Long, L.N., Hanford, S.D., and Janrathitikarn. O., "Cognitive Robotics using Vision and Mapping Systems with Soar," Invited Paper, SPIE Defense & Security Conference, Orlando, FL, April, 2010.

Cognitive robotics using vision and mapping systems with Soar

Lyle N. Long*, Scott D. Hanford, and Oranuj Janrathitikarn
The Pennsylvania State University, University Park, PA USA 16802

ABSTRACT

The Cognitive Robotic System (CRS) has been developed to use the Soar cognitive architecture for the control of unmanned vehicles and has been tested on two heterogeneous ground robots: a six-legged robot (hexapod) and a wheeled robot. The CRS has been used to demonstrate the applicability of Soar for unmanned vehicles by using a Soar agent to control a robot to navigate to a target location in the presence of a cul-de-sac obstacle. Current work on the CRS has focused on the development of computer vision, additional sensors, and map generating systems that are capable of generating high level information from the environment that will be useful for reasoning in Soar. The scalability of Soar allows us to add more sensors and behaviors quite easily.

Keywords: Cognitive, robotics, unmanned, vehicles, mapping

1. INTRODUCTION

The popularity of unmanned vehicles and the ability of computer algorithms to describe and generate intelligent behavior have increased significantly in the past few years. There has also been increased interest in using computational psychology tools to control unmanned vehicles. In particular, cognitive architectures have been used for control of unmanned vehicles [1-6]. There are many problems to consider when using cognitive architectures with unmanned vehicles, including: finding relevant symbolic or high level information from the environment, integrating robotic motor control and perception activities with the cognitive architecture, using traditional robot algorithms in conjunction with the architecture, spatial reasoning, collaboration with humans, sensor processing, coupling to sub-symbolic systems, and learning.

Soar was chosen after investigating several different software systems [7]. The use of the Soar cognitive architecture [8] to control unmanned vehicles can have several benefits. Jones et al. [9] have used the Soar architecture to autonomously fly U.S. military fixed-wing aircraft during missions in a *simulated* environment for the TacAir-Soar project using Soar agents with 5200 production rules, 450 total operators, and 130 abstract operators. This helped demonstrate the scalability of Soar to thousands of rules, due to the use of the Rete algorithm [10]. The TacAir-Soar project also demonstrated that Soar is capable of performing high-level robot activities, such as reasoning, using large agents in a simulated environment with real-time constraints. Since cognitive architectures are able to model general decision making, a single Soar agent can be used for multiple missions and can be capable of using multiple approaches to the same problem. The ability to use agents that have large amounts of knowledge and can model general decision making allows for the potential to develop Soar agents capable of robust, intelligent behavior in complex environments.

There are several key abilities for control of unmanned vehicles, however, that do not coincide with the strengths of Soar, such as performing a large amount of numerical calculations, low-level control of motors, or optimization problems. The Soar Markup Language (SML) [11] allows simplified interaction of a Soar agent with external environments and other software systems that complement Soar's strengths (e.g., neural networks, robot control algorithms, state estimation techniques, and object recognition methods).

Beyond cognitive robotics, there is a need for more generalized intelligence. Systems that can learn on their own and become more and more capable with time are needed. In the future, if done correctly, these systems could even become conscious [12]. The Cognitive Robotic System (CRS) has been developed to integrate sensors, motors, and software with the Soar cognitive architecture for the control of unmanned vehicles [7].

* lnl@psu.edu; Phone: 814-865-1172; FAX: 814-865-7092; Web: http://www.personal.psu.edu/lnl

2. THE COGNITIVE ROBOTIC SYSTEM (CRS)

We have developed a general-purpose Cognitive Robotic System (CRS), which can be use on a wide variety of robotic platforms. The primary components of CRS are:

- Software: Soar architecture, Soar agent rules, Java software, and sensor processing software
- Hardware: Computer (laptop, Via motherboard, etc.), Brainstem board, motor drivers, motors, Parallax servo controllers, servos, and batteries
- Sensors: Cameras, sonar, compass, wheel encoders, infrared, touch, and GPS
- Algorithms: SLAM, SIFT, echo location, neural networks, stereo vision, image processing, ...

