Goal-Driven Autonomy and Robust Architecture for Long-Duration Missions (Year 1: 1 July 2013 − 31 July 2014)

Michael T. Cox
Wright State Research Institute
Beavercreek, OH 45431
University of Maryland Institute for Advanced Computer Studies
College Park, MD 20742
937.705.1029; 937.705.1095(fax); Michael.Cox@wright.edu

ONR Award #N000141310798
http://mcox.org/g-reason/summit-II

Abstract

During the most recent period of performance, the team has refined the major implemented functions of our MIDCA architecture at the cognitive level (Cox, 2013; Paisner, Cox, Maynord, & Perlis, 2014) and has started work on an analogous metacognitive cycle at the meta-level (Dannenhauer, Cox, Gupta, Paisner & Perlis, 2014; Perlis & Cox, 2014). We produced substantial progress on the knowledge-rich track of the Note-Assess-Guide interpretation procedure. Additionally we started an integration of MIDCA and the T-REX planning and scheduling system. This work explores the use of meta-level control for vehicle planning and behavior execution in dynamic environments. The initial data are encouraging and the path forward is favorable.

LONG TERM GOALS

The goal of this research is to articulate a computational foundation for robust long-duration agent autonomy and to provide a prototype implementation that exhibits flexible, goal-driven autonomy on an actual physical platform.

OBJECTIVES

The research team is developing an integrated theory of intelligent action, perception, cognition, and metacognition, is constructing a *Metacognitive, Integrated Dual-Cycle Architecture (MIDCA)* for this theory, and is applying an implementation of this architecture to scenarios in the domain of long-duration robotic behavior.

APPROACH

We have begun to integrate a robotic agent architecture with a metacognitive architecture by formalizing an abstract model of the robot control structure thereby enabling the meta-level to reason over representations that instantiate this model. The key idea is to implement a metacognitive mechanism that records traces of agent reasoning as it dispatches between functional modules in service of its existing goals. If these traces are structured in a representation that is understandable by

the meta-level, then it can reason about the agent's decision making as well as the agent's behavior. Control is then passed back through the individual modules in terms of new goal structures and subsequently to agent effectors.

WORK COMPLETED

The project has completed the first year of a four year schedule and has made significant early progress. We are on schedule for spending commitments and expect to reach our expenditure target. During the reporting period, we have moved along our schedule, implementing a set of components to comprise a mature implementation of the MIDCA architecture. This will lead to a subsequent effort in the second year of the project that implements both object-level (i.e., cognitive) and meta-level (i.e., metacognitive) control of autonomous systems. The end result is robust response to unexpected and fluid environments found in long-duration missions. Finally through a recent ONR DURIP equipment grant (#N000141310890), we have acquired three Baxter humanoid robots1 on which we intend to apply this research and demonstrate the efficacy of the integrated architecture on physical, unmanned platforms.

During this first year, we organized three events to highlight our research and to look for greater synergies in the larger research community. The first was a December Workshop on Goal Reasoning (http://mcox.org/g-reason) at the 2013 Conference on Advances in Cognitive Systems in Baltimore, MD. The second event was the NRL Goal Reasoning Summit held in April, 2014 at the naval Research Lab. The final was a second Goal Reasoning Summit (http://mcox.org/g-reason/summit-II) held at the University of Maryland Institute for Advanced Computer Studies during July, 2014. These meetings involved prominent researchers in academia, industry, and government and examined multiple technical issues relevant to this project.

We have organized the subsections below by the task decomposition from the original proposal (see also Table 4 in the Work Plan section). The research tasks are as follows.

1. Architecture Integration
2. Domain and Scenario
3. MIDCA Meta-Models (starts in year 2)
4. T-REX Robustness (starts in year 2)
5. GDA Learning (option – not funded)
6. Evaluation

We will briefly highlight the most important activity in each of the year-1 tasks. Tasks 3 and 4 were scheduled to start in year two and Task 5 was not funded. So we report on Tasks 1, 2, and 6.

