International Journal of Intelligent Control and Systems

COMMANDING SENSORS AND CONTROLLING INDOOR AUTONOMOUS MOBILE ROBOTS*

MOHAMED DEKHIL and ALEXEI A. EFROS
Department of Computer Science, University of Utah
Salt Lake City, UT 84112, U.S.A

TAREK M. SOBH
Department of Computer Science and Engineering, University of Bridgeport
Bridgeport, CT 06601, U.S.A.

Received (received date)
Revised (revised date)

Any sensory system can be viewed as a passive or "dumb" element which provides raw data. It can also be viewed as an "intelligent" element which returns "analyzed" information. Finally, it can be viewed as a "commanding" element which sends commands to the physical system. Each of these views is used in different situations and for different tasks. Commanding sensors are an extension to the logical sensor approach in which a mapping from events to actions is added to the sensor model. In a previous report, we proposed a sensor-based distributed control scheme for mobile robots along with several simulation results [5]. In this paper, the application of this scheme to control a real mobile robot is presented and the results of several experiments are discussed. A server-client model is used to implement this scheme where the server is a process that carries out the commands to be executed, and each client is a process with a certain task. The logical sensor approach is used to model the sensory system which provides different levels of data representation with tolerance measures and analysis.

Keywords: Robotics, Autonomous Systems, Mobile Robots, Sensing, Control.

1. Introduction

In any closed-loop control system, sensors are used to provide the feedback information that represents the current status of the system and the environmental uncertainties. The sensors used in most control systems are considered to be passive elements that provide raw data to a central controller. The central controller computes the next command based on the required task and the sensor readings. The disadvantage of this scheme is that the central controller may become a bottleneck when the number of sensors increases which may lead to longer response time. In

*This work was supported in part by The Advanced Research Projects Agency under Army Research Office grant number DAAH04-93-G0420, and NSF grant CDA 9024721. All opinions, findings, conclusions or recommendations expressed in this document are these of the authors and do not necessarily reflect the views of the sponsoring agencies.

some applications the required response time may vary according to the required task and the environment status. For example, in an autonomous mobile robot with the task of reaching a destination position while avoiding unknown obstacles, the time to reach to the required position may not be important, however, the response time for avoiding obstacles is critical and requires fast response. Fast response can be achieved by allowing sensors to send commands directly to the physical system when quick attention is required.

In this work, several controllers (clients) are working in parallel, competing for the server. The server selects the command to be executed based on a dynamically configured priority scheme. Each of these clients has a certain task, and may use the sensor readings to achieve its goal. A special client with the task of avoiding obstacles is assigned the highest priority. The clients may also aquire the current state of the system and the command history to update their control strategy.

The logical sensor approach [11,19], which we used to model the sensory system, allows flexible and modular design of the controllers. It also provides several levels of data abstraction and tolerance analysis based on the sensor type and the required task. This approach is used to build high-level requests which may be used by the application programs.

2. Related Work

There has been a tremendous amount of research in the area of sensor-based control, including sensor modeling [8,14,20], multisensor integration [6,9,15,17], and distributed sensing [10,12,13,16].

The idea of *smart sensing* was investigated by several researchers. Yakovleff et al. [23], represented a dual purpose interpretation for sensory information; one for collision avoidance (reactive control), and the other for path planning (navigation). The selection between the two interpretations is dynamic depending on the positions and velocities of the objects in the environment. Budenske and Gini [4], addressed the problem of navigating a robot through an unknown environment, and the need for multiple algorithms and multiple sensing strategies for different situations.

Our proposed control scheme is similar to Brooks' subsumption architecture [2,3]. The controller is decomposed into parallel task achieving behaviors rather than information processing modules. However, Our control scheme is different than Brooks' in the way the parallel tasks are arranged and executed. In the subsumption architecture, layers of control system are built to let the robot operate at increasing levels of competence. These layers are built as concurrent modules that communicate over low-bandwidth channels. Our proposed scheme on the other hand, the control modules are placed at the same level but with different priorities. Different behaviors are achieved by changing the priorities among these modules.

