Transitioning From Motivated to Cognitive Agent Model

James Graham and Janusz A. Starzyk
Russ College of Engineering and Technology
School of Electrical Engineering and Computer Science
Ohio University, Athens, OH 45701 USA
{jg193404, starzykj}@ohio.edu

***Abstract*—This paper reports on the transition from a reactive motivated agent model to a cognitive agent model based on motivated learning and goal creation. The paper details the current state of our research on the cognitive agent and its implementation in a virtual world environment. In particular, we show the virtual simulation and the agent's decision making process as it transitions from a parallel to a sequential implementation, as well as several of the reasons for doing so. In addition, results of testing both models on the same simulated environment are compared. The results show that the new model performs more efficiently than the old one. Furthermore, our planned future work on cognitive agent is presented.**

Keywords—motivated learning; cognitive intelligence; semantic memory; attention switching; sequential thought process

I. INTRODUCTION

The creation of a viable cognitive architecture can be considered a holy grail for researchers involved in machine learning. To date, a number of architectures have been proposed and/or implemented with varying levels of success, benefits and detractions. Examples of some current, well known, cognitive architectures include SOAR [1], ACT-R [2], Icarus [3], LIDA [4], Polyscheme [5], and CLARION [6]. The biggest issue with many of these systems, and the one we are trying to avoid, is the reliance on pre-defined scripts, heuristic rules, and general lack of true autonomy. SOAR, for example, while a very versatile platform is purely symbolic. Although its capabilities have been enhanced with add-on modules, they have the appearance of a band-aid over the real problem, which is lack of grounding (i.e. intrinsic understanding of symbols). ACT-R is part of a family of architectures that have been used to model human cognitive processes. However, it is primarily a theoretical modeling platform designed to help researchers understand human intelligence; it is not a full platform that is designed with the goal of developing intelligence. The Icarus architecture focuses on perception and action over things like abstract problems solving. Simply put, Icarus lacks the architectural components that would allow abstract thinking. All these methods either have to rely on predefined goals or predefined rules. LIDA architecture emphasizes a "Global Workspace" as envisioned by Baar's Global Workspace Theory (GWT). The LIDA model tries to model cognition in biological systems, from low-level perception and action to high-level reasoning, however, it relies on cognitive "atoms" and "attention codelets", which are not well defined, since not much is known as to how they can be implemented. Polyscheme is an architecture designed to integrate multiple representations and inference schemes into a single system capable of thought. It combines several specialized modules, each designed with a specific role, with their own handcrafted data structures and algorithms. While the system is fairly flexible and the modules collectively choose their focus, they can be considered to be over-specialized. It is one thing to learn specialization over time; it is another to have it pre-programmed by a designer. This can lead to artificially hobbled learning. Lastly, CLARION unlike other cognitive architectures, attempts to discover explicit rules via inductive analysis of what has been implicitly learned, which typically means extracting rules from an artificial neural network.

While there are issues with all of these systems, this should not be taken to mean they are not useful, only that they are incomplete. We hope that the design and implementation of our own cognitive architecture fills some of these holes and allows our agent to operate in a more capable and truly autonomous manner.

We began our research by looking at what drives intelligence. How does intelligence create its goals? How do its needs affect its learning processes? This led us in the direction of goal creation internal to the agent and Motivated Learning (ML). Motivated Learning is an approach toward the creation of intelligent agents where the internal motivations of an agent, created by either external reward or satisfying other motivations, may take precedence over externally set goals. This is in contrast to reinforcement learning (RL), which works by maximizing external reward by learning approximating value functions. While RL may deviate from its pursuit of value function maximization to occasionally perform random actions to assist its exploratory process, this behavior usually becomes less common as time progresses. Reinforcement learning is primarily reactive, and if we want to develop a human-like agent, we need to go beyond purely reactive cognition. And while ML is always controlled by its underlying motivations, the overall ML agent architecture is able to build on them and set its own objectives and motivations. By abandoning the search to maximize externally set objects the agent is able to learn new relationships, learn new motivations,

and build a system of internal rewards that guide it in its interaction with the environment.

In previous publications [7][8][9], we presented the idea of motivation being the underlying force behind a cognitive agent's operation. Motivation itself is expressed through needs or drives present in the agent. An agent starts with one or more "primitive" needs, and as it learns to fulfill them, the agent creates many of its own abstract needs based on its accumulated knowledge and solutions to the more primitive needs. The need for money is an example of an abstract need in humans. Money by itself has no value and does not provide for basic human needs. Without the knowledge of money's uses and its ability to buy the goods and services that support basic needs such as food and shelter, it would be little more than a bunch or paper or metal coins of no significance to us.

