Simulating the Knowledge Environment for Autonomous Construction Robot Agents

Boyd Paulson, Thomas Froese and Lai-Heng Chua
Department of Civil Engineering
Stanford University
Stanford, CA 94305-4020

ABSTRACT

Future intelligent construction machines must harness considerable knowledge to plan and control autonomous tasks in spite of the fact that they will be limited in their own pre-defined knowledge. This paper first describes the need to discover and formulate a general core of theory and software for such machines so that they can access and communicate with knowledge sources in their environment. It next describes current research to simulate characteristics of the knowledge environment for robot agents. An example will illustrate an early implementation effort that uses object-oriented programming for such a simulation. This research will provide a theoretical base for the knowledge environment to sustain automation research for autonomous robots working in real and very challenging field conditions.

INTRODUCTION AND BACKGROUND

In future construction field environments, intelligent machines, like their human counterparts, will need to harness considerable knowledge to plan and control autonomous tasks in spite of the fact that they will be limited in their own knowledge and abilities. However, no unifying theory and few guidelines exist for defining and communicating knowledge about designs and field operations in a way that can effectively be utilized by such machines. At Stanford and elsewhere, researchers are working toward such a theory in order to support the work of others on practical applications of robots in field conditions [Paulson 85, Paulson et al 89]. Some of the research focuses on cognitive aspects of future machines to endow them with common modes of "thinking" and communicating—in effect a common *culture*—so that they can work together and with humans in groups.

The scope of research needed to build theories and software to support construction robotics is vast. In general, researchers need to develop machine agents having enhanced abilities to work well in relatively unstructured and fast-changing environments. Each step in this research should lead toward a general architecture handling the knowledge an agent needs to function productively in a knowledge environment. The resulting software could then be extended by developers of applications-oriented robots to handle particular areas of expertise, whether in managing other machines or in doing specific physical tasks. This basic research would provide a platform for more rapid integration of robotic systems.

Figure 1 shows a broad conceptual view of the construction knowledge environment. It illustrates the organizational context in which the robots might be working, the interfaces to computer-aided design (CAD) databases and reasoning, interactions with other field agents—both human and machine—and interfaces to knowledge sources in the world beyond the field. To establish such an environment with robotic *hardware* is not feasible now. Even if one could afford them, robots with sufficient flexibility and computing power for diverse operations do not yet exist.

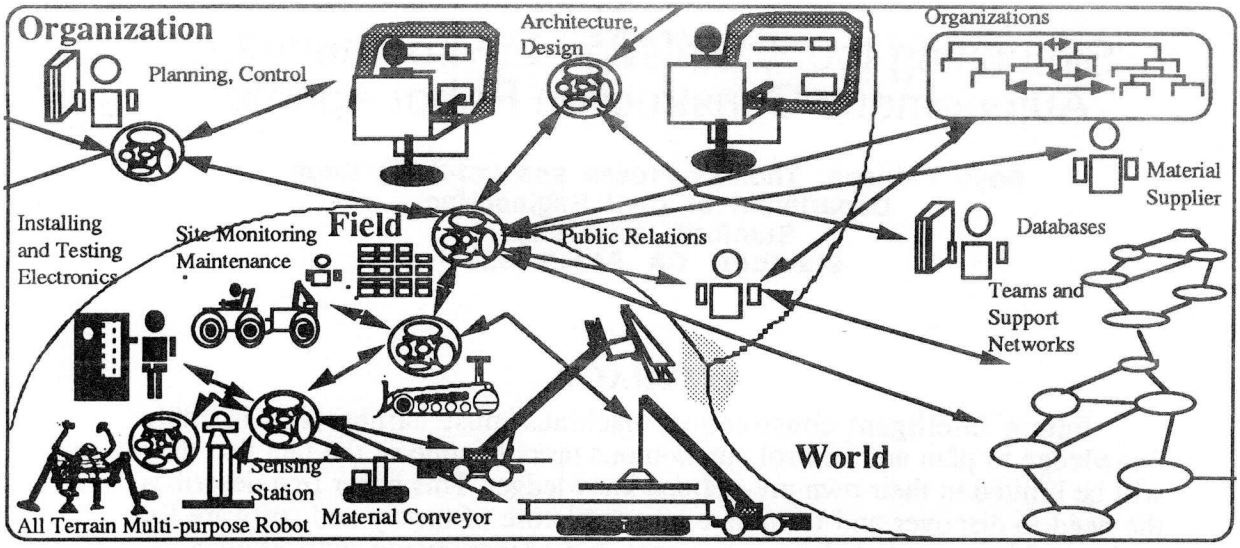

Figure 1—Conceptualization of the Knowledge Environment for Construction Automation

