The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Knowledge-based Control of a Humanoid Robot

Dongkyu Choi, Yeonsik Kang, Heonyoung Lim, and Bum-Jae You

Abstract— **Today, there are increased interest and various efforts in using cognitive architectures to control robotic platforms. Recent advances to essential capabilities in robots contributed to this trend, which tries to meet the increased demand for high-level control mechanisms. The tradition of cognitive architectures aims for general intelligence, and they have some great potential for use in robots that are now increasingly more capable of complex tasks. In this paper, we introduce one such architecture,** ICARUS**, used in a robotic environment to provide knowledge-based control for a humanoid robot, MAHRU. We show some experimental observations in the Blocks World domain.**

I. INTRODUCTION

Recently, researchers in the field of robotics show increased interests in the use of cognitive architectures [1]. There have been various efforts to use them to control robotic platforms, which now possess essential capabilities required for interesting high-level tasks. These architectures, designed as computational models of human cognition, commits to a specific set of knowledge representation and the processes that work over it. They provide mechanisms for inferring the state of the world, making decisions based on the computed state, and applying actions that are needed. In most cases, they also support learning in various ways, including, but not limited to, learning from problem solving, learning from observations, and learning from failures [2][3][4][5].

One such architecture, ICARUS, has a long history of continuous development and revisions, and it is specifically designed for physical domains [3]. Previously, it was used successfully in various domains, which include a simulated Blocks World, a FreeCell solitaire game [3], a simulation of urban driving [6], and a first-person shooter game, Urban Combat [7]. These domains provided challenging environments for the ICARUS architecture, with different levels of complexity. They served as important testbeds on which most of the components and features of the architecture were evaluated and experimented. However, there has always been the desire and need for a domain that involves some type of physical machinery.

In this paper, we report an application of the ICARUS architecture in a simulated robotic environment. The simulation is a realistic computational representation of a humanoid robot, MAHRU [8]. As a preliminary step before the actual integration of the architecture and the humanoid, we have devised an interface between ICARUS and the simulation. We selected a familiar, but often challenging domain, the Blocks World, for our first set of experiments. As a simplified version of the book shelving task, we created a scenario, in which the robot should sort blocks of different colors and stack them in towers with the same-colored blocks.

In the following sections, we first review the fundamentals of the ICARUS architecture, and provide more details on the humanoid robot, MAHRU, and its simulation. Then we present some experimental observations in the Blocks World. We conclude after a discussion on related and future work.

D. Choi is with Cognitive Systems Laboratory, Center for the Study of Language and Information, Stanford University, Stanford, CA 94305, USA. dongkyuc@stanford.edu (corresponding author)

Y. Kang and B.-J. You are with Center for Cognitive Robotics Research, Korea Institute of Science and Technology, Seoul, Korea. yeonsikkang@gmail.com, ybj@kist.re.kr

H. Lim is with Robust Design Engineering Laboratory, Seoul National University, Seoul, Korea. hylim@rodel.snu.ac.kr

II. REVIEW OF ICARUS ARCHITECTURE

The ICARUS architecture is one of the systems that grew out of the cognitive architecture movement. It shares some basic features with other architectures, like Soar [2] and ACT-R [9]. The architecture commits to a specific representation of knowledge, and distinguishes long-term and short-term memories. It has separate memories for its conceptual knowledge that describes the state of the environment, and its skill knowledge that specifies procedures to achieve specific goals. The system provides an organized way to specify domain knowledge that is readily available for execution, and supports various methods for learning to acquire the knowledge automatically.

In this section, we describe ICARUS' commitment for a certain representation of knowledge and for memories that store the knowledge. Then we show processes that work over these memories, including the inference of current situation, the evaluation of skills, and the execution of actions in the world.

A. Representation and Memories

Based on psychological evidences, ICARUS distinguishes conceptual and skill knowledge, and the architecture has separate memories for them. It further distinguishes long-term and short-term knowledge, and a long-term conceptual memory stores definitions of its conceptual knowledge, or *concepts*, and a short-term conceptual memory, or *belief memory*, houses instances of these concepts that are true in the current state. Similarly, a long-term skill memory stores definitions of ICARUS' procedural knowledge, or *skills*, and a short-term skill memory records instantiated skills. These skill instances are, however, closely tied to the goals they achieve, and the short-term skill memory also serves as a *goal memory*, and it stores substantially more information than the skill instances themselves.