Architecture Integration (Task 1)

The work performed during this current reporting period has resulted in a refinement of the MIDCA implementation. MIDCA consists of "action-perception" cycles at both the cognitive (i.e., object) level

[1] http://www.rethinkrobotics.com/baxter/

and the metacognitive (i.e., meta-) level (see Figure 1). The output side of each cycle consists of intention, planning, and action execution, whereas the input side consists of perception, interpretation, and goal evaluation. A cycle selects a goal and commits to achieving it. The agent then creates a plan to achieve the goal and subsequently executes the planned actions to make the domain match the goal state. The agent perceives changes to the environment resulting from the actions, interprets the percepts with respect to the plan, and evaluates the interpretation with respect to the goal. At the object level, the cycle achieves goals that change the environment (i.e., ground level). At the meta-level, the cycle achieves goals that change the object level. That is, the metacognitive "perception" components introspectively monitor the processes and mental state changes at the cognitive level. The "action" component consists of a meta-level controller that mediates reasoning over an abstract representation of the object level cognition.

The interpret phase of MIDCA has been the subject of much of our work, and is the focus of the experiments described in a subsequent subsection. It is implemented by two GDA processes that combine to generate new goals based on the features of the world the system observes. We call these processes the D-track, which is a data driven, bottom-up approach, and the K-track, which is knowledge rich and top-down. Both the meta-level cycle and the object level cycle contain K-track and d-track processes. At the object level, a statistical anomaly detector constitutes the first step of the D-track, a neural network identifies low-level causal attributes of detected anomalies, and a goal classifier, trained using methods from machine learning, formulates goals. The K-track is implemented as a case-based explanation process. The meta-level processes are under re-development.

The *MIDCA_1.1 model* (Paisner, Cox, Maynord, & Perlis, 2014) includes a complete planning-acting and perception-comprehension cycle at the cognitive level, and it incorporates a simple world simulator. The planning component integrates the SHOP2 (Nau, Au, Ilghami, Kuter, Murdock, Wu & Yaman, 2003) hierarchical network planner. This year we integrated the XPLAIN/Meta-AQUA (Burstein, Laddaga, McDonald, Cox, Benyo, Robertson, Hussain, Brinn & McDermott, 2008; Cox & Ram, 1999) multistrategy explanation system into the comprehension component. The simulator takes actions from the planner, calculates the changes to the world, and then passes the resulting state to the comprehension component. Comprehension examines the input for anomalies and generates new goals for the planner as warranted.