Stanford's research in this area consists of several active and planned projects to simulate characteristics of robot agents themselves and build a knowledge environment simulator in which to test them. The *Robot Agent Simulator* will model knowledge objects and provide means to give agents access to the knowledge environment. This paper will focus on the *Knowledge Environment Simulator*, where we are exploring and testing methods to formalize parts of the knowledge environment in machine-usable forms. The simulator will provide a test bed for concepts, theories and methods for researchers working toward distributed automation. It will further provide computer-based concepts to help integrate construction field processes into the other phases of the project life-cycle, such as planning, design and operations. The ultimate objective is to design and develop the general theory and software core for machine agents — the rudimentary "brains" of the beasts — which can then be embedded in agents specialized for particular tasks.

To create simulators as prototype test beds, some of this work focuses on representing the physical environment and providing inter-agent communication facilities for multiple agents working in the environment. Early stages of the research also seek to model three-dimensional construction space in a way that is compatible with the three-dimensional CAD systems now becoming the basis for design communications. Simulators can also implement basic physical mechanics to monitor agent motion and location, provide minimal representations to convert design specifications into incremental task objectives for robots, and implement a type of organizational hierarchy (e.g., a central supervisor agent to which task agents can turn for help).

Future avenues of research should address problems of modelling *within* robot agent simulators some "understanding" of the knowledge environment, such as key characteristics of objects and other agents, in ways useful for reasoning. Among other things, researchers should seek to reduce the knowledge that needs to be encoded in machine systems *a priori* by enabling them to tap the vast knowledge sources in their environment when needed. Automatons should be able to assemble knowledge and enlist other agents needed to perform a task and respond dynamically to change. For example, robot reasoning and control software should deal with unexpected obstacles, road conditions, failure of a machine positioning system, damaged material, improper tools, or imprecise instructions.

RELATED AND SUPPORTING RESEARCH

Related research occurs in several main categories: (1) general theories for simulating the knowledge environment and knowledge agents; (2) implementation methodologies; (3) distributed processing systems and concurrent programming; and (4) artificial robot languages. This section will concentrate on numbers 1, 2 and 3, but only selected topics can be highlighted in the space available here.

Key artificial intelligence areas include *blackboards* [Hayes-Roth 85] and *general problem solving architecture* [Laird et al 87]. The blackboard model started as a model of knowledge encoding for opportunistic problem solving [Erman et al 81]. [Kellogg 83, Kogan 84 and Koo 87] proposed construction of *intelligent assistants* for information management. [Euzenat et al 85] looked at intelligent database access—that is, providing an intelligent interface for databases rather than just a natural language one. In line with this has been work on *deductive databases* [Gallaire et al 85]. [Ingwersen 86, Lebowitz 86, and Shaw and Gaines 86] took it further towards an *intelligent information system.*

To support object-oriented implementations of such research, Smalltalk-80 [Goldberg and Robson 83] has been the main programming language. Since inception it has been used for a wide range of applications from AI type systems to ordinary algorithmic programming. Other important languages in this area include C++ [Stroustrup 86] and Objective C (The Stepstone Corporation). ThingLab [Borning 81; Farrah and Borning 86], a constraint-based language developed at the Xerox Palo Alto Research Center, can be used to build simulations of real-world systems. Alternate Reality Kit [Smith 86] is a simulation kit for interactive physics. Another simulator written in Smalltalk is called INSIST [Meulen 87]. ACTRA [LaLonde et al 87] is a distributed object-oriented computer system based on Smalltalk and targeted for industrial applications such as flexible manufacturing, simulation, control, CAD/CAM and project management. MACE [Gasser et al 87], an acronym for *Multi-Agent Computing Environment,* is a simulation environment designed to serve as an *instrumented* test bed for building experimental distributed artificial intelligence systems even at different levels of granularity. It has powerful constructs for describing agents, their organizational structure and interaction, and could provide a useful test bed for construction robotics research. The research most closely related to that described in this paper is that by [Keirouz et al 88] at Carnegie-Mellon University. It is using Smalltalk-based object-oriented modeling of the temporal, geometric and functional evolution of constructed facilities.

