

# A system-level brain model of spatial working memory

Alan H. Bond

*Center for Cognitive Phenomics, Neuropsychiatric Institute,  
University of California at Los Angeles, Los Angeles, California 90095*  
email: alan.bond@exso.com

**Abstract** A system-level model of spatial working memory is described, using the author's computer science and logical modeling approach. The mental image which is remembered is located in a lateral parietal area, and is part of a larger distributed representation including object identities and object appearances. The system is driven by plans which are stored in a separate module corresponding to ventral prefrontal and then copied to a planning module corresponding to dorsal prefrontal, where they are executed and sequenced. The system is integrated by an event memory module corresponding to the hippocampus, which gathers event information from most of the modules and constructs a representation of the current mental event and the current mental episode. This information is used by the mental imagery modules to construct a unified representation of the current scene. Mental images are maintained by explicit messages from a currently executed plan. Upon a cue to use a saved mental image for motor output or eye movement, a message from the plan causes the image to be temporarily reinstated allowing it to be used for motor control. The system generates predicted MRI image sequences. We developed an energy estimation function based on the different kinds of information processing being carried out and we graphed energy usage in the main modules over time during a spatial working memory experiment. We also made a study of lesioning the model to produce spatial working memory deficits. We first describe the main logical dependencies among the different brain modules giving a systematic way of finding all the vulnerabilities of the system and how lesions in different modules can interfere with the action of the complete system. The interference in most mechanisms is often catastrophic, preventing any spatial working memory behavior at all. We then describe a more gradual approach in which the goal attenuates, causing the plan to fall below a threshold. This approach allowed us to match the experimental results of Conklin et al. in which schizophrenic and schizotypal subjects could carry out a spatial working memory task with a 0.5 second delay but not when there was a 7 second delay.

# 1 Introduction

In order to more precisely characterize psychiatric disorders such as schizophrenia, bipolar disorder, autism and ADHD, in recent years there has arisen the concept of *cognitive endophenotype* [Bilder et al., 2004]. Endophenotypes are simple, biologically-based aspects of a disorder that may be governed by fewer susceptibility genes [Gottesman and Gould, 2003] [Cannon et al., 2005]. Thus a cognitive endophenotype is a particular cognitive ability that may have deficits linked to certain genes. The disorder of a given particular patient should then be characterized as a set of measures of a basis set of cognitive endophenotypes.

We need to understand these cognitive abilities and their relation to the brain, and we need to understand which underlying processes are shared in common among cognitive functions.

The initial candidates for such a basis set are the well-known cognitive abilities such as verbal long term memory, spatial long term memory, language perception, nonverbal communication and working memory. However the characterization of each type of disorder would be much improved if we could specialize and focus the types of cognitive ability that are measured. Passingham and coworkers have experimentally differentiated between the working memory ability to maintain memories as distinct from the ability to manipulate memories [Rowe et al., 2000]. Glahn et al [Glahn et al., 2003] correlated several different components of spatial working memory such as encoding, maintenance, manipulation, time-tagging of visual spatial information, storage capacity and complex motor response, against genetic predisposition to schizophrenia. They concluded that encoding and storage aspects of spatial working memory may be effective endophenotypic markers for schizophrenia.

In this study, we attempted to analyze and understand the underlying mechanisms involved in one cognitive ability, namely spatial working memory. This has been shown to be clearly impaired in schizophrenia. In order to understand it, we developed a system-level model of spatial working memory.

We used a modeling approach of a modular distributed computational architecture and an abstract logical description of data and control [Bond, 1996] [Bond, 1999] [Bond, 2004a], for which we have also analyzed its correspondence to the cortex [Bond, 2004b].

Modeling spatial working memory involved developing mechanisms for a frontal area

containing a maintenance process, a posterior area containing mental images, and an integration mechanism involving a simple model of episodic memory, corresponding to the hippocampal complex.

We implemented the model as a computer system and studied normal behavior and then abnormal behavior by introducing different types of component deficit. We specifically obtained a match to the work of Conklin et al on schizophrenic and schizotypal subjects. For short delays such as 0.5 seconds they performed normally, but for long delays such as 7 seconds they exhibited a clear deficit.

## 2 Spatial working memory

We can perhaps define spatial working memory by describing the basic experiment. Different experiments in spatial working memory have been reviewed by Curtis and D'Esposito [Curtis and D'Esposito, 2003] and by Rowe et al [Rowe et al., 2002].

The subject first fixates a central fixation point, then an additional image of a small object appears at a certain spatial location in the periphery, then there is a delay with just the fixation point visible, and then a cue is given and the subject moves either their eye gaze or their hand to the spatial location where they think the small object was. Thus, the idea is that the subject has to remember a certain spatial location for a short time of the order of a few seconds.

In our research, we used a basic experimental design from a recent standard paper by Conklin et al [Conklin et al., 2005]. This is diagrammed in Figure 1 and has four steps.

1. fixate - there is a cross at the origin and the subject has to fixate it, duration 2000 milliseconds.
2. note image - there is now also an asterix at a peripheral location, duration 200 milliseconds.
3. delay, maintain image - back to just the cross, duration either 500 milliseconds or 7000 milliseconds.
4. cue, move hand to where the asterix was duration 5000 milliseconds.

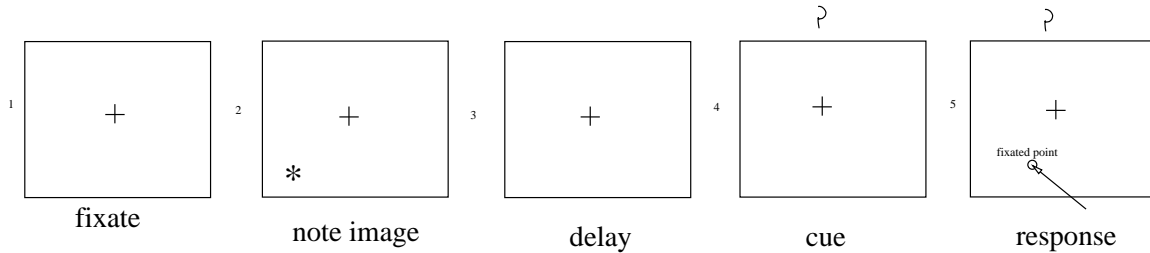


Figure 1: Experiment for testing spatial working memory

### 3 Our modeling approach

**Our general brain modeling approach.** We have developed a computer science approach to the brain [Bond, 1999] [Bond, 2004b] [Bond, 2004a].

In the last few years, we have conducted a series of studies and models concerning problem solving [Bond, 2002b], episodic memory [Bond, 2005c], natural language processing [Bond, 2005a], routinization [Bond, 2005b], and social relationships [Bond, 2002a]. For this project, we have begun integrating all of these mechanisms into a single system which we call our dynamic model.

#### Biological information-processing principles

The basic principles of our design are derived from the biology of the neocortex:

1. Each neural area stores and processes data of given types characteristic of that neural area; data items are of bounded size.
2. To form systems, neural areas are connected in a fixed network with dedicated point-to-point channels.
3. Neural areas are organized as a perception-action hierarchy.
4. Neural areas process data received and/or stored locally by them. There is no central manager or controller.
5. All neural areas have a common execution process, which constructs data items.
6. All neural areas do similar amounts of processing and run at about the same speed.
7. There is data parallelism in communication, storage and processing. Processing within a neural area is highly parallel. Parallel coded data is transmitted, stored, and triggers processing. Processing acts on parallel data to produce parallel data.
8. The data items being transmitted, stored and processed can involve a lot of information; they can be complex.

9. The neural areas act continuously and in parallel.

### **The realization of a system-level brain model using logic programming**

**Modules.** A system-level brain model is a set of parallel modules with fixed interconnectivity similar to the cortex, and where each module corresponds to a brain area and processes only certain kinds of data specific to that module.

**Data items, and their storage and transmission.** We view all data streams and storage as made up of discrete data items which we call *descriptions*. We represent each data item by a logical literal <sup>1</sup> which indicates the meaning of the information contained in the data item. An example data item is `position(adam,300,200,0)` which might mean that the perceived position of a given other agent, identified by the name “adam”, is given by (x,y,z) coordinates (300,200,0). In order to allow for ramping up and attenuation effects, we give every data item an associated strength, which is a real number. Stored data items are ramped up by incoming identical or related data items, and they also attenuate with time, at rates characteristic of the module.

**Processing within a module.** We represent the processing within a module by a set of left-to-right logical rules <sup>2</sup> which are executed in parallel. A rule matches to incoming transmitted data items and to locally stored data items, and generates results which are data items which may be stored locally or transmitted. Rule patterns also have weights, and the strength of a rule instance is the product of the matching data item weights and the rule weights, multiplied by an overall rule weight.

A rule may do some computation which we represent by arithmetic. This should not be more complex than can be expected of a neural net. The results are then filtered competitively depending on the data type. Typically, only the one strongest rule instance is allowed to “express itself”, by sending its constructed data items to other modules and/or to be stored locally. In some cases however all the computed data is allowed through.

**Uniform process.** The uniform process of the cortex is then the mechanism for storage and transmission of data and the mechanism for execution of rules.

**Perception-action hierarchy.** Modules are organized as a *perception-action hierarchy*,

---

<sup>1</sup>A logical literal is an atomic assertion of the form  $p(t_1, t_2, \dots)$  where  $p$  is a predicate and the  $t_i$ s are terms

<sup>2</sup>Our logical rules are clauses, i.e. they have no nesting, but are made up of a disjunction of possibly negated literals

which is an abstraction hierarchy with a fixed number of levels of abstraction.

The perception hierarchy receives sensory data items at the bottom and derives higher level descriptions to form a percept. The action hierarchy generates more and more detailed descriptions of action, that is, it elaborates the plan to the point where motor actions are generated at the bottom of the action hierarchy.

Figure 2 shows the correspondence of our initial model to the primate neocortex. Note that the goal module corresponds to an area on the inner (medial) surface, namely the anterior cingulate.