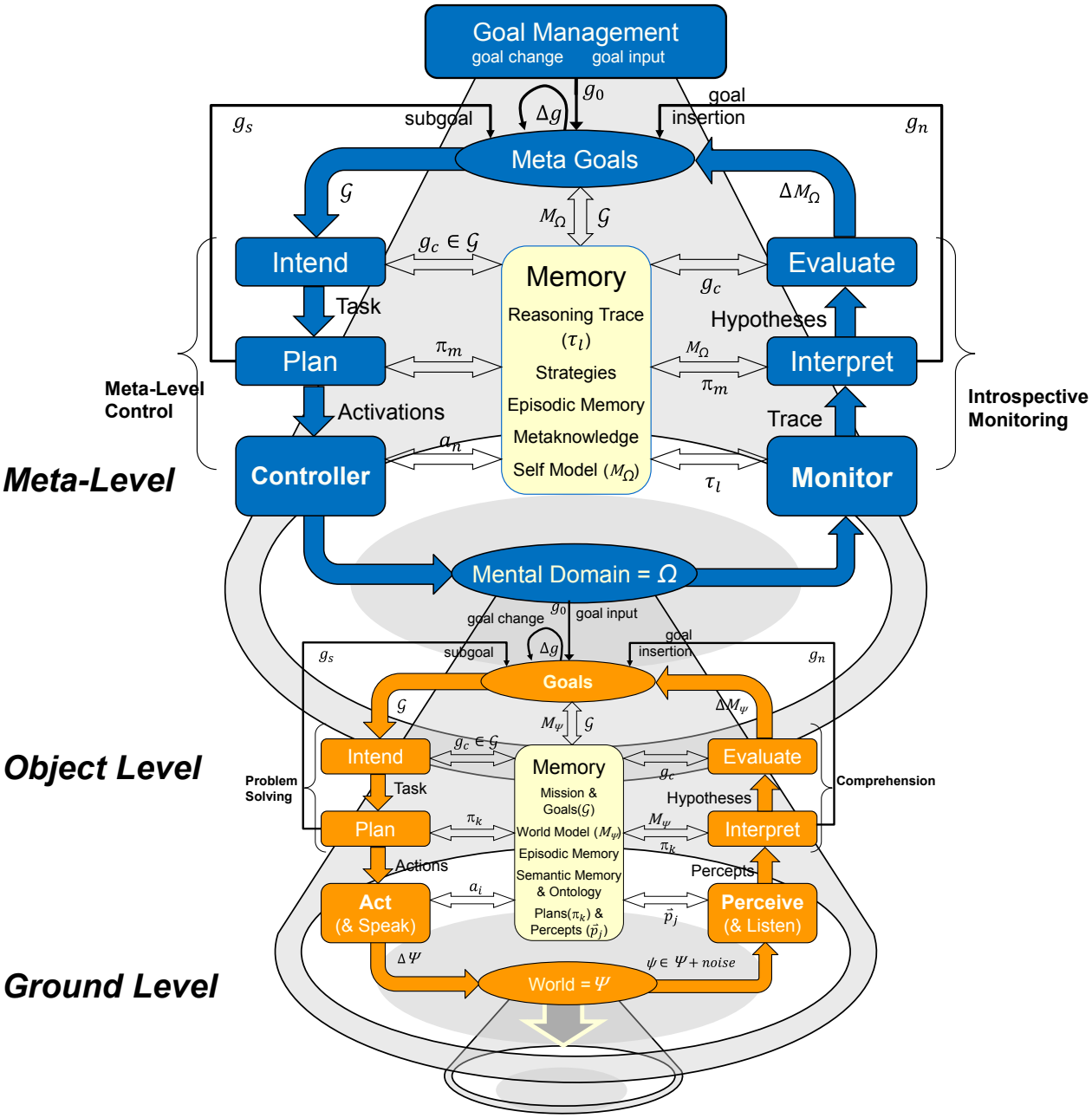

Figure 1. The current MIDCA architecture structure.

During the period of performance covered by this report, we have also made progress in identifying the details of the MIDCA architecture that most impact a computational approach to metacognition. In the comprehension processes at the meta-level, we cast introspective monitoring as an explanatory diagnosis task of the object level. That is, given an anomaly and a trace of the object-level reasoning, MIDCA must map the symptom of failure to the cause of the failure. We have analyzed numerous categories of failure symptoms and organize them in terms of expected outcomes and observed outcomes. See Table 1.

Table 1. Anomaly symptom taxonomy.

	Expectation (E) exists	Expectation (E) does *not* exist
Actual (A) event exists	Contradiction or Unexpected Success	Impasse or Surprise
Actual (A) event does *not* exist	False Expectation or Self-fulfilling Prophecy	Missed Opportunity

A contradiction exists when the *expected outcome (E)* for some cognitive computation is not equal to or inconsistent with the *actually observed outcome (A)*. The unexpected success is when the outcome is expected to be failure but success occurs nonetheless. Impasse occurs when an outcome cannot be generated; surprise is when in hindsight it is determined that an outcome should have been produced. In both cases, E does not exist. Similarly, in the bottom row, the symptom categories include conditions where A does not exist or is not known. For missed opportunities, neither E nor A are present to be compared and are available only in hindsight.

Like anomaly symptoms, we have organized the potential causes of failure into a similar taxonomy in Table 2. Four main categories exist in this taxonomy. Failure can result from object-level knowledge, goals, strategies or input from the environment. Each can be either missing or incorrect, and each has a selection component. For example forgetting is not a mistake due to incorrect knowledge, instead it is from incorrect or missing memory associations (i.e., indices). Similarly, correct information may exist in the input stream, but may not be attended to. This is a problem of input selection (i.e., attention) rather than of content.

