Simultaneous Localization and Mapping
2002 Summer School

Eduardo Nebot
Australian Centre for Field Robotics
The University of Sydney NSW 2006
Australia
nebot@acfr.usyd.edu.au
http://acfr.usyd.edu.au/homepages/academic/enebot/

July 31, 2002

Version 0.9

1 Simultaneous Localisation and Mapping (SLAM)

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. 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. Section 3 presents a Kalman filter technique to estimate the position of the vehicle based on the known position of artificial landmarks. Although this method can be made very reliable is has the drawback that requires the modification of the environment with the addition special infrastructure. In addition the location of these infrastructure need to be surveyed.

A related problem is when both, the map and the vehicle position are not known. In this case the vehicle start in an unknown location in an unknown environment and proceed to incrementally build a navigation map of the environment while simultaneously use this map to update its location. In this problem, vehicle and map estimates are highly correlated and cannot be obtained independently of one another. This problem is usually known as Simultaneous Localization and Map Building (SLAM) and was originally introduced [7]. During the past few years significant progress has been made towards the solution of the SLAM problem [3] [5] [2] [1] [6]

Kalman filter methods can also be extended to perform simultaneous localization and map building. There have been several applications of this technology in a number of different environments, such as indoors, underwater and outdoors. One of the main problems with the SLAM algorithm has been the computational requirements. Although the algorithm is originally of $O(N^3)$ the complexity of the SLAM algorithm can be reduced to $O(N^2)$, 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 N^2 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 map. This correlation is of fundamental importance for the long-term convergence of the algorithm and needs to be maintained for the full duration of the mission. This section presents an introduction to the SLAM problem, description of the computation complexity and different approaches that makes possible the implementation of SLAM in real time in very large environments.

1.1 Fundamentals of SLAM

The SLAM algorithm address the problem of a vehicle with a known kinematic model, starting at an unknown position, moving through an unknown environment populated with artificial or natural features. The objective of SLAM is then to localize the vehicle and at the same time build an incremental navigation map with the observed features. The vehicle is equipped with a sensor capable of taking measurement of the relative location of the feature and the vehicle itself. This scenario is shown if Figure 1. To facilitate the introduction of the SLAM equations a linear model for the vehicle and observation is used.

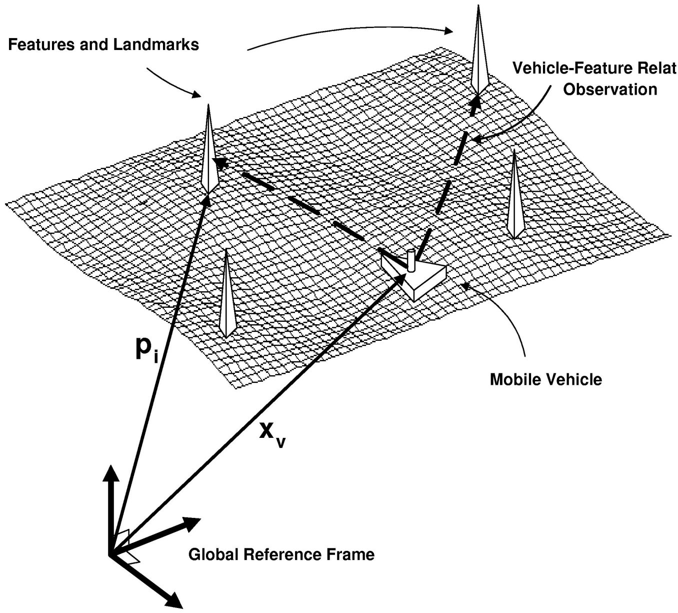


Figure 1: A vehicle taking relative measurements to environmental landmarks

1.1.1 Process Model

The state of the system consist of the position and orientation of the vehicle augmented with the position of the landmarks. Assuming that the state of the vehicle is given by $\mathbf{x}_v(k)$ the motion of the vehicle through the environment can be modeled by the following equation:

$$\mathbf{x}_v(k+1) = \mathbf{F}_v(k)\mathbf{x}_v(k) + \mathbf{u}_v(k+1)) + \mathbf{v}_v(k+1) \tag{1}$$

where $\mathbf{F}_v(k)$ is the state transition matrix, $\mathbf{u}_v(k)$ a vector of control inputs, and $\mathbf{v}_v(k)$ a vector of temporally uncorrelated process noise errors with zero mean and covariance $\mathbf{Q}_v(k)$ [[4]] for further details). The location of the i^{th} landmark is denoted $\mathbf{p}_i$. SLAM considers that all landmarks are stationary. The "state transition equation" for the i^{th} landmark is

$$\mathbf{p}_i(k+1) = \mathbf{p}_i(k) = \mathbf{p}_i, \tag{2}$$

It can be seen that the model for the evolution of the landmarks does not have any uncertainty. Assuming that N are actually validated and incorporated by the system then the vector of all N landmarks is denoted

$$\mathbf{p} = \left[\begin{array}{ccc} \mathbf{p}_{\mathit{1}}^T & \ldots & \mathbf{p}_N^T \end{array} \right]^T \tag{3}$$

The augmented state vector containing both the state of the vehicle and the state of all landmark locations is denoted

$$\mathbf{x}(k) = \left[\begin{array}{cccc} \mathbf{x}_v^T(k) & \mathbf{p}_{\mathit{1}}^T & \ldots & \mathbf{p}_N^T \end{array} \right]^T. \tag{4}$$

The augmented state transition model for the complete system may now be written as

$$\begin{bmatrix} \mathbf{x}_v(k+1) \\ \mathbf{p}_{\mathit{1}} \\ \vdots \\ \mathbf{p}_N \end{bmatrix} = \begin{bmatrix} \mathbf{F}_v(k) & 0 & \ldots & 0 \\ 0 & \mathbf{I}_{p_1} & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \mathbf{I}_{p_N} \end{bmatrix} \begin{bmatrix} \mathbf{x}_v(k) \\ \mathbf{p}_{\mathit{1}} \\ \vdots \\ \mathbf{p}_N \end{bmatrix} \tag{5}$$

$$+ \begin{bmatrix} \mathbf{u}_v(k+1) \\ \mathbf{0}_{p_1} \\ \vdots \\ \mathbf{0}_{p_N} \end{bmatrix} + \begin{bmatrix} \mathbf{v}_v(k+1) \\ \mathbf{0}_{p_1} \\ \vdots \\ \mathbf{0}_{p_N} \end{bmatrix} \tag{6}$$

$$\mathbf{x}(k+1) = \mathbf{F}(k)\mathbf{x}(k) + \mathbf{u}(k+1) + \mathbf{v}(k+1) \tag{7}$$

where $\mathbf{I}_{p_i}$ is the $\dim(p_i) \times \dim(p_i)$ identity matrix and $\mathbf{0}_{p_i}$ is the $\dim(p_i)$ null vector.

As can be seen from Equation 5 the size of the matrices involved were augmented by

$n * N$, being n the number of states required to represent a landmark and N the number of landmarks incorporated into the map. In a large environment this number will tend to grow and eventually the computer resources will not be sufficient to process the information from the external sensor in real time.

1.1.2 The Observation Model

The vehicle is equipped with a sensor that can obtain observations of the relative location of landmarks with respect to the vehicle. Assuming the observation to be linear and synchronous, the observation model for the i^{th} landmark is written in the form

$$\begin{aligned} \mathbf{z}_i(k) &= \mathbf{H}_i\mathbf{x}(k) + \mathbf{w}_i(k) & (8) \\ &= \mathbf{H}_{p_i}\mathbf{p} - \mathbf{H}_v\mathbf{x}_v(k) + \mathbf{w}_i(k) & (9) \end{aligned}$$

where $\mathbf{w}_i(k)$ is a vector of temporally uncorrelated observation errors with zero mean and variance $\mathbf{R}_i(k)$. The term $\mathbf{H}_i$ is the observation matrix and relates the output of the sensor $\mathbf{z}_i(k)$ to the state vector $\mathbf{x}(k)$ when observing the $i^(th)$ landmark. It is important to note that the observation model for the i^{th} landmark is written in the form

$$\mathbf{H}_i = [-\mathbf{H}_v, \mathbf{0}\cdots\mathbf{0}, \mathbf{H}_{p_i}, \mathbf{0}\cdots\mathbf{0}] \tag{10}$$

This structure reflects the fact that the observations are "relative" between the vehicle and the landmark, often in the form of relative location, or relative range and bearing (see Section 4).

1.2 The Estimation Process

In the estimation-theoretic formulation of the SLAM problem, the Kalman filter is used to provide estimates of vehicle and landmark location. We briefly summarise the notation and main stages of this process The Kalman filter recursively computes estimates for a state $\mathbf{x}(k)$ which is evolving according to the process model in Equation 5 and which is being observed according to the observation model in Equation 8. The Kalman filter computes an estimate which is equivalent to the conditional mean $\hat{\mathbf{x}}(p|q) = E\left[\mathbf{x}(p)|\mathbf{Z}^q\right]$ $(p \geq q)$, where $\mathbf{Z}^q$ is the sequence of observations taken up until time q. The error in the estimate is denoted $\tilde{\mathbf{x}}(p|q) = \hat{\mathbf{x}}(p|q) - \mathbf{x}(p)$. The Kalman filter also provides a recursive estimate of the covariance $\mathbf{P}(p|q) = E\left[\tilde{\mathbf{x}}(p|q)\tilde{\mathbf{x}}(p|q)^T|\mathbf{Z}^q\right]$ in the estimate $\hat{\mathbf{x}}(p|q)$. The Kalman filter algorithm now proceeds recursively in three stages:

- Prediction: Given that the models described in equations 5 and 8 hold, and that an estimate $\hat{\mathbf{x}}(k|k)$ of the state $\mathbf{x}(k)$ at time k together with an estimate of the covariance $\mathbf{P}(k|k)$ exist, the algorithm first generates a prediction for the state estimate, the observation (relative to the i^{th} landmark) and the state estimate covariance at time $k+1$ according to

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}(k)\hat{\mathbf{x}}(k|k) + \mathbf{u}(k) \tag{11}$$

$$\hat{\mathbf{z}}_i(k+1|k) = \mathbf{H}_i(k)\hat{\mathbf{x}}(k+1|k) \tag{12}$$

$$\mathbf{P}(k+1|k) = \mathbf{F}(k)\mathbf{P}(k|k)\mathbf{F}^T(k) + \mathbf{Q}(k), \tag{13}$$

respectively.