TABLE I

SOME SAMPLE CONCEPTS FOR OUR ROBOT, MAHRU IN THE SIMULATED BLOCKS WORLD DOMAIN.

```
((holding ?gripper ?block)
 :percepts  ((gripper ?gripper status ?block)
             (block ?block)))

((hand-empty ?gripper)
 :percepts  ((gripper ?gripper status ?status))
 :tests     ((eq ?status 'empty)))

((clear ?block)
 :percepts  ((block ?block))
 :relations ((not (on ?other ?block))))
```

Table I shows some sample concepts used in the simulated Blocks World domain. Concept definitions start with a head, which includes the name of the concept, and its arguments that are often variables marked with a preceeding question mark as in `?gripper` and `?block`. The definitions also have several optional fields like `:percepts` that includes perceptual matching conditions, `:tests` that specifies conditions among matched variables, and `:relations` that has references to other concepts. The first two concepts shown are *primitive*, in the sense that they consist solely of variable matchings against objects in the world like a `gripper` and a `block` and conditions among these variables. The last concept, however, is a *non-primitive* one, which refers to another concept. In this case, the reference is negated, and the concept *(clear ?block)* matches when the predicate *(on ?other ?block)* is not true in the current state.

In Table II, we provide some sample skills for MAHRU in this domain. As with concepts, ICARUS' skills consist of a head and several optional fields. While a `:percepts` field serves the same purpose as in concepts, a `:start` field specifies preconditions of the skill, an `:actions` field has its implied actions in the world, and a `:subgoals` field includes a subgoal decomposition for the particular skill. The first example shown is a *primitive* skill, which directly refers to basic actions the ICARUS agent can apply in the environment. However, the rest of skills are *non-primitive*, and they include references to subgoals that, in turn, lead to other skills. In this way, both concepts and skills in the ICARUS architecture are hierarchically organized, and they provide rich vocabulary to describe complex states and procedures required by domains in the real world.

The ICARUS architecture also has a goal memory. This memory stores the agent's top-level goals and their subgoals with the skill instances it intends to execute, or has executed for them. The information is stored in the form of *goal stacks*, as shown in Table III. Here, the top-level goal is *(color-sorted)*, which means that all the blocks are sorted by their colors in towers. The system retrieved a skill instance with id 19 that is known to achieve the goal. Also shown is the execution stack on the particular cycle, which we will discuss later in this section.

TABLE II

SOME SAMPLE SKILLS FOR OUR ROBOT, MAHRU IN THE SIMULATED BLOCKS WORLD DOMAIN.

```
((stacked ?block ?to) ID: 3
 :percepts ((gripper ?gripper)
            (block ?block)
            (block ?to x ?x y ?y z ?z height ?h))
 :start    ((stackable ?gripper ?block ?to))
 :actions  ((*release ?block ?x ?y (+ ?z ?h) pb)))

((on ?block1 ?block2) ID: 16
 :percepts ((block ?block1)
            (block ?block2))
 :subgoals ((stackable ?gripper ?block1 ?block2)
            (stacked ?block1 ?block2)))

((one-color-sorted ?color) ID: 17
 :percepts ((block ?block1 color ?color)
            (block ?block2))
 :start    ((same-color ?block1 ?block2)
            (clear ?block2))
 :subgoals ((on ?block2 ?block1)
            (one-color-sorted ?color)))

((one-color-sorted ?color) ID: 18
 :percepts ((block ?block1 color ?color)
            (block ?block2)
            (block ?block3))
 :start    ((same-color ?block1 ?block2)
            (same-color ?block1 ?block3)
            (on ?block2 ?block1))
 :subgoals ((clear ?block3)
            (on ?block3 ?block2)
            (one-color-sorted ?color)))

((color-sorted) ID: 19
 :start    ((not-color-sorted ?color))
 :subgoals ((one-color-sorted ?color)
            (color-sorted)))
```

B. Inference, Evaluation, and Execution

The ICARUS architecture runs in distinct cycles. On each cycle, it performs a series of cognitive processes, including the perception of its surroundings, the inference of concept instances based on the perceived data, the evaluation of skills under the current state, and the execution of actions implied by the chosen skill instance. Fig. 1 shows ICARUS' operation on each cycle. At the beginning of each cycle, ICARUS receives information on the objects it can perceive from the environment. This information is deposited in the perceptual buffer. ICARUS then matches its concepts stored in the long-term conceptual memory against perceived objects, infers all the concept instances that are true in the current state, and stores them in its belief memory.

