A scaleable spiking neural model of action planning

Peter Blouw (pblouw@uwaterloo.ca)
Chris Eliasmith (celiasmith@uwaterloo.ca)
Bryan P. Tripp (bptripp@uwaterloo.ca)
Centre for Theoretical Neuroscience, University of Waterloo
Waterloo, ON, Canada N2L 3G1

Abstract

Past research on action planning has shed light on the neural mechanisms underlying the selection of simple motor actions, along with the cognitive mechanisms underlying the planning of action sequences in constrained problem solving domains. We extend this research by describing a neural model that rapidly plans action sequences in relatively unconstrained domains by manipulating structured representations of objects and the actions they typically afford. We provide an analysis that indicates our model is able to reliably accomplish goals that require correctly performing a sequence of up to 5 actions in a simulated environment. We also provide an analysis of the scaling properties of our model with respect to the number of objects and affordances that constitute its knowledge of the environment. Using simplified simulations we find that our model is likely to function effectively while picking from 10,000 actions related to 25,000 objects.

Keywords: planning; affordances; spiking neurons; neural engineering framework; semantic pointer architecture

Introduction

Action selection is a topic of long-standing interest for understanding human behavior (Shallice, 1982; Daw et al., 2005). Recent results in neurophysiology have clarified some of the underlying neural mechanisms. In particular, moment-to-moment decisions about motor actions such as reaching, grasping, saccades, etc. appear to arise from parallel competitions among representations in multiple frontal and parietal areas (Andersen & Gui, 2009; Cisek, 2012).

Accordingly, action decisions are influenced by many factors (Wolpert & Landy, 2012). Often, a dominant factor is a larger goal that may be several steps removed from any immediately feasible action (for example, one may have to open a laptop and start an application before one can type something). Thus, rapid and frequent multi-action planning is an important part of life. Planning of action sequences has been studied extensively in the context of the Tower of Hanoi and Tower of London tasks, which appear to particularly involve frontal areas (Goel & Grafman, 1995), but (in contrast with decisions about immediate actions) the neural mechanisms are unclear.

Furthermore, compared to more naturalistic tasks (such as making a sandwich) these tasks are arguably relatively deliberative and constrained. In contrast, preparing a meal, cleaning up after it, fixing a loose hinge discovered on the cupboard in the process, etc. require variable (sometimes novel) action sequences that are assembled with little effort, using sophisticated knowledge of the many objects involved and the actions they afford. Human behavior in such contexts is only loosely related to artificial-intelligence planning methods (Russell & Norvig, 2003), which produce complex and optimal plans within narrow domains.

Our goal in this study is to better understand how the knowledge needed for rapid action-sequence planning might be stored and processed in the brain. To this end, we develop a spiking-neuron model that plans action sequences by chaining together action preconditions and effects (Fikes & Nilsson, 1971) while interacting with a simulated environment. Each planning step selects from actions that are related to available objects, in order to constrain each decision and allow planning to proceed quickly (about 100ms of simulated time per step).

While this kind of planning process could in principal be supported by a variety of action representations and model architectures (Krüger et al., 2011; Fincham et al., 2002), some architectures and representations require precision than is unavailable from noisy, spiking neurons, and/or require very large numbers of neurons. A working spiking neural model may therefore be a source of insight into constraints on the brain's solution to this problem.

In what follows, we present our model and analyze its performance on a naturalistic planning challenge (namely, boiling water in a simulated kitchen environment). We contend that the model satisfies two important constraints on the processes underlying action selection. First, the model gives a neurally plausible account of the kinds of representations and processes that underlie planning in cognitive systems. And second, the model can scale to accomplish goals that require performing fairly complex sequences of actions in domains that require an understanding of large numbers of objects.

The Semantic Pointer Architecture (SPA)

To implement our model, we use the Semantic Pointer Architecture (Eliasmith, 2013), a recently developed framework for constructing neurally plausible models of cognitive phenomena. Previously, the SPA has been used to build Spaun, a large-scale simulation of the brain that performs a wide range of cognitive functions (Eliasmith et al., 2012). In what follows, we provide a condensed description of the SPA drawn from material found in Stewart et al. 2014.

