Journal of Intelligent and Robotic Systems **12**: 239–258, 1995.

Ultrasonic Navigation for a Wheeled Nonholonomic Vehicle

A. CURRAN and K. J. KYRIAKOPOULOS
New York State Center for Advanced Technology in Automation and Robotics and Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, U.S.A.

(Received: 20 June 1994)

Abstract. In this paper, we demonstrate a reliable and robust system for localization of mobile robots in indoors environments which are relatively consistent to a priori known maps. Through the use of an Extended Kalman Filter combining dead-reckoning, ultrasonic, and infrared sensor data, estimation of the position and orientation of the robot is achieved. Based on a thresholding approach, unexpected obstacles can be detected and their motion predicted. Experimental results from implementation on our mobile robot, Nomad-200, are also presented.

Key words. Mobile robot, ultrasonic navigation.

1. Introduction

Consider a typical indoors structured environment as shown in Figure 1. Given ultrasonic, infrared, odometry data and a representation of the environment, a robust method for localization of a moving mobile robot is desired. This localization method should not rely on alterations of the environment (beacons or markers) or be influenced by unexpected objects not depicted in the a priori environment. The problem to design methods enabling the estimation of the position and orientation of mobile robots is not new. Various sensors have been reported in the literature (Vision [8], Optical Range finders [4], Ultrasonic Beacons [9], Ultrasonic sensors [1, 5, 7]). The main issues affecting the design of a localization system is speed, cost and accuracy.

Reported results on vision guided localization (Kak and Kosaka [8]) show superior accuracy but only with the reduction of speed and increased cost. Ultrasonic beacon estimation dramatically improves speed with an expected decrease in accuracy but also requires that the environment be altered by the addition of ultrasonic beacons. Research in pure ultrasonic ranging has been directed towards statistical map matching, wall matching, and individual sensor-matching techniques. A static ultrasonic localization method by statistical map matching [7], requires a large amount of computational time. A wall-matching technique

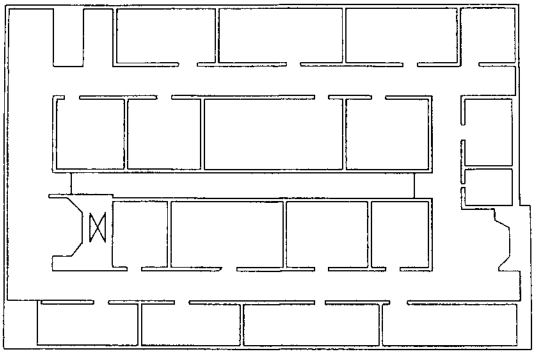


Fig. 1. Typical indoor environment.

proposed by Cox [4] performs well using optical rangefinders but its performance would likely decrease with the large beamwidth of ultrasonic sensors as compared to optical sensors since it requires a sensor with a small beamwidth. A similar wall-matching technique by Crowley [5], uses ultrasonic sensors and requires finding at least 3 consecutive sensors which form a line, thereby indicating the existence of a wall in the environment. Therefore, many sensors containing relevant data may be ignored. Predciado et al. [1] implemented a recursive method incorporating sensor information which improves the estimate. However, they do not attempt to accurately determine the expected sensor values by examining the entire sensor beamwidth but place a variance encompassing the entire beamwidth. Initially, in their recursive approach, the angle orientation is updated and, subsequently, positioned. This is less efficient than our method that updates orientation and position simultaneously.

We want to design a system which can combine all relevant sensor information to accurately localize the robot while it is moving. In addition, this system should identify areas possibly containing obstacles not indicated in the a priori map. By defining these areas, an avoidance algorithm can be designed to optimally avoid the detected obstacles [11].

We present a system that combines the extended Kalman filter with a thresholding technique and successfully localizes a mobile robot with or without the addition of unknown obstacles in the environment.

As shown in Figure 2 we obtain a rough estimation of the position and orientation based on the differential increments of the encoder values, X_r, Y_r and θ_r. If we filter the sensor-information vector, Z, to consider only relevant information and combine this with the encoder-based estimation of position and orientation, we can derive a better estimate of the position and orientation using an extended Kalman filter. Furthermore, by comparing the sensor readings with the a priori known map and using a thresholding approach, unknown obstacles are detected and their motion estimated. Finally, we use a nonholonomic motion controller which uses the position and orientation estimates to navigate the vehicle and complete the system loop.

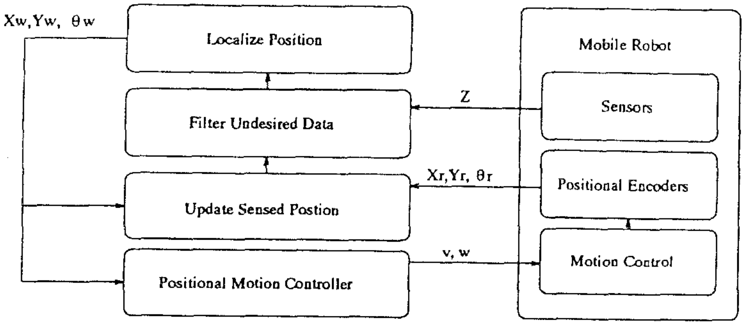


Fig. 2. Program layout.

2. Problem Statement

In order to state the problem, a descriptive analysis of both the vehicle and sensor models is necessary.

2.1. DESCRIPTION OF ROBOT

The robot (Fig. 3) is a 3-wheeled, cylindrical, zero-gyro radius robot. Sensor information is obtained from ultrasonic, tactile, and infrared sensor rings. In each ring, there are 16 individual sensors located at 22.5-degree increments around the robot. Odometry measurements are obtained from encoders located on a synchronous drive system.

