Proceedings of the 1998 IEEE
International Conference on Robotics & Automation
Leuven, Belgium • May 1998

Instrumented Logical Sensor Systems - Practice

Mohamed Dekhil and Thomas C. Henderson
Department of Computer Science
University of Utah
Salt Lake City, UT 84112

Abstract

In previous work, we introduced the notion of Instrumented Logical Sensor Systems (ILSS) that are derived from a modeling and design methodology [3, 4]. The instrumented sensor approach is based on a sensori-computational model which defines the components of the sensor system in terms of their functionality, accuracy, robustness and efficiency. This approach provides a uniform specification language to define sensor systems as a composition of smaller, predefined components. From a software engineering standpoint, this addresses the issues of modularity, reusability, and reliability for building complex multisensor systems.

In this paper, we demonstrate the practicality of this approach and discuss several design and implementation aspects in the context of mobile robot applications.

1 Introduction

Building a sensor system for a certain application is a process that includes the analysis of the system requirements, a model of the environment, the determination of system behavior under different conditions, and the selection of suitable sensors. The next step in building the sensor system is to assemble the hardware components and to develop the necessary software modules for data fusion and interpretation. Finally, the system is tested and the performance is analyzed. Once the system is built, it is difficult to monitor the different components of the system for the purpose of testing, debugging and analysis. It is also hard to evaluate the system in terms of time complexity, space complexity, robustness, and efficiency, since this requires quantitative measures for each of these measures.

This work was supported in part by NSF grant CDA 9024721 and a gift from Hewlett Packard Corporation.

In addition, designing and implementing real-time systems are becoming increasingly complex because of many added features such as fancy graphical users interfaces (GUIs), visualization capabilities and the use of many sensors of different types. Therefore, many software engineering issues such as reusability and the use of COTS (Commercial Off-The Shelf) components [18], reliability [12, 13], and embedded testing [20] are now getting more attention from system developers.

In a previous paper, we proposed to use formal semantics to define performance characteristics of sensor systems [3]. Then, we presented a theoretical framework for modeling and designing sensor systems based on a formal semantics in terms of a virtual sensing machine [4]. This framework defines an explicit tie between the specification, robustness and efficiency of the sensor system by defining several quantitative measures that characterize certain aspects of the system's behavior. In this paper, we describe our approach which provides static analysis (e.g., time/space complexity, error analysis) and dynamic handles that assist in monitoring and debugging the system.

2 Related Work

Each sensor type has different characteristics and functional description. Therefore it is desirable to find a general model for these different types that allows modeling sensor systems that are independent of the physical sensors used, and enables studying the performance and robustness of such systems. There have been many attempts to provide "the" general model along with its mathematical basis and description. Some of these modeling techniques concern error analysis and fault tolerance of multisensor systems [2, 5, 11, 16, 17]. Other techniques are model-based and require a priori knowledge of the scanned object and its environment [8, 9]. These techniques help fit data to a model, but do not provide the means

0-7803-4300-x-5/98 $10.00 © 1998 IEEE

to compare alternatives. Task-directed sensing is another approach to devise sensing strategies [1, 10], but again, it does not provide measures to evaluate the sensor system in terms of robustness and efficiency.

Another approach to modeling sensor systems is to define sensori-computational systems associated with each sensor to allow design, comparison, transformation, and reduction of any sensory system [7]. In this approach the concept of information invariants is used to define some measure of information complexity. This approach provides a very strong computational theory which allows comparing sensor systems, reducing one sensor system to another, and measuring the information complexity required to perform a certain task. However, as stated by Donald, the measures for information complexity are fundamentally different from performance measures. Also, this approach does not permit one to judge which system is "simpler," "better," or "cheaper."

In most application, performance analysis is essential to evaluate the system and compare alternative solutions. Measuring the performance of any system requires identifying a set of metrics that capture the desired characteristics of the system. In the robotics field, several research efforts has been directed towards defining such metrics and evaluating the performance of new control algorithms.

Lee and Resse proposed a quantitative evaluation approach for navigation and planning strategies for mobile robots [14]. They conducted an experimental investigation into the robots' exploration capabilities using a novel metric that predicts the effectiveness of the robot in executing a set of tasks using a map that is built using a Polaroid ultrasonic range sensor. This approach matches our view of performance evaluation of sensor systems in terms of providing a set of metrics and conducting experimental evaluation of the system to capture the dynamics and variations in the system and its environment.

3 The ILSS Approach

The *Instrumented Logical Sensor System* (ILSS) approach represents our methodology for incorporating design tools and allows static and dynamic performance analysis, on-line monitoring, and embedded testing.