Distributed systems models such as ACTORS [Agha 86, Hewitt et al 75] and distributed databases [Howard and Rehak 87] relate to this research. There has been significant work concerned with distributed problem solving [Durfee and Lesser 87, Georgeff 82, Konolige and Nilsson 80]. [Davis and Smith 83, and Koo 87] studied cooperative planning and distributed problem solving. [Corkill and Lesser 83] used the monitoring of distributed vehicles as a test bed for studying distributed problem solving. These researchers are more concerned about *global properties or behavior* of the system design while robot agents mainly need the *knowledge* to do their work and get around in their existing knowledge environment.

ENVIRONMENT SIMULATOR EXAMPLE

This section briefly introduces a prototype environment simulator being developed as a test bed for future robot agent software. The simulator replicates the process of an agent attempting to obtain information from its environment through its sensors and communication channels, and also the process of the agent performing actions which change the state of the environment. Both functions are accomplished by allowing the agent to pass messages to the environment simulator and receive response messages back.

In the current implementation, a user serves as the agent since our agent simulators have not yet been developed. The simulator acts not only as the agent's environment, but also provides some of the agent's low-level functions. For example, if a "sighted" robot wishes to find out the shape of an object, it can ask that object what it's shape is and the object will respond with a message describing its shape. An actual robot in a real environment would need to instruct its vision sensors where to point and would have to analyze the resulting signals in order to derive the object's shape. We bypass these steps in the simulator, however, since our interests lie in the robot's higher-level reasoning.

Implementation Approach. Our environment simulator is implemented in Smalltalk-80. The basic representation approach is an object-oriented *frame* scheme. Each object from the agent's environment which is to be represented within the simulator is declaratively and procedurally described by a frame. Frames contain *slots* which can hold either *attributes* of the object, *relationships* to other objects, or *procedures* related to that object. Slot *values* can be inherited along abstraction hierarchy relationships.

Messages are sent by the agent to a frame representing the environment itself, this frame subsequently passes the message on to the appropriate frame within the environment. The onus of sending appropriate messages rests mainly on the agent. For example, if a robot wishes to query a project database, it must address its message specifically to that database and must pose it's query in a form which is understandable to the database. When such a message is received by the environment, it simply determines where the message is to be directed and it passes the message on to the corresponding frame. At times, however, the interpretation of the message depends upon the context within the environment. For example, a sighted robot may attempt to discern the shape of whatever object is positioned directly in front of it without knowing what that object is. In such a case, the robot would send a message such as "what is the shape of the object directly in front of me?" to the environment. The environment frame would resolve what the object directly in front of the agent was (based on the known position of the robot, which is itself an object in the environment, and on the positions of other nearby objects). It would then pass the message on to that object's frame.

Example Scenario. This example involves earth moving. Specifically, suppose that the simulator is modelling the environment of an agent which controls the loading of a scraper under its own power (i.e., no pusher). The agent has access to several sensors and actuators on the scraper and can communicate with an equipment database and with the scraper's operator. The agent's task is to operate the scraper's throttle, transmission and bowl controls during in order to achieve optimal loading rates and durations. Figure 2 shows the primary objects represented within the simulator for this environment.

The Simulator's Performance. A sample simulator session for this scenario is summarized in Table 1, which shows the messages sent to the environment simulator by the agent (i.e., the user at present) as well as the simulator's responses. The reasoning being performed within the simulator during the course of this session includes some initial responses from other agents within the environment (the operator and the equipment database). These responses correspond to simple facts known by these other agents. The remainder of the session centers around the performance of the scraper during a loading operation. This performance is governed by simulation procedures which depend upon the values of the scraper's attributes. For example, determining the scraper's load size depends upon the scraper's speed, the cutting depth, the cutting width, the soil characteristics, and the duration of the loading; the scraper's speed is a function of it's rimpull force and the rolling, grade, cutting and loading resistances; and so on.