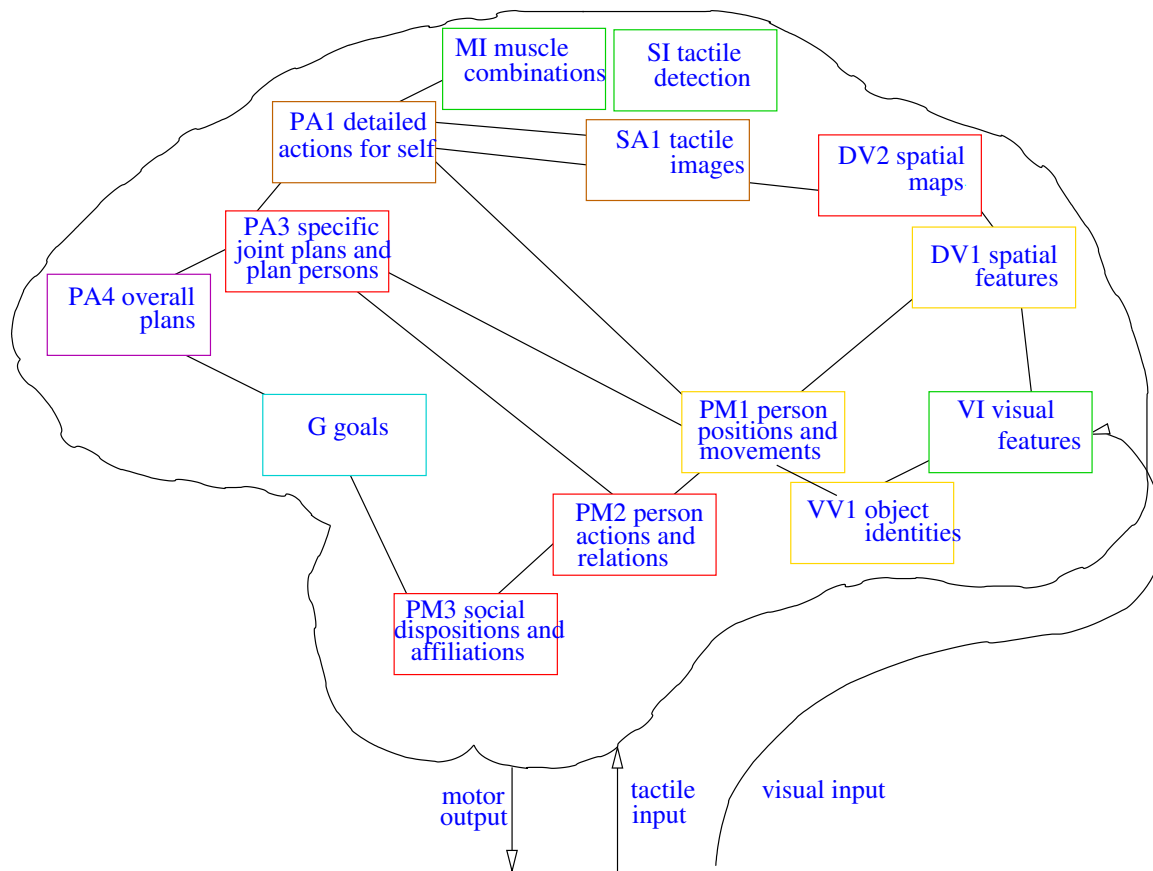


Figure 2: Our system model shown in correspondence with the neocortex

**Differences between our approach and others.** Our approach differs from present-day cognitive models such as ACT-R [Anderson, 1993], SOAR [John Laird and Paul Rosenbloom and Allen Newell, 1986], 4CAP [Haarman et al., 1997] and Kintsch’s comprehension model [Kintsch, 1998], in that:

1. It is a parallel model, with multiple modules running in parallel.
2. Its basic unit of data is the chunk, or structured packet of information, with chunks being constructed, stored and transmitted by operations of the model. Chunks are accessed associatively from memories in modules.
3. It corresponds to brain architecture, with modules corresponding to brain areas, connectivity corresponding to the connectivity among corresponding brain areas, and data types corresponding to those processed in corresponding brain areas.
4. The dynamics consists of distributed and coordinated sets of processes, for real time control, for plan elaboration and for episodic memory creation and use.
5. The computational method is based on logic programming [Kowlaski, 1974] [Hogger, 1991] [Sterling and Shapiro, 1994], and there is an underlying theory of these models which provides formal semantics and completeness and convergence properties [VanEmden and Kowalski, 1976] [Lloyd, 1987].

Incidentally, the concept of a “rule-based model” is not well founded since all known formulations of computation, including functional, automata theoretic, and logical, can be put in the form of rules, and small changes in the form of rules may lead to large changes in computational properties. Neither is a logic programming model “symbolic” since not all models of these logics contain symbols as individuals, and indeed most do not.

Our approach differs from neural network approaches [Hertz et al., 1991], such as those of Grossberg [Grossberg, 2003], Edelman [Edeleman, 1987] and Rolls [Rolls and Treves, 1998], in using an abstract method of description, so that information is represented by abstract chunks and processing by rules which describe the processing of chunks. It also differs from the abstract neural models of Cohen and Braver [Braver et al., 1999] in using complex data items and matching, and in using complex computation and control within each module.

Our approach is complementary to neural network approaches, and it should be possible, for a given abstract model, to construct corresponding neural network models.

## 4 Our approach to the design of a spatial working memory model

Our model is implemented as a set of intercommunicating brain modules that run in parallel.

Our solution to the extension of our model has involved:

- (i) generalizing our planning module to use learned plans represented as data items, so that rules competitively reconstruct data items in response to their current situation,
- (ii) adding a mental imagery module which stores and represents mental images in terms of image elements and their spatial relationships, and
- (iii) adding a module corresponding to the hippocampal complex, and which receives data from cortical areas and constructs a representation of the current mental event.

Figure 3 diagrams the design of the basic spatial working memory model.

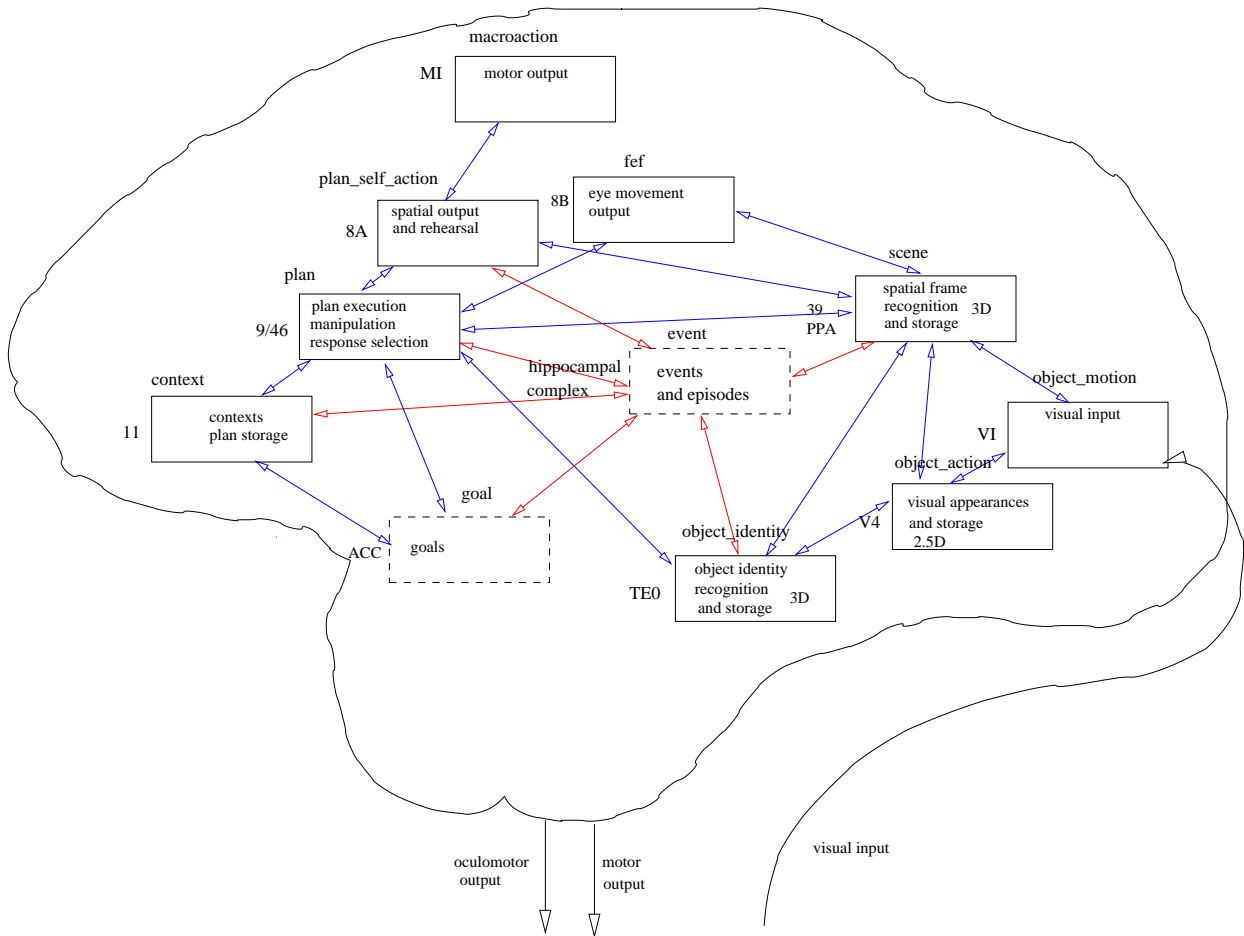


Figure 3: Outline diagram of an initial model

**Mental images.** There is by now considerable experimental evidence for a network of modules in the human brain which process visual information [Felleman and Essen, 1991] [Farah, 2000]. These include modules which process retinal image forms, which process 2.5D representations [Marr, 1982], which process 3D representations [Shepard and Cooper, 1982], which process object representations [Ungerleider and Mishkin, 1982] and which process spatial layouts [Epstein et al., 2001]. Further, these modules correspond to conscious attention [Kanwisher, 2001] [Grill-Spector and Kanwisher, 2005].

