Computationally efficient Solution to the Simultaneous Localisation and Map Building (SLAM) Problem

Gamini Dissanayake, Hugh Durrant-Whyte and Tim Bailey
Australian Centre for Field Robotics
University of Sydney
Sydney, NSW 2006
Australia

Abstract

The theoretical basis of the solution to the simultaneous localisation and map building (SLAM) problem where an autonomous vehicle starts in an unknown location in an unknown environment and then incrementally build a map of landmarks present in this environment while simultaneously using this map to compute absolute vehicle location is now well understood [5, 6]. Although a number of SLAM implementations have appeared in the recent literature [4, 3], the need to maintain the knowledge of the relative relationships between all the landmark location estimates contained in the map makes SLAM computationally intractable in implementations containing more than few tens of landmarks. In this paper the theoretical basis and a practical implementation of a computationally efficient solution to SLAM is presented. The paper shows that it is indeed possible to remove a large percentage of the landmarks from the map without making the map building process statistically inconsistent. Furthermore it is shown that the efficiency of the SLAM can be maintained by judicious selection of landmarks, to be preserved in the map, based on their information content.

1 Introduction

In the past year, there has been rapid and substantial progress on the estimation-theoretic solution to the simultaneous localisation and map building (SLAM) problem. A recent session at the International Symposium on Experimental Robotics (ISER) was devoted to describing progress by a number of research groups working on the SLAM problem. These presentations described theoretical proofs of convergence of the SLAM problem [5], simulation results showing how sub-optimal map management strategies can be used to develop efficient solutions to large-scale SLAM problems [9], and two different implementations of SLAM algorithms on outdoor and indoor vehicles [5, 3]. Together, these papers have set the foundations for a comprehensive and pervasive solution to the combined localisation and map building problem. The challenge now facing researchers in this area is to deploy a substantial implementation of a SLAM system in a large-scale unstructured environment: The ability to deploy a vehicle in an unknown environment and have it construct a map of this environment while simultaneously using this map to compute it's location (SLAM) would truely make such a vehicle autonomous.

Although the theoretical basis of SLAM is now well understood, a number of critical theoretical and practical issues need to be resolved before SLAM can be demonstrated in large unstructured environments. These include issues of computational efficiency, map management methods, local and global convergence properties of the map estimator, data association and sensor management.

1.1 Computational Efficiency of SLAM

The existence of a non-divergent estimation theoretic solution to the SLAM problem and the general structure of SLAM navigation algorithms is described in [5]. It was shown that the uncertainty in the estimates of relative landmark locations reduces monotonically, that these uncertainties converge to zero, and that the uncertainty in vehicle and absolute map locations achieves a lower bound. It was also shown that it is the cross-correlations in the map covariance matrix which maintain knowledge of the relative relationships between landmark location estimates and which in turn underpin the exhibited convergence properties. Thus propagation of the full map covariance matrix was shown to be essential to the solution of the SLAM problem.

However, the use of the full map covariance matrix at each step in the map building problem causes substantial computational problems. As the number of landmarks N increases, the computation required at each step increases as N^3, and required map storage increases as N^2. As the range over which it is desired to operate a SLAM algorithm increases (and thus the number of landmarks increases), it becomes essential to develop a computationally tractable version of the SLAM map building algorithm which maintains the properties of being consistent and non-divergent.

This paper considers an autonomous vehicle (mobile robot) equipped with a sensor capable of making measurements of the location of landmarks relative to the vehicle. The vehicle starts at an unknown location with no knowledge of the location of landmarks in the environment. As the vehicle moves through the environment (in a stochastic manner) it makes relative observations of the location of individual landmarks. It starts from the theoretical foundation presented in [5] which proves that a Kalman filter based solution to the general SLAM problem exists and it is indeed possible to construct a perfectly accurate map and simultaneously compute vehicle position estimates without any prior knowledge of vehicle or landmark locations.

The key contribution of this paper is a map management strategy that results in a computationally efficient solution to SLAM. Firstly, it shows that any landmark and associates elements of the map covariance matrix can be deleted during the SLAM process without compromising the statistical consistency of the underlying Kalman filter. Secondly, it devises a strategy to select a landmarks for deletion from the map, while minimising the loss of information during this process. Finally, it demonstrates and evaluates the implementation of the proposed algorithm in an indoor environment using a scanning laser range finder.

Section 2 of this paper summarises the mathematical structure of the estimation-theoretic SLAM problem as defined in [5]. Section 3 then establishes the theoretical basis of the map management strategy. Section 4 provides a practical demonstration of an implementation of the proposed algorithm. It is shown that the proposed strategy has a minimal effect on the convergence properties of the SLAM algorithm whilst substantially improving its computational efficiency. Discussion and conclusions are provided in Section 5.

2 The Estimation-Theoretic solution to the SLAM Problem

This section summarises the mathematical framework employed in the study of the SLAM problem in [5]. This framework is provided here for completeness and to facilitate the discussion in the later sections. An interested reader is referred to [5] for a more comprehensive description.

2.1 Vehicle and Land-Mark Models

The setting for the SLAM problem is that of a vehicle with a known kinematic model, starting at an unknown location, moving through an environment containing a population of features or landmarks. The terms feature and landmark will be used synonymously. The vehicle is equipped with a sensor that can take measurements of the relative location between any individual landmark and the vehicle itself as shown in Figure 1. The absolute locations of the landmarks are not available.

The state of the system of interest consists of the position and orientation of the vehicle together with the position of all landmarks. The state of the vehicle at a time step k is denoted $\mathbf{x}_v(k)$. Without prejudice, a linear (synchronous) discrete-time model of the evolution of the vehicle and the observations of landmarks is adopted. Although vehicle motion and the observation of landmarks is almost always non-linear and asynchronous in any real navigation problem, the use of linear synchronous models does not affect the validity of the proofs in Section 3 other than to require the same linearisation assumptions as those normally employed in the development of an extended Kalman filter. Indeed, the implementation of the SLAM algorithm described in Section 4 uses non-linear vehicle models and non-linear asynchronous observation models.