- Observation: Following the prediction, an observation $\mathbf{z}_i(k+1)$ of the i^{th} landmark of the true state $\mathbf{x}(k+1)$ is made according to Equation 8. Assuming correct landmark association, an innovation is calculated as follows

$$\nu_i(k+1) = \mathbf{z}_i(k+1) - \hat{\mathbf{z}}_i(k+1|k) \tag{14}$$

together with an associated innovation covariance matrix given by

$$\mathbf{S}_i(k+1) = \mathbf{H}_i(k)\mathbf{P}(k+1|k)\mathbf{H}_i^T(k) + \mathbf{R}_i(k+1). \tag{15}$$

- Update: The state estimate and corresponding state estimate covariance are then updated according to:

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}_i(k+1)\nu_i(k+1) \tag{16}$$

$$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{W}_i(k+1)\mathbf{S}(k+1)\mathbf{W}_i^T(k+1) \tag{17}$$

Where the gain matrix $\mathbf{W}_i(k+1)$ is given by

$$\mathbf{W}_i(k+1) = \mathbf{P}(k+1|k)\mathbf{H}_i^T(k)\mathbf{S}_i^{-1}(k+1) \tag{18}$$

The update of the state estimate covariance matrix is of paramount importance to the SLAM problem. Understanding the structure and evolution of the state covariance matrix is the key component to this solution of the SLAM problem.

1.2.1 Example

Assume a vehicle moving in one dimension and observing relative range to a number of landmarks. We would like to design a filter to track the position and velocity of the vehicle. We can also assume that the position of the vehicle is known with some uncertainty, although this is not relevant for this example. Since no additional absolute information is available such as GPS, if one assume the vehicle is traveling at constant velocity the estimation of uncertainty of its position will grow with time. In addition due to initial error in position and velocity the difference between estimated and real position will grow indefinite.

$$\begin{bmatrix} pos(k+1) \\ vel(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} pos(k) \\ vel(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v(k) \tag{19}$$

In this example we can assume that two landmarks are already incorporate into the map. The process model is then extended as follows:

$$\begin{bmatrix} pos(k+1) \\ vel(k+1) \\ p_1(k+1) \\ p_2(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} pos(k) \\ vel(k) \\ p_1(k) \\ p_2(k) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} v(k) \tag{20}$$

Once a landmark is incorporated into the map it will remain as part of the state vector. The full augmented system needs to be used each time a prediction or observation is made. In future section we will see that optimizations are possible to reduce the computation complexity of SLAM. On the other hand, the observation model will be a function of the number of landmarks observed. In the case two landmarks are observed we have:

$$\begin{bmatrix} z_1(k) \\ z_2(k) \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} pos(k) \\ vel(k) \\ p_1(k) \\ p_2(k) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} w(k) \tag{21}$$

In this example it was also assumed that both observation were taken with the same sensor or with another sensor with similar noise characteristics.

1.3 Fundamentals results in SLAM

In this section section present three fundamental results in the solution of SLAM. For a full demonstration of this results the readers are encouraged to see [1]

The state covariance matrix may be written in block form as

$$\mathbf{P}(i|j) = \begin{bmatrix} \mathbf{P}_{vv}(i \mid j) & \mathbf{P}_{vm}(i \mid j) \\ \mathbf{P}_{mv}(i \mid j) & \mathbf{P}_{mm}(i \mid j)), \end{bmatrix}$$

where $\mathbf{P}_{vv}(i \mid j)$ is the error covariance matrix associated with the vehicle state estimate, $\mathbf{P}_{mm}(i \mid j)$ is the map covariance matrix associated with the landmark state estimates, and $\mathbf{P}_{vm}(i \mid j)$ is the cross-covariance matrix between vehicle and landmark states.

Theorem 1 *The determinant of any sub-matrix of the map covariance matrix decreases monotonically as successive observations are made.*

The algorithm is initialised using a positive semi-definite (*psd*) state covariance matrix $\mathbf{P}(0|0)$. The matrices $\mathbf{Q}$ and $\mathbf{R}_i$ are both *psd*, and consequently the matrices $\mathbf{P}(k+1|k)$, $\mathbf{S}_i(k+1), \mathbf{W}_i(k+1)\mathbf{S}_i(k+1)\mathbf{W}_i^T(k+1)$ and $\mathbf{P}(k+1|k+1)$ are all *psd*. From Equation 17, and for any landmark i,

$$\begin{aligned}\det \mathbf{P}(k+1|k+1) &= \det(\mathbf{P}(k+1|k) - \mathbf{W}_i(k+1)\mathbf{S}_i(k+1)\mathbf{W}^T(k+1)) \\ &\leq \det \mathbf{P}(k+1|k) \end{aligned} \tag{22}$$

The determinant of the state covariance matrix is a measure of the volume of the uncertainty ellipsoid associated with the state estimate. Equation 22 states that the total uncertainty of the state estimate does not increase during an update.

Theorem 2 *In the limit the landmark estimates become fully correlated*

As the number of observations taken tends to infinity a lower limit on the map covariance limit will be reached such that

$$\lim_{k\to\infty} [\mathbf{P}_{mm}(k+1 \mid k+1)] = \mathbf{P}_{mm}(k \mid k) \tag{23}$$

Also the limit the determinant of the covariance matrix of a map containing more than one landmark tends to zero.

$$\lim_{k\to\infty} [det\mathbf{P}_{mm}(k \mid k)] = 0 \tag{24}$$

This result implies that the landmarks become progressively more correlated as successive observations are made. In the limit then, given the exact location of one landmark the location of all other landmarks can be deduced with absolute certainty and the map is fully correlated.

Theorem 3 *In the limit, the lower bound on the covariance matrix associated with any single landmark estimate is determined only by the initial covariance in the vehicle estimate $\mathbf{P}_{0v}$ at the time of the first sighting of the first landmark.*

When the process noise is not zero the two competing effects of loss of information content due to process noise and the increase in information content through observations, determine the limiting covariance. It is important to note that the limit to the covariance applies because all the landmarks are observed and initialised solely from the observations made from the vehicle. The covariances of landmark estimates can not be further reduced by making additional observations to previously unknown landmarks. However, incorporation of external information, for example using an observation is made to a landmark whose location is available through external means such as GPS, will reduce the limiting covariance.

In summary, the three theorems presented above describe, in full, the convergence properties of the map and its steady state behaviour. As the vehicle progresses through the environment the total uncertainty of the estimates of landmark locations reduces monotonically to the point where the map of relative locations is known with absolute precision. In the limit, errors in the estimates of any pair of landmarks become fully correlated. This means that given the exact location of any one landmark, the location of any other landmark in the map can also be determined with absolute certainty. As the map converges in the above manner, the error in the absolute location estimate of every landmark (and thus the whole map) reaches a lower bound determined only by the error that existed when the first observation was made.

Thus a solution to the general SLAM problem exists and it is indeed possible to construct a perfectly accurate map describing the relative location of landmarks and simultaneously compute vehicle position estimates without any prior knowledge of landmark or vehicle locations.

1.4 Non-linear Models

In general the models that predict the trajectory of the vehicle and the models that relates the observation with the states are non-linear. The SLAM can still be formulated but requires the linearization of these models. In this case the Jacobian of the process and observation models are used to propagate the covariances. In this section we present a more realistic model of a standard outdoor vehicle.

Assume a vehicle equipped with dead reckoning capabilities and an external sensor capable of measuring relative distance between vehicle and the environment as shown in Figure 2. The steering control α, and the speed υ_c are used with the kinematic model to predict the position of the vehicle. In this case the external sensor returns range and bearing information to the different features $Bi_{(i=1..n)}$. This information is obtained with respect to the vehicle coordinates (x_l, y_l), that is $\mathbf{z}(k) = (r, \beta)$, where r is the distance from the beacon to the range sensor, β is the sensor bearing measured with respect to the vehicle coordinate frame.

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

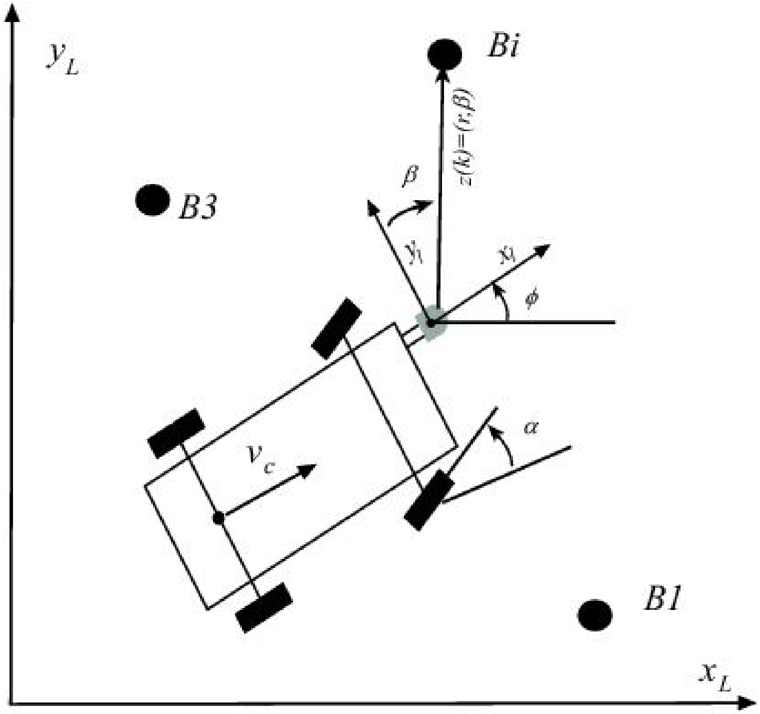


Figure 2: Vehicle Coordinate System

$$\begin{bmatrix} \dot{x}_c \\ \dot{y}_c \\ \dot{\phi}_c \end{bmatrix} = \begin{bmatrix} v_c \cdot \cos(\phi) \\ v_c \cdot \sin(\phi) \\ \frac{v_c}{L} \cdot \tan(\alpha) \end{bmatrix} + \gamma \tag{25}$$

