systems, driven by brain models. In this case, it is important for the supposed properties of the models (e.g., capacity to learn, robustness to noise or changing rules) to be assessed by rigorous evaluation procedures, as it is defined for example in the domain of machine learning. In this case also an easy access must be proposed both to the inner circuitry of the models and to the specification of the external world.

With in mind this double goal of offering convenient tools to both scientific communities, we report in this paper the specifications that we have elaborated and

present the corresponding software platform that we call VirtualEnaction. We also introduce the case study of a behavioral function presently under study in our team, pavlovian conditioning, as an illustration of the use of VirtualEnaction. Before that, some more words must be said to justify the need for such a platform.

2 Problem position

Concerning the nature of such a simulator, real robotic systems are often used and answer particularly well to the second requirement about a realistic environment. However, building viable robotic systems and making them evolve in realistic environments (e.g. natural sites) for long periods of time (e.g. several days) is just too expensive in term of cost and manpower in many circumstances and particularly during early phases of development. Furthermore, the goal of such simulation is not only to make a demo, but also, and more importantly, to study and quantify the behavior of functional models of the brain. As a consequence we not only need a complex, long-term, realistic experimental setup, but we also need a controllable and measurable setup where stimuli can be tuned and responses can be measured. In fact, real environment complexity and parameters are intrinsically difficult when not impossible to control. This is the reason why we propose to use a digital simulator implementing realistic survival and other biological scenarios.

A step further, available macroscopic models of brain functions are not designed for ”performance” but to properly implement phenomenological concepts that have been investigated in some cognitive or behavioral framework. They would therefore have ”no chance” in a real world. Note that recent computer science mechanisms designed without any constraint regarding biological plausibility but only towards final performances are nowadays probably more efficient but that are not relevant regarding the brain behavior explanation.

As a consequence we also need a setup which can provide a “simplified environment”, for systemic models of the brain at the state of the art not to fail immediately. We must also take into account the fact that (i) such models are rather slow to simulate (unless huge computer power is available), and that (ii) they are not supposed to focus on precise issues regarding low-level sensory input or motor output but on integrated cognitive functions and the resulting behaviors.

This, in addition to technical constraints, yields three key characteristics:

1. No real-time but a look-and-move paradigm : The main restriction we propose is to have the simulator running at a “slower” time (i.e. using several seconds to simulate one real-time second) and also to consider discrete time sampling. This seems obvious as far as digital simulation is concerned, but in terms of underlying framework, this has several consequences (e.g., giving up the possibility for a human observer to interact with the simulation, restraining to clock-based (and not event-based) dynamical models, etc.) [Taouali et al., 2011].
2. No real robotic control but only motor command : Since in the nervous system motor control seems to be a hierarchical system with high-level motor com-

Fig. 1 Two examples of digital experimental environments for systemic neuroscience. Left: A minimal environment corresponding to a standard maze reinforcement learning task (source: one of our virtual enaction built). Right: A complex environment in which survival capabilities are to be checked (source: landscape encountered when playing with the standard game).

mands, while their closed loop execution is delegated to the peripheral motor system [Uithol et al., 2012], we may accept to only simulate gesture and displacement at a rather symbolic level such as "stand-up" or "turn 90° rightward". This indeed cancels the possibility to study sharp phenomena of motor interactions with the environment but allows us to concentrate on high-level control such as action selection mechanism and motor planning.

3. Complex but not necessarily natural visual environment: The third main restriction we propose to accept is to consider a complex visual environment (with visual textures, several objects in motion, etc.) but not to invest in the simulation of a realistic natural scene simulation. The reason of this choice is that natural image vision is an issue already well studied [Hyvärinen, 2009]. The general conclusion is that biological visual systems are tuned to natural image statistics, decomposed by the early visual front-end in such a way that higher-level visual input only relates on cues orthogonal (in a wide sense) to natural image statistics. In other words, the job regarding this aspect is done by early-vision layers and we may consider more stylistic visual cues at a higher-level. Depending on the study, we may also wish to work on either a pixelic or a symbolic representation of the visual scene. See [Teftef et al., 2013] for details of how the early-visual system implements such dual representation.

3 System description

We consider that a "brainy-bot", i.e. the implementation of a global model of the brain functionalities, interacts with its environment with the simple goal of surviving. Our objective is to simulate the sensory-motor interactions of this bot with respect to its environment. Examples of such surroundings are shown in Fig. 1.

Survival is precisely defined as maintaining vital variable values in correct ranges, as formalized in, e.g., [Friston, 2012]. In our context, health, food, water, energy, and oxygen are the vital state variables. The bot has access to these values. These variables decrease or increase with time since the bot body is supposed to

consume the related resource depending on its activity, or change in the presence of an external event (e.g. energy during a predator attack), and restore resources. Restoring resources is obtained either by ingesting items or taking rest (i.e., when making the choice of stopping an action, with the benefit of vital resource increase and the drawback of not acting on the environement, this might be a short-term policy choice if vital variables are low and is a middle-term policy choice otherwise).

