Reverse Engineering the Brain

James S. Albus

National Institute for Standards and Technology
Gaithersburg, MD 20899
James.Albus@NIST.gov

Abstract

Reverse engineering the brain will require a deep understanding of how information is represented and how computation is performed in the brain. What are the functional operations? What are the knowledge data structures? How are messages encoded? How are relationships established and broken? How are images processed? How does the brain transform signals into symbols? How does the brain generate the incredibly complex colorful, dynamic internal representation that we consciously perceive as external reality?

The model presented here hypothesizes that each cortical hypercolumn together with its underlying thalamic nuclei performs as a Cortical Computational Unit (CCU) consisting of a frame-like data structure (containing attributes, state, and pointers) plus the computational processes and mechanisms required to build and maintain it.

In sensory-processing areas of the brain, CCUs enable segmentation, grouping, and classification. Pointers stored in CCUframes link pixels and signals to objects and events in situations and episodes that are overlaid with meaning and emotional values. In behavior-generating areas of the brain, CCUs make decisions, set goals and priorities, generate plans, and control behavior. Pointers are used to define rules, grammars, procedures, plans, and behaviors.

It is suggested that it may be possible to reverse engineer the human brain at the CCU level of fidelity using next-generation massively parallel computer hardware and software.

Introduction

It is often said that the mind is what the brain does. The question is – "What exactly does the brain do?" How does it compute? How does it represent information? How does it process data from its sensors into a percept of the world as a 4D space-time reality filled with objects, events, and relationships? How does it represent experiences and understand situations? How does it reason about the past, plan for the future, and react to unexpected events?

Reverse engineering implies that you understand the way something works, and you have the technology to duplicate its functional properties, if not its physical embodiment. Reverse engineering the brain implies building computational machines that are functionally equivalent to the brain[1] in their ability to perceive, think, decide, and act in a purposeful way to achieve goals in complex, uncertain, dynamic, and possibly hostile environments, despite unexpected events and unanticipated obstacles, while guided by internal values and rules of conduct.

To reverse engineer the brain, we need to understand the functional processes in the brain at many levels:

- overall system level (central nervous system – an integrated percept of reality)
- arrays of macro-computational units (e.g., prefrontal, premotor, motor, occipital, posterior parietal, and inferior temporal cortices – representations of egospace, situations, episodes, goals, plans, priorities)
- individual macro-computational units (e.g., cortical columns and cortico-thalamic loops – representations of entities, events, and relationships)
- individual micro-computational units (e.g., cortical microcolumns – representations of attributes, state, and pointers)
- neural clusters (e.g., spinal and midbrain sensory-motor nuclei – representation of goal-selectable stimulus-response coordination of muscles and joints)
- neurons (elemental computational units – representation of input/output functions)
- synapses (electronic gates, memory elements – representation of long-term memory)
- membrane mechanics (ion channel activity – representation of molecular interactions)

Computational modeling at each of these levels can be, and to some extent is being, addressed – some more than others. What so far is missing is a theoretical framework that integrates all these levels into a reference model architecture that specifies input/output functionality of the computational modules, data structures for the representation of knowledge, and message formats for communication of information. In the remainder of this

[1] Functional equivalence can be defined as producing the same input/output behavior. Given the same input stimulus, the engineered system should produce statistically the same output behavior as the biological system.

paper, I will suggest some elements for constructing such a theoretical framework.

Computation in the Brain

The Neuron. The basic computational module in the brain is the neuron. We can model the neuron as a computational device that operates on a time varying vector of input variables and produces a time varying scalar output variable.

The computational function performed by a single neuron can be written as:

$$p_k(t) = \mathbf{H}_k(\mathbf{S}(j, t))$$

where k = identifier (i.e., name or address) of the neuron, $p_k(t)$ = time varying output variable from the k-th neuron, $\mathbf{H}_k$ = computational function performed by the k-th neuron, $\mathbf{S}(j, t)$ = time-varying vector of input variables, j = index of input variable in the input vector.

Groups of Neurons. In the brain, neurons typically are clustered into groups (i.e., nuclei.) For a group of neurons, both input and output can be represented as time-dependent vectors. This can be written as

$$\mathbf{P}_\mathrm{m}(k, t) = \mathbf{H}_\mathrm{m}(\mathbf{S}(j, t)),$$

where, m = identifier (i.e., name or address) of the group, $\mathbf{P}_m(k, t)$ = time-varying output vector from m-th nuclei, $\mathbf{H}_m$ = functional transformation performed by the m-th nuclei, $\mathbf{S}(j, t)$ = time-varying input vector, j = index of input variable in input vector, k = index of output variable in output vector. A temporal sequence of vectors enables a representation of motion and temporal continuity.

Groups of neurons effectively compute by table look-up. For every input, there is an output. The input vector is functionally equivalent to an address, and the output vector is functionally equivalent to the contents of the address. A good example of how a group of neurons can compute arithmetic and logical functions by table look-up can be found in the cerebellar cortex. (Marr 1969; Albus 1971, 1975a, 1975b, 1981).

Arrays of neurons. Often neurons are arranged in 2D arrays. For example, the neocortex consists of a 2D array of columns of neurons. Input to, and output from, an array of neurons can be represented as a time-dependent array of vectors. This can be written as:

$$\mathbf{P}_\mathrm{a}(r, s, k, t) = \mathbf{H}_\mathrm{a}(S(u, v, j, t))$$

where u and r = horizontal indices in the input and output arrays respectively, v and s = vertical indices in the input and output arrays respectively, j and k = indices in the vectors at the respective array elements, a = identifier of the computational array.

