The Hi-NOON Neural Simulator and its Applications to Animal, Animat and Humanoid Studies

R.L.B. French*, R.I. Damper* and T.W. Scutt†

*Image, Speech and Intelligent Systems Research Group,
Department of Electronics and Computer Science,
University of Southampton,
Southampton SO17 1BJ, UK.

†Core Design Ltd.,
2 Roundhouse Road,
Pride Park,
Derby DE24 8JE, UK.

(emails: `rid@ecs.soton.ac.uk`, `rlbf98r@ecs.soton.ac.uk`
and `TomS@Core-Design.com`)

Abstract. This paper describes the Hi-NOON neural simulator, originally conceived as a general-purpose, object-oriented software system for the simulation of small systems of biological neurons, as an aid to the study of links between neurophysiology and behaviour in lower animals. As such, the artificial neurons employed are spiking in nature: to effect an appropriate compromise between computational complexity and biological realism, modeling was at the transmembrane potential level of abstraction. Further, since real neural systems incorporate different types of neurons specialized to somewhat different functions, the software was written to accommodate a non-homogeneous population of neurons. Hi-NOON has been used in animat (cricket phonotaxis) and biologically-based robot studies. In particular, it was employed to implement the nervous system of our ARBIB robot. A simple model of synaptogenesis has been added so improving the stability of its learning in the light of the stability-plasticity dilemma, and as a mechanism for long-term memory. The efficacy of the simulator is illustrated with respect to some recent applications to situated systems studies. Now that Hi-NOON has been expanded to simulate large nervous systems in a concurrent environment, it can be applied to humanoid robotics in the future.

1 Introduction

When considering a complex system, understanding is dependent on methods for conveniently representing and structuring its component parts. This is especially true at the design stage of a new project, where a high-level object-oriented approach to programming has often proved useful in building complex software applications. In much the same way as high-level development of software allows delegation of particular tasks to specialized program structures, a high-level approach to the representation of artificial nervous system function allows delegation of tasks among specialized neural structures. We regard these structures as a hierarchy: from synapses to circuits through incremental subsystems [4] to a complete nervous system.

This paper describes a program originally designed to simulate small systems of neurons within the computational neuroscience paradigm, its more recent development and applications, and its future role in the development of a humanoid nervous system. The program is called Hi-NOON, which stands for **Hi**erarchical **N**etwork of **O**bject-**O**riented **N**eurons. As the name suggests, synapses, neurons and networks are in principle represented as objects within an object-oriented hierarchy [26, 28] at various levels of abstraction. The lowest such level uses the membrane potential (strictly, transmembrane potential difference) as the observable parameter in the network model. This is a much lower-level approach than the use of activation values roughly corresponding to the spike or action potential rate of individual neurons or collections of neurons as in popular parallel distributed processing (PDP) models. By contrast, Hi-NOON retains details of individual spike generation which is lost in the traditional connectionist approach. Also, Hi-NOON facilitates simulation of a non-homogeneous population of neurons. In principle, this allows different, higher levels of abstraction to be used in a 'mixed mode'. Most obviously, PDP-type neurons modeled at the level of activation could be mixed with more biologically-realistic spiking neurons, in which spike generation is stochastic. Thus, although one would be exercising only a proportion of its flexibility and power, one could even use Hi-NOON as the simulator for a highly conventional PDP-type artificial neural network.

We concentrate in this paper on the less usual simulation of spiking behaviour. One might reasonably ask what advantages this might offer, i.e. what can a simulation based on spiking neurons achieve that cannot be done using more gross PDP-type model neurons? This is currently a vexed question in computational neuroscience, and a fully definitive answer cannot be given at this stage. Clearly, detailed timing information for individual spikes, and relative timing between spikes, offers an additional dimension to the neural code, as does the stochastic aspect. There is suggestive evidence that this sort of information is indeed important in biology. Citing Rieke et al. [25, p. 279]:

> "... under many conditions, behavioral decisions are made with of order one spike per cell, ... individual spikes can convey several bits of information about incoming sensory stimuli ... precise discriminations could ... be based on the occurrence of individual spikes ..."

The remainder of this paper is structured as follows. Next, in Section 2, we consider the structure and function of the Hi-NOON simulator. Then, in Section 3, we present a more detailed description of neurons and synapses within its neural model. In Section 4, we consider clues that biology has given us for implementing a simplified mechanism of long-term memory. In Section 5, we describe how Hi-NOON has been used to design and implement the 'nervous systems' of two rather different situated systems: a model of phonotaxis in the cricket and a model incorporating synaptogenesis in the ARBIB autonomous robot. Next in this section, a simple neural circuit for colour perception is described, before considering how a complex humanoid nervous system might be organized using the Hi-NOON architecture. Finally, we conclude (Section 6).

2 Overview of Hi-NOON

In this section, we consider the structure and function of the neural simulator Hi-NOON. The original program was written in object-oriented Pascal and was used for modeling small systems of biological neurons, notably work with *Aplysia*'s withdrawal reflex [27]. Hi-NOON has subsequently been rewritten in C using the disciplines of object-oriented programming (OOP) [8, 13]. C was used (rather than C++ with its explicit support of OOP features) to maximize portability among various realizations in different applications. The benefits of the OOP approach are two-fold. First, the ability for objects to inherit properties from other objects means that it is easy to define more physiologically exact neurons in terms of simpler neurons. Thus, the system allows a simple threshold unit as the most basic type of object. More complex objects inherit certain properties from this object (e.g. the fact that it has weighted connections to other objects). The second benefit of OOP is polymorphism. This means that the network may contain many different types of neuron, at many levels of complexity, without the programmer having to be concerned with this.

2.1 Gross Anatomy of Hi-NOON

Hi-NOON comprises two elements (Figure 1): a Message Passing Interface (MPI) based simulator which distributes its simulation over n processes, and a non-MPI simulator intended for (but not limited to) sensory and motor processing tasks. By allowing the division of the nervous system model in this way, areas that require special processing resources can be accommodated.

As shown in Figure 1, clients and servers [9, 24] are very similar in organization:

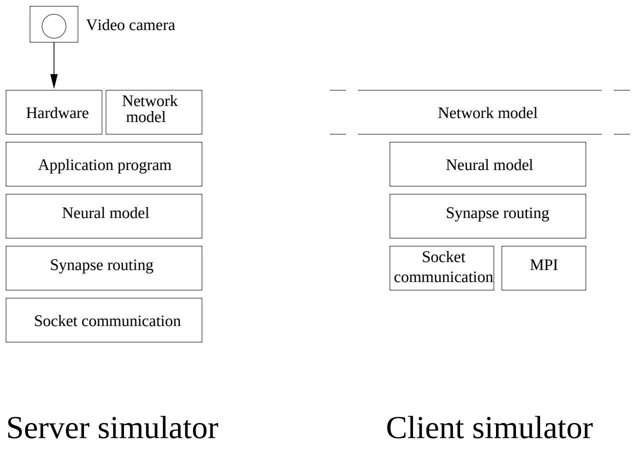
Network model: This describes the 'nervous system' that executes on Hi-NOON. The MPI simulator distributes it among all client processes, where each process forms a network object from its portion of the network model. Servers have individual models and share the network layer with hardware specific to their application in the robot. It is possible for a server to be "compute-only". That is, it has no hardware interface to the sensory or motor circuits in the robot.

Hardware: This aspect of a server simulator is the interface between the network model and the robot's sensor and motor systems. Examples include serial and parallel communication ports and video digitizers.