Unlike most robots, the software is fairly compact. The main behaviors of the robot are defined through the rules, which are executed in Soar. Java is used to interface the input and output devices with Soar. Java is an excellent language for robotics since it allows object oriented programming (OOP), the use of threads, and its built-in graphics libraries. There is also some software that has been written to execute some features of the Brainstem as well. The hardware is a fairly standard computer with a few controller boards. As we add sensors, we have also been adding well-known algorithms to process the data from these sensors, including neural networks [13].

The CRS has been implemented on two unmanned ground vehicles: a hexapod and a wheeled robot [14]. The hexapod chassis is the HexCrawler, a robotic kit from Parallax Inc. (www.parallax.com). The wheeled robot, the SuperDroid, has an aluminum base which was purchased from SuperDroid Robots (www.superdroidrobots.com).

The CRS has been developed to share as many common components between the implementations on both robots as possible. The same Soar agent can control both vehicles, using the same sensors and microcontroller to communicate with the sensors. The main difference in CRS implementations between the robots is the differences between controlling the motors and actuators. Most previous robots have had special software written specifically for them, whereas we simply have to write rules for each platform. We have air-, ground-, and sea-based unmanned vehicles, as shown in Figure 1, and the CRS could be used on all of them. Our unmanned air vehicle work is described in [15, 16].

The CRS has components that we believe are important for intelligent unmanned systems: sub-symbolic processing, symbolic processing, scalable processing, learning, and scalable I/O.

Figure 1. Examples of Penn State's unmanned air-, ground-, and underwater vehicles.

3. PREVIOUS EXPERIMENTS USING THE CRS

Previously, the CRS applications have focused on using the symbolic processing capabilities and decision cycle of Soar to make high-level decisions about how to control an unmanned vehicle. Two Soar agents were developed for the mission of navigating to a specified goal location while avoiding obstacles. The ability of the Soar agents to successfully reach a goal location was tested in experiments with different types of obstacles [14].

3.1 Initial Soar agent

The original Soar agent [17] attempted to use information from a small set of sensors and relatively simple Soar rules to complete its mission. Two sonar, or ultrasonic distance, sensors were used to detect obstacles in front of the robot. The

agent used a compass to estimate its current heading and a GPS sensor to estimate the agent's current location. The Soar agent uses the sensor information and creates a representation of its state that is useful for reasoning (e.g., convert a distance to an obstacle to higher-level concept, such as if there is something close enough that agent needs to avoid).

The knowledge, represented by rules, for this Soar agent is organized into two subgoals: avoid obstacles and go to target. The avoid obstacle subgoal is used when the agent is close to a sensed obstacle. The rules the Soar agent uses to avoid obstacles are simple: the agent tries to turn away from an obstacle until the obstacle is no longer sensed. The go to target subgoal is used when no there are no obstacles close enough to need to be avoided. The rules in this subgoal compare the agent's estimated heading and location to the heading and distance to the goal and try to minimize the difference between the agent's current heading and the heading to the goal by moving the robot.

The agent is mainly reactive; however, rules in both subgoals try to avoid undoing recent actions, which was helpful for preventing the agent from getting stuck near some kinds of obstacles. This agent was successfully able to navigate to a goal location in experiments with small obstacles, but was unsuccessful in experiments with larger, more complex obstacles such as cul-de-sacs.

3.2 Updated Soar Agent

To successfully complete the experiment with a cul-de-sac obstacle, a new Soar agent was developed to use additional information about its environment and more sophisticated rules [14]. There were three main extensions to the original agent:

- Infrared distance sensors to detect obstacles on the right and left side of the robot were added. These sensors were used along with the sonar sensors to detect obstacles in the agent's environment.
- Additional Soar rules were written to give the agent an alternate way to avoid obstacles by following the side of an obstacle until it was facing its goal location.
- The updated agent also remembers more details about the actions it has previously taken so it can reason about if its actions have been successful in accomplishing the agent's current goal. For example, if the agent is not able to avoid an obstacle using one approach, it is able to realize the current approach is not being successful and try another approach.

The updated Soar agent was able to successfully navigate to a goal location in the presence of a cul-de-sac obstacle. The successful completion of the mission showed that the addition of knowledge to the agent in the form of Soar rules and the addition more sensors to the CRS to provide more information about the environment to the agent can lead to a more capable Soar agent.

4. CURRENT WORK WITH CRS

Our recent efforts have been to increase the number of rules and sensors CRS uses. This will lead to more sophisticated and capable behaviors, as well as better situational awareness.

