VirtualEnaction: A Platform for Systemic Neuroscience Simulation.

Nicolas Denoyelle, Florian Pouget, Thierry Viéville, Frédéric Alexandre

VirtualEnaction : A Platform for Systemic Neuroscience Simulation

Nicolas DENOYELLE[12], Florian POUGET[12], Thierry VIEVILLE[1] and Frédéric ALEXANDRE[123]

[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*

Frederic.Alexandre@inria.fr

Published at NeBICA 2014.

Keywords: Simulation, Computational Neuroscience, Virtual Reality.

Abstract: Considering the experimental study of systemic models of the brain as a whole (in contrast to models of one brain area or aspect), there is a real need for tools designed to realistically simulate these models and to experiment them. We explain here why a robotic setup is not necessarily the best choice, and what are the general requirements for such a bench-marking platform. A step further, we describe an effective solution, freely available on line and already in use to validate functional models of the brain. This solution is a digital platform where 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 somesthetic cues, complex survival behaviors can be experimented. The platform is also complemented with algorithmic high-level cognitive modules, making the job of building biologically plausible bots easier.

1 INTRODUCTION

The brain is a fascinating complex structure and designing global models of such a structure is particularly difficult. This is specifically true with regard to the fact that the brain is a complex system in interaction with the body and the environment. Two important consequences can be driven. On the one hand, modeling the brain 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. On the other hand, studying and validating functional models of brain structures at a macroscopic scale cannot be performed with restrained artificial static paradigms but requires experiments in complex environments, with realistic sensory-motor tasks to perform, including high-level interactive behaviors (e.g. survival strategy in the presence of prays/predators) and long-term protocols (since both statistical studies and bio-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, 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 the 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) be easy to design.

On the other hand, they are also intended to serve computer scientists as a way to design artificial autonomous systems, driven by the brain models. In this case, it is important that the supposed properties of the models (e.g., capacity to learn, robustness to noise or changing rules) be assessed by rigorous eval-

uation 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.

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 explain nothing.

As a consequence we also need a setup which can provide a "simplified environment", in order 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 to accept here 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 commands, 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 planification.

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 relates both representations.

3 SYSTEM DESCRIPTION

We consider that a "brainy-bot", i.e. the implementation of a global model of the brain functionality, interacts with its environment with the simple goal to survive. 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.

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

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., make the choice to stop 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 sizes, 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, which access to the block characteristics.

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 behav-

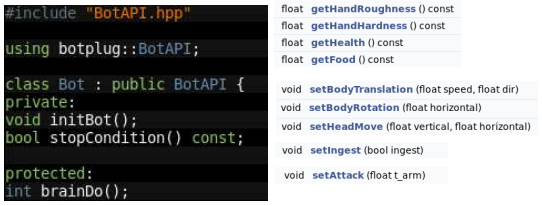

Figure 2: Left: The software interface is trivial: each implementation of a brainy-bot provides an initialization routine `initBot()` and a stop function `stopCondition()`, while at each time-step the `brainDo()` method is called. Right: All status, input and output functionality are available via a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

ior and interact with the digital experiment, a graphic user interface is available as described in Fig. 3.

More details on the computer implementation is given in Appendix B.

4 NEUROSCIENCE APPLICATION

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 (bio-mechanical simulation of a body) allowing investigate the relation between brain and body (Cofer et al., 2010). Furthermore, it proposes a neural network architecture for the implementation of cognitive function. 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 based 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 scale, 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 term 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 human. 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 explain 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 value. 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 usual model predictive verification is not only going to be possible, given a fixed data set, but for any data set obtained running experiment in the virtual environment. 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