Where L is the distance between wheel axles as shown in Figure 3. To simplify the equation in the update stage, the kinematic model of the vehicle is designed to represent the trajectory of the centre of the laser. Based on Figure 2 and 3, the translation of the centre of the back axle can be given

$$P_L = P_C + a \cdot \vec{T}_\phi + b \cdot \vec{T}_{\phi + \pi/2} \tag{26}$$

P_L and P_C are the position of the laser and the centre of the back axle in global coordinates respectively. The transformation is defined by the orientation angle, according to the following vectorial expression:

$$\vec{T}_\phi = (\cos(\phi), \sin(\phi)) \tag{27}$$

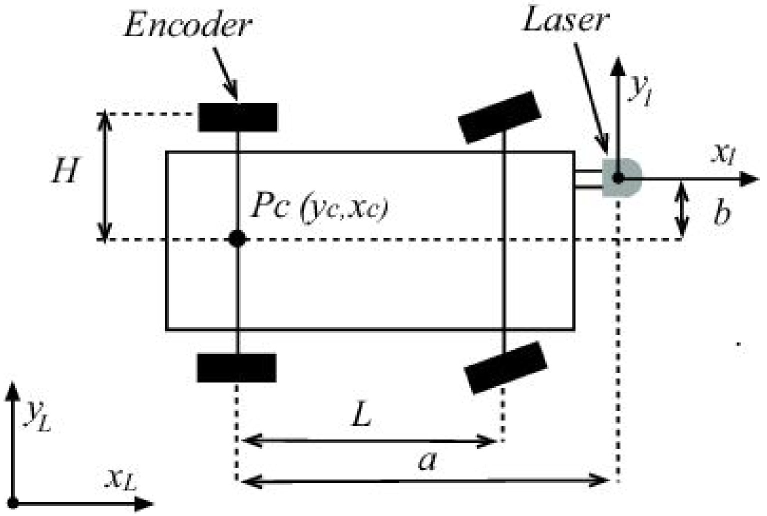


Figure 3: Kinematics parameters

The scalar representation is

$$x_L = x_c + a \cdot \cos(\phi) + b \cdot \cos(\phi + {}^{\pi}\!/_{2})$$
$$y_L = y_c + a \cdot \sin(\phi) + b \cdot \sin(\phi + {}^{\pi}\!/_{2})$$

Finally the full state representation can be written

$$\begin{bmatrix} \dot{x}_L \\ \dot{y}_L \\ \dot{\phi}_L \end{bmatrix} = \begin{bmatrix} v_c \cdot \cos(\phi) - \frac{v_c}{L} \cdot (a \cdot \sin(\phi) + b \cdot \cos(\phi)) \cdot \tan(\alpha) \\ v_c \cdot \sin(\phi) + \frac{v_c}{L} \cdot (a \cdot \cos(\phi) - b \cdot \sin(\phi)) \cdot \tan(\alpha) \\ \frac{v_c}{L} \cdot \tan(\alpha) \end{bmatrix} + \gamma \tag{28}$$

The velocity, v_c , is measured with an encoder located in the back left wheel. This velocity is translated to the centre of the axle with the following equation:

$$v_c {=} \frac{\nu_{\mathrm{e}}}{\left(1 - \tan(\alpha) \cdot \frac{H}{L}\right)} \tag{29}$$

Where for this car H = 0.75m, L=2.83 m, b = 0.5 and a = L + 0.95m. Finally the discrete model in global coordinates can be approximated with the following set of equations:

$$\begin{bmatrix} x(k) \\ y(k) \\ \phi(k) \end{bmatrix} = \begin{bmatrix} x(k-1) + \Delta T\, v_c(k-1) \cdot \cos\left(\phi(k-1)\right) - \\ \frac{v_c}{L} \cdot \left(a \cdot \sin\left(\phi(k-1)\right) + b \cdot \cos\left(\phi(k-1)\right)\right) \\ \cdot \tan\left(\alpha(k-1)\right) \\ y(k-1) + \Delta T\, v_c(k-1) \cdot \sin\left(\phi(k-1)\right) + \\ \frac{v_c(k-1)}{L} \cdot \left(a \cdot \cos\left(\phi(k-1)\right) - b \cdot \sin\left(\phi(k-1)\right)\right) \\ \cdot \tan\left(\alpha(k-1)\right) \\ \frac{v_c(k-1)}{L} \cdot \tan\left(\alpha(k-1)\right) \end{bmatrix} + \gamma \qquad (30)$$

where Δt is the sampling time, that in this case is not constant.

The observation equation relating the vehicle states to the observations is

$$z = h\left(X, x_i, y_i\right) = \begin{bmatrix} z_r^i \\ z_\beta^i \end{bmatrix} = \begin{bmatrix} \sqrt{\left(x_i - x_L\right)^2 + \left(y_i - y_L\right)^2} \\ atan\left(\frac{(y_i - y_L)}{(x_i - x_L)}\right) - \phi_L + {}^{\pi}\!/_{2} \end{bmatrix} + \gamma_h \qquad (31)$$

where z is the observation vector, is the coordinates of the landmarks, x_L , y_L and ϕ_L are the vehicle states defined at the external sensor location and γ_h the sensor noise.

The complete non-linear model can be expressed in general form as:

$$\begin{array}{l} X\left(k+1\right) = F\left(X\left(k\right), u\left(k\right) + \gamma_u\left(k\right)\right) + \gamma_f\left(k\right) \\ z\left(k\right) = h\left(X\left(k\right)\right) + \gamma_h\left(k\right) \end{array} \qquad (32)$$

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

$$\begin{array}{l} F\left(X\left(k\right), u\left(k\right) + \gamma_u\left(k\right)\right) + \gamma_f\left(k\right) \cong F\left(X\left(k\right), u\left(k\right)\right) + \gamma\left(k\right) \\ \gamma\left(k\right) = J_u \cdot \gamma_u\left(k\right) + \gamma_f\left(k\right) \\ J_u = \left.\frac{\partial F}{\partial u}\right|_{X = X(k), u = u(k)} \end{array} \qquad (33)$$

The matrix noise characteristics are assumed zero mean and white:

$$\begin{array}{l} E\left\{\gamma_f\left(k\right)\right\} = E\left\{\gamma_u\left(k\right)\right\} = E\left\{\gamma_h\left(k\right)\right\} = 0 \\ E\left\{\gamma_f\left(i\right) \cdot \gamma_f^T\left(j\right)\right\} = \delta_{i,j} \cdot Q_f\left(i\right) \\ E\left\{\gamma_h\left(i\right) \cdot \gamma_h^T\left(j\right)\right\} = \delta_{i,j} \cdot R\left(i\right) \\ E\left\{\gamma_u\left(i\right) \cdot \gamma_u^T\left(j\right)\right\} = \delta_{i,j} \cdot Q_u\left(i\right) \\ E\left\{\gamma_h\left(i\right) \cdot \gamma^T\left(j\right)\right\} = 0 \\ \delta_{i,j} = \begin{cases} 0 & i \neq j \\ 1 & i = j \end{cases} \end{array} \qquad (34)$$

$$E\left\{\gamma\left(i\right) \cdot \gamma^T\left(j\right)\right\} = \delta_{i,j} \cdot \left(J_u \cdot Q_u\left(i\right) \cdot J_u^T + Q_f\left(i\right)\right) = \delta_{i,j} \cdot Q\left(i\right)$$

An Extended Kalman Filter (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 $k-1$,

$$\begin{array}{l} X\left(k+1,k\right)=F\left(X\left(k,k\right),u\left(k\right)\right) \\ P\left(k+1,k\right)=J\cdot P\left(k,k\right)\cdot J^{T}+Q\left(k\right) \end{array} \tag{35}$$

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

$$\begin{array}{l} S\left(k+1\right)=H\cdot P\left(k+1,k\right)\cdot H^{T}(k+1)+R\left(k+1\right) \\ W\left(k+1\right)=P\left(k+1,k\right)\cdot H^{T}(k+1)\cdot S^{-1}\left(k+1\right) \\ \vartheta\left(k+1\right)=Z\left(k+1\right)-h\left(X\left(k+1,k\right)\right) \\ X\left(k+1,k+1\right)=X\left(k+1,k\right)+W\left(k+1\right)\cdot\vartheta\left(k+1\right) \\ P\left(k+1,k+1\right)=P\left(k+1,k\right)-W\left(k+1\right)\cdot S\left(k+1\right)\cdot W\left(k+1\right)^{T} \end{array} \tag{36}$$

Where

$$J=J\left(k\right)=\left.\frac{\partial F}{\partial X}\right|_{(X,u)=(X(k),u(k))}, \qquad H=H\left(k\right)=\left.\frac{\partial h}{\partial X}\right|_{X=X(k)} \tag{37}$$

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

Under the SLAM framework the system will detect new features at the beginning of the mission and when exploring 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 then given by:

$$\begin{array}{l} X=\begin{bmatrix} X_L \\ X_I \end{bmatrix} \\ X_L=\left(x_L,y_L,\phi_L\right)^{T}\in R^{3} \\ X_I=\left(x_1,y_1,..,x_N,y_N\right)^{T}\in R^{2N} \end{array} \tag{38}$$

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 consider to be static the dynamic model that includes the new states becomes:

$$\begin{array}{l} X_L\left(k+1\right)=f\left(X_L\left(k\right)\right)+\gamma \\ X_I\left(k+1\right)=X_I\left(k\right) \end{array} \tag{39}$$

It is important to remarks that the landmarks are assumed to be static. Then the Jacobian matrix for the extended system is

$$\begin{array}{l} \frac{\partial F}{\partial X} = \begin{bmatrix} \frac{\partial f}{\partial \tilde{x}_L} & \emptyset \\ \emptyset & I \end{bmatrix} = \begin{bmatrix} J_1 & \emptyset \\ \emptyset & I \end{bmatrix} \\ J_1 \in R^{3x3}, \quad \emptyset \in R^{3xN}, \quad I \in R^{2Nx2N} \end{array} \tag{40}$$

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 Equation 31 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:

$$\frac{\partial h}{\partial X} = \begin{bmatrix} \frac{\partial z_r}{\partial X} \\ \frac{\partial z_\beta}{\partial X} \end{bmatrix} = \begin{bmatrix} \frac{\partial r_i}{\partial \left(x_L, y_L, \phi_L, \{x_j, y_j\}_{j=1..N}\right)} \\ \frac{\partial \beta_i}{\partial \left(x_L, y_L, \phi_L, \{x_j, y_j\}_{j=1..N}\right)} \end{bmatrix} \tag{41}$$

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:

$$\begin{bmatrix} \frac{\partial z_r}{\partial X} \\ \frac{\partial z_\beta}{\partial X} \end{bmatrix} = \begin{bmatrix} \frac{\Delta x}{\Delta} & \frac{\Delta y}{\Delta} & 0 & 0 & 0 & ... & -\frac{\Delta x}{\Delta} & -\frac{\Delta y}{\Delta} & 0 & ... & 0 & 0 \\ -\frac{\Delta y}{\Delta^2} & \frac{\Delta x}{\Delta^2} & -1 & 0 & 0 & ... & \frac{\Delta y}{\Delta^2} & -\frac{\Delta x}{\Delta^2} & 0 & ... & 0 & 0 \end{bmatrix} \tag{42}$$

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

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.