An array of neurons can perform an image operation such as edge detection, region growing, connected components analysis, stereo correlation, range imaging, and registration and comparison of one image with another. An array of neurons can transform an image from image coordinates to world coordinates, and vice-versa. These types of computations can typically be accomplished in a two-layer array of neurons such as exists in the cerebellar cortex with time delay less than 10 ms. This could partly explain how we perceive the external world to be stable even though the retinal image bounces wildly as our eyes saccade quickly over the egosphere, and the head in which the eyes are located bobs and weaves through the environment.

Arrays of neurons are often layered in hierarchies. This can be written as:

$$\mathbf{P}_a(n, r, s, k, t) = \mathbf{H}_a(\mathbf{S}(i, u, v, j, t))$$

where i = index of the hierarchical level from which the input array originates, n = index of the hierarchical level in which the computational array reside. It should be noted that the (r, s) address pointer at level (n) need only specify the location of the level (i) input vector variable within the receptive field of the level (n) neuron.

Loops. The existence of loops in the nervous system is widely acknowledged, but seldom appreciated. It is well known from sequential machine theory that when there exists a loop from the output back to the input, a computational module acquires a whole new level of capabilities, and inherits a new set of problems. New problems include the risk of oscillation and instability. New capabilities include the ability to perform temporal analysis, such as temporal differentiation and/or integration operations on vectors and arrays. This enables groups of neurons such as reside in the cerebellum and basal ganglia to generate real-time control solutions to body-environment dynamics.

Loops also provide feedback paths for state variables. This enables computational modules to act as finite-state automata (fsa). Fsa can generate sequential strings, lists, graphs, and grammars. Fsa can perform algorithms, store and execute rules, generate plans, and execute programs. Groups of fsa can provide short-term memory in the form of delay lines, stacks, or queues.

A hierarchy of fsa can generate goal-seeking sensory-interactive behaviors. Many robot control systems use fsa as computational mechanisms. (Albus 2001, 1981, Brooks 1999, Albus and Barbera 2005, Madhaven et al 2007)

It seems likely that hierarchies of arrays of neural computational modules acting as fsa enable the frontal brain to generate complex goal-directed sensory-interactive behaviors such as hunting, foraging, telling stories, playing musical instruments, and performing routine daily tasks of going to work or school, avoiding danger, and interacting with others in social situations. Hierarchies neural fsa may enable the posterior brain to perform recursive

estimation on attributes and state of objects, events, situations, and episodes.

Communications within the Brain

Communications within and between regions in the brain take place via two fundamentally different types of neurons: drivers and modulators. (Sherman and Guillery 2001)

Drivers

Axons from driver neurons convey data in the form of attribute values that are topologically organized in the form of vectors, arrays, or matrices (e.g., signals and images or maps.) In the visual and somatosensory systems, these are sensory maps of egospace. Driver neurons are often called relay neurons because they receive topologically organized arrays from input drivers and transmit topologically organized arrays to target drivers. Driver inputs are characterized by relatively small receptive fields, and their output axons cover small fields of influence. Axons from driver neurons typically terminate in dense proximal synaptic connections on a few target neurons in a limited neighborhood. Driver neurons in both the thalamus and cortex define maps of egospace, and these maps are repeated many times, both at the same and at different levels in the sensory hierarchy. For example, receptive fields of rods and cones in the retinal image are topologically mapped with great precision to the lateral geniculate nucleus (LGN), and from there to cortical columns in the primary visual cortex V1. Inputs from the right and left retina are precisely registered in the LGN before being relayed to V1. Van Essen and colleagues have identified at least 32 separate representations of the visual field in the cortex. (Felleman & Van Essen 1991) Connections between these representations suggest that they are arranged in 12 hierarchical levels with many two-way communication pathways both within and between levels.[2]

Preservation of topology and registration with the sensory egosphere exists not only in the visual cortex, but in the somatosensory and auditory cortices as well. Receptive fields of somatosensory sensors in the skin, muscles, and joints are topologically mapped onto the ventral posterior nucleus of the thalamus before being relayed to the primary somatosensory cortex. Inputs from auditory sensors in the cochlea are tonotopically mapped in the medial geniculate nucleus of the thalamus before being relayed to the auditory cortex as spectrograms.

Topology and registration is obviously important to the computational processes in the brain because the neural circuitry goes to considerable lengths to preserve them, even though in many cases this requires that the maps be inverted or reflected between levels in the hierarchy in order to preserve the topological relationships.

Modulators

Modulator signals control the computational processes that operate on driver signals. Modulator neurons have relatively large receptive fields and their axons typically cover large fields of influence with low density and distal synaptic connections on driver neurons. Modulator neurons typically do not preserve topological order, or at best do so only generally.

Modulator inputs were traditionally thought to simply suppress or enhance general levels of activity of drivers in various regions of the brain. This traditional view of modulators is undoubtedly correct for those modulator inputs that are concerned with general levels of alertness, e.g., those originating in midbrain activating centers. However, it seems likely that modulator inputs originating in the cortex play a more sophisticated role.

