Design of a Robust Real-Time Dynamic Localization System for Mobile Robots

G. A. Borges * **M.-J. Aldon** **

Robotics Department
LIRMM, UMR CNRS/Université Montpellier II, n^o. C55060
161, rue ADA. 34392 - Montpellier - Cedex 5 - France

Abstract. This paper proposes an approach for robust real-time map-based mobile robot dynamic localization. This is a real need for most navigation algorithms which do not consider that the vehicle stops to estimate its pose. Moreover, navigating in real cluttered environments requires the use of fast and robust pose estimators. The most used approach for map-based pose estimation is the Extended Kalman Filter. However, this approach is not robust and its use in real-time may be jeopardized for large environment maps. We propose a robust Weighted Least-Squares pose estimator which satisfies the robustness and real-time requirements for this problem. The robust approach has demonstrated superior performance in experimental comparison carried-out in a real cluttered environment.

1 Introduction

Map-based mobile robot localization has been extensively studied during the last decade [1][2][3], with a growing interest on simultaneous map-building and localization [4][5]. In an autonomous robot, the localization module is responsible for providing an estimate of the robot pose $\mathbf{z} = (x, y, \theta)$ for other important modules like navigation and task planner. Such an estimate is continuously updated by dead-reckoning, which is a relative pose estimation approach. However, as the uncertainty associated to dead-reckoning grows without bounds when the robot moves, it becomes necessary to perform pose estimation using an absolute approach. Map-based absolute localization approaches are suitable for autonomous mobile robots operating in indoor environments. In such environments, GPS cannot be employed and the use of artificial landmarks increases the installation and maintenance costs. However, the robot must be equipped with an appropriate exteroceptive sensor, usually a laser rangefinder.

Nowadays, in the research community, the most common map-based approaches for robot localization and navigation employ occupancy grids [6] or geometrical features [1]. With occupancy grids, the localization module computes the conditional probability of the robot to be in each cell of the three-dimensional x-y-θ space of robot pose given the current sensor measurements. The estimated pose is the center of the cell corresponding to the largest probability. However, given the computing time and memory requirements for occupancy grid-based localization, its use in real-time is possible if some mechanisms like sensor data filtering and cell selection are applied [6].

In pose estimation using geometrical features we have a global environment map $\mathcal{M}$ that looks like a 2-D top-view representation of the navigating environment. The map is composed of geometrical features which correspond to some structures usually found in indoor environments. The commonly used features are lines, representing walls and sides of polygonal obstacles, and points, corresponding to vertical edges or small obstacles. In order to estimate the robot pose, we have two main alternatives: matching local sensor measurements to global map line features [3], or matching a local map $\mathcal{M}'$ built from sensor measurements to global map features [1] [2]. The main interest on using a local map is that we can employ different types of features. It results in a more complete environment model and reduces the probability of the robot to get lost if the number of lines is not sufficient to estimate the robot pose. Thus, this is why we have chosen to work with the local map approach.

* E-mail: borges@lirmm.fr
** E-mail: aldon@lirmm.fr

We can point out that little attention has been given for real-time aspects of mobile robot localization. In this paper, we are interested on theoretical and practical aspects of real-time dynamic localization. In the context of *dynamic localization*, the pose estimation is performed during robot motion. It does not mean that other researchers have not taken into account for robot motion during data acquisition and pose update. However, we have not found references that directly treat the problems related to localization in motion. In [7], we presented real range images which are largely deformed when acquired with robot in motion. Such deformation results in false matches and important errors in localization. In this paper we present a dynamic localization architecture which incorporates mechanisms for real-time sensor data acquisition, temporary data storage and motion compensation.

The theoretical foundations of this research are related to robustness of the map-based pose estimator. We are interested on fast pose computation and on robustness with respect to false correspondences, which often occurs in cluttered environments. The most used approach for pose estimation is the Extended Kalman Filter (*EKF*) [8][9]. However, the filter formulation relies on some assumptions which are not always verified in practice: (i) the uncertainties are supposed to follow a Gaussian distribution and (ii) the measurement model is approximated by its first order Taylor expansion. Alternative approaches rely on iterated optimization. These approaches estimate the robot pose by matching scan points to map points [10] or lines [11]. The simultaneous use of geometrical features as lines and points in a non-iterated Weighted Least-Squares (*WLS*) framework has been introduced in our previous papers [7][12]. This paper presents a new robust iterated formulation of the *WLS* which performs very well and is readily applicable for real-time pose estimation. The same algorithm also provides an estimate of the uncertainties related to the pose estimate.

This paper is organized as follows. The real-time architecture for dynamic localization is discussed in Section 2. Such an architecture allows the robot to build local maps which are consistent with the environment even when the robot is moving. Pose estimation using the *EKF* and *WLS* algorithms is briefly discussed in Section 3. Section 4 presents our new robust formulation of the *WLS* pose estimator. The experimental evaluation of the pose estimators is presented in Section 5.

2 The system architecture for dynamic localization

The main difference between static and dynamic localization is that for the last case the sensor data acquisition and pose computing times can deteriorate the quality of the pose estimation results. In this work we describe the architecture of Figure 1, which employes some mechanisms (shown in dark gray blocks) to deal with such problems. The robot pose estimate $\hat{\mathbf{z}}(t)$ is updated every 5 ms by integrating the discrete kinematics model from the initial state $\hat{\mathbf{z}}(t-5ms)$ and using the current encoders measurements $\mathbf{q}(t)$ and $\dot{\mathbf{q}}(t)$. The current pose estimate $\hat{\mathbf{z}}(t)$ is then returned to the pose queue which is a circular data structure that allows to keep in memory the pose estimates obtained during the last 5 seconds. A similar queue exists for the encoders data. At any time, a past reading can be recovered from these queues. In order to guarantee mutual exclusion, the access to each queue is controlled by atomic macros.