The environment structure is very simple and made of "blocks". Each object in this environment (including the floor, relief, ..) is a collection of blocks. Each block is defined by its 3D position, orientation and size, roughness, hardness, temperature, and color. Some blocks correspond to eatable or drinkable resources. Other entities correspond to objects that interact with the bot (e.g. predators that attack the bot or lava to avoid). At this level, survival corresponds to avoid or kill the predators and find resources to eat or drink.

The bot anatomy is functionally made of a body, an arm and an head/eye. It carries a bag with objects found in its environment. The body can move at a given speed and in a given direction, and also rotate at each step to a given angle. It can also jump up to a given relative height, or knee down to take a rest. The head/eye can gaze in a given yaw/pitch direction. The arm can perform predefined symbolic gestures : take an object in hand out of its bag, put the object in hand into its bag, either drop or throw the object in hand, ingest the block in hand (food or water), grasp the object in front of him. The arm can also attack (quantified by a force value and with or without an object in hand) the object in front of him. This is the complete description of the bot motor command output in the present context.

The bot sensory input corresponds to cues related to the blocks which are around it. The touch cues allow the bot to estimate the roughness, hardness and temperature of the object in hand. The olfactory cues allow to estimate the smell type and intensity of objects close to it (computed by integrating average values over the blocks characteristics). At the bot level, pixelic vision provides an image of the visual field view (i.e., calculating the blocks texture and color projection on the virtual retina). Finally, the proprioceptive cues correspond to gaze direction estimation.

In order to quantify the bot behavior, the interface provides an additional access to the bot absolute position and orientation in space. Symbolic vision is also available, as a list of blocks visible in its visual field, with an access to the block characteristics.

Fig. 2 Left: The software interface is trivial: each implementation of a brainy-bot provides an initialization routine `initBot()` and a stop function `stopCondition()`, while at each time-step the `brainDo()` method is called. Right: All status, input and output functionalities are available via a simple API. For instance, hand or vital input, body and head displacement, gestures of resource injection and attack against predator are shown.

An adaptation of the `minecraft` open game software yields the proper answer to this wish-list and the so called `virtualenaction` is an open-source free-license implementation of these specifications. Each user buys a end-user low-cost `mojang` license ($<$ 20\$) for `minecraft`, while `virtualenaction` is free of use under a CeCILL-C license. Fully-documented scripts facilitate the installation of the software bundle under Linux OS. The bot is implemented in either `C/C++`, or possibly in `Python` (via an existing `swig` wrapper) or other computer languages. It uses a simple API, as described in Fig. 2. Furthermore, in order to both observe in slow-down real-time the bot behavior and interact with the digital experiment, a graphic user interface is available as described in Fig. 4. The simple architecture of the platform is schematized in Fig. 3.

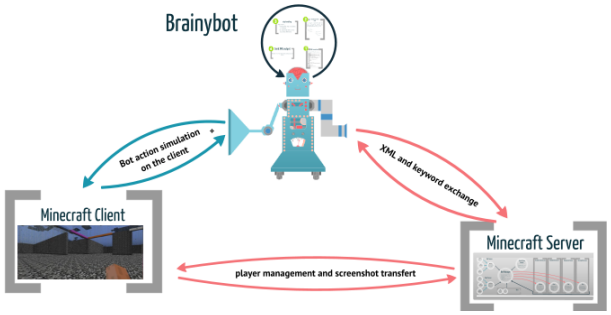

Fig. 3 The platform architecture. At the user level, only the notion of (i) game server where the environment and its mobile objects interactions are computed, (ii) game client where the game-play is rendered and (iii) brainy-bot where the brain model simulation is issued have to be taken into account.

4 Brainy-bot generic functionality

Even when restraining to functional modeling, it is not possible to simulate all sub-systems of the brain at the same level of details. We thus must represent parts of the system with efficient models, but without biologically plausible models. A few sets of generic functionalities are to be proposed to this end. Let us discuss three of them.

-1- Associative memory Any system has not only to take into account the present input and output in order to generate a proper behavior, but also to store and recall past information. Information is however never exactly the same and similar pieces of information (i.e., being almost equal up to a given threshold) have to be considered as a unique item. Such a memory must not only store and retrieve information, but also define a notion of indistinguishable information.