In more recent work we have advanced our understanding of how motivation works in a complete cognitive model and have worked toward giving additional structure to our cognitive model [10][11][12] by adding components such as memory, planning, and attention switching. This model and the more simplified version that we are currently implementing are discussed in the next section.

II. Background

Our work in cognitive models began with the investigation of memory and how goals were formed [9]**Error! Reference source not found.**. The initial work on goal creation was concerned with how a cognitive agent develops its own goals and motivations. From the creations of goals, we shifted focus to underlying motivations, hence, the focus on Motivated Learning. With the more general focus on motivated learning, other elements of the cognitive model became increasingly relevant and we started to try to develop a more complete picture of the cognitive model to which our end-goal agent would subscribe. The initial theoretical model is discussed in [9][10][14].

A. Comprehensive Cognitive Model

More recently, with our work on opportunistic motivated learning [8][15], we have made a few changes to further develop the cognitive model, with the current version shown below in Fig. 1.

This updated cognitive model has more detail added in the form of additional functional components and links that represent information passed between them. Discussion of the details of the links in the full model is not attempted due to lack of space. However, details of the links for the reduced version are presented in Part B of this section and may be extrapolated to the "Full" model of Fig. 1.

In this model we have divided the agent's functions into several distinct blocks consisting of: working memory (essentially the central executive), semantic and episodic memory, motor control, motivation and goal creation, subconscious and WTA attention switching, and sensory/motor processing. Information processing and learning in Fig. 1 is organized from the bottom up, with the bottom functional components being in the subconscious realm and performing parallel operations on the input/output data. The components toward the top of the figure process conscious observations, memories, thoughts and plans and are characterized by sequential processing.

In terms of the function of each block, the sensory motor area is where most of the initial low level processing occurs. In more advanced implementations this would be the location where raw sensory input is processed into useful information through feature extraction and object recognition, and fine motor control signals are generated for the agent's actuators. Our current implementations are entirely virtual and the bulk of our focus is on the cognitive model, so our I/O is mostly symbolic and possesses no more than feature level resolution. Hence, there is little actual processing currently occurring in these modules.

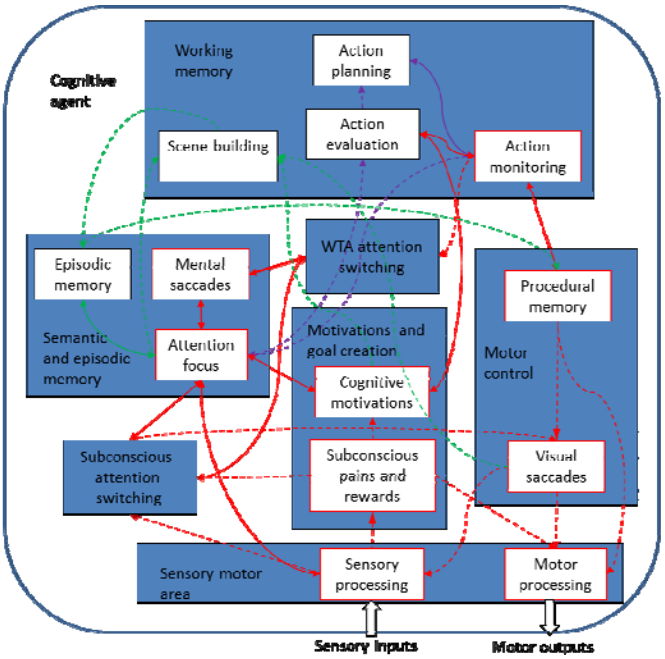

Figure 1. "Full" cognitive model.

Subconscious attention switching is a module that handles changes in the environment, new objects, unexpected events, or sudden changes in the agent's motivation state, and brings them into the agent's attention focus if needed (where the attention focus is the current object or concept selected for processing by the working memory.) It essentially controls what we may call "involuntary attention switching," with little or no conscious control. There may be some conscious control implemented in the form of conscious thresholding of environmental signals, so that the agent is able to more adeptly filter out background noise, much as we might ignore something going on in the background to focus on an important task.

The motivation and goal creation controls the agent's motivation and behavior. It is involved with the maintenance of primitive needs and the creation of new abstract needs. This block receives input from sensory processing to evaluate pains and establish the level of motivations. It interacts with the attention focus to direct it to the needs and can interact with the planning block for action evaluation. It supports scene building

by providing information about the significance of each scene element and collectively of the entire scene. In addition, subconscious pains can trigger reactive motor response to reduce the pain level. The agent's motivations have already been discussed in great detail in our previous work, so discussion here is limited to how it interacts with other functional blocks. However, motivation remains an important part of the model.