A distributed cognitive model of mental imagery was developed by Kosslyn [Kosslyn, 1981] [Kosslyn et al., 1984] [Kosslyn, 1994] which did not correspond to neuroanatomical areas but did define functionalities that may be present in an imaging store, as well as a propositional representation in a semantic store. Functions involved in imaging included image construction and image gathering, image processing functions such as translation, zooming and so on, and image attention functions.

We decided to use a simple distributed representation for mental images, for future development, so there are four modules altogether. The first two we already have been using before, `object_motion` is the basic input module for visual information and `object_action` computes basic spatial relations among perceived visual objects. Then we added two new modules, one for object identity (“what”) called `object_identity`, corresponding to the ventral temporal areas, and one for spatial layout (“where”) called `scene`, corresponding to lower parietal areas.

In the future, we will explore the use of a distributed representation of a mental image, so that different aspects of the image would be computed and stored in different visual modules. The relation among modules should allow a module to ask another for data on a demand basis, and we think this will have to use goal expressions, so different aspects of the image would be computed and emphasized depending on current demands.

However, for the current project, we developed a representation of a mental image that was only computed and stored in one module which we took to be the `scene` module.

We used three kinds of data that are computed by the `scene` module, starting from input data items of the following types: `object_position(Object,X,Y,Z)`, `object_size(Object,[S1,S2,S3])`, `object_type(Object,Type)` - all from the `object_motion` module, `object_identity(Object,Identity)` - from `object_identity` module, together with information for forming a key, such as `current_episode(Current_episode)` - from the episodic

memory module, it computes and stores:

- (i) scene\_objects, one for each object in the mental image, scene\_object(Key,Object)
  - (ii) object\_scene\_positions, where each scene object is, in the image, object\_scene\_position(Key,Object,X,Y,Z)
- and (iii) scene\_lists which give a complete list of all objects occurring in a given image: scene\_list(Key,Sorted\_object\_list).

A set of descriptions of these three types all with the same key forms a mental image. The component descriptions of a given mental image can be accessed together from the associative store.

The scene module can hold more than one mental image, each with a different key; for the most recently perceived image we took to be the episode key which it receives from the event module. At any one time, there is a unique current episode key which is computed by the event module and sent to most modules.

**Plans.** Plans are represented in a form which should be derivable from experience of action, learning by doing. This form allows a mixture of short sequences (3, 4 or 5 steps) and embedding of one subplan within another. So this corresponds to the form of episodic memory in organizing experiences of sequences of mental states.

The form of plans is such that it could be learned either by learning by doing, or by being instructed. This was not implemented for this project.

Plans are stored in the context module, corresponding to ventral prefrontal, and evoked by the existence of goals communicated from other modules.

Plan steps are a bit like rules but they are data, and they are executed by executing rules in the planning module, corresponding to dorsal prefrontal. The actual rules in the planning module are general rules which execute plan steps.

We call plans contexts. A context is then a set of context descriptions. There is a head context which is triggered by a goal and determines the sequence of steps, and then there are four contexts representing these steps.

We show below the plan we used for the basic spatial working memory experiment, Figure 4 shows the structure of the plan into a head and the set of plan steps.

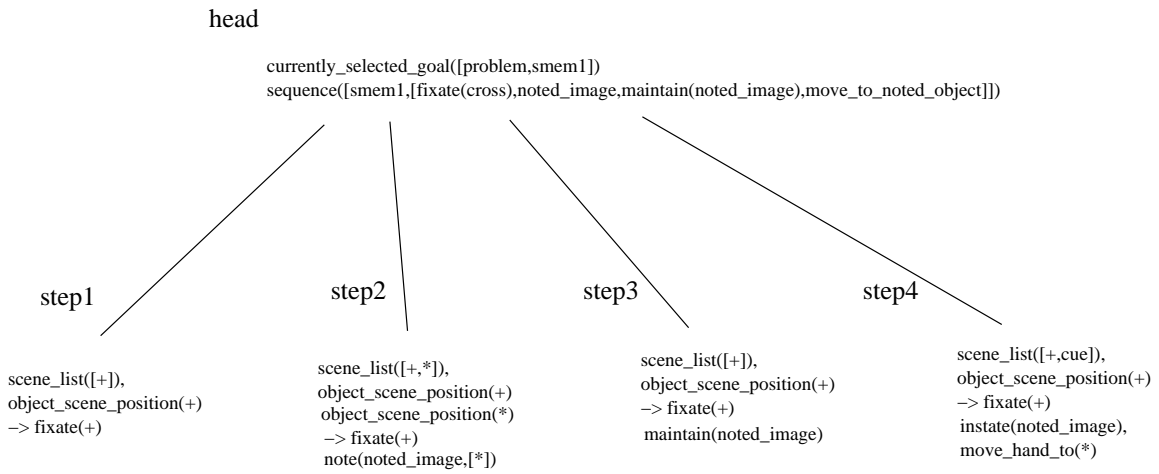


Figure 4: Diagram of structure of plan for spatial working memory

Figure 5 shows the form of the plan as stored in associative memory, as five data items. Percent characters indicate explanatory comments, the actual data item is a single logical term of the form: `context(key,if_part,then_part,provided_part,weight_part)`.

```

% prestored smem context head
:- assert((prestored_context([smem1,Current],
% If part
  (currently_selected_goal(_,[problem,smem1])),
% Then part
  [sequence(_,[smem1,[fixate(cross),noted_image,
    maintain(noted_image),move_hand_to_noted_object]])]),
% Provided part
  (true),
% Weight part
  1.0
  )).

% prestored smem context step 1
:- assert((prestored_context([fixate(cross),smem1],
% If part
  (scene_list(_,[Current_episode,[board,cross,hand]]),
    object_scene_position(_,[Current_episode,cross,X,Y,Z])),
% Then part
  [eye_movement(_,[fixate(cross)])]),
% Provided part
  (true),
% Weight part
  1.0
  )).

% prestored smem context step 2
:- assert((prestored_context([noted_image,smem1],
% If part
  (scene_list(_,[Current_episode,[asterix,board,cross,hand]]),
    object_scene_position(_,[Current_episode,cross,X,Y,Z]),
    object_scene_position(_,[Current_episode,asterix,XA,YA,ZA])),
% Then part
  [eye_movement(_,[fixate(cross)]),
    vision(_,[note(noted_image,[asterix])])]),
% Provided part
  (true),
% Weight part
  1.0
  )).

% prestored smem context step 3
:- assert((prestored_context([maintain(noted_image),smem1],
% If part
  (scene_list(_,[Current_episode,[board,cross,hand]]),
    object_scene_position(_,[Current_episode,cross,X,Y,Z])),
% Then part
  [eye_movement(_,[fixate(cross)]),
    vision(_,[maintain(noted_image)])]),
% Provided part
  (true),
% Weight part
  1.0
  )).

% prestored smem context step 4
:- assert((prestored_context([move_hand_to_noted_object,smem1],
% If part
  (scene_list(_,[Current_episode,[board,cross,cue,hand]]),
    object_scene_position(_,[Current_episode,cross,X,Y,Z])),
% Then part
  [eye_movement(_,[fixate(cross)]),
    vision(_,[instate(noted_image)]),
    plan_self_action(_,[move_hand_to_noted_object(noted_image,asterix)])]),
% Provided part
  (true),
% Weight part
  1.0
  )).

```

Figure 5: Plan for spatial working memory as represented in associative memory

## The execution of plans

In execution, steps are selected based on the current perceived image, which is taken from scene, as well as their order in the sequence. A step gets activated when its conditions are true and provided it is next in the current sequence.

A context step has a key which describes it and when it is being executed this key is asserted. A context key actually refers to the step and to its current parent, for example [noted\_image,smem1] is the key for plan step noted\_image evoked from the head with key smem1.

During the execution of a context, there will usually be two such keys in existence, one for the head, or parent, and one for the step being executed. Thus, during the second step of the plan, the cecs, currently evoked contexts, are [smem1,top] and [noted\_image,smem1], and during the thrid step they are [smem1,top] and [maintain(noted\_image),smem1].

Figure 6 shows a general case of a plan made up of several nested and sequenced plan steps. It shows the activation of step1212 and the control memory at that time consisting of a set of cec expressions.

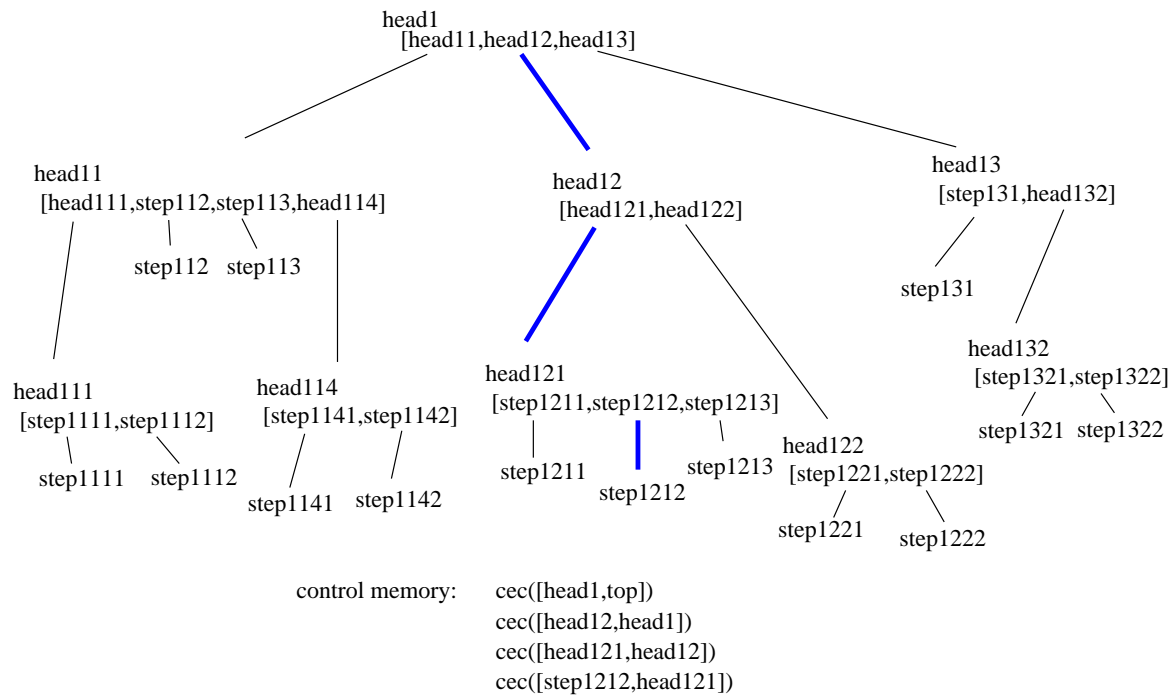


Figure 6: The memory of execution of plans

The cecs and the current sequence give us the information to maintain the orderly execution of plans. These data items form a short term memory for the execution of plans. This memory is of course in the store of the plan module.