Some of these attribute values are set by the agent (the throttle and cutting blade positioning, for example). Others are accessible to the agent through its sensors connected to the scraper (such as the scraper speed). Still other attributes are internal and cannot be determined by the agent (the loading rate of the scraper, for example). Many procedures

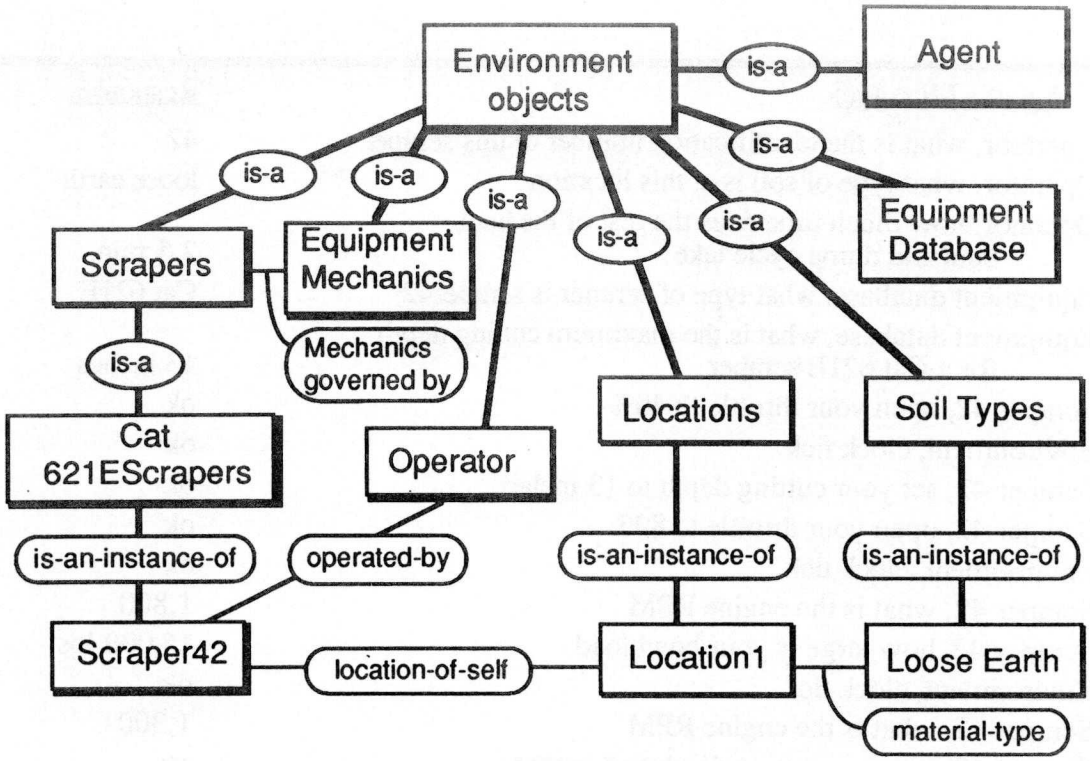

Figure 2 — Scraper Loading Example

governing the scraper's performance are associated with the equipment mechanics frame. When queries are sent to the scraper42 frame (i.e., for the specific machine), a procedure is inherited from the scrapers frame (for the general class of these machines) which uses a "mechanics-governed-by" relationship to direct them to the equipment mechanics frame. Other relationship links allow attribute values to be determined. For example, a procedure which requires the characteristics of the soil being loaded could query the scraper42 frame, which would use a "location-of-self" relation to pass the message on to the scraper's location frame (location1), which has soil type stored as an attribute. Note that we currently deal with time in the simulator by simply instructing the environment when to perform a clock tick.

The Agent's Performance. In the sample session illustrated in Table 1, the task of the agent is to control and optimize loading. This task involves gathering information from other agents (for example, determining the maximum cutting depth from the equipment database as shown in line 5), standard operating practices (such as accelerating the scraper smoothly as illustrated in lines 6 and 9), reasoning based on monitoring sensor values (such as raising the cutting blade when the engine starts to lug as shown in line 15), and reasoning about the consequences of it's actions (such as stop loading at a certain point based on optimization of the overall duration of the load, haul, dump, return cycle)

The scenario illustrated here is not complex, but it illustrates several goals of our endeavors. With this simulator we are beginning to determine what knowledge is required to represent and to model a construction environment. We are also learning how to appropriately organize environment knowledge and how to achieve interaction between bodies of knowledge. This information is important not only to the environment simulator project, but also to the robot agent project since the robot will need to store some model of its environment within its own state.