1.5 Optimization of SLAM

Under the SLAM framework the size of the state vector is equal to the number of the 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 section we present a series of optimizations in the prediction and update stages that reduce the complexity of the SLAM algorithm from $O(M^3)$ to $O(M^2)$. Then a compressed filter is presented to reduce the real time computation requirement to $O(2N_a^2)$, being N_a the landmarks in the local area. This will also make the SLAM algorithm extremely efficient when the vehicle remains navigation 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 exactly equivalent to a full SLAM implementation.

1.5.1 Standard Algorithm Optimization

Prediction Stage Considering the zeros in the Jacobian matrix of Equation 40 the prediction Equation 35 can be written:

$$
\begin{aligned}
&P^{+} = J \cdot P \cdot J^{T} + Q = \left[\begin{array}{cc} J_1 & \emptyset \\ \emptyset^T & I \end{array}\right] \cdot \left[\begin{array}{cc} P_{11} & P_{12} \\ P_{21} & P_{22} \end{array}\right] \cdot \left[\begin{array}{cc} J_1^T & \emptyset^T \\ \emptyset & I^T \end{array}\right] + \left[\begin{array}{cc} Q_V & \emptyset \\ \emptyset & \emptyset_2 \end{array}\right] \\
&J_1 \in R^{3x3}, \quad \emptyset \in R^{3x2N}, \quad I \in R^{2Nx2N} \\
&P_{11} \in R^{3x3}, \quad P_{12} \in R^{3x2N}, \quad P_{21} = P_{12}^T, \quad P_{22} \in R^{2Nx2N}
\end{aligned} \tag{43}
$$

The time sub-indexes are not used in this explanation for clarity of presentation. Performing the matrix operations explicitly the following result is obtained:

$$
\begin{aligned}
&J \cdot P = \left[\begin{array}{cc} J_1 & \emptyset \\ \emptyset^T & I \end{array}\right] \cdot \left[\begin{array}{cc} P_{11} & P_{12} \\ P_{21} & P_{22} \end{array}\right] = \left[\begin{array}{cc} J_1 \cdot P_{11} & J_1 \cdot P_{12} \\ I \cdot P_{21} & I \cdot P_{22} \end{array}\right] = \left[\begin{array}{cc} J_1 \cdot P_{11} & J_1 \cdot P_{12} \\ P_{21} & P_{22} \end{array}\right] \\
&J \cdot P \cdot J^T = \left[\begin{array}{cc} J_1 \cdot P_{11} & J_1 \cdot P_{12} \\ P_{21} & P_{22} \end{array}\right] \cdot \left[\begin{array}{cc} J_1^T & \emptyset^T \\ \emptyset & I \end{array}\right] = \\
&\left[\begin{array}{cc} J_1 \cdot P_{11} \cdot J_1^T & J_1 \cdot P_{12} \cdot I \\ P_{21} \cdot J_1^T & P_{22} \cdot I \end{array}\right] = \left[\begin{array}{cc} J_1 \cdot P_{11} \cdot J_1^T & J_1 \cdot P_{12} \\ (J_1 \cdot P_{12})^T & P_{22} \end{array}\right]
\end{aligned} \tag{44}
$$

It can be proved that the evaluation of this matrix requires approximately only $9M$ multiplications. In general, more than one prediction step is executed between 2 update steps. This is due to the fact that the prediction stage is usually driven by high frequency sensory information that acts as inputs to the dynamic model of the vehicle and 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 much lower frequency. For example, the steering angle and wheel speed can be sampled every 20 milliseconds but the laser frames can be obtained with a sample time of 200 milliseconds. In this case we have a ratio of approximately 10 prediction steps to one update step. The compact form for n prediction steps without an update is

$$P\left(k+n,k\right)=\left[\begin{array}{cc} P_{11}\left(k+n,k\right) & G_1\cdot P_{12}\left(k,k\right) \\ \left(G_1\cdot P_{12}\left(k,k\right)\right)^T & P_{22}\left(k,k\right) \end{array}\right] \tag{45}$$

where

$$G_1=G_1\left(k,n\right)=\prod_{i=0}^{n-1}J_1\left(k+i\right)=J_1\left(k+n-1\right)\cdot ....\cdot J_1\left(k\right) \tag{46}$$

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

$$\begin{array}{l} H=H\left(k\right)=\frac{\partial h}{\partial X}\Big|_{X=X(k)}=\left[H_1,\emptyset_1,H_2,\emptyset_2\right]\in R^{2xM}, \quad M=\left(2N+3\right) \\ H_1=\frac{\partial h}{\partial X_L}\Big|_{X=X(k)}=\frac{\partial h}{\partial\left(x_L,y_L,\phi_L\right)}\Big|_{X=X(k)}\in R^{2x3} \\ H_2=\frac{\partial h}{\partial X_i}\Big|_{X=X(k)}=\frac{\partial h}{\partial\left(x_i,y_i\right)}\Big|_{X=X(k)}\in R^{2x2} \\ \emptyset_1,\emptyset_2=\text{nullmatrices}\left(\frac{\partial h}{\partial X_j}=\emptyset \quad \forall j\neq i\right). \end{array} \tag{47}$$

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

$$\begin{array}{l} P\cdot H^T=P_1\cdot H_1^T+P_2\cdot H_2^T \\ P_1\in R^{Mx3} \quad , \quad P_2\in R^{Mx2} \end{array}$$

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

$$\begin{array}{l} S=H\cdot P\cdot H^T+R \quad \in R^{2*2} \\ W=P\cdot H^T\cdot S^{-1}\in R^{Mx2} \end{array} \tag{48}$$

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 $O(M^2)$.

Experimental results The SLAM algorithm presented were tested an outdoor environment with a standard utility vehicle retrofitted with dead reckoning sensors and a laser range sensor as shown in Figure 4. In this application the most common relevant feature in the environment were trees. The profiles of trees were extracted from the laser information. A Kalman filter was also 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 minutes. Figure 5 presents the vehicle trajectory and navigation landmarks incorporated into the relative map. This run includes all the features in the environment and the optimisation presented. The system built a map of the environment and localized 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. Figure 6 presents the estimated error of the vehicle position and selected landmarks. The states corresponding to the vehicle presents oscillatory behaviour 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. Figure 7 presents the final estimation of the landmarks in the map. It can be seen that after 20 minutes the estimated error of all the landmarks are below 60 cm.

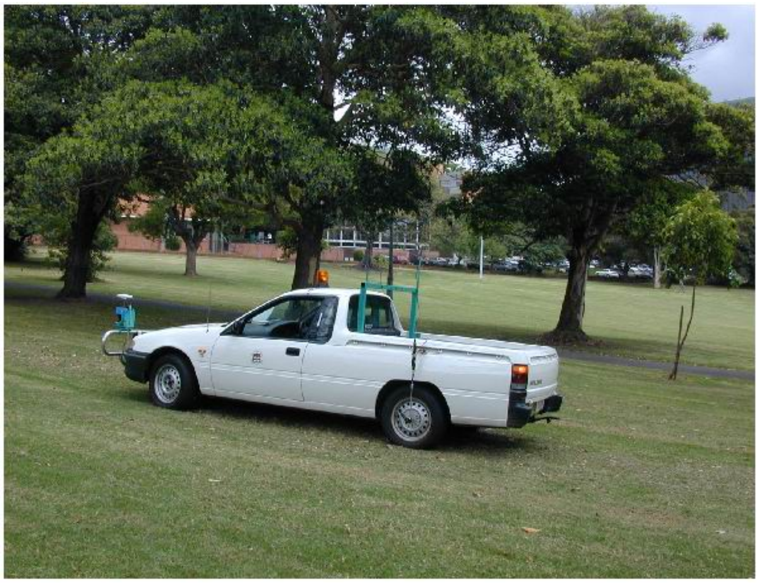


Figure 4: Utility car used for the experiments. The vehicle is equipped with a Sick laser range and bearing sensor, linear variable differential transformer sensor for the steering and back wheel velocity encoders.

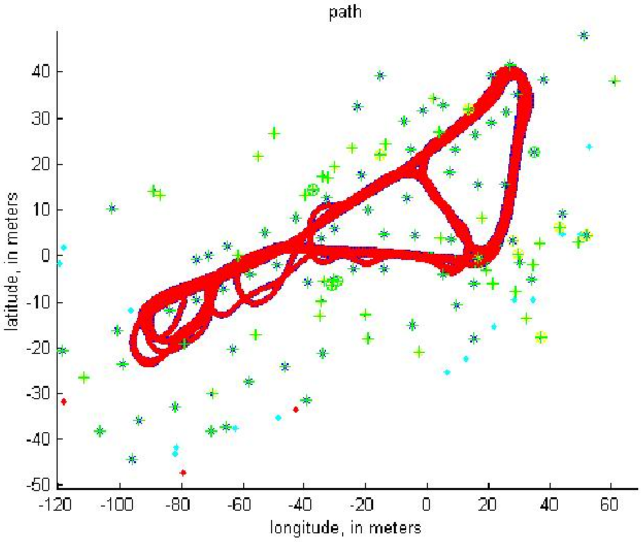


Figure 5: Vehicle trajectory and landmarks. The '*' shows the estimated position of objects that qualified as landmarks for the navigation system. The dots are laser returns that are not stable enough to qualify as landmarks. The solid line shows the 20 minutes vehicle trajectory estimation using full SLAM.

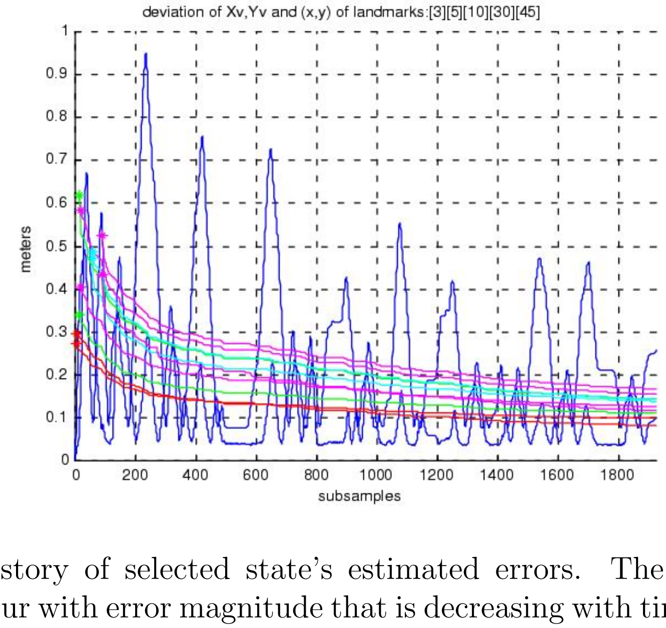


Figure 6: The History of selected state's estimated errors. The vehicle states shows oscillatory behaviour with error magnitude that is decreasing with time due to the learning of the environment. The landmarks always present a exponential decreasing estimated error with a limit of the initial uncertainty of the vehicle position.