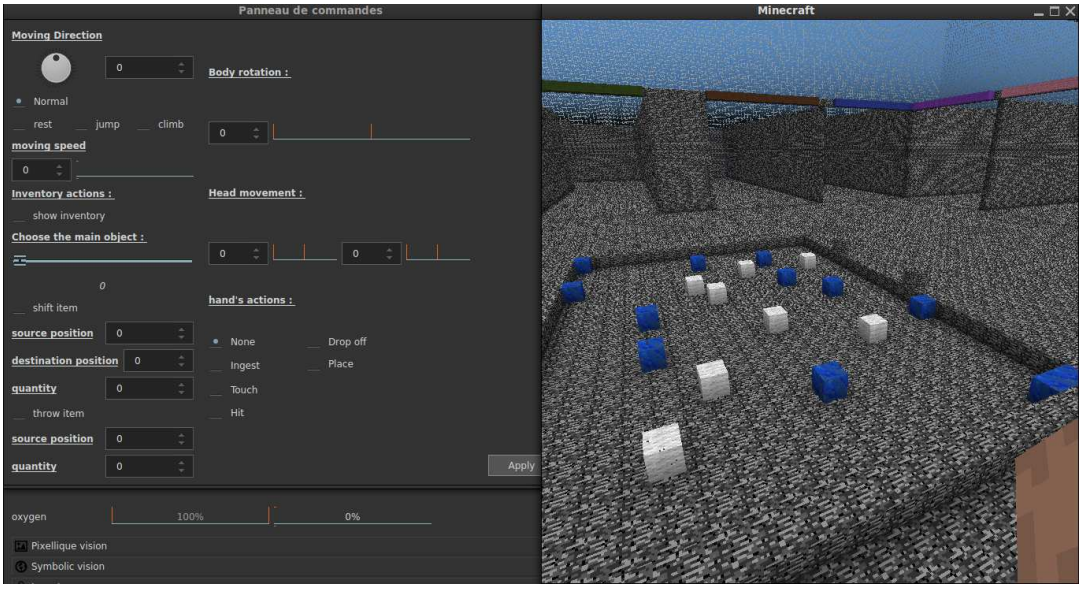

Figure 3: The software's graphic user interface. In order to observe and control the experiment, a view of the bot vision, status, input and output is available via interactive panels. It is also possible to "cheat" by controlling these variables values, in order to debug an experiment and understand in details what happens in such a complex system interaction.

the computational model to falsifiable conditions : Building on an animal model an experiment, training the animals for weeks, restarting from the beginning, if it appears that there is an unexpected trap (e.g., the task in 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 to perform realistic experiment with a computational model of the brain behavior, not only consider toy situations where the plausibility of the model can not be checked.

The degree of equivalence between simulation outcome and neuroscience experimental results is key issue. This is the reason why we have chosen a software platform where usual systemic neuroscience experiments can be reproduced "as a whole", as illustrated by several examples throughout this draft. Basis omissions (e.g., level of modeling detail) have been justified or rejected, in order to attain this objective of simulating classes of behavioral experiments where the subject is trained to realize a survival or rewarding task in a known or unexpected environment. Applications include Pavlovian (e.g., action of the amygdala) or operant conditioning, reinforcement learning, training and habituation, task oriented focus of attention, multi-sensory interaction. The main brain structures involved in such tasks are the basal ganglia system (including afferent and efferent structures) regarding selection of action, the different memory structures (e.g., episodic memory in the hippocampus, or working memory in the prefrontal cortex).

5 DISCUSSION

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 plug-ins, 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 do anything.

Building one "brainy-bot" is a rather huge task and requires several high-level cognitive functionality. 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 middle-ware (presently in development) related to the basic cognitive functionality involved in such paradigms, as listed in Appendix A. 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.

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.

As a conclusion, 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 connexionist categorization model here confronted to a realistic environment (Carrere and Alexandre, 2013).

ACKNOWLEDGMENTS This work was partly supported by the KEOpS ANR project. Huge thanks to Nicolas Rougier for precious advises and Maxime Carrere for his feedback. The NeBICA'14 review was a real chance to improve the original draft, thanks.

REFERENCES

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

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.