In our context this has a special meaning: the input/output information corresponds to a hierarchical logical-structure[1], mixing quantitative and qualitative information[2], and using[3] structures like *set* or *sequence*, in addition to *t-uples* as containers. More precisely:

```
@action: {
 life: { stop: @bool },
 body: { translation: { speed: @percent, direction: @degree }, orientation: , rest: @bool },
 head: { vertical: @percent, horizontal: @percent },
 attack: { strongness: @percent },
 jump: { height: @percent },
 gesture: { touch: @bool, ingest: @bool, throw: @bool, climb: @bool },
 hand: { put: { slot: @index }, get: { slot: @index }, count: @count }
},
@sensation: {
 location: { position: @position, orientation: { yaw: @degree, pitch: @degree} },
 vital: { health: @percent, food: @percent, water: @percent, oxygen: @percent, energy: @percent},
 vision: { blocks: [ @block* ], entities: [ @block* ] },
 touch: { blocks: [ @block* ], entities: [ @block* ] },
 hand: { roughness: @percent, hardness: @percent, item: @item },
 olfaction: { intensity: @percent, toxicity: @percent, friendliness: @percent, edibleness: @percent },
 inventory: [ @item* ]
},
@block: {
id: @index,
position: @position, roughness: @percent, hardness: @percent, temperature: @percent, color: @color
},
@item: {
id: @index, count: @count, slot: @index
},
@position: {
x: @real, y: @real, z: @real
```

Notion of divergence. As a consequence, it is not obvious to define a "distance" between to such state values (i.e. the whole input/output data structure). But this is required to decide if they are distinguishable or not. Our claim is that the correct notion is the notion of *premetric*[4] or *divergence*[4], for two state values $\mathtt{s}_1$ and $\mathtt{s}_2$:

$$d(\mathtt{s}_1,\mathtt{s}_2) \geq 0 \text{ and } d(\mathtt{s}_1,\mathtt{s}_2) = 0 \Leftrightarrow \mathtt{s}_1 = \mathtt{s}_2$$

as being the weakest concept that allows to state that two values are distinguishable if and only if $d(\mathtt{s}_1,\mathtt{s}_2) \geq \varepsilon > 0$, for some threshold ε. It appears that all premetrics we need are always symmetric (i.e., it is a semimetric[4]). In our context, the divergence corresponds to a Euclidean distance for a `position`, angular distance for an `orientation`, while the divergence between two t-uples is simply defined as the weighted sum between each t-uple values divergence, for a given set of weights, while the divergence with respect to a undefined value must be defined for this definition to be coherent. At this stage, divergence computation is linear with respect to the state structure size. The introduction of dissimilarity computation of t-uples is the cause of being in the context of a semimetric and not a complete metric with the triangle inequality verified.

[1] Such a structure is of common use in computer science: It corresponds to, e.g., a `XML` logical-structure, or the `Json` syntax underlying data model.

[2] The following scalar values are used: `bool` for an either true or false Boolean value, `percent` for a proportion value between 0 and 1, `degree` for an angle in degree, `index` stands for a non-negative integer index, `count` stands for a non-negative integer count value, `color` stands for a RGB color value, `real` stands for a unbounded decimal value.

[3] Sets are unordered lists without repetition, Sequences are sequential lists, and t-uples map strings, or names, to their corresponding value.

[4] Definition available on `https://en.wikipedia.org`

A step further, we need pairing[4] divergence between two unsorted sequences (i.e., it corresponds to the classical "stable marriage problem"[4]). This tool is required to define how two sets of blocks (e.g., for symbolic vision input) match, in order, e.g., to recognize a visual object. This pairing is not a distance in the general case.

We also need to define an edit distance (a generalization of the Levenshtein[4] distance, called the Victor-Purpura[4] metric) to define how two temporal sequences match.

Both pairing and edit divergence can be implemented as algorithms quadratic with respect to the set or sequence size.

Let us also mention that informing the bot about its absolute position and orientation or about symbolic information of the scene, as it is the case, is a way to shortcut its localization and sensory modules and provide integrated cognitive information, without considering the related computational issues.

With this set of tools, the mechanism of associative memory of state values is entirely specified, and it must be pointed out that since the space of states has no special structure the algorithms allowing to retrieve a value in the memory (namely the closest value whose divergence with respect to the input value is minimal), or insert a new value (if not already in the memory in an indistinguishable form), or delete it, can not have less than a complexity of $O(\text{memory-size})$, since hash-coding is not compatible with retrieving a similar value.

-2- State Categorization A generic key cognitive feature is the capability to "extract" symbolic information from a bundle of quantitative or qualitative values. During a supervised learning phase prototypes of state values with the corresponding known category are registered. Then, for an incoming state the most plausible category is calculated. Such mechanism includes sensory events detection (e.g., detect the presence of predator from sensory cues), object labeling (e.g., an element as a resource to ingest): Such examples are qualitative values. A biologically plausible support vector machine[4] mechanism is available to this end, with versatile uses [Viéville and Crahay, 2004]. This also includes quantitative values (e.g., associate a reward to a given sensation or action), and support vector machines also include the capability to categorize using a floating point value.

