ISAC: Foundations in Human–Humanoid Interaction

Kazuhiko Kawamura, R. Alan Peters II, D. Mitchell Wilkes, W. Anthony Alford, Tamara E. Rogers, Vanderbilt University

THE AUTHORS DESCRIBE THEIR HUMANOID SYSTEM, INTELLIGENT SOFT-ARM CONTROL, AND THEIR APPROACH TO HUMAN–HUMANOID INTERACTION. THEY PRESENT A SOFTWARE ARCHITECTURE AND TWO HIGH-LEVEL AGENTS WITHIN THEIR HHI FRAMEWORK.

AT THE INTELLIGENT ROBOTICS Laboratory of Vanderbilt University's Center for Intelligent Systems, over the past several years we have been developing a humanoid system called Intelligent Soft-Arm Control. We originally developed ISAC as a robotic aid system for the physically disabled (see Figure 1).[1] It has since evolved into a test bed for *human–humanoid interaction* research.[2,3] While working on ISAC, we developed a flexible architecture for parallel, distributed robot control; a framework for robust HHI; and a control system that will let ISAC learn. To demonstrate our approach's potential, we have implemented a handshake demonstration.

ISAC

We designed ISAC using the philosophy of embodied intelligence[4]—that is, to achieve humanlike interaction, the robot must have a humanoid shape. ISAC has two six-degrees-of-freedom arms actuated by McKibben artificial muscles (see Figure 2). These muscles are pneumatic actuators whose lengths shorten as their internal air pressure increases.[1] They are attached to the arm in antagonistic pairs. These actuators not only approximate the action of human muscles more closely than do electric motors but also have a significantly larger strength-to-weight ratio. Moreover, they are naturally compliant and are safe for use in close contact with people. The muscles are fed compressed air through servo control valves. We designed and built the PC-resident arm controller cards. The arms have optical position encoders at each joint.

ISAC has cost-effective anthropomorphic end effectors, built in-house, that we call PneuHand I and PneuHand II. Small pistons pneumatically actuate PneuHand I; PneuHand II employs electrical and pneumatic actuation in which the forefinger and thumb have a motor and a piston in parallel. The motors enable fine control in grasping, and the piston provides strength in the grasp. The arm–hand systems have six-axis force-torque sensors at each wrist, proximity sensors on the palms, and rudimentary touch sensors on each finger. ISAC also employs color, stereo, active vision with pan, tilt, and verge; sonic localization; and speech I/O.

ISAC's sensory-motor suite is not nearly as rich as those of a vertebrate animal (for example, ISAC has no sense of taste or smell and has highly limited haptic sensing). However, the diversity of its sensory modalities coupled with its 18 degrees of freedom have let us equip it with several fundamental behaviors. Also, the relative simplicity of ISAC's sensory-motor capabilities facilitates experimentation on behavior learning because there are fewer signals to associate and fewer variables to control.

The Intelligent Machine Architecture

The IMA is our software architecture for designing robot control architectures. It has sufficient generality to permit the simultaneous deployment of several robot architectures.[5] That is, we can design a behavior using whatever control strategy most simplifies its implementation. For example, we use variants of Ronald Arkin's motor schema[6] for controlling reaching and for collision avoid-

1094-7167/00/$10.00 © 2000 IEEE

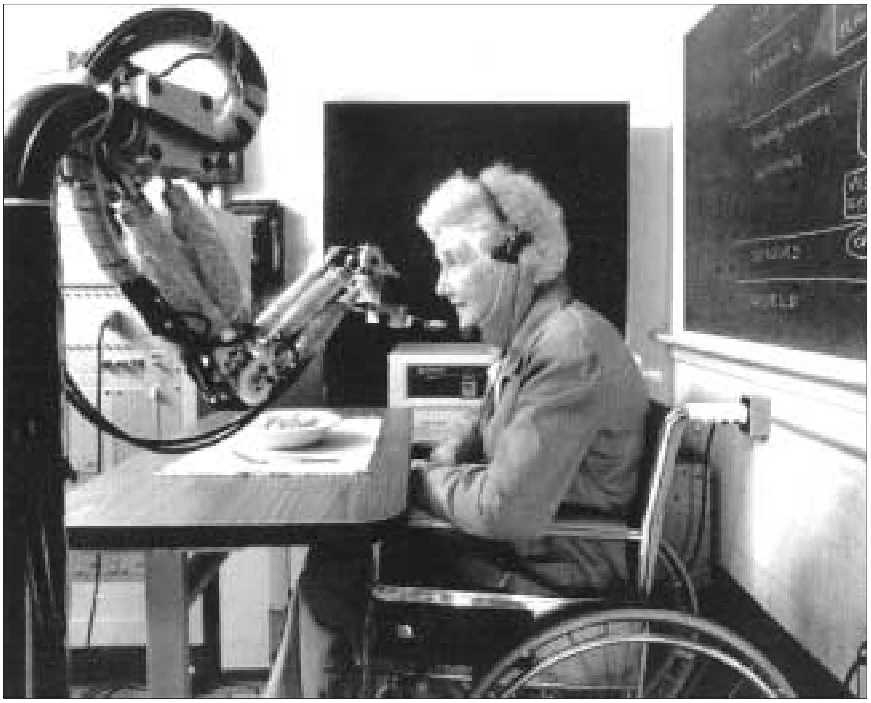
Figure 1. An early version of the Intelligent Soft-Arm Control feeding a physically disabled person.

ance in dual-arm behaviors. Some of our visual tracking routines employ standard predictive correlation methods. Other visual behaviors, such as the saccade, use variants of the subsumption architecture.[4]

The IMA decomposes the system into a set of *atomic agents*, which are independent, autonomous entities with one or more threads of execution. By *atomic*, we mean a fundamental building block; we distinguish IMA atomic agents from the more common definition of *agent* as an autonomous, intelligent entity—whether machine, software, or biological. Although intelligent behavior emerges from the interaction of atomic agents in the IMA, atomic agents are not, in general, intelligent. They are similar to the agents Marvin Minsky[7] described or to the simple agents that some authors call *actors*.