Table 2. Anomaly cause (fault) taxonomy.

	Knowledge States		Goal States		Strategy		Environment	
	Domain Knowledge	Knowledge Selection	Goal Generation	Goal Selection	Processing Strategy	Strategy Selection	Input	Input Selection
Missing	Novel Situation	Missing Association	Missing Goal	Forgotten Goal	Missing Behavior	Missing Heuristic	Missing Input	Missing Context
Incorrect	Incorrect Domain Knowledge	Erroneous Association	Poor Goal	Poor Selection	Flawed Behavior	Flawed Heuristic	Noise	Incorrect Context
Correct	Correct Knowledge	Correct Association	Correct Goal	Correct Association	Correct Behavior	Correct Choice	Correct Input	Correct Context

We are currently developing the K-track meta-level introspective monitoring process to record the knowledge used, the goals pursued, the strategies employed, and the environmental input processed at the object level. That is, reasoning traces interpreted at the meta-level are represented by information

from the causal categories that distinguish between success and failure when reasoning. Meta-level control can then leverage the structure of these taxonomies.

Domain and Scenario (Task 2)
At the current time, we have the full plan→simulate→goal-generate→plan cycle implemented for a very simple variation of the blocks domain. This world includes blocks and pyramids. But instead of arbitrary block stacking, the purpose of plan activity is to build houses such that blocks represent the wall structure and pyramids represent house roofs. Within this domain, objects may catch on fire and so impede housing construction. New operators can extinguish fires and find arsonists. We have results that demonstrate better house construction performance using a top-down goal generation strategy compared to a statistical approach. Some of these details and a description of the individual interpretation methods used for comprehension are contained below.

One of the current scenarios implements a house building cycle that transitions through states as shown in Figure 2. If a part of a house (i.e., a block) catches fire, an operator exists that can extinguish the fire. When parts of houses are on fire, they do not count toward rewards for housing construction. Rewards are provided in terms of points, one for each block or pyramid not on fire in each completed house or tower. For example if the agent Figure 2 (c) was to next pickup D and place it on A, then the result would be a tower worth 3 points.

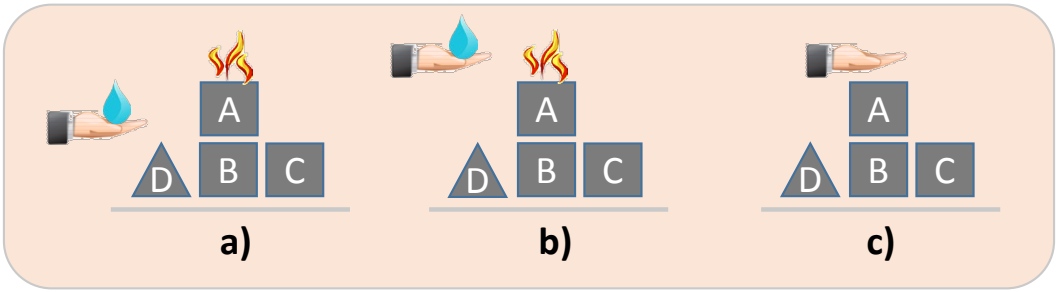

Figure 2. Unexpected events in the housing construction cycle.

Object level cycle
Most of our work this year at the object, or cognitive, level of MIDCA (i.e., the orange cycle in Figure 1), was with the knowledge-rich or K-track processes. We integrated the EXPLAIN system to perform a high-level version of a Note-Assess-Guide (NAG) procedure during the interpretation process. EXPLAIN has expectations in memory that allow it to predict features of the environment it perceives such as that houses are in stable conditions. When instead they are on fire, this causes an expectation failure (i.e., MIDCA Notes a contradiction anomaly). As a result, it explains what causes such an anomaly by retrieving prior general explanation patterns from memory and applying them to the current contradiction. The arsonist explanation shown in Table 3 is an example.