As the robot moves, the pose estimate provided by odometry becomes unreliable. Thus, it is necessary to correct the robot pose from the map-based localization method. This is done periodically, when the task responsible for global pose estimation becomes active. At this time, this task waits for the next available range image, whose end-of-acquisition occurs at time t. From this, the following steps are executed:

1. *Motion compensation and bias correction*: In this work, the laser rangefinder used for building local maps provides range images every 125 ms. The time interval corresponding to a complete scan (270°) is approximately 93 ms. Considering translational and rotational speeds of 50 m/s and 30 $^\circ/s$, during a scan, the robot may perform a displacement of 4.63 cm and a rotation of 2.79°. In [7] we have presented some range images which are largely deformed with such robot speeds. Thus, a range image correction algorithm is used to preserve the accuracy of local maps [7]. Moreover, after motion correction, range bias is compensated by using an identified polynomial function. Previous experiments have shown that this bias may be close to 20 cm for an obstacle located at 5 m;
2. *Local map building*: A local map $\mathcal{M}'(t)$ composed of features representing the observed obstacles is built from the corrected scan image. These features are lines, extracted using a new segmentation algorithm [13] and points. They correspond to the ends of lines which are also breakpoints, *i.e.* points corresponding to large discontinuities in the range image. The line extraction algorithm evaluated in [13] is fast enough for real-time use;

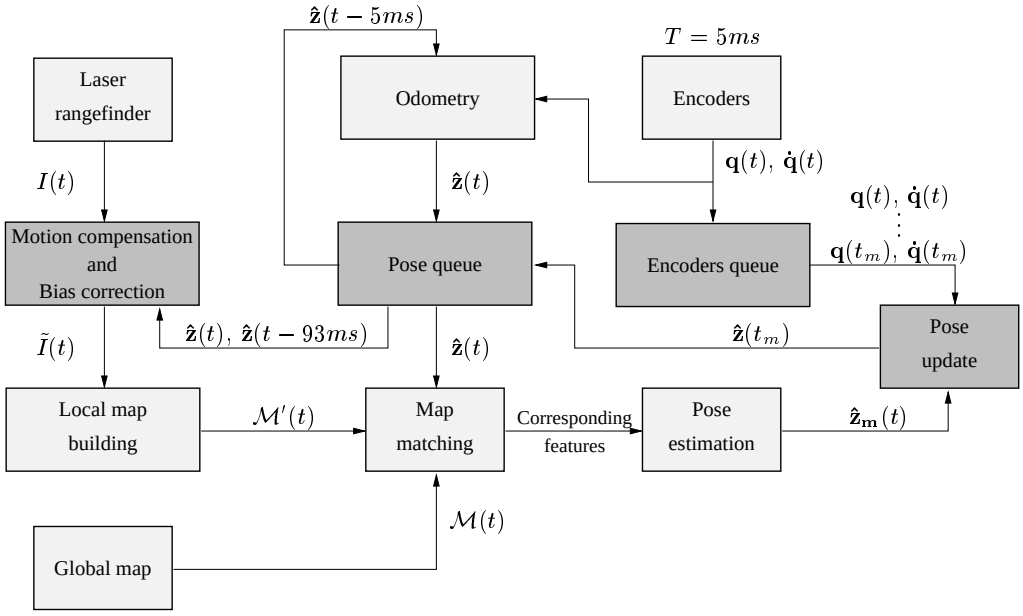


Fig. 1. Simplified block diagram of the map-based dynamic pose estimation system.

3. *Map matching*: The local map $\mathcal{M}'(t)$ is matched against the global environment map $\mathcal{M}(t)$, resulting in a set of corresponding geometrical features. The correspondences satisfy a maximum threshold on the Mahalanobis distance, which takes into account the estimate $\hat{\mathbf{z}}(t)$, recovered from the pose queue. This procedure results in a set of N feature correspondences. The n^{th} correspondence is represented by means of the feature parameters vectors $\mathbf{f}_n$ and $\mathbf{f}'_n$;
4. *Pose estimation*: From the corresponding features, an estimate $\hat{\mathbf{z}}_m(t)$ of the robot pose at time t is obtained at time t_m ($t_m > t$) by using a pose estimator. The pose estimate minimizes some statistical or numerical distance between the corresponding features obtained from the map matching phase. The pose estimators are discussed in Sections 3 and 4;
5. *Pose update*: The new pose estimate $\hat{\mathbf{z}}_m(t)$, computed in step 4, is referred to the time t. Considering the elapsed time $t_m - t$ for all processing described above, that may be in the order of some hundred milliseconds, the robot may be displaced of some decimeters and rotated of some degrees. Thus, the new pose estimate no more corresponds to the current time. That is why a pose time update is required. For this purpose, all encoders data stored in the queue from t to t_m are recovered to integrate the discrete kinematics model from the initial state $\hat{\mathbf{z}}_m(t)$ until the actual time. This phase results in the actual pose estimate $\hat{\mathbf{z}}(t_m)$ which is updated in the pose queue. It will be continuously updated by odometry until the next map-based localization cycle.

3 Pose estimation from geometrical map correspondences

In this work, the global and local maps are attached to the reference frames $(O, \mathcal{X}, \mathcal{Y})$ and $(O', \mathcal{X}', \mathcal{Y}')$, respectively. Let (p'_x, p'_y) be the coordinates of a static point in $(O', \mathcal{X}', \mathcal{Y}')$, and (p_x, p_y) its coordinates in $(O, \mathcal{X}, \mathcal{Y})$. These coordinates are related by

$$\begin{pmatrix} p'_x \\ p'_y \end{pmatrix} = \mathbf{R}(\theta) \left\{ \begin{pmatrix} p_x \\ p_y \end{pmatrix} - \begin{pmatrix} x \\ y \end{pmatrix} \right\}, \quad \text{with } \mathbf{R}(\theta) = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}. \tag{1}$$

3.1 Kalman filtering

The most used approach for map-based pose estimation is the Extended Kalman Filter (*EKF*) [8][9]. The formulation of the mobile robot localization problem using the *EKF* is elegant and allows to fuse multisensory data [5]. In a map-based mobile robot localization system, this filter embodies the prediction and estimation

phases. In the prediction phase, which corresponds to the odometric equations, the robot pose estimate $\hat{\mathbf{z}}(t)$ is continuously updated by odometry, as well as its associated covariance matrix $\mathbf{P}_{\hat{\mathbf{z}}}(t)$.