Based on the current beliefs, the architecture evaluates its hierarchical skills and finds a skill path that is most relevant to the situation. Shown in the `:EXECUTION STACK` field in Table III is an example of an executable skill path. In this case, the goal is *(color-sorted)*, and the system retrieves a non-primitive skill with the same name that has an id 19. This skill refers to another skill, *(one-color-sorted green)*, and it, in turn, refers to *(on block4 block2)*. This continues until

TABLE III

AN EXAMPLE OF A GOAL STACK STORED IN ICARUS' GOAL MEMORY. (A SIMPLIFIED VERSION IS SHOWN FOR EASY REFERENCE.)

```
[(GOAL
 :CHAINTYPE  SKILL
 :GOALTYPE   PRIMARY
 :OBJECTIVE  (COLOR-SORTED)
 :INTENTION  ((COLOR-SORTED) ID: 19 BINDINGS: NIL
 :EXECUTION STACK
  #1 (((COLOR-SORTED) ID: 19
        BINDINGS: ((?COLOR . GREEN))
      ((ONE-COLOR-SORTED GREEN) ID: 17
        BINDINGS: ((?BLOCK1 . BLOCK2)
                   (?BLOCK2 . BLOCK4)
                   (?COLOR . GREEN))
      ((ON BLOCK4 BLOCK2) ID: 16
        BINDINGS: ((?BLOCK1 . BLOCK4)
                   (?BLOCK2 . BLOCK2))
      ((STACKABLE ?GRIPPER BLOCK4 BLOCK2) ID: 12
        BINDINGS: ((?GRIPPER2 . G-R)
                   (?GRIPPER1 . G-L)
                   (?BLOCK . BLOCK4)
                   (?TO . BLOCK2))
      ((REACHABLE-BY-BOTH-ARMS BLOCK4) ID: 14
        BINDINGS: ((?FROM . T) (?BLOCK . BLOCK4)
                   (?GRIPPER . G-L))
      ((PUTDOWN-IN-THE-CENTER BLOCK4 ?TO) ID: 4
        BINDINGS: ((?HEIGHT . 0.89) (?Z . 0.89)
                   (?Y . 0.0) (?X . 0.0)
                   (?TO . T) (?GRIPPER . G-L)
                   (?BLOCK . BLOCK4))))]
```

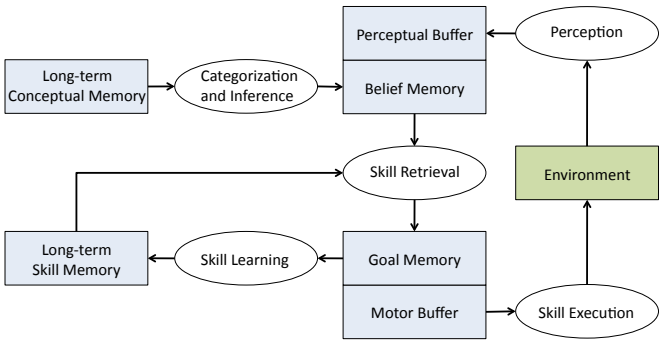

Fig. 1. A schematic of execution cycles in ICARUS architecture. Rectangles denote the environment (far right) and ICARUS' buffers and memories, while oval shapes denote processes that work over them.

it reaches a primitive skill, *(putdown-in-the-center block4 t)*. This path represents the decision ICARUS made at the moment, with the most abstract skill on top and the most specific one at the bottom.