The semantic and episodic memory blocks consist of the two memories and attention focus / saccading components that direct the focus of the memory blocks in the direction needed by the agent. For the semantic memory this typically means saccading through semantic knowledge related to the current dominant motivation in the hope that it will be able to provide information to the working memory and allow it to put together a viable plan. Episodic memory, also directed by the attention focus will pull up salient episodes related to the current focus for the agent's consideration and review. Attention focus selects a concept in focus for consideration by working memory. It is supported by the mental saccades mechanism [13]. By using attention focus, parallel processing of sensory information, internal need signals, and associations created in the semantic memory are changed to a sequential, cognitive processing.

Semantic memory is essentially a database of interlinked and associated knowledge that the agent has accumulated through learning and can access if needed. Episodic memory, on the other hand, is a memory of the agent's experiences. It provides context, spatial and temporal relations, as well as event significance and emotional attachment. Episodes are formed in the working memory, supported by scene building and scene significance mechanisms.

WTA attention switching mediates the attention switching functionality between the subconscious attention switching (which takes priority), action monitoring (planning activities), and background thought processing (general thinking). Its function is to switch attention from the current attention focus to a new one, depending on environment conditions and internal processing of information. It responds to signals from subconscious attention switching, action monitoring and mental saccades. More information on our conceptualization of these memories can be found in [13],[16].

Working memory, or the central executive, is where most of conscious thought processing occurs. It is involved in putting together "plans" for resolving the agent's needs, managing these plans, and handling the results. It also contains what we refer to as the scene building module, which manages the creation of scenes. Working memory restores the observed scene by cognitively recognizing the main objects, their location and relationships between them. It receives support from the motivations and goal creation block to establish scene significance and provides translation from pixel level visual input to cognitive representation. Scene building is important for constructing episodic memories, performing searches, spatial orientation, and map building.

Working memory is also involved in action evaluation, planning and monitoring. First, a potential action is evaluated by recognizing what needs (if any) it can satisfy. Second, after analyzing several options, the best one is selected for potential realization. Third, in action planning, conditions necessary to complete the planned action are checked. Finally an action is initiated and its progress is monitored. The system must have the ability to return to an interrupted action if action evaluation determines that its completion is useful.

Last, is the motor control area, which contains the procedural memory and visual saccade control. It handles, in conjunction with other components, the creation of procedures (which greatly simplify the execution of complex actions by reducing the amount of cognitive overhead needed) and the execution of the plans generated by action planning.

Please note, that this is only a very brief summary of the complete model and is only given to provide some context for the reduced version of the model that is discussed next. For a more detailed overview of the model see [8] or[14].

B. Reduced Cognitive Model

Fig. 2 shows the functional organization of the reduced cognitive model that is discussed in this paper. The reduced version of the cognitive model removes and simplifies several of the components from the complete version of the model in an effort to create a "stepping stone" toward a more complete and fully functional implementation.

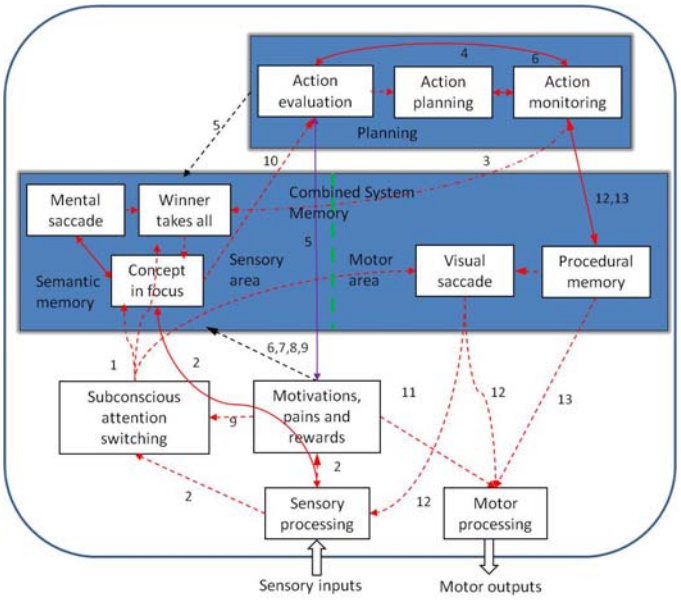

Figure 2. Simplified cognitive model.

First, notice that there is no cognitive motivation component in Fig. 2. This is because the higher level cognitive processing of our agent is not yet developed and it is necessary to use subconscious motivations (as determined by the pains and rewards block) working in the sequential model, along with the attention focus mechanism.

Next, WTA attention switching has been folded into the Combined System Memory and episodic memory has been removed. Episodic memory was removed for now since it is desirable to get simplified versions of the most important structures operational before adding more complexity. The

agent can still operate without episodic memory; it will simply be more limited in its reasoning capabilities, since it will lack much of a historical record of its activities and their significance. However, since the actual cognitive functionality is currently very limited, this is a moot point. Scene building has also been removed, since there is less need for it without a dedicated episodic memory. However, it is worth noting that the action planning component retains an internal representation of the environment which it maintains as it receives updates via memory saccades.