In our case, the issue is the following: If a simple nearest neighbor mechanism (i.e., the category of an incoming state value is set (or interpolated) using the category of the nearest neighbor (or neighbors)) is used, then the notion of divergence introduced before is sufficient. However, such a mechanism is known as being the worst in term of generalization (i.e., inferring correctly the category for incoming state value not corresponding to the prototypes) and robustness (i.e., providing reliable results even if some spurious prototypes).

Notion of gradient. In order to introduce a better mechanism of categorization that will optimize a criterion, we need to define the notion of *gradient*, i.e., small variation of a given state value. One impediment is the fact that some variables are discrete variables. In the present case, `index` and `count` are not to be optimized, thus remain fixed. Boolean variables `bool` however correspond to effective choices

or meaningful information, and must be taken into account. As a consequence, we would have to deal with the combinatory complexity of discrete values optimization, known as NP-hard. To overcome this caveat, we simply propose to embed each $b = \texttt{bool}$ value in bounded $v = [0,1]$ value, with the obvious correspondence:

$$v = \text{ if } b \text{ then 1 else 0}; \quad v = b > 0.5$$

With these specification choices, given a state s we can define a small variation ∂s as small variations of all quantitative scalar values (namely `real`, `color`, `degree`, `percent` and `bool` represented by a $[0,1]$ value). Given a state criterion, i.e., a real value function of the state, we can optimize this criterion with respect to quantitative values. This construction implicitly maps a state to a N-dimensional real space of the different quantitative variables. This is not exactly an Euclidean space since real value like color or angle lies in some non-Euclidean metric spaces, but usual differential methods can be used.

Following the method in [Viéville and Crahay, 2004] that simply requires optimization with respect to the prototypes state value (and contrary to the original SVM algorithm that proceed in a dual space, which we can not define here since we do not have a concrete metric) we thus can improve the raw nearest neighbor mechanism, by eliminating or merging useless prototypes, and modifying prototypes values to maximize the margin between classes. It must be understood that this method will not (i) perform qualitative optimization (e.g., will not find that we must add or delete a block to represent an object) and (ii) is sub-optimal since we reuse a well-founded method in a context where not all assumptions are verified.

- *-3- Gesture interpolation*: At a functional level a "behavior" (i.e., a complex gesture) can be specified as finding a path from an initial state (e.g., being hungry while food is known to be present elsewhere) to a final state (e.g., having the food ingested) taking constraints into account (e.g., avoiding or moving aside obstacles on the way). Generic specification of such problem and universal algorithms to solve them exist, in relation with harmonic control which is a biologically plausible framework (i.e., with fully distributed computation based on diffusion mechanism) [Viéville and Vadot, 2006]. This also corresponds to the fact that, in the motor cortex, a motor command is represented by its final state or "end point", while the trajectory generation to attain this final state is generated elsewhere.

In order to define such a mechanism we simply need the notion of state divergence and state gradient introduced previously. Obstacle to avoid is simply defined by constraints (e.g. real value function of the state that must remain, say, positive). The goal is attained when the divergence to the final state is below the indistinguishability threshold. The algorithm proposed in [Viéville and Vadot, 2006] builds harmonic potential, here in the real space on which the state quantitative variables are mapped. It requires the operation of projection (of a state onto a constraint) to define a repulsive point, and such algorithm reduces to an optimization problem, implementable thanks to our different design choices.

Though we present the simplest aspect of this class of methods, harmonic control is easily linked to optimal control [Connolly and Grupen, 1993], in link with reinforcement learning considering a non-finite realistic state-space [Todorov, 2004].

As a conclusion, this section has analyzed to which extent some powerful generic machine learning techniques can be reused in this context and adapted to the symbolic hierarchical data structure defining the bot input and output. The two main ideas are to use a weaker notion of distance and accept adaptation mechanism of the state restrained to quantitative values. Further developing the link with machine-learning algorithms applied on such data structure in order to define high level algorithmic ersatz of cognitive functions is a perspective of this work.