4.1 Wheeled Vehicle

The research described in previous sections demonstrated the usefulness of Soar for intelligent robots on a practical mission, but the Soar agent was able to accomplish this mission with minimal high level (symbolic) perceptual information about the environment. Incorporating improved perceptual systems that could be used to find meaningful symbols from the environment that the Soar agent can reason about could substantially increase the capabilities of the CRS and its Soar agents.

A popular problem in mobile robots is conducting a search within an unknown (no known map) indoor environment for specific objects. A well known example of this problem is search and rescue (for example, the RoboCupRescue competition, www.robocup.org). Important capabilities for successful robots for the problem of search within an unknown building environment include the ability to create a map of the environment, ideally with symbolic information that could easily be shared with humans such as first responders [18], and the ability to identify objects of interest and report where they are located within the environment. This section will describe current work on the SuperDroid and

CRS in the areas of map creation and object identification for application to the mission of search within an unknown indoor environment.

Mapping

Building a map is a basic way in which a mobile robot can interact with and learn about its environment. A robot builds a map by using its perceptual capabilities, such as vision, audio, touch, and/or range finding, to create a model of its environment. Map building for robots can be a challenging problem because of its dependence on a robot's imperfect knowledge of its environment. There are two types of representations commonly used for robot mapping: metric and topological.

A metric map is a quantitative description of an environment and can encode locations of obstacles or other objects of interest. The occupancy grid [19], which represents the environment by a grid of cells, is a popular metric map, particularly for environments with arbitrary, dense obstacles. In an occupancy grid map, each cell has a probability that an obstacle is located in the environment at that location. For large scale environments, there are challenges such as the accumulation of localization errors and large memory requirements associated with the use of occupancy grids.

Topological maps describe how objects in the environment (landmarks, for example) are connected to each other. Topological representations can be useful in large scale environments because of their compact representation. Additionally, topological representations could be very useful for representing the agent's environment and using for reasoning in Soar. Some challenges in building topological maps are identifying landmarks and permitting "distinction between places" [20].

Additionally, hybrid metric and topological representations have also been developed to combine the strengths of metric and topological maps. Thrun [19] described a system that generated a topological representation from metric grid-based maps. Tomatis [20] developed a method to use a global topological map of an office environment to connect local metric maps of rooms.

The initial use of maps in the CRS will be the creation of an occupancy grid of an indoor environment. Figure 2 shows an example occupancy grid of a small section of a hallway generated using three sonar sensors. The walls of the hallway are shown in black, the hallway and part of an office to the right of the hallway are shown in white, and the areas of the figure shown in gray are areas of the environment that have not been explored. The hardware and mapping algorithm that will be used are described by Hanford et al. [21], and will be summarized here. Incorporating occupancy grids into the CRS will provide a framework to fuse information from different sensors as well as integrate sensor data over time. In addition to being useful for tasks such as path planning and obstacle avoidance, it is possible to extract symbolic information from occupancy grids [22]. A similar approach that identified intersections between hallways or between a hallway and doors from an occupancy grid would be very useful as the intersections could be used by a Soar agent as landmarks for a topological map of an environment.

Figure 2. Occupancy grid from using three sonar sensors.

The hardware used to generate an occupancy grid is shown in Figure 3. Three sonar sensors and two infrared sensors are used to detect obstacles in front and to the sides of the SuperDroid. Wheel encoders installed on the two front wheels are used to estimate the position of the SuperDroid. Two web cameras have been installed on the SuperDroid and will be used as a stereo pair to measure distances to edge pixels in the future. Two BrainStem microcontrollers are used to obtain information from the sonar sensors, the infrared sensors, and wheel encoders and control the SuperDroid's motors using a motor driver board. The onboard laptop receives sensor information from the BrainStem network and web cameras.