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, N. and Pouget, F. (2014). Virtualenaction, user documentation. Technical report, virtualenaction.gforge.inria.fr.

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

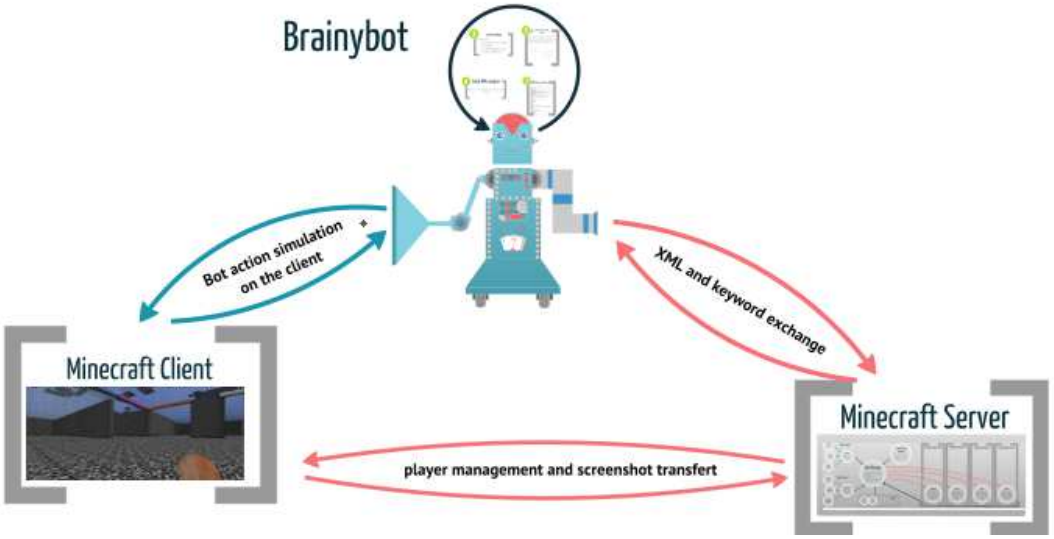

Figure 4: The platform's architecture. At the user level, only the notion of (i) game server where the environment and its mobile objects interactions are computed, (ii) game client where the game-play is rendered and (iii) brainy-bot where the brain model simulation is issued have to be taken into account.

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

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

Pouget, F. and Denoyelle, N. (2014). Virtualenaction, developer documentation. Technical report, virtualenaction.gforge.inria.fr.

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, 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.

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

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

A 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. In order to help representing the part of the system that may be simulated without biologically plausible models in a given context, a few sets of functionality are proposed. Let us briefly present the main algorithmic cues.

- *Episodic memory of input/output*: Any system has not only to take into account the present input and output in order to generate a proper behavior, but also store and consider the recent past information. In the brain, this function is mainly located in the hippocampus. In our context, thanks to the design choices, such information corresponds to a simple file of symbolic information, namely a hierarchical itemization of parameter values (i.e. a `XML` data structure). As a consequence, the memorization, transmission, compression (in the sense of eliminating negligible values or values related to older or smaller elements) is easy to well-define. It has also the consequence to propose a generic internal representation of the sensory-motor information at a symbolic level, without bothering about how this information is encoded on neural maps. It is an interesting perspective of the present work and an ongoing work to further investigate this issue, while basic algorithmic modules for such data management are already available.