	Agent's Message:	Response:
1.	Operator, what is the identification number of this scraper	42
2.	Operator, what type of soil is at this location	loose earth
3.	Operator, how much time does the rest of the haul haul and dump cycle take	2.5 min.
4.	Equipment database, what type of scraper is scraper42	Cat 621E
5.	Equipment database, what is the maximum cutting depth for a Cat 621E scraper	13 inches
6.	Scraper 42, open your throttle to 40%	ok
7.	Environment, clock tick	ok
8.	Scraper 42, set your cutting depth to 13 inches	ok
9.	Scraper 42, open your throttle to 80%	ok
10.	Environment, clock tick	ok
11.	Scraper 42, what is the engine RPM	1,800
12.	Scraper 42, how large is your bowl load	15,000 lbs
13.	Environment, clock tick	ok
14.	Scraper 42, what is the engine RPM	1,300
15.	Scraper 42, set your cutting depth to 8 inches	ok
16.	Scraper 42, how large is your bowl load	28,500 lbs
17.	Environment, clock tick	ok
18.	Scraper 42, what is the engine RPM	1,700
19.	Scraper 42, how large is your bowl load	42,000 lbs
20.	Scraper, set your cutting depth to 0	ok
21.	Operator, the scraper is loaded	thanks

Table 1 — Record of Example Scraper Scenario

In terms of providing a test bed for future robot simulators, we are presenting many challenges to such agents even with simple scenarios such as this. In order to successfully navigate this simulation, the agent must interact with other agents in the environment, must develop a plan of action based on information from its surroundings, must monitor the results of it's actions and modify its plans accordingly, and must perform complex reasoning about the results of its actions.

CONCLUSION

This has been a brief overview of a complex fabric of interwoven research needs. Machine agents in complex field environments will have to deal with many difficulties. The only tractable approach in the short term is to limit the use of machines to environments that are—or can be—carefully structured, or to leave most of the control and sensing with human agents. Researchers must look further to develop the underpinnings for more capable machines. We must discover and formulate the general computer-based reasoning and communications core for machines that would provide them with the ability to deal with unexpected events to a greater extent and exploit the opportunities provided by the

knowledge environment. In particular, the kind of machines envisaged will be able to deal with uncertainty, adapt to a dynamically changing environment, be able to seek knowledge beyond their spheres, and work in teams to perform complex tasks. Future research should provide a more robust basis for integration of machines with human organizations, with other machines and with a multitude of tools of production.

ACKNOWLEDGEMENT

For this research we are fortunate to be part of the Center for Integrated Facility Engineering (CIFE) at Stanford, which is operated jointly by the Departments of Civil Engineering and Computer Science and has corporate members from the design, construction and computer industries and representatives from owners and government agencies. We are also indebted to The National Science Foundation, which, through a series of grants (including MSM 83-12350, MSM 84-06852 and MSM 86-14238) has provided the intellectual freedom to explore the basic research that led to the formulation of the research agenda described herein.

REFERENCES

Agha, G. A., 1986. *Actors*, MIT Press, Cambridge, Massachusetts.

Borning, A., 1981. "The Programming Language Aspects of ThingLab, A Constraint-Oriented Simulation Laboratory," *ACM Transactions on Programming Languages and Systems,* Oct.

Corkill, D. D., and Lesser, V. R., 1983. "The Use of Meta-level Control for Coordination in a Distributed Problem Solving Network," *Proc. 8th International Joint Conference on Artificial Intelligence*, Vol. 2, pp. 748-756.

Davis, R., and Smith, R. G., 1983. "Negotiation as a Metaphor for Distributed Problem Solving," *Artificial Intelligence Journal,* Vol. 20, pp. 63-109.

Durfee, E. H. and Lesser, V. R., 1987. "Using Partial Global Plans to Coordinate Distributed Problem Solvers," *Proc. 10th International Joint Conference on Artificial Intelligence,* Vol. 2, pp. 875-883.

Erman, L. D., London, P. E., and Fickas, S. F., 1981. "The Design and an Example Use of HEARSAY," *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Los Altos, CA, pp. 409-415.

Euzenat, B., Normier, B., and Ogonowski, A., 1985. "SAPHIR+RESEDA, A New Approach to Intelligent Data Base Access," *Proc. 9th International Joint Conference on Artificial Intelligence*, Vol. 2, pp. 855-857.

Farrah, T., and Borning, A., 1986. *A User Interface for ThingLab Based on Direct Manipulation of Physical Objects,* Technical Report 86-09-02, Department of Computer Science, University of Washington, Sep.

Gallaire, H., Minker, J., and Nicolas, J. M., 1985. "Logic and Databases: A Deductive Approach," *ACM Computing Surveys*, Vol. 16, No. 2, (Jun.), pp. 153-185.

