Optimization of the Simultaneous Localization and Map-Building Algorithm for Real-Time Implementation

José E. Guivant and Eduardo Mario Nebot, *Senior Member, IEEE*

***Abstract*—This paper addresses real-time implementation of the simultaneous localization and map-building (SLAM) algorithm. It presents optimal algorithms that consider the special form of the matrices and a new compressed filter that can significantly reduce the computation requirements when working in local areas or with high frequency external sensors. It is shown that by extending the standard Kalman filter models the information gained in a local area can be maintained with a cost $\sim O(N_a^2)$, where N_a is the number of landmarks in the local area, and then transferred to the overall map in only one iteration at full SLAM computational cost. Additional simplifications are also presented that are very close to optimal when an appropriate map representation is used. Finally the algorithms are validated with experimental results obtained with a standard vehicle running in a completely unstructured outdoor environment.**

***Index Terms*—Autonomous vehicles, Kalman filter, map building, navigation.**

I. Introduction

Reliable localization is an essential component of any autonomous vehicle system. The basic navigation loop is based on dead reckoning sensors that predict high-frequency vehicle manoeuvres and low-frequency absolute sensors that bound positioning errors [1]. The problem of localization given a map of the environment or estimating the map knowing the vehicle position has been addressed and solved using a number of different approaches [2]–[5]. A related problem is when both the map and the vehicle position are not known. In this case, vehicle and map estimates are highly correlated and cannot be obtained independently of one another [6]. This problem is usually known as simultaneous localization and map building (SLAM) and was originally introduced in [7], [8]. During the past three years, significant progress has been made toward the solution of the SLAM problem. A number of different approaches have been presented to address this problem. In [9] and [10] a probabilistic approach is presented to solve the localization problem or the map-building problem when the map or position of the vehicle, respectively, is known. This approach is based on the approximation of the probability density functions with samples, also called particles. This idea was originally introduced in [11] as a bootstrap filter but has been more commonly known as the particle filter. In this case, no assumption needs to be made on the particular model and sensor distributions. The algorithm is suitable for handling multimodal distribution. This makes possible to start the robot in a completely unknown position. At the same time it allows for the solution of the "kidnapped robot" problem, a robot that has been suddenly moved to another position without being told. This approach has been applied very successfully to a number of indoor navigation applications, in particular in [12]. Due to the high computation requirements this method has not been used for real-time SLAM at present, although work is in progress to overcome this limitation.

One of the most appealing approaches to solving the real-time localization problem is by modeling the environment and sensors and assuming that errors have a Gaussian distribution. Then very efficient algorithms, such as Kalman filters, can be used to solve this problem in a compact and elegant manner [13]. These algorithms require the mobile robot to always be localized within certain bounds, meaning that it is not possible to address the initialization or the "kidnapped robot" problem. This is not an issue for many industrial applications, [14]–[16], where large machines weighing many tonnes operate autonomously. In fact, in these applications the navigation system has to be designed with enough integrity in order to avoid, or at least recognize such faults and provide for appropriate safety procedures, [1]. For these applications the Kalman filter with Gaussian assumptions is the preferred approach to achieve the degree of integrity required in such environments.

Kalman filter methods can also be extended to perform SLAM. There have been several applications of this technology in a number of different environments, such as indoors [17], [18], underwater [19], [20], and outdoors [21], [22]. One of the main problems with the SLAM algorithm has been the computational requirements. It is well known that the complexity of the SLAM algorithm can be reduced to $\sim O(N^2)$ [8], N being the number of landmarks in the map. For long duration missions, the number of landmarks will increase and, eventually, computer resources will not be sufficient to update the map in real-time. This scaling problem arises because each landmark is correlated to all other landmarks. The correlation appears since the observation of a new landmark is obtained with a sensor mounted on the mobile robot and thus the landmark location error will be correlated with the error in the vehicle location and the errors in other landmarks of the

Manuscript received December 11, 2000. This paper was recommended for publication by Associate Editor Y. Xu and Editor S. Hutchinson upon evaluation of the reviewers' comments.

J. E. Guivant is with the Australian Centre for Field Robotics, University of Sydney, JO4, NSW 2006, Sydney, Australia (e-mail: jguivant @acfr.usyd.edu.au).

E. M. Nebot is with the Australian Centre for Field Robotics. University of Sydney, JO4, NSW 2006, Sydney, Australia and the CRC Centre for Mining Technology and Equipment (CMTE) (e-mail: nebot@acfr.usyd.edu.au).

Publisher Item Identifier S 1042-296X(01)06733-7.

map. This correlation is of fundamental importance for the long-term convergence of the algorithm [6], and needs to be maintained for the full duration of the mission. Leonard *et al.* [19], addressed the computational issues by splitting the global map into a number of submaps, each with their own vehicle track. They present an approximation technique to address the update of the covariance in the transition between maps. Although they present impressive experimental results there is no proof of the consistency of the approach or estimation of the conservatism of the covariance over-bounding strategy.

This paper addresses real-time implementation of SLAM with a set of optimal algorithms that significantly reduce the computational requirement without introducing any penalties in the accuracy of the results. A compressed algorithm is presented to store and maintain all the information gathered in a local area with a cost proportional to the square of the number of landmarks in the area. This information can then be transferred to the rest of the global map with a cost that is similar to full SLAM but in only one iteration. These results are demonstrated theoretically and with experimental results. Finally, suboptimal simplifications are presented to update the covariance matrix of the states. With this approach the total computational cost of the algorithm can be made proportional to N. It is also shown that by using a relative map representation the algorithm become very close to optimal. The convergence and accuracy of the algorithms are tested in large outdoor environments with more than 500 states.

The paper is organized as follows. Section II presents the basic modeling background required to introduce the algorithms. Section III presents the optimization of SLAM in the prediction and update stages. In particular the compression algorithm is presented with further details given in the Appendix. Section IV introduces the SLAM simplification and proofs of the consistency of the algorithm. It also presents the relative map representation used to make the algorithm proposed very close to optimal. Experimental results in unstructured outdoor environments are presented in Section V. Finally Section VI presents the conclusions with proposed future research areas.

II. SLAM

When absolute position information is not available it is still possible to navigate with small errors for long periods of time. The SLAM algorithm use dead reckoning and relative observation to estimate the position of the vehicle and to build and maintain a navigation map. The mobile robot is equipped with dead reckoning capabilities and an external sensor capable of measuring relative distance of the vehicle to the environment as shown in Fig. 1. The steering control α, and the speed ν_c are used with the kinematic model to predict the position of the vehicle. In this case, the external sensor returns the range and bearing information to the different features $B_{i(i=1\ldots n)}$. This information is obtained with respect to the vehicle coordinates (x_l, y_l), that is $z(k) = (r, \beta)$, where r is the distance from the beacon to the range sensor and β is the sensor bearing measured with respect to the vehicle coordinate frame.

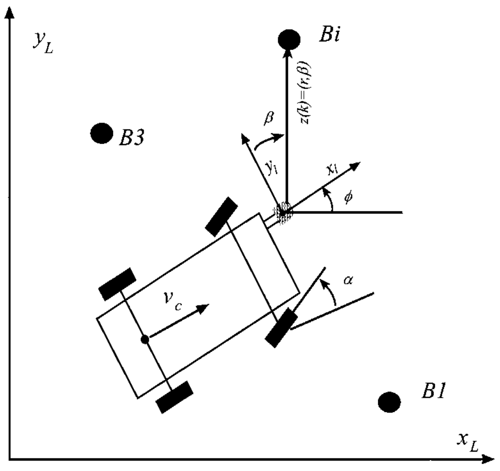

Fig. 1. Vehicle coordinate system.

Considering that the vehicle is controlled through a demanded velocity ν_c and steering angle α the process model that predicts the trajectory of the center of the back axle is given by

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\phi}_c \end{bmatrix} = \begin{bmatrix} \nu_c \cdot \cos(\phi) \\ \nu_c \cdot \sin(\phi) \\ \frac{\nu_c}{L} \cdot \tan(\alpha) \end{bmatrix} + \gamma \tag{1}$$

where L is the distance between wheel axles and γ is noise as defined in (53). The observation equation relating the vehicle states to the observations is

$$\begin{aligned} z =& h(X, x_i, y_i) = \begin{bmatrix} z_r^i \\ z_\beta^i \end{bmatrix} \\ =& \begin{bmatrix} \sqrt{(x_i - x_L)^2 + (y_i - y_L)^2} \\ \text{atan}\left(\frac{(y_i - y_L)}{(x_i - x_L)}\right) - \phi_L + \frac{\pi}{2} \end{bmatrix} + \gamma_h \end{aligned} \tag{2}$$

where

z	observation vector;
(x_i, y_i)	coordinates of the landmarks;
x_L, y_L and ϕ_L	vehicle states defined at the external sensor location; and
γ_h	noise as defined in (53).

In the case where multiple observation are obtained the observation vector will have the form

$$Z = \begin{bmatrix} z^1 \\ \vdots \\ z^m \end{bmatrix}. \tag{3}$$

The extended Kalman filter (EKF) equations to solve this estimation problem are presented in Appendix A. In this section, we present the extension of the models to address the SLAM problem.

Under this framework the vehicle starts at an unknown position with given uncertainty and obtains measurements of the environment relative to its location. This information is used to incrementally build and maintain a navigation map and to localize the vehicle with respect to this map. The system will detect new features at the beginning of the mission and when the vehicle explores new areas. Once these features become reliable and stable they are incorporated into the map becoming part of the state vector.

The state vector is now given by

$$X = \begin{bmatrix} X_L \\ X_i \end{bmatrix}$$
$$X_L = (x_L, y_L, \phi_L)^{\mathrm{T}} \in R^3$$
$$X_i = (x_1, y_1, \ldots, x_N, y_N)^{\mathrm{T}} \in R^{2N} \quad (4)$$

where $(x, y, \phi)_L$ and $(x, y)_i$ are the states of the vehicle and features incorporated into the map, respectively. Since this environment is considered to be static, the dynamic model that includes the new states becomes

$$X_L(k+1) = f(X_L(k)) + \gamma$$
$$X_i(k+1) = X_i(k). \quad (5)$$

It is important to remark that the landmarks are assumed to be static. If this is so, then the Jacobian matrix for the extended system is

$$\frac{\partial F}{\partial X} = \begin{bmatrix} \frac{\partial f}{\partial \tilde{x}_L} & \varnothing \\ \varnothing^{\mathrm{T}} & I \end{bmatrix} = \begin{bmatrix} J_1 & \varnothing \\ \varnothing^{\mathrm{T}} & I \end{bmatrix}$$
$$J_1 \in R^{3\times 3}, \ \varnothing \in R^{3\times N}, \ I \in R^{2N\times 2N}. \quad (6)$$

The observations z_r and z_β are obtained from a range and bearing sensor relative to the vehicle position and orientation. The observation equation given in (2) is a function of the states of the vehicle and the states representing the position of the landmark. The Jacobian matrix of the vector h with respect to the variables $(x_L, y_L, \phi_L, x_i, y_i)$ can be evaluated using:

$$\frac{\partial h}{\partial X} = \begin{bmatrix} \frac{\partial z_r}{\partial X} \\ \frac{\partial z_\beta}{\partial X} \end{bmatrix} = \begin{bmatrix} \frac{\partial r_i}{\partial(x_L, y_L, \phi_L, \{x_i, y_i\}_{i=1\ldots N})} \\ \frac{\partial \beta_i}{\partial(x_L, y_L, \phi_L, \{x_i, y_i\}_{i=1\ldots N})} \end{bmatrix}. \quad (7)$$

This Jacobian will always have a large number of null elements since only a few landmarks will be observed and validated at a given time. For example, when only one feature is observed the Jacobian has the following form:

$$\begin{bmatrix} \frac{\partial z_r}{\partial X} \\ \frac{\partial z_\beta}{\partial X} \end{bmatrix} = \begin{bmatrix} \frac{\Delta x}{\Delta} & \frac{\Delta y}{\Delta} & 0 & 0 & \ldots & -\frac{\Delta x}{\Delta} & -\frac{\Delta y}{\Delta} & 0 & \ldots & 0 \\ -\frac{\Delta y}{\Delta^2} & \frac{\Delta x}{\Delta^2} & -1 & 0 & \ldots & \frac{\Delta y}{\Delta^2} & -\frac{\Delta x}{\Delta^2} & 0 & \ldots & 0 \end{bmatrix} \quad (8)$$

where

$$\Delta x = (x_L - x_i), \ \Delta y = (y_L - y_i), \ \Delta = \sqrt{(\Delta x)^2 + (\Delta y)^2}.$$

These models can then be used with a standard EKF algorithm to build and maintain a navigation map of the environment and to track the position of the vehicle.