The system's decomposition into atomic agents depends on the problem's context and on the choice of robot architecture for the particular behavior. Typically, we cast the system in terms of the robot's hardware components, the sensory-motor behaviors it can perform, the objects with which it will interact, and the tasks it can perform. We then assign an atomic agent to each instance of these. For example, ISAC has two arms, two hands, and a head. Each of these has its own atomic agent. Atomic agents can and do exist at different abstraction levels.

In the IMA's context, an atomic agent is one element of a domain-level system description that tightly encapsulates all aspects of that element, much like objects in object-oriented systems. The atomic agent serves as a superstructure for everything the software system knows or does relating to an element of the robot, a task, or the environment.

The IMA can be used to implement almost any logical control architecture. Atomic agents are loosely coupled, which facilitates parallel processing. Although the IMA can run on a single PC, it enables concurrent agent execution on separate machines in a network. For larger systems, the IMA exploits both distributed and symmetric multiprocessing computer systems more effectively than do monolithic architectures. Each atomic agent acts locally, based on its internal state, and provides a set of services to other agents through various relationships. The loosely

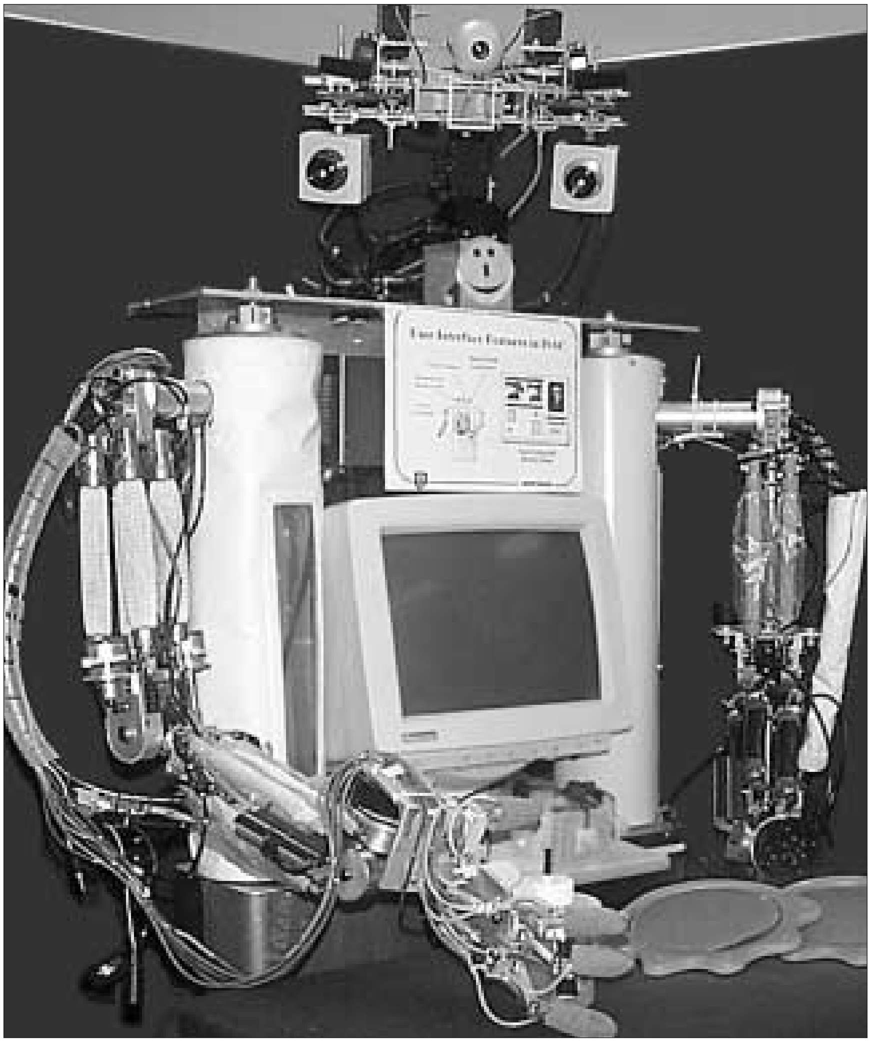
Figure 2. The ISAC (Intelligent Soft-Arm Control) humanoid system.

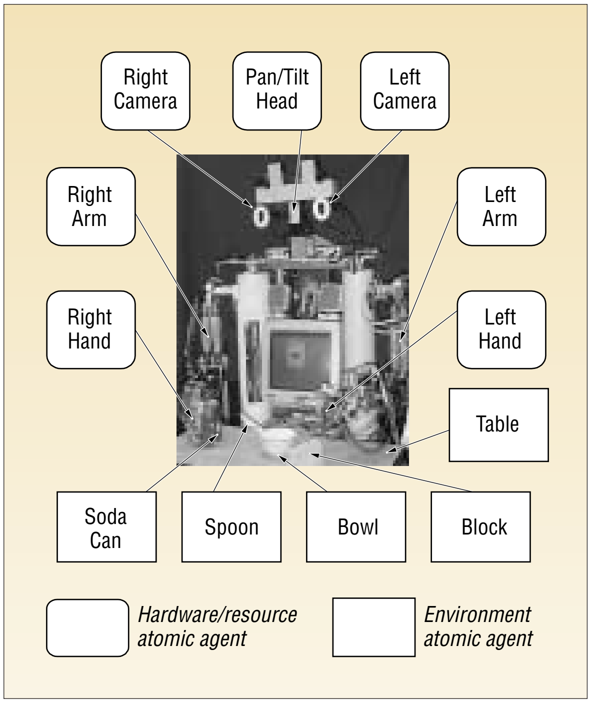

Figure 3. Examples of atomic agents for ISAC.

coupled, asynchronous operation of decision-making agents simplifies the system model at a high level. Overspecification of a system's higher levels can lead to nonrobust operation; a collection of asynchronously executing atomic agents is more stable.