2.2. VEHICLE MODEL

Assuming a two-dimensional world, we can define the robot configuration w.r.t. a world-coordinate frame W by vector $X_w = [x_w\ y_w\ \theta_w]^T$ containing its position and orientation. We consider another coordinate frame R which is the world-

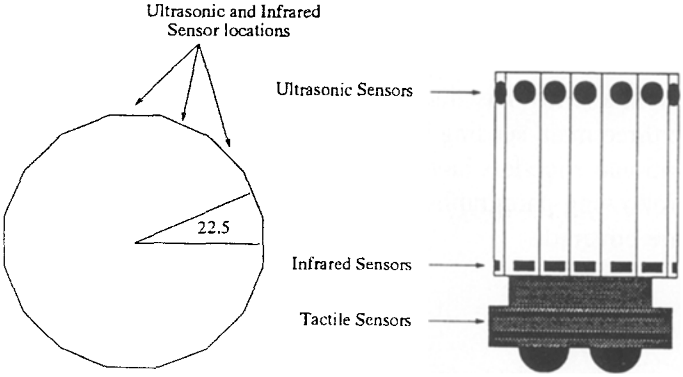


Fig. 3. Mobile robot sensor configuration.

coordinate frame that defines the motion of the robot based on odometry. W and R are not the same, in general, because R includes the uncertainty accumulated by the integration of the error during rolling. Let's denote the configuration of the robot w.r.t. R by $X_r = [x_r\ y_r\ \theta_r]^T$. The motion equations w.r.t. to R can be given by the unicycle model

$$\dot{X}_r = f(X_r) \cdot u \tag{1}$$

$$f(X_r) = \begin{bmatrix} \cos(\theta_r) & 0 \\ \sin(\theta_r) & 0 \\ 0 & 1 \end{bmatrix} \qquad u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{2}$$

described in [12], where u_1, u_2 are the translation and rotational velocities respectively. The motion equations w.r.t. W can be easily obtained if the odometry error (wheel slippage, etc.) is included as noise $n = [n_1\ n_2\ n_3]^T$ and

$$\dot{X}_w = f(X_w) \cdot u + n \tag{3}$$

It is obvious that the addition of noise creates the relative motion between frames R and W, depicted in Figure 4.

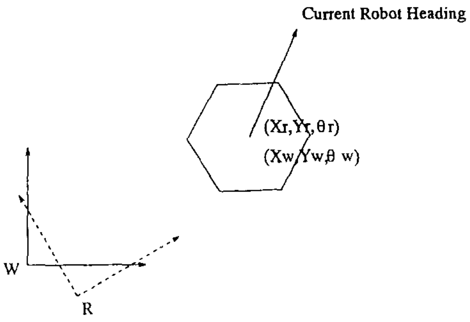


Fig. 4. World and robot frames.

2.3. DESCRIPTION OF SENSORS

Any type of sensor inherently has disadvantages which need to be considered when using it. The three main sensing instruments used for this system, infrared, ultrasonic, and positional encoders have distinct disadvantages which need to be addressed. In the following paragraphs, an overview of the problems associated with each sensor are covered.

2.3.1. Ultrasonic Sensors

Each ultrasonic sensor has a beamwidth of approximately 23.6° [3]. By examining all 16 sensors, we can obtain a 360° panoramic view fairly rapidly.

Unfortunately, ultrasonic sensors work upon a send/receive echo-type format. Therefore, no two sensors can be simultaneously activated unless specific software and/or hardware is included to distinguish the different sonar signals [2]. In order to decrease crosstalk (the transmit pulse of sensor 'i' being received and causing an erroneous value on another sensor), we must fire each sensor individually. In addition, since the Polaroid sensor modules combine a transmit and receive system into one compact package, we must blank the receive system such that the residual transmitted pulse on the sensors in not detected as a received pulse. Therefore, with the sonar system we can detect objects from a minimum range of 17 inches to a maximum range of 22 feet with a 30-degree resolution.

2.3.2. Infrared Sensors

Similarly to the sonic sensors, infrared sensors work upon a send/receive format. These sensors emit an infrared light from one source, and measure the amount of reflected light with two light detectors. Since these devices measure light differences, they are highly biased by the environment. Object color, object orientation, and ambient light all can contribute to erroneous readings but since the transmission signal is light instead of sound, we may expect a dramatically shorter cycle time for obtaining all infrared-sensor measurements. Considering all these problems as noise factors, we decided that infrared measurements are only acceptable for short distances. In our system, infrared sensors were used to provide information for the shorter than 17-in. area, where the ultrasonic sensors are not used. The overall range covered by the sensing system can be found in Figure 5.

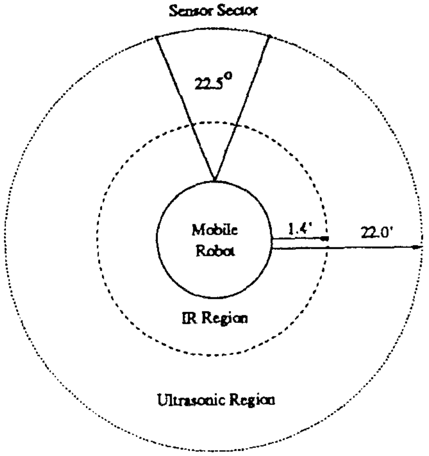


Fig. 5. Sensing-system range.