Once an executable skill path is found, the architecture executes actions implied by the skill path to change its environment. This, in turn, changes ICARUS' perception on the next cycle. Then the system continues to the next step of the procedure. It is notable that this cyclic operation gives reactivity to the ICARUS architecture while staying goal-oriented. Therefore, ICARUS agents can readily adapt to unexpected situations or outcomes. For example, if the robot fails to pick up a block or loses the grip of one, ICARUS will perceive the unexpected outcome in its subsequent cycle and correct the situation through retrials. We will discuss more on ICARUS' ability to recover from failures later in this paper. In the next section, we introduce our robot platform, MAHRU in more detail.

III. A HUMANOID ROBOT: MAHRU

We integrated the ICARUS architecture with a simulation of a humanoid robot, MAHRU, developed at Korea Institute of Science and Technology. The simulated environment is a realistic representation of the actual robot, and many components of the current interface will transfer directly to the eventual integration of ICARUS and MAHRU. In this section, we provide more details on the humanoid platform and its simulation.

A. Hardware Platform

As shown in Fig. 2, MAHRU is a humanoid robot platform that stands at 150cm tall and weighs about 57kg. It can reach the maximum speed of 1.2km/h. The robot has 12-DOF in its legs and 20-DOF in the two arms including 4-DOF in its grippers. The robot has an IMU (Inertial Measurement Unit) sensor mounted on its body, and it is equipped with four force/torque sensors on its arms and legs. For real-time control, the robot runs on a PC with real-time Linux (RTAI/Xenomai) operating system. It uses communication lines that follows the IEEE 1394 protocol to deliver control signals between the main PC and its DSP-based sub-controllers [10][11].

The robot is specifically designed as a network-based humanoid, which can send various sensor data to external servers over the wireless network to perform high-level recognition, inference, and decision-making. This type of robots costs less than standalone systems, and they have increased operation time per each battery charge. Furthermore, it provides more flexibility for features and services a robot can have.

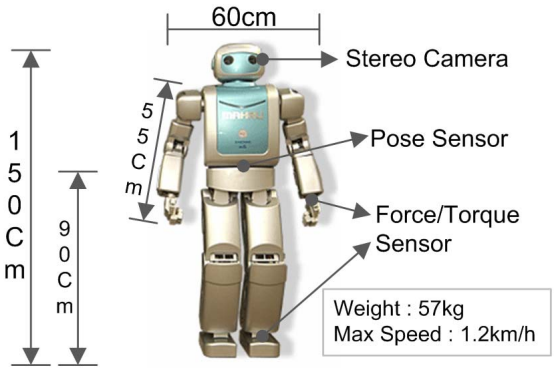

Fig. 2. A humanoid platform, MAHRU

B. Simulated Platform

We used a 3-D simulation of MAHRU in SimStudio [12], a simulation software with a built-in dynamics engine that solves complex rigid body contact problems including collision detection and resolution. The environment provides models for various sensors and actuators. It allows real-time analysis of the simulated robotic system. Fig. 3 shows a screenshot of the simulation environment.

Using the virtual sensor and actuator simulations in the software, we encoded in the simulator the same motion control algorithm as in the actual robot. Since this algorithm controls all the low-level robot motions, the simulator can interface with higher-level controllers – in this case, the ICARUS architecture – through various text-based commands [13]. For example, we can trigger a grasping motion by issuing a command like "grab object with id 1", and have the robot walk forward by sending a command, "walk forward 1 meter".

