MULTISENSOR FUSION FOR MOBILE ROBOT POSITIONING AND NAVIGATION

Agostino Martinelli* Francesco Martinelli*
Salvatore Nicosia* Paolo Valigi**

* *Dipartimento di Informatica, Sistemi e Produzione, Università
degli Studi di Roma "Tor Vergata", Via di Tor Vergata, 110 -
00133 Roma (Italy)*

** *Dipartimento di Ingegneria Elettronica e dell'Informazione
Università di Perugia, Via G. Duranti, 93 - 06125 Perugia
(Italy)*

Abstract: In this work a complete algorithm is given, to localize a mobile robot in a partial variable environment. This algorithm uses an extended Kalman filter in order to combine odometric and sonar data. Two kinds of map are adopted, in order to take into account the partial variability of the environment, the former is used to localize the robot, the latter for motion planning purposes. The localization process is carried out while the robot is in motion. The experiments have been performed using a mobile robot Nomad150 in indoor environments. *Copyright © 2000 IFAC*

Keywords: Kalman filters, sensor fusion, robot localization, motion planning, dynamic environment

1. INTRODUCTION

The use of sensors with different characteristics is quite common in mobile robots, and it is well known that integration and fusion techniques allow to largely improve the performance of the overall system. Integration and fusion approaches have usually different objectives. In particular, the sensor integration concept is mainly used to generically indicate the combined use of information from sensors of different types, without the need for a common representation of the data from the different sensors. Sensor fusion is carried out at a lower level, with the specific objective of combining information from different type of sensors, usually with different data structure, onto a single data representation, and with a degree of accuracy higher than the accuracy of the single sources. In particular, sensor fusion can be carried

out both at the level of consecutive set of data from the same sensor, and at the level of data from different sensors (see, among many others, (Luo and Kay, 1989; Alag S. and A., 1995) and the references therein).

The information processing system has usually an intrinsically hierarchical nature, with the data from the sensors transformed into representation progressively more abstract, moving upward from the low level of the sensors up to the higher level, where information may become symbolic, e.g. to represent complex concept such as moving obstacles, target positions, landmarks. This corresponds also to different control and planning levels in a navigation architecture. For example, the feedback control of the motors usually is based on measurements/estimates of the current position, while obstacle avoidance algorithms consider data at higher levels, and trajectory planning may take into account information at a further abstract level.

¹ This work has been supported by the national project RAMSETE, funded by MURST

The basic advantages of sensor fusion and integration are: redundancy, complementarity, timeliness, and cost. As a matter of fact, data from several sensors, whenever related to the same portion of the environment, are naturally redundant, and may allow uncertainty reduction. Similarly, if any type of sensor is specifically designed for a different feature, then the natural complementarity allows to reduce overall error. Finally, timeliness allows a faster data processing, while fusion and integration of low cost sensors may turn out in an overall sensing system with a good cost/performance ratio.

Data fusion has a great importance in mobile robotics. It is fundamental in order to integrate a localization method that does not use beacons or markers, with a map-building algorithm that builds and constantly updates the world model (for more details see (Anousaki and Kyriakopoulos, 1999; Jetto L., 1998; Durrant White H. and E.M., 1998) and references therein).

In this paper, attention is on the realization of a sensor fusion system. Sensor fusion may be carried out both at low level and high level. While high level fusion usually refers to some form of indirect combination of sensor data, mainly at the level of control signals, low level fusion is used to indicate direct combination of data from the sensors. The approach used in this paper is based on the use of an extended Kalman filter (EKF) for fusion of odometric data with data from sonar sensors. The fusion scheme has been implemented, and extensively tested, on a Nomad 150 mobile platform, mounting 16 sonar sensors located at 22.5-degree increments around the robot. Odometry measurements are obtained from encoders located on a synchronous drive system.

The paper describes the design, realization and experimental validation of a system for encoder and sonar data fusion, with application to mobile robot positioning and navigation.

2. DATA FUSION

2.1 Vehicle model and odometry measurements

The paper considers a robot moving on a 2-dimensional environment. The robot is a three wheel synchronous drive non-holonomic system with zero gyro-radius. It can only translate along the forward and backward directions along which the three wheels are aligned. The robot has a zero gyro-radius, i.e., the robot can rotate around its center.

It is possible to define the robot configuration with respect to a global reference W by the vector $X_W = [x_W(t), y_W(t), \theta_W(t)]^T$ containing its

position and orientation. On the other hand the robot configuration estimated by odometry measurements is different from the actual configuration X_W because of the odometric errors like drift, bias and slippage, due to unperfect rolling conditions.