An IMA agent taxonomy. There are four basic types of atomic agents and a fifth type that exists as a concession to realistic implementation considerations:

- *Hardware/resource agents* provide an interface to sensor or actuator hardware. Those interfacing to sensors can provide other atomic agents with regular updates of sensor data. Those interfacing to actuators accept commands from other atomic agents and provide updates of current state to other atomic agents.
- *Behavior/skill agents* encapsulate basic behaviors or skills.
- *Environment agents* provide an abstraction for dealing with objects in the robot's environment. This abstraction includes operations that the robot can perform on the object—for example, "look at."
- *Sequencer agents* perform a sequence of operations, often interacting with one or more environment agents or activating and deactivating one or more atomic agents. Sequencer agents can call other sequencer agents, but there should be an identifiable highest-level sequencer agent.
- *Multitype agents* combine the functionality of at least two of the first four types. For example, combining the hardware and behavior types into a single multitype agent might provide implementation efficiency.

Figure 3 shows examples of atomic agents for ISAC.

A *compound agent* is an interacting group of atomic agents that one or more sequencer agents coordinate or sequence. The highest-level sequencer agent can be envisioned as the root node of a tree with connections and dependencies on other agents on branches.

Agent communication. The IMA runs under Windows NT 4.0. The Distributed Component Object Model is a Windows service that lets remote objects be treated as if they were local. DCOM transparently handles communication between atomic agents, which are constructed from DCOM objects.

Atomic agents communicate through message passing and have flat connectivity—any agent can, in principle, communicate with any other. Implicit hierarchies are formed, however, because all but the lowest-level atomic agents employ other agents to complete their tasks or to achieve or maintain their goals.

Although IMA does not restrict how atomic agents communicate, most interagent communication occurs in one of three ways.

Sensor atomic agents commonly use *one-way data-flow* (or *observer*) *communication*. A single sensory atomic agent acts as a data server by providing periodic data updates to one or more client agents, usually at steady intervals.

Sequencer agents and environment agents usually use *master–slave communication*. A master sends its slave a signal to begin an operation. When the slave has completed its operation, either it sends an acknowledgment signal or the master can detect that it has finished by monitoring the world state.

Actuator atomic agents commonly use *command-in and position-out communication*. A single actuator atomic agent acts as a server by accepting commands from one or more clients. This actuation server performs some sort of command arbitration (for example, vector sum[6] or subsumption[4]). The server also provides the clients with information about its current state (for example, actuator position). This type of communication can be considered a mixture of observer and master–slave communication.

An HHI framework

For a humanoid robot to become a useful assistant to people, robust human–humanoid interaction is necessary. HHI differs from traditional human–robot interaction in that a humanoid should recognize not only human physical aspects but also their cognitive aspects, such as frustration, confusion, and joy.

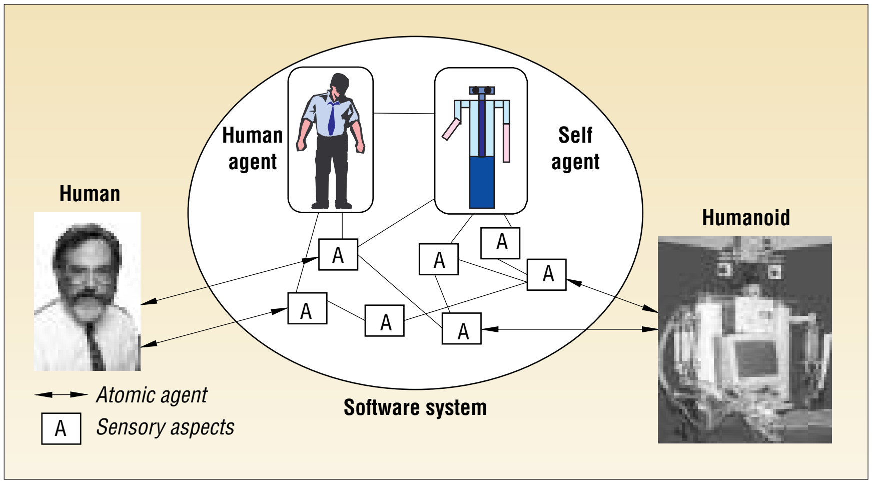

Figure 4. Human–humanoid interaction in the Intelligent Machine Architecture.

Table 1. Aspects of a human–humanoid interface.

Aspects	Human	Humanoid
Physical	Face Hand	Structure Workspace
Sensory	Audio Visual	Audio Visual Infrared
Cognitive	Emotion Goals	Affect Knowledge base

Table 1 lists aspects of humans and humanoids to consider when designing their interaction. For a maximally effective HHI interface, the robot should be able to perceive a person's physical and sensory aspects and to communicate with him or her. Modes of human expression include vocal utterances, facial expressions, gazes, and gestures. The designer should consider the nature of these modes to enable the system to respond to them. To respond to humans, humanoids can speak using text-to-speech technology, gesture using their manipulators, and make facial expressions, if so equipped. Cognitive aspects are the most difficult to implement. How to model human emotion and humanoid affect are open research questions.

Two compound agents, the *Human* and *Self agents*, facilitate HHI for ISAC. Figure 4 shows our implementation of HHI in the IMA based on these agents.

Human agent. This agent represents the human; it comprises the atomic agents that are tuned to human features. The Human agent encapsulates the information that the humanoid has determined about the human. This lets the system interact intelligently with the human by sensing these features dynamically. The Human agent detects, monitors, and identifies humans throughout its interaction. The agent's current implementation includes four modules logically grouped by function (see Figure 5a).

The *detection module* finds human features—currently faces and hands. Infrared motion detection and sound source localization can turn the system's attention toward new humans in its environment. Upon initiation, the Human Face agent connects with the cameras and initiates face tracking. The Human Hand agent determines the approximate region of the human's hand, based on the head position.