III. Optimization of SLAM

Under the SLAM framework the size of the state vector is equal to the number of vehicle states plus twice the number of landmarks, that is $2N + 3 = M$. This is valid when working with point landmarks in 2-D environments. In most SLAM applications the number of vehicle states will be insignificant with respect to the number of landmarks. The number of landmarks will grow with the area of operation making the standard filter computation impracticable for on-line applications.

In this paper, we present a series of optimizations in the prediction and update stages that reduce the complexity of the SLAM algorithm from $\sim O(M^3)$ to $\sim O(M^2)$. Then a compressed filter is presented to reduce the real-time computation requirement to $\sim O(2N_a^2)$, with N_a being the number of landmarks detected in the local area. This will also make the SLAM algorithm extremely efficient while the vehicle remains navigating in this area since the computation complexity becomes independent of the size of the global map. These algorithms do not make any approximations and the results are identical to a full SLAM implementation.

A. Standard Algorithm Optimization

1) Prediction Stage: Considering the zeros in the Jacobian matrix of (6) the prediction (55) can be written

$$J \cdot P \cdot J^{\mathrm{T}} + Q = \begin{bmatrix} J_1 & \varnothing \\ \varnothing^{\mathrm{T}} & I \end{bmatrix} \cdot \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} \cdot \begin{bmatrix} J_1^{\mathrm{T}} & \varnothing^{\mathrm{T}} \\ \varnothing & I^{\mathrm{T}} \end{bmatrix} + \begin{bmatrix} Q_V & \varnothing \\ \varnothing & \varnothing_2 \end{bmatrix}$$
$$J_1 \in R^{3\times 3}, \ \varnothing \in R^{3\times 2N}, \ I \in R^{2N\times 2N},$$
$$P_{11} \in R^{3\times 3}, \ P_{12} \in R^{3\times 2N},$$
$$P_{21} = P_{12}^{\mathrm{T}}, \ P_{22} \in R^{2N\times 2N}. \quad (9)$$

Performing the matrix operations explicitly the following result is obtained:

$$J \cdot P = \begin{bmatrix} J_1 & \varnothing \\ \varnothing^{\mathrm{T}} & I \end{bmatrix} \cdot \begin{bmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{bmatrix} = \begin{bmatrix} J_1 \cdot P_{11} & J_1 \cdot P_{12} \\ I \cdot P_{21} & I \cdot P_{22} \end{bmatrix} = \begin{bmatrix} J_1 \cdot P_{11} & J_1 \cdot P_{12} \\ P_{21} & P_{22} \end{bmatrix}$$
$$J \cdot P \cdot J^{\mathrm{T}} = \begin{bmatrix} J_1 \cdot P_{11} & J_1 \cdot P_{12} \\ P_{21} & P_{22} \end{bmatrix} \cdot \begin{bmatrix} J_1^{\mathrm{T}} & \varnothing^{\mathrm{T}} \\ \varnothing & I \end{bmatrix} = \begin{bmatrix} J_1 \cdot P_{11} \cdot J_1^{\mathrm{T}} & J_1 \cdot P_{12} \cdot I \\ P_{21} \cdot J_1^{\mathrm{T}} & P_{22} \cdot I \end{bmatrix} = \begin{bmatrix} J_1 \cdot P_{11} \cdot J_1^{\mathrm{T}} & J_1 \cdot P_{12} \\ (J_1 \cdot P_{21})^{\mathrm{T}} & P_{22} \end{bmatrix} \quad (10)$$

It can be proved that the evaluation of this matrix requires approximately only 9 M multiplications. In general, more than one prediction step is executed between two update steps. This is due to the fact that the prediction stage is usually driven by high frequency sensor information that acts as input to the dynamic model of the vehicle which needs to be evaluated in order to control the vehicle. The low frequency external sensors report the observation used in the estimation stage of the EKF. This information is processed at a much lower frequency. For example, the steering angle and wheel speed can be sampled every 20 ms, but the laser frames can be obtained with a sample time of 200 ms. In this case we have a ratio of approximately 10 prediction steps to one update step. The compact form for two prediction steps can be obtained using the results given in (10):

$$\begin{aligned} P(t+T_1+T_2,t) =& J(t+T_1,T_2) \\ & \cdot P(t+T_1,t)\cdot J^{\mathrm{T}}(t+T_1,T_2) \\ & + Q(t+T_1,T_2) \\ =& J(t+T_1,T_2)\cdot J(t,T_1)\cdot P(t,t) \\ & \cdot J^{\mathrm{T}}(t,T_1)\cdot J^{\mathrm{T}}(t+T_1,T_2) \\ & + Q(t+T_1,T_2) + J(t,T_1) \\ & \cdot Q(t,T_1)\cdot J^{\mathrm{T}}(t,T_1). \end{aligned} \tag{11}$$

By considering the special form of the matrix involved in SLAM the prediction equation can be rewritten

$$\begin{aligned} & J(k+1)\cdot\left(J(k)\cdot P(k,k)\cdot J^{\mathrm{T}}(k)+Q(k)\right)\cdot J^{\mathrm{T}}(k+1) \\ & \quad = \begin{bmatrix} P_{11}^{+2} & P_{12}^{+2} \\ P_{21}^{+2} & P_{22}^{+2} \end{bmatrix} \\ & P_{11}^{+2} = J_1(k+1)\cdot\left(J_1(k)\cdot P_{11}(k,k)\cdot J_1^{\mathrm{T}}(k)+Q_{11}(k)\right) \\ & \quad\quad \cdot J_1(k+1) \\ & P_{12}^{+2} = \left(P_{21}^{+2}\right)^{\mathrm{T}} = (J_1(k+1)\cdot J_1(k))\cdot P_{12}(k,k) \\ & P_{22}^{+2} = P_{22}. \end{aligned} \tag{12}$$

Finally, the prediction equation for two steps becomes

$$P(k+2,k) = \begin{bmatrix} P_{11}(k+2,k) & G_1\cdot P_{12}(k,k) \\ (G_1\cdot P_{12}(k,k))^{\mathrm{T}} & P_{22}(k,k) \end{bmatrix} \tag{13}$$

where

$$\begin{aligned} G_1 =& J_1(k+1)\cdot J_1(k) \\ P_{11}(k+2,k) =& J_1(k+1) \\ & \cdot\left(J_1(k)\cdot P_{11}(k,k)\cdot J_1^{\mathrm{T}}(k)+Q_{11}(k)\right) \\ & \cdot J_1^{\mathrm{T}}(k+1). \end{aligned} \tag{14}$$

From the previous considerations, in the case of n prediction steps without an update, the modified covariance matrix is

$$P(k+n,k) = \begin{bmatrix} P_{11}(k+n,k+n) & G_1\cdot P_{12}(k,k) \\ (G_1\cdot P_{12}(k,k))^{\mathrm{T}} & P_{22}(k,k) \end{bmatrix} \tag{15}$$

where

$$G_1 = G_1(k,n) = \prod_{i=0}^{n-1} J_1(k+i) = J_1(k+n-1)\cdots J_1(k) \tag{16}$$

For this vehicle model, the evaluation of G_1 requires n products of matrices of dimension 3×3. Considering that the major computational cost of the evaluation of this matrix is the calculation of P_{12} (or P_{21}), this simplification can substantially reduce the computation requirement in the prediction stage. For n prediction steps the complexity will be approximately $27n+9M$, that is smaller than the direct calculation. In this case M is the number of landmarks plus the number of vehicle states.

$$\frac{27\cdot n+9\cdot M}{9\cdot n\cdot M}\approx\frac{1}{n}, \quad M\gg n. \tag{17}$$

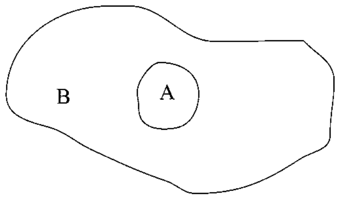

Fig. 2. Local and global areas.

In (15) P_{11} needs to be evaluated for every prediction step since the quality of the estimated position is required all the time. P_{22} remains constant between updates. The calculation of P_{12} (or P_{21}) is evaluated only before the estimation procedure using (15).

2) Update Stage: Since only a few features associated with the state vector are observed at a given time, the matrix H will have a large number of zeros. When only one feature is incorporated into the observation vector we have

$$\begin{aligned} H =& H(k) = \left.\frac{\partial h}{\partial X}\right|_{X=X(k)} = [H_1, \varnothing_1, H_2, \varnothing_2] \in R^{2\times M} \\ H_1 =& \left.\frac{\partial h}{\partial X_L}\right|_{X=X(k)} = \left.\frac{\partial h}{\partial(x_L,y_L,\phi_L)}\right|_{X=X(k)} \in R^{2\times 3} \\ H_2 =& \left.\frac{\partial h}{\partial X_i}\right|_{X=X(k)} = \left.\frac{\partial h}{\partial(x_i,y_i)}\right|_{X=X(k)} \in R^{2\times 2} \\ & \varnothing_1,\varnothing_2 = \text{null matrices } \left(\frac{\partial h}{\partial X_j} = \varnothing \quad \forall j\neq i\right). \end{aligned} \tag{18}$$

At a give time k the Kalman gain matrix W requires the evaluation of PH^{T}

$$P\cdot H^{\mathrm{T}} = P_1\cdot H_1^{\mathrm{T}} + P_2\cdot H_2^{\mathrm{T}} \quad P_1\in R^{M\times 3},\ P_2\in R^{M\times 2}. \tag{19}$$

It can be proved that the evaluation will require 10 M multiplications. Using the previous result, the matrix S and W can be evaluated with a cost of approximately 20 M

$$\begin{aligned} S =& H\cdot P\cdot H^{\mathrm{T}} + R \in R^{2*2} \\ W =& P\cdot H^{\mathrm{T}}\cdot S^{-1} \in R^{M\times 2} \end{aligned} \tag{20}$$

The cost of the state update operation is proportional to M. The main computational requirement is in the evaluation of the covariance update where complexity is $\sim O(M^2)$.

B. Compressed Filter

In this section we demonstrate that it is not necessary to perform a full SLAM update when working in a local area. This is a fundamental contribution because it reduces the computational requirement of the SLAM algorithm to the order of the number of features in the vicinity of the vehicle; independent of the size of the global map. A common scenario is to have a mobile robot moving in an area and observing features within this area. This situation is shown in Fig. 2 where the vehicle is operating in a local area A. The rest of the map is part of the global area B.

This approach will present significant advantages when the vehicle navigates for long periods of time in a local area or when the external information is available at high rate. Although high frequency external sensors are desirable to reduce position error growth, they also introduce a high computational cost in the SLAM algorithm. For example a laser sensor can return 2-D information at frequencies of 4–30 Hz. To incorporate this information the full SLAM algorithm will require to update M states at 30 Hz. In this work we show that, while working in a local area and observing local landmarks, we can preserve all the information processing a SLAM algorithm of the order of the number of landmarks in the local area. When the vehicle departs from this area, the information acquired can be propagated to the global landmarks without loss of information. This will also allow incorporating high frequency external information with very low computational costs. Another important implication is that the global map will not be required to update sequentially at the same rate as the local map.