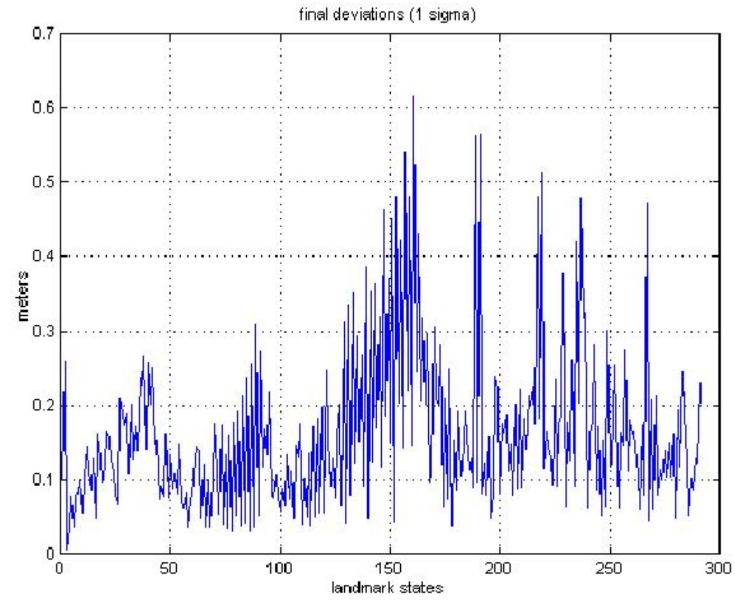


Figure 7: 11 Final estimated error of all states. For each state the final estimated error is presented. The maximum error is approximately 60 cm

1.5.2 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 result 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 Figure 8 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 also 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 to 30 Hz. To incorporate this information using 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 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 cost. Another important implication

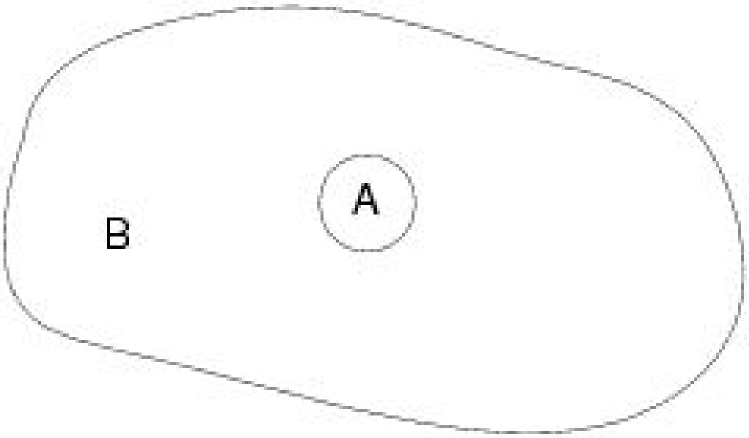


Figure 8: Local and Global areas

is that the global map will not be required to update sequentially at the same rate of the local map.

Update step Considered the states divided in two groups:

$$X = \begin{bmatrix} X_A \\ X_B \end{bmatrix}, \quad X_A \in R^{2N_A+3}, \quad X_B \in R^{2N_B}, \qquad X \in R^{2N-3}, \quad N = N_A + N_B \tag{49}$$

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 with the states X_A and do not involve states of X_B, that is

$$h(X) = h(X_A) \tag{50}$$

Then at a given time k

$$H = \left.\frac{\partial h}{\partial X}\right|_{X=X(k)} = \left.\frac{\partial h}{\partial (X_A, X_B)}\right|_{X=X(k)} = \begin{bmatrix} \frac{\partial h}{\partial X_A} & \frac{\partial h}{\partial X_B} \end{bmatrix} = \begin{bmatrix} H_a & 0 \end{bmatrix} \tag{51}$$

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

$$
\begin{aligned}
& P = \begin{bmatrix} P_{aa} & P_{ab} \\ P_{ba} & P_{bb} \end{bmatrix} \qquad P \cdot H^T = \begin{bmatrix} P_{aa} \cdot H_a^T \\ P_{ba} \cdot H_a^T \end{bmatrix} \\
& H \cdot P \cdot H^T = H_a \cdot P_{aa} \cdot H_a^T \qquad S = H_a \cdot P_{aa} \cdot H_a^T + R \\
& W = P \cdot H^T \cdot S^{-1} = \begin{bmatrix} P_{aa} \cdot H_a^T \cdot S^{-1} \\ P_{ba} \cdot H_a^T \cdot S^{-1} \end{bmatrix} = \begin{bmatrix} W_a \\ W_b \end{bmatrix}
\end{aligned}
\tag{52}
$$

From these equations it is possible to see that

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

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

$$
dP = W \cdot S \cdot W^T = \begin{bmatrix} P_{aa} \cdot \kappa \cdot P_{aa} & \xi \cdot P_{ab} \\ (\xi \cdot P_{ab})^T & P_{ba} \cdot \kappa \cdot P_{ab} \end{bmatrix} \tag{53}
$$

with $\kappa = H_a^T \cdot S^{-1} \cdot H_a$ and $\xi = P_{aa} \cdot \kappa$. 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

$$
\begin{aligned}
& P\,(k+1,k+1) = P\,(k+1,k) - dP(k+1,k) \\
& P_{aa}(k+1,k+1) = P_{aa}(k+1,k) - P_{aa}(k+1,k) \cdot \kappa\,(k) \cdot P_{aa}(k+1,k) \\
& P_{ab}(k+1,k+1) = P_{ab}(k+1,k) - \xi(k) \cdot P_{ab}^T(k+1,k) \\
& = (I - \xi(k)) \cdot P_{ab}(k+1,k) \\
& P_{bb}(k+1,k+1) = P_{bb}(k+1,k) - P_{ba}(k+1,k) \cdot \kappa\,(k) \cdot P_{ab}(k+1,k)
\end{aligned}
\tag{54}
$$

And the covariance variation after t consecutive updates:

$$
\begin{aligned}
& P_{ab}(k+t,k+t) = \Phi(k+t-1) \cdot P_{ab}(k,k) \\
& P_{bb}(k+t,k+t) = P_{bb}(k,k) - P_{ba}(k,k) \cdot \psi\,(k-1) \cdot P_{ab}(k,k)
\end{aligned}
\tag{55}
$$

with

$$\begin{array}{l}\Phi(k+t) = (I - \xi(k+t)) \cdot (I - \xi(k+t-1)) \cdot .... \cdot (I - \xi(k)) = \prod_{i=k}^{k+t} (I - \xi(i)) \\ \psi(k+t) = \sum_{i=k}^{k+t} \left(\Phi^T(i-1) \cdot \kappa(i) \cdot \Phi(i-1) \right) \\ \Phi(k-1) = I, \quad \psi(k-1) = 0 \end{array} \tag{56}$$

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

$$\begin{array}{l}\Phi(k+t) = (I - \xi(k+t)) \cdot \Phi(k+t-1) \\ \psi(k+t) = \psi(k+t-1) + \Phi^T(k+t-1) \cdot \kappa(k+t) \cdot \Phi(k+t-1)\end{array} \tag{57}$$

with $\Phi(k), \psi(k), \kappa(k), \xi(k) \in R^{2Na \times 2Na}$. 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 $N_a << Nb < M$. The matrices $\xi(k)$ and $\kappa(k)$ are sparse and the calculation of $\Phi(k)$ and $\Psi(k)$ has complexity $O(N_a^2)$. 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 states 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

$$X_b(k+t, k+t) = X_b(k+t, k) - P_{ba}(k,k) \cdot \theta(k+t) \tag{58}$$

with $\theta(k+t) = \sum_{i=k}^{k+t-1} \Phi^T(i-1) \cdot H_a^T(i) \cdot S^{-1}(i) \cdot \vartheta(i)$, m the number of observations, in this case range and bearing, $\theta(k) \in R^{2Na \times m}$, $Z(k) \in R^m, \Phi(k) \in R^{2Na \times 2Na}$, $H_a(k) \in R^{m \times 2Na}$ and $S(k) \in R^{m \times m}$. Similarly, since H_a is a sparse matrix, the evaluation cost of the matrix θ is proportional to N_a. The derivation of these equations is presented in [2]

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 Equations (35) and (36). 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:

$$
\begin{array}{l}
\text{Predictionstep}\left\{
\begin{array}{c}
\Phi\left(k\right)=J_{aa}\left(k,k-1\right)\cdot\Phi\left(k-1\right)\\
\psi\left(k\right)=\psi\left(k-1\right)\\
\psi\left(0\right)=I\\
\Phi\left(0\right)=0
\end{array}\right.\\
\text{Updatestep}\left\{
\begin{array}{c}
\Phi\left(k\right)=\left(I-\xi(k)\right)\cdot\Phi\left(k-1\right)\\
\psi\left(k\right)=\psi\left(k\right)+\Phi^{T}(k-1)\cdot\kappa(k)\cdot\Phi(k-1)
\end{array}\right.
\end{array}
\tag{59}
$$

When a full update is required the global covariance matrix P and state X is updated with equations (55) and (58) respectively. At this time the matrices Φ and Ψ are also re-set to the initial values of time 0, since the next observations will be function of a given set of landmarks.

