9 Map Building and SLAM Algorithms

José A. Castellanos, José Neira, and Juan D. Tardós

Contents

9.1 Introduction

The concept of autonomy of mobile robots encompasses many areas of knowledge, methods, and ultimately algorithms designed for trajectory control, obstacle avoidance, localization, map building, and so forth. Practically, the success of a path planning and navigation mission of an autonomous vehicle depends on the availability of both a sufficiently reliable estimation of the vehicle location and an accurate representation of the navigation area.

Schematically, the problem of map building consists of the following steps: (1) Sensing the environment of the vehicle at time k using onboard sensors (e.g., laser scanner, vision, or sonar); (2) Representation of sensor data (e.g., feature-based or raw-data-based approaches); (3) Integration of the recently perceived observations at time k with the previously learned structure of the environment estimated at time $k - 1$.

The simplest approach to map building relies on the vehicle location estimates provided by dead-reckoning. However, as reported in the literature [1], this approach is unreliable for long-term missions due to the time-increasing drift of those estimates (Figure 9.1a). Consequently, a coupling arises between the map building problem and the improvement of dead-reckoning location estimates arises (Figure 9.1b). Different approaches to the so-called *simultaneous localization and mapping* (SLAM) problem have populated the robotics literature during the last decade.

The most popular approach to SLAM dates back to the seminal work of Smith et al. [2] where the idea of representing the structure of the navigation area in a discrete-time state-space framework was originally presented. They introduced the concept of *stochastic map* and developed a rigorous solution to the SLAM problem using the extended Kalman filter (EKF) perspective. Many successful implementations of this approach have been reported in indoor [1], outdoor [3], underwater [4], and air-borne [5] applications.

The EKF-based approach to SLAM is characterized by the existence of a discrete-time augmented state vector, composed of the location of the vehicle and the location of the map elements, recursively estimated from the available sensor observations gathered at time k, and a model of the vehicle motion, between time steps $k-1$ and k. Within, this framework, uncertainty in represented by probability density functions (pdfs) associated with the state vector, the motion model, and the sensor observations. It is assumed that recursive propagation of the mean and the covariance of those pdfs conveniently approximates the optimal solution of this estimation problem.

The time and memory requirements of the basic EKF–SLAM approach result from the cost of maintaining the full covariance matrix, which is $O(n^2)$ where n is the number of features in the map. Many recent efforts have concentrated on reducing the computational complexity of SLAM in large environments. Several current methods address the computational complexity problem by working on a limited region of the map. Postponement [6] and the Compressed Filter [3] significantly reduce the computational cost without sacrificing precision, although they require an $O(n^2)$ step on the total number of landmarks to obtain the full map. The Split Covariance Intersection

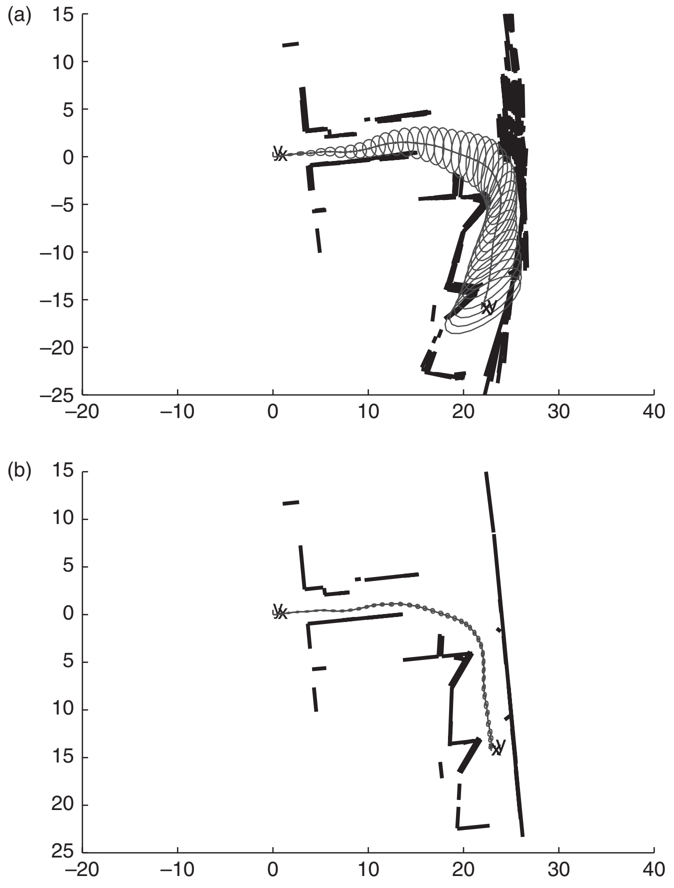


FIGURE 9.1 The need for SLAM: (a) odometric readings and segmented laser walls for 40 m of the trajectory of a vehicle at the Ada Byron building of our campus; (b) map and trajectory resulting from the SLAM algorithm using the same data (95% error ellipses are drawn)

method [7] limits the computational burden but sacrifices precision: it obtains a conservative estimate. The Sparse Extended Information Filter [8] is able to obtain an approximate map in constant time per step, except during loop closing. All cited methods work on a single absolute map representation, and confront divergence due to nonlinearities as uncertainty increases when mapping large areas [9]. In contrast, Local Map Joining [10] and the Constrained Local Submap Filter [11], propose to build stochastic maps relative to a local reference, guaranteed to be statistically independent. By limiting the size of the local map, this operation is constant time per step. Local maps are joined periodically into a global absolute map, in a $O(n^2)$ step. Given that most of the updates are carried out on a local map, these techniques also reduce the harmful effects of linearization. To avoid the $O(n^2)$ step, the Constrained Relative Submap Filter [12] proposes to maintain the independent local map structure. Each map contains links to other neighboring maps, forming a tree structure (where loops cannot be represented). In Atlas [13], Network Coupled Feature Maps [14], and Constant Time SLAM [15] the links between local maps form an adjacency graph. These techniques do not impose loop consistency in the graph, sacrificing the optimality of the resulting global map. Hierarchical SLAM [16] proposes a linear time technique to impose loop consistency, obtaining a close to optimal global map. The FastSLAM technique [17] uses particle filters to estimate the vehicle trajectory and each one has an associated set of independent EKF to estimate the location of each feature in the map. This partition of SLAM into a localization and a mapping problem, allows to obtain a computational complexity $O(\log(n))$ with the number of features in the map. However, its complexity is linear with the number of particles used. The scaling of the number of particles needed with the size and complexity of the environment remains unclear. In particular, closing loops causes dramatic particle extinctions that map result in optimistic (i.e., inconsistent) uncertainty estimations.

Another class of SLAM techniques is based on estimating sequences of robot poses by minimizing the discrepancy between overlapping laser scans. The map representation is the set of robot poses and the corresponding set of laser scans. The work in Reference 18 uses scan matching between close robot poses and global correlation to detect loops. The poses along the loop are estimated using Consistent Pose Estimation [19], whose time complexity is $O(n^3)$ on the number of robot poses, making the method unsuitable for real time execution in large environments. More recently, a similar approach to build consistent maps with many cycles has been proposed in Reference 20. This method obtains correspondences between vehicle poses using the Iterative Closest Point algorithm: Using a quadratic penalty function, correspondences are incorporated into an optimization algorithm that recomputes the whole trajectory. This process is iterated until convergence. Neither computing time nor computational complexity are reported. There are two fundamental limitations in this class of techniques, compared to EKF-based SLAM. First, there is no explicit representation of the uncertainty in the estimated robot poses and the resulting map. As a consequence, their convergence and consistency properties remain unknown. Second, they largely rely on the high precision and density of data provided by laser scanners. They seem hard to extend to sensors that give more imprecise, sparse, or partial information such as sonar of monocular vision.