When new measurements become available at time t, which in this case is a set of N feature correspondences, the *EKF* computes an estimate $\hat{\mathbf{z}}_m(t)$ based on the following measurement model for the n^{th} correspondence: $\mathbf{f}'_n = \mathbf{h}(\hat{\mathbf{z}}(t), \mathbf{f}_n) + \mathbf{w}_n(t)$,with $\mathbf{w}_n(t)$ representing the measurement noise which embodies feature and model uncertainties. It is commonly assumed that $\mathbf{w}_n$ is a Gaussian noise with $E\{\mathbf{w}_n\} = \mathbf{0}$ and $E\{\mathbf{w}_n \cdot \mathbf{w}_n^T\} = \mathbf{R}_n$.

From the different formulations of the *EKF*, the most used for pose estimation is the batch *EKF*. The batch *EKF* integrates the N measurements in only one step by using the following formulation [14]:

$$\begin{array}{ll} \boldsymbol{\nu} = \mathbf{f}' - \mathbf{h}(\hat{\mathbf{z}}(t), \mathbf{f}), & \mathbf{S} = \nabla\mathbf{h} \cdot \mathbf{P}_{\hat{\mathbf{z}}}(t) \cdot \nabla\mathbf{h}^T + \mathbf{R}, \quad \text{(innovation } \boldsymbol{\nu} \text{ and its covariance matrix } \mathbf{S}\text{)} \\ \mathbf{K} = \mathbf{P}_{\hat{\mathbf{z}}}(t) \cdot \nabla\mathbf{h}^T \cdot \mathbf{S}^{-1}, & \quad \text{(Kalman filter gain)} \\ \hat{\mathbf{z}}_m(t) = \hat{\mathbf{z}}(t) + \mathbf{K} \cdot \boldsymbol{\nu}, & \mathbf{P}_{\hat{\mathbf{z}}_m}(t) = \mathbf{P}_{\hat{\mathbf{z}}}(t) - \mathbf{K} \cdot \mathbf{S} \cdot \mathbf{K}^T. \quad \text{(estimate } \hat{\mathbf{z}}_m \text{ and its covariance matrix } \mathbf{P}_{\hat{\mathbf{z}}_m}\text{)} \end{array} \tag{2}$$

In the above equations, all measurements are stacked in high dimensional structures. $\mathbf{S}$ is composed of two terms: $\nabla\mathbf{h} \cdot \mathbf{P}_{\hat{\mathbf{z}}}(t) \cdot \nabla\mathbf{h}^T$, representing the covariance term given by the pose uncertainty with $\nabla\mathbf{h}^T = \left(\partial\mathbf{h}^T(\hat{\mathbf{z}}(\mathbf{t}), \mathbf{f_1})/\partial\hat{\mathbf{z}}, \ldots, \partial\mathbf{h}^T(\hat{\mathbf{z}}(\mathbf{t}), \mathbf{f_N})/\partial\hat{\mathbf{z}}\right)$, and $\mathbf{R}$, which is the term given by the feature uncertainties. $\mathbf{R} = diag(\mathbf{R}_1, \cdots, \mathbf{R}_N)$ with

$$\mathbf{R}_n = \mathbf{P}_{\mathbf{f}'_n} + \frac{\partial\mathbf{h}}{\partial\mathbf{f}_n}\mathbf{P}_{\mathbf{f}_n}\frac{\partial\mathbf{h}^T}{\partial\mathbf{f}_n}. \tag{3}$$

3.2 Weighted Least-Squares

In previous papers [7][12], we have introduced a Weighted Least-Squares (*WLS*) optimal pose estimator from local and global feature correspondences. The *WLS* algorithm embodies simultaneously line to line and point to point correspondences and gives the pose estimate that better match set of features given a weighted residual cost function. This algorithm also provides a consistent estimation of the associated covariance matrix $\mathbf{P}_{\hat{\mathbf{z}}_m}$. The pose estimate is computed in a non-iterated manner by solving a set of non-linear equations.

The estimation of the robot pose using simultaneously lines and points is achieved by changing the features representation to a unified form. A point is represented by $\mathbf{f} = (x_p, y_p)^T$ which are its coordinates in the frame $\mathcal{XY}$. A line is represented by $\mathbf{f} = (x_l, y_l)^T$ which are the coordinates of the point on the line that is the closest to the origin of $\mathcal{XY}$. From the classical polar representation (ρ, α), we have $x_l = \rho \cdot \cos(\alpha)$ and $y_l = \rho \cdot \sin(\alpha)$.

The above definitions are equivalent for a given feature vector $\mathbf{f}'$ in the $\mathcal{X}'\mathcal{Y}'$ coordinate frame. Using this unified representation, we can show that the global to local feature parameter transformation function $\mathbf{h}(\mathbf{z}, \mathbf{f})$ is given by

$$\mathbf{h}(\mathbf{z}, \mathbf{f}) = \mathbf{R}(\theta) \cdot (\mathbf{f} - \mathbf{E} \cdot \mathbf{t}), \text{ with } \mathbf{t} = (x, y)^T \text{ and } \mathbf{E} = \begin{pmatrix} a^2 & c \\ c & b^2 \end{pmatrix}. \tag{4}$$

The entries of the $\mathbf{E}$ matrix depend on the global feature type : $a = b = 1$ and $c = 0$ for a point, and $a = \cos(\alpha)$, $b = \sin(\alpha)$ and $c = a \cdot b$ for a line. The *WLS* algorithm computes the pose estimate $\hat{\mathbf{z}}_m$ as the $\hat{\mathbf{z}}$ that minimizes the weighted criterion

$$J(\hat{\mathbf{z}}) = \sum_{n=1}^{N} \mu_n \cdot \|\mathbf{r}_n\|^2, \tag{5}$$

with $\mathbf{r}_n = \mathbf{f}'_n - \mathbf{h}(\hat{\mathbf{z}}, \mathbf{f}_n)$ being the n^{th} residual distances between the $\mathbf{f}'_n$ local feature and the corresponding local representation $\mathbf{h}(\hat{\mathbf{z}}, \mathbf{f}_n)$ of the global feature $\mathbf{f}_n$. μ_n is a positive scalar factor that weight the n^{th} correspondence.