Procedural memory, saccade handling, and attention switching (with the exception of the involuntary attention switching mechanism) are all folded into a simplified Combined System Memory structure. Once the initial structure has been implemented and tested, these components will be separated and their full functionality restored.

In Fig. 2, we display several numbers associated with the links connecting the various modules. The lines represent information that is passed to/from the various major functional blocks, and help to provide understanding of the central role that the semantic memory plays in the system operation. Most of these signals will be used by the memory to trigger memory nodes, update node activation levels, perform saccades, or attention switching.

Next, we will discuss the role individual links play in the simplified cognitive model. Please note, that this is only a brief overview of the links and is not intended to be an in depth discussion of the interconnectivity of these modules, but is presented to provide some insight into how the simplified model operates. Tracing these links we can better understand how central the combined system memory component is, and how complex the model remains despite the simplifications we made from the full model.

Data sent between functional blocks in Fig. 2 are described according to line identification number as follows:

1. Subconscious attention switch data – drives mental and visual saccades to switch agent's attention due to a noticeable change in the environment or motivations.

2. Processed environment data – provides needed information to the primitive pains and subconscious attention switching and activation signals to the semantic memory.

3. "Next" signal from the planning block that tells the memory that it is ready for another saccade.

4. Provides action status information regarding the progress on the current action, whether it is complete, and if it is ready to be evaluated.

5. Request to estimate expected pain reduction by the evaluated action.

6. Pain/Need change information; important when actions are being evaluated.

7. Environment change information (used in calculating abstract needs and adjusting interconnection links weights in both the motivation and the memory blocks)

8. Motivation levels; used as a priming mechanism for the memory.

9. Ratio estimation for change in pain resulting in per unit resource use.

10. Winning saccade data from memory and associated information.

11. An involuntary reflex warning signal from the pain management to motor processing.

12. Sensor signal that specifies the action/saccade target

13. Motor command signal; a symbolic signal that typically represents a motor command or procedural sequence.

Most of the reductions/removals in this simplified model are done only because it is necessary to "work our way up". By implementing the full model one step at a time we minimize the amount of work we have to redo as our models evolve. This allows the evolution of our models and theories to occur as we obtain results, better understand the limitations of the current model, and improve its functionality through testing various environment scenarios.

As has already been implied, this reduced model lacks much of the higher level functionality present in the "Full" cognitive version. Obviously, when advancing the model, some components need to be implemented simultaneously. This is the case as we advance from the parallel Opportunistic Motivated Learning (OML) model, presented in[8],[15], to the sequential cognitive model presented here. For example, in order to add a semantic memory, we had to add attention switching, action monitoring, and a more advanced, distinct planning mechanism.

III. Implementation

Using the "Full" cognitive model discussed above in section II.A as a basis, we have implemented a reduced version of the model (as discussed in II.B) that lacks much of the complexity present in the "Full" model. For example, we do not have separate procedural or episodic memories. The lack of procedural memory is because we are designing our agent for a purely symbolic environment, significantly limiting its choices. This initial implementation is noticeably simplified, perhaps even more than Fig. 2 suggests. For example, actions remain in the form of sensor/motor pairs (target a sensor object and perform a motor action on it) and we have yet to implement multi-step sequences of actions. In fact, we have deliberately tailored this initial sequential implementation to match the capabilities of our parallel OML implementation [15] in order to compare them. The ability to compare two subsequent implementations is useful in this field of research where it can be difficult to effectively benchmark and/or compare the developed models.

To elaborate, we have designed the sequential implementation with the same simple initial environment and the sensor/motor pair setup that our earlier implementations used. However, we have progressed toward increasing the system capabilities such as using additional features and more complex environments. To provide some context, let us briefly compare the earlier parallel OML implementation with the current sequential implementation.

Table I summarizes the major differences between the two implementations of the motivated learning models. In Table I we note that in the sequential implementation the agent does not update abstract needs/motivations every cycle. This is because it only "updates" one resource at a time after attention was focused on this resource. On the other hand, primitive pains are updated every computational cycle since they are based on parallel and subconscious sensory inputs.

Table I. Implementation Comparison

Parallel OML implementation	Sequential Implementation
The agent is simultaneously aware of everything.	The agent is attention based, so it effectively "sees" only one object at a time.
All pains and motivations are updated every cycle.	Primitive pains are updated every cycle, but abstract needs are not.
No specific monitoring or evaluation blocks.	Has action monitoring and evaluation components.
Action choice based only on the state of the environment and the agent.	Associations and memories affect action choice in addition to the state of the environment and the agent.
Only capable of forming direct 1-1 associations.	Has semantic memory capable of forming hierarchical associations.