It is well established that the number of modulator axons flowing from neocortex back to the thalamus typically exceeds the number of driver neurons carrying sensory images from thalamus to cortex by a ratio of 20:1 or more. (Granger 2005, Sherman and Guillery 2001) This suggests that while driver axons convey data in the form of vectors or arrays of attribute values, modulator axons convey vectors that can be decoded into addresses of locations where information is stored.

Hypothesis # 1. We hypothesize that modulators access or activate locations where the following types of information are stored:

a) **arrays of parameters** that filter, enhance, or mask regions in the driver data arrays – this enables focus of attention and segmentation.

b) **filtered estimates** of pixel, entity, or event attributes – this enables the storage and retrieval of an internal model of the external world.

c) **predicted values** of pixel, entity, or event attributes – this enables recursive estimation and predictive filtering of the internal world model. It also enables planning.

d) **attributes** of entity or event class prototypes – this enables classification based on similarities between observed entity or event attributes and class prototype attributes.

[2] It should be mentioned that most, if not all, of the direct connections between cortical regions in posterior cortex are modulators. Driver outputs from the cortex rarely flow directly to another cortical region in the same hierarchy. Driver outputs from the cortex also do not loop back to the region of the thalamus from which their driver inputs originated. Driver outputs from the cortex typically travel either to: a) a higher-level thalamic relay center that sends it on to a higher-level cortical region, or b) to a comparable or lower level in the behavior generating hierarchy.

e) **procedures** that operate on driver arrays – this enables modulators to select procedures for processing of sensory information.

f) **parameters** that set decision criteria or modulate functional operations – this enables modulators to manipulate algorithms that generate and control behavior.

g) **pointers** that define relationships between entities and events – this enables modulators to make and break relationships that define situations, episodes, plans, and behaviors.

Modulator vectors that address these various types of information can be modified within a few milliseconds. Pointers between locations in the cortex and locations in the thalamus can be established within a single cortical-thalamic loop cycle.

Loops between sets of modulator neurons in computational modules at two different locations in the brain, enable the establishment of forward and backward pointers between those two locations. Time variable arrays of back-pointers enable moving sensory images to be projected onto a stable representation of the world.

The ability of modulators to access pointers, and to act as pointers, enables the apparent location of data to be modified without moving its physical location. Arrays of pointers enable the perceived position and orientation of attribute images to be shifted and rotated without actually scrolling the images. This may be another factor in the explanation of how the brain perceives the external world as stable despite rapid motion of the image of the world on the retina.

The ability of modulator neurons to set and access pointers suggests how relationships can be established, maintained, and broken between data structures in computational units at higher, lower, and the same hierarchical levels, both locally and in distant regions of the brain. *has-part* and *belongs-to* pointers define relationships between entities and events at higher and lower levels of the segmentation and grouping hierarchy. Ultimately they link pixels to objects and signals to events, and vice versa. *is-a-member-of* pointers link entity and event frames to class prototypes. Other types of pointers can link entities and events in graph structures that describe situations and episodes. In general, pointers set by modulators are able to represent many forms of symbolic knowledge, including strings, graphs, semantic nets, belief nets, grammars, rules, programs, procedures, scripts, tasks, skills, plans, and behaviors.

In summary, drivers communicate specific data (e.g., attributes) that are topologically organized as images or maps, while modulators communicate addresses (e.g. pointers) that control computational processes and define relationships.

In posterior brain, drivers convey data flowing up the sensory processing hierarchy in the form of attributes and states of pixels, signals, entities, or events. Modulators select parameters (e.g., coefficients, priorities, masks, gestalt grouping criteria) that control functional processes that operate on driver data. Modulators also set pointers that establish relationships (e.g., *belongs-to*, *is-a-member-of*, *has-part*, *is-caused-by*, *is-prevented-by*, *geometric*, *temporal*, *procedural*, and *semantic* relationships.)

In frontal brain, drivers convey commands, goals, and priorities flowing down the behavior generating hierarchy to control behavior. Modulators select parameters (e.g., priorities, modes of operation, rules, grammars) that control functional processes that make decisions, generate plans, and sequence actions.

Representation of Information

Information has two components: address and content. Address is where the information is located. Content is the information contained at the address location.

In a computer, addresses are carried on one set of wires and data is carried on a different set of wires. The appearance of an address vector on a set of address wires will access a memory location, either to read data from the address location, or to write data into the address location. The command whether to read or to write is carried on a r/w command wire.

In the brain, the input to any neuron or set of neurons is effectively an address. The output is the contents of that address. Both inputs and outputs can be drivers, or modulators, or both. Driver inputs have local sign and represent attributes or state of entities or events. Modulator inputs lack local sign and represent parameters that focus attention, select grouping criteria, and establish relationships. Driver outputs represent data in the form of attributes or state of entities or events. Modulator outputs represent pointers that index into arrays, define relationships, or modify processes.

Iconic Representations

Visual, somatosensory, and audio sensors are arranged in 2D arrays (i.e., images or maps) on the surface of the retina, the skin, and the cochlea. These sensory images can be tessellated into pixels. A pixel is a patch of real estate on the sensory surface. Within each pixel, there typically are several types of sensor. For example, pixels on the retina include rods and three kinds of cones that are distributed unevenly over the retina. The pixels on the skin contains arrays of sensors that are sensitive to touch, pressure, temperature, vibration, and pain. The pixels on the cochlea contains arrays of hair cells that differentially respond to frequency. Processing layers in the visual, somatosensory, and audio hierarchies prior to the

neocortex compute spatial and temporal derivatives of these different types of sensory excitation.