We can show that $J(\hat{\mathbf{z}})$ is a non-linear non-convex criterion. Thus, in order to minimize $J(\hat{\mathbf{z}})$, the procedure adopted consists in finding all local minima $\hat{\mathbf{z}}^*$, compute their associated value of J and choose as pose estimate $\hat{\mathbf{z}}$ the $\hat{\mathbf{z}}^*$corresponding to the $J(\hat{\mathbf{z}}^*)$ closest to zero. In doing so, all local minima of $J(\hat{\mathbf{z}})$ satisfy the following set of equations:

$$\frac{\partial J(\hat{\mathbf{z}})}{\partial\hat{x}} = \frac{\partial J(\hat{\mathbf{z}})}{\partial\hat{y}} = \frac{\partial J(\hat{\mathbf{z}})}{\partial\hat{\theta}} = 0. \tag{6}$$

This results in the following system of equations:

$$\mathbf{M}\hat{\mathbf{t}} - \mathbf{N}\hat{\boldsymbol{\beta}} = \mathbf{q}, \qquad \hat{\boldsymbol{\beta}}^T \cdot (\mathbf{r} + \mathbf{S}\mathbf{N}^T\hat{\mathbf{t}}) = 0, \tag{7}$$

with $\hat{\boldsymbol{\beta}}^T = \big(\cos(\hat{\theta})\ \sin(\hat{\theta})\big)$, and

$$\mathbf{M} = \sum_{n=1}^{N} \mu_n \begin{pmatrix} c_n^2 + a_n^4 & c_n(a_n^2 + b_n^2) \\ c_n(a_n^2 + b_n^2) & c_n^2 + b_n^4 \end{pmatrix}, \quad \mathbf{N} = \sum_{n=1}^{N} \mu_n \begin{pmatrix} -a_n^2 x'_n - c_n y'_n & a_n^2 y'_n - c_n x'_n \\ -c_n x'_n - b_n^2 y'_n & c_n y'_n - b_n^2 x'_n \end{pmatrix} \tag{8}$$

$$\mathbf{q} = \sum_{n=1}^{N} \mu_n \begin{pmatrix} c_n y_n + a_n^2 x_n \\ c_n x_n + b_n^2 y_n \end{pmatrix}, \quad \mathbf{r} = \sum_{n=1}^{N} \mu_n \begin{pmatrix} -x_n y'_n + y_n x'_n \\ -y_n y'_n - x_n x'_n \end{pmatrix}, \quad \mathbf{S} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{9}$$

In the above equations, the variables x_n and y_n (resp. x'_n and y'_n) are the components of the $\mathbf{f}_n$ (resp. $\mathbf{f}'_n$) feature parameter vector. From eqs. (7) we obtain

$$\hat{\boldsymbol{\beta}}^T \Phi \hat{\boldsymbol{\beta}} + \hat{\boldsymbol{\beta}}^T \boldsymbol{\zeta} = 0, \tag{10}$$

with $\Phi = \mathbf{S}\mathbf{N}^T\mathbf{M}^{-1}\mathbf{N}$ and $\zeta = \mathbf{r} + \mathbf{S}\mathbf{N}^{\mathbf{T}}\mathbf{M}^{-1}$. We can verify that $\Phi = (\zeta - \mathbf{r})\mathbf{N}$, and that $\mathbf{N}^T\mathbf{M}^{-1}\mathbf{N}$ is a symmetric matrix since $\mathbf{M}$ is also symmetric. Let the entries of Φ and $\boldsymbol{\zeta}$ be referred to as

$$\Phi = \begin{pmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{pmatrix}, \quad \zeta = \begin{pmatrix} \zeta_1 \\ \zeta_2 \end{pmatrix}.$$

Thus, $\mathbf{N}^T\mathbf{M}^{-1}\mathbf{N}$ being symmetric and given the form of matrix $\mathbf{S}$, we have $\Phi_{11} = -\Phi_{22}$. With this simple relation, further development of eq. (10) results in

$$\Phi_{11}\cos(2\hat{\theta}) + \frac{(\Phi_{12} + \Phi_{21})}{2}\sin(2\hat{\theta}) + \zeta_1\cos(\hat{\theta}) + \zeta_2\sin(\hat{\theta}) = 0. \tag{11}$$

Maple, a commercial symbolic mathematics software produced by Waterloo Maple, Inc., computes four roots $\hat{\theta}_i^*$, $i = 1, \ldots, 4$, for eq. (11). These solutions are the rotation components of the local minima of J, and candidate solutions to $\hat{\theta}$. The $\hat{\mathbf{t}}_i^*$ corresponding to each $\hat{\theta}_i^*$ is obtained from eq. (7) and given by

$$\hat{\mathbf{t}}_i^* = \mathbf{M}^{-1}(\mathbf{N}\hat{\boldsymbol{\beta}}_i^* + \mathbf{q}), \tag{12}$$

with $\hat{\boldsymbol{\beta}}_i^* = \big(\cos(\hat{\theta}_i)\ \sin(\hat{\theta}_i)\big)^T$. Thus, each $\hat{\mathbf{z}}_i^* = ((\hat{\mathbf{t}}_i^*)^T, \hat{\theta}_i)^T$ corresponds to a local minimum of J and satisfies eq. (6). As mentioned before, the optimal parameters $\hat{\mathbf{t}}$ and $\hat{\theta}$ are found by evaluating $J(\hat{\mathbf{z}}_i^*)$ for each solution $\hat{\mathbf{z}}_i^*$ and choosing $\hat{\mathbf{z}}_m$ as the $\hat{\mathbf{z}}_i^*$ minimizing J. In doing so, the minimization of J is achieved and the estimate $\hat{\mathbf{z}}_m$ is optimal in the weighted least-squares sense.