Gasser, L., Braganza, C. and Herman, N., 1987. "MACE: A Flexible Test bed for Distributed AI Research,"

Georgeff, M., 1982. "A Theory of Action for MultiAgent Planning," *Proc. 2nd National Conference on Artificial Intelligence*, AAAI-82, pp. 121-125.

Goldberg, A., and Robson, D., 1983. *Smalltalk-80: The Language and its Implementation*, Addison-Wesley Publishing Company, Menlo Park.

Hayes-Roth, B. et al. 1985. "Blackboard Architecture for Control," *Journal of Artificial Intelligence*, Vol. 26, pp. 251-321.

Hewitt, C., Bishop, P., and Steiger, R., 1975. "A Universal Modular ACTOR-Formalism for Artificial Intelligence," *Proc. 3rd International Joint Conference on Artificial Intelligence*, Vol. 1, pp. 235-245.

Howard, H. C., and Rehak, D. R., 1987. "KADBASE — A Prototype Expert System-Database Interface for Integrated CAE Environments," *Proceedings, AAAI-87, Sixth National Conference on Artificial Intelligence,* Seattle, Washington, (July).

Ingwersen, P., 1986. "Cognitive Analysis and the Role of the Intermediary in Information Retrieval," *Intelligent Information Systems*, progress and prospects, Roy Davis, Ed., Ellis Horwood Limited, pp. 206-237.

Keirouz, W., Rehak, D., and Oppenheim, I., 1988. "Issues in Domain Modelling of Constructed Facilities," *Fifth International Symposium on Robotics in Construction,* Tokyo, Japan, (June 6-8), pp. 341-350.

Kellogg, C. H., 1983. "Intelligent Assistants for Knowledge and Information Resources Management," *Proc. 8th International Joint Conference on Artificial Intelligence*, Vol. 1, pp. 170-172.

Kogan, D., 1984. "The Manager's Assistant — An Application of Knowledge Management," *Proceedings of the International Conference on Data Engineering*, Institute of Electrical and Electronics Engineering, IEEE Computer Society, Los Angeles, CA, (Apr.), pp. 592-595.

Konolige, K. and Nilsson, N. J., 1980. "Multiple-Agent Planning Systems," *Proc. 1st National Conference on Artificial Intelligence*, AAAI-80, pp. 138-141.

Koo, Charles C., 1987. *Synchronizing Plans among Intelligent Agents via Communication,* thesis submitted to Stanford University in partial fulfillment of the requirements for the degree of Ph.D., August.

Laird, J. E., Newell, A., and Rosenbloom, P. S., 1987. "SOAR: An Architecture for General Intelligence," *Artificial Intelligence Journal.*

LaLonde, W. R., Thomas, D. A. and Johnson, K. 1987, "Smalltalk as a Programming Language for Robotics?" *IEEE International Conference on Robotics and Automation,* Raleigh, North Carolina, Mar.

Lebowitz, M., 1986. "An Experiment in Intelligent Information Systems: RESEARCHER," *Intelligent Information Systems*, progress and prospects, Roy Davis, Ed., Ellis Horwood Limited, Chichester, West Sussex, England, pp. 127-150.

Meulen, P. S. 1987, "INSIST: Interactive Simulation in Smalltalk," *Proceedings of the 1987 OOPSLA Conference,* Orlando, Florida, Oct.

Paulson, Boyd C., Jr., 1985. "Automated Control and Robotics for Construction," *Journal of Construction Engineering and Management*, ASCE, Vol. 111, No. 3, (Sep.), pp. 190-207.

Paulson, Boyd C., Jr., Nadeem Babar, Lai-Heng Chua and Thomas Froese, 1989. "Simulating Construction Robot Agents and their Knowledge Environment," CIFE Working Paper No. 003, Stanford University, (January).

Shaw, M. L. G., and Gaines, B. R., 1986. "A Cognitive Model for Intelligent Information Systems," *Intelligent Information Systems,* progress and prospects, Roy Davies, Ed., Ellis Horwood Limited, Chichester, West Sussex, England, pp. 238-258.

Smith, R. B., 1986, "The Alternate Reality Kit: An Animated Environment for Creating Interactive Simulations," *Proceedings of 1986 IEEE Computer Society Workshop of Visual Languages,* Dallas, Texas, Jun.

Stroustrup, B., 1986. *The C++ Programming Language*, Addison-Wesley, Menlo Park, CA.