Finally, after the system notes the problem of the fire and assesses it as being potentially caused by an arsonist, it guides a response to the anomaly by generating a goal to remove the cause of the problem. The basis for such a goal is found in the antecedent of the above explanation. The fire is caused by heat, fuel and oxygen (the XP consequent) which is due to the arsonist using a lighter to provide the

heat (XP antecedent). So if we achieve a goal of removing the arsonist, then we remove the cause and hence solve the problem. This example is generalized into the approach of recognize a new problem, explain what causes the problem, and generate a goal to remove the cause (Cox, 2013). The resultant plan to achieve the goal is to apprehend the arsonist.

Table 3. Arsonist explanation pattern.

```
(define-frame ARSONIST-XP
  (actor (criminal-volitional-agent))
  (object (physical-object))
  (antecedent (ignition-xp
                (actor =actor)
                (object =object)
                (ante (light-object =l-o
                        (actor =actor)
                          (instrumental-object
                            (ignition-device))))
                (conseq =heat)))
  (consequent (forced-by-states
                 (object =object)
                 (heat =heat)
                 (conseq (burns =b
                           (object =object)))))
  (heat (temperature (domain =object)
                     (co-domain very-hot.0)))
  (role (actor (domain =ante))
              (co-domain =actor)))
  (explains =role)
  (pre-xp-nodes (=actor =consequent =object =role))
  (internal-nodes nil.0)
  (xp-asserted-nodes (=antecedent))
  (link1 (results
           (domain =antecedent))
           (co-domain =consequent)))
  (link2 (xp-instrumental-scene->actor
           (actor =actor)
           (action =l-o)
           (main-action =b)
           (role =role))))
```

Evaluation (Task 6): RESULTS

We tested our approach relative to a base line of performance against the D-track techniques we developed previously. The baseline condition called *Exogenous Goals* represents the behavior of the system with all goals provided by an exogenous user. Under the *Statistical Goal Generation* condition, we use the NAG procedures (see Maynord, Cox, Paisner, & Perlis, 2013; Paisner, Perlis, & Cox, 2013) that uses statistical machine learning techniques to learn reactive associations between states (e.g., block on fire) and goals (e.g., extinguished fire). The *GDA2 Goal Generation* condition uses D-track algorithms to generate new house construction goals and K-track explanation to generate goals for the fires (e.g., catch the arsonist). For each trial condition, MIDCA was run for 1000 time steps (equivalent to executing 1000 actions). At each step, the arsonist would have a probability p of starting a fire unless he had previously been apprehended. The value of p in the experiments described below was 0.4, allowing for enough fires to be significant without precluding progress in the tower construction project.

[2] Goal-Driven Autonomy. See Cox (2013).

Figure 3 shows that GDA approaches using only the D-Track as well as using both D-Track and K-Track perform significantly better than a baseline that does not use GDA. The reason is that the Exogenous Goals condition, like standard planning methods, does not generate new goals at run time and so here the prevalence of fires is unconstrained. Also, the combined D- and K-Track implementation outperforms the purely statistical variant by a large margin. The reason here is that the Statistical Goal Generation condition complete fewer towers because it spends effort repeatedly reacting to fires when they occur.

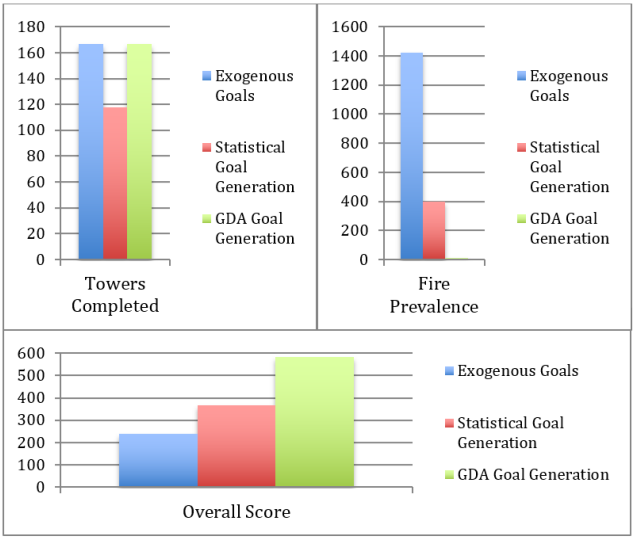

Figure 3. Results of testing using 3 conditions. Note that the value of GDA Goal Generation in the Fire Prevalence panel is 2, which is too small to show clearly in the graph.