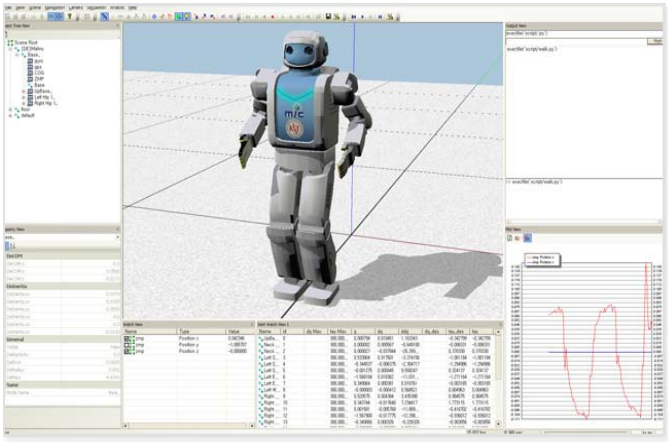

Fig. 3. 3-D dynamic simulation environment

IV. EXPERIMENTAL OBSERVATIONS

We integrated the ICARUS architecture and the MAHRU humanoid robot with a TCP connection between them, unlike some previous work on ICARUS that used shared memory structures [7]. We considered the fact that our robot is designed to be network-based, and high-level control systems for this robot should be on a server connected through a wireless interface. We gave the integrated system some basic tasks in the simulation of the Blocks World. By doing so, we wanted to verify that ICARUS can indeed make high-level decisions for the humanoid in this domain, and that MAHRU can execute these control commands smoothly and achieve its goal.

In the simulated Blocks World, we created a set of tasks with an initial condition where seven blocks are on table. There are two red blocks, three green blocks, and two blue blocks. Table IV shows the tasks we gave our robot,

TABLE IV
BLOCKS WORLD TASKS GIVEN TO MAHRU

goals	tasks	cycles
(on block0 block1)	build a tower of two red blocks	5
(on block0 block6)	build a tower of two blocks that involve a hand change	7
(color-sorted)	build towers of same-colored blocks	13

MAHRU, with the corresponding goal encoded for each task and the number of cycles it took until completion when the probability of action failures in the environment were set to zero. The first task is building a tower of two red blocks, `block0` and `block1`, and therefore achieving the goal, *(on block0 block1)*. The task requires a simple movement of a block, grasping `block0` and moving and releasing it on `block1`. It took five cycles for ICARUS-controlled MAHRU to complete this task.

The second task is more complicated, although the goal is to build a similar tower by achieving *(on block0 block6)*. In this case, one block is on the right-hand side of the robot, and the other is on the left-hand side of the robot. Due to the limited reachability of each hand, MAHRU cannot move the first block onto the second block with one hand. It should change hands in between to achieve the goal. Therefore, MAHRU first grasps `block0` with its right hand and moves it to the center of the table where both of its hands can reach. Then it uses its left hand to grasp the block and moves it onto `block6` on its left side. This task took two more cycles for MAHRU, finishing the execution in seven cycles.

The two tasks so far require two different procedures for goals in the form, *(on ?block1 ?block2)*. The first procedure simply moves a block on top of another, but the second one involves a hand change during the move. In ICARUS, we can have multiple disjunctive skills for a given goal, representing procedures for different preconditions. We will see another example of disjunctive skills below, during our discussion on the third task.

The last task is also the most complicated, and it is worth discussing it in more detail. Figure 4 shows the sequence of actions ICARUS commanded MAHRU to achieve the color-sorting goal. Here, the robot should build three different towers, one with two red blocks, another with three green blocks, and yet another with two blue blocks. For this task, ICARUS decomposes the top-level goal, *(color-sorted)* into three different subgoals, which are distinct instances of *(one-color-sorted ?color)* for the three colors, red, green, and blue.

The system then chooses a color at random, and it works on red blocks first in this case (see 1–3 in Fig. 4). Here, it uses a particular definition of the skill, the one with id 17 shown earlier in Table II. The skill is more general than the other one with id 18, and its preconditions match when there are at least two blocks of a color and one of them is clear. Once this skill finishes, the system selects the next color,

