CABot3: A Simulated Neural Games Agent

Christian Huyck, Roman Belavkin, Fawad Jamshed, Kailash Nadh, Peter Passmore,
Middlesex University
c.huyck@mdx.ac.uk
Emma Byrne
University College London
and **Dan Diaper**
DDD Systems

Abstract

CABot3, the third Cell Assembly roBot, is an agent implemented entirely in simulated neurons. It is situated in a virtual 3D environment and responds to commands from a user in that environment. It parses the user's natural language commands to set goals, uses those goals to drive its planning system, views the environment, moves through it, and learns a spatial cognitive map of it. Some systems (e.g. parsing) perform perfectly, but others (e.g. planning) are not always successful. So, CABot3 acts as a proof of concept, showing a simulated neural agent can function in a 3D environment.

1 Introduction

CABot3, the third Cell Assembly roBot, is a video game agent implemented entirely in simulated neurons. It assists a user in the game: viewing the 3D environment; processing natural language commands; making simple plans; and moving through, modifying, and learning about the environment.

As its name suggests, CABot3 makes extensive use of Cell Assemblies (CAs), reverberating circuits of neurons that are the basis of short and long-term memories [Hebb, 1949]. CABot3 represents symbolic knowledge in a neural network by CAs. Simple rules are implemented by simple state transitions, with a particular set of active CAs leading to the activation of a new set of CAs, and complex rules are implemented by variable binding combined with state transitions.

CABot3 is a virtual robot that creates and uses plans with a neural implementation of a Maes net [Maes, 1989], while natural language parsing is based around a standard linguistic theory [Jackendoff, 2002]. All agent calculations are done with Fatiguing Leaky Integrate and Fire (FLIF) neurons (see Section 2.1) and some of the network structure can be related to brain areas (see Section 4.2). The agent learns a spatial cognitive map of the rooms in the video game.

Two components of the CABots have been evaluated as cognitive models. The Natural Language Parser [Huyck, 2009] parses in human-like times, creates compositional semantic structures, and uses semantics to resolve prepositional phrase attachment ambiguities. It also learned the meaning of the verb *centre* from environmental feedback, closely related to a probability matching task [Belavkin and Huyck, 2010].

2 The Structure of CABot3

Due to space constraints, a complete description of CABot3 is not possible, though an almost complete description of an earlier version, CABot1, is available [Huyck and Byrne, 2009], and the code is available on http://www.cwa.mdx.ac.uk/cabot/cabot3/CABot3.html. A brief description of the neural model is described next, followed by a description of the subnetworks used, and a brief description of how those subnetworks are connected to generate CABot3's functionality.

2.1 FLIF Neurons

FLIF neurons are a modification of the relatively commonly used LIF model [Amit, 1989]. When a neuron has sufficient activation, it fires, and sends activation to neurons to which it is connected proportional to the weight w_{ji} of the synapse from the firing pre-synaptic neuron j to the post-synaptic neuron i. That weight can be negative. The simulations use discrete cycles, so the activation that is sent from a neuron that fires in a cycle is not collected by the post-synaptic neuron until the next cycle. If a neuron fires, it loses all its activation, but if it does not fire, it retains some, while some activation leaks away (decay); this is the leaky component and is modelled by a factor $D > 1$, where the activation is divided by D to get the initial activation at the next step. In CABot3, activation of neuron i at time t, A_{i_t} is defined by Equation 1. V_i is the set of all neurons that fired at $t-1$ connected to i.

$$A_{i_t} = \frac{A_{i_{t-1}}}{D} + \sum_{j \in V_i} w_{ji} \quad (1)$$

Additionally, FLIF neurons fatigue. Each cycle they fire the fatigue level is increased by a constant, but when they do not fire, the fatigue level is reduced by another constant, but never below 0. The neuron fires at time t if its activity A minus fatigue F is greater than the threshold, see Equation 2.

$$A_{i_t} - F_{i_t} \geq \theta \quad (2)$$

FLIF neurons are a relatively faithful model of neurons, though are relatively simple compared to compartmental models [Hodgkin and Huxley, 1952]. If each cycle is consider to take ten ms., it has been shown that 90% of the spikes emitted fall within one cycle of the spikes of real neurons on the same input [Huyck, 2011]. Aside from their biological