Without considering odometry errors the motion equations are the well known unicycle equations:

$$\dot{x}_W(t) = \nu(t)\cos\theta_W(t) \quad (1)$$

$$\dot{y}_W(t) = \nu(t)\sin\theta_W(t) \quad (2)$$

$$\dot{\theta}_W(t) = \omega(t) \quad (3)$$

where $\nu(t)$ and $\omega(t)$ are the translational and rotational velocity. If the cycle time T between two localization operations of the robot is small enough and both $\nu(t)$ and $\omega(t)$ can be considered constant in time ($\nu(t) = \nu(kT) = \nu(k)$ and $\omega(t) = \omega(kT) = \omega(k)$), it is possible to compute the incremental encoder changes by numerical integration of the previous unicycle equations, i.e.

$$x_W(k+1) = x_W(k) + T\nu(k)\cos\left(\theta_W(k) + \frac{\omega(k)T}{2}\right) \quad (4)$$

$$y_W(k+1) = y_W(k) + T\nu(k)\sin\left(\theta_W(k) + \frac{\omega(k)T}{2}\right) \quad (5)$$

$$\theta_W(k+1) = \theta_W(k) + \omega(k)T \quad (6)$$

Previous equations alone cannot be used to localize the robot over long distance paths because the errors related to odometry measurements accumulate over time as integration errors. On the other hand over short travel distances we can expect small errors and therefore encoder data can be used over short segments of the entire path starting from a position estimated by an algorithm that fuses encoder and sonar data. The motion equations with respect to the reference W are:

$$x_W(k+1) = x_W(k) + T\nu(k)\cos\left(\theta_W(k) + \frac{\omega(k)T}{2}\right) + v_x(k+1) \quad (7)$$

$$y_W(k+1) = y_W(k) + T\nu(k)\sin\left(\theta_W(k) + \frac{\omega(k)T}{2}\right) + v_y(k+1) \quad (8)$$

$$\theta_W(k+1) = \theta_W(k) + \omega(k)T + v_\theta(k+1) \quad (9)$$

where the sequence $v(k)$ is a zero-mean, white, Gaussian process noise ($v(k) = [v_x(k), v_y(k), v_\theta(k)]^T$) satisfying the following relation:

$$E\{v(k)v^T(k)\} = \delta_{kk'}Q(k, \Delta s) \quad (10)$$

i.e. there is no correlation between the error at different times. The covariance matrix $Q(k, \Delta s)$ of course depends on the time and also on the distance Δs between the two positions at two consecutive steps $(x_W(k+1), y_W(k+1))$ and $(x_W(k), y_W(k))$.

2.2 The Extended Kalman Filter

Denote the 16-dimensional vector containing sonar data at the time k by $Z(k)$. These data depend both on the robot configuration and of course on the environment. One obtains:

$$Z(k) = h(k, X_W(k)) + w(k) \quad (11)$$

where $w(k)$ (the sonar error) is assumed a zero-mean, white, Gaussian measurement noise with covariance matrix $R(k)$.

In addition, denote with $X_W(k|k)$ the state configuration of the robot estimated at the time k by using the first k vector sonar data.

An algorithm Λ providing sequentially optimal configuration estimates of the robot is looked for:

$$\begin{aligned} X_W(k+1|k+1) &= \\ &= \Lambda(X_W(k|k), Z(k+1)) \end{aligned} \quad (12)$$

by the fusion of the above motion equations (encoder data) and sonar data. The optimality is in the sense of the minimum mean-square error. One cycle of the estimation algorithm will therefore consist in mapping the estimate $X_W(k|k)$ and the associated state error covariance matrix $P(k|k) = E\{X_W(k|k)X_W(k|k)^T(k)\}$ into the corresponding variables at the next stage $X_W(k+1|k+1)$ and $P(k+1|k)$. In order to achieve this result an EKF is used.

2.2.1. Prediction Phase Starting from the configuration $X_W(k|k)$ estimated at time k , and using encoder data, the configuration at the time $(k+1)$, $X_W(k+1|k)$, is estimated.

$$\begin{aligned} x_W(k+1|k) &= x_W(k|k) + \\ &+ T\nu(k)\cos\left(\theta_W(k|k)\frac{\omega(k)T}{2}\right) \end{aligned} \quad (13)$$

$$\begin{aligned} y_W(k+1|k+1) &= y_W(k|k) + \\ &+ T\nu(k)\sin\left(\theta_W(k|k)\frac{\omega(k)T}{2}\right) \end{aligned} \quad (14)$$

$$\theta_W(k+1|k+1) = \theta_W(k|k) + \omega(k)T. \quad (15)$$

The associated state error covariance matrix is

$$P(k+1|k) =$$

$$= F(k)P(k|k)F(k)^T + Q(k, l) \quad (16)$$

where $F(k)$ is the jacobian of the motion equations (7)-(9) (whose vector form scripture is $X_W(k+1) = f(k, X_W(k)) + \nu(k+1)$) with respect to the robot configuration, i.e.

$$F(k) = [\nabla_X f^T(k, X)]_{X=X_W(k|k)} \quad (17)$$

From the *localization map* (discussed in Section 3) and the previous predicted configuration of the robot $X_W(k+1|k)$ it is possible to predict the sonar measurement at time $(k+1)$, $\hat{Z}(k+1)$.