This chapter describes the basic algorithm to deal with the SLAM problem from the above mentioned EKF-based perspective. We describe techniques that successful SLAM schemes must incorporate (1) Data association techniques, to relate sensor measurements with features already in the map, as well as to decide those that are spurious or correspond to environment features not previously observed, and (2) Loop closing and relocation techniques, that allow determination of the vehicle location and correct the map when the vehicle uncertainty increases significantly during exploration, or when there is no prior information on the vehicle location. Finally, we point out the main open problem of the current state-of-art SLAM approaches: mapping large-scale areas. Relevant shortcomings of this problem are, on the one hand, the computational burden, which limits the applicability of the EKF-based SLAM in large-scale real time applications and, on the other hand, the use of linearized solutions which jeopardizes the consistency of the estimation process. We point out promising directions of research using nonlinear estimation techniques, and mapping schemes for multivehicle SLAM.

9.2 SLAM Using the Extended Kalman Filter

In feature-based approaches to SLAM, the environment is modeled as a set of geometric features, such as straight line segments corresponding to doors or window frames, planes corresponding to walls, or distinguishable points in outdoor environments. The process of segmentation of raw sensor data to obtain feature parameters depends on the sensor and the feature type. In indoor environments, laser readings can be used to obtain straight wall segments [21, 22], or in outdoor environments to obtain two-dimensional (2D) points corresponding to trees and street lamps [3]. Sonar measurement environments can be segmented into corners and walls [10]. Monocular images can provide information about vertical lines [23] or interest points [24]. Even measurements from different sensors can be fused to obtain feature information [25].

In the standard EKF-based approach, the environment information related to a set of elements $\{B, R, F_1, \dots, F_n\}$ is represented by a map $\mathcal{M}^B = (\hat{\mathbf{x}}^B, \mathbf{P}^B)$, where $\mathbf{x}^B$ is a stochastic state vector with estimated mean $\hat{\mathbf{x}}^B$ and estimated error covariance $\mathbf{P}^B$:

$$\begin{aligned}\hat{\mathbf{x}}^B &= E[\mathbf{x}^B] = \begin{bmatrix} \hat{\mathbf{x}}^B_R \\ \vdots \\ \hat{\mathbf{x}}^B_{F_n} \end{bmatrix} \\ \mathbf{P}^B &= E[(\mathbf{x}^B - \hat{\mathbf{x}}^B)(\mathbf{x}^B - \hat{\mathbf{x}}^B)^{\mathrm{T}}] = \begin{bmatrix} \mathbf{P}^B_R & \cdots & \mathbf{P}^B_{RF_n} \\ \vdots & \ddots & \vdots \\ \mathbf{P}^B_{F_nR} & \cdots & \mathbf{P}^B_{F_n} \end{bmatrix}\end{aligned} \tag{9.1}$$

ALGORITHM 9.1
EKF–SLAM

```
$\mathbf{x}_0^B = \mathbf{0}$; $\mathbf{P}_0^B = \mathbf{0}$ {*Map initialization*}
[$\mathbf{z}_0$, $\mathbf{R}_0$] = get_measurements
[$\mathbf{x}_0^B$, $\mathbf{P}_0^B$] = add_new_features($\mathbf{x}_0^B$, $\mathbf{P}_0^B$, $\mathbf{z}_0$, $\mathbf{R}_0$)
for $k = 1$ to steps do
    [$\mathbf{x}_{R_k}^{R_{k-1}}$, $\mathbf{Q}_k$] = get_odometry
    [$\mathbf{x}_{k|k-1}^B$,$\mathbf{P}_{k|k-1}^B$] = compute_motion($\mathbf{x}_{k-1}^B$, $\mathbf{P}_{k-1}^B$,$\mathbf{x}_{R_k}^{R_{k-1}}$,$\mathbf{Q}_k$) {*EKF predict.*}
    [$\mathbf{z}_k$, $\mathbf{R}_k$] = get_measurements
    $\mathcal{H}_k$ = data_association($\mathbf{x}_{k|k-1}^B$, $\mathbf{P}_{k|k-1}^B$, $\mathbf{z}_k$, $\mathbf{R}_k$)
    [$\mathbf{x}_k^B$, $\mathbf{P}_k^B$] = update_map($\mathbf{x}_{k|k-1}^B$, $\mathbf{P}_{k|k-1}^B$, $\mathbf{z}_k$, $\mathbf{R}_k$, $\mathcal{H}_k$) {*EKF update*}
    [$\mathbf{x}_k^B$, $\mathbf{P}_k^B$] = add_new_features($\mathbf{x}_k^B$, $\mathbf{P}_k^B$, $\mathbf{z}_k$, $\mathbf{R}_k$, $\mathcal{H}_k$)
end for
```

Vector $\hat{\mathbf{x}}^B$ contains the estimated location of the vehicle R and the environment features $F_1 \dots F_n$, all with respect to a base reference B. In the case of the vehicle, its location vector $\hat{\mathbf{x}}_R^B = (x, y, \phi)^{\mathrm{T}}$ describes the transformation from B to R. In the case of an environment feature j, the parameters that compose its location vector $\hat{\mathbf{x}}_{F_j}^B$ depend on the feature type, for example, $\hat{\mathbf{x}}_{F_j}^B = (x_j, y_j)^{\mathrm{T}}$ for point features. The diagonal elements of the matrix $\mathbf{P}^B$ represent the estimated error covariance of the different features of the state vector and that of the vehicle location; its off-diagonal elements represent the cross-covariance matrices between the estimated locations of the corresponding features.

Recursive estimation of the first two moments of the probability density function of $\mathbf{x}^B$ is performed following Algorithm 9.1. There, the map is initialized using the current vehicle location as base reference, and thus with perfect knowledge of the vehicle location. Sensing and feature initialization is also performed before the first vehicle motion, to maximize the precision of the resulting map. Prediction of the vehicle motion using odometry and update of the map using onboard sensor measurements are then iteratively carried out.

9.2.1 Initialization

In the creation of a new stochastic map at step 0, a base reference B must be selected. It is common practice to build a map relative to a fixed base reference different from the initial vehicle location. This normally requires the assignment of an initial level of uncertainty to the estimated vehicle location. In the theoretical linear case [26], the vehicle uncertainty should always remain above this initial level. In practice, due to linearizations, when a nonzero initial uncertainty is used, the estimated vehicle uncertainty rapidly drops below its initial value, making the estimation inconsistent after very few EKF update steps [9].

A good alternative is to use, as base reference, the current vehicle location, that is. $B = R_0$, and thus we initialize the map with perfect knowledge of the vehicle location:

$$\hat{\mathbf{x}}_0^B = \hat{\mathbf{x}}_{R_0}^B = \mathbf{0}; \quad \mathbf{P}_0^B = \mathbf{P}_{R_0}^B = \mathbf{0} \tag{9.2}$$

If at any moment there is a need to compute the location of the vehicle or the map features with respect to any other reference, the appropriate transformations can be applied (see Appendix). At any time, the map can also be transformed to use a feature as base reference, again using the appropriate transformations [10].

9.2.2 Vehicle Motion: The EKF Prediction Step

When the vehicle moves from position $k-1$ to position k, its motion is estimated by odometry:

$$\mathbf{x}_{R_k}^{R_{k-1}} = \hat{\mathbf{x}}_{R_k}^{R_{k-1}} + \mathbf{v}_k \tag{9.3}$$

where $\hat{\mathbf{x}}_{R_k}^{R_{k-1}}$ is the estimated relative transformation between positions $k-1$ and k, and $\mathbf{v}_k$ (process noise [27]) is assumed to be additive, zero-mean, and white, with covariance $\mathbf{Q}_k$.