It should be noted that the main thrust of this paper is the control framework itself and the concept of commanding sensors and their use in building higher-level controllers. In other words, we are not proposing any new algorithms for collision

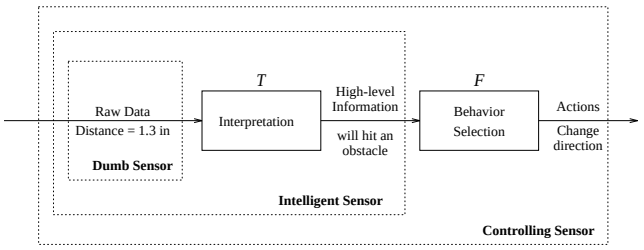

Figure 1: Three levels to view a sensor module.

avoidance or navigation for example, but we can use any available algorithm, such as in [1,7], to build the required clients within this framework.

3. The Proposed Control Scheme

The robot behavior can be described as a function $\mathcal{F}$ that maps a set of events $\mathcal{E}$ to a set of actions $\mathcal{A}$. This can be expressed as:

$$\mathcal{F}\colon \mathcal{E} \longrightarrow \mathcal{A}$$

The task of the robot controller is to realize this behavior. In general we can define the controller as a set of pairs:

$$\{(e_1, a_1), (e_2, a_2), \ldots, (e_n, a_n)\}$$

where $e_i \in \mathcal{E}$, and $a_i \in \mathcal{A}$
The events can be defined as the interpretation of the raw data perceived by the sensors. Let's define the function $\mathcal{T}$ which maps raw data $\mathcal{R}$ to events $\mathcal{E}$:

$$\mathcal{T}\colon \mathcal{R} \longrightarrow \mathcal{E}$$

The functions $\mathcal{T}$ and $\mathcal{F}$ can be closed form equations, lookup tables, or inference engine of an expert system. This depends on the kind of application and the complexity of each transformation.
Figure 1 shows the relationship between the three levels of abstractions for sensory systems in terms of the functions $\mathcal{T}$ and $\mathcal{F}$.
The dumb sensor can be used as a source for the feedback information required by the control system. It can be also used to gather measurements to construct a map for the surrounding environment. The process that uses a dumb sensor as a source of information needs to know the type of that sensor, the format of the data the sensor returns, and the location of the sensor, to be able to interpret the perceived data. The intelligent sensor may be used for monitoring activities. The process that uses an intelligent sensor needs to know only the event domain and maybe the location of the sensor. On the other hand, the commanding sensor is considered to be a "client" process that issues commands to the system.
Several sensors can be grouped together representing a logical sensor. We will assume that each logical sensor is represented as a client process which sends com-

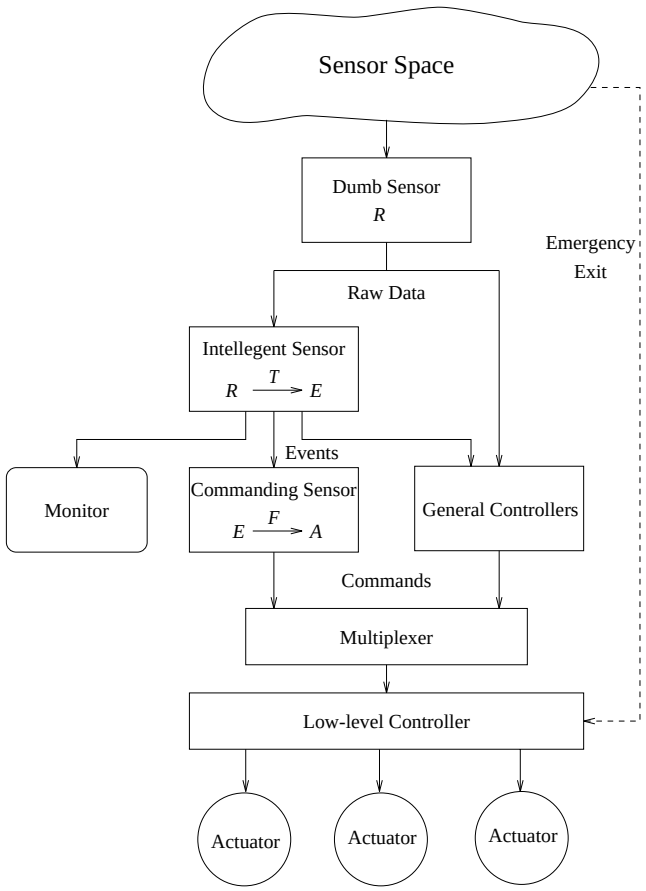

Figure 2: The proposed control scheme.