2.3.3. *Odometry Measurements*

A low-level integration routine calculates odometric location using the current and previous translation ($d_e(k)$), and rotation ($r_e(k)$) sampled encoder values, of the mobile robot according to the following equation [13].

$$\begin{aligned} x_r(k) &= x_r(k-1) + \Delta d_e \cdot \cos((r_e(k) + r_e(k-1))/2) \\ y_r(k) &= y_r(k-1) + \Delta d_e \cdot \sin((r_e(k) + r_e(k-1))/2) \\ \theta_r(k) &= r_e(k) \end{aligned} \tag{4}$$

where $\Delta d_e = d_e(k) - d_e(k-1)$. Odometry, unfortunately, is very sensitive to errors. Unless we assume perfect rolling conditions, we should expect to obtain some odometry-measuring errors in the form of drift, bias, and slippage. When dead-reckoning is solely used, errors accumulate over time as integration errors, due to the nonholonomic nature of the rolling motion. Over short travel distances we can expect small errors but over long paths these errors will grow.

2.4. MATHEMATICAL STATEMENT OF PROBLEM

If dead-reckoning was used to obtain the configuration X_w from integration (i.e. if integration of Equation (1) was only used) then over large distances, this position would contain large integration errors. Therefore, the knowledge provided by the system equations should be complemented by this of the ultrasonic and infrared sensors. Since those sensor measurements are not overlapping (i.e. only one range value is considered valid for each sector), we can combine these measurements thereby creating a sensor vector Z, which contains 16 elements signifying the distance for each sector.

Therefore, we are looking for some function $\mathcal{F}$, that sequentially provides optimal estimates

$$\widehat{X}_w(k) = \mathcal{F}(\widehat{X}_w(k-1),\ Z(k)) \qquad k = 1, 2, \ldots \tag{5}$$

of the configuration of the robot by combining the system equations knowledge and the sensor readings. Optimality is in the sense that the estimates should minimize an error criterion

$$J = E\big\{(\widehat{X}_w(k) - X_w(k))^T \cdot (\widehat{X}_w(k) - X_w(k))\big\} \tag{6}$$

where $E\{\cdot\}$ denotes the expected value of a random variable.

In order to navigate in the environment, a motion controller of the form

$$u(k) = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = G(\widehat{X}_w(k),\ X_d(k)) \tag{7}$$

is needed to compute the control inputs to bring the robot to the desired configuration $X_d(k)$.

3. Approach of Solution

The problem, as stated above, was attacked using the Extended Continuous-Discrete Kalman filter [6]. In the following subsections, the adaptation of the Kalman filter to handle this problem is presented.

3.1. MODEL EQUATIONS

The system and measurement models are:

$$\dot{X}_w = f(X_w(t)) + n \tag{8}$$

$$z_k = h_k(X(t_k)) + v_k \qquad k = 1, \ldots, 16 \tag{9}$$

where $n \sim N(0, Q)$, $v_k \sim N(0, R)$, are the odometry and sensor noises, z_k are the sector k sensor measurements and h_k, the kth measurements function, is the function relating the current configuration with the measurement that is expected to be received from the kth sensors.

3.2. MEASUREMENT FUNCTIONS

Given the current configuration X_w of the robot, we need to calculate the expected sensor values for each sector. This is actually the minimum distance ray intersecting a wall contained in the kth sensor's beamwidth. Denoting x_k^i and y_k^i as the coordinates of the minimum distance intersection for sensor k, and $x_{s_k}^i$ and $y_{s_k}^i$ as the coordinates of the kth sensor, at time instant t_i, (see Fig. 6), we formulate the distance equation as:

$$h_k(X_w(t_i)) = d_k^i = \sqrt{(x_k^i - x_{s_k}^i)^2 + (y_k^i - y_{s_k}^i)^2} \tag{10}$$

where $x_{s_k}^i, y_{s_k}^i, x_k^i$, and $y_{s_k}^i$ are all functions of X_w.

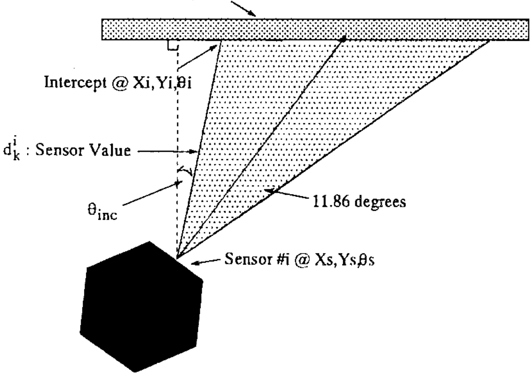


Fig. 6. Derivation of sensor measurement.

3.3. STATE PROPAGATION

The state estimation propagation equation is:

$$\dot{X}_w = f(X_w(t)) \tag{11}$$

In the implementation of the integration of the above equation we encounter timing problems due to the timing delay between decision of a new control and low-level implementation. If we assume that over short distances odometry is accurate, a fairly accurate odometry value based upon incremental encoder changes can be obtained. By reformulating the propagation equation we have:

$$\dot{X}_w \delta t = \begin{pmatrix} \cos(\theta_{\text{diff}}) & -\sin(\theta_{\text{diff}}) & 0 \\ \sin(\theta_{\text{diff}}) & \cos(\theta_{\text{diff}}) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \Delta x_r \\ \Delta y_r \\ \Delta \theta_r \end{pmatrix} \tag{12}$$

The rotation of the incremental odometry changes is necessary to realign the robot and fixed frames. θ_{diff} is the difference between the estimated and odometric angle of the wheels, which is equivalent to the difference between the world and mobile robot frames.