1) Update Step: Consider the states divided in two groups.

$$X = \begin{bmatrix} X_A \\ X_B \end{bmatrix}, \quad \begin{matrix} X_A \in R^{2N_A+3} & X \in R^{2N-3} \\ X_B \in R^{2N_B} & N = N_A + N_B \end{matrix}$$

The states X_A can be initially selected as all the states representing landmarks in an area of a certain size surrounding the vehicle. The states representing the vehicle pose are also included in X_A. Assume that for a period of time the observations obtained are only related to the states X_A and do not involve states of X_B, that is

$$h(X) = h(X_A). \tag{21}$$

Then at a given time k

$$\begin{aligned} H &= \left.\frac{\partial h}{\partial X}\right|_{X=X(t)} = \left.\frac{\partial h}{\partial (X_A, X_B)}\right|_{X=X(k)} \\ &= \left[\frac{\partial h}{\partial X_A} \frac{\partial h}{\partial X_B}\right] = [H_a \quad 0]. \end{aligned} \tag{22}$$

Considering the zeros of the matrix H the Kalman gain matrix W is evaluated as follows:

$$\begin{aligned} P &= \begin{bmatrix} P_{aa} & P_{ab} \\ P_{ba} & P_{bb} \end{bmatrix} \\ P \cdot H^{\mathrm{T}} &= \begin{bmatrix} P_{aa} \cdot H_a^{\mathrm{T}} \\ P_{ba} \cdot H_a^{\mathrm{T}} \end{bmatrix} \\ H \cdot P \cdot H^{\mathrm{T}} &= H_a \cdot P_{aa} \cdot H_a^{\mathrm{T}} \\ S &= H_a \cdot P_{aa} \cdot H_a^{\mathrm{T}} + R \\ W &= P \cdot H^{\mathrm{T}} \cdot S^{-1} = \begin{bmatrix} P_{aa} \cdot H_a^{\mathrm{T}} \cdot S^{-1} \\ P_{ba} \cdot H_a^{\mathrm{T}} \cdot S^{-1} \end{bmatrix} = \begin{bmatrix} W_a \\ W_b \end{bmatrix}. \end{aligned} \tag{23}$$

From these equations, the following is possible.

- The Jacobian matrix H_a has no dependence on the states X_B.
- The innovation covariance matrix S and Kalman gain W_a are functions of P_{aa} and H_a. They do not have any dependencies on P_{bb}, P_{ab}, P_{ba} and X_b.

The update term dP of the covariance matrix can then be evaluated

$$\begin{aligned} dP &= W \cdot S \cdot W^{\mathrm{T}} = \begin{bmatrix} P_{aa} \cdot \kappa \cdot P_{aa} & \xi \cdot P_{ab} \\ (\xi \cdot P_{ab})^{\mathrm{T}} & P_{ba} \cdot \kappa \cdot P_{ab} \end{bmatrix} \text{with} \\ \kappa &= H_a^{\mathrm{T}} \cdot S^{-1} \cdot H_a \text{ and } \xi = P_{aa} \cdot \kappa \cdot \end{aligned} \tag{24}$$

In the previous demonstration the time subindexes were neglected for clarity of the presentation. These indexes are now incorporated to present the recursive update equations. The covariance matrix after one update is

$$\begin{aligned} P(k+1,k+1) &= P(k+1,k) - dP(k+1,k) \\ P_{aa}(k+1,k+1) &= P_{aa}(k+1,k) - P_{aa}(k+1,k) \\ &\quad \cdot \kappa(k+1) \cdot P_{aa}(k+1,k) \\ P_{ab}(k+1,k+1) &= P_{ab}(k+1,k) - \xi(k+1) \cdot P_{ab}^{\mathrm{T}}(k+1,k) \\ &= (I - \xi(k+1)) \cdot P_{ab}(k+1,k) \\ P_{bb}(k+1,k+1) &= P_{bb}(k+1,k) - P_{ba}(k+1,k) \\ &\quad \cdot \kappa(k+1) \cdot P_{ab}(k+1,k). \end{aligned} \tag{25}$$

And the covariance variation after t consecutive updates

$$\begin{aligned} P_{ab}(k+t,k+t) &= \Phi(k+t-1) \cdot P_{ab}(k,k) \\ P_{bb}(k+t,k+t) &= P_{bb}(k,k) - P_{ba}(k,k) \\ &\quad \cdot \psi(k-1) \cdot P_{ab}(k,k) \end{aligned} \tag{26}$$

with

$$\begin{aligned} \Phi(k+t) &= (I - \xi(k+t)) \cdot (I - \xi(k+t-1)) \\ &\quad \dots (I - \xi(k)) \\ \psi(k+t-1) &= \sum_{i=k}^{k+t-1} \left(\Phi^{\mathrm{T}}(i-1) \cdot \kappa(i) \cdots \Phi(i-1)\right) \end{aligned} \tag{27}$$

The evaluation of the matrices $\Phi(k)$, $\psi(k)$ can be done recursively according to

$$\begin{aligned} \Phi(k) &= (I - \xi(k)) \cdot \Phi(k-1) \\ \psi(k) &= \psi(k-1) + \Phi^{\mathrm{T}}(k-1) \cdot \kappa(k) \cdot \Phi(k-1) \\ &\text{with } \Phi(k), \psi(k), \kappa(k), \xi(k) \in R^{2Na \times 2Na}. \end{aligned} \tag{28}$$

During long-term navigation missions, the number of states in X_a will be, in general much smaller than the total number of states in the global map, that is $N_a \ll N_b < M$. The matrices ξ_k and κ_k are sparse and the calculation of $\Phi(k)$ and $\psi(k)$ has complexity $\sim O(N_a^2)$.

It is noteworthy that X_b, P_{ab}, P_{ba} and P_{bb} are not needed when the vehicle is navigating in a local region "looking" only at the state X_a. It is only required when the vehicle enters a new region. The evaluation of X_b, P_{bb}, P_{ab} and P_{ba} can then be done in one iteration with full SLAM computational cost using the compressed expressions.

The estimates X_b can be updated after t update steps using

$$\begin{aligned} X_b(k+t,k+t) &= X_b(k+t,k) - P_{ba}(k,k) \cdot \theta(k+t) \\ \text{with } \theta(k+t) &= \sum_{i=k}^{k+t-1} \Phi^{\mathrm{T}}(i-1) \cdot H_a^{\mathrm{T}}(i) \cdot S^{-1}(i) \cdot \vartheta(i) \end{aligned} \tag{29}$$

m the number of observations, in this case range and bearing,

$$\begin{aligned} \theta(k) &\in R^{2Na \times m}, \quad Z(k) \in R^m, \\ \Phi(k) &\in R^{2Na \times 2Na}, \\ H_a(k) &\in R^{m \times 2Na} \text{ and } S(k) \in R^{m \times m} \end{aligned}$$

Similarly, since H_a is a sparse matrix, the evaluation cost of the matrix θ is proportional to H_a. The derivation of these equations is presented in Appendix B.

2) Mixed Update and Prediction Steps Sequences: Similar results are obtained for sequences of interlaced prediction and update steps:

$$\begin{aligned}P(k+1,k) =& J(k+1,k)\cdot P(k-1,k-1)\\ &\cdot J^{\mathrm{T}}(k+1,k)+Q\\ J =& \begin{bmatrix} J_{aa} & J_{ab}\\ J_{ba} & J_{bb}\end{bmatrix} = \begin{bmatrix} J_{aa} & 0\\ 0 & I\end{bmatrix},\\ J_{aa} =& \begin{bmatrix} J_{LL} & 0\\ 0 & I\end{bmatrix} Q = \begin{bmatrix} Q_a\\ Q_b\end{bmatrix} = \begin{bmatrix} Q_a\\ 0\end{bmatrix},\\ Q_a =& \begin{bmatrix} Q_L\\ 0\end{bmatrix}\\ P_{aa}(k+1,k) =& J_{aa}(k+1,k)\cdot P_{aa}(k,k)\\ &\cdot J_{aa}^{\mathrm{T}}(k+1,k)+Q_L\\ P_{ab}(k+1,k) =& J_{aa}(k+1,k)\cdot P_{ab}(k,k)\\ P_{ba}(k+1,k) =& (P_{ab}(k+1,k))^{\mathrm{T}}\\ P_{bb}(k+1,k) =& P_{bb}(k,k).\end{aligned} \tag{30}$$

3) Extended Kalman Filter Formulation for the Compressed Filter: In order to maintain the information gathered in a local area it is necessary to extend the EKF formulation presented in (55)–(56). The following equations must be added in the prediction and update stage of the filter to be able to propagate the information to the global map once a full update is required:

$$\text{Prediction step}\begin{cases}\Phi(k) = J_{aa}(k,k-1)\cdot\Phi(k-1)\\ \psi(k)=\psi(k-1)\\ \Phi(0)=I\\ \psi(0)=I\end{cases}$$
$$\text{Update step}\begin{cases}\phi(k)=(I-\xi(k))\cdot\Phi(k-1)\\ \psi_k=\psi_{k-1}+\Phi^{\mathrm{T}}(k-1)\cdot\kappa(k)\cdot\Phi(k-1)\end{cases} \tag{31}$$

When a full update is required the global covariance matrix P and state X is updated with (26) and (29), respectively.

4) Computational Cost: The computational cost for each "compressed" update is evaluated for the case where three states are used to represent the pose of the vehicle and one landmark is observed.

$$\left.\begin{array}{c}\kappa = H_a^{\mathrm{T}}\cdot S^{-1}\cdot H_a\\ H_a=[H_\varepsilon\quad 0],\quad H_\varepsilon\in R^{2\times 5}\\ S\in R^{2\times 2}\end{array}\right\}\rightarrow \kappa\text{ cost }\approx(3+2)^3$$
$$\kappa=\begin{bmatrix}\kappa_{11} & 0\\ 0 & 0\end{bmatrix},\quad \kappa_{11}\in R^{5\times 5}$$
$$\xi = P_{aa}\cdot\kappa=[\xi_1\quad 0],\quad \xi_1\in R^{N_a\times 5}$$
$$\xi\text{ cost }\approx 25\cdot N_a$$
$$\Phi(k)=(I-\xi(k))\cdot\Phi(k-1)$$
$$\Phi\text{ cost }\approx 5\cdot N_a^2+25\cdot N_a$$
$$\psi(k)=\psi(k-1)+\Phi^{\mathrm{T}}(k-1)\cdot\kappa(k)\cdot\Phi(k-1)$$
$$\psi\text{ cost }\approx 5\cdot N_a^2+25\cdot N_a$$

Further details for more efficient implementation of this approach are given in Appendix C.

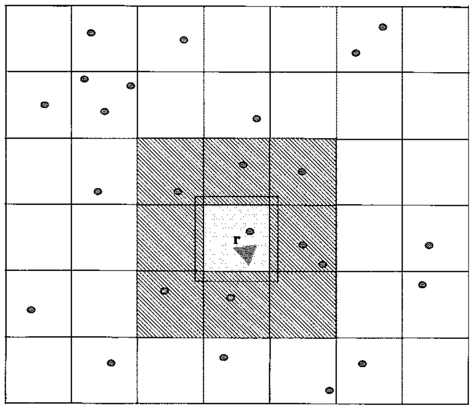

Fig. 3. Map Administration for the compressed algorithm.

The cost of the complete covariance error matrix evaluation (26) is approximately $N_a\cdot N_b^2$ according to

$$\begin{aligned}&\Phi,\psi \in R^{2N_a\times 2N_a},\ P_{ab}\in R^{2N_a\times 2N_b},\ P_{bb}\in R^{2N_b\times 2N_b}\\ &dP_{bb} =(P_{ba}\cdot\psi\cdot P_{ab})\sim O(N_a\cdot N_b^2+N_a^2\cdot N_b)\\ &dP_{ab} =(\Phi\cdot P_{ab})\sim O(N_a^2\cdot N_b)\\ &N_a \ll N_b < N.\end{aligned} \tag{32}$$

Provided that the vehicle remains for a period of time in a given area, the computational saving will be considerable. This has important implications since in many applications it will allow the exact implementation of SLAM in very large areas. This will be possible with the appropriate selection of local areas. The system evaluates the location of the vehicle and the landmark of the local map continuously at the cost of a local SLAM. Although a full update is required when the vehicle leaves the region, this update can be implemented as a parallel task. The only states that need to be fully updated are the new states in the new local area. A selective update can then be done only to those states while the full update for the rest of the map runs as a background task with lower priority. These results are important since it demonstrates that even in very large areas the computational limitation of SLAM can be overcame with the compression algorithm and appropriate selection of local areas.