Abstract motivations/needs are updated only when the agent has had an attention switch to the associated resource or concept. This impacts nearly every facet of the agent's operation. For example, the agent does not have a true "real-time" mental representation of the environment, because its internal maps can only update a feature after it has been subject to a saccade. Nor can the agent evaluate its actions until it has had the chance to saccade to the objects that its action effects.

This means that the agent has to assume that its environment does not change drastically in a short period of time (a significant change would trigger an attention switch and update the cognitive agent's memory), and that evaluating the results of actions takes longer than in the parallel OML implementation.

We also note that the sequential model has a more complex memory structure. The OML model implements a reactive agent, and while it is capable of learning new relationships, the relationships have to be direct (not conceptual, amorphous, or inferred). However, the sequential model uses a semantic memory capable of developing associations between related nodes and forming hierarchical relationships between concepts.

Nevertheless, both agents currently use the same type of symbolic I/O. Hence, in order for the agent to function, information needs to be rendered into a symbolic and/or feature based format that the agent can understand. While we could design a system that processes raw visual input and learns to dynamically control its motor functions, which would be a major project by itself (and is an active research area for others). However, in our work we rely on symbolic I/O that is easily deliverable by video game virtual environments to simplify our task and accelerate progress in research on cognitive agents.

In the OML model, we essentially executed everything in parallel, and then polled the environment. In the sequential implementation, we have a more complex data processing structure. We still have to execute each "block" in a sequence due to the nature of the programming environment; however, there are more blocks and more data flowing between them.

The following list provides a simplified look at the execution of the main *Iterative Loop* of the program to give a rough idea of the data flow. (Note that not all of the data links are mentioned, just the main links associated with the primary functionality and data flow of the model.) The loop is either executed indefinitely, until the agent "dies", or for some predetermined number of iterations (at which point the agent's "memory" state can be stored for later resumption).

Iterative Loop

1. *Update Primitive pain* – updates primitive pains
    a. Receives pain information from the Environment (Sensory processing)
    b. Sends results to Motivations
2. *Update Motivations* – passes primitives and updates motivation levels
    a. Receives data from Primitive Pains, Action evaluation, and Memory (no direct link to environment/sensory)
    b. Sends data to subconscious attention switching and the sensory area of semantic memory
3. *Update attention switch* – updates subconscious attention switching to check for a visual or motivation interrupt
    a. Receives data from Sensory processing and Motivations
    b. Sends data to semantic memory
4. *Update memory* – performs memory saccade
    a. Receives data from Environment (sensory processing), Motivations, Subconscious attention switching, and Action Evaluation
    b. Sends data to Planning and Motivations (used to pass data from visual saccades for motivation updates)
5. *Update planning* – uses memory data to update plans
    a. Receives data from Semantic memory
    b. Sends data to Action evaluation
6. *Update actions* – checks action status/monitor and tracks action evaluation
    a. Receives data from Planning
    b. Sends data to the Environment (through motor control) and Semantic Memory
7. Perform the selected motor actions in the virtual environment – move the agent if needed, etc.
8. Check the environment – update environment state based on the agent's actions and the environment response.

End Loop

Note that the current implementation uses a pseudo-semantic memory that only mimics functionality of semantic memory. It was created to act as a placeholder for the semantic memory while we develop the sequential implementation. It possesses basic attention switching and visual saccading, as well as the ability to form simple associations. However, it cannot form more complex hierarchical relationships and

concepts as true semantic memory would. However, the fully functional semantic memory is not necessary at this time because of the simplicity of the current environmental scenario used to showcase the results. In the next section, we compare similar simulation runs for the previous OML implementation and our current sequential model.

IV. Results

In this section we compare the results of a pair of simulation runs for both the parallel OML implementation and our sequential model to provide a baseline for future work. In the simulation setup, both models were given the same simple environmental scenario as shown in Table 2 where each row represents one of the possible "correct" actions like "eat food". Each such "correct" motor action induces changes in the abstract pains. For instance, "eat food" reduces the hunger pain. Simultaneously, through the goal creation mechanism, a higher order need is established and an abstract pain related to the resource used to perform the desired action, increases. The agent must learn which action reduces what pain. In this scenario we have a single primitive pain, "Hunger," and 4 abstract needs. Additionally, there are 6 resources with which to interact and 6 possible motor actions. In summary, there are 36 possible actions, of which only 5 will have any positive impact on the agent's environment.

Table 2. Functional sensory motor pairs and their effects.