Incidentally, cecs are continuously re-evoked every cycle, independently and in parallel, and if any cec is not evoked then this can cause termination of the current step at any level. This allows a plan step to succeed or fail at any level. If it terminates at a higher level, all the cecs below will not be re-evoked, since they can only be evoked if their parent exists. If the last step in a sequence terminates then this leaves just the head of the sequence, which then itself terminates.

### **Episodic memory**

We chose to use a very simple version of episodic memory, corresponding to the hippocampal complex, and following our published ideas on episodic memory [Bond, 2005c]. The main idea is to form representations of instantaneous mental events and then to form episodes in nested groups of no more than 4 events or episodes. The system computes an associative key for each event or episode that allows its unique retrieval from the store of the episodic memory module. The contents of this store is usually called a *cognitive map*.

In representing the current episode in episodic memory, we are currently taking the current episode key to be simply the set of these active cecs, so this is a path through the context nesting to the current context being executed. Thus in the above example the current episode key is [[smem1,top], [noted\_image,smem1]].

### **Noting, maintaining and using mental images.**

As we have noted, Kosslyn's model has an operation of maintaining a given mental image in the image store. In our model, plan steps send messages to the scene module to perform certain operations on mental images stored there. Basically, we need to make a note of an image of the scene to be remembered during the note phase, then we need to maintain this image and stop it from attenuating away during the maintain phase and then we need to reinstate the remembered image during the action upon cue phase.

We concluded that under normal circumstances there was a flow of continuously changing perceived images through the visual system. However, some salient images, or images that are explicitly noted, will have constructed more precise, detailed or complete representations which will be labeled with an associative key. Such a key enables the system

to store a set of different descriptions of different aspects of a given scene, and to be able to retrieve them using the associative key that all these related descriptions contain. In the 1981 version of Kosslyn's model he has a LOAD operation which basically grabs a new image from the visual system, so this is related to our own idea. Then there can be several mental images stored in the mental imagery module, as well as the ongoing instantaneously perceived visual image. The stored mental images will attenuate fairly rapidly. A given image can be prevented from disappearing by maintaining it with constant retrieval and reconstruction. Also of course images can be captured as components of episodic memory and then stored in long term memory, but this will only occur for a small number of possible images and may take longer.

Every mental image that is constructed has an associative key, we call such constructed images *scenes*. We discussed their detailed representation above. We arbitrarily chose the current episode key as a key for the current mental image, i.e., the one derived from the current percept. Normally, as the scene changes so does the mental image corresponding to the percept, and it is named with the new current episode key. The previous image will attenuate fairly fast and disappear; with the current settings this takes about 3 cycles.

Given this approach, we were able to precisely define noting, maintaining and reinstating of images:

- (i) Noting the current mental image creates a new scene which is labeled by a name sent from the planning module, instead of a name derived from the current episode key.
- (ii) Maintaining a named mental image consists of simply executing a rule which recognizes and reconstructs its components.
- (iii) Instating a named mental image. When we come to act using the stored mental image, we instate the noted image to be the current mental image and its spatial properties are used by the planning and motor hierarchy to execute the desired motor action. This temporarily deemphasizes the percept which is continuously being refreshed from the input visual stream.

**Eye movement.** We have a frontal eye field module which sends a saccade operation to the world. It does this on receipt of fixate messages from the current plan step.

**Routine action.** We currently do not model the basal ganglia and so do not have any routine action, all action is generated from stored plans.

## 5 The spatial working memory model as implemented

**The current system.** All of the above mechanisms have been designed and implemented and the system will successfully carry out the basic spatial working memory experiment.

**Implementation.** The system was programmed in Prolog and the BAD language. The BAD language and manual can be found at <http://www.exso.com/bad.html>. The system uses Sicstus Prolog and runs on Linux, Red Hat 7.2. Sicstus Prolog is an industrial strength implementation of ISO Prolog developed by the Swedish Institute of Computer Science. The system consists of the brain model and also a world program representing the external environment of the brain.

**The logical structure of the model.** Figure 7 outlines the model as implemented, showing the main descriptions and their logical dependencies. This diagram does not show the detailed temporal dynamics of how the model operates, it is a description of all the data types involved and how they are derived from each other.

**The experiment.** The experiment is specified by a temporal sequence of environment components given in a file. Figure 8 shows the time axis of the experiment and the main time points of interest.

**The action of the system during the experiment.** As can be read off from the listing of the plan components given previously in Figure 5, there are four phases of the experiment and the action of the system is driven from the current state of the environment. When the environment changes, the system perceives the new scene and this results in a new plan step being selected.

Initially, the plan is evoked from the existence of the currently selected goal which is to carry out the spatial working memory experiment. The plan consists of representations of the head and the four steps.

1. Fixating. The first step is activated by a scene with just a cross. It causes a fixate expression to be generated which is sent to the eye movement module, which sends a motor command to the environment to move the eyes.
2. Noting the image. The second step is activated because the scene has a cross and an asterix. It can run as it is next in the plan sequence. It constructs a note expression which is sent to the scene module which constructs a scene representation from the current



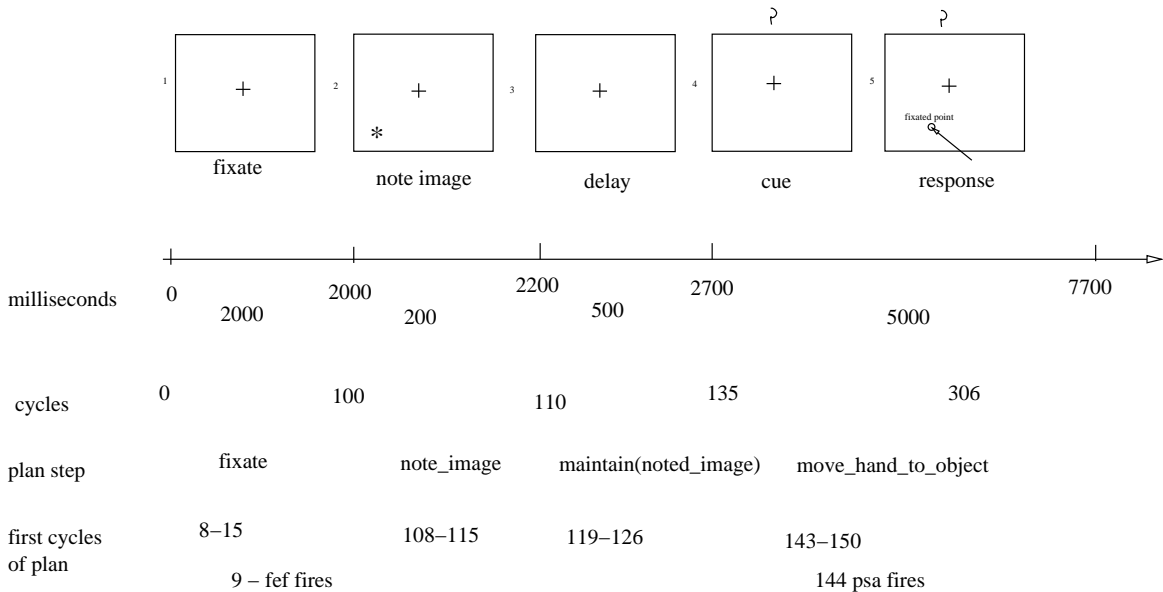


Figure 8: The time axis and main events of interest

one whose spatial coordinates are sent to the premotor module, instead of those derived from the current percept. The premotor module generates a motor command which is sent to the environment to move the hand.

**Visualization.** A VRML visualization is automatically output by the system at each cycle. It diagrams the state of the brain model at that cycle. There is also a VRML visualization of the environment produced each cycle.

**Predicted imaging files.** The program allows one to write to a file all the activation values for all modules for all cycles. We discuss in the next section the energy measure that we developed for the activation values used in visualization.

We used the afni imaging software. You can then write out files in a standard format suitable for input to afni, for particular times as desired. We wrote for example afni files for key cycles corresponding to the five phases of the experiment, at cycles 10 (fixate phase), 111 (note image phase), 126 (maintain noted image phase), 146 (move hand to object phase).

We show afni images for each of the four sample times in Figures 9, 10, 11 and 12, where we have chosen samples corresponding to sagittal, axial and coronal views for each time, and for two different geometric positions, we use Talairach coordinates:

- (i) sagittal 30, axial 10 and coronal -43, which is the plan module or Brodmann 10 and 46
- (ii) sagittal 44, axial 36 and coronal 41, which is the scene module or Brodmann 40.