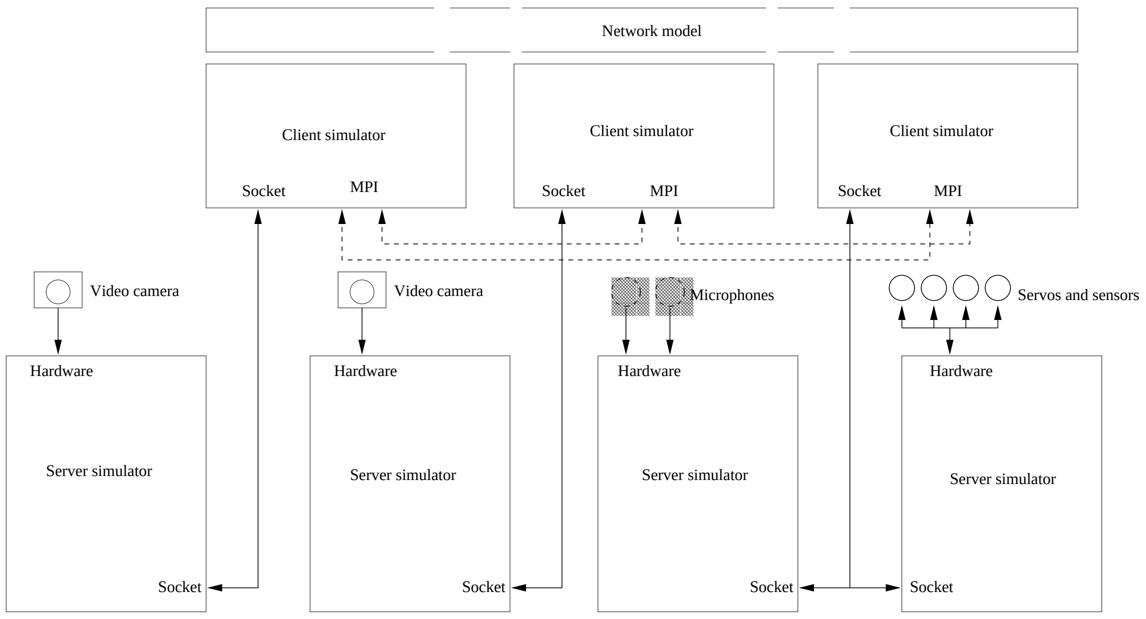
Application program: Server simulators that control interface hardware need device driver software. This is provided through the application program. Servers that have no hardware interface to the robot's sensor and motor systems have a very simple application program. In this case, it simply calls the simulator code.

Neural model: This contains the algorithms that describe neuron and synapse behaviour.

Synapse routing: This controls the destination of messages that are relayed around the system. For example, a message from a synapse in a server simulator can have a destination that exists within the same simulation, or in the simulation run by the client that the server is connected to (socket-to-socket communication). Communication routing performed by a client is a little more complex. In this case a message may have a destination that is found in part of the nervous system that exists in the same client process, or in a server (that requires socket-to-socket communication), or indeed on another client process (that requires MPI client-to-client communication.)



(a) Client and server simulator architectures



(b) Client-server simulator connectivity

Fig. 1. The architecture of Hi-NOON. (a) Client and server simulators can be standalone, but are intended to be used together to form a client-server architecture for large neurally-based experimental robot control applications. (b) In such a joint effort, the MPI simulator processes running on a high performance cluster of processors are the clients communicating through sockets to the non-MPI server(s) running on PCs with special purpose input and output (I/O) facilities.

Socket communication: Client and server use this to communicate through a socket-to-socket internet connection.

MPI: Client processes use the MPI standard for inter-process communication and synchronization. Synchronization is achieved through blocking and is necessary for processes that are running on different CPUs with different loads. Without this safeguard, the different sections of the network model would execute at different rates.

2.2 Neuron Parameters

Basic neurophysiology suggests the attributes a model spiking neuron should have. The fixed parameters `BaseMP`, `Threshold` and `TimeConst` correspond to the resting potential, threshold and time constant of the neuron, respectively. Dynamic parameters `MP`, `SynPot` and `fired` (a 1/0 predicate) model the actual membrane potential as it varies in time, accumulate the weighted sum of synaptic inputs which influence the updating of `MP` at the next time step, and indicate if the object is in the process of firing, respectively. This parameter system allows us easily to describe differences between neurons and to keep track of the changing states of neurons over time. It approximately satisfies Selverston's [30] "minimum requirements" for effective neural modeling.

2.3 Hi-NOON Objects

The neural network is held as a list of objects, where each such object corresponds to a single neuron and holds all the information about its state (see below) and about subsidiary objects. The information held in the neuron object is comprised of:

- a set of parameters which defines the neuron;
- a set of data structures which defines the 'axon terminals' for the neuron, each of which is itself an object and has its own parameters;
- a set of methods (pointers to functions) which access and alter parameter values and so determine exactly how the neuron functions.

The top-level list corresponds to the network object. This possesses two methods (called `h_access` and `add`) for accessing network objects and adding further objects onto the list, respectively. Simulation run length is handled by a global object. This stores the simulation and concurrent socket interface 'housekeeping' data, including a counter whose original value specifies the length of simulation. It decrements after each evaluation of the network object, and the simulation halts when the counter reaches zero.

As synapses are also objects, they too have fixed and dynamic parameters similar to those of neurons. Thus, `BaseWeight` is the default weight of the synapse and is a constant; `Weight` holds the present synaptic strength and is variable during simulation; `Recovery` is a constant (within each synapse) which determines how quickly `Weight` returns to `BaseWeight`. To prevent synaptic weights growing without limit, `Weight` is bounded during simulation. This models the finite stores of neurotransmitter in the synaptic terminals of biological neurons.

2.4 Neuron Types

Hi-NOON allows a non-homogeneous population of neurons to be simulated – reflecting the fact that neurons have specialized functions in real neurobiological systems – at the most appropriate level of abstraction. Modeling individual neurons at the level of membrane potential allows sub-threshold and spiking behaviors to be simulated at low computational cost. The fixed parameters cater for differences between neurons which, in this work, are of the following types:

basic: tells its synapses to fire when its membrane potential crosses threshold from below.

noisy: similar to **basic**, but has an additional internal noise component determining the weighted synaptic input, and hence influencing the membrane potential at the next time step.

ramp: similar to **noisy**, but has ability to ramp up spike generation rate. It is used as a test signal source in network development.

burst: similar to **noisy** but produces a short burst of spikes when its membrane potential crosses threshold.

sensor: similar to **basic**, but acts as a sensory neuron in a situated system, such as a mobile robot.

motor: similar to **basic**, but acts as a motor neuron in a situated system.

2.5 Approximate State System

Each neuron is treated as being in one of a number of six states depending on the present membrane potential, cell threshold and whether or not the cell has just fired, etc. For example, if the membrane potential of the basic cell is above threshold, and the cell has not just fired, then the neuron will start to generate a spike and will initiate synaptic transmission.