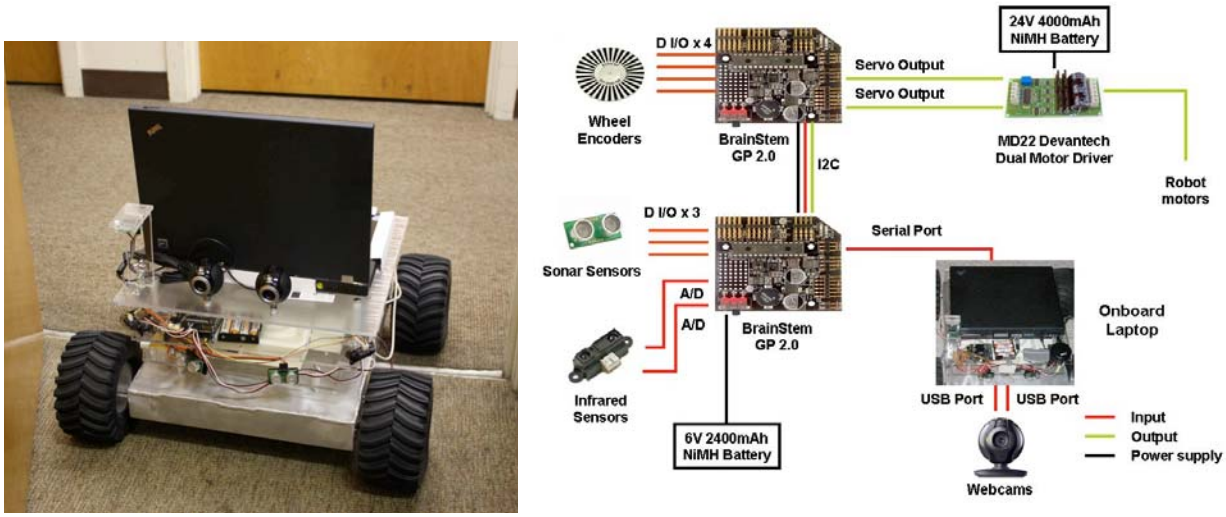

Figure 3. The SuperDroid and a schematic of the hardware used on the SuperDroid for map generation.

SIFT

Image matching is important for many computer vision problems, including object and scene recognition. For the mission of search within an unknown environment, the ability to recognize an object is a crucial capability. The Scale Invariant Feature Transform (SIFT) attempts to solve the image matching problem by finding local image features ``that can be used to perform reliable matching between different views of an object or scene" [23]. SIFT has been widely used in the computer vision and mobile robot communities.

SIFT features have also been used on mobile robots to locate objects of interest in an indoor environment. Meger et al. [24] described a system that maps its environment using an occupancy grid and appends the map with locations of objects found using SIFT features. Objects such as a basketball, recycling bin, box of detergent, and a photograph are identified by comparing SIFT features from training images of these objects to images captured using an onboard camera.

An executable file created by the developer of SIFT is available for download [25]. The executable locates the SIFT features in an image and generates the SIFT feature vector for each feature. We have incorporated using this executable into the CRS and have developed code to compare SIFT features between two images to detect SIFT features that are present in both images.

There are two ways we are interested in using SIFT in the CRS. First, similar to the work by Meger et al. [24], SIFT could be used to identify multiple objects in the robot's environment. These objects could be simulated victims (e.g., a doll) for a simulated search and rescue mission, other objects of interest (e.g., a weapon or a bomb), or items whose presence would identify what type of room they were located in (e.g., a computer is likely to be in an office). Second, SIFT features could be used to distinguish between different intersections. As an intersection is identified, a database of SIFT features from images taken at that intersection could be created. Having a database of SIFT features for each intersection would allow the robot to not only find landmarks that are important in the environment, but to be able to

uniquely distinguish each of the landmarks. Each time the robot reaches an intersection the robot could determine if it had already visited the intersection before or if it was encountering the intersection for the first time.

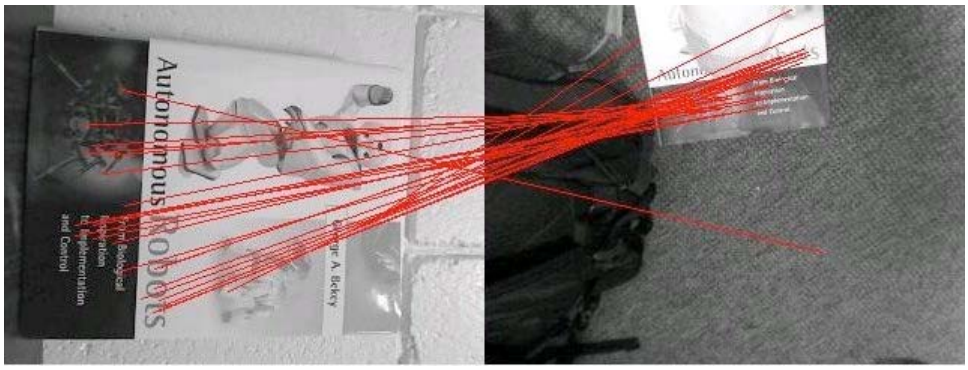

Figure 4. Matched SIFT features between two different images of a textbook.