mands through a channel to a multiplexer (the server process) which decides the command to be executed first. Besides these logical sensors, we might have other processes (general controllers) that send commands to the server process to carry out some global goals. Figure 2 shows a schematic diagram for the proposed control scheme.
Let's call any process that issues commands to the server a *client process*. In this figure, there are three types of clients:

1. Commanding sensors, that are usually used for reaction control and collision avoidance.

2. General Controllers, that carry out a general goal to be achieved (e.g., navigating from one position to another.)

3. Emergency exits, which bypass the multiplexer in case of emergencies (e.g.,

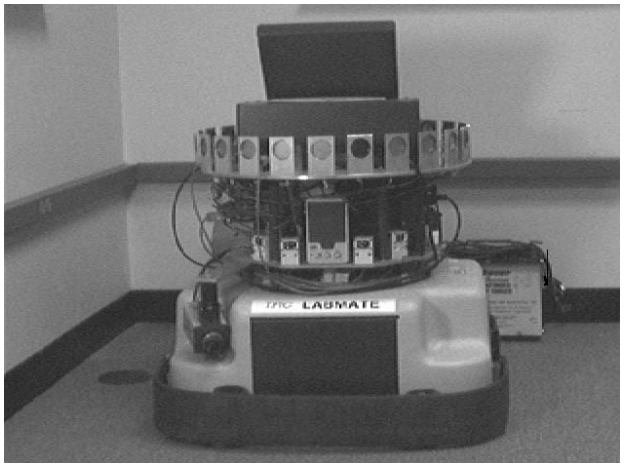

Figure 3: The LABMATE robot with its equipments.

emergency stop when hitting an obstacle.)

The low-level controller, shown in Figure 2, translates the high-level commands into low-level instructions which drive the system's actuators. The low-level controller receives its commands either from the multiplexer or from an emergency exit.

4. Experiments and Results

Several experiments were performed on a mobile robot called "LABMATE" designed by Transitions Research Corporation [21]. We checked the applicability and validity of the proposed control scheme, and the results were very encouraging. Also, a simulator called *XSim* has been developed to perform initial testing and analysis before conducting experiments on the real robot. This simulator displays the robot on the screen and accepts actual LABMATE commands like *go, turn, read-sonars,* etc. In this environment, moving from the simulation to the real robot is simply a matter of compiling the driver program with the LABMATE library rather than the simulation library.

The LABMATE was used for several experiments at the Department of Computer Science, University of Utah. It also was entered in the 1994 AAAI Robot Competition [18]. For that purpose, the LABMATE was equipped with 24 sonar sensors, eight infrared sensors, a camera and a speaker. † Figure 3 shows the LABMATE with its equipment. Simulation results and detailed implementation can be found in [5].

The hardware setup used in these experiments consisted of a PC running Linux to

† The LABMATE preparations, the sensory equipments, and the software and hardware controllers were done by L. Schenkat and L. Veigel at the Department of Computer Science, University of Utah.