2.2.2. Estimation Phase Subtracting the previous estimated measurements from the sonar data $Z(k+1)$ we obtain the vector

$$\eta(k+1) = Z(k+1) - \hat{Z}(k+1) \quad (18)$$

whose covariance is

$$\begin{aligned} S(k+1) &= H(k+1)P(k+1|k)H(k+1)^T + \\ &+ R(k+1) \end{aligned} \quad (19)$$

where $H(k)$ is the jacobian of the function $h(k, X)$ with respect to the robot configuration, i.e.

$$\begin{aligned} H(k+1) &= \\ &= [\nabla_X h^T(k+1, X)]_{X=X_W(k+1|k)} \end{aligned} \quad (20)$$

The final step consists in the computation of the Kalman gain $K(k+1)$ in order to compute $X_W(k+1|k+1)$ and its covariance $P(k+1|k+1)$. In our notation:

$$\begin{aligned} K(k+1) &= \\ &= P(k+1|k)H(k+1)^T S(k+1)^{-1} \end{aligned} \quad (21)$$

$$\begin{aligned} X_W(k+1|k+1) &= X_W(k+1|k) + \\ &+ K(k+1)\eta(k+1) \end{aligned} \quad (22)$$

$$\begin{aligned} P(k+1|k+1) &= \\ &= [I - K(k+1)H(k+1)^T]P(k+1|k) \end{aligned} \quad (23)$$

3. LOCALIZATION AND OBSTACLE MAP

In order to taking into account partial variability of the environment where the robot moves, two kinds of map are used. The localization process is carried out by using always the localization map independent of the time. The motion planning is realized using another map, the obstacles map, which is updated during the robot navigation.

The localization map consists of segments representing the main walls in our environment. It is

used during the navigation only to compute the predicted sonar values (i.e. the function h in the EKF) and its jacobian (the matrix H). Given the configuration X of the robot, the expected sensor value for the i_{th} sector is the minimum distance ray intersecting a wall contained in the i_{th} sensor beamwidth. Let s_{x_i} and s_{y_i} be the coordinates of the i_{th} sonar and x_i and y_i the coordinates of the minimum distance intersection for sonar i_{th} at a given moment. The predicted sonar value is

$$h_i(X, k) = \sqrt{(s_{x_i} - x_i)^2 + (s_{y_i} - y_i)^2} \quad (24)$$

The motion planning is obtained during navigation by using another map, that can be changed while the robot moves. This new map is in the space of the configurations of the robot.

In the implementation of the EKF a threshold is introduced, dependent on the distance covered by the robot during one cycle time in such a way that all values of the sonar data outside the threshold are considered incorrect and ignored in the localization process. On the other hand when these measured values are smaller than the predicted ones they are used in order to change *on line* the obstacle map. This map is obtained by a cell decomposition (Latombe, 1991) of the configuration space whose resolution is variable depending on the obstacle dimension. In any case cells smaller than $0.04m \times 0.04m$ are not considered. Once the decomposition has been realized, each cell is associated with a node of a graph. Two nodes in the graph are connected by an arc if the corresponding cells have a common border. The weight of the arc has been set equal to the distance between the centers of the two cells. Once the initial and final position are assigned, the nodes corresponding to the cells containing the initial and the final position are determined and the path is computed through a minimum path search in the graph, using the Dijkstra algorithm. Once the sequence of nodes has been found, the robot moves along straight lines connecting the center of the cells corresponding to the nodes of the sequence. When a change in the map occurs (as discussed before) during robot motion and the update regards a cell contained in the path planned for the robot, Dijkstra algorithm is again called.

4. MOTION CONTROLLER

Several kinds of motion control strategies have been considered. The first one is very simple and was the starting point in our research in order to build a robust localization algorithm. The trajectory consists of several points (one per

meter) and the robot moves from one point to another. The motion is decomposed in rotation and translation. In this case the localization is done when the robot is stopped.

A more advanced strategy was then developed where translation and rotation occur simultaneously and also the localization occurs while the robot is moving.