The associated covariance matrix $\mathbf{P}_{\hat{\mathbf{z}}_m}(t)$ of $\hat{\mathbf{z}}_m(t)$ is estimated by propagating the uncertainties associated with each variable $\mathbf{r}_n$, $\mathbf{f}_n$ and $\mathbf{f}'_n$, to the pose estimates $\hat{\mathbf{z}}_m$. This is done by applying the covariance propagation methodology described in [15]. From the mathematical development reported in [15], we have

$$\mathbf{P}_{\hat{\mathbf{z}}_m} = \left(\frac{\partial \mathbf{g}}{\partial \hat{\mathbf{z}}_m}\right)^{-1} \cdot (\Lambda_{\mathbf{r}} + \Lambda_{\mathbf{f}} + \Lambda_{\mathbf{f}'}) \cdot \left(\frac{\partial \mathbf{g}}{\partial \hat{\mathbf{z}}_m}\right)^{-1}, \tag{13}$$

with $\mathbf{g} = \partial J / \partial \hat{\mathbf{z}}_m$ and

$$\Lambda_{\mathbf{r}} = \sum_{n=1}^{N} \frac{\partial \mathbf{g}}{\partial \mathbf{r}_n} \cdot \mathbf{P}_{\mathbf{r}} \cdot \frac{\partial \mathbf{g}}{\partial \mathbf{r}_n}^T, \quad \Lambda_{\mathbf{f}} = \sum_{n=1}^{N} \frac{\partial \mathbf{g}}{\partial \mathbf{f}_n} \cdot \mathbf{P}_{\mathbf{f}_n} \cdot \frac{\partial \mathbf{g}}{\partial \mathbf{f}_n}^T, \quad \Lambda_{\mathbf{f}'} = \sum_{n=1}^{N} \frac{\partial \mathbf{g}}{\partial \mathbf{f}'_n} \cdot \mathbf{P}_{\mathbf{f}'_n} \cdot \frac{\partial \mathbf{g}}{\partial \mathbf{f}'_n}^T.$$

The feature covariance matrices $\mathbf{P}_{\mathbf{f}_n}$ and $\mathbf{P}_{\mathbf{f}'_n}$ are provided as inputs, and $\mathbf{P}_{\mathbf{r}}$ is the estimated residual covariance matrix. We use the weighted sample covariance as an estimate of $\mathbf{P}_{\mathbf{r}}$. It is computed by $\mathbf{P}_{\mathbf{r}} = \sum_{n=1}^{N} \mu_n(\mathbf{r}_n \cdot \mathbf{r}_n^T) / \sum_{n=1}^{N} \mu_n$. $\partial \mathbf{g} / \partial \hat{\mathbf{z}}_m$ is the Hessian of J with respect to $\hat{\mathbf{z}}_m$. This matrix is always non-singular on the minimum $\hat{\mathbf{z}}_m$ of the cost function. Its reciprocal always exists and is used to compute $\mathbf{P}_{\hat{\mathbf{z}}_m}$.

Remark 1 *The* WLS *provides the optimal pose estimate that minimizes the weighted residual error between the corresponding features after global to local transformation.* EKF *computes an approximated sub-optimal estimate that takes into account all system uncertainties. In the* WLS*, the correspondence weights may be computed in order to take into account the feature uncertainties.*

Remark 2 *It must be pointed out that the* WLS *algorithm can only give pose estimates if the feature correspondences allow a solution. In such a case, a simple test on the rank of the* $\mathbf{M}$ *matrix is sufficient for fault detection.*

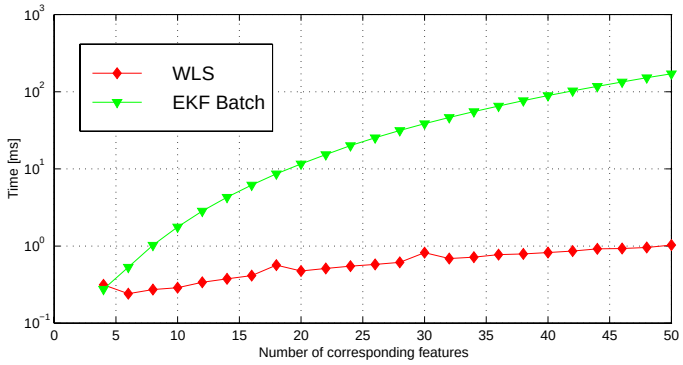


Fig. 2. *WLS* and *EKF* computing times in function of the number of corresponding features.

3.3 Complexity and computing time requirements

In the *EKF*, the computing of $\mathbf{K}$ involves the inversion of the $\mathbf{S}$ matrix, which has dimension $2N \times 2N$. Being $\mathbf{S}$ a non-diagonal matrix, it results in an $\mathcal{O}(N^3)$ computing complexity for the inverse of $\mathbf{S}$ [16]. Thus the *EKF* complexity is $\mathcal{O}(N^3)$. Without the need of inverting matrices whose size depends of the number of correspondences, the *WLS* implementation presents an $\mathcal{O}(N)$ polynomial complexity. So, this property allows the *WLS* to give pose estimates faster than the *EKF*, which is verified even for a few number of corresponding features. In Figure 2, the computing time of the *C* language implementations of these algorithms was measured in a real-time kernel, the *RTX* produced by *Venturcom Inc*, running on a PII 333 MHz CPU. The corresponding features were randomly generated. For 50 corresponding features, the *WLS* is two order of magnitude faster than the *EKF*. This results in a faster pose estimation, even with large environment maps.

4 Robust pose estimation from geometrical map correspondences

In order to achieve a robust pose estimation algorithm, the correspondence weights μ_n of the *WLS* algorithm must be computed using robust weighting functions. It is well known that the Mahalanobis distance is not a robust correspondence measure. Furthermore, in order to have a consistent uncertainty measure, it is a common practice in modelling odometry to take into account for encoders and model uncertainties in the updating of the covariance matrix $\mathbf{P}_{\hat{\mathbf{z}}}$. As non-systematic errors are not easy to model, the uncertainties associated to the model approximation are increased to obtain a consistent estimate of $\mathbf{P}_{\hat{\mathbf{z}}}$. It means that even if the non-systematic errors are not very important for some parts of robot trajectory, $\mathbf{P}_{\hat{\mathbf{z}}}$ may be larger than the real pose error covariance. Thus, if the current pose estimate provided by odometry has a large uncertainty $\mathbf{P}_{\hat{\mathbf{z}}}$, the Mahalanobis distance tends to be smaller and some extra correspondences may be confirmed. These correspondences present large residuals, which can be seen as outliers.