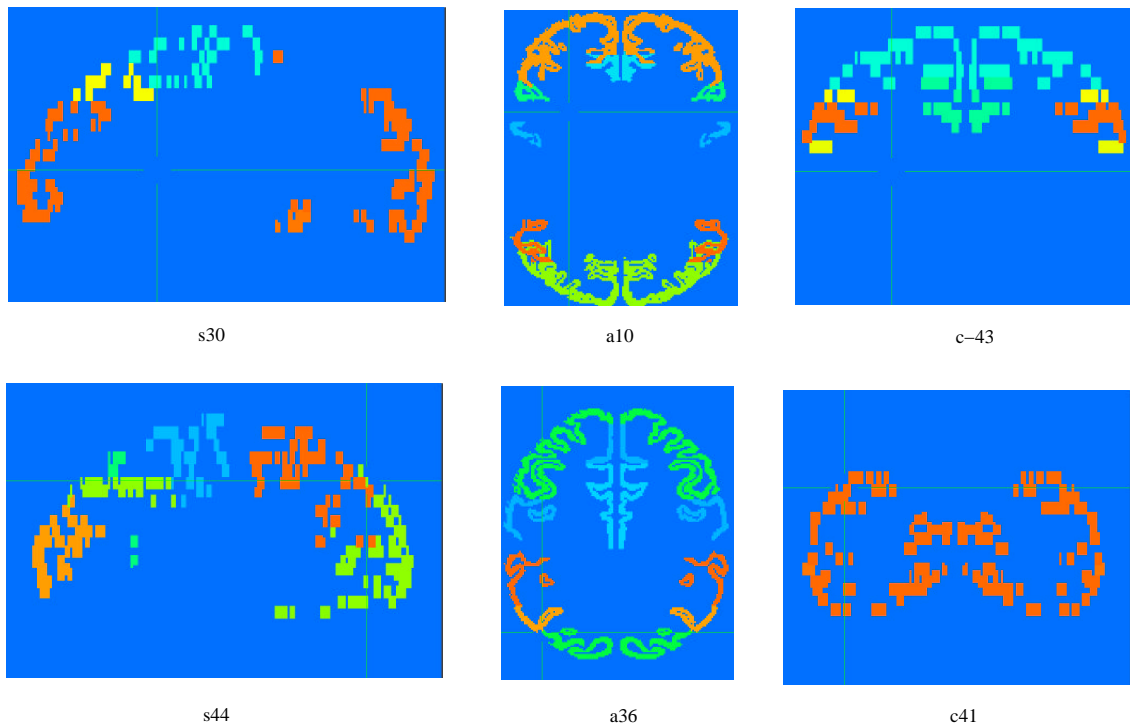


Figure 9: Predicted imaging files for cycle 10, in the fixate phase

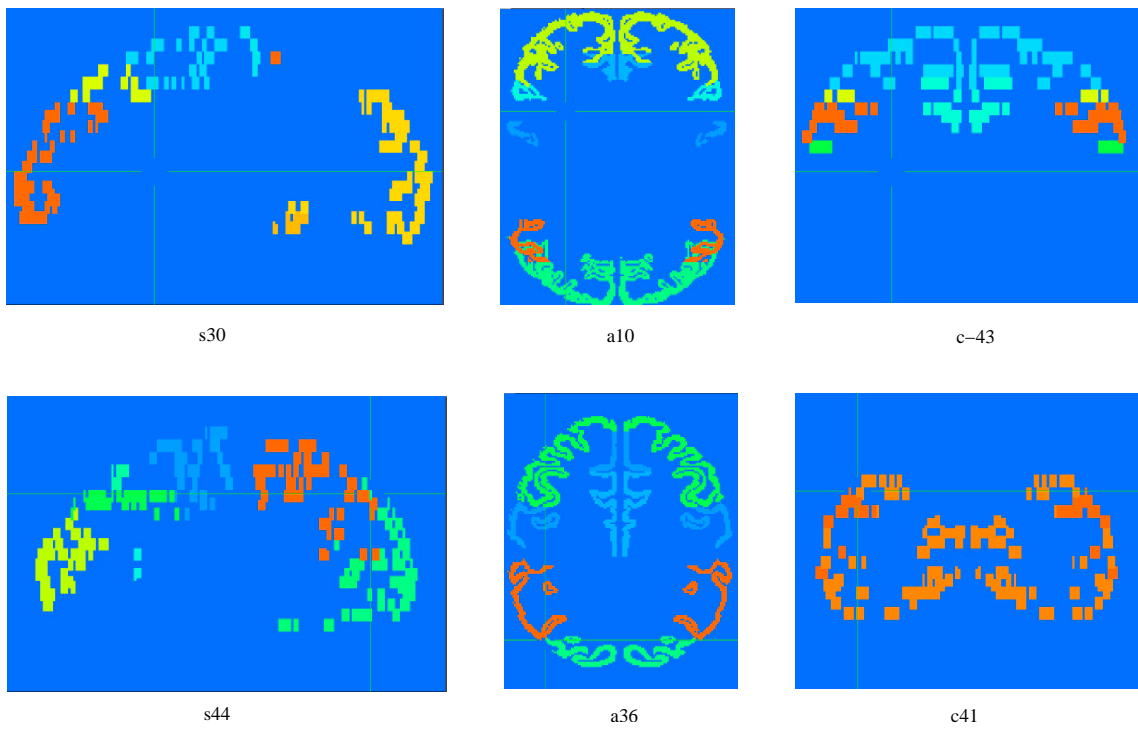


Figure 10: Predicted imaging files for cycle 111, in the note image phase

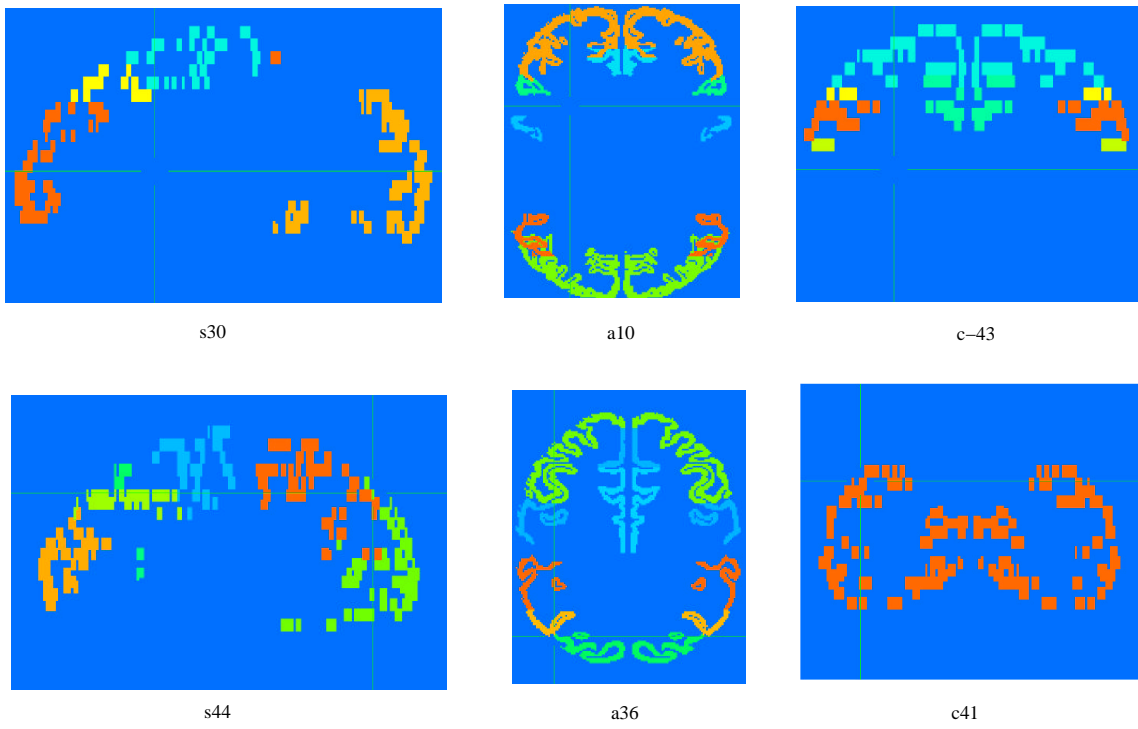


Figure 11: Predicted imaging files for cycle 126, in the maintain noted image phase

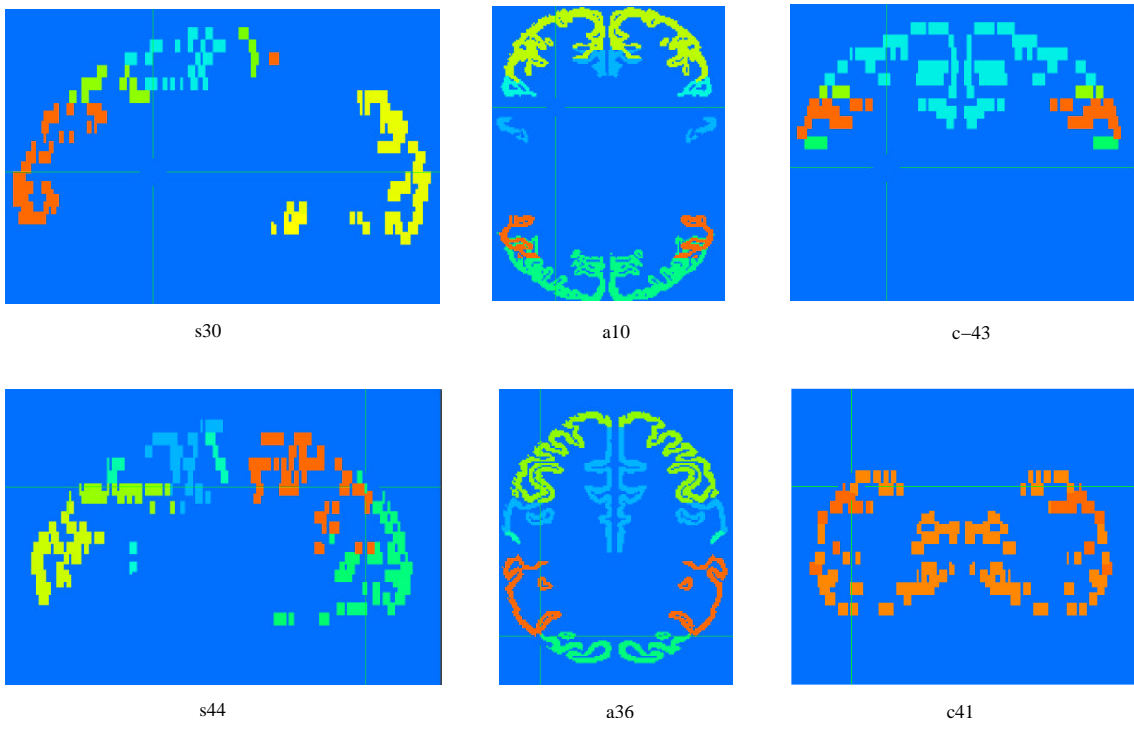


Figure 12: Predicted imaging files for cycle 146, in the move hand to object phase