The *monitor module* tracks and perceives humans in various modes. The system uses audio, visual, and infrared sensors. The module uses information from the sensory atomic agents associated with these sensors. The cameras visually track features. The Human Motion Detection agent monitors an infrared sensor array and directs the system's attention toward human motion.

The *identification module* determines individuals in the system's environment. This module monitors features, such as a person's height and clothing color, to identify and detect changes among the people in its environment.

The *interaction module* monitors speech and gesture input. It passes speech commands to the Self agent and interprets gesture in the context of speech. The Human Finger agent detects a human pointing with its finger. This agent lets the human supplement its speech by diectic gesturing.

Self agent. This agent addresses the humanoid's cognitive aspects. The physical and sensory aspects are mapped to various atomic agents.

As part of the Self agent's cognitive aspect, it determines the human's goals from the information it receives from the Human agent. It also selects actions for the humanoid to achieve these goals. The agent selects actions by activating sequencer and environment atomic agents. As another part of the cognitive aspect, the Self agent integrates failure information from the atomic agents that evaluate the system status at lower levels. It also maintains information about the humanoid's overall state.

The Self agent also generates part of the humanoid's communication to the human. The agent does this in response to the human's input, acknowledging receipt of commands. It also takes initiative in generating output (for example, asking for help) when the status information that it maintains indicates a problem with the humanoid's functioning.

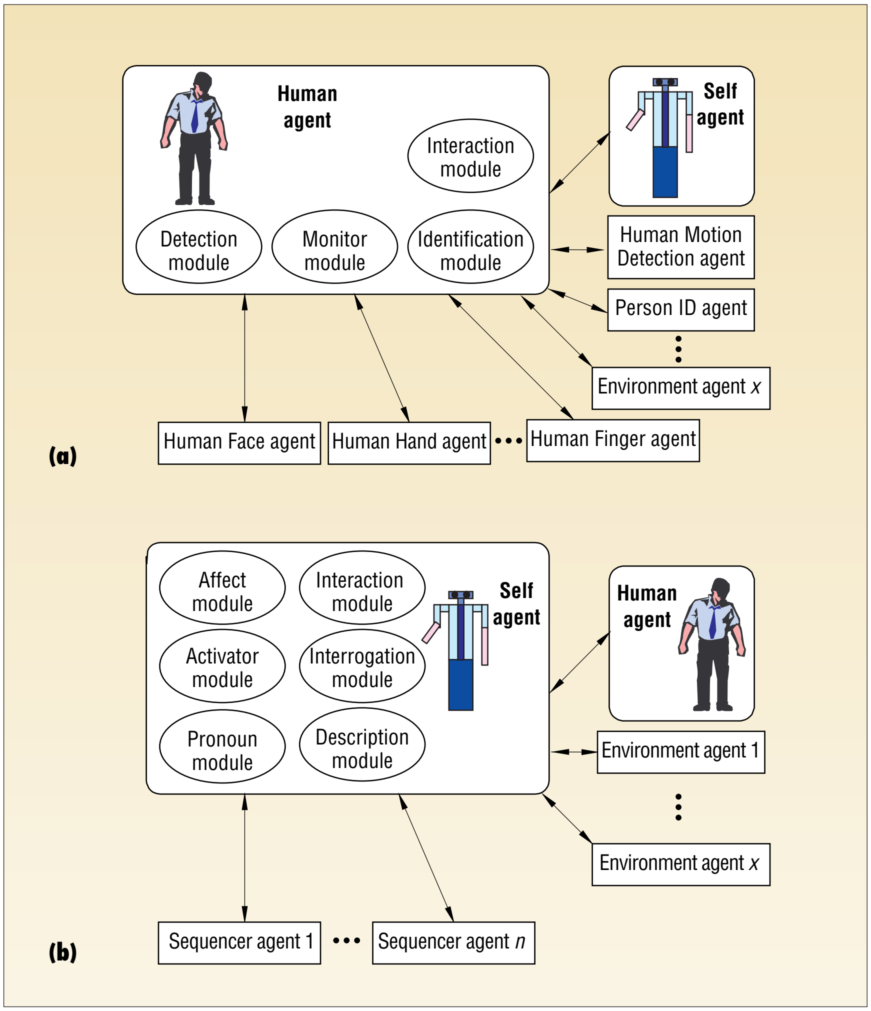

Figure 5. The (a) Human agent and (b) Self agent and their constituent modules in relation to other agents.

