MIT Libraries Document Services/Interlibrary Loan

ILLiad TN: 349320

ILL Number: 110800255

Yes No Cond

Borrower: UBM

Request Date: 11/5/2013 11:54:57 AM

Lending String: *MYG,YRM

Journal Title: Robot control 1997 ; SYROCO '97 ; a proceedings volume from the 5th IFAC symposium, Nantes, France, 3-5 September 1997 /

Vol.: 2 **Issue:**
Month/Year: 1998
Pages: 459-464

Author: Sobh, T

Title: Robust sensing for mobile robot control

Imprint: Oxford ;, Tarrytown, N.Y. ; Published fo

PLETED
NOV 0 2013
MENT SERVICES

Delivery Method: Email

Call #: TS191.8.I34 1997 v.2
Location: b

Maxcost: 40.00IFM

Patron: Sobh, Tarek for Institutional Repository

Library Address:
University of Bridgeport
Library, ILL
126 Park Avenue
Bridgeport, CT 06601

Request Type: Article

Document Type: Article

OCLC#: 38494450

Notes: Billing Notes; Please respond COND if fee. We do not charge you if you do not charge us.

ddulepsk@bridgeport.edu

Odyssey:

Ariel:

Email Address: ddulepsk@bridgeport.edu

US Copyright Notice

The copyright law of the United States (Title 17, United States Code) governs the making of reproductions of copyrighted material.
Under certain conditions specified in the law, libraries are authorized to furnish a reproduction. One of these specified conditions is that the reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a user makes a request for, or later uses, a reproduction for purposes in excess of "fair use," that user may be liable for copyright infringement.
This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of Copyright Law.

Copyright © IFAC Robot Control, Nantes, France, 1997

Robust Sensing for Mobile Robot Control

Tarek M. Sobh, Mohamed Dekhil, and Alyosha A. Efros

Department of Computer Science and Engineering
School of Science, Engineering and Technology
University of Bridgeport
Bridgeport, CT 06601, USA

Abstract: In this work we present a control strategy under sensing uncertainties for mobile robot navigation. The sensing system is represented using three levels of abstraction. Sensing uncertainties are embedded into the model to perform tolerance analysis for data recovery and timing. A server-client model is implemented where the server executes commands and clients run in parallel.

Keywords: Mobile Robots, Distributed Control, Sensing.

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 main component in such systems is the transformation of sensor outputs to the decision space, then the computation of the error signals and the joint-level commands (see Figure 1). Fast response can be achieved by allowing sensors to send commands directly to the physical system when quick attention is required. This is analogous to human reactions to some events. 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 can use the sensor readings to achieve its goal. A special client with the task of avoiding obstacles is assigned the highest priority.

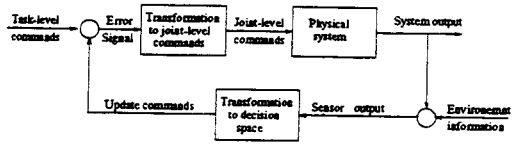

Figure 1: Closed loop control system.

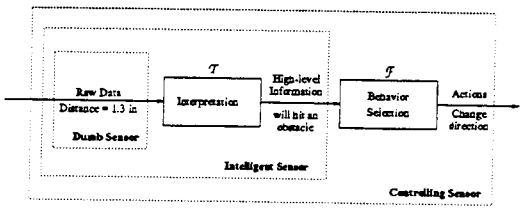

Figure 2: Three levels to view a sensor module.

2 The Proposed Control Scheme

2.1 Abstract Sensor Model

One can view the sensory system using three different levels of abstractions (see Figure 2.)

1. **Dumb sensor:** which returns raw data without any interpretation. For example, a range sensor might return a real number representing the distance to an object in inches, and a camera may return an integer matrix representing the intensity levels of each pixel in the image.

2. **Intelligent sensor:** which interprets the raw data into an even. For example, the sensor might return something like "will hit an object," or "a can of Coke is found."

3. **Controlling sensor:** which can issue commands based on the received events. for example, the sensor may issue the command "stop" or "turn left" when it finds an obstacle ahead.

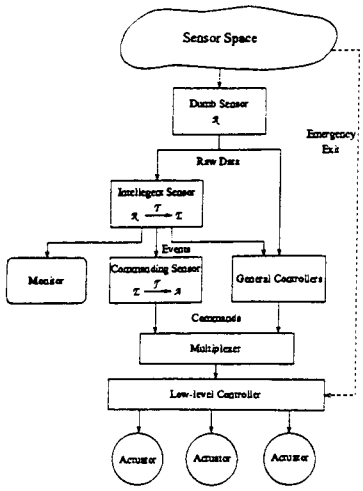

Figure 3: The proposed control scheme.