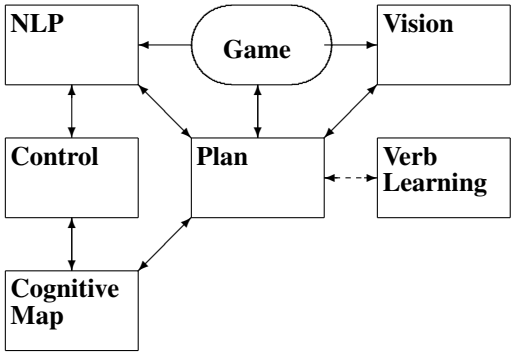

Figure 1: Gross Topology of CABot3. Boxes represent subystems of subnets. The oval represents the environment.

fidelity, another benefit is that 100,000 FLIF neurons with a 10ms cycle can be simulated in real time on a standard PC.

Neurons are grouped into CAs, either manually by the developer, or through emergent connectivity. A given neuron may be part of one or more CAs.

2.2 SubNetworks

The FLIF neurons in CABot3 are grouped into 36 subnetworks. Each subnet is an array of neurons, and each may have different FLIF parameters and learning parameters, including no learning. In CABot3, connectivity within a subnet is always sparse, but it varies between subnets; this connectivity may have some degree of randomness, but in some cases it is tightly specified by the developer to guarantee particular behaviour. Subnets may also be connected to each other with neurons from one sending synapses to others; these types of connections vary similarly. These reflect differences, possibly caused in part by genetics, between different types of biological neuron.

Apart from biological fidelity, another advantage of subnets is that they facilitate software engineering. Tasks can be partitioned, with one developer working on one net or a set of nets for a particular subsystem. Communication with other subsystems may take place via only one subnet allowing a degree of modularity[1].

2.3 Gross Topology

CABot3 can be divided into a series of subsystems each consisting of subnets (Figure 1. Arrows show directed connections from one subsystem to another, each, aside from the game, representing a large number of synapses. Verb learning is not tested in CABot3, thus the connection is represented with a dotted line and is omitted in later diagrams. Also, for clarity in later diagrams, due to the prevalence of connections from control, connections from the control subsystems to other subsystems are omitted.

The basic subsystems are described below. Section 3.1 describes the game and the control subsystem; the game receives simple commands from the agent. Section 3.2 describes the vision subsystem; 3.3 the planning subsystem, 3.4 the natural language processing (NLP) subsystem, 3.5 verb learning, and Section 3.6 describes the spatial cognitive map learning subsystem. Connections between the subsystems are also described in these sections. Section 4 summarizes the evaluation of CABot3.

3 Subsystems

Each subsystem is explained below, concentrating on those that have not been explained elsewhere.

3.1 Communication, Control and the Game

The game was developed using the Crystal Space [Crystal Space, 2008] games engine. It is a black and white 3D environment with an agent, a user, four rooms connected by four corridors, and a unique object in each room (see Figure 4); the objects were vertically or horizontally striped pyramids or stalactites (down facing pyramids). The agent and user can move around the rooms independently. The game provides the input to the vision system using a dynamically updated picture of the game from the agent's perspective. The user issues text commands as input to the NLP system. The game also has a bump sensor, and this ignites a CA in the fact subnet in the planning system (see Section 3.3) when the agent bumps into a wall. Similarly, the game takes commands from the agent's planning system to turn left or right, or move forward or backward.

The control subsystem consists of one subnet, the control subnet, which in turn consists of five orthogonal CAs[2]. These CAs mark the state of the agent, either parsing or clearing a parse, setting a goal or clearing it, or a stub. The initial state is turned on at agent start up, and one state is always on.

In the first state, the system is waiting for input or parsing a sentence. This state has connections to most of the NLP subnets to facilitate the spread of activation. When the last grammar rule ignites, it forces the control state to move on.

Most of the CAs involved in parsing, and planning, and all of the control CAs are orthogonal oscillators. When active, they oscillate from having one half of the neurons firing to having the other half firing, then back to the first set. This allows the CA to avoid fatigue as its neurons only fire half the time. This is not biologically accurate, but enables precise behaviour with relatively few neurons.