Thus, each pixel arriving at the thalamus can be represented as a vector of attributes (both sensed and computed), and the sensory image can be represented as a 2D array of pixel attribute vectors. Visual pixel attribute vectors may consist of intensity and color (r-g-b), plus spatial and temporal gradients of intensity and color. Tactile pixel attribute vectors may consist of pressure, temperature, vibration, and pain variables; plus spatial and temporal derivatives of these variables. Audio pixel attribute vectors may consist of frequency, amplitude, and impulse magnitude variables.

Each pixel attribute varies with time. If we sample the attributes along the time axis we get a string of attribute values for each pixel. A 2D array of pixels, each with an attribute vector sampled in time results in a 3D matrix of attribute vectors, or a 4D matrix of attribute values of the form:

$$pixel(u, v, j, t)$$

where u = horizontal row index of the pixel in the sensory array, v = vertical column index of the pixel in the sensory array, j = index of attributes and pointers within the pixel, t = time. We can then represent the entire body surface, or the entire retinal image, as a two-dimensional array of time dependent attribute vectors **A**(u, v, j, t). We can, in fact, represent any driver image in the periphery, thalamus, or cortex as an array of attribute vectors.[3]

Cortical Columns

The cortex is a massively parallel structure. The human neocortex is a thin sheet of computational units about 2000 cm^2 (2.2 ft^2) in area and about 3 mm thick. The cortex is remarkably uniform throughout its extent. It consists of six layers of neurons axons, dendrites, and synapses that are specialized in form and function.

The surface of the cortex is tessellated into an array of columns of neurons. (Montcastle 1997, Asanuma 1975) There are two types of cortical columns:

1) microcolumns contain 100 – 250 neurons. These are only 30 - 50 μ in diameter (scarcely more than one neuron wide) and about 3000 μ long (i.e., the full thickness of the cortex from layer 6 to layer 1)
2) hypercolumns (a.k.a. columns) contain 100 + microcolumns in a bundle. These are on the order of 500 μ in diameter and also about 3000 μ long.

A typical hypercolumn occupies about .2 mm^2 of cortical real estate. Thus, there are about a million hypercolumns in the cortex, and more than 100 million microcolumns.
It has long been suspected that these cortical columns perform some kind of computational function in the cortex. (Montcastle 1997)

Cortical Computational Units (CCUs)

Each of the hypercolumns in the cortex is serviced by underlying thalamic nuclei that are connected to the cortex through looping communication pathways. The uniformity of structure in the cortex and the cortico-thalamic loops suggests that similar computational mechanisms may be used throughout the cortex despite the differences in functional processes that are performed in different regions.

Hypothesis # 2. We hypothesize that each cortical hypercolumn, together with its underlying thalamic and other subcortical nuclei, plus the looping circuitry that connect them together, are functionally equivalent to a *Cortical Computational Unit* (CCU.)

Hypothesis #3. Each CCU represents an entity or event. This implies that each cortical hypercolumns in the brain represents a spatial or temporal pattern (i.e., an entity or event.) Herein lies the foundation for symbol grounding.

Hypothesis #4. Each CCU has three parts:

1) an abstract data structure in the form of an entity or event CCUframe that:
    a) has a name (i.e., an address where it is located in the array of CCUs)
    b) has slots for attributes and state
    c) has slots for pointers that define relationships
    d) has grouping criteria (that may be variable)
2) a set of computational processes that are able to:
    a) within a spatial receptive field, segment spatial patterns that satisfy spatial grouping criteria into entities, and
    b) within a temporal receptive field, segment temporal patterns that satisfy temporal grouping criteria into events
    c) set *belongs-to* and *has-part* pointers between spatial pattern components and entities, and between temporal pattern components and events
    d) compute entity or event attributes and state and store them in CCUframe slots
    e) use computed attributes to generate predictions for recursive estimation and planning

[3] The size of the pixels is defined by the spacing of sensors. Pixel size may vary within the array. For example in the retina, the size of the pixels is small in the fovea, and large in the periphery. For tactile sensors, pixel size is small in the lips, tongue, and finger tips, and larger elsewhere.

f) compare entity or event attributes with class prototypes, and when a match occurs, set *is-a-member-of* pointers to classes

3) a set of computational processors (e.g., synapses, neurons, and circuits in microcolumns and hypercolumns) that implement the above processes and data structures.

A sketch of the internal structure of our proposed CCU is shown in Figure 1.

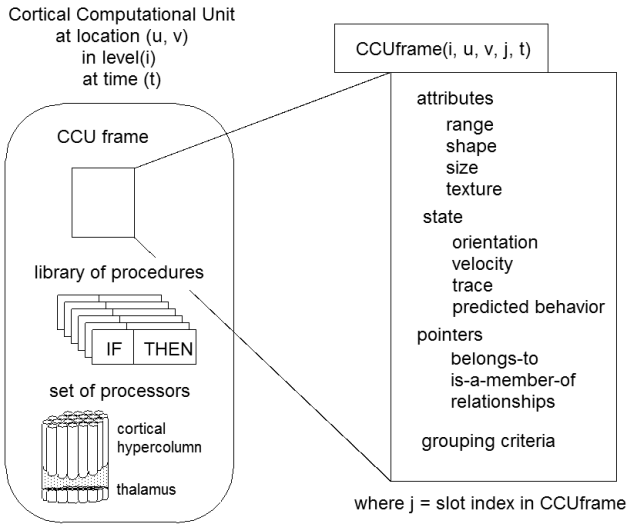

Figure 1. Internal structure of a Cortical Computational Unit (CCU) consisting of a CCUframe, a library of procedures that maintain the frame, and a set of processors that implement the procedures.