A typical SPA model defines a set of subsystems corresponding to particular brain regions. Each subsystem is implemented as a collection of simulated spiking neurons (we use the leaky-integrate-and-fire (LIF) model in this case). Synaptic connections between the neurons in distinct subsystems are then optimized to compute some desired function on a latent vector space that is represented by the neurons'

spiking activities.

For example, a common subsystem is a working memory. Formally, working memory can be described as a differential equation. The neurons in a working memory subsystem can be characterized as representing a vector (or scalar) x, and the input to these neurons can be taken to represent another vector, u. Assuming that x remains constant when u is zero (i.e. a memory is stored), and that x changes proportionally to value of u when u is non-zero, the representational state of the subsystem can be written as $dx/dt = u$.

Helpfully, an arbitrary differential equation of this kind can be approximated with an ensemble of spiking neurons using the Neural Engineering Framework (NEF; Eliasmith & Anderson, 2003). This approximation is achieved by randomly assigning a 'tuning curve' to each neuron that specifies its spike rate in relation to the represented value x. For instance, a given neuron might fire rapidly when the value of x is zero, but fire much more slowly when the value of x is positive. For a given neuron in a SPA model, this tuning curve is assigned randomly using a distribution of firing patterns that is consistent with available empirical evidence.

Once synaptic connections are introduced between two populations of neurons, it is possible to use a local optimization technique to ensure that the tuning curves in each population are appropriately related to the variables they are hypothesized to represent. For example, if the tuning curves in the first population are defined in relation to a variable x, while the tuning curves in the second population are defined in relation to a variable y, then it is possible to find a set of connection weights that approximate the computation $y = f(x)$. In the presence of recurrent connections, this technique can be used to approximate any function of the form $dx/dt = f(x,u)$. The quality of the approximation depends on both the number of neurons being used and the degree of non-linearity present in the function.

In general, the SPA suggests that the representations being manipulated by the brain are semantic pointers (SPs), which are compressed neural representations that can be identified with vector variables such as x and y above. In the context of action planning, however, we need to manipulate symbol-like, structured information using SPs. Consequently, the forms of compression that are most relevant are identified by Vector Symbolic Architectures (VSAs; Gayler, 2004). VSAs are a set of mathematical formalisms that enable structured collections of symbols to be represented as high-dimensional vectors. For example, a symbol corresponding to the concept of $KETTLE$ could be defined in a VSA as a 500-dimensional vector (i.e. a distributed representation). These vectors can be randomly chosen (as they are here), or they can be chosen such that similar terms ($KETTLE$ and POT, for example) correspond to similar vectors.

To encode structured combinations of vectors, VSAs introduce a compressive binding operation. Different VSAs choose different binding operators, and for our work we use circular convolution, written as $\otimes$, meaning that this particular VSA uses holographic reduced representations (HRRs; Plate, 2003).

To give an example of how structured information is encoded using this operator, suppose we want to represent the knowledge that kettles tend to be used to boil water and tend to be located in kitchens and staff lounges. We might represent this as: $KETTLE = LOCATION \otimes (KITCHEN + STAFF\_LOUNGE) + GOALS \otimes WATER\_BOILED$. Importantly, VSAs also define an inverse operation: given an element of the structure, we can determine the associated representations by computing, e.g., $KETTLE \otimes LOCATIONS^{-1}$, which is approximately equal to $KITCHEN + STAFF\_LOUNGE$.

These VSA operations can be computed within the SPA to create structured semantics pointers. Moreover, the SPA allows such semantic pointers to be routed between different subsystems and manipulated in various ways. For instance, sensory areas can transform stimuli into appropriate conceptual SPs, and motor areas can take SPs representing actions and transform them into a sequence of muscle movements (Eliasmith, 2013). In order to implement these kinds of transformations, the SPA includes a model of the cortex-basalganglia-thalamus loop that performs (cognitive) action selection. Connections between cortex and the basal ganglia compute utilities over a set of possible actions. The basal ganglia identify the highest utility value, and pass this information on to the thalamus, wherein all of the neurons corresponding to actions with lower utility values are suppressed.