3.4. COVARIANCE MATRIX PROPAGATION

The error covariance matrix is defined as $P = E\{(\widehat{X}_w(k) - X_w(k)) \cdot (\widehat{X}_w(k) - X_w(k))^T\}$ and its time propagation is governed by the following matrix Ricatti equation

$$\dot{P}(t) = F(\widehat{X}_w(t))P(t) + P(t)F^T(\widehat{X}_w(t)) + Q \tag{13}$$

In order to improve the accuracy of the error covariance propagation, we decided to use a Taylor series second order approximation to the propagation equation by using the 2nd order time derivative of the error covariance matrix

$$\ddot{P}(t) = \frac{\partial \dot{P}(t)}{\partial \widehat{X}_w} \cdot \dot{\widehat{X}}_w \tag{14}$$

where F is defined as

$$F(\widehat{X}_w(t), t) = \left.\frac{\partial f(X(t))}{\partial X(t)}\right|_{X(t)=\widehat{X}_w(t)} \tag{15}$$

3.5. FILTER EQUATIONS

The model of the mobile robot and the measurement function h_k that is of trigonometric form, are both inherently nonlinear. Therefore, an extended version of the Kalman filter is necessary. Listed below are the Filter Equations:

$$\begin{aligned}
&\widehat{X}_k(+) = \widehat{X}_k(-) + K_k[z_k - h_k(X_k(-))] \\
&P_k(+) = [I - K_k H_k^T(\widehat{X}_k(-))] - P_k(-) \\
&K_k = P_k(+) H_k^T R_k^{-1}
\end{aligned} \tag{16}$$

with H_k defined as

$$H_k(\widehat{X}_k(-) = \left.\frac{\partial h_k(X(t_k))}{\partial X(t)}\right|_{X(t_k)=\widehat{X}_k(-)} \tag{17}$$

where $\widehat{X}_k(-)$ is the expected state just before the arrival of the kth measurement (it is obtained by state equation integration, as described in Section 3.3), $\widehat{X}_k(+)$ is the expected state just after the arrival of the kth measurement, $P_k(-)$ is the error covariance matrix just before the arrival of the kth measurement (it is obtained by error covariance matrix equation integration, as described in Section 3.4) and $P_k(+)$ is the error covariance matrix just after the arrival of the kth measurement.

There are two methods in order to calculate the Kalman gain matrix K_k [6]. We chose the form shown above so that the largest matrix inversion would be of a 3 × 3 matrix.

3.6. UNEXPECTED OBSTACLE CLASSIFICATION

During robot operation, various unexpected obstacles may interfere with robot localization. Suppose, an unexpected obstacle detected by sensor j is located in the environment as shown in Figure 7. In addition, suppose that sensor 'i' expected to catch the corner at A as its distance measurement, missed corner A.

Consider now the profile of the difference value (i.e. actually received range signals - predicted range signals) from the 16 sensors as presented in Figure 8. If we examine the difference between predicted and actual sensor information for

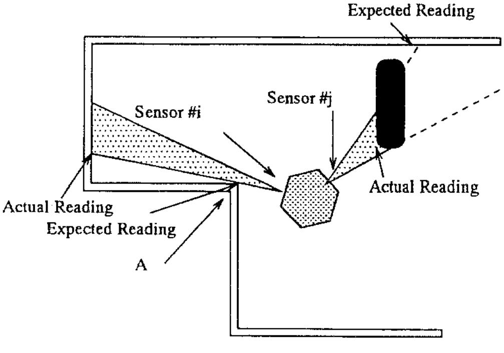


Fig. 7. Sensing discrepancies.

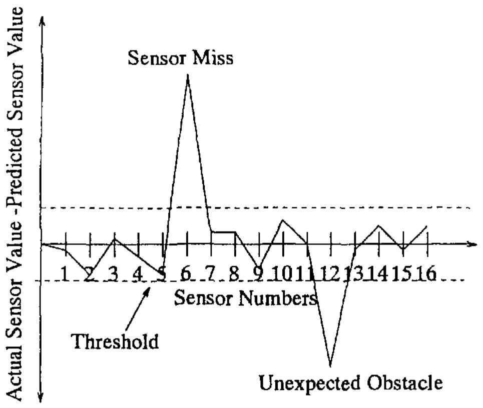
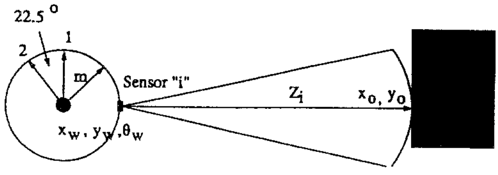


Fig. 8. Sensor comparison.

this example, we would notice a large discrepancy between valid and incorrect data for those two cases. Therefore, an alteration is introduced so that the Kalman filter will not be incorrectly biased by sensor readings i and j. In normal operation, each sensor has a variance associated with it, indicating the uncertainty that we have about the incoming data from the sensor. If we decrease this uncertainty factor for a given sensor, we decrease the bias that the robot has from that sensor. By significantly increasing this uncertainty factor we essentially remove its contribution to the Kalman filter. If a threshold is created, as shown in 8, then all values outside the threshold are considered incorrect and ignored.

In addition to unbiasing the filter due to incorrect measurements, this method also helps to detect possible unexpected obstacles. In the event that an unmodelled object is in the field of view of the sensors, these return values are considerably lower than the predicted values. When comparing the actual to predicted sensors value, we will obtain a large negative discrepancy. Now, if we consider the missed edges problem, we will obtain the inverse. In the missed edge case, we will obtain actual values dramatically larger than the predicted. Therefore, through this thresholding method we can identify the type of sensor discrepancy, missed edge or unmodelled feature.

In the case of unpredictable obstacle detection no additional information, except that it exists somewhere in the sensors beamwidth, can be obtained. To simplify the calculations, it is assumed that the object is centered in the middle of the beam. The sensor indicates the existence of an object at a certain distance from the robot in a specific sector. If we project from this sensor, the distance measured, as shown in Figure 9, we obtain as estimated position of the unexpected obstacle (x_o, y_o) using the following equation:



Note : Sensor "1" Aligned with θ_w

Fig. 9. Estimating obstacle location.