- *Trajectory generation*: 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). Such issue may be a topic, or not (e.g., when studying the selection of action we may prefer not to bother with the planification and execution of such actions). In the latter case,

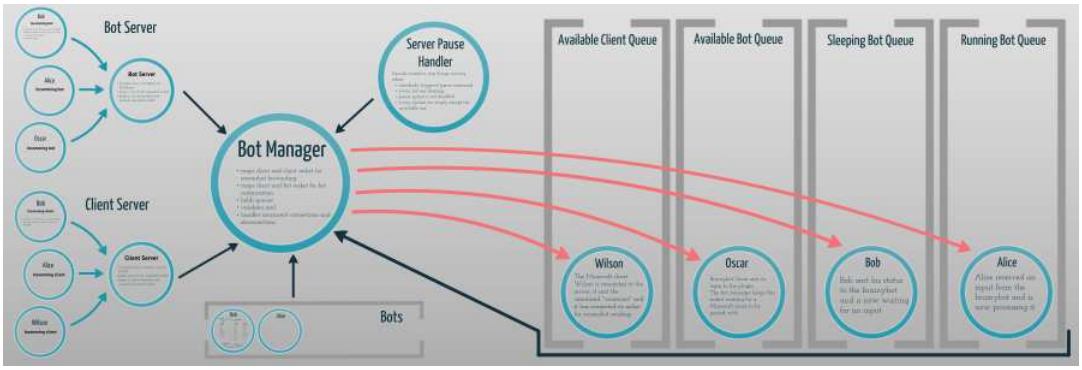

Figure 5: The platform underlying protocol. At the system level, the synchronization between the existing game events and the different system component is a non trivial middle-ware, taking into account both "real players" (thus playing in real time) and "bot players" (with slow reaction delays). The developed middle-ware can cope with several bots and humans (which will have to be patient, since the game runs at a reduced rate). Furthermore, the design allows the platform to be distributed on a cluster (server, client and bots running on different machines).

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).
- *Generic categorization*: Another generic key cognitive feature is the capability to "extract" symbolic information from a bundle of quantitative or qualitative values. This 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). Though such issue is indeed a topic on its own, we may wish in some context to have it available as a black box. A biologically plausible support vector machine mechanism is available to this end, with versatile uses (Viéville and Crahay, 2004). Let us also mention that informing the bot about its absolute position and orientation or about symbolic information of the scene is a way to shortcut its sensory modules and provide integrated cognitive information.

B COMPUTER IMPLEMENTATION

This experimental `virtualenaction` setup is fully documented both at the user (Denoyelle and Pouget, 2014) and developer (Pouget and Denoyelle, 2014) levels. The user view of the computer implementation is oversimple as illustrated in Fig. 4, though the underlying computer development is not that trivial as sketched out in Fig. 5. It runs a `bukkit` server in the 1.4.5 version and is designed as a set of plugin for this server, with minimal changes in the original code. This design choice was optimal in terms of middle-ware development effort and final code stability. At the server level, two plugins have been designed. The `VirtualEnaction` plugin allows us to replace usual player mouse and keyboard interactions with the game-play by a programmable API. The `Characteristics` plugin allows us to implement blocks and entities additional characteristics such as temperature, hardness or smell.

The minecraft server and client have been modified in order to be able to run in this programmatic mode, and to get a full access to all game gauges, because we not only must run the bot but also observe in details its behavior and measure all interactions with the environment. As a consequence, the actual source code corresponds to a fork of a frozen version of the minecraft game, while everything is documented in order to easily migrate to some more recent version.

Another key point is the modification of the environment. Using the standard game, it is possible to build an environment, include other entities, and so on. This is simply realized via the game interface. Futhermore, scripted scenarios can also be introduced (e.g., some resource appears if and only if a block is put in a given position), as additional plugins. In addition, configuration files on the server side, allows us to act on available resources or survival gauges, and build versatile survival situations. It is thus a fully editable setup. As being an open-source collaborative software, it is available on the `Inria forge` with more than 15 co-developers or project followers.

A step further, in order this virtual experimentation platform to be extensible in the perspective of new setups, the `botplug` that allows the bot to be plugged in the client/server system is also designed with a notion of plugin. For instance, a tactile sense (not described here), and a variant of the pixelic vision to be connected to different image sequence computation libraries are made available as plugins.

It is clear that several extensions are going to be developed in a near future thanks to this modular architecture choice.