When it has finished parsing, control moves to the clear parse state. This changes the instance counters in the NLP subsystem preparing it for the next sentence. After a few steps, activation accumulates in the set goal state causing it to ignite, and suppress the clear parse state.

In the third state, the goal in the planning system is set from the semantics of the parse via the intermediate goal set subnet. In the fourth state, information is cleared from the NLP system after the goal is met, and the fifth is a stub.

The control allows the system to parse while still processing a goal. Vision remains active at all times.

[1] Note this modularity may conflict with actual brain topology.

[2] A neuron in an orthogonal CA belongs to that and only that CA.

3.2 Vision

The visual system of CABot3 consists of six subnets: visual input, retina, V1, gratings, V1Lines, and object recognition. The retina, V1, gratings, and V1Lines share some similarities with their human counterparts, but are much simplified models. Higher-level object recognition in CABot3 is not biologically plausible and does not mimic known mechanisms in the human visual system. It does however carry out two important functions of the visual system: the simultaneous identification of *what* is seen and *where* it is in the visual field.

The visual input, retina, V1 and object recognition nets have been described elsewhere and are only slightly modified [Huyck *et al.*, 2006]. The most important modification is the addition of *grating cells* that mimic known properties of the primate visual system, in that they respond selectively to textures of a certain orientation and frequency [DeValois *et al.*, 1979].

The visual input subnet is a 50x50 network of FLIF neurons that do not fatigue. Input to this subnet is clamped to the external stimulus, thus activation is constant until the agent's point of view changes. Each neuron in the 50x50 subnet corresponds to an identically located "cell" in a 50x50 grid of light levels from the environment.

The CABot1 retina subnet contains six 50x50 grids of FLIF neurons. Each subnet contains retinotopic receptive fields of a single size and polarity: 3x3 receptive fields with single-cell centre; 6x6 receptive fields with a 2x2 cell centre and the 9x9 receptive fields with a 3x3 cell centre. For each of these sizes there is a subnet with an on-centre/off-surround polarity (neurons fire when the centre of the receptive field is stimulated and the surround is not) and an off-centre/on surround polarity.

In the V1 area of the human visual system there are neurons, known as simple cells, that are tuned to specific edge and angle orientations. These simple cells are location specific. In the CABot3 V1 and V1Lines subnets, FLIF neurons have been connected to replicate this behaviour. V1 and V1Lines were split for engineering convenience. Weighted connections feed activation from on-centre and off-centre cells in the retina subnet. There are eight orientation specific edge detectors and four angle detectors.

The edge detectors in V1Lines also have recurrent connections to grating detector subnets. Grating detector cells identify repeated patterns of edges of a given orientation and frequency. These grating detectors allow CABot3 to recognise textures in the environment. This allows CABot3 to distinguish between objects of the same shape but that are 'painted' with different textures.

The object recognition net is the least biologically plausible of the visual subnets. There are five modules in the subnet, made up of a number of overlapping cell assemblies. These specialise to recognise pyramids, stalactites, door jambs, doors, or unknown objects. The same modules also carry the "where" (position) as each subnet is a retinotopic representation of the visual field.

3.3 Planning

The planning system is basically a Maes net [Maes, 1989]. The gross topology is shown in Figure 2. All subsystems link

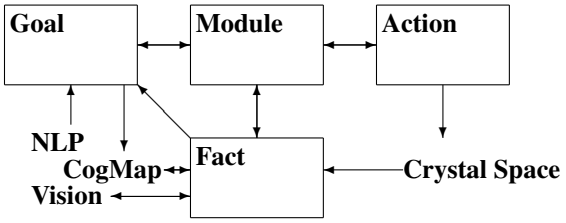

Figure 2: Gross Topology of the Planning Subsystem. Boxes represent subnets.

to the planning subsystem. Its primary entry point is from the NLP subsystem, which sets the goal. The primary outcome is to the game; the CAs in the action subnet are polled and a symbolic command is emitted to the game.

This subnet structure was used throughout CABot1, 2 and 3, and a simple example is the command, *Move forward.* When parsing is completed, the control subnet in combination with the NLP subnets cause an orthogonal oscillating CA in the goal net to ignite. This is equivalent to a goal being set in the Maes net. With a simple action, this goal CA causes the corresponding module subnet CA to ignite, which in turn causes the corresponding CA in the action subnet to ignite. The action CA is then polled to emit the command to the game. Backward inhibition extinguishes the goal and module CAs, and accumulated fatigue causes the action CA to stop.