The state of the system of interest consists of the position and orientation of the vehicle together with the position of all landmarks. The state of the vehicle at a time step k is denoted $\mathbf{x}_v(k)$. The motion of the vehicle

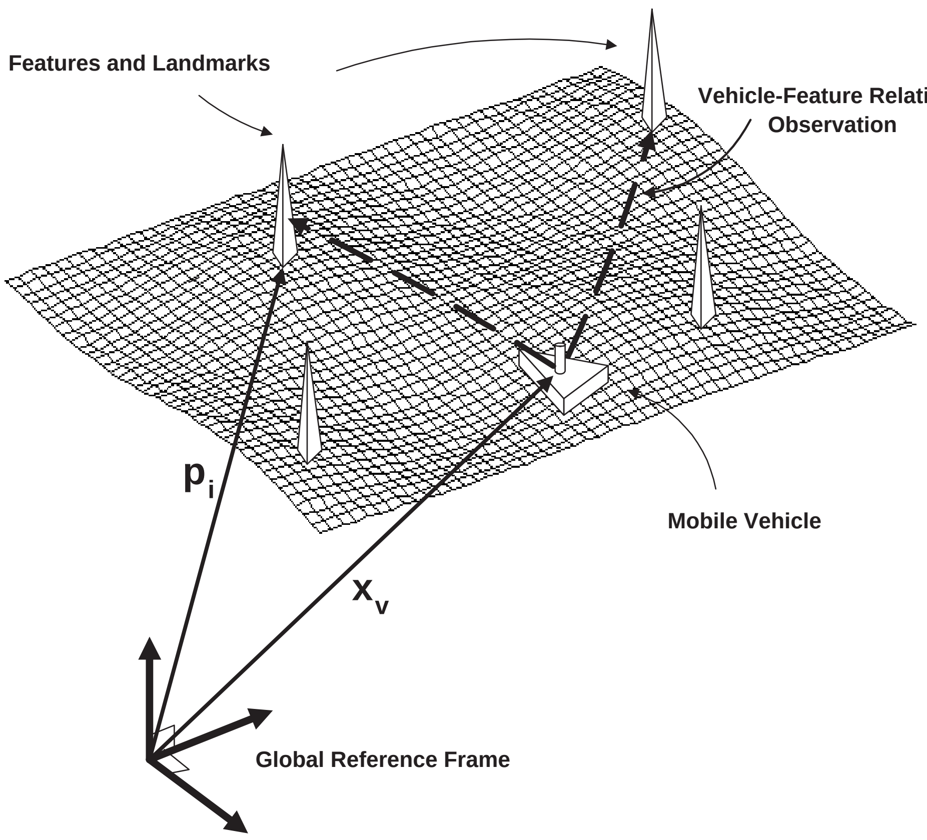

Figure 1: The Structure of the SLAM problem. A vehicle takes relative observations of environment landmarks. The absolute location of landmarks and vehicle are unknown.

through the environment is modeled by a conventional linear discrete-time state transition equation or process model of the form

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

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 $\mathbf{Q}_v(k)$ (see [2] and [10] for further details). The location of the i^{th} landmark is denoted $\mathbf{p}_i$. The "state transition equation" for the i^{th} landmark is

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

since landmarks are assumed stationary. Without loss of generality the number of landmarks in the environment is arbitrarily set at N. The vector of all N landmarks is denoted

$$\mathbf{p} = \left[\begin{array}{ccc} \mathbf{p}_1^T & \ldots & \mathbf{p}_N^T \end{array} \right]^T \tag{3}$$

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

$$\mathbf{x}(k) = \left[\begin{array}{cccc} \mathbf{x}_v^T(k) & \mathbf{p}_1^T & \ldots & \mathbf{p}_N^T \end{array} \right]^T. \tag{4}$$

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

$$\begin{bmatrix} \mathbf{x}_v(k+1) \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_N \end{bmatrix} = \begin{bmatrix} \mathbf{F}_v(k) & 0 & \dots & 0 \\ 0 & \mathbf{I}_{p_1} & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & \mathbf{I}_{p_N} \end{bmatrix} \begin{bmatrix} \mathbf{x}_v(k) \\ \mathbf{p}_1 \\ \vdots \\ \mathbf{p}_N \end{bmatrix} + \begin{bmatrix} \mathbf{u}_v(k+1) \\ \mathbf{0}_{p_1} \\ \vdots \\ \mathbf{0}_{p_N} \end{bmatrix} + \begin{bmatrix} \mathbf{v}_v(k+1) \\ \mathbf{0}_{p_1} \\ \vdots \\ \mathbf{0}_{p_N} \end{bmatrix} \tag{5}$$

$$\mathbf{x}(k+1) = \mathbf{F}(k)\mathbf{x}(k) + \mathbf{u}(k+1) + \mathbf{v}(k+1) \tag{6}$$

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

2.2 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. Again, without prejudice, observations are assumed to be linear and synchronous. The observation model for the i^{th} landmark is written in the form

$$\begin{aligned} \mathbf{z}_i(k) &= \mathbf{H}_i\mathbf{x}(k) + \mathbf{w}_i(k) && (7) \\ &= \mathbf{H}_{p_i}\mathbf{p} - \mathbf{H}_v\mathbf{x}_v(k) + \mathbf{w}_i(k) && (8) \\ & && (9) \end{aligned}$$

where $\mathbf{w}_i(k)$ is a vector of temporally uncorrelated observation errors with zero mean and variance $\mathbf{R}_i(k)$. It is important to note that the observation model for the i^{th} landmark is written in the form

$$\begin{aligned} \mathbf{H}_i &= [-\mathbf{H}_v, \mathbf{0} \cdots \mathbf{0}, \mathbf{H}_{p_i}, \mathbf{0} \cdots \mathbf{0}] && (10) \\ &= [-\mathbf{H}_v, \mathbf{H}_{mi}] && (11) \end{aligned}$$

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).