The data structure for entity frames in CCUs in posterior cortex can be represented as a matrix

$$\text{CCUframe}(i, u, v, j, t)$$

where i = level index in the sensory processing hierarchy, u = row index of the CCU in the cortical array at level(i), v = column index of the CCU in the cortical array at level(i), j = index of slots containing attributes and pointers in the CCUframe, t = time.

An input/output diagram of a typical posterior CCU is shown in Figure 2.

Driver inputs convey attributes of entity or event frames from lower-level CCUs in the input receptive field.

Driver outputs convey attributes of the entity or event frame in the local CCU to higher-level CCUs in the output field of influence.

Modulator inputs from above convey commands, priorities, and pointers from above that select processing algorithms, generate masks and windows, provide prioritized list of entities of attention, suggest gestalt grouping hypotheses, and set *belongs-to* pointers in the local CCU.

Modulator outputs to above convey status of processing results to higher-level CCUs, set alerts regarding regions of the field of view that require attention, and set *has-part* pointers in level above.

Modulator inputs from below convey status of processing results from lower-level CCUs, set alerts regarding regions of the field of view that require attention, and set *has-part* pointers in the local CCU.

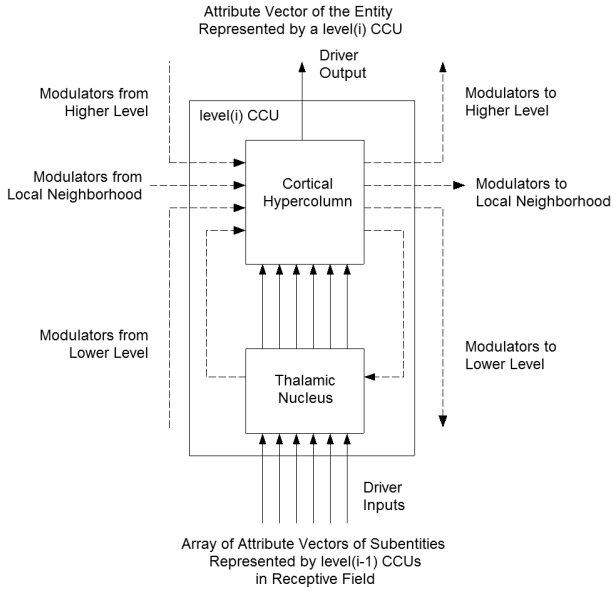

Figure 2. Inputs and outputs of a Cortical Computational Unit (CCU) in posterior cortex.

Modulator outputs to below convey commands and priorities to lower-level CCUs that select processing algorithms, generate masks and windows, provide prioritized list of entities of attention, suggest gestalt grouping hypotheses, and set *belongs-to* pointers in the level below.

Modulator inputs and outputs to and from the same level provide lateral inhibition for winner take all segmentation and grouping.

Modulator signals between the cortex and thalamus within CCUs provide information for windowing, segmentation, and comparing of model-based predictions with observations.

Segmentation and Classification

Our model assumes that CCUs at all levels in the sensory processing hierarchy contain CCUframe data structures with slots for attributes, state, pointers, and grouping criteria. Within each receptive field at every sensory level, lower level CCUs are segmented into groups (or patterns) that are linked to higher level CCUs. At each level, CCUframes have *belongs-to* pointers to higher level CCUframes, *has-part* pointers to lower level entity (or event) frames, and *is-a-member-of* pointers to class frames. CCUframes may also be linked together by *relationship* pointers that define situations, episodes, rules, procedures, strings, lists, graphs, and grammars.

Our model assumes that segmentation and grouping processes are embedded in CCUs at each level in the receptive field hierarchy. This is illustrated in Figure 3.

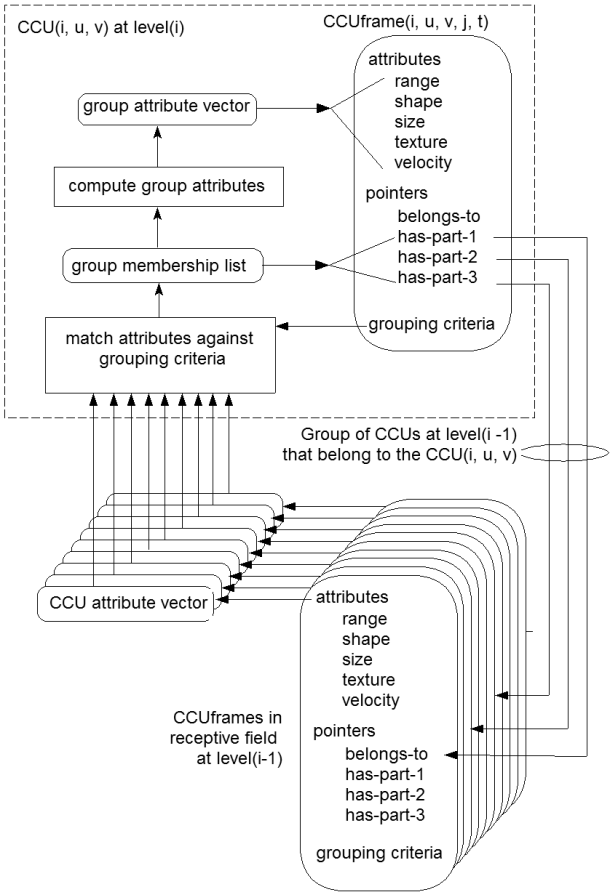

Figure 3. Segmentation and grouping processes in posterior cortex compare lower level CCU attributes against upper level CCU grouping criteria. When attributes of lower level CCUs meet higher level grouping criteria, *has-part* pointers are set in the higher level CCUframe, and *belongs-to* pointers are set in the lower level CCUframes.