Thus, given a map $\mathcal{M}_{k-1}^B = (\hat{\mathbf{x}}_{k-1}^B, \mathbf{P}_{k-1}^B)$ at step $k-1$, the predicted map $\mathcal{M}_{k|k-1}^B$ at step k after the vehicle motion is obtained as follows:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k-1}^B &= \begin{bmatrix} \hat{\mathbf{x}}_{R_{k-1}}^B \oplus \hat{\mathbf{x}}_{R_k}^{R_{k-1}} \\ \hat{\mathbf{x}}_{F_{1,k-1}}^B \\ \vdots \\ \hat{\mathbf{x}}_{F_{m,k-1}}^B \end{bmatrix} \\ \mathbf{P}_{k|k-1}^B &\simeq \mathbf{F}_k \mathbf{P}_{k-1}^B \mathbf{F}_k^{\mathrm{T}} + \mathbf{G}_k \mathbf{Q}_k \mathbf{G}_k^{\mathrm{T}} \end{aligned} \tag{9.4}$$

where $\oplus$ represents the composition of transformations (see Appendix), and:

$$\mathbf{F}_k = \left.\frac{\partial \mathbf{x}^B_{k|k-1}}{\partial \mathbf{x}^B_{k-1}}\right|_{(\hat{\mathbf{x}}^B_{k-1},\, \hat{\mathbf{x}}^{R_{k-1}}_{R_k})} = \begin{bmatrix} \mathbf{J}_{1\oplus}\left\{\hat{\mathbf{x}}^B_{R_{k-1}},\, \hat{\mathbf{x}}^{R_{k-1}}_{R_k}\right\} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & & \vdots \\ \vdots & & \ddots & \\ \mathbf{0} & & \cdots & \mathbf{I} \end{bmatrix}$$

$$\mathbf{G}_k = \left.\frac{\partial \mathbf{x}^B_{k|k-1}}{\partial \mathbf{x}^{R_{k-1}}_{R_k}}\right|_{(\hat{\mathbf{x}}^B_{k-1},\, \hat{\mathbf{x}}^{R_{k-1}}_{R_k})} = \begin{bmatrix} \mathbf{J}_{2\oplus}\left\{\hat{\mathbf{x}}^B_{R_{k-1}},\, \hat{\mathbf{x}}^{R_{k-1}}_{R_k}\right\} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix}$$

where $\mathbf{J}_{1\oplus}$ and $\mathbf{J}_{2\oplus}$ are the Jacobians of transformation composition (see Appendix).

9.2.3 Data Association

At step k, an onboard sensor obtains a set of measurements $\mathbf{z}_{k,i}$ of m environment features E_i $(i = 1, \ldots, m)$. Data association consists in determining the origin of each measurement, in terms of the map features $F_j, j = 1, \ldots, n$. The result is a hypothesis:

$$\mathcal{H}_k = [\mathtt{j}_1 \mathtt{j}_2 \cdots \mathtt{j}_\mathtt{m}]$$

associating each measurement $\mathbf{z}_{k,i}$ with its corresponding map feature F_{j_i} ($\mathtt{j}_i = 0$) indicates that $\mathbf{z}_{k,i}$ does not come from any feature in the map). The core tools of data association are a prediction of the measurement that each feature would generate, and a measure of the discrepancy between a predicted measurement and an actual sensor measurement.

The measurement of feature F_j can be predicted using a nonlinear measurement function $\mathbf{h}_{k,j}$ of the vehicle and feature location, both contained in the map state vector $\mathbf{x}^B_{k|k-1}$. If observation $\mathbf{z}_{k,i}$ comes from feature F_j, the following relation must hold:

$$\mathbf{z}_{k,i} = \mathbf{h}_{k,j}(\mathbf{x}^B_{k|k-1}) + \mathbf{w}_{k,i} \tag{9.5}$$

where the measurement noise $\mathbf{w}_{k,i}$, with covariance $\mathbf{R}_{k,i}$, is assumed to be additive, zero-mean, white, and independent of the process noise $\mathbf{v}_k$. Linearization of Equation 9.5 around the current estimate yields:

$$\mathbf{h}_{k,j}(\mathbf{x}^B_{k|k-1}) \simeq \mathbf{h}_{k,j}(\hat{\mathbf{x}}^B_{k|k-1}) + \mathbf{H}_{k,j}(\mathbf{x}^B_k - \hat{\mathbf{x}}^B_{k|k-1}) \tag{9.6}$$

with

$$\mathbf{H}_{k,j} = \left.\frac{\partial \mathbf{h}_{k,j}}{\partial \mathbf{x}^B_{k|k-1}}\right|_{(\hat{\mathbf{x}}^B_{k|k-1})} \tag{9.7}$$

The discrepancy between the observation i and the predicted observation of map feature j is measured by the innovation term $\nu_{k,ij}$, whose value and covariance are:

$$\begin{aligned} \nu_{k,ij} &= \mathbf{z}_{k,i} - \mathbf{h}_{k,j}(\hat{\mathbf{x}}^B_{k|k-1}) \\ \mathbf{S}_{k,ij} &= \mathbf{H}_{k,j}\mathbf{P}^B_k\mathbf{H}^{\mathrm{T}}_{k,j} + \mathbf{R}_{k,i} \end{aligned} \tag{9.8}$$

The measurement can be considered corresponding to the feature if the Mahalanobis distance $D^2_{k,ij}$ [28] satisfies:

$$D^2_{k,ij} = \nu^{\mathrm{T}}_{k,ij}\mathbf{S}^{-1}_{k,ij}\nu_{k,ij} < \chi^2_{d,1-\alpha} \tag{9.9}$$

where $d = \dim(\mathbf{h}_{k,j})$ and $1 - \alpha$ is the desired confidence level, usually 95%. This test, denominated individual compatibility (IC), applied to the predicted state, can be used to determine the subset of map features that are compatible with a measurement, and is the basis for some of the most popular data association algorithms discussed later in this chapter.

An often overlooked fact, that will be discussed in more detail in Section 9.3, is that *all* measurements should be jointly compatible with their corresponding features. In order to establish the consistency of a hypothesis $\mathcal{H}_k$, measurements can be jointly predicted using function $\mathbf{h}_{\mathcal{H}_k}$:

$$\mathbf{h}_{\mathcal{H}_k}(\mathbf{x}^B_{k|k-1}) = \begin{bmatrix} \mathbf{h}_{j_1}(\mathbf{x}^B_{k|k-1}) \\ \vdots \\ \mathbf{h}_{j_m}(\mathbf{x}^B_{k|k-1}) \end{bmatrix} \tag{9.10}$$

which can also be linearized around the current estimate to yield:

$$\mathbf{h}_{\mathcal{H}_k}(\mathbf{x}^B_{k|k-1}) \simeq \mathbf{h}_{\mathcal{H}_k}(\hat{\mathbf{x}}^B_{k|k-1}) + \mathbf{H}_{\mathcal{H}_k}(\mathbf{x}^B_k - \hat{\mathbf{x}}^B_{k|k-1}); \mathbf{H}_{\mathcal{H}_k} = \begin{bmatrix} \mathbf{H}_{j_1} \\ \vdots \\ \mathbf{H}_{j_m} \end{bmatrix} \tag{9.11}$$

The joint innovation and its covariance are:

$$\begin{aligned} \nu_{\mathcal{H}_k} &= \mathbf{z}_k - \mathbf{h}_{\mathcal{H}_k}(\hat{\mathbf{x}}^B_{k|k-1}) \\ \mathbf{S}_{\mathcal{H}_k} &= \mathbf{H}_{\mathcal{H}_k}\mathbf{P}^B_k\mathbf{H}^{\mathrm{T}}_{\mathcal{H}_k} + \mathbf{R}_{\mathcal{H}_k} \end{aligned} \tag{9.12}$$

Measurements $\mathbf{z}_k$ can be considered compatible with their corresponding features according to $\mathcal{H}_k$ if the Mahalanobis distance satisfies:

$$D^2_{\mathcal{H}_k} = \nu^{\mathrm{T}}_{\mathcal{H}_k}\mathbf{S}^{-1}_{\mathcal{H}_k}\nu_{\mathcal{H}_k} < \chi^2_{d,1-\alpha} \tag{9.13}$$

where now $d = \dim(\mathbf{h}_{\mathcal{H}_k})$. This consistency test is denominated joint compatibility (JC).