2.3 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 as a necessary prelude to the developments in Sections 3 and 4 of this paper. Detailed descriptions may be found in [2],[10] and [8]. The Kalman filter recursively computes estimates for a state $\mathbf{x}(k)$ which is evolving according to the process model in Equation 2.1 and which is being observed according to the observation model in Equation 7. The Kalman filter computes an estimate which is equivalent to the conditional mean $\hat{\mathbf{x}}(p|q) = E[\mathbf{x}(p)|\mathbf{Z}^q]$ $(p \geq q)$, 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 2.1 and 7 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

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{F}(k)\hat{\mathbf{x}}(k|k) + \mathbf{u}(k) \quad (12)$$
$$\hat{\mathbf{z}}_i(k+1|k) = \mathbf{H}_i(k)\hat{\mathbf{x}}(k+1|k) \quad (13)$$
$$\mathbf{P}(k+1|k) = \mathbf{F}(k)\mathbf{P}(k|k)\mathbf{F}^T(k) + \mathbf{Q}(k), \quad (14)$$

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 7. Assuming correct landmark association, an innovation is calculated as follows

$$\nu_i(k+1) = \mathbf{z}_i(k+1) - \hat{\mathbf{z}}_i(k+1|k) \quad (15)$$

together with an associated innovation covariance matrix given by

$$\mathbf{S}_i(k+1) = \mathbf{H}_i(k)\mathbf{P}(k+1|k)\mathbf{H}_i^T(k) + \mathbf{R}_i(k+1). \quad (16)$$

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

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}_i(k+1)\nu_i(k+1) \quad (17)$$
$$\mathbf{P}(k+1|k+1) = \mathbf{P}(k+1|k) - \mathbf{W}_i(k+1)\mathbf{S}_i(k+1)\mathbf{W}_i^T(k+1) \quad (18)$$

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

$$\mathbf{W}_i(k+1) = \mathbf{P}(k+1|k)\mathbf{H}_i^T(k)\mathbf{S}_i^{-1}(k+1) \quad (19)$$

The update of the state estimate covariance matrix is of paramount importance to the SLAM problem. It is shown in [5] that as the vehicle progresses through the environment the errors in the estimates of any pair of landmarks become more and more correlated. Furthermore, the entire structure of the SLAM problem critically depends on maintaining complete knowledge of the cross correlation between landmark estimates. Minimizing or ignoring cross correlations is precisely contrary to the structure of the problem.

3 Map management by deleting landmarks

This section examines in detail, the evolution of the state covariance matrix $\mathbf{P}(k)$ that results from the special structure of the matrix $\mathbf{H}_i(k)$ give in Equation 10. It is shown that when a landmark is no longer visible from the current robot location, it can be deleted from the map without affecting the statistical consistency of the underlying estimation process. It is also shown that once deleted, all information accumulated so far about the landmark need to be discarded and that the landmark need to be re-initialised when it comes back into view. A process for selecting the landmarks to be removed is proposed.

3.1 Evolution of the state covariance matrix

The state covariance matrix at any instant k may be written as

$$\mathbf{P} = \sum_{j=v,1,2,..n} \sum_{l=v,1,2,..,n} \mathbf{P}_{jl}$$

where $\mathbf{P}_{vv}$ is the error covariance matrix associated with the vehicle state estimate, $\mathbf{P}_{jl}, j = 1..n, l = 1..n$ are the covariances associated with the landmark state estimates, and $\mathbf{P}_{vl}, l = 1..n$ are the cross-covariances between vehicle and landmark states. Dropping the index k for clarity and using Equation 10 in Equation 16, the innovation covariance after i^{th} landmark is observed is given by

$$\mathbf{S}_i = \mathbf{H}_v\mathbf{P}_{vv}\mathbf{H}_v^T - \mathbf{H}_{pi}\mathbf{P}_{vi}\mathbf{H}_v^T - \mathbf{H}_v\mathbf{P}_{vi}\mathbf{H}_{pi}^T + \mathbf{H}_{pi}\mathbf{P}_{vi}\mathbf{H}_{pi}^T$$

Clearly $\mathbf{S}_i$ is a function of the covariances and cross-covariance of the vehicle and landmark i, and is independent of all other landmark covariances and cross-covariances.

During the update stage that follows the observation of feature i, the Kalman gains and the change to the state covariance matrix can be written as

$$\mathbf{W}_{ij} = \left[-\mathbf{P}_{vj}\mathbf{H}_v^T + \mathbf{P}_{ij}\mathbf{H}_{pi}^T\right]\mathbf{S}_i^{-1}, j = v, 1, 2, ..., n$$

$$\delta\mathbf{P}_{jl} = \left[-\mathbf{P}_{vj}\mathbf{H}_v^T + \mathbf{P}_{ji}\mathbf{H}_{pi}^T\right]\mathbf{S}_i^{-1}\left[-\mathbf{H}_v\mathbf{P}_{vl} + \mathbf{H}_{pi}\mathbf{P}_{li}\right], j = v, 1, 2, ..., n; l = v, 1, 2, ..., n \tag{20}$$

Effect of the observation of a given landmark i on the evolution of the vehicle and landmarks states and the state covariance matrix can now be summerised as follows

- The Kalman gains associated with the vehicle state and all the landmark states are non-zero. Therefore the estimate of all the landmark states are updated even when many of these landmarks are not visible from the current robot location.

- All the elements of the state covariance matrix need to be modified after each observation. It is not sufficient to update the elements of the state covariance matix corresponding to the vehicle and the landmark currently being observed. This is the main reason for the computational complexity of the SLAM process.

- The Kalman gain and the changes to the elements of the state covariance matrix associated with the j^{th} landmark are a function of the covariances and cross-covariances associated with the vehicle state and the landmark states i and j. These updates are independent of covariances and cross-covariances of all other landmark states.

- Updates of the vehicle and landmark states and the associated covariance matrix after the observation of a landmark i are unaffected by the removal of the state corresponding to any landmark j and the corresponding rows and columns of the state covariance matrix. Therefore any landmark that is currently not being observed can be removed from the map without affecting the statistical consistancy of the map building process.

- When a removed landmark is observed again, the vehicle location and the states of other landmarks can not be updated using the state and the associated covariances of the landmark at the instant it was removed as these are no longer valid. The landmark need to be reinitialised. Therefore, removing a landmark from the map results in a loss of information.

3.2 Procedure for selecting landmarks for removal

The prime objective of the SLAM process is to maintain a good estimate of the vehicle location by observing and estimating locations of landmarks. The information available at any instant for localisation depends on the accuracy of the sensor and the accuracy of the location estimates of the landmarks visible from the vehicle. The geometrical distribution of the these landmarks are also of importance. For example, when the number of landmarks to be maintained in the map is to be restricted, it is more advantageous to keep landmarks that are physically far apart and those that are close to each other. Also, the landmark location estimates that are less correlated provide more information. In the limiting case, when two landmark location estimates are fully correlated, there is no advantage to be gained by maintaining both these landmarks in the map except for geometric considerations. Most importantly, the computational effort involved with selecting the landmarks to be removed need to be efficient so as not to offset the computational efficiency gained in the SLAM process by deleting these landmarks.