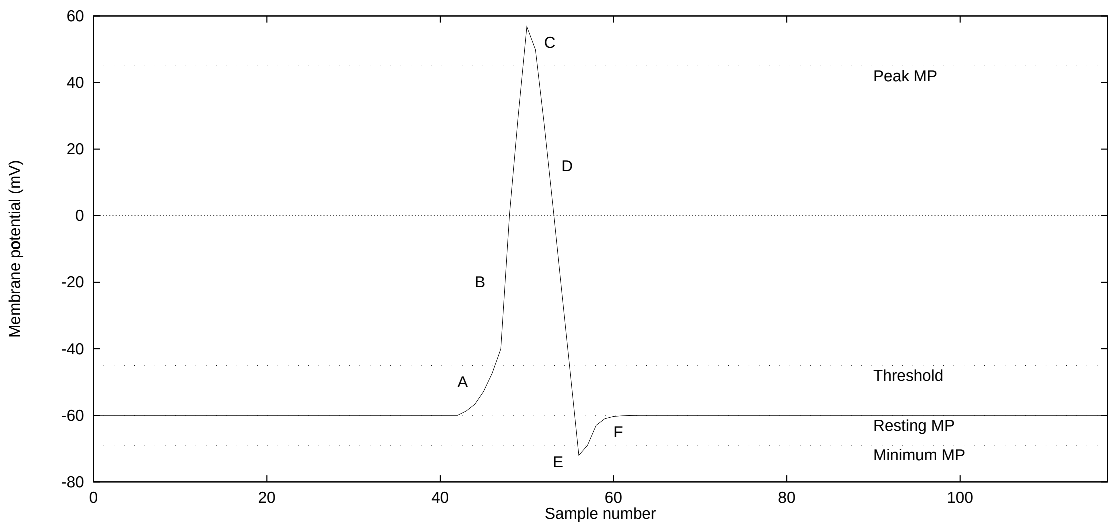


Fig. 2. Time evolution of typical action potential (spike) of a **basic** neuron in a Hi-NOON simulation. See text for specification of the states (A..F) passed through by a neuron during firing. Here, the sample period is approximately 4 ms (this varies with the machine on which the simulation runs.)

Figure 2 (taken from a Hi-NOON simulation) shows the states passed through by a neuron during firing. In the case illustrated, the minimum, resting and peak potentials of the neuron are set at -69, -60 and +45 mV respectively, and the threshold value was -45 mV. Note that actual values will under/overshoot these settings before state can change at the next iteration of simulation. The states are:

A:	`MP` above resting potential and below threshold
B:	above threshold and below peak
C:	at peak
D:	post-firing
E:	at minimum
F:	hyperpolarised

The equations governing the membrane potential in each of these states and the synaptic weights are given in Section 3 below.

The use of a state system for controlling the membrane potential facilitates the addition of new features to the program; it is only necessary to identify which of the states may trigger this feature and to add a procedure call at that particular state. This, coupled with OOP's inheritance, allows models to be developed and altered with a minimum of changes to the source code.

2.6 Axonal and Synaptic Transmission

Our neurons model sub-threshold behaviour but sub-threshold potentials are not propagated (from axon hillock to terminal fibers) in real neurons, only action potentials are. For computational practicality, we do not attempt to model (regenerative) spike transmission along the axon. This, however, is not a serious concern because the model's behaviour depends entirely on how pre-synaptic activity is transformed into post-synaptic activity. It is only in supra-threshold states B, C and D (see Figure 2 and Section 3.2) that synaptic communication can take place. Hence, it is irrelevant that we are, in some sense, modeling sub-threshold behaviour incorrectly. An alternative view is that we are not modeling axonal transmission, i.e. we have ‘point’ neurons as is common in neural modeling [22, pp. 21–24].

2.7 Learning in Hi-NOON

There is no specific support for learning in Hi-NOON. Thus, if PDP-type learning (e.g. back-propagation) is to be used, this must be implemented external to the simulator. In light of Hi-NOON’s ability to model at the level of transmembrane potential, however, there is implicit support for biologically-based forms of learning, such as habituation, sensitization and classical conditioning [12, 19, 21]. Generally, these simple forms of learning are implemented using synapse-on-synapse connections in Hi-NOON.

3 Neurons and Synapses

In this section, we present more detailed descriptions of neurons and synapses within Hi-NOON. Since it is intended for (among other things) applications in situated robotics studies, there is provision for sensory and motor neurons which connect to the environment, as well as for more prosaic ‘basic’ (information processing) neurons.

3.1 Neurons

The ‘basic’ neuron type has the state system functionality which is subsequently embedded in all derivatives, such as the sensory and motor cells.

Basic neurons Updating equations for the membrane potential (MP – in millivolts) for this neuron type are:

$$\begin{aligned}
&\text{state A:} \quad MP(t+1) = MP(t) - \tau + S(t) \\
&\text{state B:} \quad MP(t+1) = MP(t) - \alpha + S(t) \\
&\text{state C:} \quad MP(t+1) = h + S(t) \\
&\text{state D:} \quad MP(t+1) = MP(t) - \mu + S(t) \\
&\text{state E:} \quad MP(t+1) = l + S(t) \\
&\text{state F:} \quad MP(t+1) = MP(t) + \tfrac{BaseMP - MP(t)}{\eta} + S(t)
\end{aligned}$$

where:

$S(t)$ is the synaptic potential (`SynPot`) at time t, equal to $\sum_i w_i \kappa \left(MP_i(t) - BaseMP_i\right)$

i is a counter which counts over active pre-synaptic cells.

w_i is the synaptic weight from a pre-synaptic neuron.

τ is the neuron time constant.

$\eta = 1.5$ is the post-undershoot increment rate.

$\mu = 25$ is the post-action potential peak-MP decrement.

$\kappa = 1/450$ is a heuristically-set learning constant.

$\alpha = 20$ is the post-threshold attack increment.

$h = 45$ is the post-threshold maximum MP.

$l = -69$ is the pre-undershoot minimum MP.

Certain of the above parameters (e.g. τ, η) are time-dependent and have been set empirically to suit a range of processor speeds and implementations. However, they may be inappropriate in some circumstances (as when implementing a real-time robotic system using a fast processor).

Sensory neurons The sensory neurons react to changes in light level, object colour and proximity.

1. Changes in light level: Hi-NOON is equipped with light sensing cells inspired from the visual ability of the scallop *Pecten maximus* [33]. Light, shadow and movement detection in this animal is provided through a compound of approximately 60 eyes. Each consists of a cornea, a large lens, sense (Δ) cells arranged as proximal and distal retinae, a reflecting argentae and a layer of screening pigment cells. Proximal cells respond to an increase in light intensity by generating APs until a constant illumination has been detected, at which point spike generation decays. Discharge ceases with a decrease in light intensity. By contrast, distal cells respond to a decrease in light intensity, showing no activity during constant illumination or an increase in light intensity [33, pp. 244–245].

Optical devices feed delta sensory neurons with a sampled value $\lambda(t)$. There are two modes of operation: *proximal* and *distal*. Distal mode operation simply inverts the measurement taken from the environment. However, distal mode depolarizes the MP in response to measured decreases.

Proximal mode: depolarizes the membrane with an increase in light intensity, according to:

$$MP(t+1) = \begin{cases} MP(t) + \Delta\lambda & \text{if } \Delta\lambda > 0 \\ MP(t) & \text{otherwise} \end{cases}$$

Distal mode: depolarizes the membrane with a decrease in light intensity, according to:

$$MP(t+1) = \begin{cases} MP(t) - \Delta\lambda & \text{if } \Delta\lambda < 0 \\ MP(t) & \text{otherwise} \end{cases}$$

where:

$$\Delta\lambda = k[\lambda(t+1) - \lambda(t)]$$

and k is an empirically set scaling factor whose value depends on the particular physical sensor employed.

2. Proximity sensing: Distance sensors feed sensory cells with a sampled value (approximately) proportional to object range, R, at time t, whence:

$$MP(t+1) = MP(t) + \lfloor sR(t) \rfloor$$

where $\lfloor\,\rfloor$ represents the floor function and s is an empirically set scaling factor whose value depends on the particular physical sensor employed.

In a similar way, sensory neurons can be grouped to form a retina-like structure. In this case, each neuron responds to red, green, or blue light instead of a proximity stimulus, having its receptive field defined at the start of the simulation.

Motor neurons These close the loop between the nervous system and the environment. Motor drive activity is given as:

$$A(t) = \left\lfloor \frac{\pm\gamma MP(t)}{h} \right\rfloor$$

where $A(\,)$ is used at a lower level at abstraction (closer to the hardware) to determine the speed of motor output, and γ is a scaling constant set to suit the particular robot hardware in use. Its sign is determined by the requirement for forward or reverse drive.