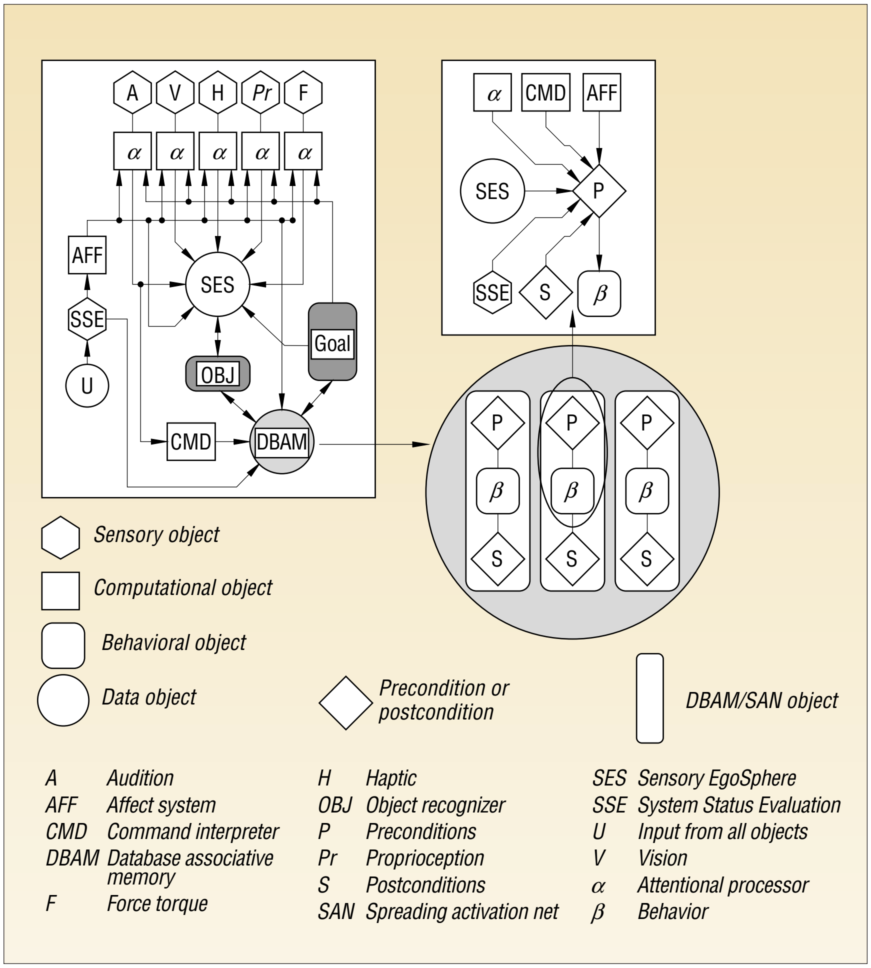

Figure 6. ISAC's control system: A sensory-motor control perspective.

Figure 5b shows the Self agent's current implementation.

The *affect* and *description modules* maintain information about the system's status. The affect module is an artificial emotional model that describes the humanoid's current state and provides internal feedback to the humanoid's software agents, similar to what Cynthia Breazeal described.[8] The description module provides descriptions of the atomic agents and information about their status: whether they are active and whether they are achieving their goals.

The *interaction*, *interrogation*, and *activator modules* interpret the human's input and generate responses to the human or increase an atomic agent's activation level.

The *pronoun module* acts as a conversational referent—that is, a pointer to what the human is talking about. It resolves human words and phrases such as "it," "this," and "the blue block" to environment agents. The pronoun module then acts as a pointer or reference for use by sequencer agents. This concept is similar to Minsky's pronome.[7]

The control system

Until recently, robots were useful only in highly structured environments such as assembly lines, where they perform repetitive tasks. Robots have been quite unlike animals, which must react and adapt to survive while pursuing their goals in a dynamic environment. Moreover, a successful animal is an opportunist; it can recognize and exploit unexpected events that are beneficial. To be an effective assistant to people in a human-centered environment, a robot must likewise be able to react and adapt to the flow of events and to recognize opportunities.

To endow the robot with the capacity to react, adapt, and recognize opportunity, we have designed the ISAC control system with structures analogous to those in mammalian brains. Concurrent sensory-processing modules gather audio, visual, haptic, proprioceptic, and force-torque information from the environment. The system filters these sensory streams through attentional networks that detect important events and ignore others. The actions of the attentional networks are functions of the desired goal state, the current task, and other factors. Incoming sensory information is stored in a short-term memory.

Sensory-motor coordination. SMC is fundamental to the development of robot intelligence.[9] It provides the basis for physical control over objects. Both sensory and motor processes play an integral role in perception. SMC uses correlations between sensory events and motor actions to reduce the dimensionality of both the perceptual and motor control spaces. SMC by itself, however, is useless if it is not coupled with a capacity to learn those sensory-to-motor couplings that lead the robot to success. This can be accomplished in several ways, all of which entail the forming of spatiotemporal associations between sensory and motor events.

When you view our control system from the SMC perspective, it has a very different structure from the high-level agents shown in Figures 4 and 5 (see Figure 6). The system has parallel sensory processing (shown as hexagons in the upper left box of Figure 6) and motor control modules, one for each sensory modality and one for each appendage. These modules interface with the robot hardware and perform the computations necessary to acquire and format sensory data and to provide actuators with appropriate control parameters. The result of these modules is a collection of sensory data streams and motor control sequences.

The Sensory EgoSphere. All ISAC's attended sensory information is registered on a 2D short-term memory structure we call the *Sensory EgoSphere*.[10] The SES contains a relatively simple data structure in which the sensory-attention modules record the current sensory data segment's acquisition time and spatial location.

The System Status Evaluation. In the ISAC control system, an atomic agent's failure to complete its task is not catastrophic. It is simply one possible outcome of normal operation. The *System Status Evaluation*, a low-level performance analyzer in each atomic agent, monitors atomic-agent failures. The SSE supplies an affect system that computes a global context vector as a dynamical function of the SSE events and sensory events.