For each CCU at level(i), there are segmentation processes that group level(i-1) CCUs into level(i) CCUs based whether or not their attributes satisfy the level(i) CCU grouping criteria. Group attribute criteria may define gestalt properties such as similarity, symmetry, or spatial or temporal proximity or continuity. For example, level(i-1) pixels with similar attributes of color, range, and range gradient, may be grouped into level(i) surface patch entities. Level(i-1) pixels with similar intensity, color, or range gradients may be grouped into level(i) edge-entities.

Once level(i-1) entities (or events) have been grouped into level(i) entities (or events), level(i) computational processes in the CCU compute level(i) entity (or event) attributes and store them in the appropriate CCUframe slots. Recursive estimation processes can then compare model-based predictions with sensory observations. Finally, classification processes compare entity (or event) attributes against class prototypes. This establishes *is-a-member-of* pointers that define class membership and enable observed entities and events to inherit class attributes.

The result of segmentation and grouping is a set of *belongs-to* and *has-part* pointers that link level(i-1) entities (or events) to level(i) entities (or events.) The set of *has-part* pointers define a membership list for each level(i) entity (or event) that includes all the level(i-1) entities or events that belong to it. This is illustrated in Figure 4.

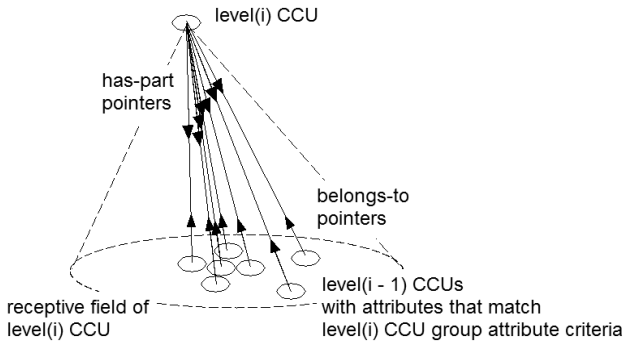

Figure 4. The result of segmentation and grouping. A group of CCUs at level(i-1) are linked by belongs-to pointers to an CCU at level(i). A list of pointers in the level(i) CCU points back to the CCUs at level(i-1).

Hierarchies of Entities and Events

Each level of the sensory processing hierarchy consists of an array of cortical hypercolumns. Each hypercolumn and its underlying subcortical nuclei are modeled by a single CCU. An example of a two level entity hierarchy resulting from two levels in the sensory processing hierarchy is shown in Figure 5.

A similar hierarchy of events can be generated for CCUs that segment events based on time and frequency criteria.

Thus, for both spatial entities and temporal events, there are CCU processes that maintain pointers that define spatial and temporal relationships in situations and episodes, and provide the basis for grammar, scripts, plans, and patterns of locomotion, manipulation, and language behavior. At all levels, in all of the CCUs in both sensory processing and behavior generating hierarchies, all of these processes execute in parallel simultaneously.

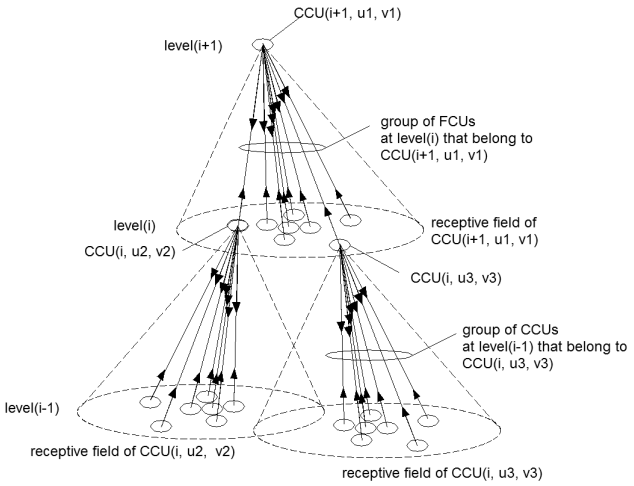

Figure 5. Two levels of segmentation and grouping of entity frames.

In frontal cortex, CCUframes contain slots for goals, priorities, mode, rules, task state, and pointers to plans and behaviors. Driver information flows down the hierarchy in the form of commands and task parameters. CCU computational processes make decisions, select goals, decompose tasks, make plans, and sequence behaviors.

Reverse Engineering

Computational Modeling

There are about a million hypercolumns in the human neocortex. Thus, it would take about a million CCUs to build a full-scale computational model of the human cortex. Because each of these computational units has a similar internal structure, this model is well suited to the massively parallel architecture of modern supercomputers.