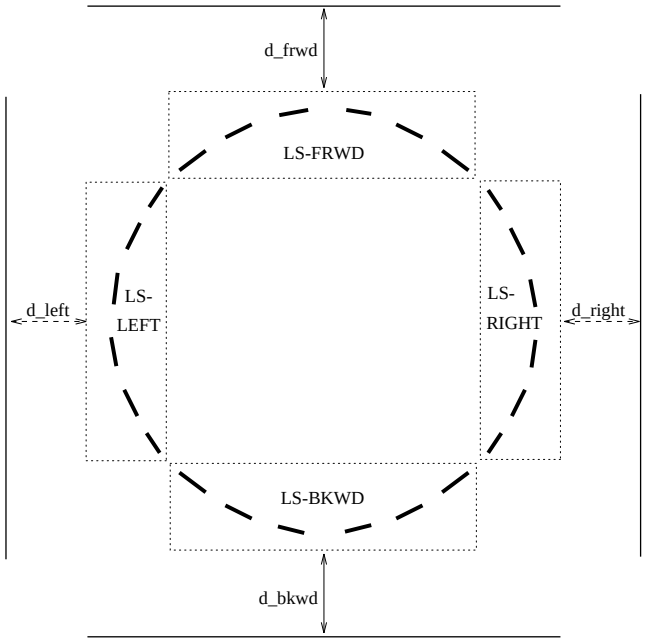

Figure 4: Dividing the sonar sensors into four logical sensors.

provide parallel processing capabilities, and an RS232 serial cable connecting the PC to the LABMATE. No special hardware was used and this setup was inexpensive and easy to use.
The message passing paradigm is used for process communication. This allows processes to be running on different platforms without the need for shared memory. In our implementation, MPI, Message-Passing Interface was used [22].
The priority scheme in our application is set by each client as a number from 1 to 10, with 1 as the highest priority. Normally, 1 is reserved for the collision avoidance client. The server checks for the priority associated with each command, and executes the command with the highest priority while notifying the "losers" which command was executed. If two commands with the same priority arrive at the same time, the server arbitrarily selects one of them and ignores the other. Commands that were not selected are cleared since the state of the robot has been changed after executing the command with the highest priority.

4.1. *Commanding Sensors and Reaction Control*

To simplify our model, the 24 sonar sensors are divided into four logical sensors as shown in Figure 4.

1. **LS-FRWD** consists of the front 6 sensors.
2. **LS-BKWD** consists of the rear 6 sensors.

d_right	d_left	d_frwd	d_bkwd	*FORWARD*	*BACKWARD*
c	c	c	c	*STOP*	*STOP*
c	c	c	f	*GO-BKWD*	—
c	c	f	c	—	*GO-FRWD*
c	c	f	f	—	—
c	f	c	c	*TURN-RIGHT*	*TURN-LEFT*
c	f	c	f	*TURN-RIGHT*	*TURN-LEFT*
c	f	f	c	*TURN-RIGHT*	*TURN-LEFT*
c	f	f	f	*TURN-RIGHT*	*TURN-LEFT*
f	c	c	c	*TURN-LEFT*	*TURN-RIGHT*
f	c	c	f	*TURN-LEFT*	*TURN-RIGHT*
f	c	f	c	*TURN-LEFT*	*TURN-RIGHT*
f	c	f	f	*TURN-LEFT*	*TURN-RIGHT*
f	f	c	c	*TURN-L/R*	*TURN-L/R*
f	f	c	f	*TURN-L/R*	—
f	f	f	c	—	*TURN-L/R*
f	f	f	f	—	—

Table 1: An example of a decision function for reaction control.

3. **LS-RIGHT** consists of the right 6 sensors.

4. **LS-LEFT** consists of the left 6 sensors.

These logical sensors communicate with each other to decide the command to be issued. This makes the job of the multiplexer easier, since it will deal with the four logical sensors as one client. The goal of the reactive control in this experiment is two fold:

1. Avoid obstacles.

2. Keep the robot in the middle of hallways, specially when moving through narrow corridors or going through a door.

We define two abstract values: *close* (c) and *far* (f). These two values represent the distance between the robot and the closest object at any of the four sides. The range for c and f are usually user defined values. The command to be issued as a reaction control depends on the current state of the system and the distance value at each side. There are several ways to define a command function $\mathcal{F}$ to achieve the required goal. The assumption here is that there is always enough space for the robot to rotate left or right, therefore there is no need to define any reaction control when the robot is rotating. One such function is shown in Table 1.
In this table, *TURN-L/R* means the command can be either *TURN-LEFT* or *TURN-RIGHT*, and a dash "—" means no command is issued. Notice that, in case d_left and d_right have different values, the values for d_frwd and d_bkwd are not important. This is because we need to balance the distance to the left and to

the right of the robot, and if, for example, the distance in front (d_frwd) is c, and the robot state is *FORWARD*, then moving to the left (or to the right) will serve both avoiding the object in front, and balancing the distance on both sides. In the first case of the table, when the distance is c on all sides, the robot will not be able to move anywhere, and the sensor readings will not change. This will result in a deadlock which requires external help by moving at least one of the obstacles for the robot to be able to move.