The following two step process is therefore proposed for selecting the landmarks for removal from the map.

- Collect the set of landmarks S_L that changed from visible to not visible over a time interval during which the vehicle travelled a predetermined distance d_v. Select one landmark to be preserved in the map and remove all the remaining landmarks in S_L. This will result in an even distribution of landmarks over the area traversed by the vehicle. The landmark density of the final map obtained will be closely related to d_v. Suitable choice for d_v is a function of the range, accuracy and the speed of the sensor used, extent of process noise and the tradeoff between the localisation accuracy required and the computational efficiency. Such an incremental process of deletion, to a large extent, eliminates the need to consider effect of geometry in evaluating the landmarks for their information content.

- The next step is to select the landmark that has the maximum information content from the set of landmarks S_L. This landmark now serves to localise the vehicle when the vehicle revisits the region surrounding the landmark. The selected landmark should provide the best information to improve the location of the vehicle when the landmark is observed from any position within this region. Clearly, lower the uncertainty of the location estimate of the landmark, better the effect of observing this landmark on the estimate of the vehicle location. Therefore, it is proposed to use the reciprocal of the trace of the covariance matrix $\mathbf{P}_{jj}$ of the of the landmark location where j is the landmark under consideration.

  Various other information measures such as the Shannon or Fisher information can also be used. However experiments indicated that the effect of the strategy used to evaluate the merit of the landmarks does not have a significant effect on the accuracy of the vehicle location estimates.

4 Experimental Results

In this section a practical implementation of the proposed simultaneous localisation and map building (SLAM) algorithm on an indoor robot is presented. The robot is equipped with a Laser range finder which provides a two-dimensional range scans. A feature detector [12] is used to extract location of landmarks with respect to the vehicle. This implementation serves to highlight the statistical consistency and effectiveness of the proposed map management strategy.

4.1 Experimental setup

Figure 2 shows the test vehicle; a three wheeled vehicle fitted with a laser range finder as the primary sensor used in the experiments. Encoders are fitted to the drive shafts and to the steering motor to provide a measure of the vehicle speed and vehicle heading.

The laser range finder is a time of flight device that scans the range to the surrounding environment within a planar sweep of 180^o. It returns 361 range measurements in a single sweep, with a range resolution of 5 cm and an angular resolution of 0.5^o.

The vehicle is driven manually. Range scans are logged together with encoder and steer information by an on-board computer system. Range scans are processed to obtain point landmarks correspond to foreground points, edges and corners in the environment. A detailed description of the feature detector can be found in [12].In the evaluation of the SLAM algorithm, this information is employed without any *a priori* knowledge of landmark location to deduce estimates for both vehicle position and landmark locations.

In the following, the vehicle state is defined by $\mathbf{x}_v = [x, y, \varphi]^T$ where x and y are the coordinates of the centre of the front wheel of the vehicle with respect to some global coordinate frame and φ is the orientation of the vehicle axis. The landmarks are modeled as point landmarks and represented by a cartesian pair such that $\mathbf{p}_i = [x_i, y_i], i = 1 \ldots N$. Both vehicle and landmark states are registered in the same frame of reference.

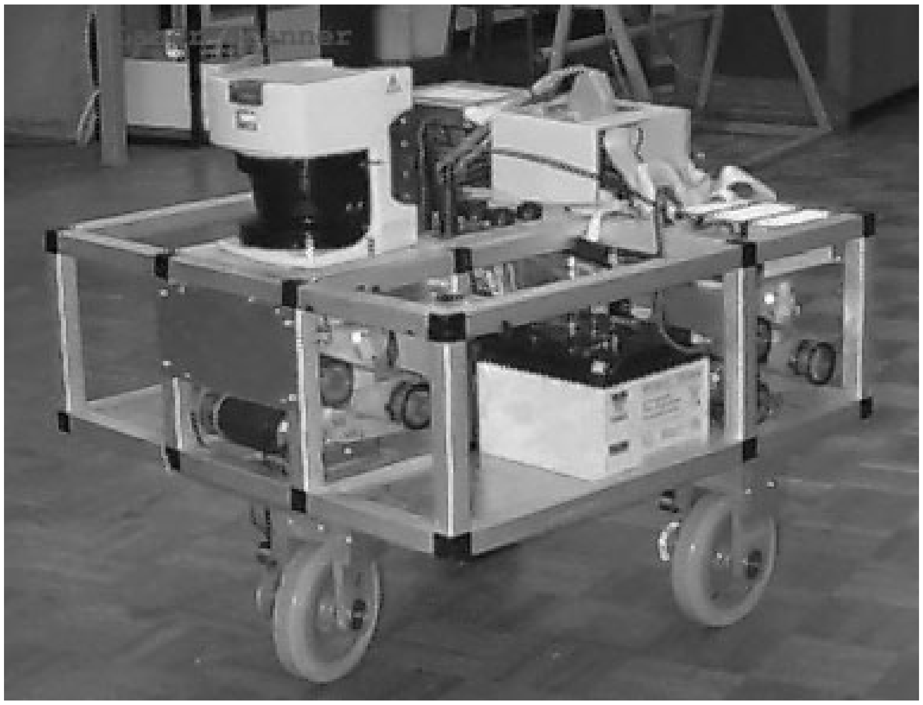
Figure 2: The test vehicle. The laser range finder is seen in the front

4.1.1 The process model

Figure 3 shows a schematic diagram of the vehicle in the process of observing a landmark. The following kinematic equations can be used to predict the vehicle state from the steering γ and velocity V of the front wheel;

$$\begin{aligned}\dot{x} &= V\cos(\varphi+\gamma)\\ \dot{y} &= V\sin(\varphi+\gamma)\\ \dot{\varphi} &= \frac{V\sin(\gamma)}{L},\end{aligned}$$

where L is the wheel-base length of the vehicle. These equations can be used to obtain a discrete-time vehicle process model in the form

$$\begin{bmatrix} x(k+1)\\ y(k+1)\\ \varphi(k+1)\end{bmatrix} = \begin{bmatrix} x(k)+\Delta T V(k)\cos(\varphi(k)+\gamma(k))\\ y(k)+\Delta T V(k)\sin(\varphi(k)+\gamma(k))\\ \varphi(k)+\frac{\Delta T V(k)\sin(\gamma(k))}{L}\end{bmatrix} \quad (21)$$

for use in the prediction stage of the vehicle state estimator.