Denote by $[x_d(t), y_d(t)]$ the desired trajectory and by $x_{err}(t)$ and $y_{err}(t)$ the distance in the x axis and y axis respectively between the desired position and the estimated one by the extended kalman filter at the time t . We adopted the following controller (Pappas and Kyriakopoulos, 1993)

$$\nu = -k(x_e \cos \theta_w + y_e \sin \theta_w) + \dot{x}_d \cos \theta_w + \dot{y}_d \sin \theta_w \quad (25)$$

$$\omega = \frac{1}{\epsilon} [-k(y_e \cos \theta_w - x_e \sin \theta_w) + \dot{x}_d \sin \theta_w + \dot{y}_d \cos \theta_w] \quad (26)$$

Finally, a very simple and efficient controller is used, which does not require any constraint on the time, contrary to the previous controller. The translational velocity is, in this case, always the same during the robot motion. The rotational velocity is chosen in order to give to the robot after one cycle time T , a direction that is a weighted average of the direction needed to bring the robot towards the desired trajectory and the direction tangential to the trajectory in the nearest point from the estimated position.

$$\omega T = \frac{l(\alpha + \frac{\pi}{2}) + L\alpha}{l + L} \quad (27)$$

The parameter L characterizes the distance that the robot travels from the desired trajectory. If $L = 0$ the robot moves towards the nearest point of the desired trajectory. If L is very large, the robot moves along a direction parallel to the line tangent to the trajectory in the nearest point of it. The other parameters appearing in equation (27) are explained in the Fig. 1.

5. EXPERIMENTAL RESULTS

5.1 The Robot

The experiments described in the following, have been carried out on the Nomad150 platform.

5.2 Sonar Sensors

The sonar ranging system consists of 16 independent channels. This system can give range

Fig. 1. The performed control strategy. The translational velocity does not change during the motion. The assigned direction after one cycle time T is a weighted average of the direction needed to bring the robot towards the desired trajectory and the direction tangential to the trajectory in the nearest point from the estimated position.

information from $0.4m$ to $6.5m$ with 1% accuracy over the entire range. It measures the time of flight ranging sensor based upon the return time of an acoustic signal. Each sonar sensor has a beamwidth of approximately 22.5° .

5.3 Results

The experiments were carried out using a Nomad150 mobile robot in the hallways of the Computer Science, System and Production department of the University of Rome Tor Vergata (Fig 2). From this figure it is possible to see that the path from the initial to the final position is chosen in order to minimize the distance covered. However, it is very easy to change the proposed algorithm in order to minimize some other cost function taking into account not only the distance covered by the robot but also the distance of the desired trajectory from the obstacles. For example, in the experiments, also a different measure is used, to minimize a cost function that increases with the distance covered by the robot but decreases with the size of the cells where the robot moves. In other words one can privilege in this way the obstacle avoidance with respect to the distance covered by the robot depending on the particular weight given to the size of the cell in the cost function adopted in the Dijkstra algorithm.

The robot was able to move with a translational velocity of about $0.3ms^{-1}$. The total cycle time for

Fig. 3. The error in the position of the robot during navigation estimated with odometry data alone (dotted line) and by the algorithm of localization implemented (solid line). The translational velocity was $0.3ms^{-1}$

localization (the most of that was due to modem communication) ranged in the interval $(0.7-0.8)s$.

In Fig. 3 odometry and the estimated position against the actual position are compared. These data refer to the case when the localization is performed when the robot is moving. The translational velocity is $0.3ms^{-1}$. One may conclude that the proposed algorithm is able to remove the integration error of the encoder.

The accuracy of the localization process strongly depends on the translational velocity, since this process occurs while the robot is in motion. In

Fig. 2. The map of the environment with the path of the robot. The path is chosen in this case in order to minimize the distance covered by the robot using the Dijkstra algorithm.

particular, it has been found that the error related to a motion with a speed of $0.1ms^{-1}$ is of the order of an half of that one showed in the figure. In particular this same error is obtained by using the first very simple controller discussed in section 4 where the localization is carried out when the robot is not moving.

REFERENCES

- Alag S., Goebel K. and Agogino A. (1995). A methodology for intelligent sensor validation and fusion used in tracking and avoidance of objects for automated vehicles. In: *ACC '95*. Seattle, WA.
- Anousaki and Kyriakopoulos (1999). Simultaneous localization and map building for mobile robot navigation. *IEEE Transactions on Robotics* pp. 42–53.
- Durrant White H., Pagac D. and Nebot E.M. (1998). An evidential approach to map building for autonomous vehicles. *IEEE Transactions on Robotics and Automation* **14**, 623–629.
- Jetto L., Longhi S., Venturini G. (1998). Development and experimental validation of an adaptive extended. In: *WODES '98, the International Workshop on Discrete Event Systems*. Cagliari, Italy. pp. 373–378.
- Latombe, J.C. (1991). *Robot Motion Planning*. Kluwer Academic Publishers.
- Luo and Kay (1989). Multisensor integration and fusion in intelligent systems. *IEEE Trans. Systems, man and Cybernetics* **19**, 901–931.
- Pappas and Kyriakopoulos (1993). Dynamic modelling and tracking control of non-holonomic wheeled vehicles. In: *IFAC '93, World Congress*.