Simple movements do not require any facts, but actions are often predicated on facts that are set by the environment. For example, an environmentally sensitive command is *Turn toward the pyramid.* In this case, the vision system ignites a fact CA expressing the target's location in the visual field, for instance, "target on left". The combination of activity from the fact net and the goal net cause the appropriate module CA to ignite, which in turn causes the appropriate action CA to ignite. This is an example of needing two (or more) CAs ignited to ignite a third. This is done by allowing the activation of the neurons in the third CA to rise, but which is below threshold when one CA is ignited. The second CA then provides enough activation to ignite the third CA.

Note that the full Maes net has a concept of Maes module activation. In CABot3, the module CAs are either on or off, and there is no activation level (but see Sections 3.4 and 5).

The system executes 21 commands, four primitives (e.g. *Turn right*), two compounds (e.g. *Move left* which executes a left then forward), turn toward pyramid or stalactite, go to seven objects, explore, stop, and move before four objects. The seven objects are door, and pyramid or stalactite either (vertically) barred, (horizontally) striped, or unspecified.

Moving to an object may require several steps. CABot3 centres the object in the visual field and then moves to it until the object fills the visual field, possibly centring again along the way. Any command can be stopped by the *Stop* command.

The most sophisticated thing the system does, in response to the *Explore* command, is to explore the four rooms and memorize the objects in the room (see Section 3.6). To test that the system has correctly memorized the map, a command

such as *Move before the striped pyramid* may be used. The system then moves to the room before the striped pyramid and stops without having seen it again, showing it has memorized its location (see Section 4.1).

In all, the goal subnet contains 26 CAs, including subgoals. The fact subnet has 66 CAs, the module subnet seven, and the action subnet six including two error conditions.

3.4 Natural Language Processing

The stackless parser has been described elsewhere [Huyck, 2009]. Input is provided symbolically from Crystal Space, each word is associated with an orthogonal set of neurons in the input net, and they are clamped on when the particular word is being processed.

The subnets involved follow Jackendoff's Tripartite theory, with NLP broken into three main systems, lexicon, syntax and semantics, and the systems communicate via subsystems.

Stackless parsing is done by activation levels, with the number of neurons in a CA firing in a cycle reflecting CA activity. In practice this is done by a tightly specified topology that has the number of neurons firing in the CA decaying over time; activation levels reflect the order of items.

Semantics are handled by overlapping encoding derived from WordNet. This could be useful in resolving parsing ambiguities, though this is not implemented in CABot3.

Grammar rule CAs are selected by activation of component (lexical or higher order category) CAs. Variable binding is done with short-term potentiation [Hempel *et al.*, 2000], and this is how instances store their semantics. Noun instances represent noun phrases and verb instances, verb phrases including their arguments. A case frame is generated for each parse, and the slots are bound to other instances or to the semantics of words. These bindings are learned but decay over time. The next time they are used, two parses later, the instance frames have been erased by automatic weight decay.

3.5 Motivation and Reinforcement Learning

Hebbian learning strengthens the connections between CAs as well as within a CA. CAs are associated with some atomic propositions, and more complex propositions (such as implication rules) are represented by groups (e.g. pairs) of associated CAs. However, Hebbian rules do not differentiate between learning 'good' or 'bad' propositions. After several atomic propositions or symbols have been learnt in the form of corresponding CAs, the main problem is to learn the correct or favourable propositions from these.

This problem was solved by a motivational system that is used to control Hebbian learning so that propositions with higher utility values or rewards are reinforced [Belavkin and Huyck, 2008]. The mechanism uses two specialised subnets: utility and explore. Neurons in the utility network output signals corresponding to a reward or payoff obtained from the environment. Neurons in the explore network output signals that represent random noise and they can be connected to any set of CAs that needs to be randomised to allow stochastic exploration of their interrelations. The utility network has inhibitory connections to the explore network so that high values of utility correspond to low level of randomness at the output of the explore network.