In this framework, a sensor system is composed of several ILSS modules connected together in a certain structure. We defined operations for composing ILSS modules, and defined the semantics of these operations in terms of the performance parameters [4]. The semantics of these construction operations is defined as a set of functions that propagate the required performance measures. Several techniques can be used for propagation. Best case analysis, worst case analysis, average, etc. Selecting among these depends on the application, hence it should be user defined.

Our main objective is to develop a modeling scheme that incorporates tools for analysis, debugging, and monitoring sensor systems as an integral part of the design, with emphasis on mobile robot control applications.

3.1 Implementing ILS Modules

An object-oriented approach is used to develop the ILS modules, where each module is implemented as an object that possesses several basic features that are common to all ILS modules, plus some additional features that are specific to each ILS type.

We use these modules to build complex sensor systems in a hierarchical structure. Each component can run as a separate process or several of them can run as one process depending on the application requirements and the degree of concurrency needed. Monitors, taps, and embedded tests are implemented as sub-processes generated from the main ILSS process. These components communicate through the COV and the Command lines using shared-memory structures. This shared memory architecture was developed to design and implement distributed control systems for mobile robots (see [4, 6] for more detail.) Figure 1 shows the structure of one ILSS component with its different modules and communication lines.

A complex sensor system may have tens or hundreds of these components connected together in a certain structure. This structure is kept in a small database that contains all the necessary information about each components and the way they are connected. This adds more flexibility to the system and allows for rapid construction and modification of the system components and its parameters. Figure 2 shows the database schema used for this purpose.

4 Experiment: Adaptive Wall Following

Several experiments has been conducted to demonstrate the usability of the proposed framework in modeling and designing sensor systems [4, 3]. In the following experiment we implement a simple wall-following

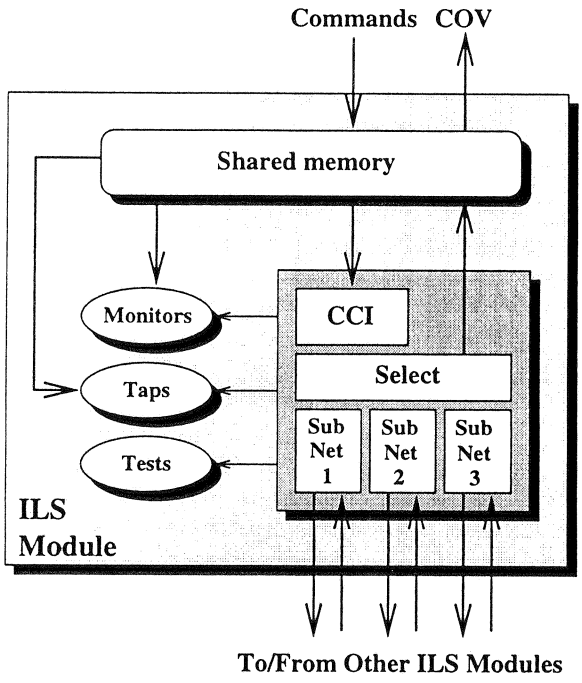

Figure 1: ILSS component structure.

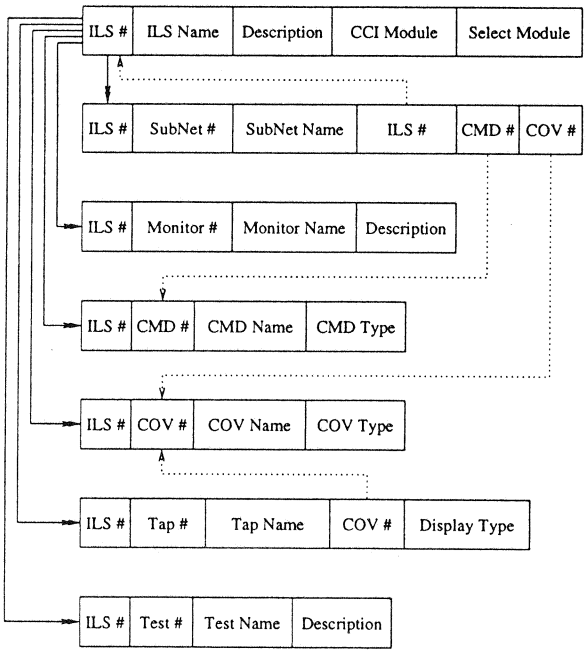

Figure 2: Database schema for ILSS Structures.