A modern supercomputer has hundreds of thousands of computational cores that provide computational capacity on the order of 300 teraflop (3×10^{14} floating operations per second). If we divide that up between a million CCUs, we get 3×10^{8} flops per CCU. For real-time operation at 200 compute cycles per second, we obtain about 1.5×10^{6} flops per CCU per compute cycle. This seems adequate to perform the operations we envision taking place in our CCUs. For some types of CCU functions, special purpose hardware may enable significant increase in computational speed.

Communications modeling

Of course, modeling the computational units is only part of the problem. The CCUs need to be connected such that outputs from CCUs in one compute cycle become inputs to other CCUs in the next compute cycle. While it seems probable that a modern supercomputer has sufficient computational power to emulate the functionality of the CCUs, communication between CCUs may turn out to be the bigger technical challenge.

Communications may, in fact, be the limiting factor in engineering a computational model of the brain. This needs much more study. And it needs more information from the neuroscience community. How many CCUs communicate with each other? How often? How much information is communicated? How is it routed? And most importantly, "What is the syntax (i.e., format) and semantics (i.e., meaning) of the messages?"

The new technique of diffusion spectrum imaging promised to provide a detailed map of the major communications pathways in the brain. (Wiegell et al 2001) This may generate the level of detail needed to model communications in the brain.

The modeling of mouse cortex achieved recently at the IBM Almaden Research Center provides some insight into how to model communications in the brain. (Ananthanarayanan) In this project, a simulator called C2 was designed to operate on a Blue-Gene/L supercomputer with 32,768 processors. Neurons were simulated on a 1 ms clock rate, and synapses were simulated on an event-driven basis. There are about 55 million neurons and 440 billion synapses in the rat brain. The IBM team was able to simulate the entire rat brain including communications between neurons and synapses in 1/9 real time (i.e., 1 s of model time in 9 s real time.)

One approach to specification of the messages in a CCU level simulation might be to adopt the Neutral Messaging Language (NML)[4] developed at NIST for communications between modules in real-time control systems (Gazi) and extend it to become a *Neural* Messaging Language (*N*ML). NML is a publish-subscribe communication architecture that uses a configuration file to define the connectivity between modules. The connectivity specified in a *N*ML configuration file needs to be informed by neuro-anatomical data. Receptive fields and fields of influence of both drivers and modulators should be reflected in the publish/subscribe structure of the *N*ML configuration files.

At the end of each computational cycle, drivers and modulators in each CCU must write into their *N*ML output

[4] NML is only one among many similar message passing systems that have been developed in the robotics community. NML is mature, widely used, and in the public domain.

buffers. At the beginning of the next cycle, each CCU must read from its *N*ML input buffers both driver and modulator inputs. In the period between compute cycles, *N*ML must move messages from publishers' output buffers to subscribers' input buffers. To mimic real-time performance, the combined computation-communication cycle should repeat at least every 5 ms.

Learning and Memory

The basic architecture and circuitry of the brain is not learned. It is genetically determined just as is the basic architecture and connectivity of body parts. The body does not learn to have ten fingers and two eyes. Neither does the brain learn to organize itself into an architecture. Learning occurs within the anatomical architecture of the brain, and makes only microscopic modifications in the synaptic connections at a relatively few select locations in that architecture.

There are many types of learning, and a wide variety of processes and mechanisms that produce learning. Learning processes include reinforcement learning based on pain, pleasure, and perceived success or failure; error correction learning from a teacher (e.g., an external instructor or an internal critic); and Hebbian learning based on the timing of pre- and post-synaptic activity. Learning mechanisms involve a variety of complex biochemical and biological processes that are specific to particular synaptic junctions.

Most research on learning is based on mimicry of synaptic modifications. Contents of long-term memories are stored in synaptic strengths. These are relatively permanent, but storage typically requires many repetitions or rehearsals. However, this is only one form of learning. Contents of short-term memories are stored in fsa formed by loops that are linked into lists, graphs, and queues. These memories are volatile and transient, but storage occurs in milliseconds. The transfer of memory from short-term to long-term storage requires the involvement of the hippocampus. Among other things, the hippocampus apparently determines whether what is temporarily stored in short-term memory is important enough to be transferred to long-term memory.

To understand learning in the brain, it is necessary to understand where in the brain the learning takes place, what mechanisms are involved in each location, and what the storage media are. Where are the inputs arriving from, and where are the outputs destined to go? In short, we need to understand how the brain is confugured to do what it *does* before we can understand how it *learns to* do what it does. Until we understand what the functional architecture of the brain is – what the various computational modules do, and how they are linked together in a cognitive architecture – we are unlikely to understand how skills are learned, how knowledge is acquired and structured, or how relationships between entities, events, and classes in situations and episodes are established and broken.

Finally, a word about what is stored in memory. The behaviorist hypothesis that there is no internal model is simply wrong. The internal model is all we consciously perceive. The cortex has no direct connection with the periphery. Our conscious self does not perceive the external world at all. We only perceive our internal model, which is inferred from patterns of spikes on sensory pathways. The remarkable thing about our internal model is that it is far richer and more meaningful than the sensory input. (Albus 2001)

Summary

Reverse engineering the brain implies building machines that are functionally equivalent to the brain in their ability to perceive, think, decide, and act in a purposeful way to achieve goals in complex, dynamic, and possibly hostile environments, despite unexpected events and unanticipated obstacles, while guided by internal values and rules of conduct.