The landmarks in the environment are assumed to be stationary point targets. The landmark process model is thus

$$\begin{bmatrix} x_i(k+1)\\ y_i(k+1)\end{bmatrix} = \begin{bmatrix} x_i(k)\\ y_i(k)\end{bmatrix} \quad (22)$$

for all landmarks $i = 1 \ldots N$. Together, Equations 21 and 22 define the state transition matrix $\mathbf{f}(\cdot)$ for the system.

4.1.2 Feature Extraction

Feature extraction is performed independently on each laser scan and makes the assumption that each scan is an instantaneous snapshot of the environment. That is, it is assumed that the speed of the laser sweep is much faster

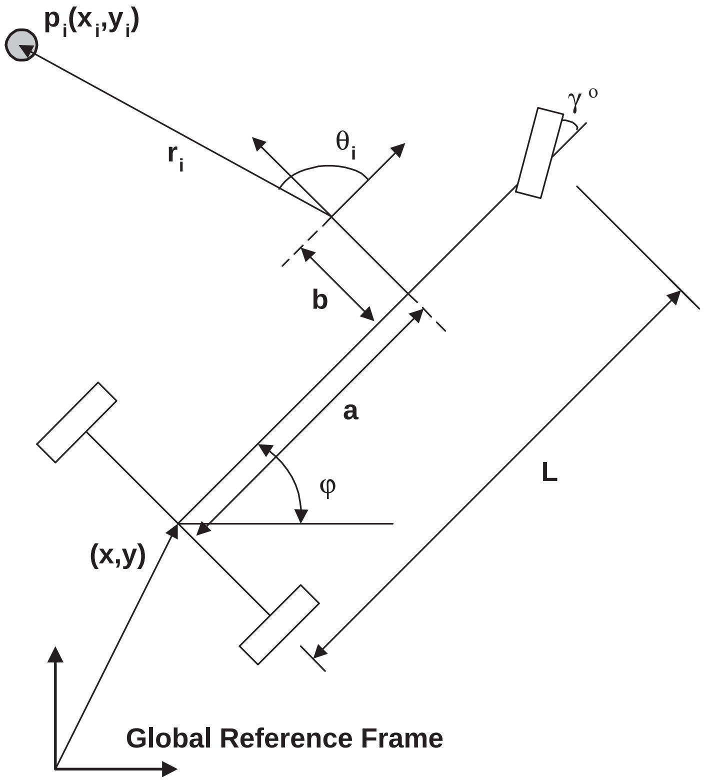

Figure 3: Vehicle and observation kinematics

than the speed of the vehicle such that the vehicle motion does not significantly distort the scan. This assumption is reasonable for these experiments as the scan sweep takes about 40ms and the indoor robot moves at less than 0.5m/s. The first step in feature extraction is to cluster the range data. This is performed as follows. The first range measurement, R_1, is compared with the next, R_2. If the difference between the two measurements is less than a threshold, ΔR_{max}, then R_2 is added to the cluster containing R_1; otherwise a new cluster is started. This process is performed sequentially through the list of 361 measurements, with the general algorithm shown below:

```
$\Delta R_{max} = C_1 + C_2 * \min(R_i, R_{i+1})$
$\Delta R = abs(R_i - R_{i+1})$
IF $\Delta R < \Delta R_{max}$
        add to cluster
ELSE
        start new cluster
END
```

The constants C_1 and C_2 are tunable to the laser's noise and resolution characteristics. They were set to 0.07m and 0.04 respectively for the experimentation in this paper. The constant C_1 is designed to accommodate for the small noise/discretisation element in the measurement of a given range due to the range resolution of the laser. The constant C_2 caters for the fact that the laser measurements occur at fixed angular increments and, therefore, the minimum distance between any two (cartesian) point measurements increases with the distance from the sensor; so the threshold must increase with range. After clustering, there are three types of features that are extracted: foreground points, foreground edges and corners. The first two types are obtained by checking the two end-points of each cluster and comparing them with the end-points of the two adjacent clusters. The latter type is found by fitting lines to the data points and looking for (nearly) perpendicular intersections. A foreground point exists if the first point in the cluster, C_iP_{first}, has a shorter range than the last point in the previous cluster, $C_{i-1}P_{last}$, and the last point in the cluster, C_iP_{last}, has a shorter range than the first point in the next cluster, $C_{i+1}P_{first}$, and the distance between the two end points, C_iP_{first} and C_iP_{last}, is less than a predefined distance K. The location of the foreground point is taken as the mean of all the points in the cluster. K was set to 0.3m for these experiments. Foreground edges exist if the cluster has not been classified as a foreground point and the range of either end-point is less than the range of the appropriate end-points of the adjacent clusters. The location of a foreground edge is simply the location of the end-point(s) which fulfills this criterion. This algorithm is summarised as follows:

```
IF $R(C_iP_{first}) < R(C_{i-1}P_{last})$
        AND $R(C_iP_{last}) < R(C_{i+1}P_{first})$
        AND $dist(C_iP_{first}, C_iP_{last})$
THEN
        cluster $C_i$ is a foreground point
ELSE
        IF $R(C_iP_{first}) < R(C_{i-1}P_{last})$
        THEN
                $C_iP_{first}$ is a foreground edge
        IF $R(C_iP_{last}) < R(C_{i+1}P_{first})$
        THEN
                $C_iP_{last}$ is a foreground edge
END
```

where $dist(a, b)$ is the Euclidean distance between a and b and $R()$ is the distance from the origin.

Extracting corner points is a two-step operation. First, lines are fitted within each cluster and, second, the intersections of adjacent lines are examined for corner validity. Line fitting is itself a two-step process. This process is performed individually on each cluster. First, the points in a cluster are subdivided using a recursive curve approximation method found in [1]. This serves to break up the cluster into nearly linear subgroups. Second, a least squares line [11] is fitted each point subgroup. Corners are found be examining the angle between two adjacent lines and the closeness of the line segment endpoints to their intersection. For these experiments, intersections were accepted if the angle between two lines was between 30^o and 150^o, and the distance between the intersection and each line segment endpoint was less than 0.25m.