In this section we present a robust version of the *WLS* algorithm that can reject correspondences with large residuals. We have chosen the following weighting function:

$$\mu_n = \exp(-\frac{\|\mathbf{r}_n\|}{\eta}), \text{ with } \eta = -2.6 \cdot median(\|\mathbf{r}_1\|, \ldots, \|\mathbf{r}_N\|)/\ln(0.1). \tag{14}$$

In eq. (14), $\mathbf{r}_n = \mathbf{f}'_n - \mathbf{h}(\hat{\mathbf{z}}, \mathbf{f}_n)$ is the n^{th} residual in the unified feature representation for the current pose estimate $\hat{\mathbf{z}}$ and η determines the acceptation window width. The robust *WLS* has an iterated form, where for each iteration the weights are updated as (14). For the first iteration, all weights are equal to 1. In this way, the robust *WLS* first computes a pose estimate which may be better then the $\hat{\mathbf{z}}$ provided by odometry. In our implementation, the maximum number of iterations is fixed at 5. During the iterations, only the pose estimate is computed. Its associated covariance matrix is computed using the weights of the last iteration. This allows to obtain a faster algorithm. If we use the robust Tukey's biweight function [17], we would obtain an iterated M-estimator. However, in order to converge, the M-estimator may require a large number of iterations. In our experiments, we have used eq. (14) that has presented good performance in few iterations.

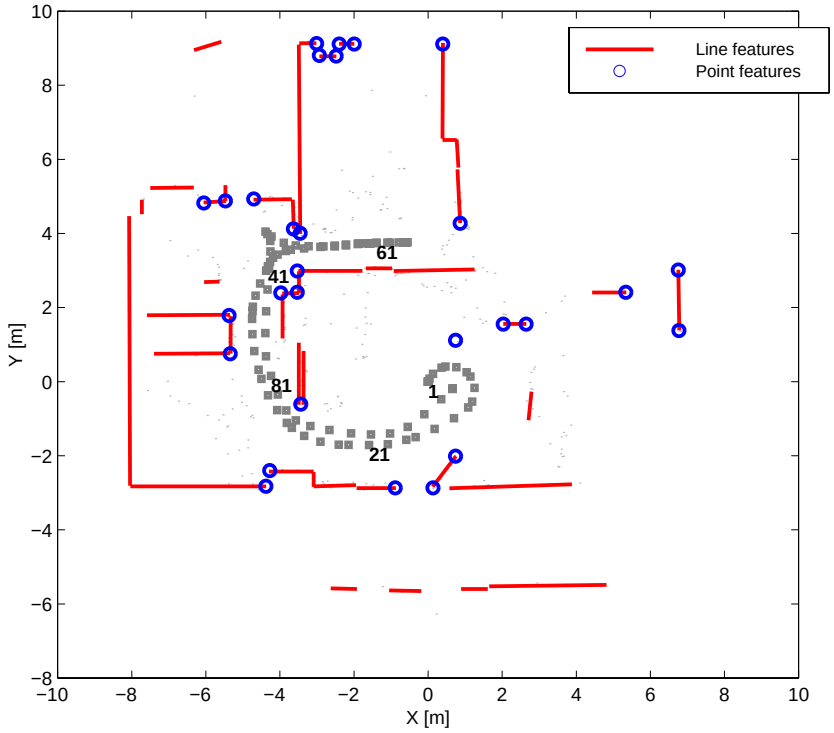


Fig. 3. The global map of the experimental environment. The superimposed scans are shown to illustrate how cluttered the environment is.

5 Experimental evaluation

The dynamic localization architecture of Figure 1 was implemented in our mobile robot Omni equipped with a personal computer based on a PII 333 MHz CPU. The real-time extension RTX[3] for Windows NT 4.0 fills the real-time requirements of this project. Interprocess communication objects allow communication between hard real-time tasks running on RTX (*i.e.* the dynamic localization architecture) and soft real-time tasks running on Windows NT (*i.e.* user interface and disk storage of data). With this system, it is allowed to acquire sensor data for off-line evaluation.

The global map used for this experimental evaluation (Figure 3) was hand built from a set of superimposed range images acquired in a real cluttered environment. The reference robot poses are represented by small squares. In this experiment, there were acquired 95 range images with the robot in motion during 95 seconds. Some indexes associated with the measured reference poses are also shown. The reference poses are provided by fusing a high precision laser gyrometer and encoders data.

In this section, we compare the performance of two pose estimators: the *EKF* (Section 3.1) and the robust *WLS* (Section 4). The unified feature representation was used for the two algorithms. The performance measures are the mean absolute deviations ε_x, ε_y and ε_θ of the respective x, y and θ pose estimates with respect to the reference poses.

We have performed three experiments with these data, denoted as experiments A, B and C. In order to allow a comparison of the *EKF* with the robust *WLS* in the same conditions, its evaluation was carried-out off-line but using the real data acquired for this experiment, *i.e.* laser range images, encoders readings and reference poses at time t. Thus, we would expect the same results as if the algorithms were running on-line. The three experiments are described below:

– *Experiment A*: in this experiment, $\hat{\mathbf{z}}(t)$ is provided by odometry only, which is far less precise than the reference pose. $\hat{\mathbf{z}}(t)$ is not simulated, but computed from the real encoders measures acquired during the experiment. This experiment allows to evaluate the performance of the different approaches for robots that are not equipped with an expensive high-precision gyrometer;

[3] RTX is manufactured by Venturcom Inc.

Table 1. Experimental performance measures for the robust *WLS* and *EKF* algorithms

	robust *WLS*			batch *EKF*		
	ε_x	ε_y	ε_θ	ε_x	ε_y	ε_θ
Experiment A	**6.88** *cm*	**5.77** *cm*	1.26 °	7.54 *cm*	8.21 *cm*	**1.06** °
Experiment B	**8.82** *cm*	**9.43** *cm*	1.47 °	12.15 *cm*	15.23 *cm*	**1.35** °
Experiment C	**8.61** *cm*	**7.58** *cm*	**1.13** °	11.63 *cm*	12.62 *cm*	1.22 °

- *Experiment B*: in this experiment, $\hat{\mathbf{z}}(t)$ is simulated as being the reference pose contaminated with additive Gaussian noise with covariance matrix $\mathbf{P}_w = diag\left((0.25m)^2\ (0.25m)^2\ (3°)^2\right)$. It must be pointed out that such uncertainty is larger than the one of the experiment *A*. This experiment evaluates the robustness of the different approaches when the provided odometric pose estimate $\hat{\mathbf{z}}(t)$ is not accurate;
- *Experiment C*: in this experiment, $\hat{\mathbf{z}}(t)$ is simulated as being the reference pose perturbed by two random noises: one Gaussian noise with covariance matrix $\mathbf{P}_w = diag\left((0.15m)^2\ (0.15m)^2\ (2°)^2\right)$, and another noise with uniform distribution in the range $[-0.3m, 0.3m]$ for $\hat{x}(t)$ and $\hat{y}(t)$, and in the range $[-3°, 3°]$ for $\hat{\theta}(t)$. The Gaussian noise represents the uncertainty propagated by odometry, and the uniform noise represents wheel slippage or other non-Gaussian error sources. The map matching and pose estimation procedures have knowledge of the Gaussian effect only, *i.e.* $\mathbf{P}_{\hat{\mathbf{z}}}(t) = \mathbf{P}_w$. This experiment evaluates the robustness of the different approaches when the errors on $\hat{\mathbf{z}}(t)$ do not follow a Gaussian distribution.