C. Map Management

It has been demonstrated that while the vehicle operates in a local area all the information gathered can be maintained with a cost complexity proportional to the number of landmarks in this area. The next problem to address is the selection of local areas. One convenient approach consists of dividing the global map into rectangular regions with size at least equal to the range of the external sensor.

The proposed method is presented in Fig. 3. When the vehicle navigates in the region r the compressed filter includes in the group X_A the vehicle states and all the states related to landmarks that belong to region r and its eight neighboring regions. This implies that the local states belong to nine regions, each of size of the range of the external sensor. The vehicle will be able to navigate inside this region using the compressed filter. A full update will only be required when the vehicle leaves the central region r.

Every time the vehicle moves to a new region, the active state group X_A changes to those states that belong to the new region r and its adjacent regions. The active group always includes the vehicle states. In addition to the swapping of the X_A states, a global update is also required at full SLAM algorithm cost.

Each region has a list of landmarks that are known to be within its boundaries. Each time a new landmark is detected the region that owns it appends an index of the landmark definition to the list of owned landmarks. It is not critical if the landmark belongs to this region or a closely adjacent region. In case of strong updates, where the estimated position of the landmarks changes significantly, the owners of those landmarks can also be changed.

Hysteresis region is included bounding the local area r to avoid multiple map switching when the vehicle navigates in areas close to the boundaries between the region r and surrounding areas.

If the side lengths of the regions are smaller than the range of the external sensor, or if the hysteresis region is made too large, there is a chance of observing landmarks outside the defined local area. This observation will be discarded since they cannot be associated with any local landmarks. In such case the resulting filter will not be optimal since this information is not incorporated into the estimates. Although these marginal landmarks will not incorporate significant information since they are far from the vehicle, this situation can be easily avoided with appropriate selection of the size of the regions and hysteresis band.

Fig. 3 presents an example of the application of this approach. The vehicle is navigating in the central region r and if it never leaves this region the filter will maintain its position and the local map with a cost of a SLAM of the number of features in the local area formed by the nine neighboring regions.

IV. Suboptimal SLAM

A. Algorithm Description

In this section we present a series of simplifications that can further reduce the computationally complexity of SLAM. This suboptimal approach reduces the computational requirements by considering a subset of navigation landmarks present in the global map. It is demonstrated that this approach is conservative and consistent, and can generate close to optimal results when combined with the appropriate relative map representation.

Most of the computational requirements of the EKF are needed during the update process of the error covariance matrix. Once an observation is being validated and associated to a given landmark, the covariance error matrix of the states is updated according to

$$\begin{aligned} P &= P - \Delta P \\ \Delta P &= W \cdot S \cdot W^{\mathrm{T}}. \end{aligned} \tag{33}$$

The time subindexes are neglected when possible to simplify the equations. The state vector can be divided in two groups, the Preserved "P" and the Discarded "D" states

$$X = \begin{bmatrix} X_P \\ X_D \end{bmatrix}, \quad \begin{matrix} X_P \in R^{N_P} & X_D \in R^{N_D} \\ X \in R^N & N = N_P + N_D \end{matrix}. \tag{34}$$

With this partition it is possible to generate conservative estimates by updating the states X_D but not updating the covariance and cross-covariance matrices corresponding to this subvector. The covariance matrix can then be written in the following form:

$$\begin{aligned} P &= \begin{bmatrix} P_{PP} & P_{PD} \\ P_{DP}^{\mathrm{T}} & P_{DD} \end{bmatrix}, \\ \Delta P &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP}^{\mathrm{T}} & \Delta P_{DD} \end{bmatrix} = W \cdot S \cdot W^{\mathrm{T}}. \end{aligned} \tag{35}$$

Conservative updates are obtained if the nominal update matrix ΔP is replaced by the suboptimal ΔP^*

$$\begin{aligned} \Delta P^* &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP} & \varnothing \end{bmatrix} = \Delta P - \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \Delta P_{DD} \end{bmatrix}, \\ P^* &= P - \Delta P^* = P - \Delta P + \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \Delta P_{DD} \end{bmatrix} \end{aligned} \tag{36}$$

It can be shown that the simplification proposed generates consistent error covariance estimates.

Demonstration: The covariance error matrix $P^*(k+1)$ can be rewritten as follows:

$$P^*(k+1) = P(k) - \Delta P^* = P(k) - \Delta P + \delta \tag{37}$$

where

$$\begin{aligned} \Delta P^* &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP} & \varnothing \end{bmatrix} = \Delta P - \delta, \\ \Delta P &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP} & \Delta P_{DD} \end{bmatrix} \geq 0 \\ \delta &= \begin{bmatrix} \varnothing & \varnothing \\ \varnothing & \Delta P_{BB} \end{bmatrix} \geq 0. \end{aligned} \tag{38}$$

The matrices ΔP and μ are positive semidefinite since

$$\begin{aligned} \Delta P &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP}^{\mathrm{T}} & \Delta P_{DD} \end{bmatrix} = W \cdot S \cdot W^{\mathrm{T}} \geq 0 \text{ and} \\ \Delta P_{DD} &= W_D \cdot S_D \cdot W_D^{\mathrm{T}} \geq 0. \end{aligned} \tag{39}$$

As given in (37), the total update is formed by the optimal update plus an additional positive semidefinite noise matrix δ. the matrix δ will increase the covariance uncertainty

$$P^*(k+1) = P(k+1) + \delta \tag{40}$$

then the suboptimal update of P^* becomes more conservative than the full update

$$P^*(k+1) \leq P(k+1) \leq P(k). \tag{41}$$

Finally, the submatrices that need to be evaluated are P_{PP}, P_{PD}, and P_{DP}. The significance of this result is that P_{DD} is not evaluated. In general, this matrix will be of high order since it includes most of the landmarks.

The fundamental problem becomes the selection of the partition P and D of the state vector. The diagonal of matrix ΔP can be evaluated on-line with low computational cost. By inspecting the diagonal elements of ΔP we can see that many terms are very small compared to the corresponding previous covariance value in the matrix P. This indicates that the new observation does not provide a significant information contribution to this

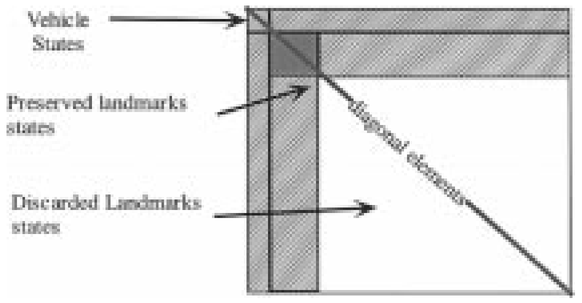

Fig. 4. Full covariance matrix divided into the covariance blocks corresponding to the Vehicle and Preserved landmarks states (X_P) and Discarded landmarks states (X_D). The cross-correlation covariance between the Preserved and Discarded states are fully updated as shown in grey. Finally, the cross correlation between the elements of the states corresponding to the "Discarded landmarks" are not updated as shown in white.

particular state. This is an indication to select a particular state as belonging to the subset D.

The other criterion used is based on the value of the actual covariance of the state. If it is below a given threshold, it can be a candidate for the subvector D.

In many practical situations a large number of landmarks can usually be associated to the subvector D. This will introduce significant computational savings since P_{DD} can potentially become larger than P_{PP}. The cross correlation P_{PD} and P_{DP} are still maintained but are in general of lower order as can be appreciated in Fig. 4.

Finally, the selection criteria to obtain the partition of the state vector can be given with the union of the following I_i sets:

$$\begin{aligned} I_1 =& \{i\colon \Delta P(i,i) < c_1 \cdot P(i,i)\} \\ I_2 =& \{i\colon P(i,i) < c_2\} \quad I = I_1 \cup I_2. \end{aligned} \tag{42}$$

Then ΔP^* is evaluated as follows:

$$\begin{aligned} \Delta P^*(i,j) =& 0 \quad \forall i,j\colon i \in I \text{ and } j \in I \\ \Delta P^*(i,j) =& \Delta P(i,j) \quad \forall i,j\colon i \notin I \text{ or } j \notin I. \end{aligned} \tag{43}$$

The error covariance matrix is updated with the simplified matrix ΔP

$$P^*(k+1,k+1) = P(k+1,k) - \Delta P^*. \tag{44}$$

The practical meaning of the set I_1, is that with the appropriate selection of c_1 negligible small updates of covariances can be ignored. As mentioned before, the selection of I_1 requires the evaluation of the diagonal elements of the matrix ΔP. The evaluation of the $\Delta P(i,i)$ elements requires a number of operations proportional to the number of states instead of the quadratic relation required for the evaluation of the complete matrix ΔP.

The second subset defined by I_2 is related to the states whose covariances are small enough to be considered practically zero. In the case of natural landmarks they become almost equivalent to beacons at known positions. The number of elements in the set I_2 will increase with time and can eventually make the computational requirements of SLAM algorithms comparable to the standard beacon localization algorithms.

Finally, the magnitude of the computation saving factor depends on the size of the set I. With appropriate exploration policies and real-time mission planning, the computation requirements can be maintained within the bounds of the on-board resources.

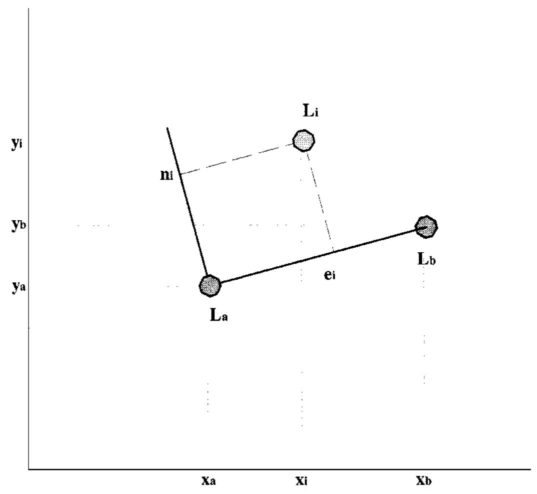

Fig. 5. Local reference frame. The reference frame is formed with two landmarks. The observations are then obtained relative to this frame.

B. Relative Map Representation

The suboptimal approach presented becomes less conservative when the cross correlation between the non relevant landmarks becomes small. This is very unlikely if an absolute reference frame is used, that is when the vehicle, landmarks and observation are represented with respect to a single reference frame. The cross correlations between landmarks of different regions can be substantially reduced by using a number of different bases and making the observation relative to those bases. With this representation, the map becomes grouped into five constellations. Each constellation has an associated frame based on two landmarks that belong to this constellation. The landmarks forming the bases are selected as a function of the range of the external sensor. The objective of the base manager is to create a new base when the old base is no longer within range of the sensor. The 'base' landmarks that define the associated frame are represented in a global frame. All the other landmarks that belong to this constellation are defined in the local frame. For a particular constellation, the local frame is based on two base landmarks:

$$L_a = \begin{bmatrix} x_a \\ y_a \end{bmatrix}, \quad L_b = \begin{bmatrix} x_b \\ y_b \end{bmatrix}. \tag{45}$$

As shown in Fig. 5, it is possible to define two unitary vectors that describe the orientation of the base frame:

$$\begin{aligned} \nu_1 =& \frac{1}{\|L_b - L_a\|} \cdot (L_b - L_a) \\ =& \frac{1}{\sqrt{(x_b-x_a)^2 + (y_b-y_a)^2}} \\ & \cdot \begin{bmatrix} x_b - x_a \\ y_b - y_a \end{bmatrix} = \begin{bmatrix} \nu_{11} \\ \nu_{12} \end{bmatrix} \\ \nu_2 =& \begin{bmatrix} \nu_{21} \\ \nu_{22} \end{bmatrix} = \begin{bmatrix} -\nu_{12} \\ \nu_{11} \end{bmatrix}, \quad \langle \nu_2, \nu_1 \rangle = 0. \end{aligned} \tag{46}$$

The rest of the landmarks in this particular constellation are represented using a local frame with origin at L_a and axes parallel to the vectors ν_1 and ν_2

$$L_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}, \quad L_i^a = \begin{bmatrix} \xi_i \\ \eta_i \end{bmatrix} \tag{47}$$

with

$$\begin{aligned} \xi_i &= \langle (L_i - L_a), \nu_1 \rangle = (L_i - L_a)^{\mathrm{T}} \cdot \nu_1 \\ \eta_i &= \langle (L_i - L_a), \nu_2 \rangle = (L_i - L_a)^{\mathrm{T}} \cdot \nu_2. \end{aligned} \tag{48}$$

The following expression can be used to obtain the absolute coordinates from the relative coordinate representation

$$L_i = L_a + \zeta_i \cdot \nu_1 + \eta_i \cdot \nu_2. \tag{49}$$

Assuming that the external sensor returns range and bearing, the observation functions are

$$\begin{aligned} h_i &= L_i - X_L - R_i \cdot (\cos(\beta_i), \sin(\beta_i)) = 0 \\ \beta_i &= \alpha_i + \phi - \frac{\pi}{2} \\ &\alpha\text{: object angle w.r.t. laser frame} \\ &R_i\text{: object range w.r.t. laser} \\ (X_L, \phi) &= (x_L, y_L, \phi)\text{: vehicle states.} \end{aligned} \tag{50}$$

Finally

$$\begin{aligned} h_i =& L_a + \zeta_i \cdot \nu_1 + \eta_i \cdot \nu_2 - P_L - R_i \\ & \cdot (\cos(\beta_i), \sin(\beta_i)) = 0. \end{aligned} \tag{51}$$

With this representation the landmark defining the bases will become the natural "Preserved" landmarks. Provided that the landmarks representing the basis are visible in a sufficient number of observations, the entire observation set may be regarded as being contaminated by white noise. The Gaussian characteristics of the observations cause the relative elements of the constellation to be uncorrelated with the other constellation relative elements. The only landmarks that will maintain strong correlation will be the ones defining the bases that are represented in absolute form.