Our fundamental hypothesis is that neocortical hypercolumns and their associated thalamic and other subcortical support nuclei are functionally equivalent to *Cortical Computational Units* (CCUs), each of which contains a CCUframe data structure, plus the computational procedures and mechanisms that populate and maintain it. Each CCUframe has slots for attributes, state variables, and pointers.

In the posterior cortex, CCUframes have slots that define entity or event attributes and pointers that define relationships. Relationships include *belongs-to* and *has-part* links to higher and lower level frames, relationships that characterize situations and episodes, as well as relationships that define class membership, cause and effect, and more general relationships such as strings, graphs, rules, and grammars, procedures, and scripts.

Arrays of CCUs enable windowing, segmentation, recursive estimation, and classification of images from visual, somatosensory, and auditory sensors. Hierarchies of arrays of CCUs enable the transformation of sensory images into an internal representation of the external reality that is both richer and more meaningful than the sensory input. At each level in the sensory processing hierarchies, image elements are windowed, segmented, and grouped into patterns, and those patterns are labeled, characterized, and classified. At the bottom are pixels. At the top are rich dynamic representations of space and time, objects and events, situations and episodes. And the entire structure is linked from top to bottom by pointers defined by modulator neurons. Top-level abstract data structures are linked by *has-part* pointers all the way down to the pixels of immediate experience. Bottom-level pixels are

linked all the way up to situations and episodes by *belongs-to* pointers.

In frontal cortex, CCUframes have slots that define goals, priorities, tasks, objects, and agents. Pointers define plans, skills, and behaviors. CCU corticothalamic loops include the striatium, pallidum, and in some cases the cerebellum – where models of body dynamics and kinematics are stored. At the top are high-level goals and priorities. At the bottom are individual motor neurons that send commands to control muscles. At all levels, links to the limbic system provide evaluation of cost, risk, and benefit of goals, plans, and actions. At every level, decisions are made, goals are selected, plans are generated and evaluated, tasks are decomposed, and behavior is controlled so that millions of individual muscles work together to accomplish high level goals despite unexpected disturbances and obstacles.

The massively parallel nature of this architecture enables networks of pointers to be established quickly and updated rapidly. For real-time behavior, our model requires that pointers be established or updated between any two hierarchical levels within about 10 ms, and all the way from bottom to top of the unimodal hierarchies in less than 100 ms. With each saccade of the eyes or motion of the fingers, low-level pixels and list entities are re-linked to high-level objects and situations that have spatial and temporal continuity in a fixed world coordinate frame.

Finally, we have suggested that reverse engineering the brain at the CCU level of fidelity might be feasible in the near term with high performance sensors and super computer processing capabilities. With continued progress in neuroscience, computer science, and electronics engineering, it may eventually be possible to build a laptop equivalent to the human brain.

References

Albus, J. S., and Barbera, A. J. (2005) "RCS: A cognitive architecture for intelligent multi-agent systems," *Annual Reviews in Control*, Vol. **29**, Issue 1 87-99

Albus, J. S., and Meystel, A. (2001) *Engineering of Mind: An Introduction to the Science of Intelligent Systems*, Wiley, New York.

Albus, J. S. (1981) *Brains, Behavior, and Robotics,* BYTE/McGraw-Hill, Peterborough, N.H.

Albus, J.S. (1975a) "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC),"*Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, September pp. 220-227.

Albus, J.S., (1975b) "Data Storage in the Cerebellar Model Articulation Controller (CMAC),*" Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, September pp. 228-233.

Albus, J.S. (1971), "A Theory of Cerebellar Function," *Mathematical Biosciences*, **10**, pp. 25-61.

Ananthanarayanan, R. and Modha, D. (2007) "Anatomy of a Cortical Simulator", *Supercomputing 07: Proceedings of the ACM/IEEE SC2007 Conference on High Performance Networking and Computing,* November 10-16, Reno, NV.

Asanuma, H. (1975) Recent developments in the study of the columnar arrangement of neurons within the motor cortex. *Physiol. Rev.* **55**: 143-156.

Brooks, R.A. (1999), *Cambrian Intelligence: The Early History of the New AI*, MIT Press, Cambridge, Mass.

Felleman, D. J., and Van Essen, D. C. (1991) "Distributed hierarchical processing in primate visual cortex. *Cerebral Cortex,* **1**, pp. 1-47.

Gazi, V., Moore, M.L., Passino, K.M., Shackleford, W.P., Proctor, F.M., and Albus, J.S. (2001) *The RCS Handbook: Tools for Real Time Control Systems Software Development,* John Wiley & Sons, NY

Granger, R., (2005) "Engines of the brain: The computational instruction set of human cognition." *AI Magazine*

Marr, D. (1969), "A Theory of Cerebellar Cortex", *Journal of Physiology (London),* **202,** pp. 437-470.

Madhaven, R., Messina, E., Albus, J. (2007) *Intelligent Vehicle Systems: A 4D/RCS Approach,* Nova, New York.

Montcastle, V. B. (1997) The columnar organization of the neocortex. *Brain* **120** (Pt. 4) 701-722.

Sherman, S. M., and Guillery, R.W. (2001) *Exploring the Thalamus and It's Role in Cortical Function*, MIT Press, Cambridge, MA.

Wiegell, M., Reese, T., Tuch, D., Sorensen, G., and Wedeen, V. (2001) "Diffusion Spectrum Imaging of Fiber White Matter Degeneration," *Proc. Intl. Soc. Mag. Reson. Med* **9**, p. 504.