Table 1 presents the performance measures of the algorithms for the three experiments. The best performances for each experiment are in bold font. As a result, ε_θ is almost the same for the two algorithms. However, the robust *WLS* gave better results for x and y estimation in all experiments. The results of the robust *WLS* are still better for experiments *B* and *C*. In these experiments, the theoretical foundations on which the *EKF* is based are not verified. In experiment *B*, $\hat{\mathbf{z}}(t)$ is far from the true pose, and truncation errors caused by the linearization of the measurement model jeopardize the *EKF* performance. In experiment *C*, the uncertainties on $\hat{\mathbf{z}}(t)$ are not Gaussian.

Figure 4 shows the pose estimation errors with respect to the reference poses for the experiments *A*, *B*, and *C*. These errors are computed . The estimated $[-\sigma, \sigma]$ intervals are represented in gray, with σ being the estimated standard-deviation obtained from $\mathbf{P}_{\hat{\mathbf{z}}_m}(t)$. Based on the $[-\sigma, \sigma]$ intervals, we can see that the *WLS* estimated uncertainties are generally consistent with respect to the observed errors. Thus, an automatic pose supervisor may take decisions on the acceptance or not of a new pose estimate based on statistical tests on the $[-3\sigma, 3\sigma]$ limits.

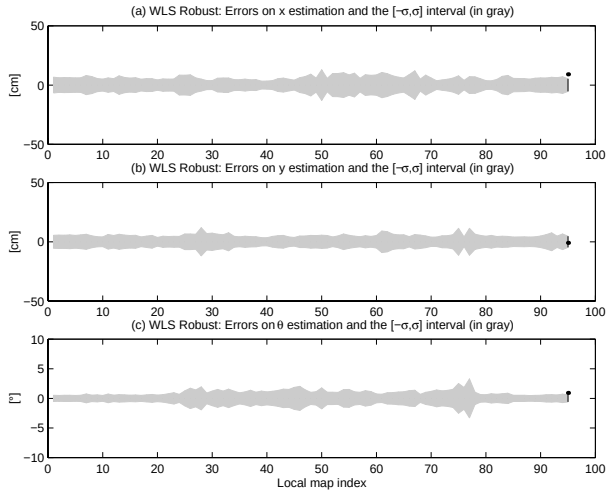
In some cases the number of matched features reached 50. According to Figure 2, for these cases, the required computing times are on about 100 *ms* for the *EKF*, and 1 *ms* for one iteration of the robust *WLS*. It means that using the *WLS*, pose estimation can be performed at a superior frequency and with a deterministic behaviour appropriate for real-time design.

The interested reader can find further experimental results for absolute pose estimation in [7], and for relative motion estimation in [12].

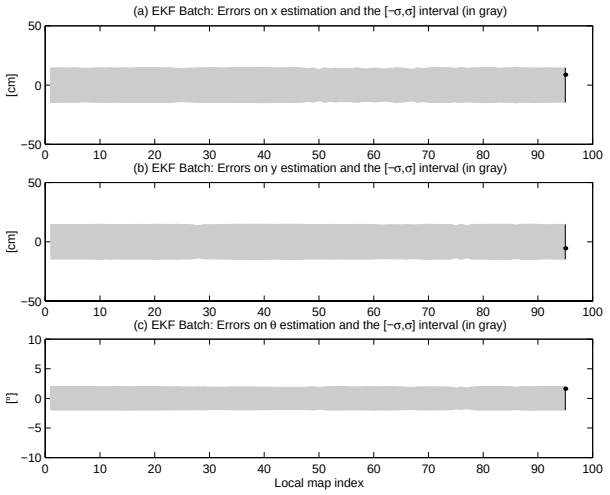
6 Conclusion

This article presented the design of a dynamic localization architecture for real-time mobile robot localization. Design details were discussed and important factors as computing time complexity were taken into account. In an experimental evaluation, two algorithms for geometrical map-based pose estimation were compared: the classical *EKF* and our proposed *WLS* approach. Under normal conditions, *i.e.* no wheel slippage and accurate odometry, the two approaches present similar results. However, the performance of the *WLS* algorithm was less affected by simulated non-systematic odometric errors than the *EKF*. It indicates, at our point of view, that the *WLS* is more robust than the *EKF*. Moreover, the small computing time of the *WLS*, even for a large number of feature correspondences, is an important criterion for real-time localization. An other important aspect of the *WLS* is that robustness mechanisms well developed in robust statistics [17] may be directly adapted for mobile robot localization.

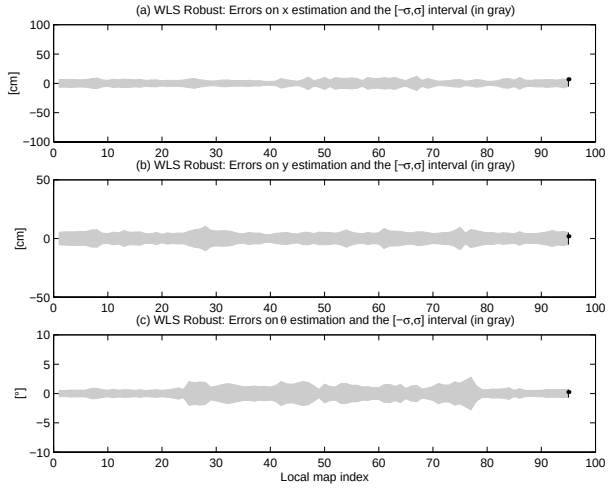
The authors believe that the same architecture can be used as a subset of a simultaneous localization and mapping (*SLAM*) system. In such an application, the time-critical phase is the pose estimation. After