Figure 4 shows a view of the environment as seen from the robot. Figure 5 shows the clusters generated and the features detected in the scan taken at this location. Note that the photograph has a limited field of view while the laser scans 180^o. The poles of the hand-rail correspond to some of the foreground points detected while the edges and corners correspond to the steps in the walls and the door.

4.1.3 The observation model

The feature detector used in the experiments returns the range $r_i(k)$ and bearing $\theta_i(k)$ to a landmark i. Referring to Figure 3, the observation model can be written as

$$\begin{aligned} r_i(k) &= \sqrt{(x_i - x_r(k))^2 + (y_i - y_r(k))^2} + w_r(k) \\ \theta_i(k) &= \arctan\left(\frac{y_i - y_r(k)}{x_i - x_r(k)}\right) - \varphi(k) + w_\theta(k) \end{aligned} \tag{23}$$

where $w_r$and w_θ are the noise sequences associated with the range and bearing measurements, and $[x_r(k), y_r(k)]$ is the location of the radar given, in global coordinates, by

$$\begin{aligned} x_r(k) &= x(k) + a\cos(\varphi(k)) - b\sin(\varphi(k)) \\ y_r(k) &= y(k) + a\sin(\varphi(k)) + b\cos(\varphi(k)) \end{aligned}$$

Equation 23 defines the observation model $\mathbf{h}_i(\cdot)$ for a specific landmark.

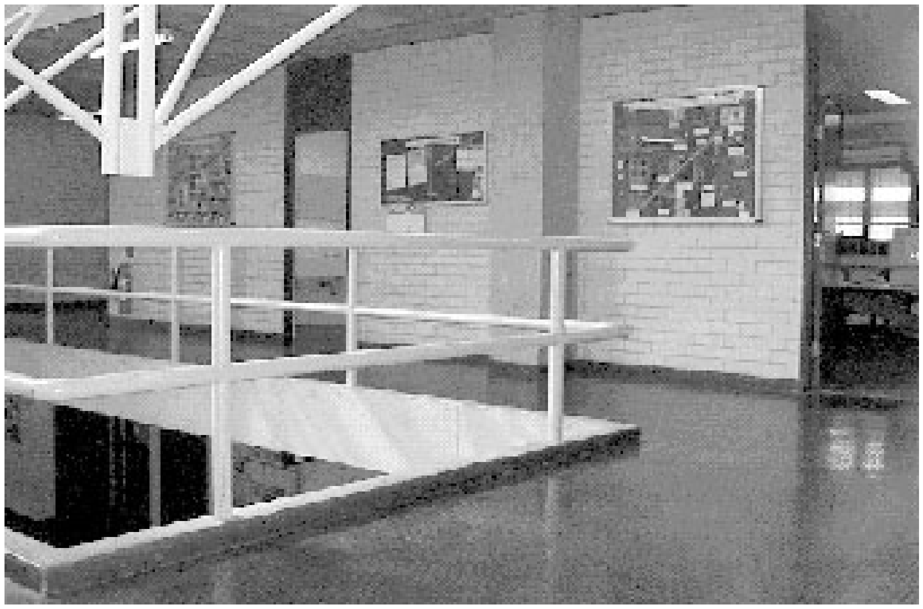

Figure 4: A view from the mobile robot at the test site. Foreground points detected correspond to the vertical metal rails seen.

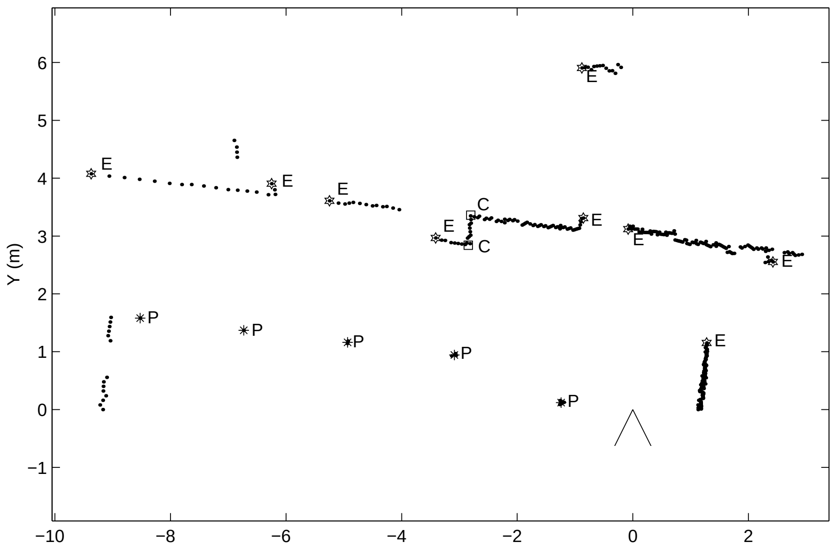

Figure 5: Clusters and features detected in the laser scan that correspond to 4. 'P', 'E', and 'C' denote the foreground points, edges and corners respectively.

4.1.4 Estimation equations

The theoretical developments in this paper employed only linear models of vehicle and landmark kinematics. This was necessary to develop the necessary proofs of convergence. However, the implementation described here requires the use of non-linear models of vehicle and landmark kinematics $\mathbf{f}(.)$ and non linear models of landmark observation $\mathbf{h}(.)$.

Practically an Extended Kalman Filter (EKF) rather than a simple linear Kalman filter is employed to generate estimates. The EKF uses linearised kinematic and observation equations for generating state predictions. The use of the EKF in vehicle navigation and the necessary assumptions needed for successful operation is well known (see for example the development in [7]), and is thus not developed further here.

4.1.5 Map initialisation

In any SLAM algorithm the number and location of landmarks is not known *a priori*. Landmark locations must be initialised and inferred from observations alone. The feature detector returns a significant number of potential landmarks but only the observations resulting from landmarks that are invariant with respect to the location of the vehicle should be used in the estimation process. An algorithm described in [6] is used to deal with these issues. The algorithm uses two landmark lists to record "tentative" and "confirmed" targets. A tentative landmark is initialised on receipt of a range and bearing measurement. A tentative target is promoted to a confirmed landmark when it receives a sufficient number of hits. Once confirmed, the landmark is inserted into the augmented state vector to be estimated as part of the SLAM algorithm. The landmark state location and covariance is initialised from observation data obtained when the landmark is promoted to confirmed status.