2.2 A Distributed Control Architecture

Several sensors can be grouped together representing a logical sensor [3, 8]. We will assume that each logical sensor is represented as a client process which sends commands through a chanel to a multiplexer (the server process) which decides the command to be executed first. Besides these logical sensors, there might have other processes (general controllers) that send commands to the server process to carry out some global goals. Figure 3 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., emergency stop when hitting an obstacle.)

2.3 Time vs. Accuracy

The most important criteria in any sensory system are *time* and *accuracy.* Time is the time elapsed between issuing a read request to the logical sensor and the reply to that request. This time depends on the physical aspects of the sensory system, and on the sensing strategy implemented in the logical sensor. Tolerance is defined in this scheme as the region in which the measurement resides.

The following are some variables that will be used in the tolerance analysis for our experiment.

- v_s : sound velocity.
- y_{max} : maximum distance in our indoor environment.
- y_{min} : minimum distance in our indoor environment.
- t_m : the maximum time to get a measurement by the physical sonar sensor.

$$t_m = 2y_{max}/v_s$$

- v_r : the linear velocity of the robot in meter/sec.
- ω_r : the angular velocity of the robot in rad/sec.
- t_d : decision time; the time to decide the next action based on the current reading.

  In most cases, one cannot satisfy both requirement at the same time. Since the physical sensor has its accuracy limitations, therefore, we might need to get several readings regarding the same measured point to increase the accuracy. This of course with increase the time of measurement. In case of multisensor system, the accuracy can be increased by considering the readings from more than one sensor. In such cases, one should consider the time of the data fusion algorithms used.

3 Tolerance Analysis for Sonar Sensors

Sonar sensors have been widely used in several robotic applications. This is due to their low cost and reasonable reliability. However, sonar sensors, like most ultrasonic sensors, have several drawbacks. One problem is that the data measurements depends on the speed of sound, which vary according to the atmospheric conditions such as temperature and humidity. This usually results in inaccurate and inconsistent readings. Another problem is that the ultrasonic echoes might cause the sensor to measure totally incorrect values.

The use of ultrasonic sensors for mobile robots has been investigated by a lot of researcher [1, 4, 5, 6]. The main goal is to increase the accuracy and reliability of these sensors, and to filter the noise and echoes to get more consistent data.

In this analysis, we will ignore measurement noise, and errors due to the interference between the sensors. Also, we will use a simplified beam pattern for the sonar sensors which is a conic shape centered at the center of the sensor. Our goal here is to be able to determine the location of the measured point within a certain tolerance. Also, we would like to locate edges and door ways within a reasonable tolerance. First, the case of using one sensor will be considered, then the use of multiple sensors to get more accurate measurements will be discussed.

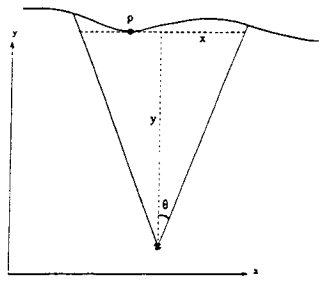

Figure 4: The simplified sonar model.

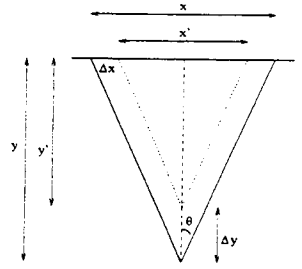

Figure 5: Translation in the y direction.

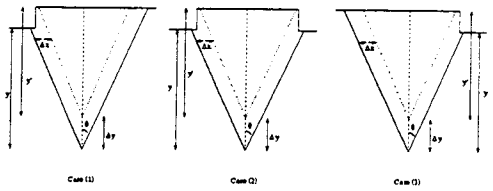

Figure 6: Different situations when moving in the y direction.

3.1 Using One Sensor

Figure 4 shows the simplified beam pattern of a sonar sensor. We assume that the sensor will return the distance y of the closest point p from the center of the sensor within a tolerance region $2x$, where $x = y \tan\theta$.

It is clear that the tolerance area depends on the angle θ and the distance y. However, the angle is fixed for most sonar sensors (It may vary according to the operating frequency.) In our experiment for example, $\theta = 11$. Also, there is physical limitations on the minimum distance y_{min} which means there is an upper limit on the accuracy that we can get with the sensor. The upper limit is:

$$x_{min} = y_{min} \tan\theta$$

There are several ways to minimize the tolerance region and to detect the existence of edges within this region. One way is to move in the y direction towards the measured point. Another way is to move small movements in the x direction, and a third technique is to rotate with small angle ϕ. In each case, the readings are combined to get smaller tolerance region. Now, let's discuss each of these techniques in more details.