4.2 Hexapod Vehicle

This section describes the implementation of CRS on a six-legged vehicle, shown in Figure 5. The main objective of using the legged robot is to develop a platform that is able to traverse all kinds of terrains especially where the wheeled robots can not pass easily. Therefore, after the Soar agent described in section 3.1 was successfully implemented on the hexapod, force sensors were added to the front legs to provide the sense of the terrain to the robot [26]. A new set of Soar agents incorporating two sonar sensors in the front of the hexapod and two force sensors attached to the two front legs were experimented with on unlevel terrain. The results showed that, in order to traverse the terrain more effectively, the robot should have sensors available on all feet to detect the terrain. This section will describe the current work on the hexapod on the touch sensors, webcams, and microphones.

Figure 5 also shows the most up-to-date hardware elements on the hexapod. Two sonar sensors, six touch sensors, and one electronic compass send inputs to the computer via two BrainStem modules. A GPS and two web cameras are also connected to the computer directly on the USB ports. Each camera has built-in microphone. The onboard laptop retrieves all sensor information from the BrainStem modules by using Java classes and processes the information either in Java or Soar rules. Then, the output to control the robot movement is sent to the servos through the Parallax Servo Controller board on another USB port.

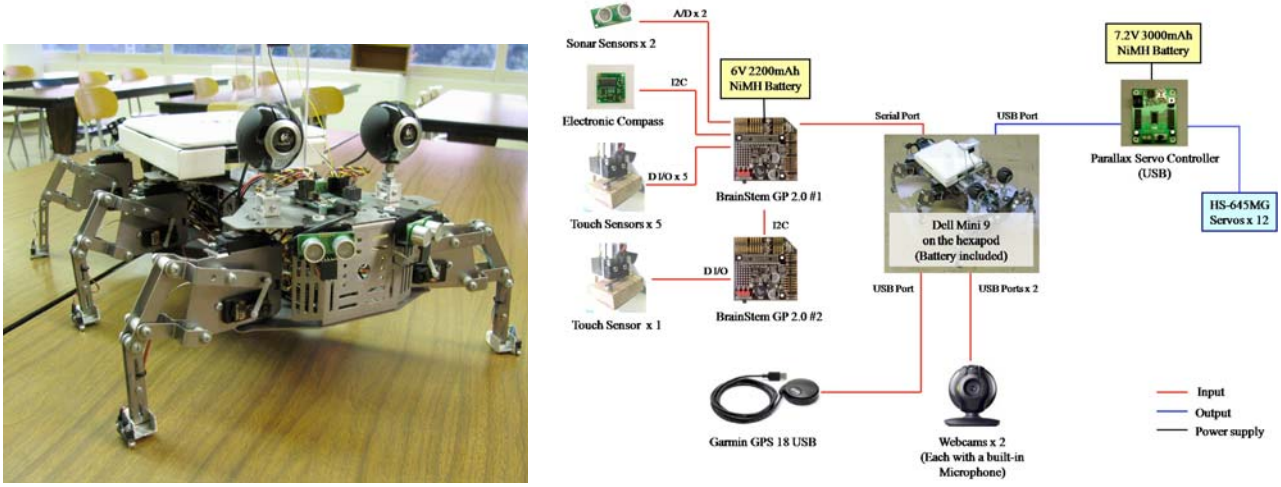

Figure 5. Hexapod and hardware schematic.

Since the previous force sensors attached to the two front legs of the hexapod had limited sensitivity, they were replaced by a new mechanism which is the hinge support shown in Figure 6, which will press a push button switch once each foot touches the ground. The new mechanism was attached to all six legs of the hexapod and the robot was tested on the outdoor unlevel terrain. The touch sensors (shown in Figure 6) demonstrated better performance in the different terrains. However, the performance is also limited by the leg clearance of the robot, which caused some problems. With two degree of freedom provided by the horizontal and vertical servos, each leg can move approximately one inch above the ground. Consequently, the robot legs should be modified to have more clearance above the ground or the robot should be used mainly on the surface which has roughness less than one inch. The existing platform could be used to detect holes in the terrain, as well as work better on unlevel terrain.