SSE strategy. Our strategy for the SSE

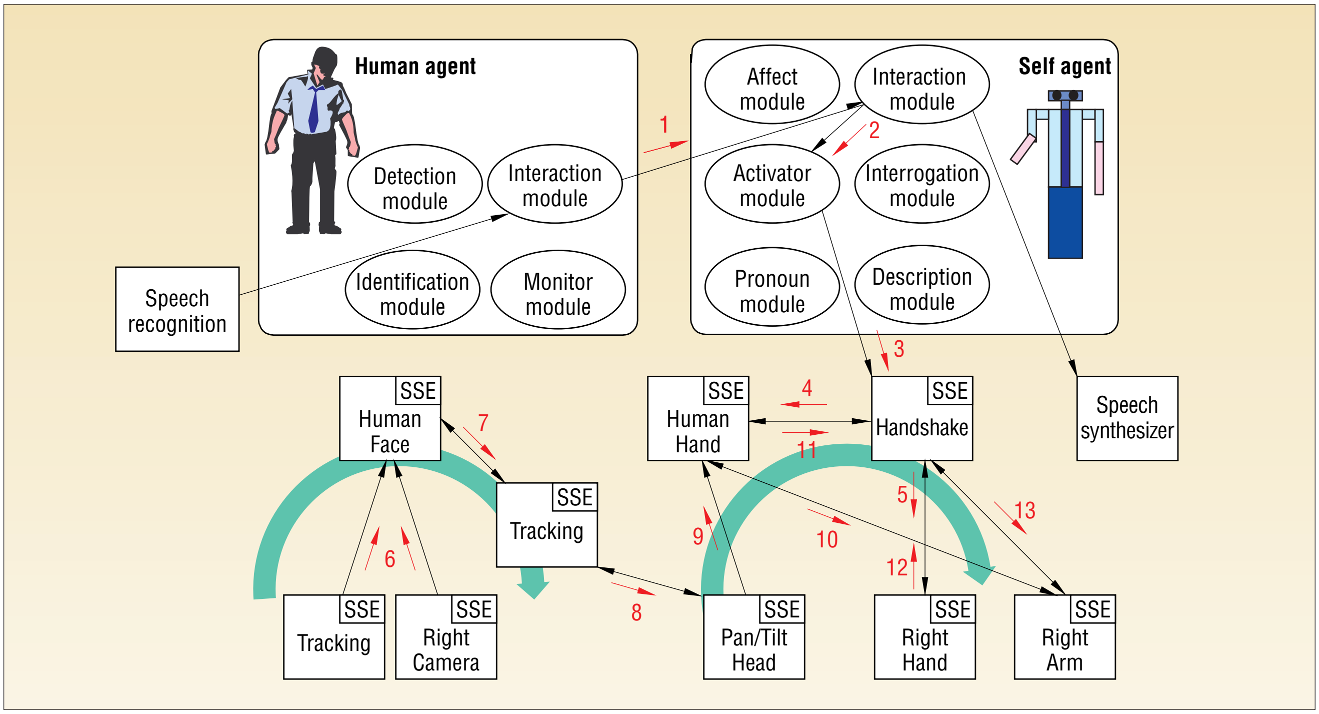

Figure 7. Agents for the handshaking demonstration from the Self agent's viewpoint. The red numbers and arrows indicate communication during the demonstration. SSE stands for System Status Evaluation.

attempts to infer causes of agent errors by using information about communication with other atomic agents. We base this strategy on these assumptions:

- If an agent cannot meet its goals (as defined by task-specific criteria) and is experiencing abnormal communication with another agent, the other agent has a problem.
- An atomic agent's failure to meet its goals will cause abnormal communication to propagate upward in the atomic-agent hierarchy.

Detection of abnormal communication with another agent is based on communication timing patterns. Each agent collects statistics on the timing of communication with other agents. For one-way data-flow communication, the quantity of interest is the elapsed time between successive updates. For master–slave communication, we are interested in the elapsed time between the initial signal and the completion of the slave's operation.

Each agent collects histograms of the pertinent communication timing. Using the histogram data, each agent classifies the status of its communication with other agents as normal or abnormal.

A logical hierarchy of agent communication. We can define a logical hierarchy of atomic agents based on the flow of agent communication. Agents can measure the timing of communication with agents directly below them in the hierarchy. We assume that one agent's error will cause abnormal communication to propagate up the hierarchy because, in general, the atomic agents higher up require the services of those below. If this assumption is accurate, we should expect that the highest-level atomic agents will eventually detect some abnormal communication and that we can begin any debugging session with them. Additionally, we should expect to find the error's source by tracing down the hierarchy through any agents that experienced abnormal communication.

SSE limits. Obviously, we cannot expect this strategy to work for all possible errors. Based on our experience with the IMA, we've developed these error categories:

- *Systemic* errors are related to hardware and software infrastructure.
- *Physical-control* errors are failures to achieve a trajectory in a space.
- *Classification* errors occur in mapping stimuli to symbols or behavior.
- *Policy* errors occur in choosing control and classification tasks to achieve a goal (planning).
- *Goal identification* errors occur in determining the user's goal.

We are investigating the SSE's ability to detect errors of these different types.

A handshaking demonstration

In this demonstration, ISAC interacts with a human in several ways. Each atomic agent in the demonstration adheres to a simple script, but the interplay among atomic agents—and with the human—provides for a variety of situations. This demonstration centers on a small number of fundamental behaviors that we are implementing for ISAC, including

- looking at a person who is talking,
- tracking the person visually as he or she moves, and
- reaching out with its hand to meet the person's hand.

A typical script begins when ISAC detects a human's presence and begins tracking the human's face. At this point, the interaction could take one of several paths. For example, if the human is outside ISAC's workspace, ISAC might ask the human to come closer before attempting to shake hands. Or, if the human is in ISAC's workspace, ISAC might initiate handshaking.

Figure 7 shows the interconnections among some atomic agents in the system. The bottom two layers of atomic agents are resource and behavior atomic agents. These agents—Left and Right Camera, Pan/Tilt Head, Right Hand, Right Arm, and Tracking—would be used in almost any task that ISAC performs. The Human Hand and Human Face agents are environment agents. Human Face supports a look-at operation that causes ISAC to track the human's face. Human Hand supports a move-to operation that causes ISAC to move the end-effector of one of its manipulators to the human's hand. In response to input from the Human agent, the Self agent activates the Handshake sequencer agent.

Operation of the handshaking sequencer. The agent's core is a state machine; Figure 8a shows its state transition diagram. When the Human agent detects a human's presence, the Self agent sends an activation signal to the Handshake agent. This causes Handshake's state machine to move from State 0—an "idle" state—to State 1. In State 1, Handshake sends a signal to Human Hand to activate its move-to operation and a signal to Hand to activate its auto-close behavior.