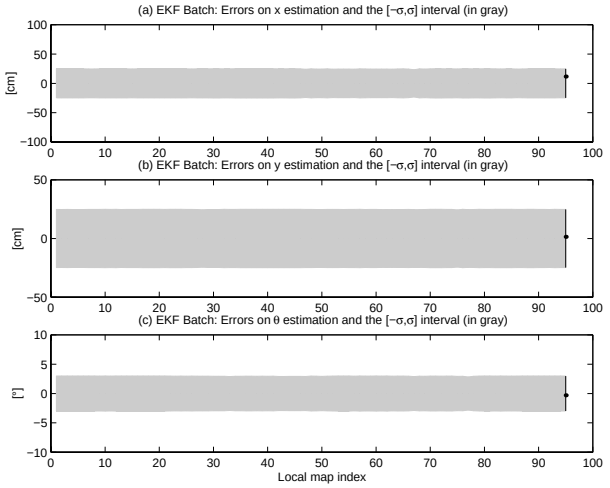
(a)



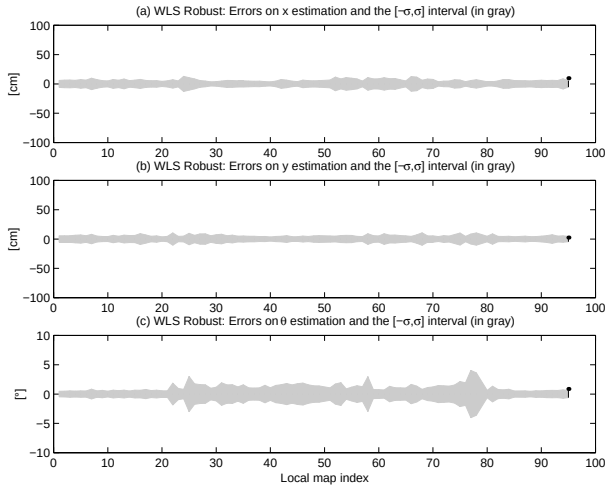
(b)



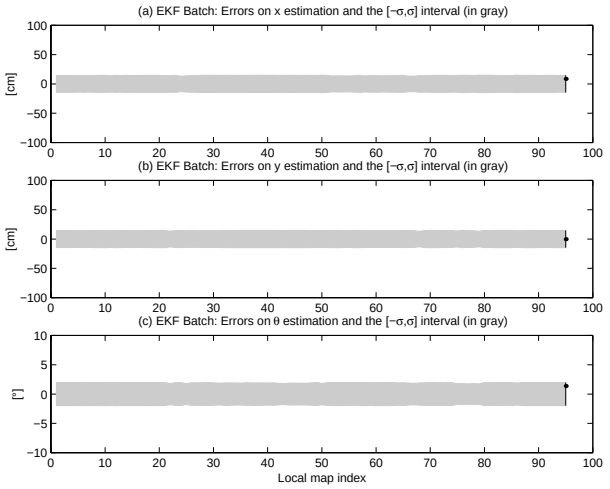
(c)



(d)



(e)



(f)

Fig. 4. Results for experiment A: (a) robust WLS and (b) EKF; Results for experiment B: (c) robust WLS and (d) EKF; Results for experiment C: (e) robust WLS and (f) EKF.

pose estimation, map building would be performed using the classical *EKF* for global map update using an augmented state vector which incorporates the consistent pose estimate and the global map features [18]. The use of the consistent pose estimate reduces the linearization effects of the measurement function on map consistency in *SLAM*.

Acknowledgement

The authors thank Capes, Brasília - Brazil, for the award of research during the course of this work.

References

1. J.J. Leonard and H.F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–382, June 1991.
2. J. Neira, J.D. Tardós, J. Horn, and G. Schmidt. Fusing range and intensity images for mobile robot localization. *IEEE Transactions on Robotics and Automation*, 15(1):76–84, February 1999.
3. J.-S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. In *International Conference on Intelligent Robots and Systems*, 1998.
4. S. Thrun, D. Fox, and W. Burgard. Probabilistic mapping of an environment by a mobile robot. In *IEEE International Conference on Robotics and Automation*, 1998.
5. J.A. Castellanos, J.M. Martinez, J. Neira, and J.D. Tardós. Simultaneous map building and localization for mobile robots: A multisensor fusion approach. In *IEEE International Conference on Robotics and Automation*, pages 1244–1249, 1998.
6. W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114:3–55, 2000.
7. G.A. Borges, M.-J. Aldon, and T. Gil. An optimal pose estimator for map-based mobile robot dynamic localization: Experimental comparison with the EKF. In *IEEE International Conference on Robotics and Automation*, 2001.
8. A.H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.
9. Y. Bar-Shalom and X.-R. Li. *Estimation and Tracking: Principles, Techniques and Software*. Norwood: Artech House, 1987.
10. Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2D range scans. *Journal on Intelligent and Robotic Systems*, 18(3):249–275, 1997.
11. J.-S. Gutmann and C. Schlegel. Amos: Comparison of scan matching approaches for self-localization in indoor environments. In *1st Euromicro Workshop on Advanced Mobile Robots*, 1996.
12. G.A. Borges and M.-J. Aldon. Motion estimation by iterative 2-D features matching in range images. In *IEEE International Conference on Robotics and Automation*, 2000.
13. G. A. Borges and M.-J. Aldon. A split-and-merge segmentation algorithm for line extraction in 2-D range images. In 15^{th} *International Conference on Pattern Recognition*, 2000.
14. M.A. Abidi and R.C. Gonzalez, editors. *Data fusion in Robotics and Machine Intelligence*. Academic Press, 1992.
15. R.M. Haralick. Propagating covariances in computer vision. In *International Conference on Pattern Recognition*, pages 493–498, 1994.
16. W.H. Press, S.A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerial Recipes in C: The Art of Scientific Computing*. Cambridge University Press, second edition, 1992.
17. F.R. Hampel, E.M. Ronchetti, P.J. Rousseeuw, and W.A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley & Sons, first edition, 1986.
18. Gamini Dissanayake. On the estimation theoretic solution to simultaneous localization and map building (SLAM). In *Workshop W4: Mobile Robot Navigation and Mapping (ICRA 2000)*, April 2000.

This article was processed using
the TEX macro package with SIRS2001 style