5 Neuroscience application

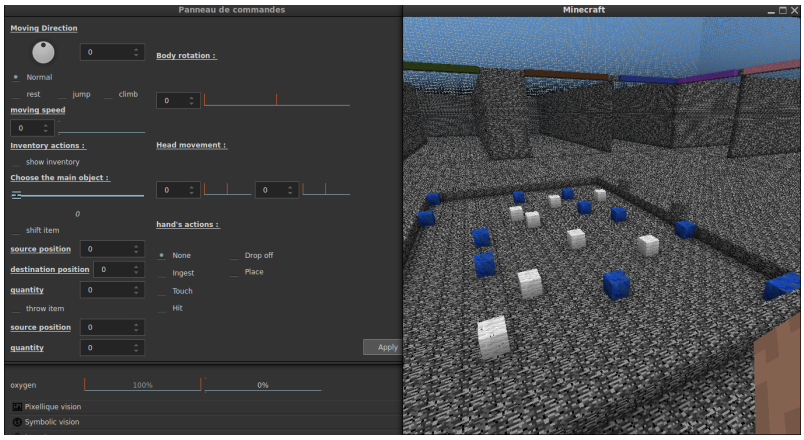

Fig. 4 The software's graphic user interface. In order to observe and control the experiment, a view of the bot vision, status, input and output is available via interactive panels. It is also possible to "cheat" by controlling these variables values, in order to debug an experiment and understand in details what happens in such a complex system interaction.

Let us now discuss how this setup constitutes a step towards integrative neuroscience digital experimentation.

First of all, let us compare this project with complementary connected projects. The AnimatLab is a software platform allowing to simulate embodiment (biomechanical simulation of a body) allowing to investigate the relation between brain and body [Cofer et al., 2010]. Furthermore, it proposes a neural network architecture for the implementation of cognitive functions. On the contrary, the present framework has a rather limited description of the embodiment, but a much larger set of possible interactions with the environment. A step further, not only artificial neural network models are usable in VirtualEnaction, whereas the interface with

any existing neuroscience simulation tool (e.g., python based neural simulators, see [Brette et al., 2007, Davison et al., 2008]) is straightforward. This feature is essential, since we must simulate the system at different modeling scales, as developed now. The Morse is a generic simulator for academic robotics, with realistic 3D simulation of small to large environments, allowing complete integration with any simulation tools. It outperforms concurrent systems like Webot. The interest of VirtualEnaction with respect to Morse is twofold: Since we target integrative cognitive tasks of survival which is exactly what happens with the Minecraft environment, using this specialized product is far simpler and somehow more demonstrative. In terms of performances, as being less sophisticated (using a simplified 3D rendering, while Morse has all 3D capabilities) and being agnostic in terms of programming languages (i.e., allowing fast C/C++ implementation of user modules, whereas Morse is limited to Python scripts) the VirtualEnaction platform is a priori expected to be more efficient in terms of CPU usage. However, with a larger humanpower all what has been developed within VirtualEnaction could have been developed in Morse.

The main application regarding neuroscience is to test *cognitive computational models* in realistic conditions. Very simply, a behavioral experiment is performed on an animal model or on humans, usually with a training phase, the measurement phase and the data analysis. In order to formalize the obtained result a computational model is proposed that explains the data, and may also have prediction regarding other falsifiable future experiments. The present software and methodological tool allow us to propose to enhance this very general paradigm in the following directions:

- Test the model prediction for several others experimental conditions or model parameter ranges : The idea is to reproduce the experimental setup in this virtual environment (e.g., a delayed reward task, an exploration paradigm) and connect the computational model to this paradigm. As for usual computational modeling, the chosen biological measurements (e.g. neural activity, task success performance) are simulated when running the model. Such model is indeed expected to reproduce qualitatively the ground truth, for a given set of parameters values. A step further, it is very important to numerically verify what happens when modifying any quantitative or qualitative parameter value. If the numerical sensibility is so strong that the results cannot be reproduced for some tiny parameter variation, the model is meaningless because biological values make sense as a numerical range, not a single number. If the numerical sensibility is so weak that any parameter value produces the expected result, this parameter is meaningless and a simpler model very likely explains the same data set. The key-point here is that such predictive verification is not only going to be possible, given a fixed data set, but for any data set obtained during virtual environment simulations. In other words, the computational model variants are going to be always tested in-situ. With no practical bounds on the experimental variants (e.g., number of trials, sensory input precision, task complexity).
- Design new experimental paradigms to confront the computational model to falsifiable conditions : Building an experiment considering an animal model, training the animals for weeks, restarting from the beginning if it appears that there

is an unexpected trap (e.g., the task is too simple or unfeasible, or does not allow to discriminate between two concurrent models) may be a huge work. Starting to design the experiment in a virtual environment completely changes the method: the hypothesized computational model is first tested in silico (i.e., neither in vivo, nor in vitro, but in a software environment) and only confronted to the biological reality in a second stage. The workplan is inverted with respect to usual neuroscience studies, but in computational engineering (e.g., designing new airplanes) this is exactly the way it goes until a few decades. It is however not new in neuroscience, at the scale of mesoscopic brain map study (see e.g., [Chemla et al., 2007] or [Brette et al., 2007]). The key point here is that such approach is now possible at the behavioral level, considering sensory-motor interactions with a simple or complex environment. Such process also obviously yields a parsimonious use of animal models. A step further, it lays down the challenge of performing realistic experiment with a computational model of the brain behavior and not only considering toy situations where the plausibility of the model can not be checked.