Models constructed using the SPA therefore define a set of cognitive actions that can be performed (note that these are distinct from the physical actions discussed throughout the rest of this paper). Each action is defined in terms of a set of effects E_i (e.g. the routing of an SP from one subsystem to another), and a utility function U_i that indicates when the action ought to be performed. For example, the following action results in the contents of a subsystem labeled *ultimate_goal* being routed to a subsystem labeled *immediate_goal* when a subsystem labeled *signal* represents an SP *PLAN*. This action will be selected by the basal ganglia if it has the highest utility of all actions.

$$U_i : signal \cdot PLAN$$
$$E_i : immediate\_goal \leftarrow ultimate\_goal$$

The NEF provides an efficient method for defining these actions, with each one requiring roughly 300 basal ganglia neurons to implement (Stewart et al., 2014).

A Neural Action Planning Model

Given that the set of actions needed to accomplish a goal depends on the state of the world, and that such actions modify the state of the world once performed, we define a simple simulated environment for our model to interact with. This environment consists of an arbitrary number of entities that have particular states and locations. For example, a 'ket-

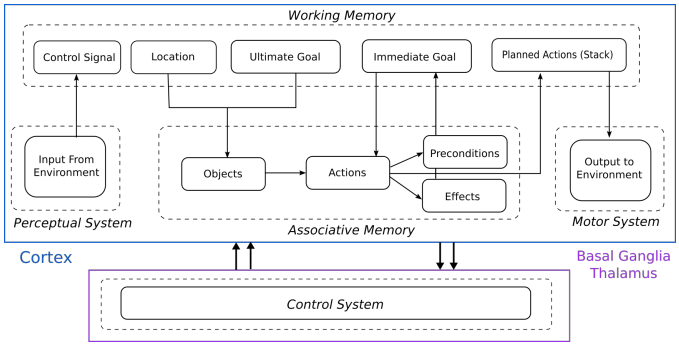

Figure 1: Functional architecture of the model. Regions surrounded by the dashed lines correspond to broadly individuated components responsible for perception, action, memory and cognitive control. Regions surrounded by solid black lines correspond to potentially simpler subsystems of these components. Arrows between subsystems indicate the main pathways through which the control system is able to manipulate the flow of information.

tle' entity might be located on a counter, and might have the state of being plugged in. At an implementational level, the environment is a program that the planner interacts with by sending commands that instruct the program to either return the state of an entity or execute an action. Importantly, actions can only successfully modify the state of the environment in particular circumstances. For instance, the action *BOIL_KETTLE* can only be successfully executed if the kettle contains water and is plugged in.

The planner itself comprises five main components: a perceptual system that monitors the the environment, a motor system that manipulates the environment, a working memory system that stores goals and planned sequences of actions, an associative memory system that matches certain locations and goals to certain actions and objects, and a control system that controls how all of these components interact with one another. Each component is implemented using the SPA, as described above. Figure 1 provides a high-level depiction of the model's functional architecture.

Functionally, the model is provided with input in the form of SPs representing a location and an "ultimate goal" (or "prior intention", Jeannerod, 2006), along with an input that signals the model to start planning. The location and goal representations are mapped by an associative memory to an SP representing a set of objects that are relevant to accomplishing the given goal in the given location. For example, given the location *KITCHEN* and the goal *WATER_BOILED*, the model will represent the objects *TAP* and *KETTLE* as being relevant.

Next, this object representation and the goal representation are mapped by an associative memory to a representation of an appropriate action, which is then stored in working memory as the final item in the action plan being constructed. Here, the action would be *BOIL_KETTLE*, which, simplifying somewhat, would be added to the plan via the following cognitive action:

$$U_i : control\_signal \cdot GET\_ACTION$$
$$E_i : stack \leftarrow stack \otimes PUSH + action$$

where *PUSH* is a random SP that is used to bind action SPs to particular positions in a structured representation of an action sequence.