Translation in the y direction

Moving Δy in the y direction towards the measured point, the tolerance region is decreased by:

$$\Delta x = \Delta y \tan\theta$$

This is shown in Figure 5 where y is the initial distance and x is the initial tolerance region, and y' is the new distance and x' is the new tolerance region. The time to make this movement t_y is equal to:

$$t_y = \Delta y / v_r$$

and total time to make this reduction in the tolerance region is equal to

$$t_y + t_d$$

If the new distance y' is different than $y - \Delta y$, this means that we encountered an edge or a door way. Figure 6 shows different situations in which this may occur.

In this case, the edge can be located with tolerance $2\Delta x$ since the edge ma be at either sides as in cases 1 and 3 of Figures 6 or at both side as in case 2 of the same figure. To determine on which side the edge is located, we can move the robot very small distances in the x direction to left and to the right, and by combining these readings we can determine the edge location within Δx tolerance. Another way to determine the edge location is by rotating the robot clockwise and counter-clockwise using small angles, and combining the readings as before to determine the edge location.

Translation in the x direction

Moving the robot in the x direction will result in an overlapping region equals to:

$$y \tan\theta - 2\Delta x$$

as shown in Figure 7. The time needed for this movement is equal to:

$$t_x = \Delta x / v_r$$

However, in our experiment, the robot can only move forward and backward. To move in the x direction, we need to rotate the robot 90 degrees, then move forward Δx, and finally rotate back 90 degrees again. Therefore, the time needed to move Δx is:

$$t_x = \pi\omega + \Delta x$$

and by adding the time to decide taking this movement, the total time will be:

$$t_x + t_d$$

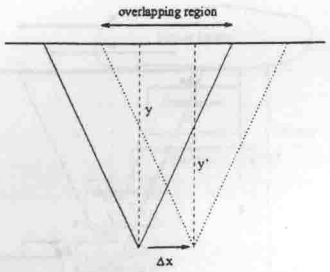

Figure 7: Translation in the x direction.

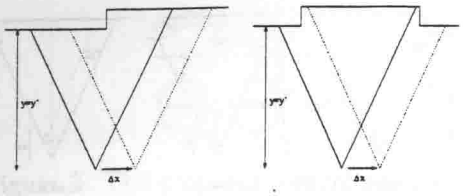

Figure 8: moving in the x direction with $y' = y$

If the two readings are the same (i.e., $y' = y$), then we have two possibilities; the measured point is in the overlapping region, or it is outside the overlapping region. Figure 8 shows the two cases. The probability that the two readings correspond to points in the overlapping region is:

$$\frac{y\tan\theta - 2\Delta x}{y\tan\theta + \Delta x}$$

This formula shows that the probability of the point to be in the overlapping region decreases by increasing Δx

If the two readings are different (i.e., $y' \neq y$), then again, we have two cases as shown in Figure9; $y' > y$ which means that there is an edge within the left Δx region, and $y' < y$ which means that there is an edge within the right Δx region.

Rotating the robot ϕ degrees

Rotation is similar to moving in the x direction. By rotating a small angle ϕ, where

$$-2\theta < \phi < 2\theta$$

and with rotation radius r, there will be an overlapping area as shown in Figure 10. This overlapping area starts at a distance d_o which can be calculated as follows:

$$d_o = ec - gc$$

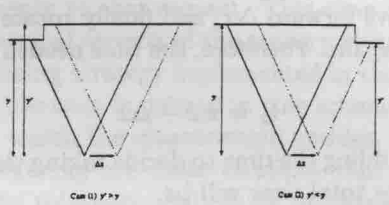

Figure 9: moving in the x direction with $y' \neq y$

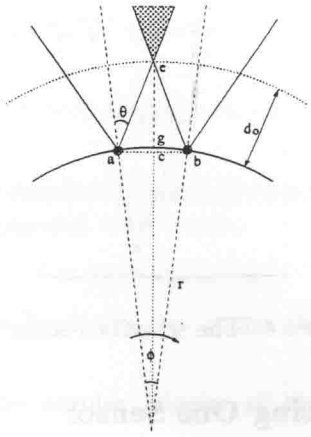

Figure 10: Rotating the robot ϕ degrees.