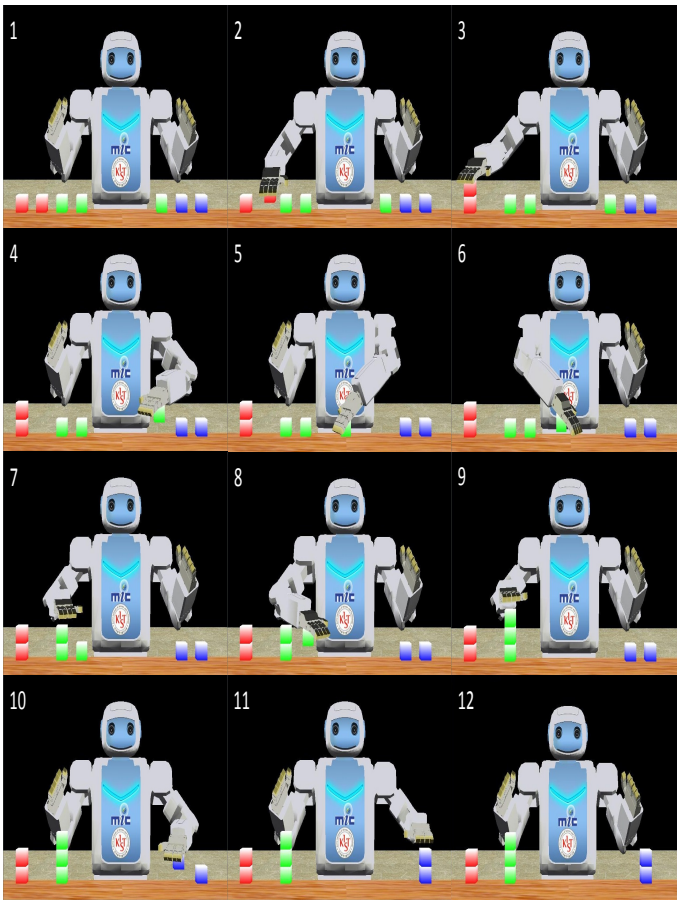

Fig. 4. Screenshots of MAHRU performing the color sorting of blocks.

green (see 4–9). In this case, it first uses the same skill as above, and build a green tower with two blocks with another green block still on table. Then it uses the skill with id 18, for which more specific preconditions match – there are three blocks that are green, and two of them are stacked. After the system successfully completes the procedure, it builds a blue tower in a similar way as it did with red blocks (see 10–12).

As noted earlier, we acquired the results above with the probability of action failures set to zero. When we increased the probability, ICARUS sometimes faced unexpected outcomes such as failing to grab a block or dropping a block elsewhere than its intended location. In such cases, the architecture reacted to the unexpected situation, and persistently made attempts to fix the failure, or proceeded with a different skill that is known to work in the new situation. Of course, these failures caused an inevitable increase in ICARUS' cognitive cycles to complete its tasks. For instance, at 10% probability of action failures, it took 17.1 cycles on average for ICARUS to complete the color sorting task, which is an increase of approximately 30%.

We verified that the ICARUS architecture is able to make high-level decisions in the robotic Blocks World domain even under the possibility of action failures, and that the humanoid, MAHRU can execute the commands from ICARUS correctly. The physical restrictions the robot platform has add complexity in various dimensions, but the architecture successfully manages to produce valid procedure to achieve the given goals.

V. RELATED AND FUTURE WORK

Recently, we have seen increased interests in cognitive architectures among researchers in robotics and its related fields. Cognitive psychologists and researchers in artificial intelligence have developed these architectures as steps toward general intelligence over a relatively long period of time, and they usually provide stable architectural support for robotic applications in many different directions. For instance, Benjamin et al. [15] developed a new architecture specifically for robots based on the Soar architecture. They used the system with a mobile robot designed to interact with people under different circumstances.

Kieras and Meyer [14] used the EPIC architecture for human-computer interaction. Taking advantage of the architecture's multi-task performance, they showed that modeling human-computer interaction in EPIC results in a precise computational models. The EPIC architecture successfully simulates human behavior including visual and auditory processes.

Meanwhile, other researchers developed their own architectures for cognitive control of robots. Kawamura et al. [16] developed a humanoid robot, ISAC and extended the system to a cognitive architecture. The architecture was multiagent-based, with attention, memory, and internal rehearsal happening in parallel.

Kasderidis [17] introduced the GNOSYS architecture, as an execution system for robotic agents. It provides many features found in traditional cognitive architectures, including execution for multiple goals, dynamic prioritization, and opportunistic behavior.

Although our current work shows some promising results, our research on using the ICARUS architecture in the robotic platform is still at its preliminary stage. We will continue our work by replacing the simulation with the actual robot, MAHRU, and testing the integrated system in the Blocks World. Then we will attempt to use more capabilities supported by the ICARUS architecture in the robotic environment, especially, learning from problem solving and learning from observations. We would also like to use more of the features our robot has. For example, the robot's capability to move around with its biped will open up new possibilities in its applications and it certainly will challenge the ICARUS architecture's ability to control more complicated behaviors. We plan to find another domain that involves all of these additional capabilities that are readily available in our integrated system. We hope to report more on this topic in the near future.