Map Management It has been shown 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 map management method is presented in Figure 9 . 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 9 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 close connected region. In case of strong updates, where the estimated position of the landmarks changes significantly, the owners of those landmarks can also be changed. An 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 length of the regions are smaller that 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

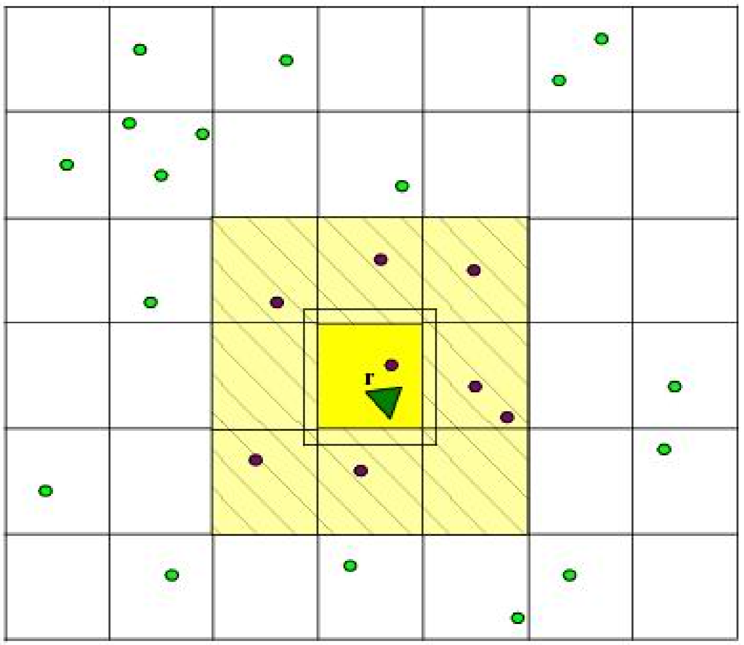


Figure 9: Map Management for the compressed Algorithm. The local area is composed of nine squares of length approximate of the range of the sensor. The vehicle is always within the central region of inside the threshold area.

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. Figure 9 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 9 neighbour regions. A full SLAM update is only required when the vehicle leaves the region.

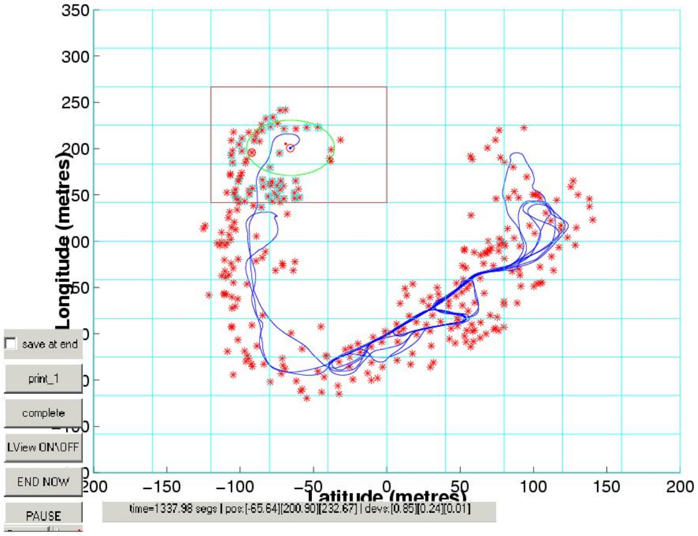
Computational Cost The total computational requirement for this algorithm is of $O(N_a^2)$ and the cost of the update when the vehicle leaves the local area is of $O(N_a N_b^2)$. Provided that the vehicle remains for a period of time in a given area, the computational saving will be considerable. This 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 at a transition, 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.

Experimental Results The compressed algorithm was implemented using local regions of 40x40 meters square. These regions are appropriate for the Sick laser range sensor used in this experiment. Figure 10 shows part of the trajectory of the vehicle with the local area composed of 9 squares surrounding the vehicle. To demonstrate that the compressed algorithm 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. Figure 11 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 landmark 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 Figure 12 where at the full update time stamps both estimated covariances become identical. Figure 13 shows the difference between full SLAM and compressed filter estimated landmarks covariance. It can be seen that at the full update time stamps the difference between the estimation using both algorithms becomes zero as demonstrated before. This shows 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.

1.5.3 Sub-Optimal SLAM

In this section we present a series of simplification that can further reduce the computationally complexity of SLAM. This sub-optimal 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

$$\begin{aligned} P &= P - \Delta P \\ \Delta P &= W \cdot S \cdot W^T \end{aligned} \tag{60}$$



time=1337.98 segs | pos:[-65.64][200.90][232.67] | devs:[0.85][0.24][0.01]

Figure 10: 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 button corner of the figure

The time subindexes are neglected when possible to simplify the equations. The state vector can be divided in 2 groups, the Preserved "P" and the Discarded "D" states

$$X = \begin{bmatrix} X_P \\ X_D \end{bmatrix}, \quad X_P \in R^{N_P}, \quad X_D \in R^{N_D}, \quad X \in R^N, \quad N = N_P + N_D \tag{61}$$

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 sub-vector. The covariance matrix can then be written in the following form:

$$P = \begin{bmatrix} P_{PP} & P_{PD} \\ P_{DP}^T & P_{DD} \end{bmatrix}, \quad \Delta P = \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP}^T & \Delta P_{DD} \end{bmatrix} = W \cdot S \cdot W^T \tag{62}$$

Conservative updates are obtained if the nominal update matrix ΔP is replaced by the sub-optimal ΔP^*

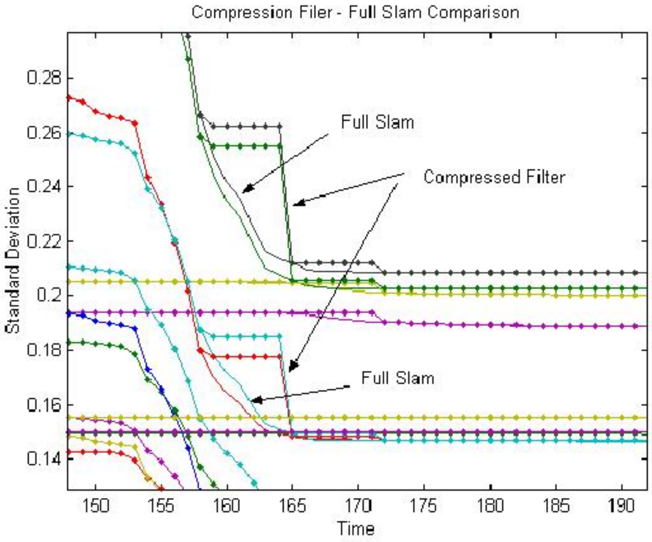


Figure 11: Landmark estimated position error for full Slam and compressed filter. The solid line shows the estimated error provided by the full SLAM algorithm. This algorithm updates all the landmarks with each observation. The dotted line shows the estimated error provided by the compressed filter. The landmark that are not in the local area are only updated when the vehicle leaves the local area. At this time a full updates is performed and the estimated error becomes exactly equal to full SLAM

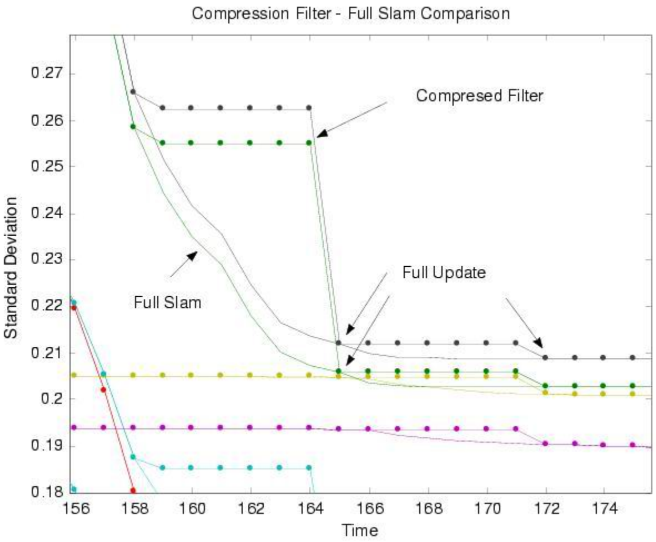


Figure 12: Landmark estimated position error for full Slam and compressed filter (enhanced). This plot presents an enhanced view of the instant when the compressed algorithm performed a full update. A time "165" the full slam (solid line) and the compressed algorithm (solid lines with dots) report same estimated error as predicted.

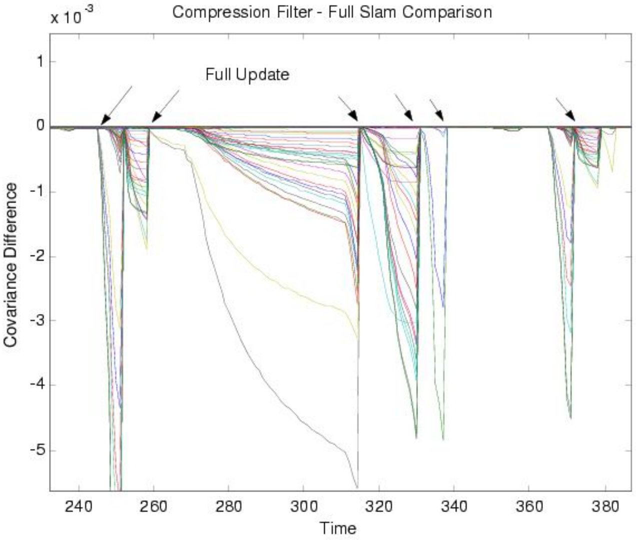


Figure 13: Estimated error differences between full slam and compressed filter. The estimated error difference between both algorithms becomes identically zero when the full update is performed by the compressed algorithm.

$$\Delta P^* = \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP} & \emptyset \end{bmatrix} = \Delta P - \begin{bmatrix} \emptyset & \emptyset \\ \emptyset & \Delta P_{DD} \end{bmatrix}$$

$$P^* = P - \Delta P^* = P - \Delta P + \begin{bmatrix} \emptyset & \emptyset \\ \emptyset & \Delta P_{BB} \end{bmatrix}$$

It can be shown that this simplification generates consistent error covariance estimates. Demonstration: The covariance error matrix $P^*(k+1)$ can be rewritten as follows

$$P^*(k+1) = P(k) - \Delta P^* = P(k) - \Delta P + \delta \tag{63}$$

where

$$\begin{aligned} \Delta P^* &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP} & \emptyset \end{bmatrix} = \Delta P - \delta \\ \Delta P &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP} & \Delta P_{DD} \end{bmatrix} \geq 0 \qquad \delta = \begin{bmatrix} \emptyset & \emptyset \\ \emptyset & \Delta P_{BB} \end{bmatrix} \geq 0 \end{aligned} \tag{64}$$

The matrices ΔP and μ are positive semi-definite since:

$$\begin{aligned} \Delta P &= \begin{bmatrix} \Delta P_{PP} & \Delta P_{PD} \\ \Delta P_{DP}^T & \Delta P_{DD} \end{bmatrix} = W \cdot S \cdot W^T \geq \quad 0 \\ \Delta P_{DD} &= W_D \cdot S_D \cdot W_D^T \geq 0 \end{aligned} \tag{65}$$

As given in Equation 63, the total update is formed by the optimal update plus an additional positive semi-definite noise matrix δ. The matrix δ will increase the covariance uncertainty:

$$P^*(k+1) = P(k+1) + \delta \tag{66}$$

then the sub-optimal update of P^* becomes more conservative than the full update:

$$P^*(k+1) \leq P(k+1) \leq P(k) \tag{67}$$

Finally the sub-matrices 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 have 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 have a significant information contribution to this 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 sub-vector D. In many practical situations a large number of landmarks can usually be associated to the sub-vector 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 Figure 14 .