Since the action can only be executed in specific circumstances, an associative memory is used to map it to a representation of a set of preconditions that the environment must satisfy. The immediate goal of planning is then updated by adding in these preconditions and subtracting out the effects of the planned action. This operation is performed by a cognitive action of the form:

$$U_i : control\_signal \cdot SET\_GOAL$$
$$E_i : i\_goal \leftarrow i\_goal - effects + precons$$

where *i_goal* and *precons* abbreviate the subsystems labeled 'Immediate Goal' and 'Preconditions', respectively.

This whole process of finding an appropriate action and updating the immediate goal of planning is repeated until an action whose preconditions are satisfied by the environment is found. At this point, perceptual feedback from the environment prompts the model to begin executing the actions in the planned sequence. First, the most recently added action in the planned sequence is routed to the planner's motor system, where it is subsequently presented as a command to the simulated environment. Then, the action just executed is removed from the planned sequence and the process repeats (i.e. the next action in the sequence is routed to the motor system etc.):

$$U_i : control\_signal \cdot POP\_STACK$$
$$E_i : stack \leftarrow stack \otimes PUSH^{-1} - motor$$

The representation of the action sequence in working memory is the VSA equivalent of a stack, and the the process of executing actions amounts to popping items off of this stack and routing them through the motor system as commands to the environment. However, because the SPA makes use of a compressive binding operator, the stack is not perfect (Plate, 2003). As more actions are added to a planned sequence, the likelihood of recovering all of the actions drops considerably. For this reason, our model is designed to execute all of the actions that can be recovered from the planned sequence, after which it begins to re-plan in light of the changed environment.

The fact that the model is able to switch from acting to planning in this manner is important for enabling it to recover from errors. For example, if the planner chooses an incorrect action representation at a given point in the planning process, or fails to execute items in the correct order, it will then re-plan and correct its mistake. The process of re-planning in this way is typically successful because the model relies on

the perceptual feedback from the environment when deciding whether or not the preconditions of a particular action are satisfied.

We emphasize that the model does not directly implement the above logic or symbolic variables. The model consists of 341380 spiking neurons, with synaptic weights that are optimized to approximately perform this information processing. An implementation of the model is available online at https://github.com/pblouw/action-planning/

Results

In what follows, we report results concerning the consistency with which our model is able to successfully perform a task over numerous simulations involving different neuron parameters. We also report results concerning the scalability of the model to situations involving large numbers of possible objects and actions.

First, however, we discuss the results of an example simulation to provide greater insight into the behavior of the model. As shown in Figure 2, the model plans by chaining backwards through a sequence of actions and continually updating its immediate goal. Once an action with preconditions satisfied by the environment is added to the plan, the model ceases planning and begins acting (this can be observed in the change to the Control Signal representation in Figure 2). Once no more actions can be extracted from the current plan, the model stops acting and proceeds to re-plan until it achieves its ultimate goal.

Goal Completion Analysis

We first test the model by assessing whether or not it is able to consistently accomplish its ultimate goal when allowed to interact with the environment. Each row of Table 1 reports the results of an experiment in which the model's behavior is simulated for up to 4 seconds over 50 trials. Each experiment involves setting the environment to an initial state that requires the planner to perform an increasing number of actions. For example, in the first experiment, only two actions need to be performed to accomplish the ultimate goal of boiling water. Each trial involves a unique instance of the model, in that a new random seed is used to generate all of the LIF neuron parameters (each instance of the model is accordingly analogous to a unique experimental subject). A trial is deemed successful if the model achieves the ultimate goal within the simulation time. We report the percentage of successful trials, along with the average time needed per trial to achieve the goal. The time needed to achieve the goal typically varies due to the differences in the number of actions the model is able to retrieve from a planned sequence. Occasionally, the model also makes an error that forces it to re-plan so as to accommodate an unexpected environmental state. Note that the environment model changes instantaneously, so these time scales correspond only to the neural model's internal processing time. Overall, our results indicate that the model is robust to changes in the environment that increase the number of actions required to achieve the goal.