9.2.4 Map Update: The EKF Estimation Step

Once correspondences for measurements $\mathbf{z}_k$ have been decided, they are used to improve the estimation of the stochastic state vector by using the standard EKF update equations as follows:

$$\hat{\mathbf{x}}^B_k = \hat{\mathbf{x}}^B_{k|k-1} + \mathbf{K}_{\mathcal{H}_k}\nu_{\mathcal{H}_k} \tag{9.14}$$

where the filter gain $\mathbf{K}_{\mathcal{H}_k}$ is obtained from:

$$\mathbf{K}_{\mathcal{H}_k} = \mathbf{P}^B_{k|k-1}\mathbf{H}^{\mathrm{T}}_{\mathcal{H}_k}\mathbf{S}^{-1}_{\mathcal{H}_k} \tag{9.15}$$

Finally, the estimated error covariance of the state vector is:

$$\begin{aligned} \mathbf{P}^B_k &= (\mathbf{I} - \mathbf{K}_{\mathcal{H}_k}\mathbf{H}_{\mathcal{H}_k})\mathbf{P}^B_{k|k-1} \\ &= (\mathbf{I} - \mathbf{K}_{\mathcal{H}_k}\mathbf{H}_{\mathcal{H}_k})\mathbf{P}^B_{k|k-1}(\mathbf{I} - \mathbf{K}_{\mathcal{H}_k}\mathbf{H}_{\mathcal{H}_k})^{\mathrm{T}} + \mathbf{K}_{\mathcal{H}_k}\mathbf{R}_{\mathcal{H}_k}\mathbf{K}^{\mathrm{T}}_{\mathcal{H}_k} \end{aligned} \tag{9.16}$$

9.2.5 Adding Newly Observed Features

Measurements for which correspondences in the map cannot be found by data association can be directly added to the current stochastic state vector as new features by using the relative transformation between the vehicle R_k and the observed feature E. Therefore, updating of $\mathbf{x}_k^B$ takes place as follows:

$$\mathbf{x}_k^B = \begin{bmatrix} \mathbf{x}_{R_k}^B \\ \vdots \\ \mathbf{x}_{F_{n,k}}^B \end{bmatrix} \Rightarrow \mathbf{x}_{k+}^B = \begin{bmatrix} \mathbf{x}_{R_k}^B \\ \vdots \\ \mathbf{x}_{F_{n,k}}^B \\ \mathbf{x}_{E_k}^B \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{R_k}^B \\ \vdots \\ \mathbf{x}_{F_{n,k}}^B \\ \mathbf{x}_{R_k}^B \oplus \mathbf{x}_E^{R_k} \end{bmatrix} \tag{9.17}$$

Additionally, the updated covariance matrix $\mathbf{P}_{k+}^B$ is computed using the linearization of Equation 9.17.

9.2.6 Consistency of EKF–SLAM

A state estimator is called *consistent* if its state estimation error $\mathbf{x}_k^B - \hat{\mathbf{x}}_k^B$ satisfies [29]:

$$\begin{aligned} E[\mathbf{x}_k^B - \hat{\mathbf{x}}_k^B] &= \mathbf{0} \\ E[(\mathbf{x}_k^B - \hat{\mathbf{x}}_k^B)\,(\mathbf{x}_k^B - \hat{\mathbf{x}}_k^B)^{\mathrm{T}}] &\leq \mathbf{P}_k^B \end{aligned} \tag{9.18}$$

This means that the estimator is *unbiased* and that the actual mean square error matches the filter-calculated covariances. Given that SLAM is a nonlinear problem, consistency checking is of paramount importance. When the ground truth solution for the state variables is available, a statistical test for filter consistency can be carried out on the normalized estimation error squared (NEES), defined as:

$$\text{NEES} = (\mathbf{x}_k^B - \hat{\mathbf{x}}_k^B)^{\mathrm{T}}\,(\mathbf{P}_k^B)^{-1}\,(\mathbf{x}_k^B - \hat{\mathbf{x}}_k^B) \tag{9.19}$$

Consistency is checked using a chi-squared test:

$$\text{NEES} \leq \chi^2_{d,1-\alpha} \tag{9.20}$$

where $d = \dim(\mathbf{x}_k^B)$ and $1 - \alpha$ is the desired confidence level. Since in most cases ground truth is not available, the consistency of the estimation is maintained by using only measurements that satisfy the innovation test of Equation 9.13. Because the innovation term depends on the data association hypothesis, this process becomes critical in maintaining a consistent estimation [9] of the environment map.

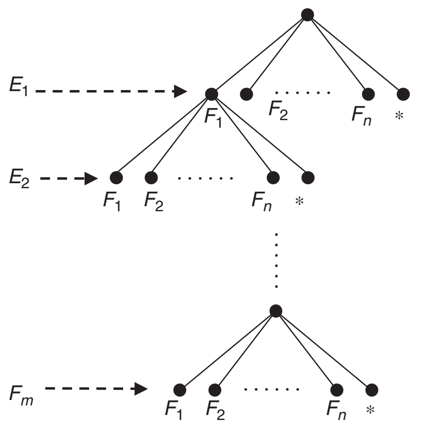


FIGURE 9.2 Interpretation tree of measurements $E_1, \dots, E_m$ in terms of map features $F_1, \dots, F_n$

9.3 Data Association in SLAM

Assume that a new set of m measurements $\mathbf{z} = \{\mathbf{z}_1, \dots, \mathbf{z}_m\}$ of the environment features $\{E_1, \dots, E_m\}$ have been obtained by a sensor mounted on the vehicle. As mentioned in Section 9.2, the goal of data association is to generate a hypothesis $\mathcal{H} = [\mathtt{j}_1\mathtt{j}_2 \cdots \mathtt{j}_m]$ associating each measurement E_i with its corresponding map feature $F_{j_i}(\mathtt{j}_i = 0)$ indicating that $\mathbf{z}_i$ does not correspond to any map feature). The space of measurement-feature correspondences can be represented as an *interpretation tree* of m levels [30] (see Figure 9.2); each node at level i, called an *i-interpretation*, provides an interpretation for the first i measurements. Each node has $n + 1$ branches, corresponding to each of the alternative interpretations for measurement E_i, including the possibility that the measurement be spurious and allowing map feature repetitions in the same hypothesis. Data association algorithms must select in some way one of the $(n+1)^m$ m-interpretations as the correct hypothesis, carrying out validations to determine the compatibility between sensor measurements and map features.

9.3.1 Individual Compatibility Nearest Neighbor

The simplest criterion to decide a pairing for a given measurement is the *nearest neighbor* (NN), which consists in choosing among the features that satisfy Individual Compatibility of Equation 9.9, the one with the smallest Mahalanobis distance. A popular data association algorithm, the Individual Compatibility Nearest Neighbor (ICNN, Algorithm 9.2), is based on this idea. It is frequently used given its conceptual simplicity and computational efficiency: it performs $m \cdot n$ compatibility tests, making it linear with the size of the map.