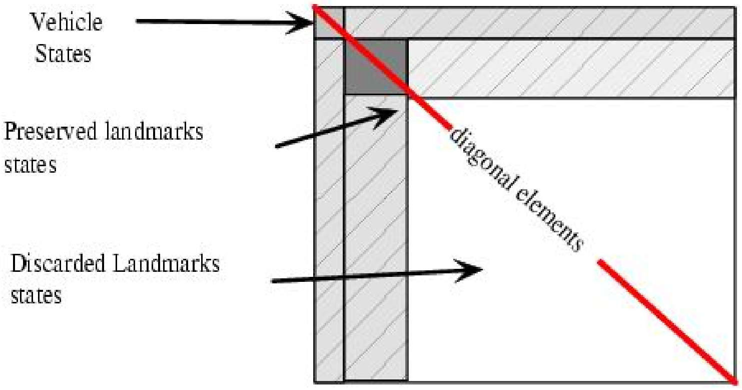


Figure 14: Full covariance matrix divided into the covariance blocks corresponding to the Vehicle and Preserved landmarks states (X_P) and Discarded landmarks states (X_D). The cross-correlation covariance between the Preserved and Discarded states are fully updated as shown in grey. Finally the cross-correlation between the elements of the states corresponding to the "Discarded landmarks" are not updated as shown in white.

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

$$I_1 = \{ i \quad : \quad \Delta P(i,i) < c_1 \cdot P(i,i) \}, \quad I_2 = \{ i \quad : \quad P(i,i) < c_2 \}, \quad I = I_1 \cup I_2 \tag{68}$$

Then $\Delta P*$ is evaluated as follows:

$$\begin{array}{ll} \Delta P^*(i,j) = 0 & \forall i,j \ : \ i \in I \quad and \quad j \in I \\ \Delta P^*(i,j) = \Delta P(i,j) & \forall i,j \ : \ i \notin I \quad or \quad j \notin I \end{array} \tag{69}$$

The error covariance matrix is updated with the simplified matrix D_P

$$P^{*}\left(k+1,k+1\right)=P\left(k+1,k\right)-\Delta P^{*} \tag{70}$$

The practical meaning of the set I_1, is that with the appropriate selection of c_1 we can reject negligible update of covariances. 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 localisation algorithms . Finally, the magnitude of the computation saving factor depends of the size of the set I. With appropriate exploration policies, real time mission planning, the computation requirements can be maintained within the bounds of the on-board resources.

Implementation Issues: Relative Map Representation The sub-optimal 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 decreased by using a number of different bases and making the observation relative to those bases. With this representation the map becomes grouped in constellations. Each constellation has an associated frame based on two landmarks that belong to this constellation. The 'base' landmarks that define the associated frame are represented in a global frame. All the others landmarks that belong to this constellation are defined in the local frame. For a particular constellation, the local frame is based on the 2 base landmarks

$$L_a=\left[\begin{array}{c} x_a \\ y_a \end{array}\right] \quad , \quad L_b=\left[\begin{array}{c} x_b \\ y_b \end{array}\right] \tag{71}$$

It is possible to define 2 unitary vectors that describe the orientation of the base frame:

$$\begin{array}{l} v_1=\frac{1}{\|L_b-L_a\|}\cdot\left(L_b-L_a\right)=\frac{1}{\sqrt{\left(x_b-x_a\right)^2+\left(y_b-y_a\right)^2}}\cdot\left[\begin{array}{c} x_b-x_a \\ y_b-y_a \end{array}\right]=\left[\begin{array}{c} v_{11} \\ v_{12} \end{array}\right] \\ v_2=\left[\begin{array}{c} v_{21} \\ v_{22} \end{array}\right]=\left[\begin{array}{c} -v_{12} \\ v_{11} \end{array}\right] \quad , \qquad \langle v_2,v_1\rangle=0 \end{array} \tag{72}$$

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.

$$L_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad , \quad L_i^a = \begin{bmatrix} \xi_i \\ \eta_i \end{bmatrix} \tag{73}$$

with

$$\begin{array}{l} \xi_i = \langle (L_i - L_a) \, , v_1 \rangle = (L_i - L_a)^T \cdot v_1 \\ \eta_i = \langle (L_i - L_a) \, , v_2 \rangle = (L_i - L_a)^T \cdot v_2 \end{array} \tag{74}$$

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

$$L_i = L_a + \varsigma_i \cdot v_1 + \eta_i \cdot v_2 \tag{75}$$

The reference frame is formed with two landmarks as shown in Figure 15. The observation are then obtained relative to this frame. Assuming that the external sensor returns range and bearing, the observation functions are:

$$\begin{array}{l} h_i = L_i - X_L - R_i \cdot (\cos(\beta_i) \, , \sin(\beta_i)) = 0 \\ \beta_i = \alpha_i + \phi - {}^{\pi}\!/_{2} \\ \alpha_i \quad : \text{objectanglewithrespecttolaserframe} \\ R_i \quad : \text{objectrangewithrespecttolaser} \\ (X_L, \phi) = (x_L, y_L, \phi) \quad : \text{vehiclestates} \end{array} \tag{76}$$

Finally

$$h_i = L_a + \varsigma_i \cdot v_1 + \eta_i \cdot v_2 - P_L - R_i \cdot (\cos(\beta_i) \, , \sin(\beta_i)) = 0 \tag{77}$$

With this representation the landmark defining the bases will become the natural ”Preserved” landmarks. The observations in each constellation will be evaluated with respect to the bases and can be considered in the limit as observation contaminated with white noise. This will make the relative elements of the constellation 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.

Experimental Results The next set of plots present a comparison of the performance of the sub-optimal algorithm. Figure 16 and 17 present two runs, one using most of the

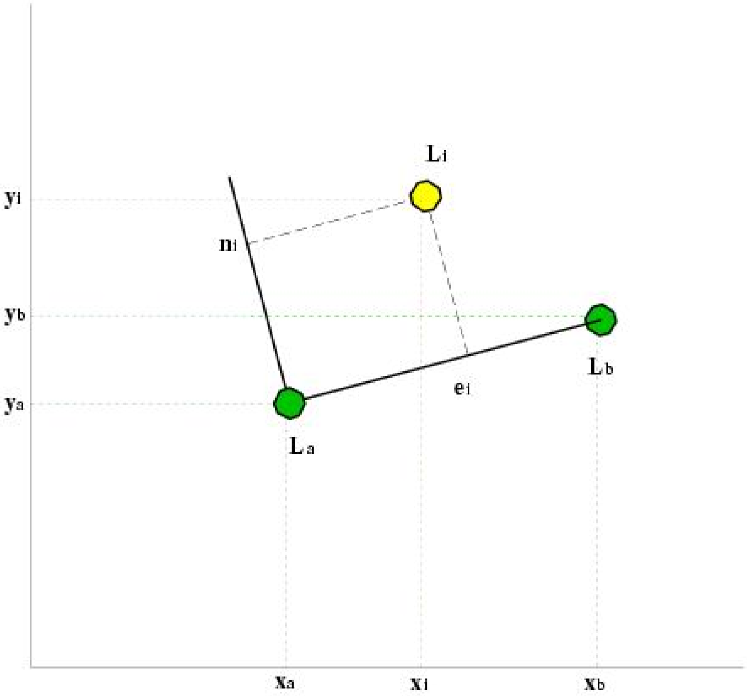


Figure 15: Local reference frame

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. Figure 17 shows that a large number of states are not updated at every time step resulting in a significant reduction in the computational cost of the algorithm. From Figures 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. One important remark regarding the advantage of the relative representation with respect to the simplification presented: 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. 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 is shown in Figures 20 and 21 where the correlation of the relative and absolute map respectively is presented. In Figure 20 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 de-correlated from each other and only correlated to the first block whose cross correlation are updated by the sub-optimal algorithm presented. These results imply that with the relative representation the cross correlation between constellation becomes zero and the sub-optimal algorithm presented becomes close to optimal. This is not the case for the absolute representation as shown in Figure 21 where all the states maintained strong cross-correlations.

Finally Figure 22 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. The system created 19 different constellations to implement the relative map. The cross-correlation coefficients between the different constellations become very small as shown in Figure 23. This demonstrates the advantages of the compressed algorithm since 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 the computational limitation of SLAM can be overcame with the compressed algorithm and appropriate selection of local areas.

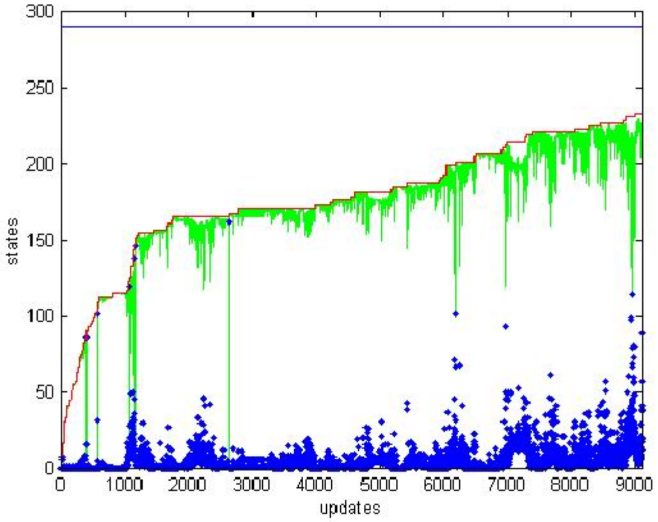


Figure 16: Total number of states and states used and not updated. The figure presents the total number of states with a solid black line. This number is increasing because the vehicle is exploring new areas and validating new landmarks. The states used by the system are represented in grey. The number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

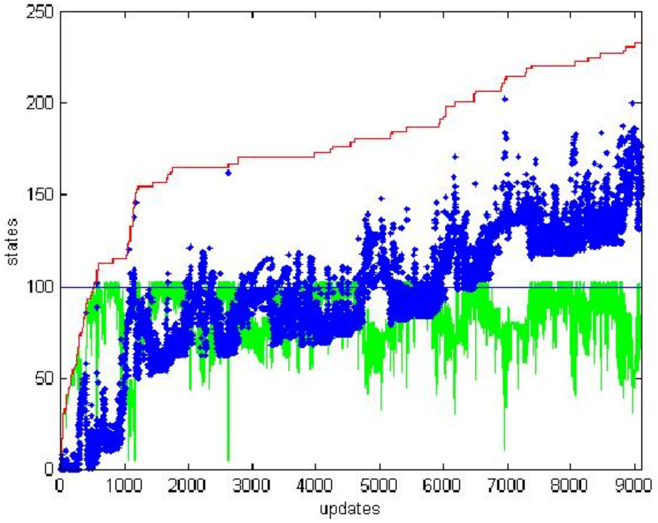


Figure 17: Total number of states and states used and not updated. In this run a maximum number of states was fixed as constraint for the sub-optimal SLAM algorithm. This is appreciated in the grey plot where the maximum number of states remain below a given threshold. The number of states not updated increases with time.