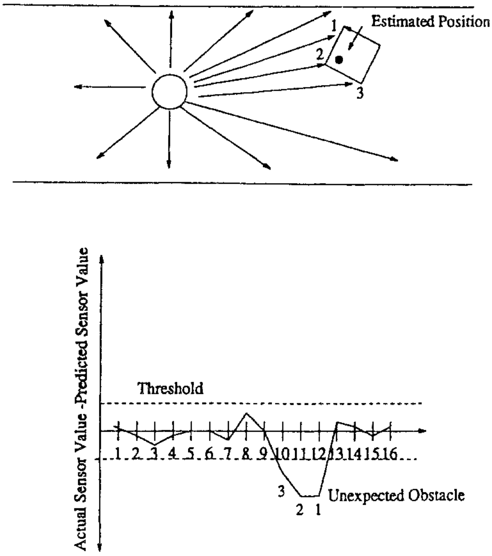


Fig. 10. Unexpected obstacle detection.

$$x_o = \widehat{x}_w + (m + z_i) \cdot \cos(\widehat{\theta}_w + (i - 1) \cdot 22.5^\circ) \tag{18}$$

$$y_o = \widehat{y}_w + (m + z_i) \cdot \sin(\widehat{\theta}_w + (i - 1) \cdot 22.5^\circ) \tag{19}$$

where m is the robot's radius and z_i is a sensor detecting the unexpected obstacle. In that way a collection of obstacle positions corresponding to the same obstacle is gathered. For example, in Figure 10, position 1, 2 and 3 correspond to the actual obstacle faces.

Obstacle movement can be extrapolated using equations:

$$v_x(k) = \frac{x_o(k) - x_o(k-1)}{T} \tag{20}$$

$$v_y(k) = \frac{y_o(k) - y_o(k-1)}{T} \tag{21}$$

where v_x, v_y are approximate estimates of the translational velocities along the x and y axis' based on position (x_o, y_o) at times $k, k-1$. Position (x_o, y_o) can be obtained by averaging three (3) obstacle face locations obtained in 18, 19 as shown in Figure 10.

If the relative obstacle motion is radial w.r.t. the robot, the proposed scheme works quite well. For tangential motion it may give poor results. This is primarily due to the beamwidth of the sensors. Tangential motion within a sensor's beam is not detected and unless a new sensor is effected by the obstacle motion the obstacle will appear to be stationary w.r.t. the robot.

Since this detection scheme will be primarily used to prevent collision this deficiency is not catastrophical. The algorithm will accurately estimate any object with relative motion towards the robot while objects not on collision paths are estimated with less accuracy. Implementation problems and results are discussed in the following sections.

4. Implementation

Implementation of the localization system required compromises based on numerous experiments. In the following paragraphs, problems and solutions associated with each area are highlighted.

4.1. DEFINITION OF THE A PRIORI MAP

In order to make the system stable, we needed to minimize the number of calculations per cycle. We decided to simplify the a priori map definition so that the calculations involved in $h(X_w)$ are minimized. In ignoring any fine detail of the hallway (i.e. the 49 doors which are depressed from the hallway by approximately 5 inches) and selecting motion control gains accordingly, we decrease the robot's sensitivity to a non-ideal map. By ignoring these doors, we unfortunately increase the uncertainty in position and orientation.

4.2. SENSOR VARIANCES

The accuracy of ultrasonic sensors is highly dependent on the texture of the reflecting surface. Rough surfaces will return the wavefront produced by the sensor at any incident angle as opposed to smooth surfaces which will return only wavefronts which are incident on the surface at near to perpendicular angles as depicted in Figure 11. In our environment, we have several plasterboard-type walls which are considered to be of smooth texture. Based on the assumption, that we will not receive valid data from those walls due to the reflectivity properties mentioned above, all sensors that did not meet an incident angle requirement, θ_{req} on a smooth wall, were essentially removed by increasing the sensor variance on that sensor.

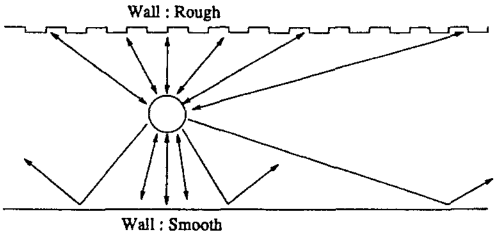


Fig. 11. Ultrasonic wave behavior.

Through experimentation, the variance dependency on distance was found. Generic ultrasonic sensors do not contain any orientation information of the reflecting surface, though it may be possible to obtain feature information as proposed by [3]. If we assume that we can accurately predict robot location and orientation, we may place an orientation variation to improve the system responce. Through additional experimentation with the entire system an orientational variance was determined. The resulting variance equation for sensor 'i' is:

$$R_{ii} = d_k^i \cdot (0.04 + 0.5|\sin(\theta_{\text{inc}})|) \tag{22}$$

where d_k^i is defined in Equation (10) and θ_{inc} is the angle from perpendicular that the predicted sensor ray encounters the wall. Both parameters are depicted in Figure 6.

The variance defined above is used unless, as detailed before, either of the two conditions is encountered

(1) $|d_i - z_i| > d_{\text{threshold}}$
(2) $\theta_{\text{inc}} > \theta_{\text{req}}$

4.3. ODOMETRIC VARIANCES

Odometric variances are dependent on the type of floor surface. In our experiments, we have a carpeted floor. Through experimentation we obtained the variances diag$(Q) = [0.001\ 0.001\ 0.0001]^T$, where Q was defined in Section 3.