ALGORITHM 9.2
ICNN

```
ICNN ($E_{1\cdots m}, F_{1\cdots n}$)
for $i = 1$ to $m$ do {measurement $E_i$}
    $D^2_{\mathtt{min}} \leftarrow$ mahalanobis2 ($E_i$, $F_1$)
    nearest $\leftarrow 1$
    for $j = 2$ to $n$ do {feature $F_j$}
        $D^2_{ij} \leftarrow$ mahalanobis2 ($E_i$, $F_j$)
        if $D^2_{ij} < D^2_{\mathtt{min}}$ then
            nearest $\leftarrow j$
            $D^2_{\mathtt{min}} \leftarrow D^2_{ij}$
        end if
    end for
    if $D^2_{\mathtt{min}} \leq \chi^2_{d_i,1-\alpha}$ then
        $\mathcal{H}_i \leftarrow$ nearest
    else
        $\mathcal{H}_i \leftarrow 0$
    end if
end for
return $\mathcal{H}$
```

Individual compatibility considers *individual* compatibility between a measurement and a feature. However, individually compatible pairings are not guaranteed to be *jointly* compatible to form a consistent hypothesis. Thus, with ICNN there is a high risk of obtaining an inconsistent hypothesis and thus updating the state vector with a set of incompatible measurements, which will cause EKF to diverge. As vehicle error grows with respect to sensor error, the discriminant power of IC decreases: the probability that a feature may be compatible with an unrelated (or spurious) sensor measurement increases. ICNN is a greedy algorithm, and thus the decision to pair a measurement with its most compatible feature is never reconsidered. As a result, spurious pairings may be included in the hypothesis and integrated in the state estimation. This will lead to a reduction in the uncertainty computed by the EKF with no reduction in the actual error, that is, inconsistency.

9.3.2 Joint Compatibility

In order to limit the possibility of accepting a spurious pairing, reconsideration of the established pairings is necessary. The probability that a spurious

ALGORITHM 9.3
JCBB

```
Continuous_JCBB ($E_{1\cdots m}, F_{1\cdots n}$)
Best = [ ]
JCBB ( [ ], 1)
**return** Best

**procedure** JCBB ($\mathcal{H}$, $i$): {*find pairings for observation* $E_i$}
**if** $i > m$ **then** {*leaf node*?}
    **if** pairings($\mathcal{H}$) $>$ pairings(Best) **then**
        Best $\leftarrow \mathcal{H}$
    **end if**
**else**
    **for** $j = 1$ to $n$ **do**
        **if** individual_compatibility($i,j$) **and then** joint_compatibility($\mathcal{H}, i, j$)
        **then**
            JCBB([$\mathcal{H}$ j], i + 1) {*pairing* $(E_i, F_j)$ *accepted*}
        **end if**
    **end for**
    **if** pairings($\mathcal{H}$) $+ m - i >$ pairings(Best) **then** {*can do better*?}
        JCBB([$\mathcal{H}$ 0], $i + 1$) {*star node*, $E_i$ *not paired*}
    **end if**
**end if**
```

pairing is jointly compatible with all the other pairings of a given hypothesis decreases as the number of pairings in the hypothesis increases. The JC test can be used to establish the consistency of a hypothesis $\mathcal{H}_m$, using Equation 9.13. The JC test is the core of the joint compatibility branch and bound data association algorithm (JCBB, Algorithm 9.3), that traverses the interpretation tree in search for the hypothesis that includes the largest number of *jointly compatible* pairings. The quality of a node at level i, corresponding to a hypothesis $\mathcal{H}_i$, is defined as the number of non-null pairings that can be established from the node. In this way, nodes with quality lower than the best available hypothesis are not explored, bounding the search [30]. The NN rule using the Mahalanobis distance $D^2_{\mathcal{H}_i}$ is used as heuristic for *branching*, so that the nodes corresponding to hypotheses with a higher degree of JC are explored first. The size of both $\mathbf{h}_{\mathcal{H}_i}$ and $\mathbf{S}_{\mathcal{H}_i}$ increase with the size of hypothesis $\mathcal{H}_i$. This makes this test potentially expensive to apply (see References 31 and 32 for techniques for the efficient computation of the Mahalanobis distance).

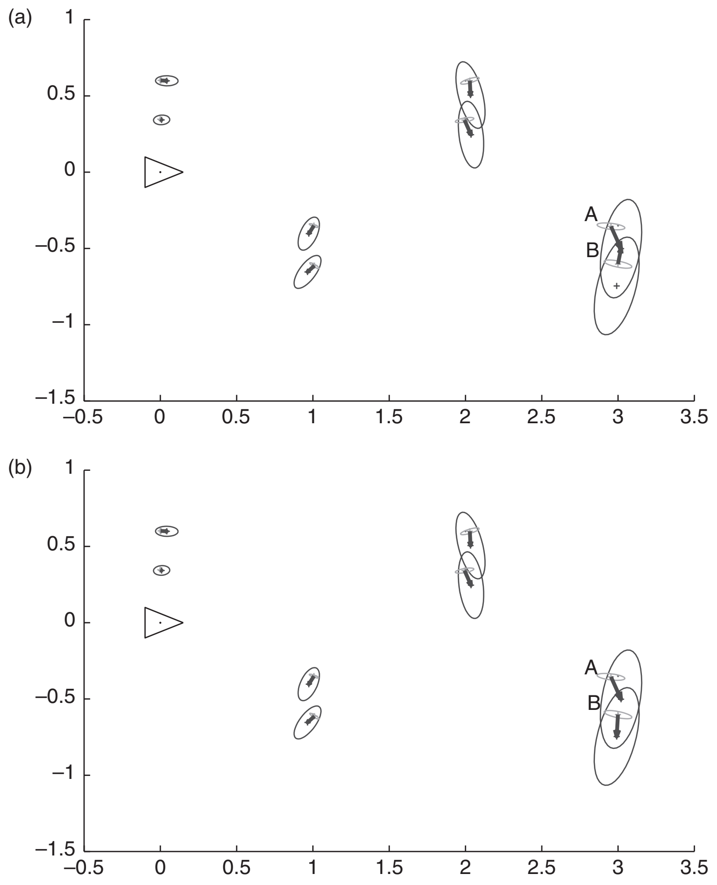


FIGURE 9.3 Predicted feature locations relative to vehicle (large ellipses), measurements (small ellipses), and associations (bold arrows). According to the ICNN algorithm observation B is incorrectly matched with the upper map point (a) and according to the JCBB algorithm (b) all the matches are correct

During continuous SLAM, data association problems may arise even in very simple scenarios. Consider an environment constituted by 2D points. If at a certain point the vehicle uncertainty is larger than the separation between the features, the predicted feature locations relative to the robot are cluttered, and the NN algorithm is prone to make an incorrect association as illustrated in Figure 9.3a where two measurements are erroneously paired with the same map feature. In these situations, the JCBB algorithm can determine the correct associations (Figure 9.3b), because through correlations it considers the relative location between the features, independent of vehicle error.

The robustness of JCBB is especially important in loop-closing operations (Figure 9.4). Due to the big odometry errors accumulated, simple data association algorithms would incorrectly match the signaled point with a point feature previously observed in the pillar. Accepting an incorrect matching will cause the EKF to diverge, obtaining an inconsistent map. The JC algorithm takes into account the relative location between the point and the segment and has no problem in finding the right associations. The result is a consistent and more precise global map.

Joint Compatibility is a highly restrictive criterion, that limits the combinatorial explosion of the search. The computational complexity does not suffer with the increase in vehicle error because the JC of a certain number of measurements fundamentally depends on their *relative error* (which depends on sensor and map precision), more than on their *absolute error* (which depends on robot error). The JC test is based on the linearization of the relation between the measurements and the state (Equation 9.6). JCBB will remain robust to robot error as long as the linear approximation is reasonable. Thus, the adequacy of using JCBB is determined by the robot orientation error (in practice, we have found the limit to be around 30°). Even if the vehicle motion is unknown (no odometry is available), as long as it is bounded by within this limit, JCBB can perform robustly. In these cases, the predicted vehicle motion can be set to zero ($\hat{\mathbf{x}}_{R_k}^{R_{k-1}} = \mathbf{0}$, Figure 9.5a), with $\mathbf{Q}_k$ sufficiently large to include the largest possible displacement. The algorithm will obtain the associations, and during the estimation stage of the EKF the vehicle motion will be determined and the environment structure can be recovered (Figure 9.5b).