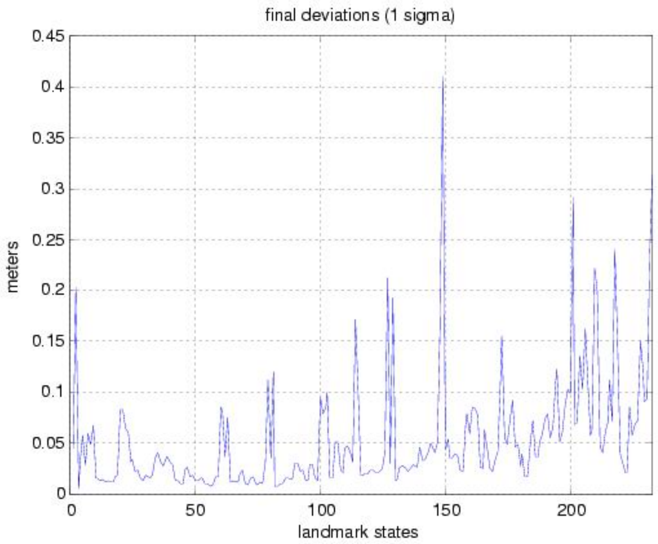


Figure 18: final estimation errors for relative and absolute states using most of the states.

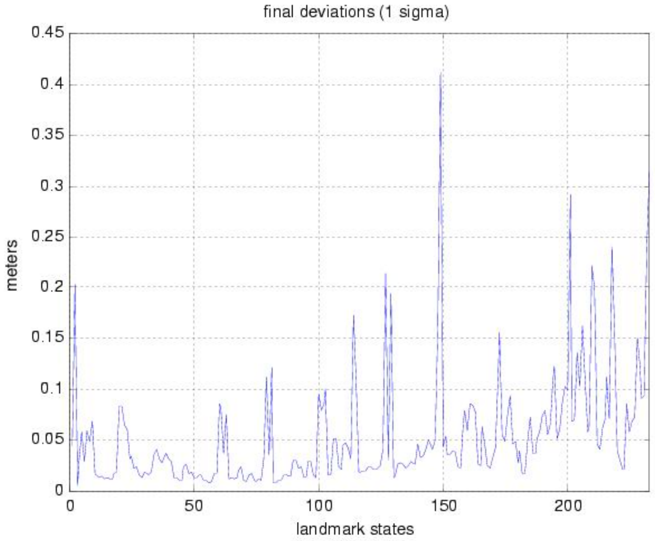


Figure 19: Final estimated error of relative and absolute states using a reduced number of states. These results are similar to the ones using most of the states. This result shows that the algorithm is not only consistent but close to optimal when used with the appropriate map representation.

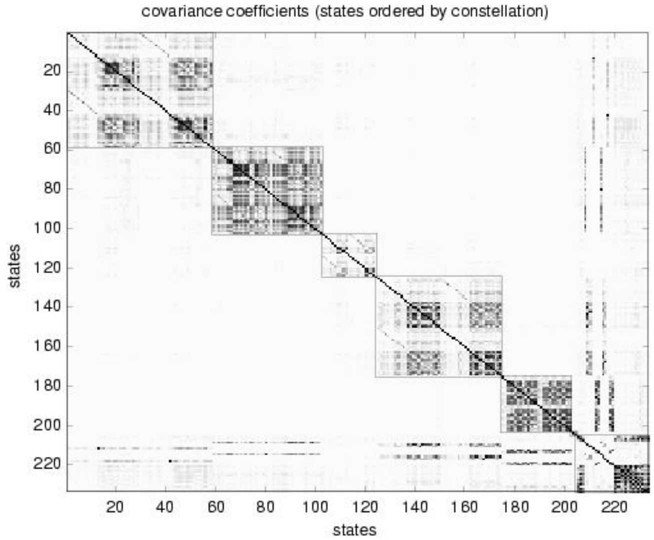


Figure 20: Correlation coefficient for the relative representation. Each block represents the cross-correlation coefficient of the elements of the different constellation. The block in the right corner contains the vehicle states and all the bases. It can be seen that the cross-correlation between different constellations is very small. It is also clear the non-zero cross-correlation between the bases and the different constellations. These correlations are updated by the sub-optimal filter.

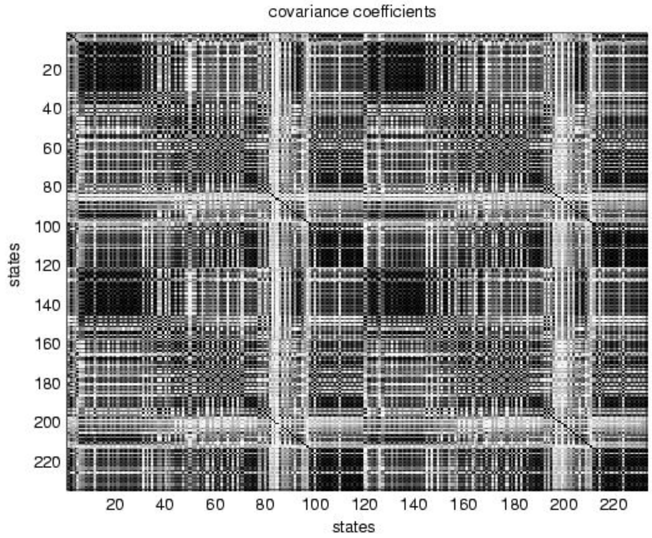


Figure 21: Correlation Coefficient for the absolute representation. In this case the map appears completely correlated and the sub-optimal algorithm will generate consistent but more conservative results.

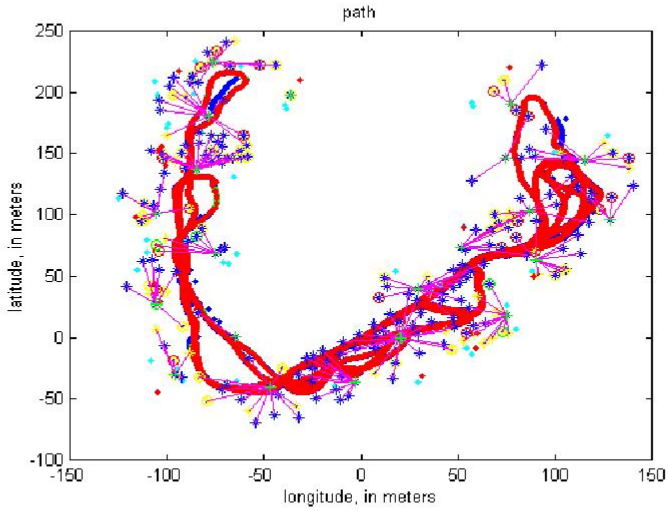


Figure 22: Constellation map and vehicle trajectory. 19 constellations were formed by the algorithm. The intersection of the bases are presented with a '+' and the other side of the segment with a 'o'. The relative landmarks are represented with '*' and its association with a base is represented with a line joining the landmark with the origin of the relative coordinate system

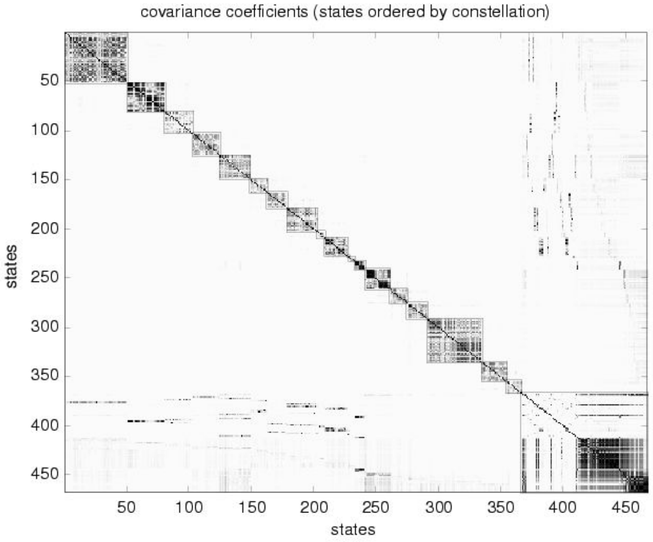


Figure 23: Cross correlation coefficients. The plots shows 19 constellation and a block in the right hand corner containing the correlation coefficient for the bases and the vehicle states. It can be appreciated that the crosscorrelation between the relative states of the different bases is very small.

References

[1] Clark S. Dissanayake G., Newman P. and Durrant-Whyte H. A Solution to Simultaneous Localisation adn Map Building (SLAM) Problem. In *IEEE Journal of Robotics and Automaton*, volume 17, No.3, June 2001.

[2] Guivant J. and Nebot E. Optimization of Simultaneous Localization and Map Building Algorithm for Real Time Implementation. 17, No.3:731–747, June 2001.

[3] John Leonard and H. Feder. Decoupled stochastic mapping for mobile robot and auv navigation. *IEEE Journal of Oceanic Engineering*, 66, No.4:561–571, 2001.

[4] P. Maybeck. *Stochastic Models, Estimation and Control*, volume 1. Academic Press, 1982.

[5] J. Neira and J.D. Tardos. Data association in stochastic mapping using the joint compatibility test". *IEEE Transaction of Robotics and Automation*, pages 890–897, 2001.

[6] Michael Montemerlo Sebastian. Fastslam: A factored solution to the simultaneous localization and mapping problem, http://citeseer.nj.nec.com/503340.html.

[7] Cheeseman P. Smith R., Self M. A stochastic map for uncertain spatial relationships. In *4th International Symposium on Robotic Research, MIT Press*, 1987.