4.2 Results

Figure 6 shows the feature map consisting of 148 features and the path of the vehicle computed using the full SLAM algorithm that maintains the complete set of features in the state vector. In this experiment the robot starts at the origin, remaining stationery for approximately 15 seconds and then is driven along the corridor partly shown in Figure 4. Total duration of the test run is about 800 seconds. Vehicle path and the map generated by the full SLAM algorithm was taken to be the basis for comparison of the computationally efficient SLAM algorithm. Although the true path of the vehicle was not available, the map was verified by measuring the relative locations of some of the features. Furthermore, the full SLAM algorithm has previously been verified on a number of different environments [5, 4].

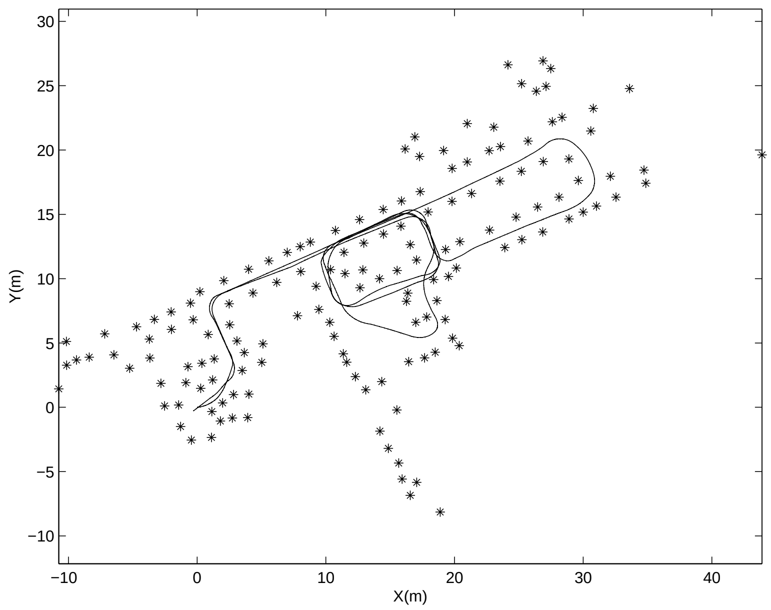

Figure 6: Vehicle path and the map obtained using the full SLAM algorithm

Figures 7 and **??** show the feature maps maintained, at two different instances, by the algorithm described in this paper. In this example, 90% of the features that are further than 15 m from the current vehicle location are removed from the map. The number of features maintained in the map changes as the vehicle moves. On the average, this strategy results in a map that contains about 70 features resulting in about an 8 fold reduction in the computational effort.

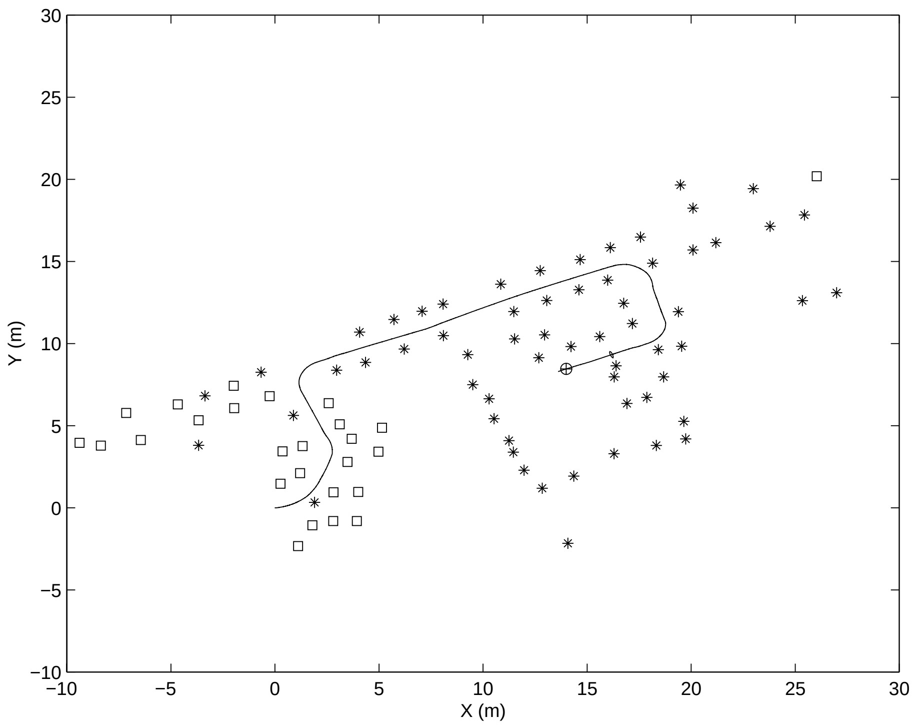

Figure 7: Vehicle path and the map obtained using the proposed computationally efficient SLAM algorithm after 100 seconds. The '*' show the features currently active whereas ' ' show the features that are inactive

Figure 8 that compares the vehicle location estimates obtained using the full slam and the proposed algorithm shows that the differences are hardly noticeable. Furthermore, Figure 9 shows that the increase in the uncertainty of the vehicle location estimates due to the removal of beacons is of the order of ??%.

5 Conclusions

In this paper a map management strategy to increase the computational efficiency of the estimation-theoretic solution to the simultaneous map building and localisation is presented. It is shown that deleting features from the map does not compromise the statistical consistency of the SLAM algorithm. The information loss due to the removal of a feature is quantified and a strategy to select features to be removed is described. Experimental results show that removing suitably selected features does not significantly increase the errors in the estimated vehicle location errors. However, the computational efficiency of the SLAM process is significantly reduced.

References

[1] D. H. Ballard and C. M. Brown. *Computer Vision.* Prentice-Hall, 1982.

[2] Y. Bar-Shalom and T.E. Fortman. *Tracking and Data Association.* Academic Press, 1988.