4.2. *Experiments*

The following is a description of three of the experiments that were performed on both the simulator and the real robot. The results of the real robot were qualitatively very similar to the simulation results except for some deviations due to the noise in the real sonar sensors.

Experiment (1)

This was the first experiment performed to demonstrate the applicability of this control scheme. In this experiment, two clients were running simultaneously; the collision avoidance client implemented as a commanding sensor, and a simple navigator which always commands the robot to move forward. The collision avoidance has priority 1, which is the highest priority, and the navigation client has priority 9. The following shows part of the output printed during this experiment which shows the commands that has been executed by the server.

```
Collision Avoidance: client #1.
Simple Navigation: client #2.
Server Starts as process #0.

* Accepted RESET from 1 *
- Rejected RESET from 2 *
* Accepted GO-FRWD from 2 *
* Accepted GO-FRWD from 2 *
* Accepted GO-FRWD from 2 *
* Accepted GO-FRWD from 2 *
* Accepted TURN-LEFT from 1 *
- Rejected GO-FRWD from 2 -
* Accepted TURN-LEFT from 1 *
- Rejected GO-FRWD from 2 -
* Accepted GO-FRWD from 2 *
* Accepted GO-FRWD from 2 *
         . . .
```

Figure 5 shows the trajectory of the robot using the simulation program and the trajectory of the real robot with the sonar readings at each step. Notice that there is some deviations in the real trajectory from the simulated one due to the noise in the sonar readings as shown in Figure 5 environment.

Experiment (2)

In the second experiment, we added another goal-directed client which tries to move the robot to a certain goal location. This client has priority 5 which is higher than the simple navigator process. This new client sends commands to the server to update the direction of the robot such that it moves towards the goal location. In this experiment, the initial and the final points were chosen such that there is a wall between them. Figure 6 shows the trajectory for the simulated and the real robot. Notice that at several points, the collision avoidance client took over and moved the robot away from the wall, then the new client updates the direction towards the goal point.

Experiment (3)

In the third experiment, we replaced the goal-directed client with a door-finding client which is another commanding sensor. This new client tries to find open doors and direct the robot to go through these doors. Finding doors using sonar sensor is very hard and problematic, and there is a lot of research in this area. For this experiment we used a very crude algorithm and a simple hallway structure just to demonstrate the capabilities of the proposed control scheme. Figure 7 shows the trajectory for the simulated and the real robot while moving in a hallway environment with two open doors at different locations.

5. Conclusion and Future Work

In this paper, a distributed sensor-based control scheme was proposed. In this scheme, each sensor can be viewed with three different levels of abstraction: *dumb sensors* which provide raw data, *intelligent sensors* which provide high level information in the form of events, and finally, *commanding sensors* which can issue commands representing a reaction behavior for the system. Commands can be issued by different processes called *clients*. Each client may issue commands at any time, and a multiplexer (the server) selects the command to be executed. A priority scheme has to be defined as a basis for selection. Examples for applying this control scheme to a mobile robot were described. We believe that this scheme provides for more flexible and robust control systems, and allows more modular design for the whole control system. It also provides fast response for reaction behavior which is an essential requirement in real-time systems.

Currently, we are working on designing and implementing higher level controllers that use the lower-level clients to perform more complicated tasks. This includes selecting the appropriate clients and dynamically changing their priorities to realize the required behavior. Also, some aspects of tolerance analysis will be incorporated in the proposed scheme to provide quantitative measures for the accuracy of the location of measured points. It also serves as the basis for devising sensing strategies to enhance the measured data for localization and map construction.