3.2 Synapses

The basic synapse (which is noise free) is shown in Figure 3. It has functionality which is subsequently embedded in all derivatives such as the habituating, sensitizing and conditioning types. These allow us to implement a simple, biologically-based form of learning.

Weights are updated as:

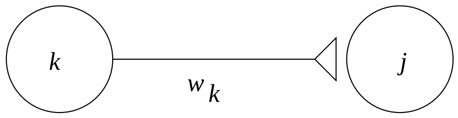


Fig. 3. Synapse from parent neuron k to target neuron j, with weight w.

$$w(t+1) = \begin{cases} w(t) - \beta & \text{if } w(t) > w_{\text{base}} \\ w(t) + \beta & \text{if } w(t) \leq w_{\text{base}} \\ w_{\max} & \text{if } w(t) > w_{\max} \\ w_{\min} & \text{if } w(t) < w_{\min} \\ w_{\min} & \text{otherwise} \end{cases}$$

where β is the MP recovery parameter and w_{base} is the base weight (typically 0). These are individually set (together with $w_{\min}$ and $w_{\max}$, typically ± 16) for each neuron.

Noise-free synapse A noise-free synapse is guaranteed to fire whenever the parent neuron fires, according to:

$$\texttt{fired}(t) = \begin{cases} \text{TRUE} & \text{if (state B, C, D)} \\ \text{FALSE} & \text{otherwise} \end{cases}$$

Noisy synapse A noisy synapse will fire, with a probability set by the amount by which membrane potential exceeds threshold, whenever the parent neuron fires:

$$\texttt{fired}(t) = \begin{cases} \text{TRUE} & \text{if } \texttt{cond1} \\ \text{FALSE} & \text{otherwise} \end{cases}$$

where:

$$\texttt{cond1} = (\text{state B, C, D}) \wedge \left(\frac{MP_p - \theta_p}{h - \theta_p} \times 100 \geq \text{rand mod100} \right)$$

and p denotes a parent (pre-synaptic) neuron.

Habituating type The magnitude of the synaptic weight w_k, shown in Figure 3, is simply decreased every time the synapse fires.

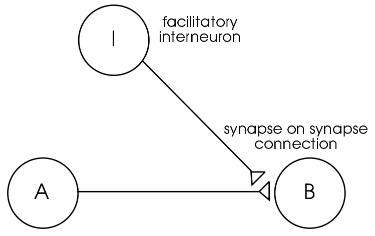
$$|w(t+1)| = \begin{cases} |w(t)| - d & \text{if state C} \\ |w(t)| & \text{otherwise} \end{cases}$$

where d is a constant decrement (typically ~ 1).

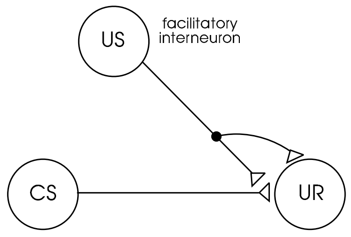
Sensitizing type The synapse from A to B shown in Figure 4 is strengthened every time I fires, according to:

$$w(t+1)_{\text{targ}} = \begin{cases} w(t)_{\text{targ}} + w(t)_{\text{sos}} & \text{if } \texttt{cond2} \\ w(t)_{\text{targ}} & \text{otherwise} \end{cases}$$

where $\texttt{cond2}$ is $\texttt{fired}_{\text{targ}} \wedge \texttt{fired}_{\text{sos}}$, ‘targ’ denotes the target synapse (to be sensitized) and ‘sos’ denotes the synapse-on-synapse influence.



(a)



(b)

Fig. 4. (a) Sensitization and (b) classical conditioning are modeled using synapse-on-synapse connections.

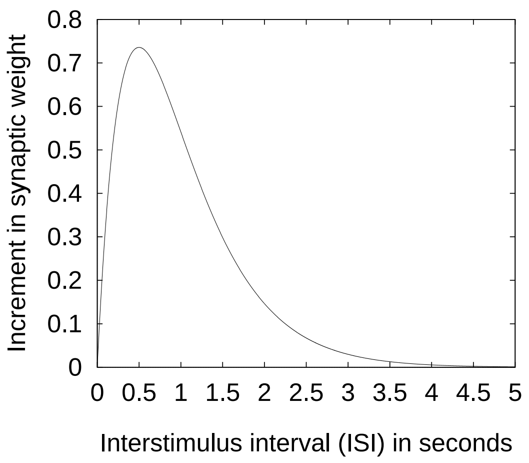


Fig. 5. Increment in synaptic weight through conditioning as a function of the ISI.

Conditioning type The synapse from conditioned stimulus (CS) to unconditioned response (UR), shown in Figure 4, is strengthened proportionally to the interstimulus interval (ISI) between the CS and the US firing. Maximum strengthening occurs when the CS precedes the US (the ISI) by 0.5 s. This is illustrated in Figure 5. Thus:

$$w(t+1)_{\mathrm{targ}} = \begin{cases} w(t)_{\mathrm{targ}} + kw(t)_{\mathrm{sos}} & \text{if } \texttt{cond2} \\ w(t)_{\mathrm{targ}} & \text{otherwise} \end{cases}$$

where:

$$k = \frac{nT}{\psi} \exp\left(\frac{-nT}{\varsigma}\right)$$

and nT is a count of sample periods initiated by encountering state C for the target neuron, ψ (= 250) is an empirically-set scaling factor and ς (= 500) is a constant chosen to maximize the effect of conditioning when the conditioning stimulus precedes the unconditioned stimulus by 0.5 s.

4 Clues from Biology for Long-Term Memory

As previously implemented [11] ARBIB suffered a major shortcoming in that its learning was far too plastic (because of recovery of synaptic weights). This problem has been well recognized in the *stability-plasticity* dilemma ([16, p. 63] and [7]). That is: How can a nervous system retain its stability of learning while still being plastic enough to adapt in a changing environment?

In this section, we consider the clues that biology has given us for building a long-term memory that will correct this shortcoming. A possible biological substrate for long-term memory is the synapse. Sensitization experiments with *Aplysia* have associated long-term memory with the growth of new synaptic connections activated by altered gene expression and protein synthesis [1, 35]. Studies of rats given reaching [15] or acrobatic training tasks [3] have also shown an increase in synapse count.

4.1 Synaptogenesis Through Long Term Potentiation (LTP)

Synaptogenesis, which has been found to occur after LTP, duplicates (and so strengthens) localized synapse connectivity. Hence, Toni et al.[34, p. 421] write:

> "As pharmacological blockade of LTP prevented these morphological changes, we conclude that LTP is associated with the formation of new, mature and probably functional synapses contacting the same presynaptic terminal and thereby duplicating activated synapses."

These authors continue:

> "This mechanism of synapse duplication maintains specificity in information processing and provides a framework for understanding the numerous *in vivo* experiments that have demonstrated an increase in synapse number and complexity of dendritic arborization following exposure to an enriched environment or learning paradigms"(p. 424).

Also, Levitan and Kaczmarek [20, p. 501] write on the subject of associative LTP:

> "The requirements for temporal pairing of the two stimuli are identical to those required for associative learning paradigms."

So the question arises: can we implement synaptogenesis in Hi-NOON using a variant of the associative mechanism? That is, can a temporal relationship be used to decide the strength of new synapses? This question is taken up in Section 5.2 below.