VI. CONCLUSIONS

In this paper, we integrated a cognitive architecture, ICARUS and a humanoid robot, MAHRU. We tested the integrated system in a robotic Blocks World domain, and it successfully performed tasks at different levels of complexity, which included a task to building towers of same-colored blocks from an initial condition with blocks of three different colors. Although the integration still used a simulation of the actual robot, the simulated environment provided a realistic representation of the platform with adjustable probability of action failures, to which ICARUS reacted properly. Therefore, we are confident that the overall system will work smoothly with the robot, and we hope that we can report on the integration in the near future.

VII. ACKNOWLEDGMENTS

This research was funded in part by an individual research contract for Project No. 2E20870 from Korea Institute of Science and Technology. Discussions with Pat Langley, Chunki Park, and Negin Nejati contributed to many ideas presented here.

REFERENCES

[1] A. Newell, *Unified Theories of Cognition*, Harvard University Press, Cambridge, MA; 1990.

[2] J.E. Laird, P.S. Rosenbloom, and A. Newell, Chunking in Soar: The anatomy of a general learning mechanism, *Machine Learning*, vol. 1, 1986, pp. 11–46.

[3] P. Langley, D. Choi, and S. Rogers, Acquisition of hierarchical reactive skills in a unified cognitive architecture, *Cognitive Systems Research*, vol. 10, 2009, pp. 316–332.

[4] R.M. Jones, and P. Langley, A Constrained Architecture for Learning and Problem Solving, *Computational Intelligence*, vol. 21, 2005, pp. 480–502.

[5] N. Nejati, P. Langley, and T. Könik, "Learning Hierarchical Task Networks by Observation", *in Proceedings of the Twenty-Third International Conference on Machine Learning*, Pittsburgh, PA, 2006.

[6] D. Choi, M. Morgan, C. Park, and P. Langley, "A testbed for evaluation of architectures for physical agents", *in Proceedings of the AAAI-2007 Workshop on Evaluating Architectures for Intelligence*, Vancouver, BC, 2007.

[7] D. Choi, T. Könik, N. Nejati, C. Park, and P. Langley, "Structural transfer of cognitive skills", *in Proceedings of the Eighth International Conference on Cognitive Modeling*, Ann Arbor, MI, 2007.

[8] B.-J. You, Y. Choi, M. Jeong, D. Kim, Y. Oh, C. Kim, J. Cho, M. Park, and S. Oh, "Network-based humanoids 'Mahru' and 'Ahra'", *in Proceedings of the International Conference on Ubiquitous Robots and Ambient Intelligence*, 2005, pp. 376–379.

[9] J.R. Anderson, *Rules of the Mind*, Lawrence Erlbaum, Hillsdale, NJ; 1993.

[10] S. Kim, C. Kim, and J.H. Park, "Human-like arm motion generation for humanoid robots using motion capture database", *in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[11] Y.-H. Chang, Y. Oh, D. Kim, and S. Hong, "Balance control in whole body coordination framework for biped humanoid robot MAHRU-R". *in Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, 2008.

[12] SimLab co., ltd., "SimStudio2 User Manual (English, v2.4.1)", *http://www.simstudio.co.kr/english/*.

[13] H. Lim, Y. Kang, J. Lee, J. Kim, and B.-J. You, "Multiple humanoid cooperative control system for heterogeneous humanoid team", *in Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, 2008.

[14] D.E. Kieras and D.E. Meyer, An overview of the EPIC architecture for cognition and performance with application to human-computer interaction, *Human Computer Interaction*, vol. 12, 1997, pp. 391–438.

[15] D.P. Benjamin, D. Lonsdale, and D. Lyons, "A cognitive robotics approach to comprefending human language and behaviors", *in Proceedings of the Conference on Human-Robot Interaction*, 2007.

[16] K. Kawamura, S.M. Gordon, P. Ratanaswasd, E. Erdemir, and J.F. Hall, Implementation of cognitive control for a humanoid robot, *International Journal of Humanoid Robotics*, vol. 5, 2008, pp. 547–586.

[17] S. Kasderidis, "An executive system for cognitive agents", *in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.