algorithm using two alternatives; sonar sensors and a camera. The sonars and the camera are mounted on a LABMATE mobile robot designed by Transitions Research Corporation. The LABMATE was used for several experiments in the Department of Computer Science at the University of Utah. It was also entered in the 1994 and 1996 AAAI Robot Competition [19] and it won sixth and third place, respectively. For that purpose, the LABMATE was equipped with 24 sonar sensors, eight infrared sensors, a camera and a speaker.[1]

Figure 3 shows the ILSS structure used for this experiment along with the robot controller and the follow-wall components. This experiment illustrates the usefulness of the design tools provided by the ILSS architecture such as taps, monitors, and embedded tests. The idea of using two different (and independent) ILSs is to increase the reliability and the robustness of the system. For example, if the sonar sensors are not working probably due to audio interference or damage, the system detects that through a certain monitor, and automatically switches to using the camera. Similarly, if the system detects any malfunction with the camera (e.g., lights off, occlusion, etc.) it switches to the sonars. (Also, see [15].)

Using various reliability measures, the system can determine the reliability of the overall system at anytime given the reliability of each components and its current status.

4.1 Using Sonar Sensors

Two sonar sensors located on one side of the robot are used to determine the the orientation of the wall relative to the robot. We call it *ILS_Sonar_Pose*. It gets the sonar values from the *ILS_Sonar* component and generate one of three values: 1, 0, or -1 to represent if the robot should turn right, no turn, or left. The *ILS_Pose* component selects the sonar first since it is faster than the camera. If the *ILS_Sonar_Monitor* detects failure it reports that and the *ILS_Camera_Pose* is selected. In this case failure is detected if one of the sonars generates an out-of-range value.

4.2 Using the Camera

The camera is located on the robot and pointed towards the side where the two sonars are. The ori-

[1]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.

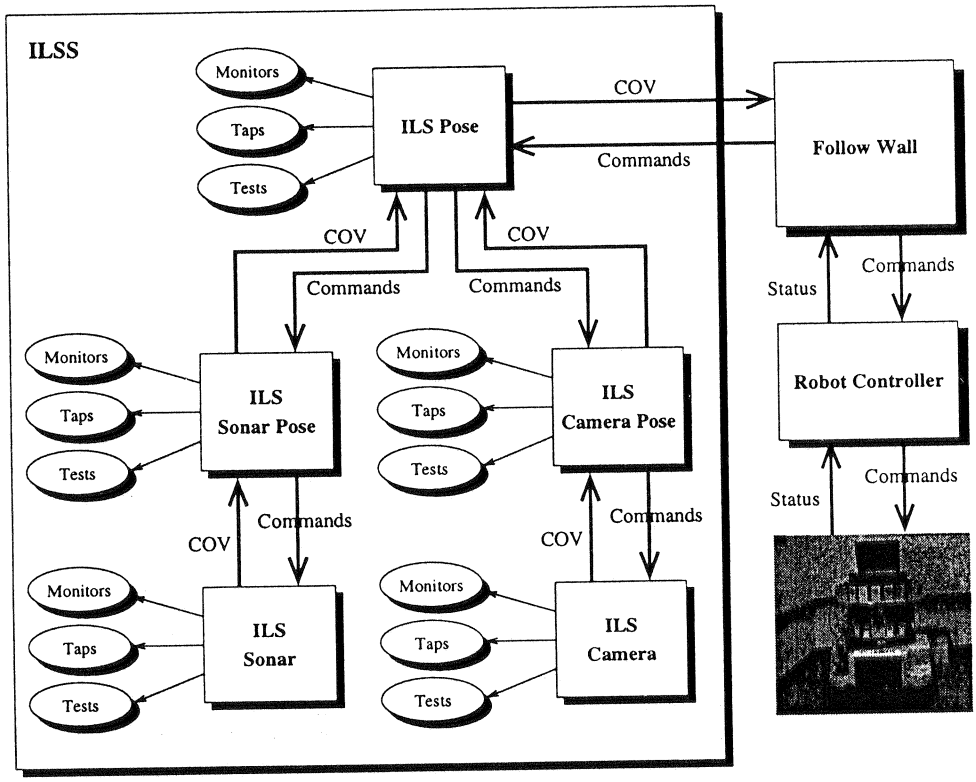

Figure 3: The wall-following system using sonars and a camera.

entation of the wall is determined using a horizontal dark line on the wall (we used electrical tape for that purpose.) The idea is to find the location of both ends of the line in the image. by location here we mean the image row number. By comparing the row numbers for both ends we can determine the way the robot should turn to be parallel to the wall. the *ILS_Camera_Pose* is used for that purpose. The *ILS_Camera_Monitor* checks for insufficient light or occlusion of the line.