4.2 Cell Biological Alphabet

An intriguing possibility proposed by Hawkins and Kandel [19], following studies of *Aplysia*, is that the (associative) learning mechanism of classical conditioning may be based on the same neural mechanisms as (non-associative) habituation and sensitization. They go on to state (p. 375): "This finding suggests that the mechanisms of yet higher forms of learning may similarly be based on the mechanisms of . . . simple forms of learning" referring to this as "an elementary cellular alphabet of learning" (p. 376).

This idea can be extended to synaptogenesis in Hi-NOON where the temporal relationship used for determining the strength of a new synapse can be an evolved form of the associative classical conditioning mechanism. The newly-created synapse has a strength calculated from the difference between the elapsed time of post-synaptic cell firing and elapsed time of conditioned synapse firing.

Pursuing this idea, it was decided to constrain the formation of new synapses by introducing a new predicate into the Hi-NOON model. This is simply the condition that a new synapse is only created, parallel to an existing conditioned synapse, once the conditioned synaptic strength reaches some percentage (say, 90%) of the allowed maximum. This is illustrated in Figure 6 with the gradual creation four new synapses between neurons A and B. The same temporal mechanism is common to both classical conditioning and to synaptogenesis in our simulator.

5 Situated Systems

In this section we begin by describing how the Hi-NOON simulator has been used to design and implement the 'nervous systems' of two rather different situated systems, a model of phonotaxis behaviour in the cricket, and a model illustrating synaptogenesis using the ARBIB autonomous robot. Also, a simple neural circuit for colour perception, taken from related work, is described as an example of a specialized sense modality interfaced to a Hi-NOON server simulator. This section then concludes with a look at how a complex humanoid nervous system might be organized using the Hi-NOON architecture.

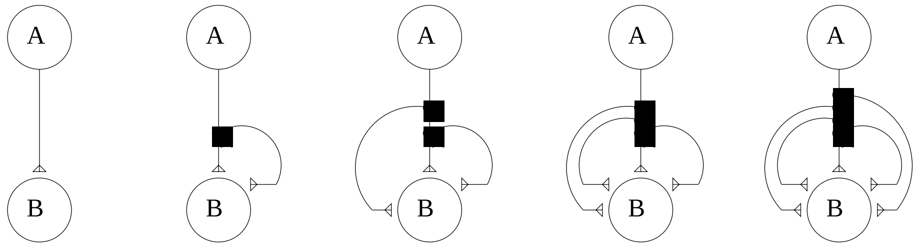


Fig. 6. Synaptogenesis leads to the growth of new synaptic connections, so strengthening the overall $A \rightarrow B$ connection weight.

5.1 Cricket Phonotaxis

Webb and Scutt [36] have used Hi-NOON to simulate and then implement the auditory system of the cricket within a mobile robot, to study the neurophysiological underpinnings of phonotaxis, i.e. the movement of the female towards the male's mating song. The robot produces behaviour closely similar to the cricket in most situations. In their words: "No alternative models have as yet been presented with a comparable detail or evaluation".

5.2 ARBIB

Another situated system based on Hi-NOON simulations is the ARBIB autonomous robot [10] which has been implemented on a variety of hardware and software platforms. ARBIB learns from and adapts to its environment, which consists of hard objects and light sources casting shadows. A primary goal of this work was to test the notion that effective robot learning can be based on neural habituation and sensitization, so validating the suggestion of Hawkins and Kandel [19] that (associative) classical and 'higher order'conditioning might be based on an elaboration of these (non-associative) forms of learning, cf. their 'cell biological alphabet'.

Accordingly, ARBIB's 'nervous system' has a non-homogeneous population of spiking neurons, its drive to explore its environment was provided by a simple central pattern generator neural circuit [29], and learning was by modification of a basic, pre-existing ('hard-wired') reflex to reverse and turn on hitting an obstruction. By monitoring firing rates of specific neurons and synaptic weights between neural connections as ARBIB learns, we have confirmed that both classical and higher-order conditioning occur, leading to the emergence of interesting and ecologically-valid, obstacle-avoidance behaviors.

Synaptogenesis ARBIB's synaptic plasticity, provided by models of associative classical conditioning and non-associative sensitization and habituation, allows it to learn collision avoidance skills by switching from short- to long-range sensory modalities. However, once the long-range modality has become dominant in eliciting the avoidance reflex, this conditioned response will eventually extinguish via autonomous decay [10, 32], which constitutes ARBIB's short-term memory. Thus, the useful information is lost until ARBIB once again re-acquires the skill through stimulus substitution for a collision avoidance reflex. This is the stability-plasticity dilemma [7, 16].

ARBIB therefore needs a way to keep useful information acquired by its short-term memory before it extinguishes. Here, we apply the synaptogenesis introduced in the previous section to ARBIB's nervous system in an attempt to stabilize the retention of useful information.

Obstacle avoidance experiments A Nomad Scout 2 robot (see `http://www.robots.com`), controlled through the Nomadic Technologies *Nserver* software via the Scout's *host* port, was used for all experiments reported in this subsection. The Scout has 6 bump sensors and 16 sonar devices arranged around its circumference (Figure 7). In this instantiation of ARBIB, the infrared sensory neurons have been replaced by sensory cells that are coupled to the Scout's forward facing sonar devices and the delta light sensory neurons have been disabled.

A total of 14 runs was carried out. Each 11 minute run consisted of ARBIB having free range to travel around the robot laboratory, negotiating obstacles in its path. The first seven runs were made with synaptogenesis enabled in the Hi-NOON model. For comparison, the remaining runs were made with synaptogenesis disabled [14].

By interaction with the environment, ARBIB learns (by stimulus substitution) to elicit its avoidance reflex. Hence, the measure of bump sensory neuron activity (the unconditioned stimulus) is a useful indicator of how successfully it has learned to avoid direct contact with obstacles. Figure 8 shows the average action potential count for bins of 100 membrane potential sample points throughout the tests.

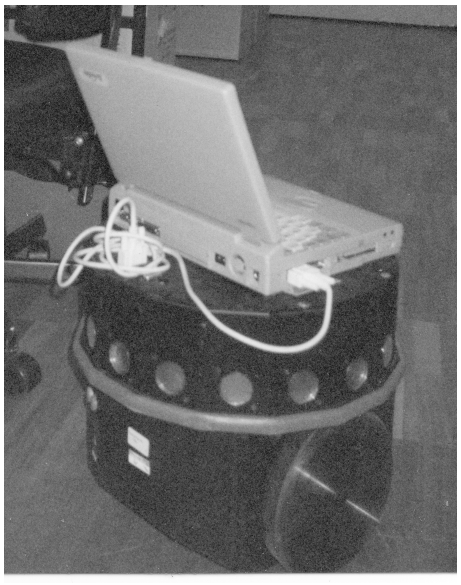


Fig. 7. The Nomad Scout 2 instantiation of ARBIB used in the synaptogenesis experiments.

Results of the first seven runs are shown in Figures 8(a) and 8(b). Comparing these results with Figures 8(c) and 8(d), we see a decrease in activity during the runs with synaptogenesis enabled. An interesting observation is that Figure 8(b) still shows activity at the end of the test. This means that stimulus substitution during the synaptogenesis runs has improved the obstacle avoidance skill of ARBIB, but seems not to have saturated the network with an overwhelming population of new synapses.

However, the important subject of 'forgetting' in the context of synaptogenesis must be made a focus of future work. As pointed out by Bailey and Kandel [1, p. 42]:

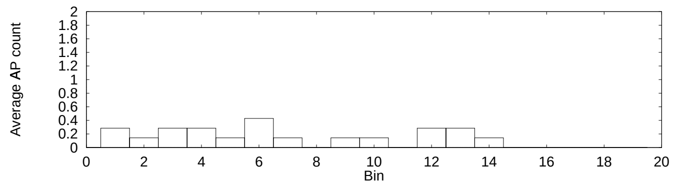
> "Recent studies in vertebrates further strengthen the idea that the nervous system may utilize similar cell biological mechanisms to achieve learning-related plasticity in the mature animal as it does for synapse formation during development . . . at critical developmental stages the refinement of synaptic connections, their *growth* and *regression*, is determined by an activity dependent process that seems related to LTP in the hippocampus."