MOTOR FUNCTION	SENSOR OBJECT	REDUCES PAIN	INCREASES PAIN
Eat	Food	Hunger	Lack of Food
Buy Food at	Grocery Store	Lack of Food	Lack of Money
Withdraw from	Bank Account	Lack of Money	Overdrawn Account
Work in	The office	Overdrawn Account	Lack of job opportunities
Study at	School	Lack of job opportunities	-
Play with	Toys	-	-

Figs. 3 and 4, present the results from the OML simulation. The OML simulation performed 100,000 iterations in 168 seconds and was sufficient for the agent to learn its skills.

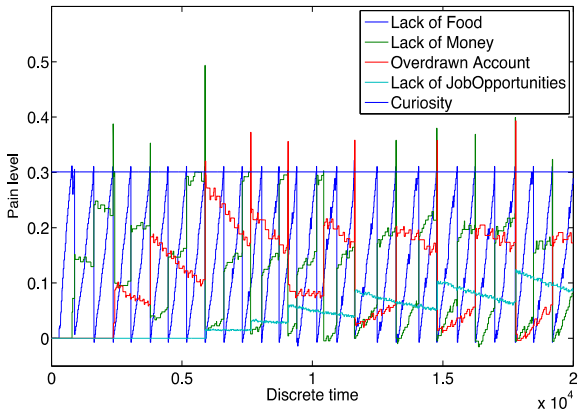

Figure 3. OML simulation pain results.

The figures only display the first 20,000 iterations to preserve the readability of the results, although, "Prim Hunger" has been hidden in Fig. 3. Its presence would interfere with visualization of the results, since it is a primitive pain and its change occurs much more frequently than the others. In Fig. 3, we see that the pains did not go much beyond the threshold of 0.3 before they were reduced, which indicates that the agent learned how to manage its needs and lower internal pain signals.

In Fig. 4, we can observe the impact of the pain/need responses on resource usage (with resource values normalized to 1.0). Notice that "Job Opportunities" did not fluctuate as much as most of the other resources because the agent did not need to restore its money supply frequently enough in the 20,000 iterations for the resource to drop significantly. However, as expected, we can see that drops in "Job Opportunities" directly correlate to jumps in Money supply.

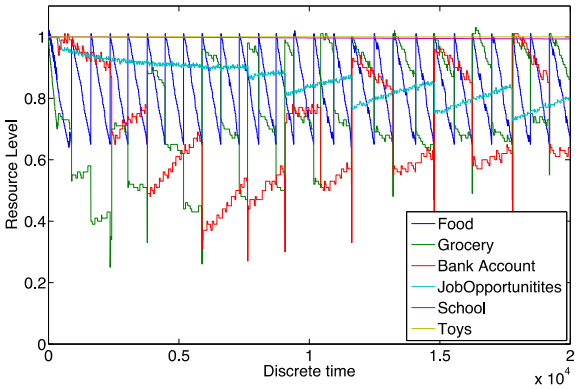

Figure 4. OML simulation resource results.

Figs. 5 and 6 depict the corresponding pain/resource results for the sequential model simulation. Execution of this program took 322 seconds and also ran for 100,000 iterations (although, again, we only display the first 20,000 iterations.) The sequential simulation took roughly twice as long as the OML simulation. This additional time needed by the sequential algorithm to make decisions makes perfect sense considering the additional computational overhead brought on by the use of the sequential model. However, since the 322 seconds used by sequential algorithm represents total time needed to make 100,000 decisions, it translates to 3.22 msec/decision, which is hardly a problem for real-time control of the robot in either the real or virtual world.

Note that in Fig. 5 the Hunger pain has been hidden to allow the lower level pains to be visible. Most of the pains are resolved (reduced below threshold) shortly after they pass the threshold. Similar behavior was observed in Fig. 3. In fact, shortly after 10K iterations the sequential agent appears to no longer rely on pain as a trigger for its actions and instead relies on its internal saccade mechanisms to present useful actions to the planning mechanism. This is implied because after 15K iterations the pains do not appear to reach values of much more than 0.15, well below the threshold of 0.3. This behavior of the sequential model is promising as it results in better pain management than in the OML model.

The amount of time needed to perform an action in the sequential model is greater due to its saccade to the selected solution, performance of the required actions, and then evaluation of the results, whereas in the OML implementation, evaluation and selection of the action happens within the execution cycles, which needs fewer processing steps.

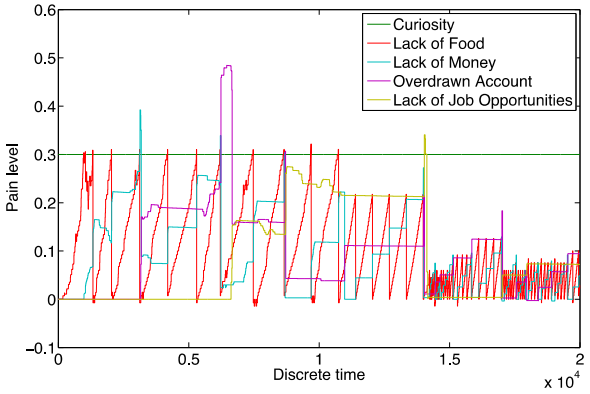

Figure 5. Sequential model simulation pain results.