6 Case study: pavlovian conditioning

The degree of equivalence between simulation outcome and neuroscience experimental results is a key issue. This is the reason why we have chosen a software platform where usual behavioral neuroscience experiments can be reproduced "as a whole". Such classical experiments include pavlovian and operant conditioning, spatial navigation with tasks involving the focus of attention and multi-sensory integration and other high level cognitive functions involving working memory and planning. The main brain structures involved in such tasks are the thalamus and posterior cortex for sensory processing, the basal ganglia system for selection of action and decision, the hippocampus for episodic memory and the prefrontal cortex for the temporal organization of behavior. Depending on the task, other more specialized structures like the amygdala, cerebellum, superior colliculus, hypothalamus and others may also be considered. These tasks are often related to fundamental survival programs (nurishment, reproduction, integrity of the body) and are organized towards goals, defined from the environment (acquiring appetitive goals or avoiding noxious ones) or from the body (satisfying internal needs), which endows them with a strong behavioral ecological anchoring.

As an illustration, we introduce pavlovian conditioning, presently studied in our team, including the integration of related computational models in VirtualEnaction. Pavlovian conditioning is a fundamental learning capability, allowing to identify aversive and appetitive events in the environment and also exploited in many complex tasks. This learning relies on the ability of animals to automatically detect biologically significant stimuli (also called US, unconditional stimuli) and to trigger corresponding reflexes. Pavlovian learning occurs when a neutral stimulus (conditioned stimulus, CS) reliably predicts the occurrence of the US. After acquisition,

the CS presented alone will also trigger a response and allows the animal to anticipate the nature of the US to come. For example, in the case of fear conditioning [Herry et al., 2008], the US can be an electric shock automatically triggering freezing. Subsequently to the pairing of the US with a CS like the auditory perception of a tone, the CS alone will evoke freezing. It will also allow the animal to anticipate the nature of the US and prepare more adapted behavior like an escape, depending on the nature of the task and the environment.

Fear conditioning has been extensively studied because in its basic forms, it involves a rather simple and well know cerebral circuit, associated to accessible stimuli in the environment. The amygdala is the key structure for the acquisition and expression of fear conditioning [LeDoux, 2007]. Its lateral nucleus receives extero and interoceptive information from the thalamus and the sensory cortex and is reported to perform the acquisition of CS-US associations and to activate its central nucleus, a motor structure responsible for the expression of pavlovian responses.

This simple associative learning has also been studied because beyond its simple expression, it demonstrates non trivial characteristics. For example, the CS can be more complex than a simple tone, because it includes a temporal structure (delays) or specific spatial characteristics like the context of learning (the room in which conditioning occurs) or the composition of several stimuli (compound stimuli). In this case, the sensory cortex is not able to supply the information in an adapted configuration and the hippocampus has been shown to be necessary for the acquisition of this CS-US association, through its projections to the basal nucleus of the amygdala [Eichenbaum et al., 2012]. The involvement of the hippocampus is not really surprising here, since this structure is known for its role in episodic learning, another kind of associative learning of arbitrary relations between features including spatial and temporal contexts [Rolls, 2010].

Learning CS-US association can also become more complex when the association is not deterministic but can vary in time. This can be due to stochasticity (the association is probabilistic) and also to changes in the rules of our dynamic and changing world, as it is observed in the case of extinction [Herry et al., 2008]. Experiments in neuroscience have shown that when a predicted US no longer occurs, this does not correspond to the removal of the learned CS-US association but to its temporary inhibition by the medial prefrontal cortex, learning history of performance in behavior. Particularly, this means that, in the case of renewal (the old rule becomes valid again), reacquisition of the CS-US association is immediate because it was not forgotten but only inhibited.

Pavlovian conditioning is also studied for its impact, at a systemic level, on other kinds of learning in the brain. Without going too deep in details, it can be mentioned that sucesses and errors of US prediction have also a deep impact on episodic learning in the hippocamus and on perceptive category learning in the sensory cortex and also in operant conditioning. It must be also underlined that stimuli learned by the pavlovian procedure are often re-used as goals in operant conditioning.

This short overview of pavlovian conditioning was only meant to indicate that behavioral sequences related to this kind of learning can be studied at different levels of complexity, which is the case in neuroscience, depending on the context of

the experiment. Accordingly, our VirtualEnaction platform must adapt to these contexts. In its simplest form, the environment includes simple sensory CS and US and the body must express stereotyped avoidance or attraction movements. In this case, our brainy-bot integrates a model of the amygdala as we can find in the literature or as we developed in the team. In more complex cases, we might for example take into account the spatial context in which the CS is perceived, which requires to integrate a functionality like episodic memory. In this case, we can integrate a generic functionality of symbolic information manipulation as described in section 4 or a more advanced neuronal model of the hippocampus, depending on the aspects that are to be more deeply studied in relation to biological experiments. The principle would be the same for integrating a model of the medial prefrontal cortex or a simple statistical analysis of the recent history of received and predicted US, in case of an experiment related to an extinction procedure.