where

$$ec = r\sin(\phi)\tan(\phi + \frac{\pi}{2} - \theta)$$

$$gc = r - r\cos(\phi) = r(1 - \cos(\phi))$$

finally, we have:

$$d_o = r[\sin(\phi)\tan(\phi + \frac{\pi}{2} - \theta) + \cos(\phi) - 1]$$

and for small values of ϕ, d_o can be approximated by:

$$d_o = \frac{r\phi}{\tan(\theta)}$$

Therefore, if the sensor reading is y, we should rotate the robot such that $y > d_o$, otherwise, the reading will be outside the overlapping region. In other words, the rotation angle ϕ is limited by the sensor reading as follow:

$$\phi < \frac{y\tan(\theta)}{r}$$

Using case analysis similar to what we did for moving the robot in the x direction, with substituting Δx with $r\phi$, we can get new tolerance regions with probabilities associated to them in the same way we did before for the translation in the x direction.

The time needed to rotate ϕ degrees t_{rot} is equal to $\omega\phi$ and adding the decision time t_d we get the total time.

3.2 Using Multiple Sensors

This case is exactly the same as rotating the robot, except for the fact that the angle ϕ is fixed. In case where the sensors are arranges in a circle and distributed on equal spacing angles, ϕ depends on the number of sensors used. For example, if we have 24 sensors, then $\phi = \frac{2\pi}{24}$. To have an overlapping region, ϕ should be less than 2θ. Also, to consider this overlapping region, the sensor readings y for both sensors should be greater than d_o as discussed before. Again, the case analysis that we did for translation in the x direction can be

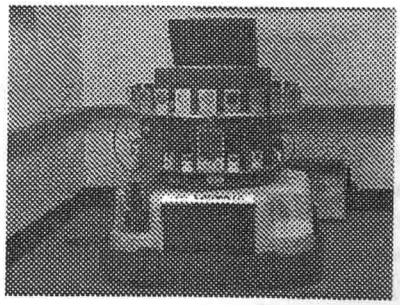

Figure 11: The LABMATE robot with its equipments.

used here by replacing Δx with $r\phi$. This way we can get smaller tolerance areas with certain probabilities. The probability that the reading is in the overlapping area depends on the value of the readings and on the angles ϕ and θ.

4 Experiments and Simulation Results

A simulator called *XSim* has been developed to examine the applicability of the proposed control scheme. This simulator is based on a mobile robot called "LABMATE" designed by Transitions Research Corporation [9]. The LABMATE is equipped with 24 sonar sensors, eight infrared sensors, a camera and a speaker. [1] Figure 11 shows the LABMATE with its equipment.

4.1 Simulation Results

Several experiments were performed on the simulator to check the applicability and validity of the proposed control scheme, and the results were very encouraging.

Experiment (1)

In this experiment, we implement a goal-directed client which tries to move the robot to a certain goal location. This client has priority 5 which is higher than a 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 are some obstacles between them. Figure 12 shows the robot trajectory for this experiment from the initial location to the goal location. Notice that at several points, the collision avoidance client took over and moved the robot away from the obstacles, then the new client updates the direction towards the goal point.

[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.

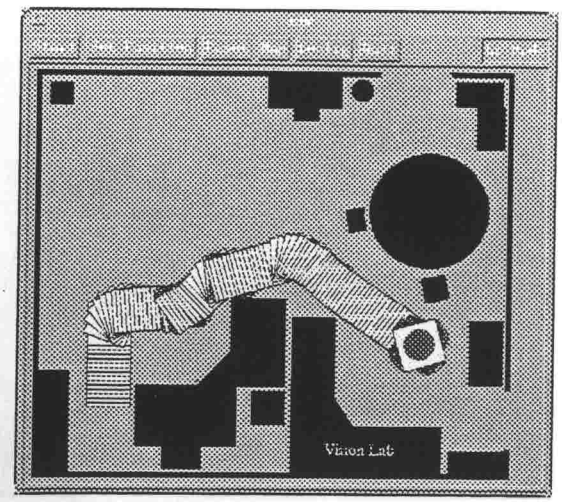

Figure 12: The trajectory of the robot from the initial to the goal point.

Experiment (2)

In this experiment, We demonstrate the use of the tolerance measures discussed in Section 3. this experiment also illustrates the use of the logical sensors concept to implement high-level requests which incorporate tolerance measures and time calculations.

The request which was implemented for this experiment is *measure* which has the following syntax:

$$measure(tolerance, time, preference)$$

where *tolerance* is the required tolerance with 0 meaning get the best tolerance, and -1 means tolerance is not important. *time* is the required response time, and again 0 means as fast as possible, and -1 means time is not important. When both, *time* and *tolerance* are specified, the logical sensor may not be able to satisfy both criteria, and this is when *preference* is used to specify which criteria should be preferred. This request returns the resulting tolerance and the time consumed into the same parameters that were sent.