V. Experimental Results

The navigation algorithms presented were tested in the outdoor environment shown in Fig. 6. A standard utility vehicle was fitted with dead reckoning sensors and a laser range sensor as shown in Fig. 7.

The landmarks detection and extraction process is essential for SLAM. In this particular application, the most common relevant feature in the environment were trees. The profiles of trees were extracted from the laser, as shown in Fig. 8, and the most likely center of the trunk was estimated. A Kalman filter was implemented to reduce the errors due to the different profiles obtained when observing the trunk of the trees from different locations.

The vehicle was started at a location with known uncertainty and driven in this area for approximately 20 min. Fig. 9 presents the vehicle trajectory and navigation landmarks incorporated into the relative map. This run includes all the features in the environment and the optimization presented in Section III. The system built a navigation map of the environment and localized

Fig. 6. Outdoor environment used for this experiment. This is a large area with different type of surfaces and different levels.

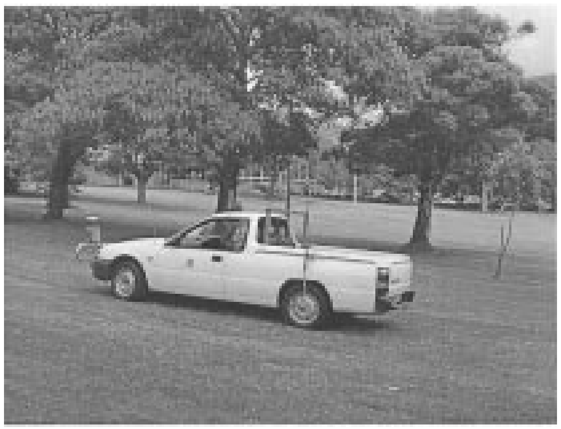
Fig. 7. The utility car used for the experiments is equipped with a Sick laser range and bearing sensor, linear variable differential transformer sensor for the steering and back wheel velocity encoders.

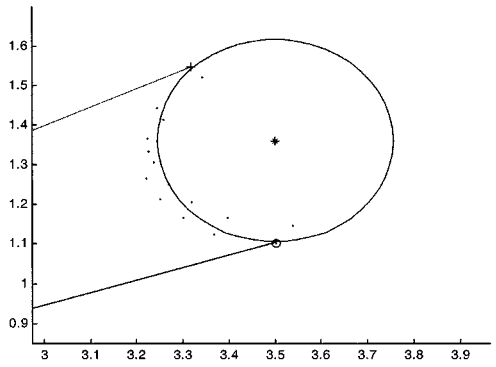

Fig. 8. Tree profile and trunk approximation. The dots indicate the laser range and bearing returns. The filter estimates the radius of the circumference that approximates the trunk of the tree and center position.

itself. The accuracy of this map is determined by the initial vehicle position uncertainty and the quality of the combination of dead reckoning and external sensors. In this experimental run an initial uncertainty in coordinates x and y was assumed. Fig. 10 presents the estimated error of the vehicle position and selected landmarks. Although the vehicle was equipped with a kinematic GPS to evaluate the ground truth it was not accurate enough due to poor satellite availability. This is a common problem in this type of environment. Nevertheless the information gathered

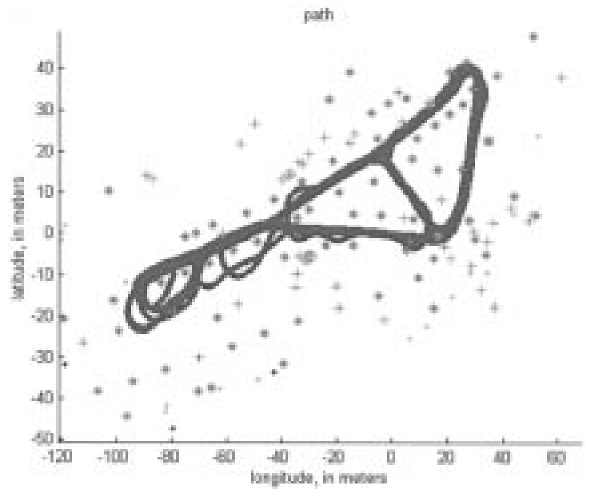

Fig. 9. Vehicle trajectory and landmarks. The ‘o’, “$*$,” and “$+$” show the estimated position of objects that qualified as landmarks for the navigation system. The “o” are the landmarks more frequently detected. The dots are laser returns that are not stable enough to qualify as landmarks. The solid line shows the 20–min vehicle trajectory estimation using full SLAM.

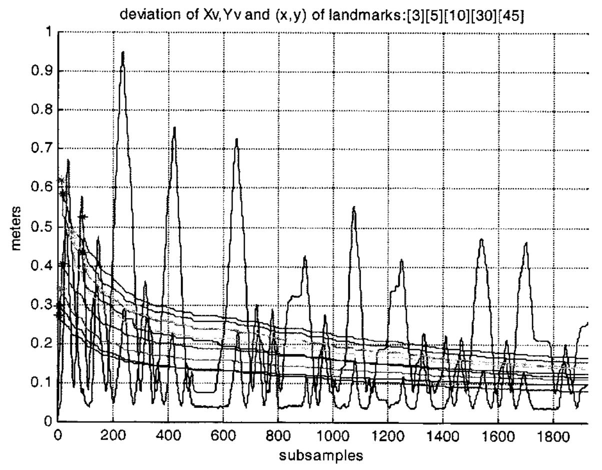

Fig. 10. History of selected state’s estimated errors. The vehicle states show oscillatory behavior with error magnitude that is decreasing with time due to the learning of the environment. The landmarks always present an exponential decreasing estimated error with a lower limit bounded by the initial uncertainty of the vehicle position.

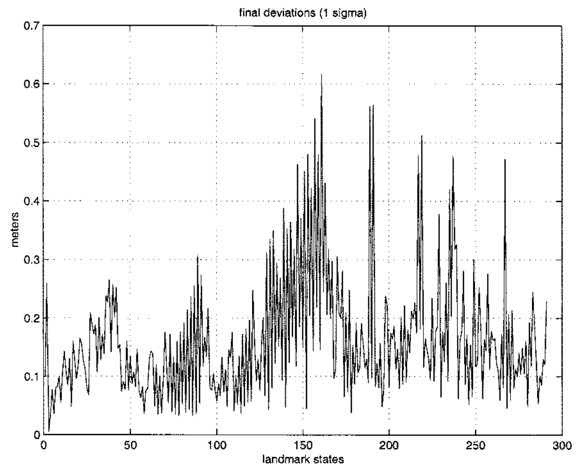

Fig. 11. Final estimated error of all states. The maximum error is approximately 60 cm.

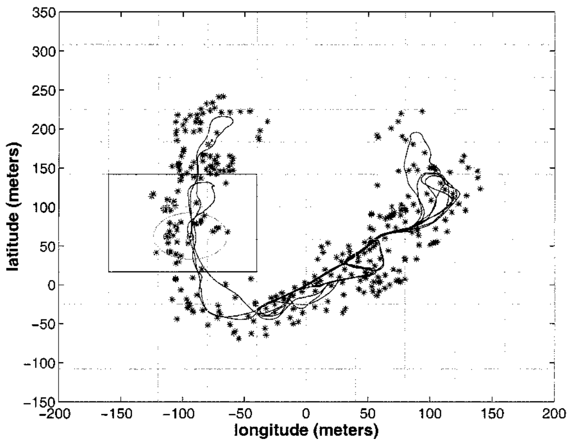

Fig. 12. Vehicle and local areas. This plot presents the estimated trajectory and navigation landmark estimated position. It also shows the local region “r” with its surrounding regions. The local states X_A are the ones included in the nine regions shown enclosed by a rectangle in the left side of the plot.

was used to verify that actual errors were consistent with the filter-estimated errors. The states corresponding to the vehicle present oscillatory behavior displaying the maximum deviation farther from the initial position. This result is expected since there is no absolute information incorporated into the process. The only way this uncertainty can be reduced is by incorporating additional information not correlated to the vehicle position, such as GPS position information or recognizing a beacon located at a known position. It is also appreciated that the covariances of all the landmarks are decreasing with time. This means that the map is learned with more accuracy while the vehicle navigates. The theoretical limit uncertainty in the case of no additional absolute information will be the original uncertainty vehicle location. Fig. 11 presents the final estimation of the landmarks in the map. It can be seen that after 20 min the estimated error of all the landmarks are below 60 cm.

The compressed algorithm was implemented using local regions of 40×40 meters square. These regions are appropriate for the laser range sensor used in this experiment. Fig. 12 shows part of the trajectory of the vehicle with the local area composed of 9 squares surrounding the vehicle. To verify the fact that the algorithm proposed maintains and propagates all the information obtained, the history of the covariances of the landmarks were compared with the ones obtained with the full SLAM algorithm. Fig. 13 shows a time evolution of standard deviation of few landmarks. The dotted line corresponds to the compressed filter and the solid line to the full SLAM. It can be seen that the estimated error of some landmarks are not continuously updated with the compressed filter. These landmarks are not in the local area. Once the vehicle makes a transition, the system updates all the landmarks performing a full SLAM update. At this time, the landmarks outside the local area are updated in one iteration and its estimated error become exactly equal to the full SLAM. This is clearly shown in Fig. 14 where at the full update time stamps both estimated covariances become identical. Fig. 15 shows the difference between full SLAM and compressed filter estimated landmarks covariance. It can be seen that at the full update time

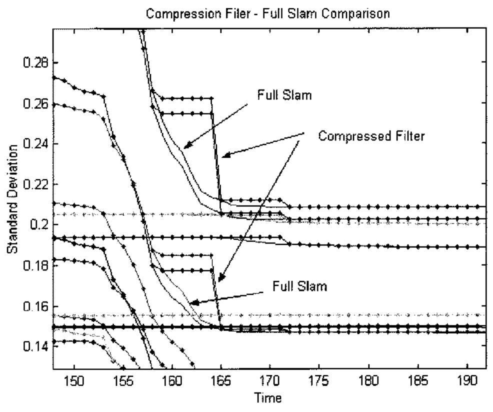

Fig. 13. Landmark estimated position error for full SLAM and compressed filter. The solid line shows the estimated error evaluated with the full SLAM algorithm. This algorithm updates all the landmarks with each observation. The dotted line shows the estimated error evaluated with the compressed filter. The landmark that are not in the local area are only updated when the vehicle leaves the local area. At this time a full update is performed and the estimated error becomes exactly equal to full SLAM.