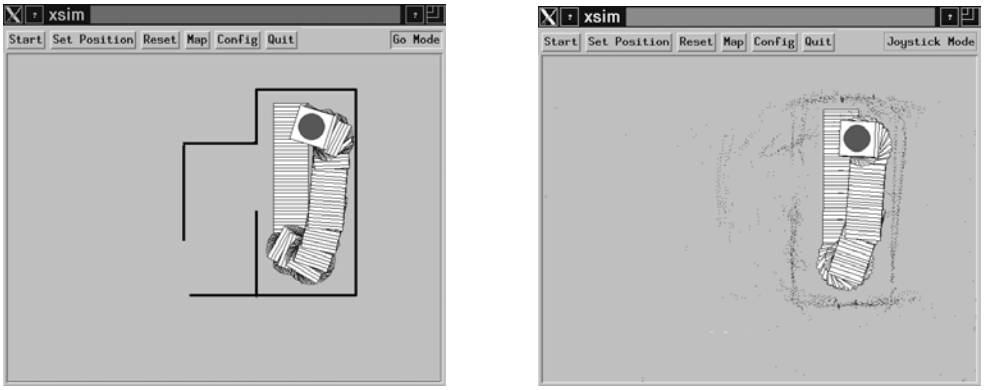

Figure 5: The trajectories for the simulated and the real robot for experiment (1).

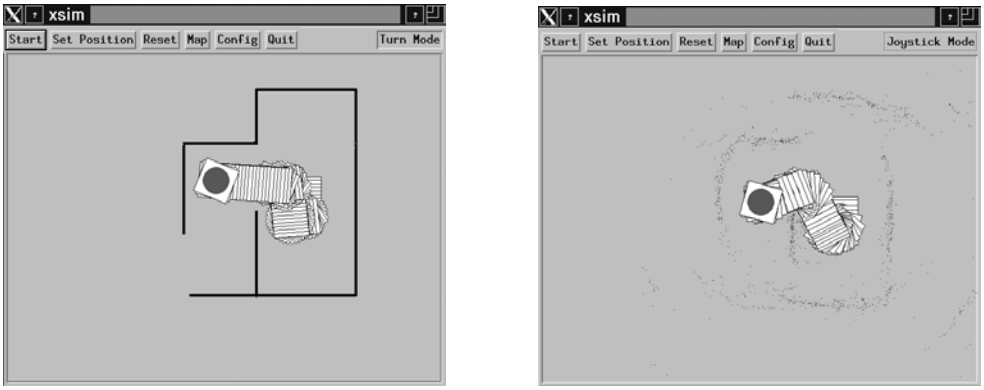

Figure 6: The trajectories for the simulated and the real robot for experiment (2).

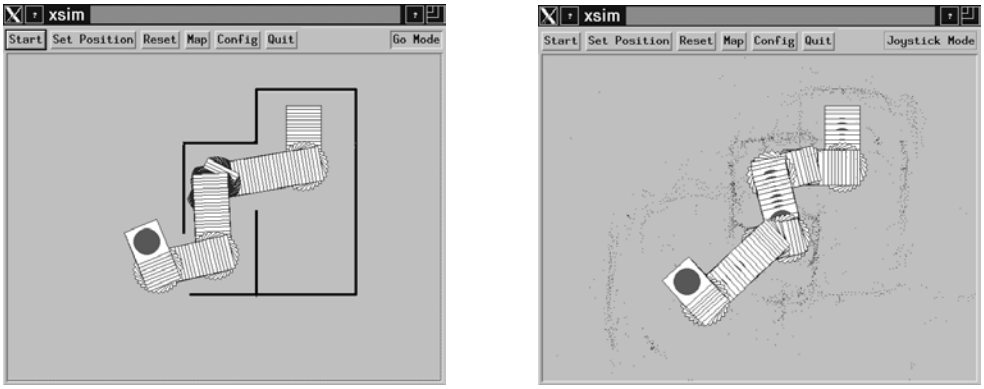

Figure 7: The trajectories for the simulated and the real robot for experiment (3).