4.4. MOTION CONTROLLER

The selected motion control algorithm was of the form

$$u_1 = -k(x_e \cos\theta_w + y_e \sin\theta_w) + \dot{x}_d \cos\theta_w + \dot{y}_d \sin\theta_w \tag{23}$$

$$u_2 = \frac{1}{\varepsilon}\,[-k(y_e \cos\theta_w - x_e \sin\theta_w) - \dot{x}_d \sin\theta_w + \dot{y}_d \cos\theta_w] \tag{24}$$

where $x_e(t) = x_w(t) - x_d(t)$, $y_e(t) = y_w(t) - y_d(t)$ are the position errors and $x_d(t), y_d(t)$ are the reference trajectories. It was implemented in order to track a desired trajectory and designed by Pappas and Kyriakopoulos [12]. This algorithm was chosen because of it is highly robust in correcting any type of error in robot position.

4.5. SOFTWARE

The navigation and localization algorithm was developed in C. Using a Nomadic Technologies interface, we communicated with the mobile robot. The structure of the code is depicted in the flow chart of Figure 12. New sensor information for all sensors, can be obtained roughly every 2 seconds. Since the Kalman filter utilizes past information, only new sensor information should be incorporated. Therefore localization was performed only once every five cycles i.e. approximately 2 seconds. Using dedreckoning between localization cycles we can have fairly accurate estimates of the position. Additionally, the error covariance matrix needs to be updated since error covariance varies into proportion to the traveled distance.

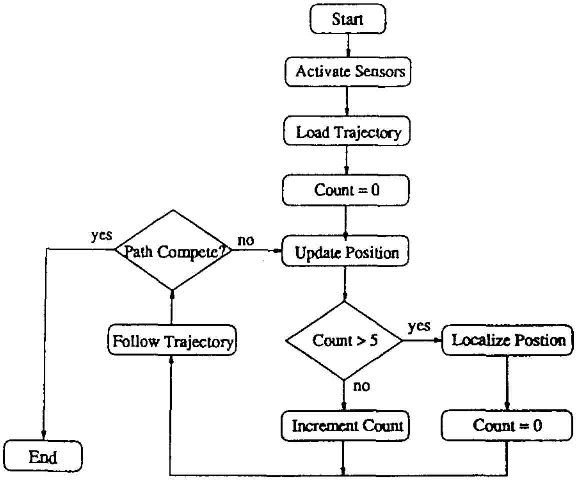


Fig. 12. Main program flow chart.

5. Experimental Results

The proposed Extended Kalman Filter localization algorithm was tested using a Nomadic Technologies' Nomad 200 mobile robot. The Nomad 200 is comprised of a 486 33Mhz Computer, with a radio link of 19200 baud rate to a Unix sparc2 Computer. Low-level motion control is achieved through a Gahlil motion controller. Currently, we are controlling motion through a rotational and translational velocity command. The Sensing capabilities of the Nomad 200 are 16 ultrasonic, infrared and tactile sensors. Currently, the localization algorithm is running on a Sparc2 computer. This was only done to facilitate corrections to the localization code. The total cycle time for localization and Motion control, including modem communication to the robot, is approximately 150–200 milliseconds.

5.1. LOCALIZATION

In order to test the localization method, we selected to experiment with the robot in our floor hallways. In these experiments, some of the laboratory doors were left open to test if erroneous reading are rejected. Seventy percent (70%) of the walls in our lab are smooth plasterboard which results in poor ultrasonic measurement for sensors not parallel to the wall. The remaining thirty percent of the walls are 9-in. square bricks. The depressions in the brick usually give a fairly accurate reading up to 50 degrees from perpendicular. The entire path is a carpeted floor. A rectangular path was selected with a total distance of 360 feet. Robot Velocity was set at 5 in./second because of the slow time to update all sonic sensors.

We can see in Figure 13 that we can succesfully localize the mobile robot, even though odometry has dramatically failed. The slight deviations we see from a path directly down the center of the hallway is due to room and elevator doors that were not modelled. If we compare the difference between odometry and the Kalman Filter based estimated position against the desired trajectory (see Fig. 14) we can see that our localization algorithm successfully removes the integration error existent coming from odometry. Finally, if we compare odometry and the estimated orientation against the desired orientation we also see an improvement (see Fig. 15). The 10-degree oscillations in the routine are sue to the oscillatory converging behavior of the control algorithm and the unmodelled doors.

In tabular form the results obtained from 10 successive runs are presented:

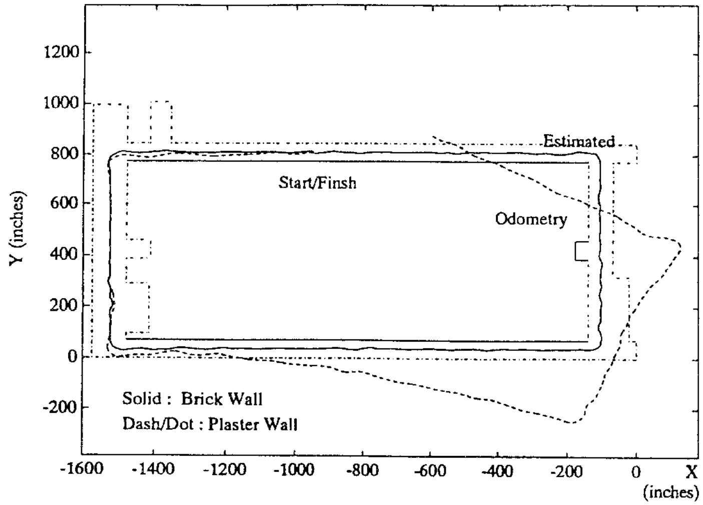


Fig. 13. Mobile robot navigation.