9.3.3 Relocation

Consider now the data association problem known as vehicle relocation, first-location, global localization, or "kidnapped" robot problem, which can be stated as follows: *given a vehicle in an unknown location, and a map of the environment, use a set of measurements taken by onboard sensors to determine the vehicle location within the map.* In SLAM, solving this problem is essential to be able to restart the robot in a previously learned environment, to recover from localization errors, or to safely close big loops.

When there is no vehicle location estimation, simple *location independent* geometric constraints can be used to limit the complexity of searching the correspondence space [30]. Given a pairing $p_{ij} = (E_i, F_j)$, the unary geometric constraints that may be used to validate the pairing include length for segments, angle for corners, or radius for circular features. Given two pairings $p_{ij} = (E_i, F_j)$ and $p_{kl} = (E_k, F_l)$, a binary geometric constraint is a geometric relation between measurements E_i and E_k that must also be satisfied between their corresponding map features F_j and F_l (e.g., distance between two points,

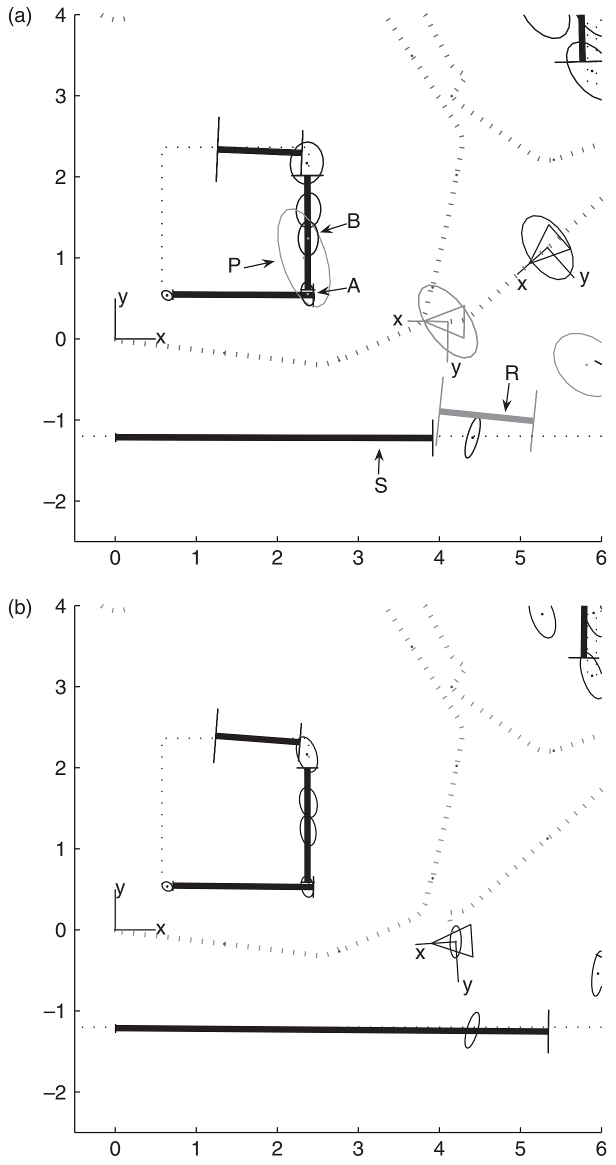


FIGURE 9.4 A loop-closing situation. (a) Before loop closing, potential matches have been found for measurements signaled with an arrow: measurement R is compatible only with feature S, but measurement P is compatible with both features A and B. The NN rule would incorrectly match P with A. (b) The JCBB algorithm has correctly matched both observations with the corner (P with A) and the lower wall (R with B), and the map has been updated

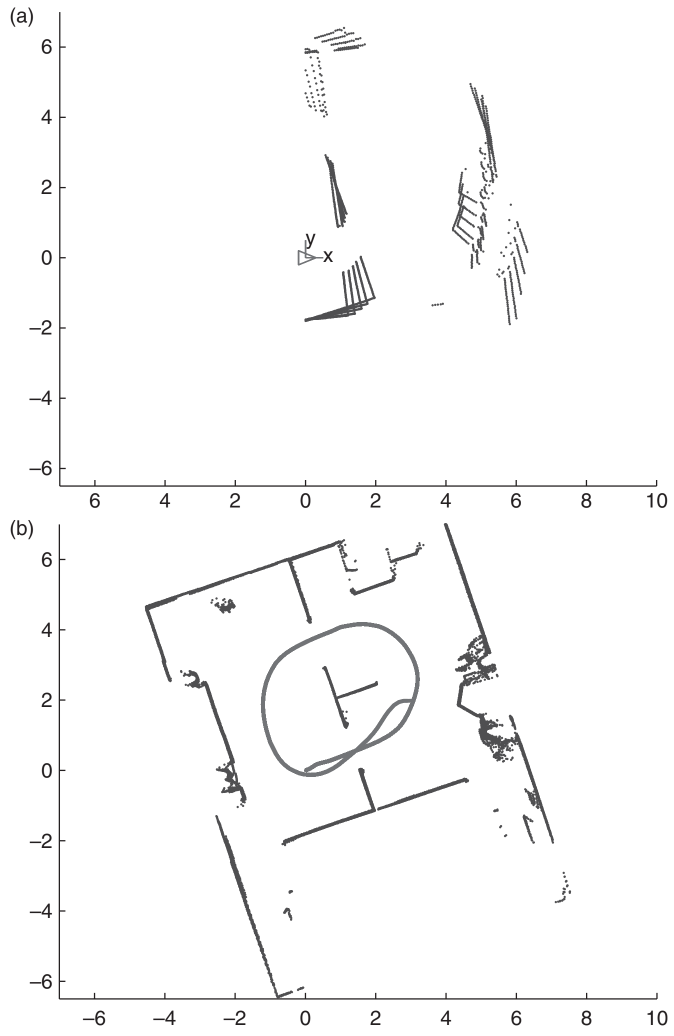


FIGURE 9.5 Data association using JCBB without odometry: (a) laser data in the absolute reference with null vehicle motion; (b) map and vehicle trajectory resulting from the SLAM algorithm

angle between two segments). For stochastic geometric constraint validation in SLAM, see Reference 33.

Grimson [30] proposed a branch and bound algorithm for model-based geometric object recognition that uses unary and binary geometric constraints. A closely related technique also used in object recognition consists in building a compatibility graph whose nodes are unary compatible matchings and whose arcs represent pairs of binary compatible matchings. Finding the largest hypothesis consistent with unary and binary constraints is equivalent to finding the maximum clique in the compatibility graph (see Reference 30 for a discussion and references). This idea has been applied recently by Bailey et al. [34] to the problem of robot relocation with an a priori map.

Branch and bound algorithms are forced to traverse the whole correspondence space until a good bound is found. In the SLAM relocation problem, when the vehicle is not within the mapped area, a good bound is never found. Since the correspondence space is exponential with the number of measurements, in this worst case the execution times of branch and bound algorithms are very long. To overcome this limitation, the data association process can be done using random sampling (RS) instead of by a full traversal of the interpretation tree. The RS algorithm that we use (Algorithm 9.4) is an adaptation of the RANSAC Algorithm [35] for the relocation problem. The fundamental idea is to randomly select p out of the m measurements to try to generate vehicle localization hypotheses using geometric constraints, and verify them with all m measurements using joint compatibility. If P_g is the probability that a randomly selected measurement corresponds to a mapped feature (not spurious) and P_{fail} is the acceptable probability of not finding a good solution when it exists, the required number of tries is:

$$t = \left\lceil \frac{\log P_{\text{fail}}}{\log(1 - P_g{}^p)} \right\rceil \tag{9.21}$$

Hypothesis generation-verification schemes such as this one perform better because feature location is a tighter consistency criterion than geometric constraints, and thus branch pruning is more effective. The potential drawback of this approach is that hypothesis verification is *location dependent*, and thus the constraints to be used for validation cannot be precomputed. To limit the amount of location dependent constraints to apply, verification can take place when a hypothesis contains at least *three* consistent pairings. Choosing $P_{\text{fail}} = 0.05$ and considering a priori that only half of the measurements are present in the map $P_g = 0.5$, the maximum number of tries is $t = 23$. If you can consider that at least 90% of the measurements correspond to a map feature, the number of required tries is only three. The RS algorithm randomly permutes the measurements and performs hypothesis generation considering the first three measurements not spurious (without star branch). The number of tries is recalculated to adapt to the current best hypothesis, so that no unnecessary tries are carried out [36].