Cell		Synapses	
Pre	Post	Average number	Collective weight
RIR	IIL	18	22
RIR	IIR	30	139
LIR	IIL	18	128
LIR	IIR	6	30

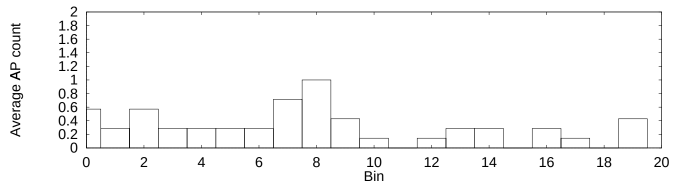
Table 1. Summary of synaptic growth and collective synaptic strength. KEY: LIR – left infrared sensory neuron; RIR – right infrared sensory neuron; IIL – left ipsilateral infrared interneuron; IIR – right ipsilateral infrared interneuron. See [10, Fig. 5] for full details. *Pre* and *Post* are the identifiers of pre- and post-synaptic cells, respectively. (In the implementation used here, 'infrared' sensory neurons were actually connected to the forward sonar sensors of Nomad.)

In Table 1, *Pre* and *Post* are the identifiers of pre- and post-synaptic cells, respectively. The terms *average number* and *collective weight* refer to the average number of synapses created between the pre- and post-synaptic cells over the first seven test runs, and the total synaptic weight between the pre- and post-synaptic cells also averaged over the first seven test runs, respectively.

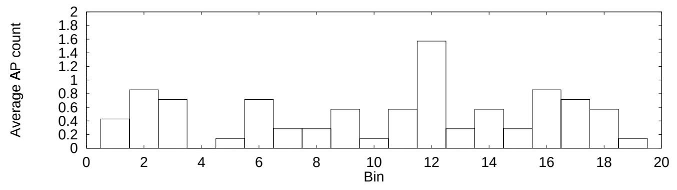
The collective weight values of 139 and 128 in Table 1 show an important adaptation in ARBIB's nervous system and so are worthy of closer inspection. These two figures are interesting because their pre- and post-synaptic cells are also the ipsilateral pairs of neurons which allow the sonar sensors to act, through stimulus substitution,



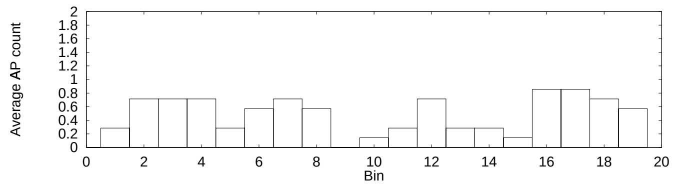
(a) Left bump sensory neuron firing activity with synaptogenesis.



(b) Right bump sensory neuron firing activity with synaptogenesis.



(c) Left bump sensory neuron firing activity without synaptogenesis.



(d) Right bump sensory neuron firing activity without synaptogenesis.

Fig. 8. Neural activity in left and right bump sensory neurons with and without synaptogenesis.

for bump sensors on the same side [10, Figure 5]. This makes more sense than if the contralateral connections – the pre- and post-synaptic pairs with collective weights of 22 and 30, respectively – had become dominant. If the sonar sensory transducers were to be swapped over, we expect that the contralateral pairs would become dominant instead. This experiment remains to be done.

5.3 Colour Perception

This subsection describes related work with Hi-NOON and is given as an example of interfacing a complex sense modality to a Hi-NOON server simulator. The Hi-NOON retina is constructed from an array of (10 × 10) receptive fields. Each receptive field is inspected by three neurons sensitive to red, green, and blue light. This arrangement results in 300 sensory neurons spread over a grid of 100 receptive fields, shown in Figure 9.

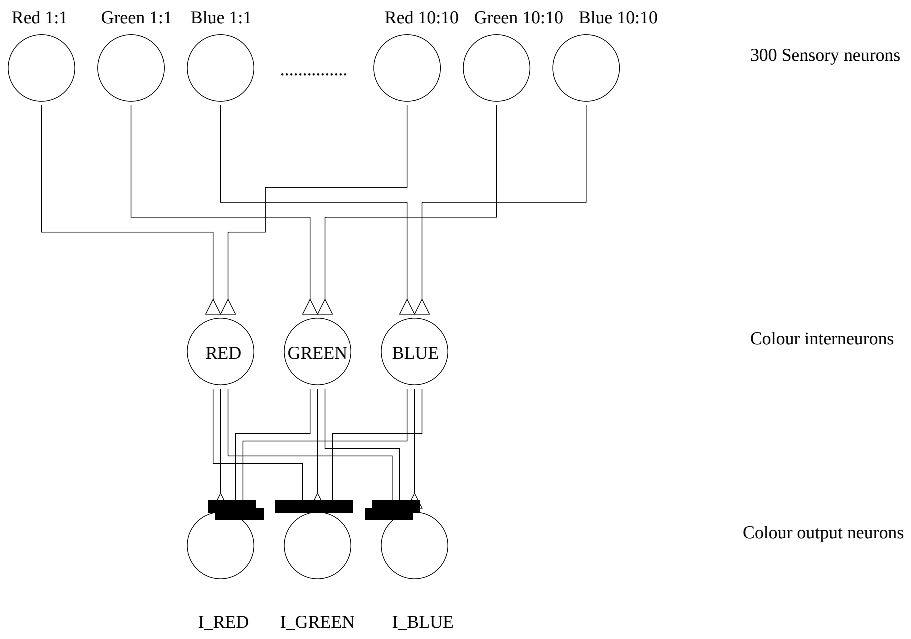


Fig. 9. A neural circuit for colour detection in the retina.

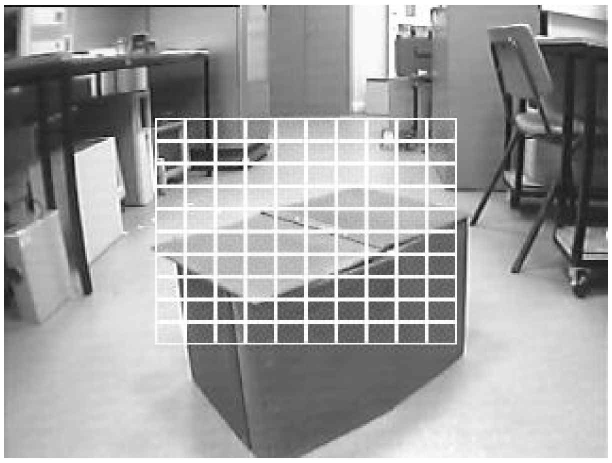
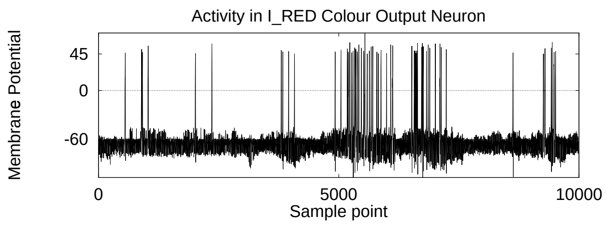
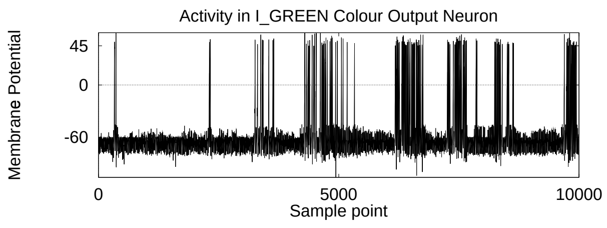


Fig. 10. View from the robot's camera of a blue object, showing the retina.