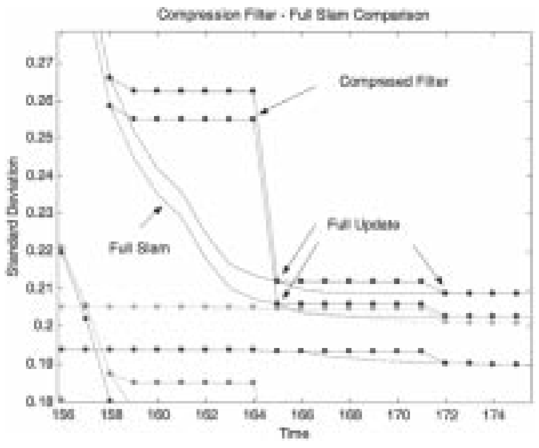

Fig. 14. Landmark estimated position error for full SLAM and compressed filter (enhanced). This plot presents a clear view the instant when the compressed algorithm performed a full update. At this time (165) the full SLA M (solid line) and the compressed algorithm (solid lines with dotes) report same estimated error as predicted.

stamps the difference between the estimation using both algorithms becomes zero as demonstrated in this paper. This demonstrates that while working in a local area it is possible to maintain all the information gathered with a computational cost proportional to the number of landmarks in the local area. This information can then be propagated to the rest of the landmarks in the map without any loss of information.

The next set of plots present a comparison of the performance of the suboptimal algorithm proposed in Section IV using the relative map representation with full SLAM. Figs. 16 and 17 present two runs, one using most of the states and the other with only 100 states. The plots show that the total number of states used by the system grows with time as the vehicle explores new areas. It is also shown the number of states used by the system in grey and the number of states not updated with stars "$*$." In the first run, very conservative values for the constant I_1 and I_2 were selected so most of the states were updated with each observation. The second run corresponds to a less conservative selection plus a limitation in the maximum number of states.

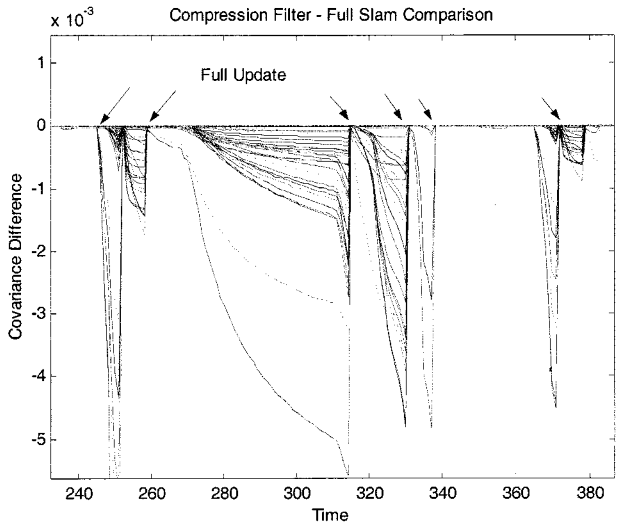

Fig. 15. Estimated error differences between full SLAM and compressed filter. The estimated error difference between both algorithms becomes identically zero when the full update is performed by the compressed algorithm.

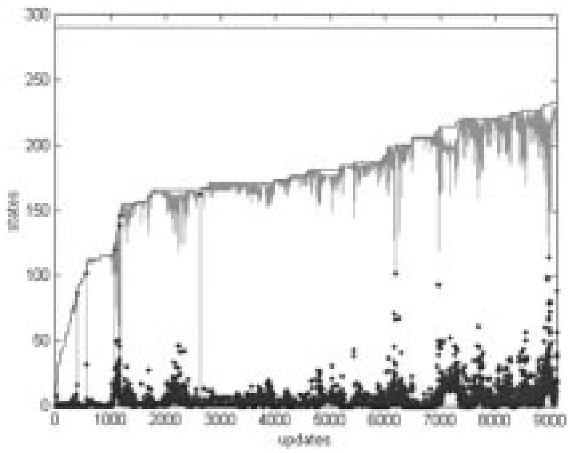

Fig. 16. Total number of states and states used and not updated. The figure presents the total number of states with a solid black line. This number is increasing because the vehicle is exploring new areas and incorporating new landmarks. The states used by the system are represented in grey. The number of states not used is represented with "$*$." In this run, the system used most of the states available.

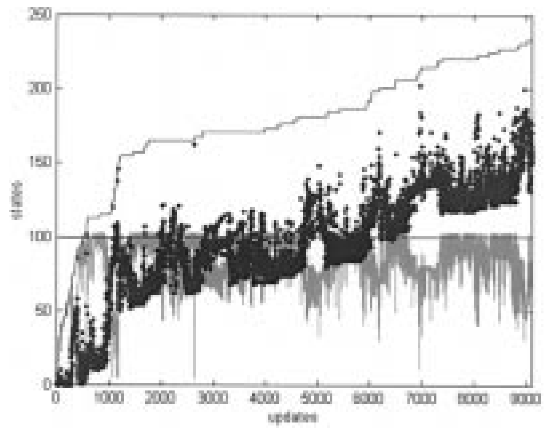

Fig. 17. Total number of states and states used and not updated. In this run a maximum number of states was fixed as constraint for the suboptimal SLAM algorithm. This is appreciated in the gray plot where the maximum number of states remains below a given threshold. The number of states not updated increases with time.

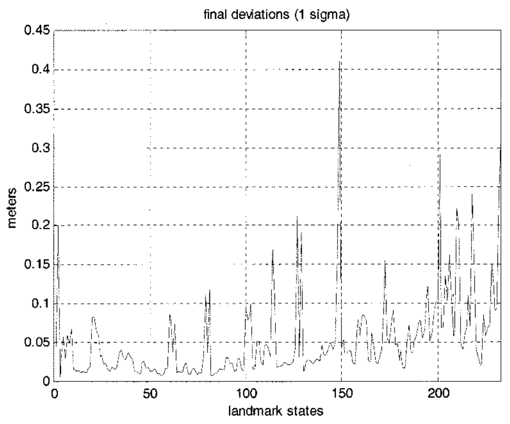

Fig. 18. Final estimation errors of relative and absolute states using most of the states.

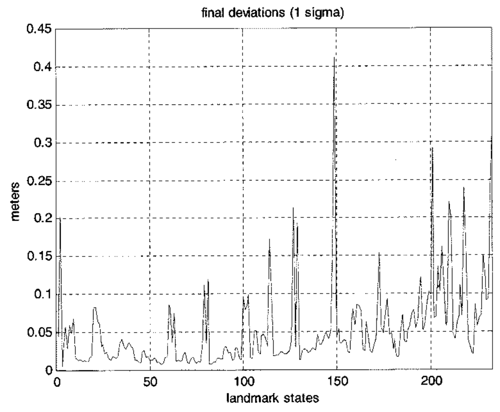

Fig. 19. Final estimated error of relative and absolute states using a reduced number of states. These results are similar to the ones using most of the states. This result shows that the proposed algorithm is not only consistent but close to optimal when used with the appropriate map representation.

Fig. 17 shows that a large number of states were not updated at every time step resulting in a significant reduction in the computational cost of the algorithm. From Figs. 18 and 19 it can be seen that the accuracy of the SLAM algorithm has not been degraded by this simplification. These figures present the final estimated error of all the states for both runs. It is noteworthy that only the bases are represented in absolute form. The other states are represented in relative form and its standard deviation becomes much smaller. This can also be appreciated in Figs. 20 and 21 that present the estimated error history of the states selected as bases. The constellation map and vehicle trajectory of part of the run are shown in Fig. 22. The systems build five constellations in this area. The intersection of each group of bases is represented with a "+" and the landmark used as bases with a " o." All the other relative landmarks are represented with "$*$."

There is one important remark regarding the advantage of the relative representation with respect to the simplification proposed: Since the bases are in absolute form they will maintain a strong correlation with the other bases and the vehicle states. They will be more likely to be chosen as "preserved" landmarks since the observations will have more contribution to them than the relative states belonging to distant bases. In fact the states that will be chosen will most likely be the bases and the states associated with the landmarks in the local constellation.

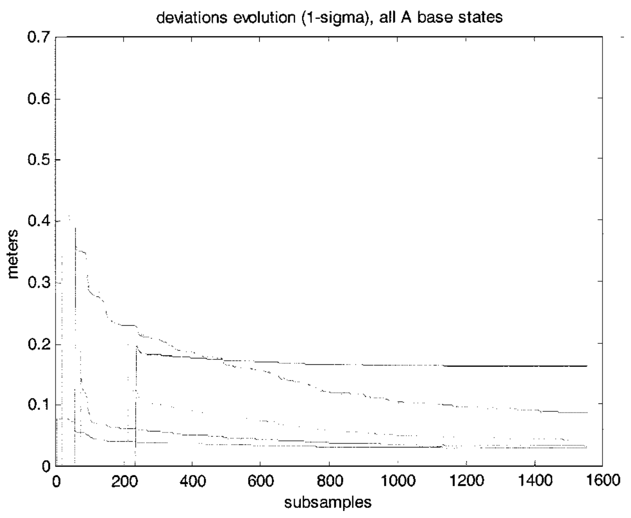

Fig. 20. Estimated error history of the bases using most of the states.

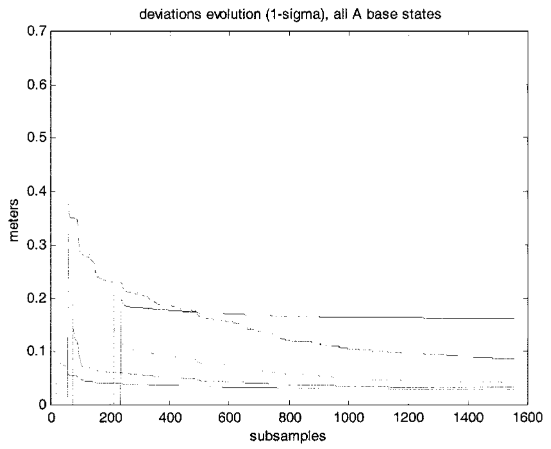

Fig. 21. Estimated error history of the bases with reduced number of states.

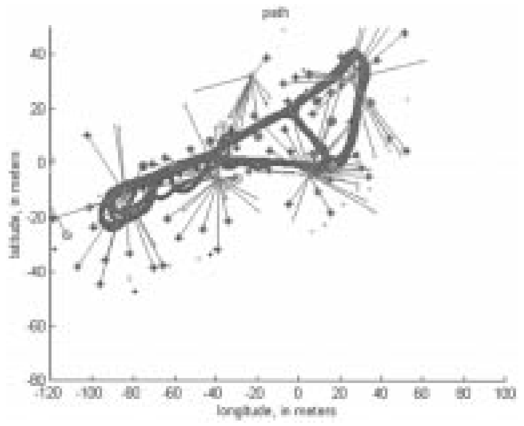

Fig. 22. Constellation map and vehicle trajectory. Five constellations were created by the algorithm. The intersection of the bases are presented with a "+," and the other side of the segment with a "o." The relative landmarks are represented with "$*$" and its association with a base is represented with a line joining the landmark with the origin of the relative coordinate system.

It is also important to remark that with this representation the simplification becomes less conservative than when using the absolute representation. This can clearly be seen by looking at the correlation coefficients for all the states in each case. This

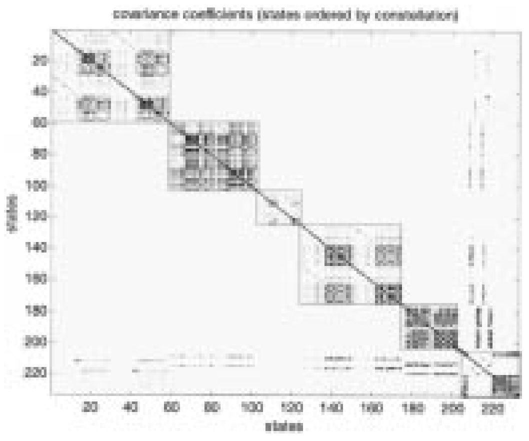

Fig. 23. Correlation coefficients for the relative representation. Each block represents the cross-correlation coefficient of the elements of the different constellations. The block in the right corner contains the states corresponding to the vehicle and the bases. It can be seen that the cross correlation between different constellations is very small. It is also clear the nonzero cross correlation between the bases and the different constellations. These correlations are updated by the suboptimal filter.