Notice that the maximum number of tries does *not* depend on the number of measurements. Experiments show that this fact is crucial in reducing the computational complexity of the RS algorithm.

ALGORITHM 9.4
Relocation using RANSAC

```
Relocation_RS ($\mathcal{H}$)
$P_{\text{fail}}$ = 0.05, p = 3, $P_g$ = 0.5
Best = []
$i$ = 0
**repeat**
   $\hat{\mathbf{z}}$ = random_permutation($\hat{\mathbf{z}}$)
   RS([], 1)
   $P_g$ = max($P_g$, pairings(Best) / m)
   t = log $P_{\text{fail}}$ / log $\left(1 - {P_g}^p\right)$
   i = i + 1
**until** $i \geq t$
**return** Best

**procedure** RS ($\mathcal{H}$):
{$H$ : *current hypothesis*}
{$i$ : *observation to be matched*}
**if** $i > m$ **then**
   **if** pairings(H) > pairings(Best) **then**
      Best = H
   **end if**
**else if** pairings(H) == 3 **then**
   $\mathbf{x}_R^B$ = estimate_location_(H)
   **if** joint_compatibility(H) **then**
      JCBB(H, i) { *hypothesis verification*}
   **end if**
**else** {*branch and bound without star node*}
   **for** j = 1 to $n$ **do**
      **if** unary(i, j) $\wedge$ binary(i, j, H) **then**
         RS([H j], i + 1)
      **end if**
   **end for**
**end if**
```

9.3.4 Locality

As explained in Section 9.3.3, the main problem of the interpretation tree approach is the exponential number of possible hypotheses (tree leaves): $N_h = (n+1)^m$. The use of geometric constraints and branch and bound search dramatically reduce the number of nodes explored, by cutting down entire branches of the tree. However, Grimson [30] has shown that in the general case where spurious measurements can arise, the amount of search needed to find the best interpretation is still exponential. In these conditions, the interpretation tree approach seems impracticable except for very small maps.

To overcome this difficulty we introduce the concept of *locality*: given that the set of measurements has been obtained from a unique vehicle location (or from a set of nearby locations), it is sufficient to try to find matchings with *local* sets of features in the map. Given a map feature F_j, we define its *locality* $L(F_j)$ as the set of map features that are in the vicinity to it, such that they can be seen from the same vehicle location. For a given mapping problem, the maximum cardinality of the locality sets will be a constant c that depends on the sensor range and the maximum density of features in the environment.

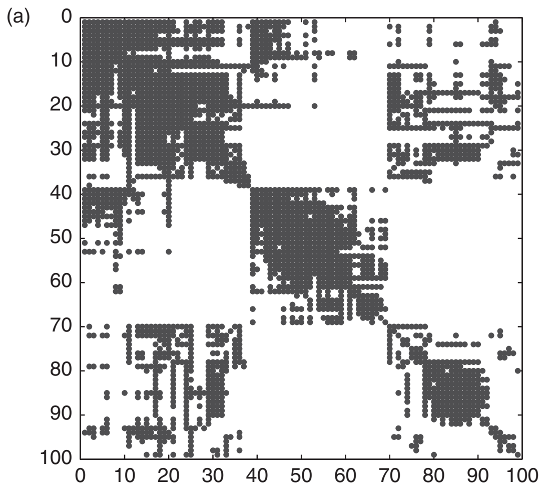
During the interpretation tree search, once a matching has been established with a map feature, the search can be restricted to its locality set. For the first measurement, there are n possible feature matchings. Since there are at most c features covisible with the first one, for the remaining $m-1$ measurements there are only c possible matches, giving a maximum of $n(c+1)^{m-1}$ hypotheses. If the first measurement is not matched, a similar analysis can be done for the second tree level. Thus, the total number of hypotheses N_h will be:

$$N_h \leq n(c+1)^{m-1} + \cdots + n + 1 = n\frac{(c+1)^m - 1}{c} + 1 \qquad (9.22)$$

This implies that, using locality, the complexity of searching the interpretation tree will be *linear* with the size of the map.

There are several ways of implementing locality:

1. SLAM can be implemented by building sequences of independent local maps [10]. If the local maps are stored, the search for matchings can be performed in time linear with the number of local maps. In this case, the locality of a feature is the set of features belonging to the same local map. A drawback of this technique is that global localization may fail around the borders between two local maps.
2. Alternatively, the locality of a feature can be computed as the set of map features within a distance less than the maximum sensor range. There are two drawbacks in this approach: first, this will require $O(n^2)$ distance computations, and second, in some cases features that are close cannot be seen simultaneously (e.g., because they are in different rooms), and thus should not be considered local.
3. The locality of a feature can be defined as the set of features that have been seen simultaneously with it at least once. We choose this last alternative, because it does not suffer from the limitations of the first



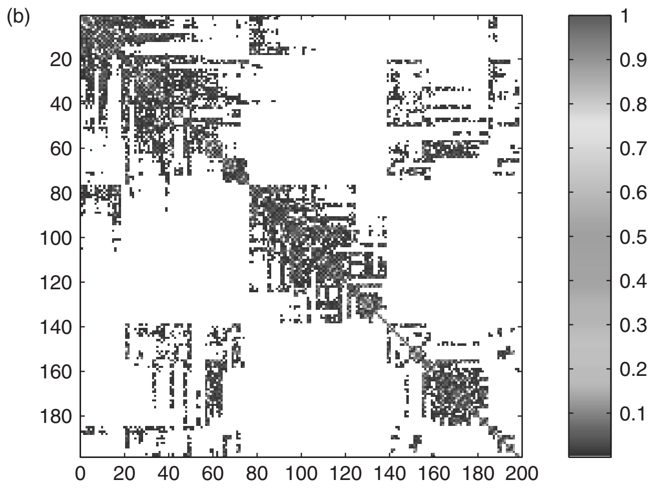


FIGURE 9.6 Covisibility matrix (a) and normalized information matrix (b)

two, and additionally it can be done during map building without extra cost.

Figure 9.6a shows the covisibility matrix obtained during map building for the first 1000 steps of the dataset obtained by Guivant and Nebot [3], gathered with a vehicle equipped with a SICK laser scanner in Victoria Park, Sydney. Wheel encoders give an odometric measure of the vehicle location. The laser scans are processed using Guivant's algorithm to detect tree trunks and estimate their radii (Figure 9.7). As features are added incrementally during map

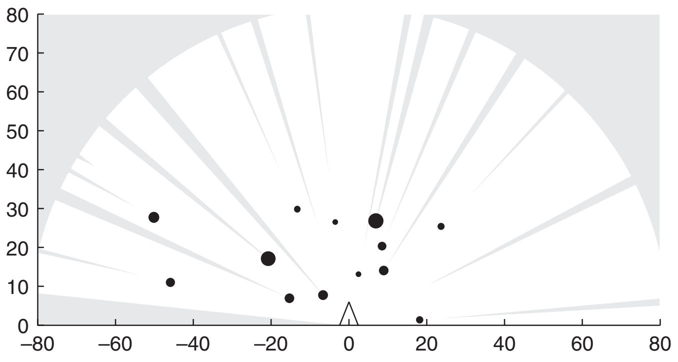


FIGURE 9.7 Segmentation of scan 120, with $m = 13$ tree trunks detected. Radii are magnified $\times 5$

building, the typical form of the covisibility matrix is band-diagonal. Elements far from the diagonal appear when a loop is closed, because recently added features become covisible with previously mapped features. In any case, the number of elements per row or column only depends on the density of features and the sensor reach. Using a sparse matrix representation, the amount of memory needed to store the covisibility matrix (or any other locality matrix) is $O(n)$.