In addition to the design of the system adapted to the task under study, another critical step is about the design of the scenarios to be tested. Here, scenario stands for the description of the environment (stimuli and their perceptual dimensions, contexts like a cage or a maze), of the laws ruling this environment (which stimulus is a US and triggers an unconditional response; which stimulus is a CS and how reliably it predicts the US; and other physical laws of the environment like objects that can be moved) and of the precise timing of sequence of events (defining protocols implementing acquisition, extinction and renewal procedures). In its most basic form, such a design is generally thought to fit biological experiments and will be used for the purpose of comparison with behavioral performances as well as with patterns of neuronal population activation and learning. In a more advanced form, this system could be also exploited on the side of computer science, either in terms of performance of learning, to be evaluated in a machine learning perspective, or in terms of autonomous agent, to be evaluated with regard to the range of available behaviors and to the pertinence of their selection. Implementing not only pavlovian conditioning but also operant conditioning is a strong prerequisite for this latter perspective, as the authors are presently considering.

7 Discussion

In this paper, we have presented the platform VirtualEnaction, for implementing brain models and their bodily and external environment. Beyond the description of the functionalities of the platform, we have also explained why it was corresponding to a need in the behavioral and computational neuroscience community. We have illustrated these elements with a case study related to an important class of learning, from a behavioral point of view. Indeed, let us mention that this platform has already been used for preliminary digital experiments about Pavlovian conditioning [Gorojosky and Alexandre, 2013] involving the functional modeling of the amygdala and hippocampus, decision making mechanisms in link with reinforcing signals yielded by aversive or appetitive stimuli and internal computation

[Beati et al., 2013], plus a student work of the AGREL connectionist categorization model here confronted to a realistic environment [Carrere and Alexandre, 2013].

Building one "brainy-bot" is a rather huge task and requires several high-level cognitive functionalities. However, though systemic neuroscience requires to study the system as a whole, it does not imply that each functionality has to be studied at the same level of details. Several blocks may be considered as black-boxes interacting with the part of the system to be extensively studied. This is the reason why the present platform is not limited to a survival environment, but comes also with middleware (presently in development) related to the basic cognitive functionality involved in such paradigms [Denoyelle et al., 2014]. Some modules will thus be implemented according to a rough description, e.g., via an algorithmic ersatz. The nervous sub-system under study on the reverse, is going to be implemented at a very fine scale (neural network mesoscopic models or even spiking neural networks).

The key features of this digital experimentation platform include the capability to perform experiments involving both long-term continuous time paradigms or short-term decision tasks with a few time-steps. It also allows us to consider either symbolic motor command or sensory input (e.g., ingest or not food, detect the presence of a stimulus) or quantitative gestures and complex trajectory generation (e.g., find resources in an unknown environment). A key point is to be able to mimic and repeat at will experiments performed in neuroscience laboratory on animals. Here, the obtained computational models are not only going to "fit the data" but to be explored far beyond, yielding the possibility to study long-term adaptation, statistical robustness, etc. Not only one instance of a bot can be checked, but several parallel experiments can be run in order to explore different parameter ranges, or compare alternative models.

Beyond these basic features, the input and output can be easily manipulated in order to enlarge the experimental setup. Up to now, the main aspect is "input or output degradation", i.e. adding noise. Originally, bot perception and action are performed without any added noise or random mistake. Depending on the experiment to be conducted, it is obvious (i.e. inserting a few lines of code between the platform and the bot), either to reduce variables precision range (e.g., add noise to the pixelic image) or to randomly draw the fact that a gesture may succeed or fail (e.g., introduce spurious command).

A step further, as already implemented as plugins, since all environment elements are available (not to the bot, but to the experiment software), it would be possible to design other cues, or more generally other interactions with the environment. However, in collaboration with neuroscience experimentalists, we have carefully selected what seems useful to explore biological systemic models, and avoided to provide a too general tool that does anything.

It would also be instructive to better understand to which extent such bio-inspired architectures actually required to control a biological system could enhance artificial control rules commonly applied in robotics or game engines. This is a challenging issue, beyond the present study, but an interesting perspective of the present work.

Acknowledgments Huge thanks to Nicolas Rougier for precious advice.

References

Beati et al., 2013. Beati, T., Carrere, M., and Alexandre, F. (2013). Which reinforcing signals in autonomous systems? In *Third International Symposium on Biology of Decision Making*, Paris, France.