Meta-level cycle

In the current reporting period we have begun to implement the meta-level monitoring and control cycle shown in blue within Figure 1. Concurrently we have started implementations of both D-track and K-track processes. For the D-track, we have developed a method of separating interesting conditions in the trace of object level processing. This uses an echo network that is a version of reservoir nets. The idea is to learn mappings from time-series representations of the cognitive cycle (as opposed to traces of behavior in the environment) to changes of goals (and their associated priorities) and thus cognitive performance. So far we have used this approach to distinguish fire scenarios from non-fire scenarios in the domain discussed above. As such this is an early exploration of the Note phase of the NAG procedure at the meta-level.

In an approach to the K-track at the meta-level, we have started to integrate MIDCA with the *Teleo-Reactive EXecutive (T-REX)* autonomous agent framework (Py, Rajan & McGann, 2010) that uses planning to satisfy system goals. In other research projects, T-REX has been deployed on underwater unmanned vehicles and used on the PR2 land robot. This framework balances the need for robotic agents to react quickly as well as the need to pursue long-term goals. This is achieved by a non-cyclic hierarchy of internal components, referred to as reactors, where each reactor has its own sense-plan-act loop. Reactors communicate with one another using constraints on state variables. State variables are visible to all reactors but are "owned" by at most one reactor. The reactor that owns a state variable is responsible for updating its value. Reactors that wish to change a state variable (i.e., the altitude of a

robot) owned by another reactor post a goal, which is a constraint on a state variable. All T-REX reactors have two important parameters that define a planning window in the future - *lookahead (θ)* and *latency (λ)* - and which are fixed and are set by the system designer before deployment. Lookahead specifies how far in the future the reactor should plan, and latency is how much time the reactor has to finish planning. However under different conditions, the parameter values may be inappropriate for the current tasks because no constant window size is best for all cases. By integrating the metacognitive layer of MIDCA with T-REX acting as the MIDCA object level, our approach is to allow MIDCA to monitor T-REX processing and adjust these parameter settings at run time (Dannenhauer, Cox, Gupta, Paisner, & Perlis, 2014).

In order to modify T-REX in real time, MIDCA will make use of an application programming interface (API) with the T-REX architecture that provides introspective monitoring and meta-level control functionality. This API exposes function calls to update latency and lookahead values, as well as read current lookahead and latency values of any reactor. The API also allows MIDCA to read and write goals to T-REX which are interpreted as if they were goals from a user (the same way the agent receives goals at the start of a mission). To explore this idea we have experimented with different parameter settings under various goal conditions.

Looking at Figure 4 we see that for both low window-parameter values ($\theta + \lambda < 9$) and high values ($\theta + \lambda > 17$) the agent is able to achieve 2 goals, but when ($9 <= \theta + \lambda <= 17$) we see that the agent is only able to achieve a single goal. While counter-intuitive, after further analysis it was discovered that when there were low values, the planner had to be very reactive and was able to achieve planning for both goals. When there were high values the planner had enough time to explore many different possible plans fully. However, during the middle part of the graph, the values were sufficient to allow the planner to explore multiple plans (i.e., more than just reactive behavior) but not large enough to allow the planner to find the end of the correct plan in time (i.e., planning was cut off). This is the result that we observed from running manual scripts. For our work in year 3, this is the kind of insight (i.e., low or high values achieve more goals than middle values) we expect MIDCA to be able to learn over time in a single mission and direct the behavior of the T-REX agent to either be in the low or high values, thus improving the number of goals achieved over time.