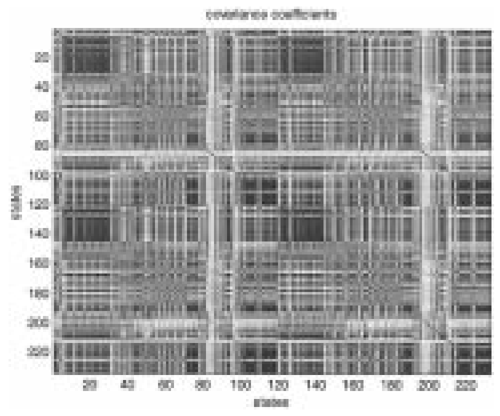

Fig. 24. Correlation coefficients for the absolute representation. In this case the map appears completely correlated and the suboptimal algorithm will generate consistent but more conservative results.

is shown in Figs. 23 and 24 where the correlation of the relative and absolute map, respectively, is presented. In Fig. 23 each block of the diagonal corresponds to a particular constellation and the last block has the vehicle states and the bases. The different constellations becomes decorrelated from each other and only correlated to the first block whose cross correlation are updated by the suboptimal algorithm presented. These results imply that with the relative representation the cross correlation between constellations becomes zero and the suboptimal algorithm presented becomes close to optimal. This is not the case for the absolute representation as shown in Fig. 24 where all the states maintained strong cross correlations.

Finally, Fig. 25 presents the results of a 4–km trajectory using the compressed algorithm in a large area. In this case there are approximately 500 states in the global map and their final estimated errors are shown in Fig. 27. The system creates 18 different constellations to implement the relative map. The cross-correlation coefficients between the different constellations become very small as shown in Fig. 26. This run is useful to demonstrate the advantages of the compressed algorithm since

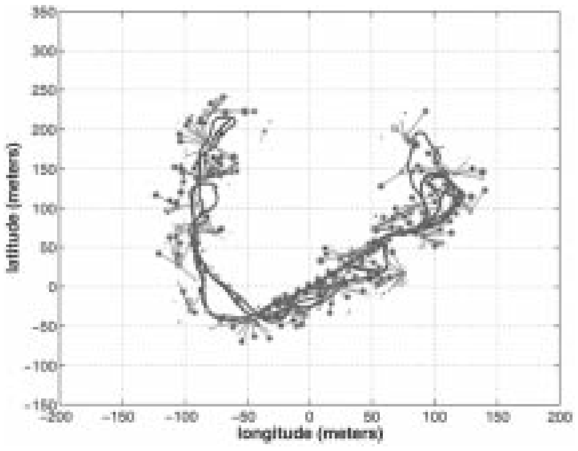

Fig. 25. Long trajectory using the Compressed SLAM algorithm. This plot presents a run in a very large area where 18 constellations were created.

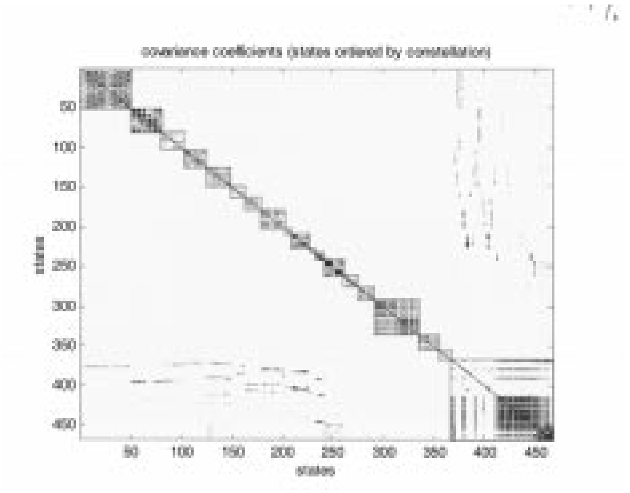

Fig. 26. Cross correlation coefficients. The plot shows 18 constellation and a block in the right hand corner containing the correlation coefficient for the bases and the vehicle states. It can be appreciated that the cross correlation between the relative states of the different bases is very small.

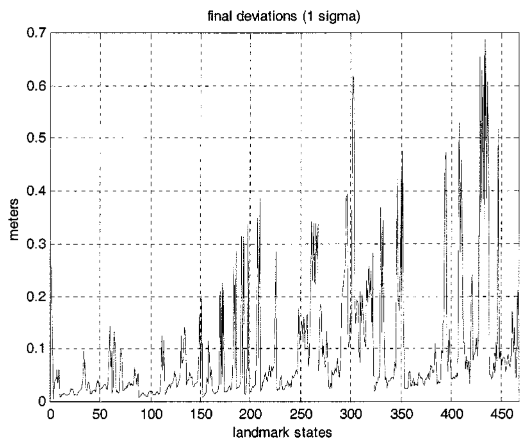

Fig. 27. Final estimated error of the 500 states. It can be seen that the maximum estimated error is smaller than 0.7 m.

the local areas are significantly smaller than the global map. When compared with the full SLAM implementation the algorithm generated identical results (states and covariance) with the advantage of having very low computational requirements. For larger areas the algorithm becomes more efficient since the cost is basically function of the number of local landmarks. These results are important since it demonstrates even in very large areas ($>$ 10 000 landmarks) the computational limitation of SLAM can be overcome with the compressed algorithm and appropriate selection of local areas.

VI. Conclusion

The paper presented efficient algorithms for real-time implementation of SLAM. In particular a compressed algorithm was introduced that is very attractive in applications where high frequency external sensor information is available or when the vehicle navigates for long periods of time in a local area. It is shown that the information gathered in a local area can be incorporated into the vehicle states and the local map with a computational cost similar to a standard local SLAM algorithm and can then be transferred to an arbitrarily large global map with the implementation of full SLAM algorithm in only one iteration without loss of information.

A simplification to the SLAM algorithm has also been proposed with theoretical proofs of the consistency of the approach. Furthermore, it has also been shown with experimental results that, by using a relative map representation, the algorithm becomes very close to optimal. With this approach the user can allocate a maximum number of landmarks, according to the computational resources available, and the system will optimally select the ones that provide the maximum information.

Future work will address the extension of the compression filter results in decentralized SLAM where different platforms can update their own map with a particular sensor and then transfer all the information gained to the rest of the system.

The incorporation of high frequency information increases the exploration range of the SLAM algorithm. This is also another important area of research. If no absolute position data is made available, the system will not be able to navigate for extended periods of time in new areas without returning to known areas. Although standard sensors allow SLAM to perform in significantly large areas, in order to extend this range there are two important problem to be solved: The reregistration (association) of a known revisited area and the back-propagation of the corrections once a large loop is traversed. The first problem looks solvable working with the geometry of the environment [25], or using more complex data association methods [23]. The other problem is not solved yet and subject of current research.

Appendix A

A. Modeling

Under the general EKF framework we can have nonlinear models for the process and observations in the form:

$$\begin{aligned} X(k+1) &= F(X(k), u(k)+\gamma_u(k)) + \gamma_f(k) \\ z(k) &= h(X(k)) + \gamma_h(k) \end{aligned} \tag{52}$$

where X are the states of the system, in this case position x, y and orientation ϕ. $F(\cdot)$ is a non linear function that propagates the states based on the inputs u and the state's previous value. The nonlinear observation equation $h(\cdot)$ maps the states to the observation vector z.

The effect of the input signal noise is approximated by a linear representation

$$\begin{aligned} F(X(k), u(k)+\gamma_u(k)) + \gamma_f(k) &\cong F(X(k), u(k)) \\ &\quad + \gamma(k) \\ \gamma(k) &= J_u \cdot \gamma_u(k) + \gamma_f(k) \\ J_u &= \left.\frac{\partial F}{\partial u}\right|_{X=X(k), u=u(k)}. \end{aligned} \tag{53}$$

The matrix noise characteristics are assumed zero mean and white

$$\begin{aligned} E\{\gamma_f(k)\} = E\{\gamma_u(k)\} &= E\{\gamma_h(k)\} = 0 \\ E\{\gamma_f(i)\cdot\gamma_f^{\mathrm{T}}(j)\} &= \delta_{i,j}\cdot Q_f(i) \\ E\{\gamma_h(i)\cdot\gamma_h^{\mathrm{T}}(j)\} &= \delta_{i,j}\cdot R(i) \\ E\{\gamma_u(i)\cdot\gamma_u^{\mathrm{T}}(j)\} &= \delta_{i,j}\cdot Q_u(i) \\ E\{\gamma_h(i)\cdot\gamma^{\mathrm{T}}(j)\} &= 0 \\ \delta_{i,j} &= \begin{cases} 0 & i\neq j \\ 1 & i=j \end{cases} \\ E\left\{\gamma(i)\cdot\gamma^{\mathrm{T}}(j)\right\} &= \delta_{i,j}\cdot\left(J_u\cdot Q_u(i)\cdot J_u^{\mathrm{T}} + Q_f(i)\right) \\ &= \delta_{i,j}\cdot Q(i). \end{aligned} \tag{54}$$

An EKF observer based on the process and output models can be formulated in two stages: Prediction and Update stages. The Prediction stage is required to obtain the predicted value of the states X and its error covariance P at time k based on the information available up to time $k-1$,

$$\begin{aligned} X(k+1,k) &= F(X(k,k), u(k)) \\ P(k+1,k) &= J\cdot P(k,k)\cdot J^{\mathrm{T}} + Q(k). \end{aligned} \tag{55}$$

The update stage is function of the observation model and the covariances

$$\begin{aligned} S(k+1) &= H\cdot P(k+1,k)\cdot H^{\mathrm{T}}(k+1) + R(k+1) \\ W(k+1) &= P(k+1,k)\cdot H^{\mathrm{T}}(k+1)\cdot S^{-1}(k+1) \\ \vartheta(k+1) &= Z(k+1) - h(X(k+1,k)) \\ X(k+1,k+1) &= X(k+1,k) + W(k+1)\cdot\vartheta(k+1) \\ P(k+1,k+1) &= P(k+1,k) - W(k+1)\cdot S(k+1) \\ &\quad \cdot W(k+1)^{\mathrm{T}} \end{aligned} \tag{56}$$

where

$$\begin{aligned} J &= J(k) = \left.\frac{\partial F}{\partial X}\right|_{(X(k),u(k))}, \\ H &= H(k) = \left.\frac{\partial h}{\partial X}\right|_{X=X(k)} \end{aligned} \tag{57}$$

are the Jacobian matrices of the vector functions $F(x,u)$ and $h(x)$ with respect to the state X and R is the covariance matrix characterizing the noise in the observations.

Appendix B

The general form of the update for the states belonging to the global area will have the following form:

$$X_b^{\text{new}} = X_b^{\text{old}} + dX_b. \tag{58}$$

In this section we present the evaluation of dX_b to update the vector X_b after t local updates were performed. For $t = 1$, that is, when the vector X_b is updated after one observation, we have

$$\begin{aligned} dX_b =& W(k+1) \cdot (h(X_a(k+1,k)) - Z(k+1)) \\ =& W(k+1) \cdot \vartheta(k+1). \end{aligned} \tag{59}$$

Since the Kalman gain matrix can be partitioned in two main components:

$$W(k+1) = \begin{bmatrix} W_a(k+1) \\ W_b(k+1) \end{bmatrix}. \tag{60}$$

Then the state update can be simplified as shown

$$X_b(k+1,k+1) = X_b(k+1,k) - W_b(k+1) \cdot \vartheta(k+1). \tag{61}$$

Finally, the update after t local updates are performed using

$$\begin{aligned} X_b(k+t,k+t) =& X_b(k+t,k) - \sum_{i=k}^{k+t-1} W_b(i) \cdot \vartheta(i) \\ =& X_b(k+t,k) - P_{ba}(k,k) \cdot \\ & \sum_{i=k}^{k+t-1} \Phi^{\mathrm{T}}(i-1) \\ & \cdot H_a^{\mathrm{T}}(i) \cdot S^{-1}(i) \cdot \vartheta(i) \end{aligned} \tag{62}$$

the expression can be simplified as follows:

$$X_b(k+t,k+t) = X_b(k+t,k) - P_{ba}(k,k) \cdot \theta(k+t) \tag{63}$$

with

$$\theta(k+t) = \sum_{i=k}^{k+t-1} \Phi^{\mathrm{T}}(i-1) \cdot H_a^{\mathrm{T}}(i) \cdot S^{-1}(i) \cdot \vartheta(i). \tag{64}$$

Appendix C

Taking advantage of the sparseness in the local area.