Brette et al., 2007. Brette, R., Rudolph, M., Carnevale, T., Hines, M., Beeman, D., Bower, J. M., Diesmann, M., Morrison, A., Goodman, P. H., Harris, F. C., Zirpe, M., Natschläger, T., Pecevski, D., Ermentrout, B., Djurfeldt, M., Lansner, A., Rochel, O., Vieville, T., Muller, E., Davison, A. P., El Boustani, S., and Destexhe, A. (2007). Simulation of networks of spiking neurons: a review of tools and strategies. *Journal of computational neuroscience*, 23(3):349–398.

Carrere and Alexandre, 2013. Carrere, M. and Alexandre, F. (2013). Émergence de catégories par interaction entre systèmes d'apprentissage. In Preux, P. and Tommasi, M., editors, *Conférence Francophone sur l'Apprentissage Automatique (CAP)*, Lille, France.

Chemla et al., 2007. Chemla, S., Chavane, F., Vieville, T., and Kornprobst, P. (2007). Biophysical cortical column model for optical signal analysis. *BMC Neuroscience*, 8(Suppl 2):P140.

Cofer et al., 2010. Cofer, D., Cymbalyuk, G., Reid, J., Zhu, Y., Heitler, W. J., and Edwards, D. H. (2010). AnimatLab: a 3D graphics environment for neuromechanical simulations. *Journal of neuroscience methods*, 187(2):280–288.

Connolly and Grupen, 1993. Connolly, C. I. and Grupen, R. A. (1993). The applications of harmonic functions to robotic. *Journal of Robotic Systems*, 10(7):931–946.

Davison et al., 2008. Davison, A. P., Brüderle, D., Eppler, J., Kremkow, J., Muller, E., Pecevski, D., Perrinet, L., and Yger, P. (2008). PyNN: A Common Interface for Neuronal Network Simulators. *Frontiers in neuroinformatics*, 2.

Denoyelle et al., 2014. Denoyelle, N., Pouget, F., Viéville, T., and Alexandre, F. (2014). VirtualEnaction: A Platform for Systemic Neuroscience Simulation. In *International Congress on Neurotechnology, Electronics and Informatics*, Rome, Italy.

Eichenbaum et al., 2012. Eichenbaum, H., Sauvage, M., Fortin, N., Komorowski, R., and Lipton, P. (2012). Towards a functional organization of episodic memory in the medial temporal lobe. *Neurosci Biobehav Rev.*, 36(7):1597–1608.

Friston, 2012. Friston, K. (2012). A Free Energy Principle for Biological Systems. *Entropy*, 14(11):2100–2121.

Gorojosky and Alexandre, 2013. Gorojosky, R. and Alexandre, F. (2013). Models of Hippocampus for pavlovian learning. Rapport de recherche RR-8377, INRIA.

Herry et al., 2008. Herry, C., Ciocchi, S., Senn, V., Demmou, L., Muller, C., and Luthi, A. (2008). Switching on and off fear by distinct neuronal circuits. *Nature*, 454(7204):600–606.

Hyvärinen, 2009. Hyvärinen, A. (2009). Natural image statistics a probabilistic approach to early computational vision. Hardcover.

LeDoux, 2007. LeDoux, J. (2007). The amygdala. *Current Biology*, 17(20):R868–R874.

Rolls, 2010. Rolls, E. T. (2010). A computational theory of episodic memory formation in the hippocampus. *Behav Brain Res*, 215(2):180–96.

Taouali et al., 2011. Taouali, W., Viéville, T., Rougier, N. P., and Alexandre, F. (2011). No clock to rule them all. *Journal of physiology, Paris*, 105(1-3):83–90.

Teftef et al., 2013. Teftef, E., Escobar, M.-J., Astudillo, A., Carvajal, C., Cessac, B., Palacios, A., Viéville, T., and Alexandre, F. (2013). Modeling non-standard retinal in/out function using computer vision variational methods. Rapport de recherche RR-8217, INRIA.

Todorov, 2004. Todorov, E. (2004). Optimality principles in sensorimotor control. *Nature Neuroscience*, 7(9).

Uithol et al., 2012. Uithol, S., van Rooij, I., Bekkering, H., and Haselager, P. (2012). Hierarchies in action and motor control. *Journal of cognitive neuroscience*, 24(5):1077–1086.

Viéville and Crahay, 2004. Viéville, T. and Crahay, S. (2004). Using an Hebbian learning rule for multi-class SVM classifiers. *Journal of Computational Neuroscience*.

Viéville and Vadot, 2006. Viéville, T. and Vadot, C. (2006). An improved biologically plausible trajectory generator. Technical Report 4539-2, INRIA.