**Trace files.** A trace file for each experiment is produced. The program can generate many different of alphanumeric character output, depending on the settings of the many print switches available. The simplest form is a print out of each rule as it is executed. Here is a typical rule execution record for one rule:

```
rule scene_1 has fired
Instantiated LHS = object_position([1.0,0.5818836516,1.0,129,[]],_48735,[hand,20.0,20.0,0.0]),object_size([1.0,0.5155,1.0,129,[]],_48720,[hand,[5.0,5.0,1.0]]),object_type([1.0,0.5155,1.0,129,[]],_48703,[hand,hand]),object_identity([1.0,0.6574404203415274,1.0,129,[]],_48692,[hand,hand]),current_episode([0.9,0.8630147432325322,0.9,129,[]],_48684,[[[smem1,top],[maintain(noted_image),smem1]])]
Instantiated RHS = [scene_object([0.9800000000000001,0.6266677630348119,0.9800000000000001,129,[]],_48676,[[[smem1,top],[maintain(noted_image),smem1]],hand]),object_scene_position([0.9800000000000001,0.6266677630348119,0.9800000000000001,129,[]],_48666,[[[smem1,top],[maintain(noted_image),smem1]],hand,20.0,20.0,0.0])]
Wa = 0.6266677630348119
Confirm list = [confirm(alan,object_action,object_position([1.0,0.5818836516,1.0,129,[]],_48735,[hand,20.0,20.0,0.0])),confirm(alan,object_motion,object_size([1.0,0.5155,1.0,129,[]],_48720,[hand,[5.0,5.0,1.0]])),confirm(alan,object_motion,object_type([1.0,0.5155,1.0,129,[]],_48703,[hand,hand])),confirm(alan,object_identity,object_identity([1.0,0.657440420341274,1.0,129,[]],_48692,[hand,hand])),confirm(alan,event,current_episode([0.9,0.8630147432325322,0.9,129,[]],_48684,[[[smem1,top],[maintain(noted_image),smem1]])]))]
```

During the first experiment, which lasts about 6 seconds, something like 15,000 rule firings occur in the model.

## 6 The development of an energy measure

We developed an approach to the issue of estimating the energy that is consumed by the brain when it carries out various kinds of process upon information. We are mainly interested in developing estimates for the corresponding processes in our system-level model of the brain. Our approach is to discuss the issues and measures purely in terms of information, without considering how this processing is implemented or coded in any natural mechanism or modeling approach.

We developed a scheme which considers eight kinds of information process, namely for the energy consumed in:

- (i) accessing stored memories, both short and long term,
- (ii) combining these data items,

- (iii) the amount of iteration or repetition of combination operations,
- (iv) the generation of products of these operations,
- (v) the resolution of competition among resulting products,
- (vi) the storage of products into store, both from local processing within the same store and
- (vii) from external processing in other more remote stores.
- (viii) the transmission of data from one store to another.

All of these processes are dependent on the number and type of information processing events and also the intensity or strength of the data items involved.

**The energy estimation method as implemented.** We calculate the energy expenditure estimates usually starting from a set of contributions. This is a list, one item for each information processing event, of the form [type,W] where type is identical, same, similar or novel, and W is the weight of the element involved, usually a literal. Two literals  $\text{pred1}(W1,C1,Terms1)$  and  $\text{pred2}(W2,C2,Terms2)$  are rated as:

- (i) identical if  $\text{pred1} == \text{pred2}$ ,  $W1 == W2$ , and  $Terms1 == Terms2$ ,
- (ii) same if  $\text{pred1} == \text{pred2}$ ,  $W1 \backslash == W2$ , and  $Terms1 == Terms2$ , and
- (iii) similar if  $\text{pred1} == \text{pred2}$ ,  $W1 \backslash == W2$ , and  $Terms1 \backslash == Terms2$ .

We are interested in measuring the energy consumed by each information process during the cycle, and also in discounting energy consumption measures for information processes that are similar to those in the previous cycles. Initially, we are only using the immediately previous cycle for comparison. We calculate energy from eight different types of information process.

1. accessing the store, how the inputs to rules,  $I\_contribution$ , the left hand sides of rules, are changing from the last cycle.
2. amount of combination of results of access,  $R\_contribution$ , how the rules firing are different from the last cycle.
3. amount of executive iteration,  $E\_contribution$ , the number of iterations to rule quiescence.
4. competition,  $C\_contribution$ , the degree of filtering as a ratio of numbers of rule instances used.
5. the generation of products,  $P\_contribution$ , the right hand sides of rules.

6. updating store, `U_int_contribution` and `U_ext_contribution`, the items stored during the cycle, from internal sources, i.e. rules within the module, and from external sources, i.e. items received from other modules, and how they are different from or the same as items is already in the store.
7. transmission, `T_contribution`, the number of items transmitted to other modules during the cycle.

The various components within individual types of measure were combined using parameters based on common sense. Then the set of eight individual measures was then multiplied by overall coefficients, set by statements of the form `contribution_coefficients(M,Mod,Type,Coeffs)`, and summed to give a total numerical value which is the estimate of the energy consumed during the cycle. We initially assigned values to these coefficients by a common sense argument, however there exists the possibility of determining them by fitting to experimental data.

The actual values we are currently using are as follows. The four values are for access or storage of identical, same, similar, novel items respectively, and a set of items takes the value of its highest valued member. For the basic types of operation: accessing stored values `[0.5,1.0,4.0,8.0]`, for combining values: `new_rule,[4.0]`, `old_rule,[1.0]`, `rule,[1.0,1.0]`, `lhs_rule,[0.5,1.0,4.0,8.0]`, `rhs_rule,[0.5,1.0,4.0,8.0]`, and for updating storage: `update_int,[0.5,1.0,4.0,8.0]`, `update_ext,[0.5,1.0,4.0,8.0]`.

The top level combination of contributions used coefficients: inputs 2.0, results 1.0, executive iteration 1.0, competition 1.0, updating store, from internally generated data 2.0, from externally received data 2.0, transmission 1.0.

**Accessing the store.** It takes energy to access the store to find an item that matches a pattern, or left hand side literal of a rule. In addition there may be more than one item matching one left hand side literal. Thus, we want to count an energy contribution for each accessed item. There is no consideration in one such access if similarity or otherwise between the pattern and the item, however the item could be more or less complex, and it will have a strength. However, we also want to discount the contribution if it was accessed during the last cycle. We keep a list of all the literals that were accessed in the last cycle, from all rule instantiations, and we compare the currently accessed literal with this list. For each currently accessed literal, we find whether it is identical to some previous literal, the same, similar, or novel, i.e., if there was no similar literal accessed in the previous cycle.

**Combination of accessed values.** We interpret this as a measure of the complexity of a rule, but more importantly whether a rule had fired last time. This measure is only for the additional effort in combination, after access of the store has been carried out. Initially, we took this as simply a measure of whether a rule instance of the same rule had occurred last time or not. We have not yet investigated rule instances that are identical, similar or the same.

**Iteration of processing.** We interpret this as concerning the repetition of rule firing until a steady state where further repetition makes no difference. We took this measure as based simply on the number of repetitions. We believe this will be a fairly small contribution, since in the brain the convergence to equilibrium is probably achieved in an efficient manner and not by anything corresponding to explicit repetition.

**Competition.** This concerns the competition of rules and their products, mainly the information processing required to make the decisions involved. In our model, we take raw rule products and subject them to filtering whereby certain sets of rules and certain sets of data items are compared and some number, usually one, is/are selected to continue processing. We used a simple measure of the ratio of the number of rules before and after filtering.

**Generation of products.** We did not have a separate measure for this, but included it in the energy involved in storage of products.

**Updating of the store.** We saw this as an important and potentially large measure. We measured whether the literal to be stored was identical, same, similar or novel compared to the existing stored members. We also calculated separately for updates of items generated within the same module, and for items generated in another module and transmitted to the module and stored.

**Transmission.** We had a small contribution for transmission of items from one module to another. This may involve energy as the axons are long, and there may be some filtering and thresholding involved. We took as a measure simply the number of items transmitted from the module during the current cycle. We did not compare this with the transmission activity of the previous cycle.

**Determining the values of the coefficients.** So far we have taken the values of coefficients from a common sense theoretical argument; we have yet not tried to determine them by fitting the estimated energy values to observed experimental imaging data.

**Results for the time course of energy consumption.** Figure 13 shows the time course of energy consumption by the plan module (red), the scene module (blue), the context module (purple) and the goal module (green).

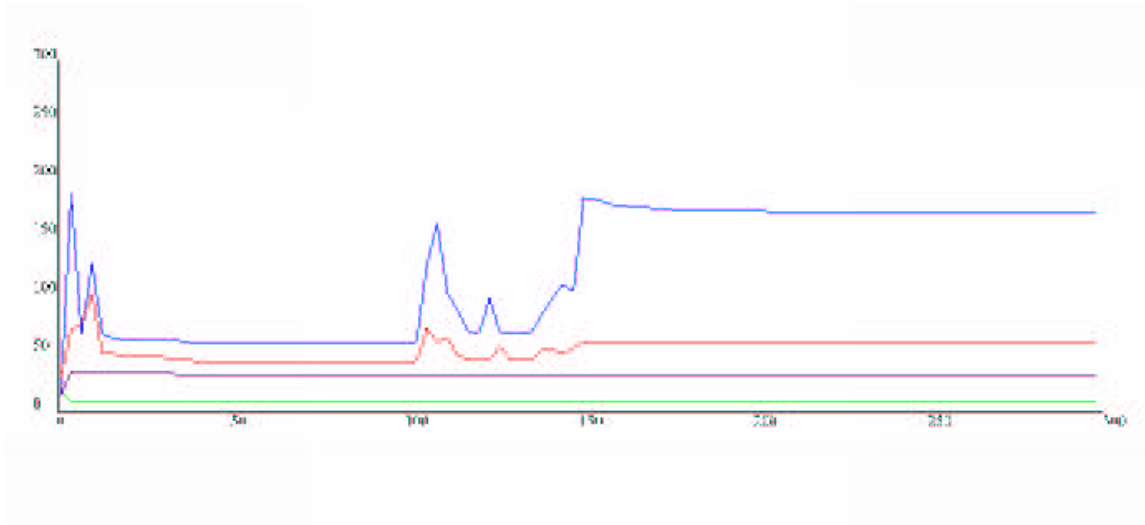


Figure 13: Time course of energy consumption, by the plan module (red), the scene module (blue), the context module (purple) and the goal module (green)

We can see the spurts in energy at the main transition points between phases, at cycles 10, 111, 126 and 146. We also see the imaging system staying active as it visually tracks the movement of the hand, after which the system falls into a rest state.