4.3 Results

We started the experiment with both sensors working. The select function chooses the sonar first then the camera. Figures 4 and 5 show two scenarios with two different initial orientations of the robot relative to the wall. In both scenarios we induced malfunction to the sonar (by covering it) to test the monitoring and debugging capabilities of the system. The system detected this malfunction and automatically switched to the camera. We also induced a malfunction to the camera (by turning off the lights) and the system detected that as well and started performing the appropriate embedded tests for both sensors to pinpoint the problem.

The following script shows parts of the system output while running the experiment.

```
-- Start Initializations
-- Initialize ILS_Pose
-- In ILS_Sonar_Init
-- In ILS_Camera_Init
-- In Camera_Init
-- Start The Robot hardware

-- Starting main loop

TAP -- ILS_Sonar:
        Direction = 1, time = 0.236589 sec.
TAP -- ILS_Pose:
        Direction = 1, time = 0.243303 sec.
TAP -- ILS_Sonar:
        Direction = 1, time = 0.246920 sec.
TAP -- ILS_Pose:
        Direction = 1, time = 0.248939 sec.
  . . .

TAP -- ILS_Sonar:
        Direction = -1, time = 0.228644 sec.
Monitor -- ILS_Sonar:
        Sonar values out of range (541, 171, 174)
```

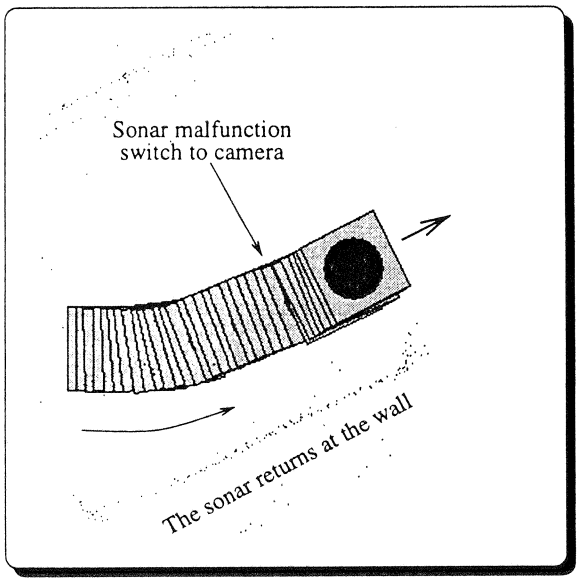

Figure 4: The first test run.

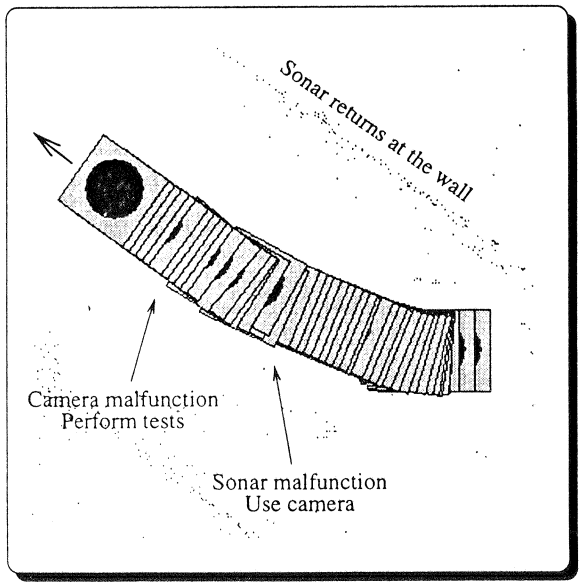

Figure 5: The second test run.

```
!!!!!! Switching to ILS_Camera !!!!!!

TAP -- ILS_Camera:
        Direction = 1, time = 0.004258 sec.
TAP -- ILS_Pose:
        Direction = 1, time = 0.246355 sec.
. . .

TAP -- ILS_Camera:
        Direction = 0, time = 0.003417 sec.
Monitor -- ILS_Sonar:
        Image too dark! Lights might be off!

-- Start embedded testing

-- In ILS_Pose_Test
TEST -- ILS_Sonar:
        Place the robot parallel to the wall at
        about 1 meter distance
        and press any key when ready ...

         --> Sonar 5 is out of range (137)

!!! ILS_Sonar_Test Failed !!!

TEST -- ILS_Camera:
        Place the robot parallel to the wall at
        about 1 meter distance
        and press any key when ready ...

-- Camera test is Ok!
```

5 Conclusion

In this paper we presented a modeling and design methodology that facilitates interactive, on-line monitoring for different components of the sensor system. It also provides debugging tools and analysis measures for the sensor system. The instrumented sensor approach can be viewed as an abstract sensing machine which defines the semantics of sensor systems. This provides a strong computational and operational engine that can be used to define and propagate several quantitative measures to evaluate and compare design alternatives. The implementation of this framework for several mobile robot applications were conducted and the wall-following experiment was presented along with a discussion of the results.