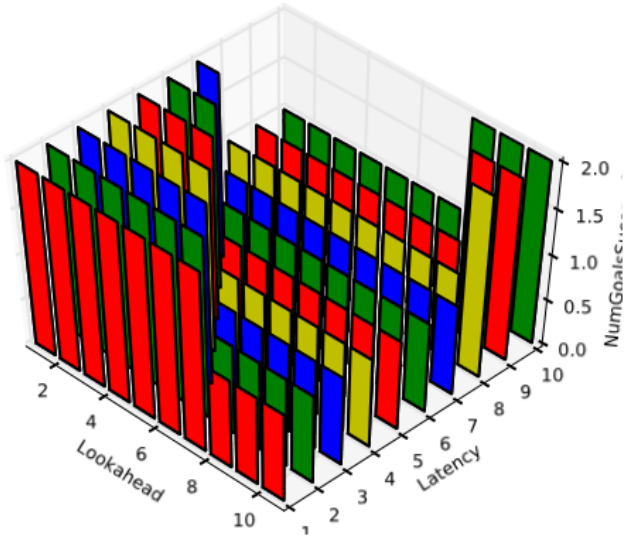

Figure 4. Number of goals achieved as a function of latency (λ) and lookahead (θ) values a simple domain.

WORK PLAN

Table 4 enumerates the 7 major tasks proposed for this project. We are roughly on schedule with respect to this outline, although we have changed our team structure and approach significantly to stay on schedule. See the section on Major Problems/Issues below. We also intend to push Tasks 3 and 4 approximately 6 months into the future. That is, we will begin them half way into year 2.

During the second year of the project, we intend to further develop the theoretical and computational foundations with the goal of developing a uniform approach to development and implementation. Our development work will focus on the metacognitive cycle and will strive to integrate K-track processes (e.g., planner monitoring and control) with D-track processes (e.g., echo net representations). Finally we plan to fully integrate the meta-level with the object level cycles which up to this point have been treated as independent systems, and we intend to demonstrate our results in simple tasks on physical research platforms by the end of year two.

Table 4. Original Project Schedule and Task Assignments

Tasks	*Year 1*	*Year 2*	*Year 3*	*Year 4*
1. Architecture Integration	Team	Team	Team	
2. Domain and Scenario	Team			
3. MIDCA Meta-Models		UMD	UMD	UMD
4. T-REX Robustness		MBARI	MBARI	MBARI
5. GDA Learning (option)	Lehigh	Lehigh	Lehigh	Lehigh
6. Evaluation	Team	Team	Team	Team

Major Problems/Issues

We had intended to pursue the complete research project using the T-REX system as the object-level reasoning system to drive the physical platform; MIDCA would interpret and control T-REX while T-REX would interpret sensor information and control the Dorado underwater platform. The plan was to use MBARI as a subcontractor, especially for Task 4. However this strategy has proved difficult due to contracting issues and the fact that the MBAR PI has left the organization and has still not settled on a home institution. Therefore we were never able to execute a subcontract during year 1. As a result we have decided to switch subcontract efforts to Lehigh University and will use the SHOP2 planner instead of T-REX for object-level control. As a result, Task 4 is renamed to Planner Robustness. The planner will probably be SHOP2, but we will ascertain the feasibility of this choice for the particular planner during the initial stages of Task 4. The target physical platform has also shifted from the MBARI Dorado to the Baxter research robot.

REFERENCES

Burstein, M. H., Laddaga, R., McDonald, D., Cox, M. T., Benyo, B., Robertson, P., Hussain, T., Brinn, M., & McDermott, D. (2008). POIROT - Integrated learning of web service procedures. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence* (pp. 1274-1279). Menlo Park, CA: AAAI Press.

Cox, M. T. (2013). Question-based problem recognition and goal-driven autonomy. In D. W. Aha, M. T. Cox, & H. Munoz-Avila (Eds.), *Goal Reasoning: Papers from the ACS workshop* (pp. 10-25). Tech. Rep. No. CS-TR-5029. College Park, MD: University of Maryland, Department of Computer Science.

Cox, M. T., & Ram, A. (1999). Introspective multistrategy learning: On the construction of learning strategies. *Artificial Intelligence, 112*, 1-55.

Dannenhauer, D., Cox, M. T., Gupta, S., Paisner, M. & Perlis, D. (2014). Toward meta-level control of autonomous agents. In *Proceedings of the 2014 Annual International Conference on Biologically Inspired Cognitive Architectures: Fifth annual meeting of the BICA Society* (pp. 121 -126). Elsevier/Procedia Computer Science.