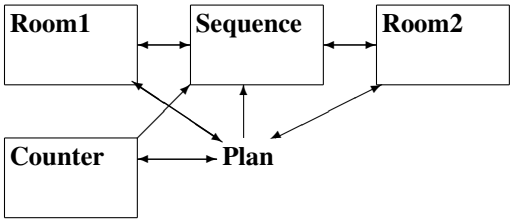

Figure 3: Subnets involved in spatial cognitive mapping.

It has been demonstrated previously that the mechanism described above can be used to learn simple sets of rules in a CA-based architecture [Belavkin and Huyck, 2008], and that it can be used to model probability matching observed in animals and people [Belavkin and Huyck, 2010]. The mechanism was used by CABot2 to learn the verb *centre* and the corresponding action associated with a visual stimulus. It is unplugged in the currently available version of CABot3.

3.6 Spatial Cognitive Map Learning

Spatial cognitive mapping is the psychological process of recording, recollecting and acting on locations and objects in a physical environment [Downs and Stea, 1973]. CABot3 implements a simple version of this complex process based on the authors' previous work [Huyck and Nadh, 2009]; the CABot3 agent explores the rooms, learns the objects, associations between them, and navigates to specific rooms.

Figure 3 shows the subnets involved. Room1 and room2 encode adjacent rooms that the agent moves through, where room1 is the prior room and room2 is the current room. The sequence net encodes the associations between the rooms, and the objects in them. The counter net supports the order.

On receiving the *Explore* command, the agent goes around the environment, room by room, learning the objects it sees. When an object is in its visual field, for instance a *striped pyramid*, the current room in association with it is encoded as a CA in Room1. The object in view is recognised from activity in the fact net, and learning lasts 200 cycles as it has been observed to be the minimum number of cycles required for CAs to be learnt. When the agent moves to the next room, the same routine happens, but as it has come from an adjacent room, the current room is also encoded in room2. The previous room CA in room1 is still active, the current room CA in room2 ignites, and the association between the two rooms learnt as a CA in the sequence net. Learning in the sequence subnet happens via co-activation with the two active room CAs in the two room nets lasting 200 cycles. This in essence creates individual CAs representing the rooms and their constituent objects in the two room nets, and the association between the rooms the agent passes through in sequence. Counter keeps track of the room the agent is currently in. When the agent is done exploring, room1 and room2 have a CA associated with the item in the fact net, and the sequence net has five CAs representing the association between each room and its adjacent room.

After exploration, when the agent is issued with a com-

mand such as *Move before the striped pyramid*, the involved fact such as "striped pyramid" ignites in fact (Figure 2). Fact in turn ignites the learnt CA in room2 representing the room with the "striped pyramid". As the sequence net has encoded the association between rooms, the active CA in room2 activates the associated room in room1, which is the room before the room in room2 that the agent entered through while exploring. Thus the agent *deduces* the target room from its simple learnt cognitive map. With the target room active, the agent starts moving, and when it reaches the target room, activity in the goal subnet informs it of task completion.

4 Evaluation

The evaluation of a CABot3 agent is a complex process. Many of the components have been evaluated separately. For the purposes of testing CABot3 itself, parsing, for example, consists of a few dozen grammar rules that it uses to parse all of the acceptable commands correctly, so as to set an appropriate goal. In parsing, all of the connections are deterministic, and the parsing subnets are insulated by layers of connections from the more stochastic areas.

The evaluation of the planning system and cognitive mapping systems are briefly described in Section 4.1. The control system is a simple finite state automata which switches states when other systems reach certain states, for example when the parser finishes, the control state changes. This system largely switches states when appropriate, but occasional errors do occur, but these are largely self correcting. However, it occasionally gets into states from which it cannot recover.

The vision system works robustly for a limited range of textures. There are two orientations and a limited range of spatial frequencies that the grating cells can accommodate due to the size and resolution of the retinal nets. Within these limitations, however, the system identifies textures reliably. Where objects are presented clearly on the retina (that is, where the viewing angles are not extreme) the visual system robustly identifies the objects in the 3D world.

4.1 Explore Evaluation

The planning system is responsible for a relatively wide range of activities. Most of these it performs entirely correctly; for example the command Turn left. always works correctly. The most sophisticated physical task the agent performs is to explore all of the rooms, making use of vision and spatial cognitive mapping (see Section 3.6). This exploration is relatively simple though it can take several hundred moves. An example is shown in Figure 4.