As demonstrated, the maintenance of the auxiliary matrices in the compression algorithm involves products of matrices with dimensions not higher than N_a, $N_a \ll N$. In addition most of these matrices are sparse. This fact can be exploited to improve the efficiency of the algorithm. This is important since the local updates are done at the rate of the external sensor.

To facilitate the representation of nonnull matrices a submatrix M^* can be defined according to $M^* = M(i_a, i_b)$, where i_a and i_b are integer arrays that define subsets of indexes. With this convention the submatrix M^* can be expressed as function of the original matrix M as follows:

$$M^*(r,c) = M(i_a(r), i_b(c)). \tag{65}$$

The subset of all the valid indexes in a row or column will be indicated with ':'. The first stage requires the evaluation of the matrices κ and ξ

$$\begin{aligned} \kappa =& H_a^{\mathrm{T}} S^{-1} H_a \\ \xi =& P_{aa} \kappa. \end{aligned} \tag{66}$$

Most of the elements of the Jacobian matrix of the observation function are zero. Assuming that only one landmark is being observed, we can define i_{obs} as index subsets that correspond to the vehicle states and the observed landmark states. Then the matrix H will have the following null-matrices:

$$\begin{aligned} H_a(:,i) =& 0 \quad \forall\, i,\ \ i \notin i_{\text{obs}} \\ (H_a(j,i) =& 0 \quad \forall\, (j,i),\ \ i \notin i_{\text{obs}}). \end{aligned} \tag{67}$$

Considering that

$$\kappa(i,j) = 0 \quad \forall\, (i,j),\ \ i \notin i_{\text{obs}} \text{ or } j \notin i_{\text{obs}}$$

then evaluation of κ only requires the computation of κ^*

$$\begin{aligned} \kappa(i,j) =& H_a^{\mathrm{T}}(i,:) S^{-1} H_a(:,j) \\ \kappa^* =& \kappa(i_{\text{obs}}, i_{\text{obs}}) = H_a^{\mathrm{T}}(i_{\text{obs}},:) S^{-1} H_a(:,i_{\text{obs}}) \\ =& H_a(:,i_{\text{obs}})^{\mathrm{T}} S^{-1} H_a(:,i_{\text{obs}}). \end{aligned} \tag{68}$$

Similar simplification can be done for ξ

$$\begin{aligned} \xi =& P_{aa} \cdot \kappa \\ \xi(:,j) =& 0 \quad \forall\, j,\ \ j \notin i_{\text{obs}} \\ \xi^* =& \xi(:,i_{\text{obs}}) = P_{aa}(:,i_{\text{obs}}) \kappa^* \end{aligned} \tag{69}$$

The evaluation of the matrices given in (31) can also be simplified by using a different representation for Φ:

$$\Phi = I + \pi \tag{70}$$

then

$$\begin{aligned} \Phi(k) =& I + \pi(k) = (I - \xi(k))(I + \pi(k-1)) \\ =& I + \pi(k-1) - \xi(k)\pi(k-1) - \xi(k) \\ \pi(k) =& \pi(k-1) - \xi(k)\pi(k-1) - \xi(k) \end{aligned} \tag{71}$$

The new subset i_{ee} involves all the states that were used in previous observations or predictions. Then the following simplifications are possible:

$$\begin{aligned} \pi(k-1)(:,j) =& \bar{0} \quad \forall\ j \notin i_{\text{ee}} \\ \xi(k)(:,j) =& \bar{0} \quad \forall\ j \notin i_{\text{obs}}. \end{aligned} \tag{72}$$

Defining the two auxiliary variables λ and ω:

$$\begin{aligned} \lambda =& \xi(k)\pi(k-1) \\ \lambda^* =& \lambda(:,i_{\text{ee}}) = \xi^*(k)\pi(k-1)(i_{\text{obs}}, i_{\text{ee}}) \\ \lambda(:,j) =& \bar{0} \quad \forall\ j \notin i_{\text{ee}} \\ \hat{\omega} =& \pi(k-1)(:, i_{\text{obs}} \cup i_{\text{ee}}) \\ \tilde{\omega}(:,i_{\text{ee}}) =& \hat{\omega}(:,i_{\text{ee}}) - \lambda^* \\ \tilde{\omega}(:,i_{\text{obs}}) =& \hat{\omega}(:,i_{\text{obs}}) - \xi^*(k). \end{aligned} \tag{73}$$

Finally π can be updated using

$$\begin{aligned}\pi(k)(:,i_{\rm obs}\cup i_{\rm ee}) &=\tilde{\omega}(:,i_{\rm obs}\cup i_{\rm ee})\\ \pi(k)(:,j) &=\pi(k-1)(:,j) \quad \forall\ j\notin\{i_{\rm obs}\cup i_{\rm ee}\}\\ i_{\rm ee}(k) &=i_{\rm obs}\cup i_{\rm ee}(k-1). \end{aligned} \tag{74}$$

The array index $i_{\rm ee}$ takes into account the increase in the population of observed landmarks. It includes all the observed states since the last full update. This implies that the computational of the full update will have a computational cost proportional to $N_b^2 N_{\rm e}$, being $N_{\rm e}$ the number of elements in $i_{\rm ee}$.

Acknowledgment

The authors would like to thank the Australian Centre for Field Robotics staff, students, and researchers for their support and valuable comments. Also thanks to A. Maclean for proofreading this work.

References

[1] E. Nebot and H. Durrant-Whyte, "High integrity navigation architecture for outdoor autonomous vehicles," *J. Robot. Auton. Syst.*, vol. 26, no. 1, pp. 81–97, Feb. 1999.

[2] A. Elfes, "Occupancy grids: A probabilistic framework for robot perception and navigation," Ph. D., Dept. Elect. Eng., Carnegie Mellon Univ., 1989.

[3] ——, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, June 1989.

[4] R. Chatila and J. Laumond, "Position reference and consistent world modeling for mobile robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 1985, pp. 138–145.

[5] H. Durrant-Whyte, "An autonomous guided vehicle for cargo handling applications," *Int. J. Robot. Res.*, vol. 15, no. 5, pp. 407–441, Oct. 1996.

[6] J. Castellanos, J. Tardos, and G. Schmidt, "Building a global map of the environment of a mobile robot: the importance of correlations," in *Proc. IEEE Int. Conf. Robot. Automat.*, New Mexico, 1997, pp. 1053–1059.

[7] R. Smith, M. Self, and P. Cheeseman, "Estimating Uncertain spatial relationships in robotics," in *Uncertainty in Artificial Intelligence*, New York: Elsevier Science, 1988, vol. 2, pp. 435–461.

[8] P. Motarlier and R. Chatila, "Stochastic multi-sensory data fusion for mobile robot location and environmental modeling," in *Fifth Symp. Robot. Res.*, Tokyo, 1989, pp. 85–94.

[9] S. Thrun, D. Fox, and W. Bugard, "Probabilistic mapping of an environment by a mobile robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, Belgium, May 1998, pp. 1546–1551.

[10] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust Monte Carlo Localization for Mobile Robots," School of Comput. Sci., Carnegie Mellon Univ., CMU-CS-00–125, 2000.

[11] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/nongaussian Bayesian State Estimation," *IEE Proc.*, vol. 140, no. 2, pp. 107–113, Mar. 1993.

[12] W. Burgard, A. B. Cremers, D. Fox, D. Ahnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun, "Experiences with an interactive museum tour-guide robot," *Artif. Intell.*, vol. 114, no. 1–2, pp. 3–55, Jan. 1999.

[13] P. S. Maybeck, *Stochastic Models, Estimation, and Control*. New York: Academic, 1979, vol. I.

[14] A. Stentz, M. Ollis, S. Scheding, H. Herman, C. Fromme, J. Pedersen, T. Hegardorn, T. McCall, J. Bares, and R. Moore, "Position measurement for automated mining machinery," in *Proc. Int. Conf. Field Service Robot.*, Pittsburgh, PA, Aug. 1999, pp. 299–304.

[15] S. Sukkarieh, E. Nebot, and H. Durrant-Whyte, "A High Integrity IMU/GPS Navigation Loop for Autonomous Land Vehicle applications," *IEEE Trans. Robot. Automat.*, vol. 15, pp. 572–578, June 1999.

[16] S. Scheding, E. Nebot, and H. Durrant-Whyte, "High integrity Navigation using frequency domain techniques," *IEEE Trans. Syst. Contr. Technol.*, vol. 17, pp. 676–694, July 2000.

[17] J. Leonard and H. Durrant-Whyte, "Simultaneous map building and Localization for an autonomous mobile robot," in *Proc. IEEE Int. Wkshp. Intell. Robots Syst.*, Osaka, Japan, pp. 1442–1447.

[18] J. Castellanos, J. Martinez, J. Neira, and J. Tardos, "Simultaneous map building and localization for mobile robots: a multisensor fusion approach," in *Proc. IEEE Conf. Robot. Automat.*, Belgium, May 1998, pp. 1244–1249.

[19] J. Leonard and H. J. S. Feder, "A computationally efficient method for large-scale concurrent mapping and localization," in *Proc. 9th Int. Symp. Robot. Res.*, UT, Oct. 1999, pp. 316–321.

[20] S. Williams, P. Newman, M. Dissanayake, J. Rosenblatt, and H. Durrant-Whyte, "A decoupled, distributed AUV control architecture," in *Proc. 31st Int. Symp. Robot.*, vol. 1, Montreal, PQ, Canada, May 2000, pp. 246–251.

[21] J. Guivant, E. Nebot, and H. Durrant-Whyte, "Simultaneous localization and map building using natural features in outdoor environments," in *Proc. IAS-6 Intell. Auton. Syst.*, Italy, July 2000, pp. 581–586.

[22] J. Guivant, E. Nebot, and S. Baiker, "Localization and map building using laser range sensors in outdoor applications," *J. Robot. Syst.*, vol. 17, pp. 565–583, Oct. 2000.

[23] J. Neira and J. Tardos, "Robust and feasible data association for simultaneous localization and map building," in *IEEE Int. Conf. Robot. Automat., Wkshp. W4*, San Francisco, CA, Apr. 2000.

[24] J. Castellanos, M. Devy, and J. Tardos, "Simultaneous localization and map building for mobile robots: a landmark based approach," in *IEEE Conf. Robot. Automat., Wkshp. W4*, San Francisco, CA, Apr. 2000.

[25] T. Bailey, E. Nebot, J. Rosenblat, and H. D. Whyte, "Data association for mobile robot navigation: a graph theoretic approach," in *Proc. IEEE Int. Conf. Robot. Automat.*, San Francisco, CA, Apr. 2000, pp. 2512–2517.

José E. Guivant received the Electronic Eng. degree from the Universidad Nacional del Sur (UNS), Bahía Blanca, Argentina, in 1993. Since 1999, he has been with the Australian Centre of Field Robotics, University of Syndey, pursuing the Ph.D. degree.

Since 1995 he has been with the control group of the Departamento de Ingenieria Eléctrica, UNS. His research interests include nonlinear control systems, autonomous navigation, localization and navigation algorithms, and sensor-based map building.

Eduardo Mario Nebot (S'79–M'81–M'89–SM'01) received the Bachelor's degree in electrical engineering from the Universidad Nacional del Sur (UNS), Argentina, and the M.S. and Ph.D. degrees from Colorado State University.

He is an Associate Professor with the Department of Mechanical and Mechatronic Engineering, University of Sydney, where he has also been a faculty member since 1992. His current research interests include: Inertial sensors, GPS and inertial sensing in land vehicle applications, navigation and guidance algorithms; decentralized and distributed navigation, map building and terrain aiding. He has also been involved in different automation projects in the stevedoring, mining, and cargo handling industries. He is currently the Automation Program Manager for the Centre for Mining, Technology and Equipment (CMTE).