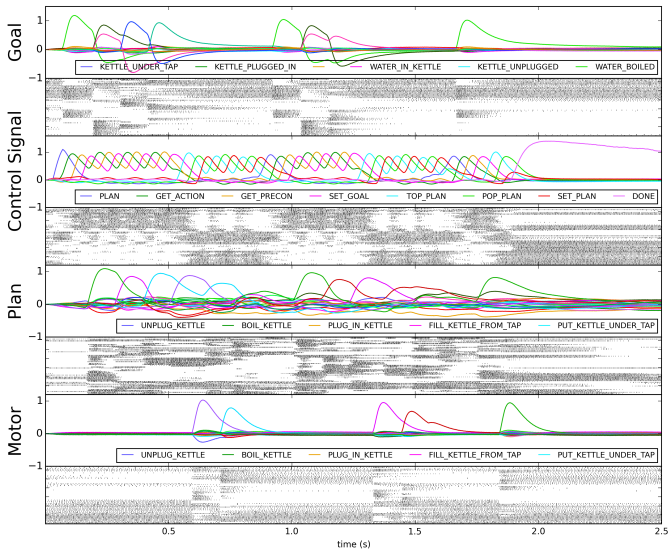

Figure 2: A sample run of the model interacting with the environment. Each colored plot depicts the similarity between the representational state in the indicated component of the model and a set of known representational states, shown under each plot. Underneath each representational plot is the spiking activity from a set of randomly sampled neurons from the labeled model component.

Table 1: Performance Analysis of Planner over 50 Trials

Sequence Length	Success Rate (%)	Average Time (s)
2	94	0.48 (SD 0.17)
3	98	0.90 (SD 0.20)
4	94	1.40 (SD 0.51)
5	94	1.92 (SD 0.64)

Scaling Analysis

Given these successful results, we decided to test whether the action-selection mechanism of this model would work with human-scale knowledge bases. Our model selects an object or action from an associative memory if its SP has a high inner product with an SP that represents the current context (e.g. location; goal). An unrelated item is unlikely to be accidentally selected, because random SPs in high-dimensional spaces tend to be nearly orthogonal. However, if the memory contains a very large number of entries, the total probability of an incorrect selection may become problematic. We explored this issue in simplified HRR models with no neurons.

Figure 3 illustrates how such confusion can arise. An associative memory was created with 25000 500-dimensional object HRRs, each of which contained lists of goal and location vectors of varying length (drawn from a Poisson distribution with a mean of 2). The associative memory was then queried with random goal/location combinations. The inner products

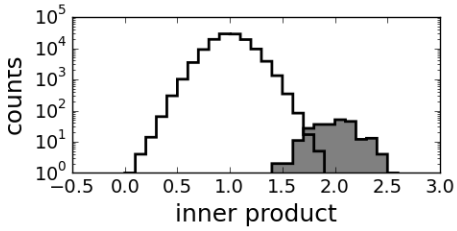

Figure 3: Inner products of 500 random goal/location context HRRs with 25000 object HRRs (each with two goals and locations on average). Histograms of inner products of queries with partially matching (open), and matching (filled) object vectors. There is some overlap between partial and full matches due to the large number of partial matches (note the log scale). The non-matches (not shown) have a mean of zero and do not overlap the matches.

of these queries with the associative memory entries was distributed around two for correct matches, i.e. for entries that actually contained both the specified goal and the specified location (bottom panel), and around zero for entries that did not contain either the goal or location (top panel), with no overlap. However, there was some overlap between matches and partial matches (which contained either the goal or location but not both). Because there are a large number of partial matches, substantial confusion could arise if a small fraction of them overlap.