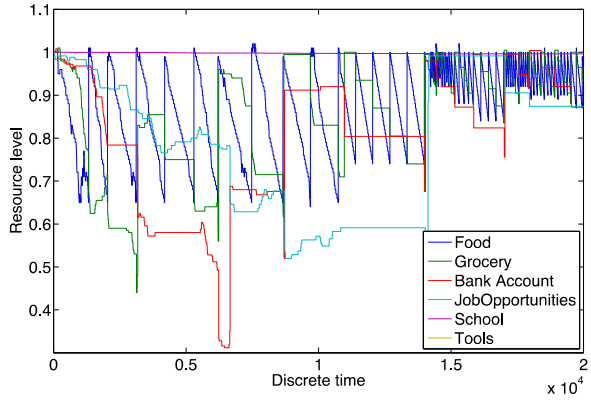

Figure 6. Sequential model simulation resource results.

More specifically, the motor command for "eating" in both simulations requires 2 cycles. In the OML simulation the action is selected and started during the first cycle, and evaluated on the start of the third cycle (during which a new action starts). In the sequential simulation, the agent has to switch attention to the dominant motivation, and then one or more mental saccades within the memory to provide the correct solution to the planning block. Then two more cycles are required to actually perform the action. At least one additional cycle is needed to evaluate the action. Hence, the sequential model's execution of the "Eat Food to reduce Hunger" action takes four cycles compared to the two cycles needed by the OML simulation. This 4 cycle count is a minimum for the sequential agent, since there could easily be other things happening in the environment or within the agent's saccade mechanism that could extend the time needed to select and execute the appropriate action.

This higher efficiency of OML model may be superfluous as it benefits from parallel vector processing. In a full cognitive system, where the parallel nature of sensory data will be significantly higher than in the simplified symbolic information, this advantage will disappear. In fact, we claim that the opposite may be true and that sequential systems will be able to process massive amounts of sensory data more efficiently, by focusing on changes in the observed scene and on those sensory inputs that it must cognitively recognize.

Despite the longer time required by the sequential model from the time it initiates an action to the time it finishes evaluating it, the sequential model appears to be more efficient in its operation. Some of this can be because even when it is performing an action the agent is still saccading and evaluating other actions. However, this concurrent processing behavior cannot be responsible for keeping the pain values below threshold as occurs at about 14K iterations or for faster cycling between pains.

The main reason for the sequential model's improved performance is that it exhausted its curiosity about the environment sometime between 11-14K iterations. When there is no pain above threshold, the pseudo-memory saccades to the next best option, related to the highest current pain. Ultimately, this caused the behavior observed in Figs. 5 and 6 after 14K iterations with the agent making optimum use of its time. This has the desirable effect of keeping overall pain values much lower. In a more complex environment, we might see this kind of behavior if the agent had fully explored its immediate vicinity. At such a point it could choose between moving elsewhere, or working to resolve its needs within the known area.

Disregarding the different rates of pain management in the two models we can see that their general behavior is similar and that they are both able to successfully handle their respective environments. Let us now analyze Figs. 4 and 6, which concern the resource levels in both simulations. There are two significant and noticeable differences between the two plots. The first one corresponds to the faster rate of pain handling in Fig. 5 in the sequential model with resources being restored at a greater frequency after reaching equilibrium at 14K iterations. The other difference is in the curves' variability character (for example the Money resource).

In Fig. 4, Money, after being depleted, is slowly regenerated in a series of several small steps, while later on, around 9K iterations, it experiences a large jump. This jump can be attributed to the agent finally learning the value of money. This does not account for the earlier gradual increase in the money supply. This increase is due to the agent having learned to restore money, but not knowing its specific usefulness (i.e. it's ability to bring grocery pain below threshold). In Fig. 6, we do not see much of this behavior at all. This is because the sequential model focuses its attention on the dominant motivation, and if curiosity doesn't dominate long enough and a more pressing goal oriented need wins in the planning block, a curiosity based action will not be executed. This is not the case in the OML algorithm where actions tend to take slightly less time and the algorithm can simultaneously "see" all of the available options.

V. Environment Simulation

So far, we have focused primarily on the models and their functions. Yet, while we have been improving our models, we have also been improving our environmental simulations. If you compare the environment we presented in [8] or [15] to the one depicted in Fig. 7 several improvements are noticeable. In addition to the direct reporting of pains and resources to the screen, we have also begun incorporating animations into the agent's action to better illustrate its interactions with the environment. Eventually, we hope to link animations together in sequences of motor control functions to effectively perform and display/represent more complex procedures.