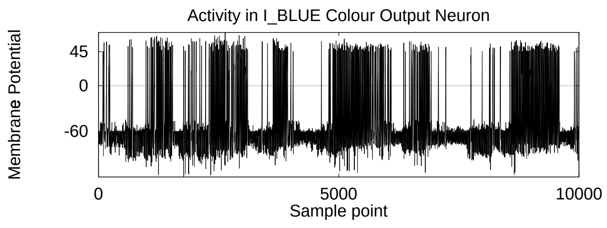
Each red sensory neuron makes an excitatory connection with the RED interneuron. The same scheme is followed by the green and blue sensory neurons (which in turn connect to the GREEN and BLUE interneurons, respectively). Inhibitory connections made from the RED interneuron to the I_GREEN and I_BLUE colour output neurons (with the same reasoning applying to the GREEN and BLUE interneurons) create competition between the colour outputs. An image taken from the retina of a blue object is shown in Figure 10, with an example of circuit activity is shown in Figure 11.



(a) Activity due to the colour red in the image.



(b) Activity due to the colour green in the image.



(c) Activity due to the colour blue in the image.

Fig. 11. Neural activity in the I_RED, I_GREEN, and I_BLUE colour output neurons of Figure 9, in response to the image formed on the retina shown in Figure 10.

As can be seen in Figure 11(c), the I_BLUE neuron shows the most activity. Hence, this indicates correctly that the image formed on the retina has a greater blue component than red (Figure 11(a)) or green (Figure 11(b)).

5.4 Applying Hi-NOON to Humanoids

Brooks, perhaps the best known exponent of behavioural robotics, has raised two concerns about biology as a motivation for robotics. First, a danger lies in modeling biological systems too closely because this may result in a model that is inefficient or indeed has redundant structures [5]. Second, any attempt at functional decompositions from what is known of the brain should be treated with scepticism because it "… does not have the modularity that either a carefully designed system might have, nor a philosophically pure, wished for, brain might have." [6] Our work recognizes these points, as Hi-NOON abstracts away from fine grain details of, for example, ion channels, and is not explicitly targeted at the modeling of "modules extracted from" vertebrate brain.

Humanoids and the Turing test Harnad [18] has observed that the Turing test is underdetermined and so has described a hierarchy of tests which could be applied to robots of varying sophistication. The simplest level of the hierarchy, T1, is concerned with models that describe a fragment of our total abilities. Next in the hierachy is T2, the original Turing test. Human sensory-motor capabilities are the domain of T3, with T4 concerned with robots which are neuromolecularly identical to their human counterparts.

Hence, a humanoid enters the hierarchy as a candidate for T3. It is a machine which duplicates our sensory-motor capabilities in such a thorough way that it is potentially able to gain experience of the world comparable to our own.

Motivational states – a clue to scaling? So far our machines are candidates for a Turing test falling between T1 and T2 of Harnad's hierarchy. We are still very much on the learning curve and don't know the details of cognitive architectures needed by a humanoid. However, through animal and animat studies, we are able to identify some of the low-level attributes that exist in all creatures, from simple molluscs and insects upwards. For example, animals have motivational states, they experience hunger and fear to name but two. Motivational states can be thought of as the driving force for an animal's activities in its environment.

Let us provide our Hi-NOON based robot, ARBIB, with motivational states for hunger (for appetitive stimuli), and fear (of aversive stimuli). Once active, a motivational state will prime appropriate behaviors which are then selected by releasers in the environment. For example, if the robot is in the presence of an aversive stimulus then it may freeze or attempt to escape. Alternatively, if it is hungry it may forage for appetitive stimuli, and when in the presence of an appetitive stimulus it may approach and attempt to collect it. These motivational states are mutually exclusive. ARBIB cannot exhibit both hunger and fear at the same time. Similarly, it cannot both freeze and attempt escape when fear is the dominant motivational state. It would be interesting to see how ARBIB copes with motivational conflicts.

Also, ARBIB, on the route from animat to humanoid, must be able to set its own goals. This aspect of autonomy has been defined by Smithers [31]:

> "…the underlying notion is one of self-law making, or self-governing, and it is closely related to the concepts of self-identity and self-determination: an autonomous agent is one whose behavior is regulated by rules or laws generated by itself. As a result, its identity is a product of itself: it has self-identity."

The principle of an agent's identity being a product of itself (and interactions with its environment) has been commented on by Black [2]:

> "… Animals raised in complex environments are superior at many different types of learning tasks. These abilities are generalized across a wide range of learning tests, … the enriched rat appears to have *learned how to learn* better."

Perhaps the way to approach this is by basing all behaviour upon lower level motivational states. For example, the conditioning of motivational states to chained external stimuli might lead to a kind of "expectation of a unconditioned stimulus (US)", and from there to the *dual-brain hypothesis* as outlined by Lieberman [21]:

> "Perhaps the first learning system to evolve was a relatively primitive one in which the conditioned stimulus (CS) was simply associated with the US, and thus elicited the same responses. In the course of time, a higher, cognition-based system appeared that involved active anticipation of the US, thus allowing subjects to select a wider range of preparatory responses than the innate ones. Insofar as both systems still coexist in vertebrates, this would explain why animals sometimes act as if the CS were a signal for food and at other times as if it were actually food."

Interaction A humanoid's competences must include social interaction. For this to be realistic, a humanoid must be accepted as an individual through the display of human-like pre-programmed behaviors. Such built-in 'knowledge' will help people relate to, and interact with the robot.

Through social interaction, a humanoid learns the correct (and incorrect) approaches to achieving its goals, many of which will be inextricably linked with those of the people interacting with it. This can be approached by learning secondary reinforcers [21, p.202] which satisfy the robot's active motivational state. Thus, cooperation between humanoid and humans should emerge quite naturally as the robot's experience with people develops.

Generating circuits The next stage of Hi-NOON's development is the automation of neural circuit design. This will go some way towards solving the problem of designing the neural architecture of an artificial creature – building Minsky's 'library' [23]. Such artificial creatures will adapt to their environment at synaptic plasticity and nervous system architectural levels.

Implementation Building a humanoid robot is probably one of the most complex engineering tasks that could ever be undertaken. Everything from compliant limbs to colour vision must close the perception-action loop through the environment, controlled by a nervous system which has suitably specialized structures for these different aspects of the robot. Hence, a method of high-level implementation is needed.

One approach is an examination of neural subsystems in an object-oriented 'top down' manner. At the top level of abstraction, a humanoid's nervous system is one large network object. This in turn is represented as modules that handle specific I/O operations (e.g. pattern generators for controlling actuators with position feedback, or image processing provided by edge detection of moving stimuli). Decomposition continues until each module becomes a network object whose I/O requirement is small enough to be handled by a server simulator.

Hi-NOON's servers create a natural environment for this circuitry, supporting specialized hardware that in turn provides facilities such as image capture for vision and servo controllers for motor control, or serial and parallel communication ports to 'intelligent' subsystems. Distributing the specialized areas of the nervous system over many processors using the MPI and socket programming standards allows the heterogeneity of the system to span multiple platforms. If a different processor is particularly suited to a function, then it is only necessary to re-compile the source code for a new platform and connect the server with the rest of the network.

Hence, the large network object is eventually broken down into smaller, and as mentioned above, autonomously generated, objects that execute concurrently. The MPI client simulators connect the server simulators together. These are intended to form the foundations for circuitry capable of abstract reasoning that is grounded in the sensory-motor activity of the servers [17, 18].