## 7 Modeling the spatial working memory deficit in schizophrenia

**Systematic analysis of the effect of different lesions.** We investigated in what ways the system could be compromised. First, by examining the dependencies of one data type on others, we can determine the effects of weakening each component of the model. These dependencies are given in Figure 7.

Thus, for the plan to continue working correctly it needs `scene_list` and `object_scene_position` data from `scene`, the `currently_selected_goal` from `goal`, and `context_selected` from `context`, and in addition it has `cec` (`currently_evoked_context`) information which is maintained within the plan module. Thus, if any one of these kinds of data are not available then the plan will not be executed and the system will fail to carry out the experiment successfully.

The scene module will not function properly unless it receives the object information from the perceptual stream, `object_identity` from the object identity module, the current episode key from the event module, and the vision messages from the plan module. It also needs to maintain `scene_object` information within the scene module.

The event module will not function properly unless it receives plan step information from the plan module.

We showed that we can compromise the functioning of the system by increasing the attenuation of the `cec` descriptor in the planning module. This keeps track of the current plan step being executed and if it attenuates too fast then the system cannot carry out the spatial working memory task.

**Issues in achieving better control in the model.** However, we would prefer to be able to compromise the system more subtly. We have shown that the system will carry out successfully the spatial working memory task with a much increased delay, whereas, from the work of Conklin et al [Conklin et al., 2005], schizophrenics have more difficulty with such tasks but can successfully carry out the task with the smaller delay.

It is not obvious how to achieve this with our system. We have started to make several changes in how strengths of descriptions are handled. We designed some modifications, implemented them, and got the system working again, doing the experiment with short and with long delays, however it is not yet clear how to degrade the system in a controlled manner. What we introduced was the idea of long term and short term memory descriptions, with long term memories attenuating to a quiescent state, whereas short term memories are erased when reaching noise level. In addition, we implemented a priming mechanism, so that descriptors occurring in the left hand sides of rules which fire are boosted in strength. It is not yet clear how to use this, as it is similar to reducing the attenuation rate, however this can lead to memories staying too strong and interfering with the system's transitioning into new states. There is a related issue with the bihemispheric version of the system, where data exchanged between the hemispheres can lead to data which do not die out.

The system was originally designed to be able to do complex tasks such as problem solving, social interaction and natural language processing, and no consideration was given to this kind of fine tuning on simple tasks. The system acts continuously, with rules firing continuously, even though most of this is in some sense not really needed. The changes we are now introducing are an attempt to have a more subtle control over rule firing. We would like to investigate the detection of rule repetitions across cycles and to reduce the number of rule firings to something more efficient, by the system checking the firing of rules rather than expending energy in actually firing them.

There are also a lot of issues to do with trying to find the amount of effort and energy that the brain would be expending on carrying out the various operations of the model. We need this in order to get better activation values, and also we think we can from this develop an idea of how much real time would be used and hence compute an EEG-like signal from the system. These issues lead to the more comprehensive idea of understanding the expenditure of resources by the brain in carrying out different kinds of information processing operation.

In general, these new demands on the model are making us investigate these more detailed issues. This means that we will be able to make more use of the corresponding experimental data as a guide to modifying the model. For example, experimental data on the temporal pattern of activation of ventral prefrontal areas during the task would be very interesting, since it might indicate whether knowledge is passed to dorsal prefrontal for use or whether it is used in situ. There could also be a similar issue with plan

sequencing information, whether it is passed to SMA from dorsal or ventral prefrontal or some other arrangement.

**Graded deficits due to attenuation.** We were able to obtain a graded deficit in spatial working memory performance by allowing the goal to attenuate faster in time. For normal performance the attenuation rate for goals was set to a very small amount, corresponding to extinction in 1000 cycles, whereas for the pathological case we set it to attenuate faster, corresponding to extinction in about 400 cycles. This resulted in the pathological case being able to carry out the task with a 0.5 second delay but not for a 7 second delay. Figure 14 shows the energy curves for the 0.5 second delay case, with the normal and pathological systems behaving very similarly. Figure 15 show the energy curves in the 7 second delay case. The graph shows the reduction of the goal and context energy, and it also show that the move hand operation did not occur. This approach reduces the activation level of the frontal planning area, which agrees with some experimental findings.

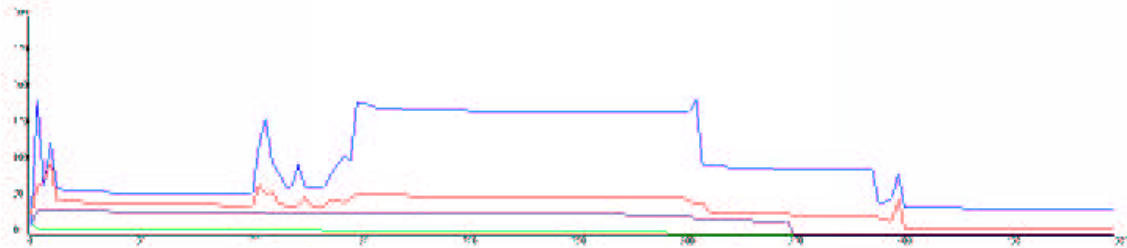
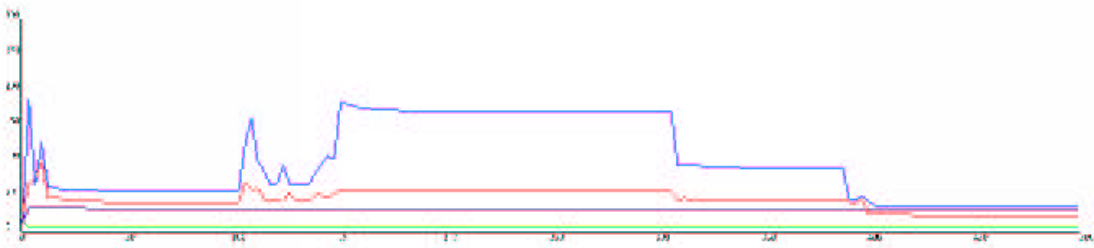


Figure 14: Time course for 0.5 second delay experiment, (i) normal case and (ii) pathological case

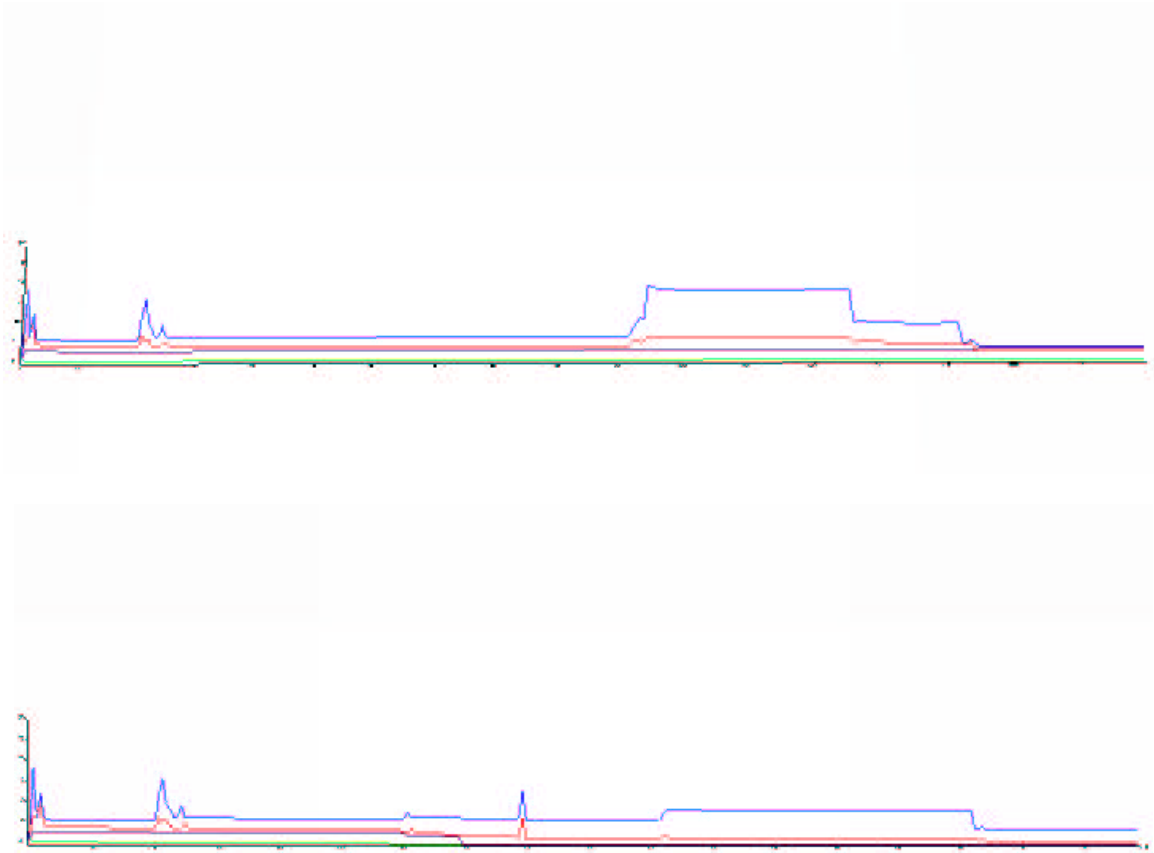


Figure 15: Time course for 7 second delay experiment, (i) normal case and (ii) pathological case

## 8 Summary and conclusions

We have shown how a general system-level model of the brain can be used to model the brain mechanisms involved in carrying out spatial working memory experiments. This involved the development of mechanisms for mental imagery, for planning and for episodic memory.