Figure 7. Snapshot of simulation environment.

In Fig. 7 there are three panes indicating the state of the environment and the agent. The left pane shows the state of the resources in the environment. The middle pane indicates the agent's current action. And the right pane shows the agent's pain levels. Red bars in the right pane indicate that a pain is above threshold. While this environment simulation is currently only capable of running the OML simulation, we will have it ready for the sequential model soon.

Further improvements to the simulation will include using sequences of animations and more complex object representation. We plan to add more features to the objects. We also intend to add a focused view of the environment, since now the agent can "see" all of the objects around it at once. By reducing its field of view we will increase the element of exploration. This will likely be a priority once the agent becomes capable of more complex actions. We will also add elements of human-machine interaction, to allow the agent to learn and/or interact with human avatars.

VI. Conclusion

A viable cognitive architecture is needed in developmental robotics and mission critical autonomous robots capable of independent operation in dynamically changing complex environments. This paper presents the progress being made in this direction by chronicling the advancement of our motivated learning model and its implementation and providing a brief comparison of the results between the two most recent model implementations. Our motivated learning model has advanced significantly towards a fully cognitive model. We plan to further enhance its functionality and memory organization. The next step will be to implement the semantic memory, followed by the implementation of a basic procedural memory to perform complex action sequences. Subsequently, this will be followed by development of episodic memory and enhancement of action planning, evaluation and monitoring.

References

[1] J.E. Laird, "Extending the Soar cognitive architecture," in Artificial General Intelligence 2008, Memphis, TN: IOS Press, 2008, pp. 224-235.

[2] J.R. Anderson, D. Bothell, M.D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," Psychological Review, vol. 111, no. 4, pp. 1036–1060, 2004.

[3] P. Langley and D. Choi, "A unified cognitive architecture for physical robots," in Proc. 21st Nat. Conf. Artificial Intelligence, Boston, MA: AAAI Press, 2006, pp. 1469-1474.

[4] B.J. Baars and S. Franklin, "Consciousness is computational: the LIDA model of global workspace theory," Int. J. Machine Consciousness. Vol. 1, No. 1, pp. 23-32, 2009.

[5] N. Cassimatis and L. Nicholas, Polyscheme: A Cognitive Architecture for Integrating Multiple Representation and Inference Schemes, MIT Ph.D. Dissertation, 2002.

[6] R. Sun, "The importance of cognitive architectures: an analysis based on CLARION," J. Experimental and Theoretical Artificial Intelligence, vol. 19, no. 2, pp. 159-193, 2007.

[7] J. A. Starzyk, J. T. Graham, P. Raif, and A-H.Tan, "Motivated Learning for Autonomous Robots Development", Cognitive Science Research, v.14, no.1, 2012, p.10(16) pp. 10-25.

[8] J. Graham, J. A. Starzyk, D. Jachyra, "Opportunistic Motivated Learning Agents", 11th Int. Conf. on Artificial Intelligence and Soft Computing, Zakopane, Poland, Apr 29, May 3, 2012.

[9] J. A. Starzyk, "Motivation in Embodied Intelligence," in Frontiers in Robotics, Automation and Control, I-Tech Education and Publishing, Oct. 2008, pp. 83-110.

[10] J. A. Starzyk and D. K. Prasad, "A Computational Model of Machine Consciousness," International Journal of Machine Consciousness, vol. 3, No. 2, (2011) pp. 255-281.

[11] D. K. Prasad and J. A. Starzyk, "A Perspective on Machine Consciousness", Second Int. Conf. on Advanced Cognitive Technologies and Applications, Lisbon, Portugal, Nov. 21-26, 2011.

[12] J. A. Starzyk, Motivated Learning for Computational Intelligence, in Computational Modeling and Simulation of Intellect: Current State and Future Perspectives, IGI Publishing, ch.11, pp. 265-292, 2011.

[13] J. A. Starzyk and J. Graham "A Goal Creation System With Curiosity" 13th Int. Conf. on Cognitive and Neural Systems (ICCNS), Boston University, May 27-30, 2009.

[14] J.A. Starzyk, "Mental Saccades in Control of Cognitive Process", IJCNN, San Jose, CA, July 31 - August 5, 2011.

[15] J. Graham, J. A. Starzyk, and D. Jachyra, "Opportunistic Behavior in Motivated Learning Agents", submitted to IEEE Trans on Neural networks and Learning Systems, 2012.

[16] W. Wang, B. Subagdja, A.-H. Tan, and J. A. Starzyk, "Neural Modeling of Episodic Memory: Encoding, Retrieval, and Forgetting" IEEE Trans. on Neural Networks, vol. 23, no. 10, Oct. 2012, pp. 1574 – 1586.