References

1. AHLUWALIA, R. S., AND HSU, E. Y. Sensor-based obstruction avoidance technique for a mobile robot. *Journal of Robotic Systems 1*, 4 (Winter 1984), pp. 331–350.
2. BROOKS, R. A. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation RA-2*, 1 (March 1986), pp. 14–23.
3. BROOKS, R. A. A hardware retargetable distributed layered architecture for mobile robot control. In *IEEE Int. Conf. Robotics and Automation* (1987), pp. 106–110.
4. BUDENSKE, J., AND GINI, M. Why is it difficult for a robot to pass through a doorway using altrasonic sensors? In *IEEE Int. Conf. Robotics and Automation* (May 1994), pp. 3124–3129.
5. DEKHIL, M., GOPALAKRISHNAN, G., AND HENDERSON, T. C. Modeling and verification of distributed control scheme for mobile robots. Tech. Rep. UUCS-95-004, University of Utah, April 1995.
6. DURRANT-WHYTE, H. F. *Integration, coordination and control of multisensor robot systems.* Kluwer Academic Publishers, 1988.
7. GOURLEY, C., AND TRIVEDI, M. Sensor-based obstacle avoidance and mapping for fast mobile robots. In *IEEE Int. Conf. Robotics and Automation* (1994).
8. GROEN, F. C. A., ANTONISSEN, P. P. J., AND WELLER, G. A. Model based robot vision. In *IEEE Instrumentation and Measurment Technology Conference* (1993), pp. 584–588.
9. HAGAR, G. D., AND MINTZ, M. Task-directed multisensor fusion. In *IEEE Int. Conf. Robotics and Automation* (1989).
10. HENDERSON, T. C., HANSEN, C., AND BHANU, B. The specification of distributed sensing and control. *Journal of Robotic Systems* (Mar. 1985), pp. 387–396.
11. HENDERSON, T. C., AND SHILCRAT, E. Logical sensor systems. *Journal of Robotic Systems* (Mar. 1984), pp. 169–193.
12. IYENGAR, S. S., AND PRASAD, L. A general computational framework for distributed sensing and falt-tolerant sensor integration. *IEEE Trans. Systems Man and Cybernetics* (May 1994).
13. IYENGAR, S. S., SHARMA, M. B., AND KASHYAP, R. L. Information routing and reliability issues in distributed sensor networks. *IEEE Trans. Signal Processing* (Dec. 1992), pp. 3012–3021.
14. JOSHI, R., AND SANDERSON, A. C. Model-based multisensor data fusion: a minimal representation approach. In *IEEE Int. Conf. Robotics and Automation* (May 1994).
15. LUO, R. C., AND KAY, M. G. *Multisensor integration and fusion for intelligent machines and systems.* Ablex Publishing Corporation, 1995.
16. NADIG, D., IYENGAR, S. S., AND JAYASIMHA, D. N. New architecture for distributed sensor integration. In *IEEE SOUTHEASTCON Proceedings* (1993).
17. OKAFOR, A. C., ERTEKIN, Y. M., AND ARORA, S. Intelligent machining, monitoring, and diagnostic system for quality assurance based on multisensor integration. In *NSF Design and Manufacturing Systems Conference* (Jan. 1993).
18. SCHENKAT, L., VEIGEL, L., AND HENDERSON, T. C. Egor: Design, development, implementation – an entry in the 1994 AAAI robot competition. Tech. Rep. UUCS-94-034, University of Utah, Dec. 1994.
19. SHILCRAT, E. D. Logical sensor systems. Master's thesis, University of Utah, August 1984.
20. SPIEWAK, S. A., CHUNG, Y. L., AND HUANG, M. S. Analytical modeling of sensors. In *NSF Design and Manufacturing Systems Conference* (Jan. 1993).

21. TRC Transition Research Corporation. *LABMATE user manual, version 5.21L-f.*, 1991.
22. University of Tennessee, Knoxville. *MPI: a message-passing interface standard.*, May 1994.
23. Yakovleff, A., Nguyen, X. T., Bouzerdoum, A., Moini, A., Bogner, R. E., and Eshraghian, K. Dual-purpose interpretation of sensory information. In *IEEE Int. Conf. Robotics and Automation* (1994).