Currently, we are working on building an ILSS library with several design tools which will assist in rapid prototyping of sensor systems and will provide an invaluable design tools for monitoring, analyzing

and debugging robotic sensor systems.

Acknowledgment

We would like to thank Professor Robert Kessler and Professor Gary Lindstrom for their helpful discussions and suggestions.

References

[1] BRIGGS, A., AND DONALD, B. Automatic sensor configuration for task-directed planning. In *IEEE Int. Conf. Robotics and Automation* (May 1994), pp. 1345–1350.

[2] BROOKS, R. R., AND IYENGAR, S. Averaging algorithm for multi-dimensional redundant sensor arrays: resolving sensor inconsistencies. Tech. rep., Louisiana State University, 1993.

[3] DEKHIL, M., AND HENDERSON, T. C. Instrumented sensor systems. In *IEEE International Conference on Multisensor Fusion and Integration (MFI 96), Washington D.C.* (December 1996), pp. 193–200.

[4] DEKHIL, M., AND HENDERSON, T. C. Instrumented sensor system architecture. *Int. J. Robotics Research* (April 1998).

[5] DEKHIL, M., AND SOBH, T. M. Embedded tolerance analysis for sonar sensors. In *Invited paper to the special session of the 1997 Measurement Science Conference,* Measuring Sensed Data for Robotics and Automation, *Pasadena, California* (January 1997).

[6] DEKHIL, M., SOBH, T. M., AND EFROS, A. A. Sensor-based distributed control scheme for mobile robots. In *IEEE International Symposium on Intelligent Control (ISIC 95), Monterey, California* (August 1995).

[7] DONALD, B. R. On information invariants in robotics. *Artificial Intelligence 72* (1995), pp. 217–304.

[8] DURRANT-WHYTE, H. F. *Integration, coordination and control of multisensor robot systems.* Kluwer Academic Publishers, Boston, 1988.

[9] GROEN, F. C. A., ANTONISSEN, P. P. J., AND WELLER, G. A. Model based robot vision. In *IEEE Instrumentation and Measurement Technology Conference* (1993), pp. 584–588.

[10] HAGER, G. D., AND MINTZ, M. Computational methods for task-directed sensor data fusion and sensor planning. *Int. J. Robotics Research 10,* 4 (August 1991), pp. 285–313.

[11] IYENGAR, S. S., AND PRASAD, L. A general computational framework for distributed sensing and falt-tolerant sensor integration. *IEEE Trans. Systems Man and Cybernetics 25,* 4 (April 1995), 643–650.

[12] KAPUR, R., WILLIAMS, T. W., AND MILLER, E. F. System testing and reliability techniques for avoiding failure. *IEEE Computer* (November 1996), pp.28–30.

[13] KIM, K. H., AND SUBBARAMAN, C. Fault-tolerant real-time objects. *Communications of the ACM 40,* 1 (January 1997), pp.75–82.

[14] LEE, D. C., AND RECCE, M. Quantitative evaluation of the exploration strategies of a mobile robot. *IJRR 16,* 4 (Aug. 1997), pp. 413–447.

[15] MURPHY, R. R., AND HERSHBERGER, D. Classifying and recovering from sensing failures in autonomous mobile robots. In *Proceedings of the AAAI-96* (Aug. 1996), pp. 922–929.

[16] NADIG, D., IYENGAR, S. S., AND JAYASIMHA, D. N. New architecture for distributed sensor integration. In *IEEE SOUTHEASTCON Proceedings* (1993).

[17] PRASAD, L., IYENGAR, S. S., RAO, R. L., AND KASHYAP, R. L. Fault-tolerence sensor integration using multiresolution decomposition. *Journal of The American Physical Society 49,* 4 (April 1994), pp. 3452–3461.

[18] PROFETA, J. A. Safety-critical systems built with COTS. *IEEE Computer 29,* 11 (November 1996), pp.54–60.

[19] 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.

[20] WELLER, G. A., GROEN, F. C. A., AND HERTZBERGER, L. O. A sensor processing model incorporating error detection and recoverry. In *Traditional and non-traditional robotic sensors.* Edited by T. C. Henderson. (1990), Springer-Verlag, Berlin, pp. 351–363.