An important result of this hierarchy is that high-bandwidth sense modalities such as vision are pre-processed neurally by the host server. Hence, a higher-level and lower-bandwidth representation can be projected to an MPI client process.

Additionally, because there is no constraint on the number of servers that define a humanoid, a cluster of server simulators that form its sensory and motor facilities can be grouped, together with one or more supporting clients, in an individual robot. Thus, a group of humanoids whose clients are linked through wireless ethernet allows a natural communication path between them.

6 Conclusions

Hi-NOON is an object-oriented neuronal circuit simulator specifically developed for studying the neurophysiological basis of behaviour in real animals and in situated artificial systems. It simulates changes in membrane potential (including spiking behaviour) rather than using the continuous activation functions typical of PDP-style artificial neural nets. The main simplifications are that it treats each neuron as a single compartment, with inputs modeled as added voltage. In place of differential equations, a state system is used, along with a flexible parameter system to cater for differences between neuron types and to keep track of the changing state of each neuron over time. This allows circuits of heterogeneous neurons modeled on real neurophysiological data to be constructed with a minimum of effort and processed with relative ease. The capability of Hi-NOON for use in situated systems studies is illustrated with examples. The expansion of Hi-NOON to a concurrent environment addresses the possibility of using it for realizing the large-scale, complex nervous system of a humanoid robot.

Acknowledgements

Many thanks to Ulrich Nehmzow, Ten Min Lee, Stephen Marsland, Jason Fleischer and Andrew Pickering of the Robot Research Laboratory, Department of Computer Science, University of Manchester, for their interest, help and use of equipment. The work reported here was supported by a studentship from the UK Engineering and Physical Sciences Research Council.

References

1. C. H. Bailey and E. R. Kandel. Structural changes underlying long-term memory storage in *Aplysia*: a molecular perspective. *The Neurosciences*, 6:35–44, 1994.
2. J. E. Black. How a child builds its brain: Some lessons from animal studies of neural plasticity. *Preventitive Medicine*, 27:168–171, 1998.
3. J. E. Black, K. R. Isaacs, B. J. Anderson, and A. A. Alcantara. Learning causes synaptogenesis, whereas motor activity causes angiogenesis, in cerebellar cortex of adult rats. *Proceedings of the National Academy of Sciences of the United States of America*, 87:5568–5572, 1990.

4. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 1:14–23, 1986.
5. R. A. Brooks. Intelligence without reason. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 569–595, Nagoya, Japan, 1991.
6. R. A. Brooks. From earwigs to humans. *Robotics and Autonomous Systems*, 20:291–304, 1997.
7. G. A. Carpenter and S. Grossberg. The ART of adaptive pattern recognition by a self-organizing neural network. *IEEE Computer*, 21(3):77–88, 1988.
8. P. Coad and E. Yourdon. *Object Oriented Analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1991. Second Edition.
9. D. E. Cormer and D. Stevens. *Internetworking with TCP/IP Volume 3: Client-Server Programming and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1996.
10. R. I. Damper, R. L. B. French, and T. W. Scutt. ARBIB: an autonomous robot based on inspirations from biology. *Robotics and Autonomous Systems*, 31:247–274, 2000.
11. R. I. Damper and T. W. Scutt. Biologically-based learning in the ARBIB autonomous robot. In *Proceedings of IEEE International Symposia on Intelligence and Systems*, pages 49–56, Washington DC, 1998.
12. N. H. Donegan, M. A. Gluck, and R. F. Thompson. Integrating biological and behavioral models of classical conditioning. In R. D. Hawkins and G. H. Bower, editors, *Computational Models of Learning in Simple Neural Systems*, pages 109–156. Academic, San Diego, CA, 1989.
13. A. Eliëns. *Principles of Object-Oriented Software Development*. Addison-Wesley, Wokingham, UK, 1994.
14. R. L. B. French and R. I. Damper. Stability of learning in the ARBIB autonomous robot. *6th International Conference on Simulation of Adaptive Behavior*, Paris, France, September 2000, forthcoming.
15. W. T. Greenough, J. R. Larson, and G. S. Withers. Effects of unilateral and bilateral training in a reaching task on dendritic branching of neurons in the rat motor sensory forelimb cortex. *Behavioral and Neural Biology*, 44:301–314, 1985.
16. W. Grey Walter. A machine that learns. *Scientific American*, 185(5):60–63, 1951.
17. S. Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.
18. S. Harnad. Grounding symbolic capacity in robotic capacity. In L. Steels and R. Brooks, editors, *The Artificial Life Route to Artificial Intelligence: Building Situated Embodied Agents*, pages 277–286. Lawrence Erlbaum, New Haven, 1995.
19. R. D. Hawkins and E. R. Kandel. Is there a cell biological alphabet for simple forms of learning? *Psychological Review*, 91:375–391, 1984.
20. I. B. Levitan and L. K. Kaczmarek. *The Neuron, Cell and Molecular Biology*. Oxford University Press, New York, NY, 1997.
21. D. A. Lieberman. *Learning: Behavior and Cognition (2nd edition)*. Brooks/Cole, Pacific Grove, CA, 1993.
22. R. J. MacGregor. *Neural and Brain Modeling*. Academic, London, UK, 1987.
23. M. Minsky. Analogical vs. logical or symbolic vs. connectionist or neat vs. scruffy. In P. H. Winston, editor, *Artificial Intelligence at MIT: Epanding Frontiers*, volume 1, pages 219–243. MIT Press, Cambridge, MA, 1990.
24. P. Renaud. *Introduction to Client/Server Systems*. John Wiley, New York, NY, 1996.
25. F. Rieke, D. Warland, R. de Ruyter van Steveninck, and W. Bialek. *Spikes: Exploring the Neural Code*. Bradford Books/MIT Press, Cambridge, MA, 1997.
26. T. W. Scutt and R. I. Damper. Computational modelling of learning and behaviour in small neuronal systems. In *Proceedings of International Joint Conference on Neural Networks*, pages 430–435, Singapore, 1991.
27. T. W. Scutt and R. I. Damper. Object-oriented modelling of small neuronal systems. *Artificial Intelligence and Simulation of Behaviour Quarterly*, 80:24–33, 1992.
28. T. W. Scutt and R. I. Damper. Designing a nervous system for an adaptive mobile robot. In A. Browne, editor, *Neural Network Perspectives on Cognition and Adaptive Robotics*, pages 220–250. Institute of Physics Press, Bristol, UK, 1997.
29. A. I. Selverston. A consideration of invertebrate pattern generators as computational databases. *Neural Networks*, 1:109–117, 1988.
30. A. I. Selverston. Modeling of neural circuits – What have we learned? *Annual Review of Neuroscience*, 16:531–546, 1993.
31. T. Smithers. Autonomy in robots and other agents. *Brain and Computation*, 34:88–106, 1997.
32. R. S. Sutton and A. G. Barto. Towards a modern theory of adaptive networks: Expectation and prediction. *Psychological Review*, 88:135–170, 1981.
33. U. Thurm. Eyes specialized for dark responses. In W. Reichardt, editor, *Processing of Optical Data by Organisms and Machines*, pages 236–255. Academic, New York, NY, 1969.
34. N. Toni, P. A. Buchs, I. Nikonenko, C. R. Bron, and D. Muller. LTP promotes formation of multiple spine synapses between a single axon terminal and a dendrite. *Nature*, 402:421–425, 1999.
35. T. Tully. Toward a molecular biology of memory: the light's coming on! *Nature Neuroscience*, 1(7):543–545, November 1998.
36. B. Webb and T. Scutt. A simple latency-dependent spiking-neuron model of cricket phonotaxis. *Biological Cybernetics*, 82(3):247–269, 2000.