Figure 4 estimates the effect of these matching properties on our model. The top panel shows average numbers of objects (out of 25000) that match random combinations of locations and goals (as a function of the mean numbers of goals and locations per object, which were set equal). Our model would not perform well if this number rose above about ten, because the inner product of an HRR x with a sum that includes x becomes noisy if the sum is large (Plate, 2003). In these simulations, locations were drawn from a list of 250 and goals from a list of 1000. The numbers of objects, goals, and locations were meant to correspond roughly to the numbers of these things that are familiar to a person. No explicit category structure is imposed on the object representations, though the fact that these representations draw on a shared stock of locations and goals yields varying degrees of representational similarity.

The center panel shows the precision, i.e. (true positives) / (true positives + false positives), of object selection with various HRR dimensions, over 500 random queries in each condition, with the threshold set below at least 90% of the true positives. Higher HRR dimensions improve precision.

Finally, the bottom panel shows the precision of action selection (with 10000 possible actions), using queries that combined desired effects (from 2500 possibilities) with sums of object vectors (including false positives) from the object queries. The precision is fairly high with higher-dimensional vectors. The consequence of an error would be to plan ei-

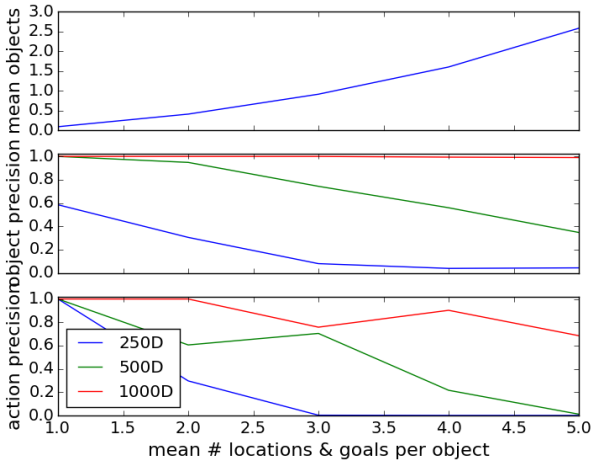

Figure 4: The scaling effects of the average numbers of locations and goals per object concept. Top: Average numbers of objects matching goal/location queries. Middle: Precision of object selection for HRRs of 250, 500, and 1000 dimensions. Bottom: Precision of action selection based on single effects and sums of object vectors from the queries in the middle panel (including false positives). Large HRR vectors are required for high precision with large numbers of object, goal, etc. concepts.

ther an irrelevant action or an action that requires an object that isn't available. One likely reason precision is lower for action selection than object selection is that multiple objects are selected in the first step, and their sum has a somewhat noisy inner product with the individual-object components of action vectors in the second step.

Overall, these results demonstrate that the neural model is likely to function effectively while picking from 10,000 actions related to 25,000 objects. Preliminary simulations indicate the neural model is able to scale to at least 5000 objects with no loss of performance in the kettle boiling scenario. However, additional neural simulations are needed to fully confirm the scaling analysis provided here.

We also found that somewhat greater precision could be obtained by restructuring action queries to include effects and locations, rather than effects and objects. To obtain these results, we built a memory of 10000 actions, each with one effect and a Poisson random number of locations. In this case we performed each query by randomly selecting an action with one or more locations, and querying with its effect and its first location (so there was always at least one correct match). The precision of these queries was somewhat better than that of the affordance-based queries.

This simple analysis does not account for a number of factors, including (for example) correlations between goals and locations. However, it suggests that the action-selection mechanism of our spiking model is likely to scale to relatively

complex environments.

Discussion

The main contribution of our work is to present a neurally plausible model of relatively domain-general action planning. The model is able to plan both quickly and effectively, and it is robust to various changes to the planning environment. The representational properties of the model, moreover, indicate that it is capable of scaling to naturalistic planning environments in which there are vast numbers of potential actions that could be relevant to accomplishing a particular goal. This is an important feature given the degree to which many existing models in action planning literature are restricted to highly specific problem domains (e.g. the Tower of Hanoi puzzle). Interestingly, the scope of our model is closely related to deficits in ideational apraxia (Zadikoff & Lang, 2005).