CABot3 initially tries to identify the room it is in by the unique object it sees. In the case of Figure 4, it sees the striped pyramid, and this is put into its spatial cognitive map. It then finds the corridor, which it can see at a distance. It moves to the front of the corridor keeping to the left edge, stopping when it bumps into the edge of the corridor. It then turns right and moves through the corridor along the edge. At the end of the corridor it turns right to see the object in the next room. It can see there is an object but the agent is not close enough to identify it. It moves toward the object, in this case the barred pyramid, until it can identify it. It then puts that in the cognitive map, and repeats the process for the next two rooms, stopping when it identifies the object in the initial room.

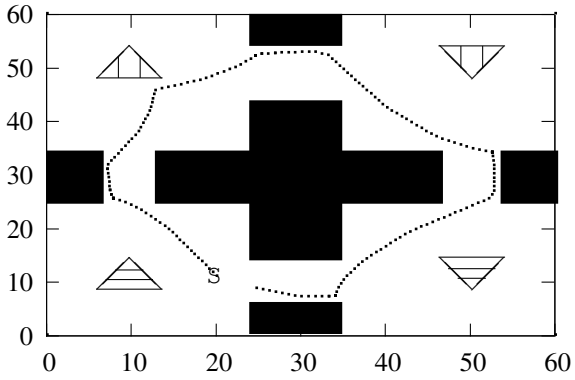

Figure 4: Forward moves of CABot3 while exploring the rooms, starting at **S** with moves marked by dots.

Explore works about half the time. It appears cognitive mapping works each time, and all of the failures are due to navigation problems.

4.2 Subnet Evaluation

The subnet topology is important both for software engineering and for relating to brain areas. From the software engineering perspective, the method has been successful. Breaking the full network into subnets has enabled development of systems to be partitioned with one developer working on one task, (e.g. vision) in isolation. The systems have then been combined to work together in the full CABot3 agent.

The brain did not evolve this way, so it is also important to see how different subnets might map to brain areas. There is a strong correlation between CABot3's early vision areas and biological vision areas, with both accounting for similar behaviour. There is a looser correlation between the explore subnet in reinforcement learning and the basal ganglia. However, in most cases the subnets have little correlation with brain areas. None the less, the basic subnet topology could be used to closely mimic known brain area topology and behaviour. As subnets still have connections to and from other subnets, so CABot3 is one large network.

5 Conclusion

Many researchers thought that implementing AI systems with simulated neurons was too complex (e.g. [Smolensky, 1988]). Perhaps this was true a few decades ago, but the authors believe that CABot3 shows that this fear has passed.

The mere implementation of a relatively simple agent may miss the point that many connectionists hope to make: that the neural level is not the correct level to study the brain. While the authors would agree that many complex behaviours, such as attractor dynamics and supervised learning, are being effectively studied with non-neural connectionist systems, this does not mean that the same problems cannot be effectively studied in neural systems.

Moreover, simulated neural systems have an important advantage over connectionist systems when it comes to study-

ing AI: existing intelligent agents (humans and other animals) use neurons to think, and the neural and cognitive behaviour of these animals is being studied. Simulated neural systems, which match sensible intermediate behaviour, can be developed as milestones on the way to full fledged AI systems.

During the project, it was shown that in general a network of CAs, and in particular a network of FLIF neuron CAs, was Turing complete [Byrne and Huyck, 2010]. In some sense, this makes the implementation of CABot3 unsurprising. While CABot3 is obviously not a neuron by neuron simulation of a human brain, it does have a series of links to neurobiological and cognitive behaviour that increase its validity. The base neural model is a relatively accurate if simplified model of neurons. In CABot3, some subnets are reasonable approximations of brain areas. The use of CAs for long and short-term memories and as the basis of symbols is neuropsychologically supported, and provides a bridge between subsymbolic and symbolic processing. Cognitive models provide solid links to psychological behaviour from a neural system.