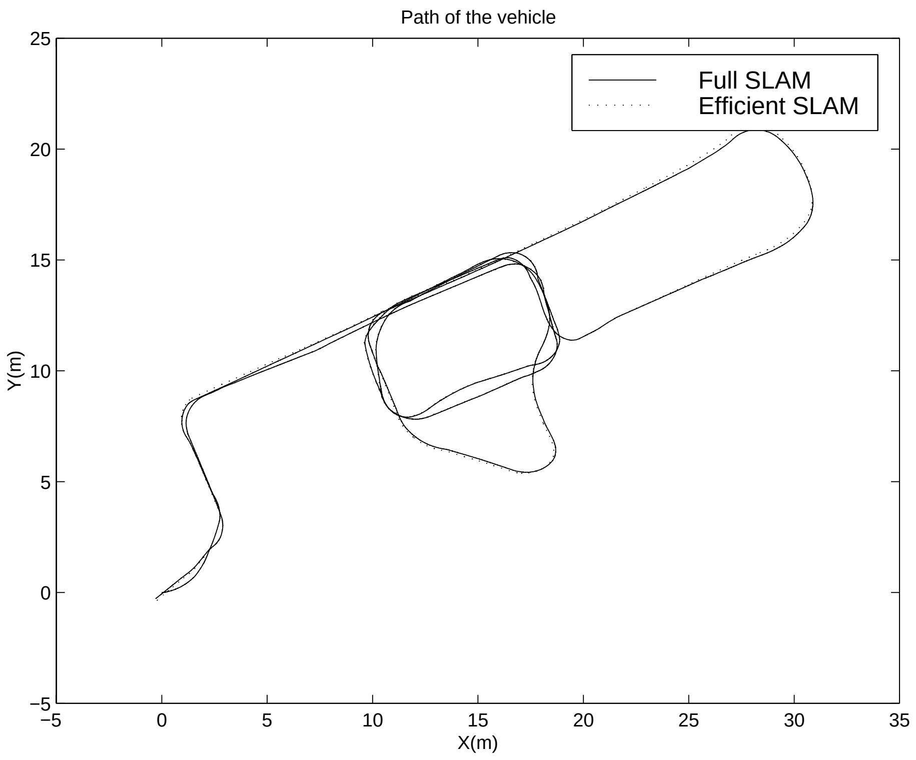

Figure 8: Vehicle path obtained using the full SLAM algorithm and the proposed computationally efficient SLAM algorithm

[3] J.A. Castellanos, J.M.M. Montiel, J. Neira, and J.D. Tardos. Sensor influence in the performance of simultaneous mobile robot localization and map building. In *Proc. 6th International Symposium on Experimental Robotics*, pages 203 – 212, Sydney, Australia, March 1999.

[4] S. Clark and G. Dissanayake. Simulataneous localisation and map building using millimeter wave radar to extract natural features. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 1316–1321, Detroit, USA, May 1999.

[5] M.W.M.G. Dissanayake, P. Newman, H.F. Durrant-Whyte, S. Clark, and M. Csobra. An experimental and theoretical investigation into simultaneous localisation and map building (slam). In *Proc. 6th International Symposium on Experimental Robotics*, pages 171 – 180, Sydney, Australia, March 1999.

[6] M.W.M.G. Dissanayake, P. Newman, H.F. Durrant-Whyte, S. Clark, and M. Csobra. An experimental and theoretical investigation into simultaneous localisation and map building (slam). Technical report, ACFR-TR-01-99, Australian Centre for Filed Robotics, Sydney, Australia, 1999.

[7] H. Durrant-Whyte. An autonomous guided vehicle for cargo handling applications. *Internation Journal of Robotics Research, Vol 15, No.5, October*, pages 407–440, 1996.

[8] J. Leonard and H. Durrant-Whyte. *Directed Sonar Sensing for Mobile Robot Navigation.* Kluwer Academic Publishers, 1992.

[9] J.J. Leonard and H.J.S. Feder. Experimental analysis of adaptive concurrent mapping and localisation using sonar. In *Proc. 6th International Symposium on Experimental Robotics*, pages 213 – 222, Sydney, Australia, March 1999.

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

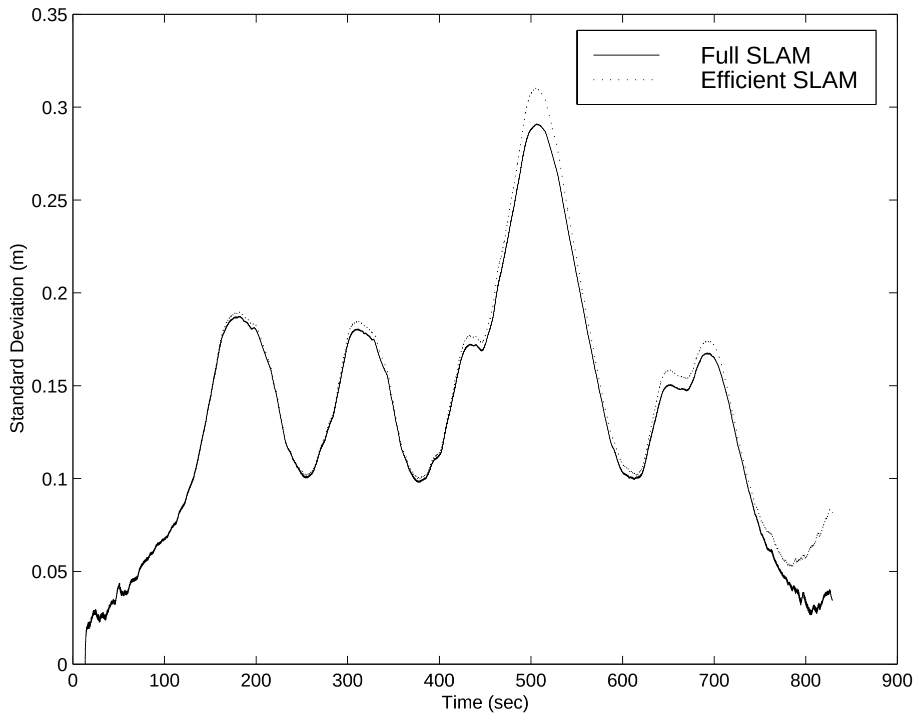

Figure 9: Standard deviation of the position estimated by the full SLAM algorithm and the proposed computationally efficient SLAM algorithm

[11] L. Kitchen P. Khan and E.M. Riseman. A fast line finder for vision guided robot navigation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(11):1098-1102*, 1990.

[12] J.K. Rosenblatt T. Bailey, E.M. Nebot and H.F. Durrant-Whyte. Robust distinctive place recognition for topological maps. In *International Conference of Field and Service Robotics*, pages 347–352, Pittsburgh, USA, August 1999.