One potential concern about our model is that it is only able to produce short sequences of 1-3 actions before re-planning (i.e. the stack that stores planned sequences of actions degrades easily). This seems unrealistic in the context of a task such as water-boiling, which most people do with minimal conscious reflection. Some kind of sequence consolidation for routine actions is clearly needed (Taatgen & Lee, 2003; Cooper et al., 2014). However, we are not arguing that the planning performed by our model is reflective of conscious deliberation, in which case the time course over which planning occurs is not implausible.

Finally, given some of the limitations of our model, an important direction for future work concerns grounding it in a richer environment. One possibility would involve integrating the model into a robot that operates in a real kitchen. Among the many practical challenges this would entail, a key problem concerns recognizing object states using a model of the visual system.

Acknowledgments

This work was supported by a NSERC Discovery Grant and an Ontario Graduate Scholarship. We thank Terry Stewart, Ashley Kleinhans, Renaud Detry, Benjamin Rosman, Nasim Mortazavi, and Serge Thill for helpful discussions.

References

Andersen, R., & Gui, H. (2009). Intention, action-planning, and decision making in parietal-frontal circuits. *Neuron*, *63*(5), 568-583.

Cisek, P. (2012). Making decisions through a distributed consensus. *Current Opinion in Neurobiology*, *22*(6), 927-936.

Cooper, R., Ruh, N., & Mareschal, D. (2014). The goal circuit model: A hierarchical multi-route model of the acquisition and control of routine sequential action in humans. *Cognitive Science*, *38*, 244–274.

Daw, N., Niv, T., & Dayan, P. (2005). Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature Neuroscience*, *8*(12), 1704–1711.

Eliasmith, C. (2013). *How to build a brain: An architecture for neurobiological cognition*. Oxford University Press.

Eliasmith, C., & Anderson, C. (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT Press.

Eliasmith, C., Stewart, T., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., & Rasmussen, D. (2012). A large-scale model of the functioning brain. *Science*, *338*(6111), 1202- 1205.

Fikes, R. E., & Nilsson, N. J. (1971). STRIPS : A New Approach to the Application of . Theorem Proving to Problem Solving '. *Artificial Intellegence*, *2*, 189 – 208.

Fincham, J., Carter, C., van Veen, V., Stenger, A., & Anderson, J. (2002). Neural mechanisms of planning: a computational analysis using event-related fMRI. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(5), 3346–3351.

Gayler, R. W. (2004). Vector Symbolic Architectures Answer Jackendoffs Challenges for Cognitive Neuroscience. *arXiv preprint cs/0412059*.

Goel, V., & Grafman, J. (1995). Are the frontal lobes implicated in "planning" functions? Interpreting data from the Tower of Hanoi. *Neuropsychologia*, *33*(5), 623–642.

Jeannerod, M. (2006). *Motor Cognition: What actions tell the self*. Oxford University Press.

Krüger, N., Geib, C., Piater, J., Petrick, R., Steedman, M., Wörgötter, F., ... Dillmann, R. (2011). Objectaction complexes: Grounded abstractions of sensorymotor processes. *Robotics and Autonomous Systems*, *59*(10), 740–757.

Plate, T. (2003). *Holographic reduced representations*. CSLI Publications.

Russell, S. J., & Norvig, P. (2003). *Artificial intelligence: A modern approach* (2nd ed.). Pearson Education.

Shallice, T. (1982). Specific impairments of planning. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *298*(1089), 199–209.

Stewart, T., Choo, F.-X., & Eliasmith, C. (2014). Sentence processing in spiking neurons: a biologically plausible left-corner parser. In *Proceedings of the 36th annual conference of the cognitive science society* (p. 1533-1538). Cognitive Science Society.

Taatgen, N., & Lee, F. (2003). Production compilation: a simple mechanism to model complex skill acquisition. *Human Factors*, *45*(1), 61–76.

Wolpert, D., & Landy, M. (2012). Motor control is decision-making. *Current Opinion in Neurobiology*, *22*(6), 996-1003.

Zadikoff, C., & Lang, A. E. (2005). Apraxia in movement disorders. *Brain*, *128*(7), 1480–1497. doi: 10.1093/brain/awh560