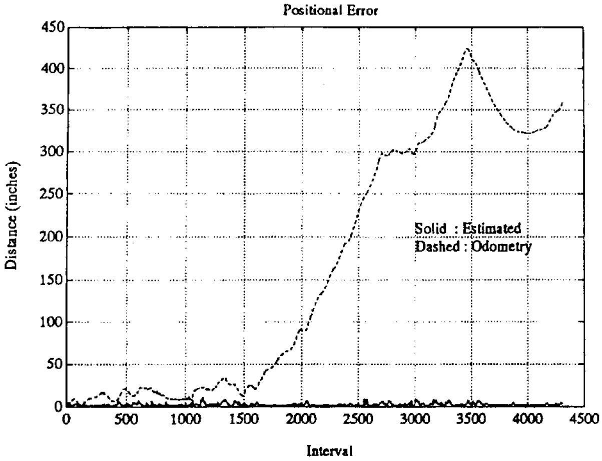


Fig. 14. Distance from desired trajectory.

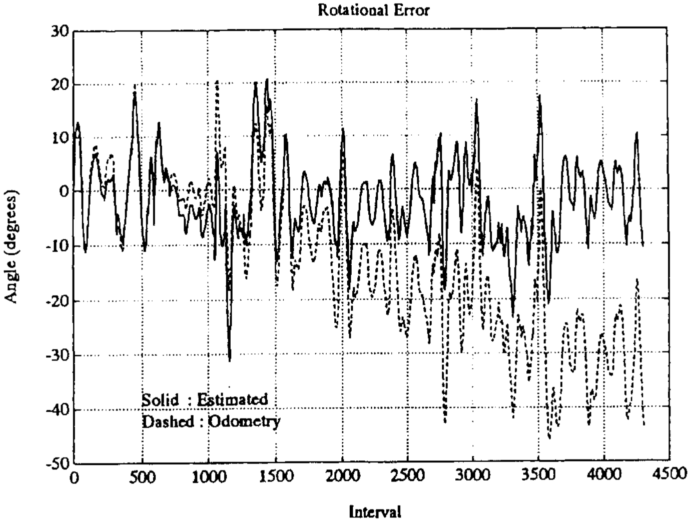


Fig. 15. Angle from desired rotation.

Table 1

Maximum X error (navigation)	18.1″
Maximum Y error (navigation)	12.5″
Maximum θ (navigation)	25.8°
Maximum normed error (navigation)	18.1″
Final X error	30.0″
Final Y error	3.0″
Final θ error	10°

The errors in Y and θ during navigation can mostly be attributed to the unmodelled door jams. The X error is attributed to the featureless in X information obtained from the long hallways. The larger errors at the final position are partly due to the nonholonomic controller.

5.2. OBSTACLE DETECTION

Equations (18), (19) and (21) show that in order to accurately determine the velocity of an obstacle the time at which each sensor is triggered is needed. In order to achieve this, serious hardware modifications should be made. But in order to test the value of the idea Matlab simulations were performed. Two cases were tested:

(1) Object Moving Directly to the Robot. The motion is shown on Figure 16. If we examine normed error between the position of the nearest obstacle face and the estimated position of that face of the robot we see that we can obtain a fairly accurate estimation of the position as shown in Figure 17. If we examine the difference between actual and estimated velocity a fairly accurate estimation of the velocity is obtained as displayed on Figure 18.

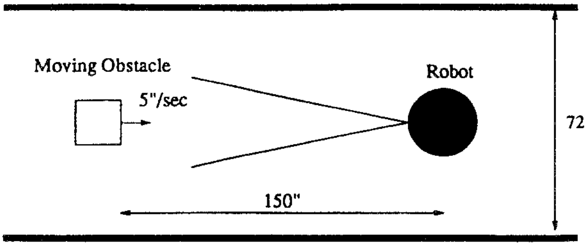


Fig. 16. Radial relative motion.

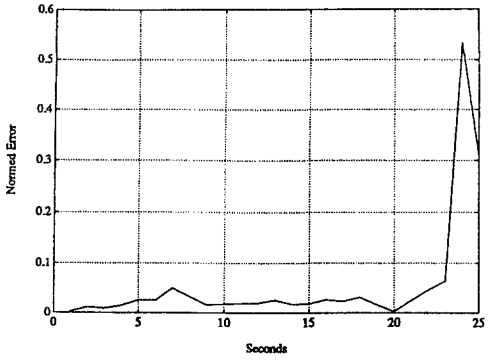


Fig. 17. Position error, radial approach.

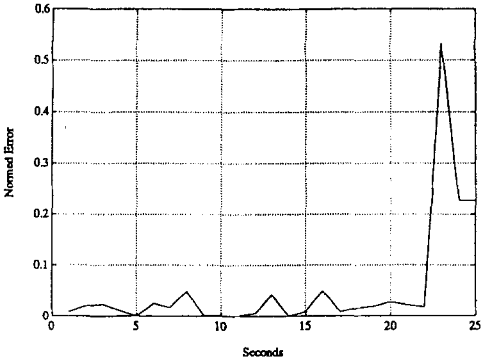


Fig. 18. Velocity error, radial approach.

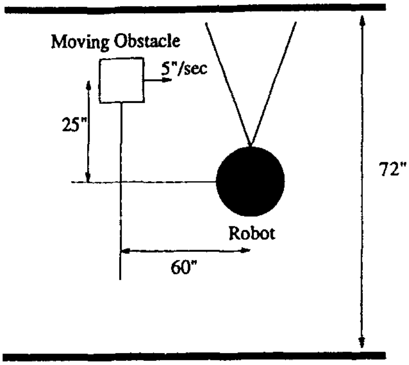


Fig. 19. Tangential relative motion.