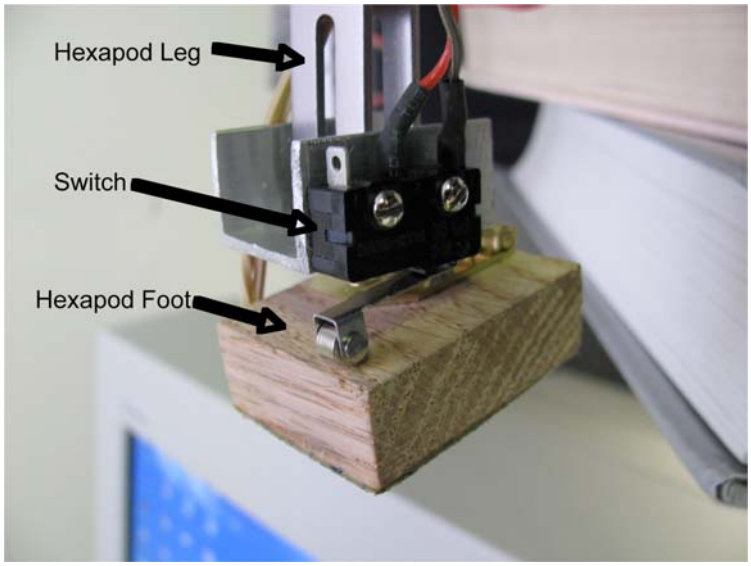

Figure 6. Touch sensor on hexapod foot.

Sonar sensors and touch sensors can sense different features of the terrain. For example, a sonar sensor can detect whether there is an obstacle in front of it and a touch sensor can detect whether the foot is touching the ground. The more sensors we have the better. and then the data can be fused together. A vision system combined with sonar and touch sensors would be very useful.

Two webcams were attached to the front of the hexapod. Quicktime for Java was used to obtain two images from the two webcams and image processing algorithms were developed. A combination of Canny edge detection and disparity mapping is used [27]. This allows finding distances to objects and the detection of walls and hallways indoors.

The two webcams also have built-in microphones which can be accessed by Quicktime for Java. The Java code to access the microphones was developed and a sound localization algorithm [28] was implemented in Java. This method uses the cross-correlation of the signals from two different microphones. The time shift that corresponds to the highest correlation is converted to a distance and an angle from the robot. This has not yet been tested in CRS, but it has been implemented on a Pioneer Robot at Aberdeen Proving Grounds [29]. This will provide very interesting behaviors such as following a sound source or avoiding a sound source.

5. SUMMARY

Previously, the CRS has used a small set of sensors that provided a minimal amount of information about the environment to its Soar agent, limiting the level of interaction between the Soar agent and its environment. This paper has described the current extensions to Cognitive Robotic System. These extensions focus on the addition of sensors and algorithms that can provide more information, especially symbolic information, about the environment to a Soar agent. This additional information will permit Soar agents in the CRS to have more interesting interactions with their environment and be applied to more challenging and useful problems.

REFERENCES

[1] W. G. Kennedy, M. Bugajska, M. Marge *et al.*, "Spatial Representation and Reasoning for Human-Robot Collaboration," Proceedings of the Twenty-Second Conference on Artificial Intelligence, 1554-1559 (2007).

[2] T. D. Kelley, " Developing a Psychological Inspired Cognitive Architecture for Robotic Control: The Symbolic and Sub-symbolic Robotic Intelligence Control System (SS-RICS)," International Journal of Advanced Robotic Systems, 3(3), (2006).

[3] J. G. Trafton, A. C. Schultz, N. L. Cassimatis *et al.*, [Communicating and Collaborating with Robotic Agents, in Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation] Cambridge University Press, Cambridge(2006).

[4] T. D. Kelley, [Using a Cognitive Architecture to Solve Simultaneous Localization and Mapping (SLAM) Problems] Aberdeen Proving Ground, Aberdeen, MD(2006).

[5] T. D. Kelley, E. Avery, L. N. Long *et al.*, "A Hybrid Symbolic and Sub-Symbolic Intelligent System for Mobile Robots," InfoTech@Aerospace Conference, (2009).

[6] P. Benjamin, D. Lyons, and Lonsdale, "Designing a Robot Cognitive Architecture with Concurrency and Active Perception," AAAI Fall Symposium on Cognitive Science and Robotics, (2004).