While it is possible to continue to program new and improved neural systems, the authors believe the key is to have the system learn its behaviour. Thus, a vast range of future work is possible such as: improving existing systems; adding new sensory modalities, for example sound detection and speech recognition; moving from virtual to physical robots; improving the fit with biological data, for example more neurons, more realistic topologies, and more accurate neural models; new and more sophisticated cognitive models; and improving computation, for example by use of specialised neural hardware. Simulated CAs themselves could also be improved so that a single CA could be learned, and persist for an appropriate duration. More radical improvements also present themselves including improved learning, for example at the CA level and in combination with variable binding, improved understanding of dual attractor dynamics, integration of attention, and experiments with agents that continue to improve over several days or longer.

CABot3 is an agent in an environment functioning in real time, implemented in simulated neurons. It is a solid step in the development of agents implemented in simulated neurons, and it is intended that more sophisticated agents will be derived from it. Building systems like this will involve trade offs between biological and psychological fidelity, and computational constraints. By building more biologically and psychologically plausible systems that perform more tasks, significant advancements in the understanding of general cognition can be made.

Acknowledgements:
This work was supported by EPSRC grant EP/D059720.

References

[Amit, 1989] D. Amit. *Modelling Brain Function: The world of attractor neural networks*. Cambridge University Press, 1989.

[Belavkin and Huyck, 2008] R. Belavkin and C. Huyck. Emergence of rules in cell assemblies of fLIF neurons. In *The 18th European Conference on Artificial Intelligence*, 2008.

[Belavkin and Huyck, 2010] R. Belavkin and C. Huyck. Conflict resolution and learning probability matching in a neural cell-assembly architecture. *Cognitive Systems Research*, 12:93–101, 2010.

[Byrne and Huyck, 2010] E. Byrne and C. Huyck. Processing with cell assemblies. *Neurocomputing*, 74:76–83, 2010.

[Crystal Space, 2008] Crystal Space. $http : //www.crystalspace3d.org/main/main\_page$. 2008.

[DeValois *et al.*, 1979] K. DeValois, R. DeValois, and E. Yund. Responses of striate cortex cells to grating and checkerboard patterns. *The Journal of Physiology*, 291(1):483–505, 1979.

[Downs and Stea, 1973] Roger M. Downs and David Stea. *Cognitive maps and spatial behaviour. Process and products*, pages 8–26. Aldine, Chicago, 1973.

[Hebb, 1949] D. O. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. J. Wiley & Sons, 1949.

[Hempel *et al.*, 2000] C. Hempel, K. Hartman, X. Wang, G. Turrigiano, and S. Nelson. Multiple forms of short-term plasticity at excitatory in rat medial prefrontal cortex. *Journal of Neurophysiology*, 83:3031–3041, 2000.

[Hodgkin and Huxley, 1952] A. Hodgkin and A. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, 117:500–544, 1952.

[Huyck and Byrne, 2009] C. Huyck and E. Byrne. CABot1: Technical report. Technical report, Middlesex University, 2009.

[Huyck and Nadh, 2009] C. Huyck and K. Nadh. Multi-associative memory in fLIF cell assemblies. In R. Cooper (Eds.) A. Howes, D. Peebles, editor, *9th International Conference on Cognitive Modeling - ICCM2009*, pages 81–87, Manchester, UK, 2009.

[Huyck *et al.*, 2006] C. Huyck, D. Diaper, R. Belavkin, and I. Kenny. Vision in an agent based on fatiguing leaky integrate and fire neurons. In *Proceedings of the FifthInternational Conference on Cybernetic Intelligent Systems*, 2006.

[Huyck, 2009] C. Huyck. A psycholinguistic model of natural language parsing implemented in simulated neurons. *Cognitive Neurodynamics*, 3(4):316–330, 2009.

[Huyck, 2011] C. Huyck. Parameter values for FLIF neurons. In *Complexity, Informatics and Cybernetics: IMCIC 2011*, 2011.

[Jackendoff, 2002] R. Jackendoff. *Foundations of Language: Brain, Meaning, Grammar, Evolution*. Oxford University Press, 2002.

[Maes, 1989] P. Maes. How to do the right thing. *Connection Science*, 1:3:291–323, 1989.

[Smolensky, 1988] P. Smolensky. On the proper treatment of connectionism. *Behavioral and Brain Sciences*, 11:1:1–22, 1988.