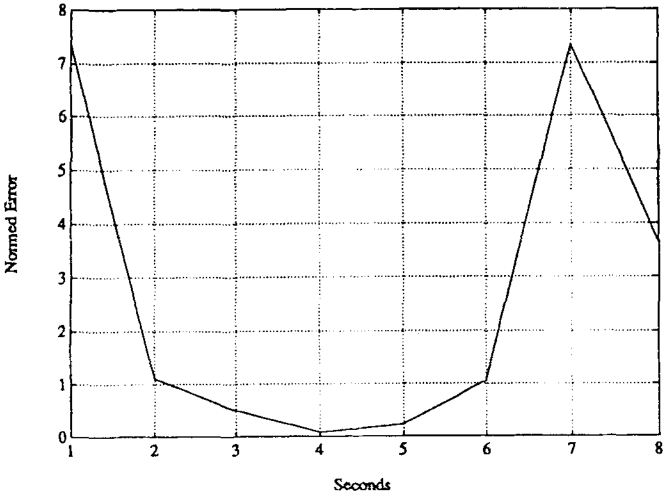


Fig. 20. Position error, tangential approach.

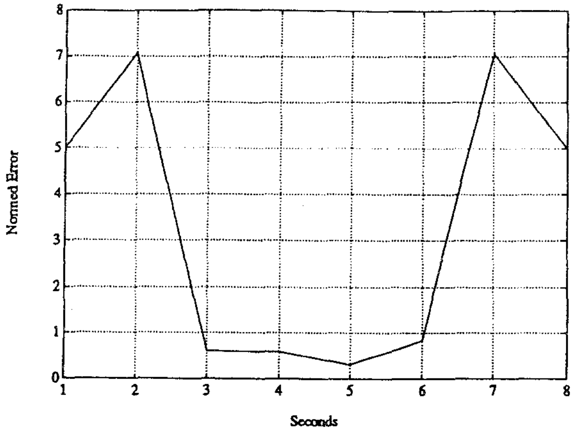


Fig. 21. Velocity error, tangential approach.

(2) Object Moving Tangentially to the Robot. Similarly, the motion is shown on Figure 19. Poor estimates of position/velocity are obtained as depicted on Figures 20, 21. As explained above this is attributed to the beamwidth of the sensors.

6. Discussion – Future Work

Using this method, we have shown that we can successfully localize a mobile robot in a partially known environment. Currently, the entire system is controlled from a Sparc2 computer. Initially, we plan to transfer the algorithm to the mobile robot's 486 computer to improve speed and autonomy. Since sensor information is obtained while the robot is in motion, we have essentially reduced the accuracy of the localization system (i.e. sensors are not all localized to one point in space). By possibly translating sensor information, we can effectively bring all sensor information to one point in space. In order to do so, the times where sensors are triggered should be available. To achieve that, serious hardware modifications should be made. We have shown that we can detect unexpected objects in the environment (see Figure 8). By correlating the unexpected objects detected between successive samples, we can obtain a rough approximation of object location and speed. Using this information, we are going to implement a collision prediction and avoidance scheme designed by Kyriakopoulos and Saridis [10].

Acknowledgements

This work has been conduxted in the New York State Center for Advanced Technology (CAT) in Automation and Robotics at Rensselaer Polytechnic Institute. The CAT is partially funded by a block grant from the New York State Science and Technology Foundation. The first author has been additionally supported by a scholarship from Raytheon Company. The authors would also like to thank Nomadic Technologies for their assistance.

References

1. Predciado, A., Meizel, D., Segovia, A. and Rombaut, M.: Fusion of multi-sensor data: A geometric approach, in *Proc. 1991 IEEE Int. Conf. Robotics and Automation*, April 1991, pp. 277–289.
2. Borenstein, J. and Koren, Y.: Obstacle avoidance with ultrasonic sensors, *IEEE J. Robotics Autom.* (April 1988), 213–218.
3. Bozma, O., and Kuc, R.: Building a sonar map in a specular environment using a single mobile sensor, *IEEE Trans. Pattern Analysis Machine Intelligence* (December 1991), 1260–1269.
4. Cox, I. J.: Blanche – an experiment in guidance and navigation of an autonomous robot vehicle, *IEEE Trans. Robitics Automat.* (April 1991), 193–204.
5. Crowley, J. L.: Asynchronous control of orientation and displacement in a robot vehicle, in *Proc. 1989 IEEE Int. Conf. Robotics and Automation* (May 1989), 1277–1282.
6. Gelb, A.: *Applied Optimal Estimation*, MIT Press, Cambridge, MA, 1980.
7. Holenstein, A., Muller, M. and Badreddin, E.: Mobile robot location in a structured environment cluttered with obstacles, in *Proc. 1992 IEEE Int. Conf. Robotics and Automation*, May 1992, pp. 2576–2581.
8. Kosaka, A. and Kak, A. C.: Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties, *Comp. Vision Graph. Image Proc. Image Understanding* (November 1992).
9. Kleeman, L.: Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead-reckoning, in *Proc. 1992 IEEE Int. Conf. Robotics and Automation*, 1992, pp. 2582–2587.
10. Kyriakopoulos, K. J. and Saridis, G.: An integrated collision prediction and avoidance scheme for mobile robots in non-stationary environments, in *Proc. 1992 Int. Conf. Robotics and Automation*, and to Appear in *AUTOMATICA* (1992).
11. Kyriakopoulos, K. J. and Saridis, G.: Optimal motion planning for collision avoidance of mobile robots in non-stationary environments, in *1992 Amer. Control Conf.* and submitted to *Transactions in Automatic Control* (June 1992).
12. Pappas, G. J. and Kyriakopoulos, K. J.: Dynamic modelling and tracking control of nonholonomic wheeled vehicles, Submitted to *IFAC'93 World Congress*, 1993.
13. Nomadic Technologies, Nomad 200 motion control software, Technical Report, Nomadic Technologies, Inc., 1991.