The model allows one to analyse the degradation of performance of the spatial working memory task. We showed how to find all the data dependencies in the system, which allows us to see the dependencies of the overall performance on the different mechanisms comprising it.

We then discussed how the performance can be continuously degraded by reducing various parameters representing the strength of goals, and the attenuation of different kinds of data item in the system. We concluded that the most general way to obtain the deficits observed in schizophrenia is by attenuating the goal description, which then lead to the attenuation of the plan being executed.

## References

- [Anderson, 1993] Anderson, J. R. (1993). *Rules of the mind*. Lawrence Erlbaum Associates, Hillsdale, New Jersey.
- [Bilder et al., 2004] Bilder, R. M., Volavka, J., M.Lachman, H., and A.Grace, A. (2004). The Catechol-O-Methyltransferase polymorphism: relations to the tonic-phasic dopamine hypothesis and neuropsychiatric phenotypes. *Neuropsychopharmacology*, 29:1943–1961.
- [Bond, 1996] Bond, A. H. (1996). A Computational Architecture for Social Agents. In *Proceedings of Intelligent Systems: A Semiotic Perspective, An International Multidisciplinary Conference, National Institute of Standards and Technology, Gaithersburg, Maryland, USA, Oct 20-23*.
- [Bond, 1999] Bond, A. H. (1999). Describing Behavioral States using a System Model of the Primate Brain. *American Journal of Primatology*, 49:315–388.
- [Bond, 2002a] Bond, A. H. (2002a). Modeling social relationship: An agent architecture for voluntary mutual control. pages 29–36. in [Dautenhahn et al., 2002].
- [Bond, 2002b] Bond, A. H. (2002b). Problem-solving behavior in a system model of the primate neocortex. *Neurocomputing*, 44-46C:735–742.

- [Bond, 2004a] Bond, A. H. (2004a). A Computational Model for the Primate Neocortex based on its Functional Architecture. *Journal of Theoretical Biology*, 227:81–102.
- [Bond, 2004b] Bond, A. H. (2004b). An Information-processing Analysis of the Functional Architecture of the Primate Neocortex. *Journal of Theoretical Biology*, 227:51–79.
- [Bond, 2005a] Bond, A. H. (2005a). A psycholinguistically and neurolinguistically plausible system-level model of natural-language syntax processing. *Neurocomputing*, 65-66:833–841.
- [Bond, 2005b] Bond, A. H. (2005b). Brain mechanisms for interleaving routine and creative action. Presented at CNS\*05, Madison, Wisconsin, July 2005, to be published in *Neurocomputing*.
- [Bond, 2005c] Bond, A. H. (2005c). Representing episodic memory in a system-level model of the brain. *Neurocomputing*, 65-66:261–273.
- [Braver et al., 1999] Braver, T. S., Barch, D. M., and Cohen, J. D. (1999). Cognition and control in schizophrenia: a computational model of dopamine and prefrontal function. 46:312–328.
- [Cannon et al., 2005] Cannon, T. D., Hennah, W., van Erp, T. G. M., Thompson, P. M., Lonnqvist, J., Huttunen, M., Gasperoni, T., Tuulio-Henriksson, A., Pirkola, T., Toga, A. W., Kaprio, J., Mazziotta, J., and Peltonen, L. (2005). Association of DISC1/TRAX haplotypes with schizophrenia, reduced prefrontal gray matter, and impaired short- and long-term memory. *Archives of General Psychiatry*, 62:1205–1213.
- [Conklin et al., 2005] Conklin, H. M., Curtis, C. E., Calkins, M. E., and Iacono, W. G. (2005). Working memory functioning in schizophrenia patients and their first-degree relatives: cognitive functioning shedding light on etiology. *Neuropsychologia*, 43:930–942.
- [Curtis and D’Esposito, 2003] Curtis, C. E. and D’Esposito, M. (2003). Persistent activity in the prefrontal cortex during working memory. *Trends in Cognitive Sciences*, 7:415–423.
- [Dautenhahn et al., 2002] Dautenhahn, K., Bond, A. H., Canamero, D., and Edmonds, B. (2002). *Socially Intelligent Agents: Creating relationships with computers and robots*. Kluwer Academic Publishers, Norwell, Massachusetts.
- [Edeleman, 1987] Edeleman, G. M. (1987). *Neural Darwinism: The theory of neuronal group selection*. Basic Books, New York.
- [Epstein et al., 2001] Epstein, R., Yoe, E. D., Press, D., and Kanwisher, N. G. (2001). Neuropsychological Evidence for a Topographical Learning Mechanism in Parahippocampal Cortex. *Cognitive Neuropsychology*, 18:481–508.

- [Farah, 2000] Farah, M. J. (2000). *The cognitive neuroscience of vision*. Basil Blackwell, Oxford.
- [Felleman and Essen, 1991] Felleman, D. J. and Essen, D. C. V. (1991). Distributed hierarchical processing in the primate visual cortex. *Cerebral Cortex*, 1:1–47.
- [Glahn et al., 2003] Glahn, D. C., Therman, S., marko Manninen, Huttunen, M., Kaprio, J., Lonnqvist, J., and Cannon, T. D. (2003). Spatial working memory as an endophenotype for schizophrenia. *Biological Psychiatry*, 53:624–626.
- [Gottesman and Gould, 2003] Gottesman, I. I. and Gould, T. D. (2003). The endophenotype concept in psychiatry: etymology and strategic intentions. *American Journal of Psychiatry*, 160:636–645.
- [Grill-Spector and Kanwisher, 2005] Grill-Spector, K. and Kanwisher, N. G. (2005). Visual Recognition: As Soon as You Know It Is There, You Know What It Is. *Psychological Science*, 16:152–160.
- [Grossberg, 2003] Grossberg, S. (2003). From normal brain to schizophrenia. *Psychopathology Research*, 13:5–10.
- [Haarman et al., 1997] Haarman, H. J., Just, M. A., and Carpenter, P. A. (1997). Aphasie sentence comprehension as a resource deficit: a computational approach. *Brain and Language*, 59:76–120.
- [Hertz et al., 1991] Hertz, J., Krogh, A., and Palmer, R. G. (1991). *Introduction to the theory of neural computation*. Perseus Books, Reading, Massachusetts.
- [Hogger, 1991] Hogger, C. J. (1991). *Essentials of Logic Programming*. Oxford University Press, Oxford.
- [John Laird and Paul Rosenbloom and Allen Newell, 1986] John Laird and Paul Rosenbloom and Allen Newell (1986). *Universal Subgoaling and Chunking : The Automatic Generation and Learning of Goal Hierarchies*. Kluwer Academic Publishers, Boston, Dordrecht, London.
- [Kanwisher, 2001] Kanwisher, N. (2001). Neural Events and Perceptual Awareness. *Cognition*, 79:89–113.
- [Kintsch, 1998] Kintsch, W. (1998). *Comprehension : A Paradigm for Cognition*. Cambridge University Press, Cambridge, England.
- [Kosslyn, 1994] Kosslyn, S. (1994). *Image and Brain*. M.I.T. Press, Cambridge, Massachusetts.

- [Kosslyn, 1981] Kosslyn, S. M. (1981). The medium and the message in mental imagery: A theory. In Block, N., editor, *Imagery*, pages 207–258. M.I.T. Press, Cambridge, Massachusetts.
- [Kosslyn et al., 1984] Kosslyn, S. M., Brunn, J., Cave, K. R., and Wallach, R. W. (1984). Individual differences in mental imagery ability: A Computational analysis. In Pinker, S., editor, *Visual cognition*, pages 195–271. M.I.T. Press, Cambridge, Massachusetts.
- [Kowlaski, 1974] Kowlaski, R. A. (1974). Predicate logic as a programming language. In *IFIP74*, pages 569–574.
- [Lloyd, 1987] Lloyd, J. (1987). *Foundations of Logic Programming*. Springer-Verlag, Berlin.
- [Marr, 1982] Marr, D. (1982). *Vision : a computational investigation into the human representation and processing of visual information*. W. H. Freeman, San Francisco.
- [Rolls and Treves, 1998] Rolls, E. T. and Treves, A. (1998). *Neural Networks and Brain Function*. Oxford University Press, Oxford.
- [Rowe et al., 2002] Rowe, J., Friston, K., Frackowiak, R., and Passingham, R. (2002). Attention to action: specific modulation of corticocortical interactions in humans. *Neuroimage*, 17:988–998.
- [Rowe et al., 2000] Rowe, J. B., Toni, I., Josephs, O., Frackowiak, R. S. J., and Passingham, R. E. (2000). The prefrontal cortex: response selection or maintenance within working memory. *Science*, 288:1656–1660.
- [Shepard and Cooper, 1982] Shepard, R. N. and Cooper, L. A. (1982). *Mental images and their transformations*. M.I.T. Press, Cambridge, Massachusetts.
- [Sterling and Shapiro, 1994] Sterling, L. and Shapiro, E. (1994). *The Art of Prolog: advanced programming techniques*. M.I.T. Press, Cambridge, Massachusetts.
- [Ungerleider and Mishkin, 1982] Ungerleider, L. G. and Mishkin, M. (1982). Two cortical visual systems. In Ingle, D. J., Goodale, M. A., and Mansfield, R. J. W., editors, *Analysis of Visual Behavior*, pages 549–586. M.I.T. Press, Cambridge, Massachusetts.
- [VanEmden and Kowalski, 1976] VanEmden, M. H. and Kowalski, R. A. (1976). The Semantics of Predicate Logic as a Programming Language. *Journal of the Association for Computing Machinery*, 23:733–742.