Nau, D., Au, T., Ilghami, O., Kuter, U., Murdock, J., Wu, D., & Yaman, F. (2003). SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research 20*, 379–404.

Maynord, M., Cox, M. T., Paisner, M., & Perlis, D. (2013). Data-driven goal generation for integrated cognitive systems. In C. Lebiere & P. S. Rosenbloom (Eds.), *Integrated Cognition: Papers from the 2013 Fall Symposium* (pp. 47-54). Technical Report FS-13-03. Menlo Park, CA: AAAI Press.

Paisner, M., Cox, M. T., Maynord, M., Perlis, D. (2014). Goal-driven autonomy for cognitive systems. In *Proceedings of the 36th Annual Conference of the Cognitive Science Society* (pp. 2085-2090). Austin, TX: Cognitive Science Society.

Paisner, M., Perlis, D., & Cox, M. T. (2013). Symbolic anomaly detection and assessment using growing neural gas. In *Proceedings of the 25th IEEE International Conference on Tools with Artificial Intelligence* (pp. 175-181). Los Alamitos, CA: IEEE Computer Society.

Perlis, D., & Cox, M. T. (2014). MetaMAc, or what do I do now? A strategic perspective on autonomy beyond anomalies and goals. In *Proceedings of the Tenth International Conference on Autonomic and Autonomous Systems* (pp. 1-4). Red Hook, NY: Curran Associates.

Py, F., Rajan, K., & McGann, C. (2010). A systematic agent framework for situated autonomous systems. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 2*. International Foundation for Autonomous Agents and Multiagent Systems.

PUBLICATIONS

Workshops and Conferences

Cox, M. T. (2013). Question-based problem recognition and goal-driven autonomy. In D. W. Aha, M. T. Cox, & H. Munoz-Avila (Eds.), *Goal Reasoning: Papers from the ACS workshop* (pp. 10-25). Tech. Rep. No. CS-TR-5029. College Park, MD: University of Maryland, Department of Computer Science.

Dannenhauer, D., Cox, M. T., Gupta, S., Paisner, M. & Perlis, D. (2014). Toward meta-level control of autonomous agents. In *Proceedings of the 2014 Annual International Conference on Biologically Inspired Cognitive Architectures: Fifth annual meeting of the BICA Society* (pp. 121 -126). Elsevier/Procedia Computer Science.

Paisner, M., Cox, M. T., Maynord, M., Perlis, D. (2014). Goal-driven autonomy for cognitive systems. In *Proceedings of the 36th Annual Conference of the Cognitive Science Society* (pp. 2085-2090). Austin, TX: Cognitive Science Society.

Perlis, D., & Cox, M. T. (2014). MetaMAc, or what do I do now? A strategic perspective on autonomy beyond anomalies and goals. In *Proceedings of the Tenth International Conference on Autonomic and Autonomous Systems* (pp. 1-4). Red Hook, NY: Curran Associates.

Presentations

Cox, M. T. (2014, July). *Goal-driven autonomy and robust architectures for long-duration missions*. Invited presentation, Goal Reasoning Research Summit II. University of Maryland Institute for Advanced Computer Studies. College Park, MD.

Cox, M. T. (2014, April). *Recent work on goal reasoning at the University of Maryland*. Invited presentation, Goal Reasoning Research Summit. Naval Research Laboratory, Washington, DC.

AWARD PARTICIPANTS

Faculty

- **Michael T. Cox** (Wright State Research Institute; University of Maryland, Institute for Advanced Computer Studies)
- **Hector Munoz-Avila** (Lehigh University; visiting professor University of Maryland, College Park, summer 2014)

Students

- **Matthew Paisner** (Ph.D. student, University of MD, Department of Computer Science)
- **Ben Bengfort** (PhD student, University of MD, Department of Computer Science)
- **Dustin Dannenhauer** (PhD student, Lehigh University, Computer Science and Engineering Department; summer intern, University of Maryland)