The following is the output of a program which uses this request to measure a point in front of the robot. First it sends a request to get the measure as fast as possible ignoring the tolerance.

```
Fast response required ...
!!! minimum time ...
!!!  current reading is 2071 mm,
     with tolerance 402.4 in time 0.9 sec.

Result: distance = 2071 mm,
tolerance = 402.4, and time = 0.9 sec.
```

Second, the program sends a request to get the best accuracy (minimum tolerance), and the time is irrelevant.

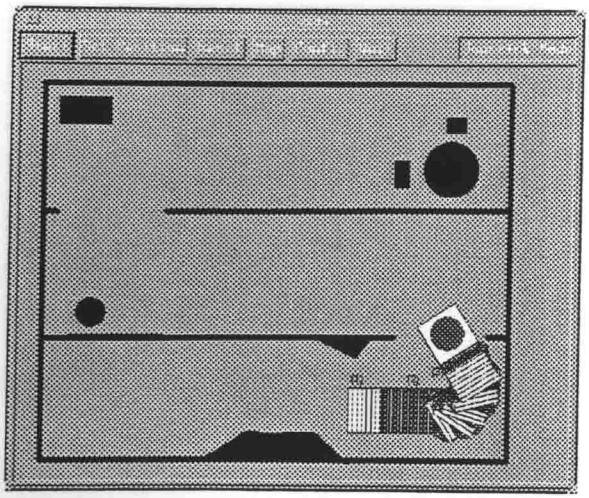

Figure 13: The trajectory of the robot while performing the requests.

```
Best tolerance required ...
!!! minimize the tolerance ...
!!!  current reading is 1536 mm,
     with tolerance 298.4 mm. in time 0.6 msec.
.........
!!!  current reading is 1091 mm,
     with tolerance 212.0 mm. in time 10.4 msec.
.........
!!!  current reading is 603 mm,
     with tolerance 117.2 mm. in time 19.0 msec.

  distance = 1536 mm,  tolerance = 117.2,
  and time = 19.0 msec.
```

Finally, the program specifies both time and tolerance to be met, preferring the time.

```
Tolerance required = 150.0, time required = 6.0 msec.
!!! both criteria are specified ...
!!!  current reading is 2190 mm,
     with tolerance 425.5 mm. and time 0.9 msec.
!!!  current reading is 2101 mm,
     with tolerance 408.2 mm. and time 2.7 msec.
!!!  current reading is 2068 mm,
     with tolerance 401.8 mm. and time 4.1 msec.
!!!  current reading is 2026 mm,
     with tolerance 393.6 mm. and time 5.6 msec.

Result: distance = 2190 mm,
tolerance = 393.6, and time = 5.6 msec.
```

Figure 13 shows the movement of the robot while taking these measurements. The first request did not cause any movement since it required minimum time. The second request cased the robot to move forward to minimize the tolerance region. During this movement, the speed of the robot decreases to get better accuracy. Finally, the last request also caused the robot to move forward, but it stopped before reaching the required tolerance since the time was preferred.

In this experiment we used only the translation in the y direction to minimize the tolerance. However, the other two approaches mentioned in Section 3 could be used to get better results with probability measures as well.

5 Conclusions

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. 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. Tolerance measures for sonar sensors were proposed and different strategies to increase position accuracy were investigated.

References

[1] 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.

[2] 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.

[3] HENDERSON, T. C., AND SHILCRAT, E. Logical sensor systems. *Journal of Robotic Systems* (Mar. 1984), pp. 169–193.

[4] KLEEMAN, L., AND KUC, R. An optimal sonar array for target localization and classification. In *IEEE Int. Conf. Robotics and Automation* (May 1994), pp. pp. 3130–3135.

[5] KORBA, L. Variable aperture sonar for mobile robots. In *IEEE Int. Conf. Robotics and Automation* (May 1994), pp. 3142–3147.

[6] SABATINI, A. M., AND BENEDETTO, O. D. Towards a robust methodology for mobile robot localization using sonar. In *IEEE Int. Conf. Robotics and Automation* (May 1994), pp. 3136–3141.

[7] 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.

[8] SHILCRAT, E. D. Logical sensor systems. Master's thesis, University of Utah, August 1984.

[9] TRC TRANSITION RESEARCH CORPORATION. *LABMATE user manual, version 5.21L-f.*, 1991.

[10] UNIVERSITY OF TENNESSEE, KNOXVILLE. *MPI: a message-passing interface standard.*, May 1994.