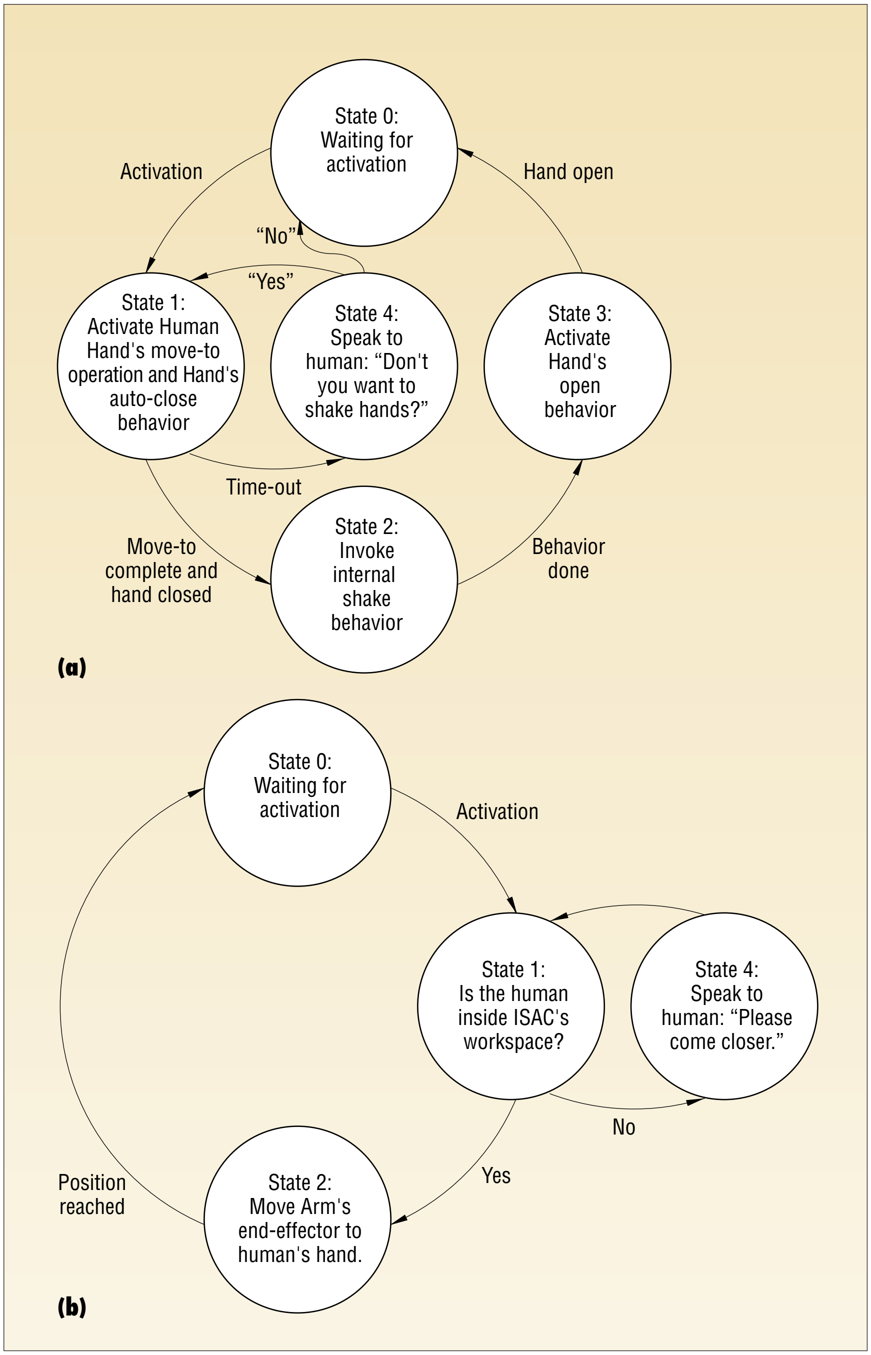

Figure 8. State transition diagrams for the (a) Handshake sequencer agent and (b) Human Hand environment agent.

Figure 8b shows Human Hand's state transition diagram. When Handshake activates the move-to operation, Human Hand checks to ensure that the human is in ISAC's workspace. The agent does this by querying the Arm agent for the workspace boundaries; it can then compare the human's position with the boundary information. If the human is in the workspace, the agent sends commands to the Arm agent to move ISAC's arm toward the human's hand. However, if the human is outside the workspace, the agent will cause ISAC to ask the human to move closer.

Agent communication flow. The red arrows and numbers in Figure 7 show a typical communication flow among agents during the handshaking demonstration. The figure also shows how groups of agents can form sensory-motor behavior loops. One of these, a visual-tracking loop, involves Left Camera, Right Camera, Human Face, Tracking, and Pan/Tilt Head. The other, a visually guided motion loop, involves Pan/Tilt Head, Human Hand, and Right Arm.

Example dialog. The following is a dialog between the human and humanoid:

Human: "Hello."
Humanoid: "Shake!"
Humanoid: "Please move closer."
Humanoid: "Don't you want to shake hands?"
Humanoid: "Something is wrong."
Human: "What's wrong?"
Humanoid: "I had a problem trying to shake hands."
Human: "What happened?"
Humanoid: "I was asking the human to shake hands when timeout happened."

During this trial, the human was standing outside the humanoid's workspace when he greeted the humanoid. The humanoid responded by inviting the human to shake hands and asking him to move closer. However, the human did not approach the humanoid. The humanoid then asked the human if he really wanted to shake hands. Because the human did not respond, the Handshake agent reported a failure to the

Self agent. The Self agent then caused the humanoid to report that something was wrong. In response to the human's queries, the humanoid explained what happened.

THROUGH ISAC, WE ARE WORKING to achieve efficient and effective human–humanoid teaching. We presume that intelligence is an emergent property of an autonomous agent (person, creature, or robot); that intelligent behavior requires the coupling of mind, body, and world; and that the agent itself must learn that coupling's details. Therefore, rather than programming ISAC directly to perform various tasks, we are programming the robot to learn, from its own experiences, how to perform its tasks. We call this approach *learning SMC*.

We are creating a control system that will let ISAC learn SMC while being led repeatedly by a person through a sequence of behaviors to perform a task. During these trials, the robot will identify those sensory events that consistently co-occur with specific changes in motor state. We call such a co-occurrence an *SMC event*. We believe that the SMC events associated with a task—if encoded as competency modules for a spreading activation network—will let the robot perform the task autonomously. As the number of learned competency modules increases, the spreading activation network will tend to select the most efficient sequence of behaviors to achieve a goal whether or not ISAC explicitly learned that sequence.