[7] S. D. Hanford, and L. N. Long, "Evaluating Cognitive Architectures for Unmanned Autonomous Vehicles," 22nd Conf. on AI, Association for the Advancement of Artificial Intelligence (AAAI), (2007).

[8] J. E. Laird, A. Newell, and P. S. Rosenbloom, "Soar: An Architecture for General Intelligence," Artificial Intelligence, 33(3), 1-64 (1987).

[9] R. M. Jones, J. E. Laird, R. E. Nielsen *et al.*, "Automated Intelligent Pilots for Combat Flight Simulation," AI Magazine, 27-41 (1999).

[10] C. Forgy, [On the efficient implementation of production systems] Ph.D. Thesis, Carnegie-Mellon, Pittsburgh(1979).

[11] , [SML Quick Start Guide] ThreePenny Software L. L. C., (2005).

[12] L. N. Long, Kelley, Troy D., "A Review of Consciousness and the Possibility of Conscious Robots," Journal of Aerospace Computing, Information, and Communication, 7(2), (2010).

[13] A. Gupta, and L. N. Long, [Hebbian Learning with Winner Take All for Spiking Neural Networks] IEEE, Atlanta, GA(2009).

[14] S. D. Hanford, O. Janrathitikarn, and L. N. Long, "Control of Mobile Robots Using the Soar Cognitive Architecture," Journal of Aerospace Computing, Information, and Communication, 6(2), 69-91 (2009).

[15] G. L. Sinsley, L. N. Long, B. R. Geiger *et al.*, "Fusion of Unmanned Aerial Vehicle Range and Vision Sensors Using Fuzzy Logic and Particles," AIAA InfoTech@Aerospace Conference, AIAA Paper No. 2009-1890, Seattle, WA, (2009).

[16] B. R. Geiger, J. F. Horn, G. L. Sinsley *et al.*, "Flight Testing a Real Time Implementation of a UAV Path Planner Using Direct Collocation," Journal of Guidance Control and Dynamics, 31(6), (2008).

[17] S. D. Hanford, Long, Lyle N., Janrathitikarn, Oranuj, "Control of a Six-Legged Mobile Robot Using the Soar Cognitive Architecture," AIAA Paper No. 2008-0878, AIAA Aerospace Science Meeting, (2008).

[18] B. Balaguer, S. Balakirsky, S. Carpin *et al.*, "Evaluating maps produced by urban search and rescue robots: lessons learned from RoboCup," Autonomous Robots, (2009).

[19] H. Moravec, and A. Elfes, "High-resolution maps from wide-angle sonar," Proceedings of the 1985 IEEE International Conference on Robotics and Automation, 116-121 (1985).

[20] N. Tomatis, M. E. Jefferies, and W. K. Yeap, [Hybrid, Metric-Topological Representation for Localization and Mapping, in Robot and Cognitive Approaches to Spatial Mapping] Springer, Berlin/Heidelberg(2008).

[21] S. D. Hanford, G. L. Sinsley, and L. N. Long, "Integration of Maps into the Cognitive Robotic System," AIAA InfoTech@Aerospace, (2010).

[22] O. M. Mozos, and W. Burgard, "Supervised Learning of Topological Maps using Semantic Information Extracted from Range Data," Proc. of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, (2006).

[23] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60(2), 91-110 (2004).

[24] D. Meger, P. E. Forssén, K. Lai *et al.*, "Curious George: An attentive semantic robot," Robotics and Autonomous Systems Journal, 56(6), 503-511 (2008).

[25] D. G. Lowe, [http://people.cs.ubc.ca/~lowe/home.html], (Accessed 2010).

[26] O. Janrathitikarn, Long, Lyle N., “Gait Control of a Six-Legged Robot on Unlevel Terrain using a Cognitive Architecture,” IEEE Aerospace Conference, (2008).
[27] S. E. Palmer, [Vision Science: Photons to Phenomenology] MIT Press, Cambridge(1999).
[28] J. C. Murray, Erwin, H., Wermter, S., “Robotic Sound-Source Localization and Tracking Using Interaural Time Difference and Cross-Correlation,” Proceedings of NeuroBotics Workshop, (2004).
[29] L. N. Long, [Neural Network Integration With The Symbolic And Sub-Symbolic Robotic Intelligence Control System (SS-RICS) (Contract No. TCN 07-305)], Aberdeen, MD(2009).