With the NASA Robonaut team at the Johnson Space Center, we plan to develop the world's first natural teaming arrangement between a human (for example, an astronaut) and a humanoid (for example, ISAC or Robonaut). (For more on Robonaut, see "Robonaut: NASA's Space Humanoid," in this issue.) One development objective is a natural teaming arrangement, where the humanoid serves as the junior partner, handing tools to the human on request and receiving tools from the human. By the project's end, the humanoid should be able to manage a tool chest of instruments to support the astronaut's work. Repetitive tool handling (simpler than tool use) is a central set of skills that we believe can be automated, letting the robot's remote supervisor respond more quickly to the human's requests.

We are firmly committed to the interactive human–humanoid robot teaming paradigm and believe that it will lead to the successful integration of humanoid robots into society. The hypothesis that a robot must develop intelligence through its own interactions with the environment is relatively new. To be validated (or refuted), the hypothesis must be tested under controlled conditions. Our research with ISAC is helping us achieve this.

Acknowledgments

The DARPA Mobile Autonomous Robotics Systems program partially funded this work.

References

1. K. Kawamura et al., "Intelligent Robotic Systems in Service of the Disabled," *IEEE Trans. Rehabilitation Eng.,* Vol. 3, No. 1, Mar. 1995, pp. 14–21.
2. K. Kawamura et al., "Design Philosophy for Service Robots," *Robotics and Autonomous Systems*, Vol. 18, Nos. 1–2, July 1996, pp. 109–116.
3. D.M. Wilkes et al., "Designing for Human-Robot Symbiosis," *Industrial Robot*, Vol. 26, No. 1, 1999, pp. 49–58.
4. R.A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE J. Robotics and Automation*, Vol. 2, No. 1, Mar. 1986, pp. 14–23.
5. R.T. Pack et al., "A Software Architecture for Integrated Service Robot Development," *Proc. 1997 IEEE Conf. Systems, Man, and Cybernetics*, IEEE Press, Piscataway, N.J., 1997, pp. 3774–3779.
6. R.C. Arkin, "Integrating Behavioral, Perceptual, and World Knowledge in Reactive Navigation," *Robotics and Autonomous Systems*, Vol. 6, No. 1, June 1990, pp. 105–122.
7. M. Minsky, *The Society of Mind*, Simon and Schuster, New York, 1985.
8. C. Breazeal, "A Motivational System for Regulating Human–Robot Interaction," *Proc. 15th Nat'l Conf. Artificial Intelligence and 10th Innovative Applications of Artificial Intelligence Conf.* (AAAI '98/IAAI '98), AAAI Press, Menlo Park, Calif., 1998, pp. 54–61.
9. R. Pfeifer and C. Scheier, "Sensory-Motor Coordination: The Metaphor and Beyond," *Robotics and Autonomous Systems*, Vol. 20, Nos. 2–4, June 1997, pp. 157–178.
10. J.S. Albus, "Outline for a Theory of Intelligence," *IEEE Trans. Systems, Man, and Cybernetics*, Vol. 21, No. 3, May/June 1991, pp. 473–509.

Kazuhiko Kawamura is the director of the Center for Intelligent Systems and a professor of electrical engineering and computer engineering at Vanderbilt University. He is also professor of management of technology. His research interests include intelligent robotic systems, human–robot symbiosis, and intelligent manufacturing systems. He received his PhD in electrical engineering from the University of Michigan. Contact him at the Center for Intelligent Systems, Electrical Eng. and Computer Science Dept., Box 131 Station B, Vanderbilt Univ., Nashville, TN 37235; kawamura@vuse.vanderbilt.edu; www.vuse.vanderbilt.edu/~kawamura.

R. Alan Peters II is an associate professor of electrical engineering at Vanderbilt University. His research interests are intelligent robotics, computer vision, and image processing through mathematical morphology. He received his PhD in electrical engineering from the University of Arizona, where he was a fellow of the American Electronics Association. Contact him at the Center for Intelligent Systems, Electrical Eng. and Computer Science Dept., Box 131 Station B, Vanderbilt Univ., Nashville, TN 37235; rap2@vuse.vanderbilt.edu; www.vuse.vanderbilt.edu/~rap2/AlsMainPage.html.

D. Mitchell Wilkes is an associate professor of electrical engineering and computer engineering at Vanderbilt University and an assistant director of the Center for Intelligent Systems. His research interests include intelligent robotics and control, signal processing, image processing, and intelligent manufacturing. He received his PhD in electrical engineering from Georgia Tech. Contact him at the Center for Intelligent Systems, Electrical Eng. and Computer Science Dept., Box 131 Station B, Vanderbilt Univ., Nashville, TN 37235; wilkes@vuse.vanderbilt.edu; www.vuse.vanderbilt.edu/~eceinfo/faculty/wilkes.html.

W. Anthony Alford is a research assistant at the Center for Intelligent Systems at Vanderbilt University. His research interests are human–humanoid interaction and humanoid system analysis and evaluation. He received his MS in electrical engineering from the University of Tennessee Space Institute. Contact him at the Center for Intelligent Systems, Electrical Eng. and Computer Science Dept., Box 131 Station B, Vanderbilt Univ., Nashville, TN 37235; alford@vuse.vanderbilt.edu.

Tamara E. Rogers is a PhD student in the Center for Intelligent Systems at Vanderbilt University. Her current work deals with human–system interaction. She received her MS in electrical engineering from Vanderbilt University. Contact her at the Center for Intelligent Systems, Electrical Eng. and Computer Science Dept., Box 131 Station B, Vanderbilt Univ., Nashville, TN 37235; tamarar@vuse.vanderbilt.edu; www.vuse.vanderbilt.edu/~tamarar.