An important property of the covisibility matrix is its close relation to the information matrix of the map (the inverse of the map covariance matrix). Figure 9.6b shows the normalized information matrix, where each row and column has been divided by the square root of the corresponding diagonal element. It is clear that the information matrix allows the determination of those features that are seen from the vehicle location during map building. The intuitive explanation is that as the uncertainty in the absolute vehicle location grows, the information about the features that are seen from the same location becomes highly coupled.

This gives further insight on the structure of the SLAM problem: while the map covariance matrix is a full matrix with $O(n^2)$ elements, the normalized information matrix tends to be sparse, with $O(n)$ elements. This fact can be used to obtain more efficient SLAM Algorithms [37].

Running continuous SLAM for the first 1000 steps, we obtain a map of $n = 99$ point features (see Figure 9.8). To verify the vehicle locations obtained by our algorithm, we obtained a reference solution running continuous SLAM until step 2500. Figure 9.8 shows the reference vehicle location for steps 1001 to 2500. The RS relocation algorithm was executed on scans 1001 to 2500. This guarantees that we use scans statistically independent from the stochastic map. The radii of the trunks are used as unary constraints, and the distance between the centers as binary constraints.

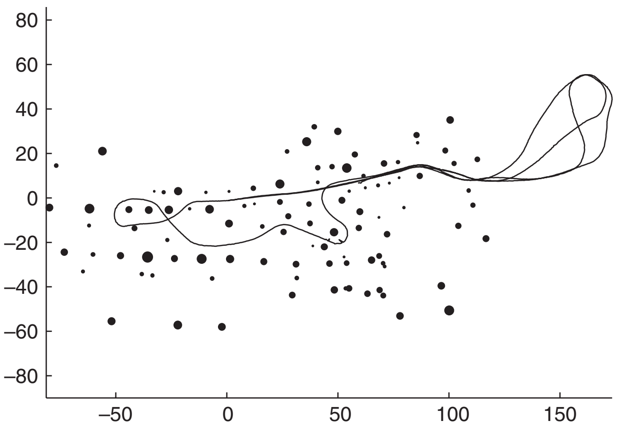


FIGURE 9.8 Stochastic map of 2D points (tree trunks) built until step 1000. There are $n = 99$ features. Reference vehicle trajectory for steps 1001 to 2500. Trunk radii are magnified $\times 5$

In this experiment, when six or more measurements are paired, the algorithm finds the solution with no false positives. Otherwise, the solution must be discarded as being unreliable. In case that less than six points are segmented from the scan, more sensor information is necessary to reliably determine the vehicle location. When the vehicle is in the map, the RS algorithm finds the solution with a mean execution time of less than 1 sec (in MATLAB, and executed on a Pentium IV, at 1.7 GHz). When the vehicle is not in the mapped area, for up to 30 measurements, RS runs in less than 2 sec (see Reference 33 for full details).

9.4 MAPPING LARGE ENVIRONMENTS

The EKF–SLAM techniques presented in previous sections have two important limitations when trying to map large environments. First, the computational cost of updating the map grows with $O(n^2)$, where n is the number of features in the map. Second, as the map grows, the estimates obtained by the EKF equations quickly become inconsistent due to linearization errors [9].

An alternative technique that reduces the computational cost and improves consistency is local map joining [10]. Instead of building one global map, this technique builds a set of independent local maps of limited size. Local maps can be joined together into a global map that is equivalent to the map obtained by the standard EKF–SLAM approach, except for linearization errors. As most of the mapping process consists in updating local maps, where errors remain small, the consistency of the global map obtained is greatly improved. In the following sections we present the basics of local map joining.

9.4.1 Building Independent Local Maps

Each local map can be built as follows: at a given instant t_j, a new map is initialized using the current vehicle location as base reference B_j. Then, the vehicle performs a limited motion (say k_j steps) acquiring sensor information about the neighboring environment features $\mathcal{F}_j$. The standard EKF-based techniques presented in previous sections are used to obtain a local map $\mathcal{M}_{\mathcal{F}_j}^{\mathcal{B}_j} = (\hat{\mathbf{x}}_{\mathcal{F}_j}^{\mathcal{B}_j}, \mathbf{P}_{\mathcal{F}_\mathbf{j}}^{\mathcal{B}_\mathbf{j}})$. This local map is independent of any prior estimation of the vehicle location because it is built relative to the initial vehicle location B_j. The local map depends only on the odometry and sensor data obtained during the k_j steps. This implies that, under the common assumption that process and measurement noise are *white* random sequences, two local maps built with the same robot from *disjoint* sequences of steps are functions of independent stochastic variables. Therefore, the two maps will be *statistically independent* and *uncorrelated*. As there is no need to compute the correlations between features in different local maps and the size of local maps is bounded, the cost of local map building is constant per step, independent from the size of the global map.

The decision to close map $\mathcal{M}_j$ and start a new local map is made once the number of features in the current local map reaches a maximum, or the uncertainly of the vehicle location with respect to the base reference of the current map reaches a limit, or no matchings were found by the data association process for the last sensor measurements (a separate region of the environment is observed). Note that the new local map $\mathcal{M}_{j+1}$ will have the current vehicle position as base reference, which corresponds to the last vehicle position in map $\mathcal{M}_j$. Thus, the relative transformation between the two consecutive maps $\mathbf{x}_{j+1} = \mathbf{x}_{B_{j+1}}^{B_j}$ is part of the state vector of map $\mathcal{M}_j$.

9.4.2 Local Map Joining

Given two uncorrelated local maps:

$$\mathcal{M}_{\mathcal{F}}^{B} = (\hat{\mathbf{x}}_{\mathcal{F}}^{B}, \mathbf{P}_{\mathcal{F}}^{B}); \quad \mathcal{F} = \{B, F_0, F_1, \ldots, F_n\}$$
$$\mathcal{M}_{\varepsilon}^{B'} = (\hat{\mathbf{x}}_{\varepsilon}^{B'}, \mathbf{P}_{\varepsilon}^{B'}); \quad \varepsilon = \{B', E_0, E_1, \ldots, E_m\}$$

where a common reference has been identified $F_i = E_j$, the goal of map joining is to obtain one full stochastic map:

$$\mathcal{M}_{\mathcal{F}+\varepsilon}^{B} = (\hat{\mathbf{x}}_{\mathcal{F}+\varepsilon}^{B}, \mathbf{P}_{\mathcal{F}+\varepsilon}^{B})$$

containing the estimates of the features from both maps, relative to a common base reference B, and to compute the correlations appearing in the process. Given that the features from the first map are expressed relative to reference B, to form the joint state vector $\mathbf{x}^B_{\mathcal{F}+\varepsilon}$ we only need to transform the features of the second map to reference B using the fact that $F_i = E_j$:

$$\hat{\mathbf{x}}^B_{\mathcal{F}+\varepsilon} = \begin{bmatrix} \hat{\mathbf{x}}^B_{\mathcal{F}} \\ \hat{\mathbf{x}}^B_{\varepsilon} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{x}}^B_{\mathcal{F}} \\ \hat{\mathbf{x}}^B_{\mathcal{F}_i} \oplus \hat{\mathbf{x}}^{E_j}_{E_0} \\ \vdots \\ \hat{\mathbf{x}}^B_{\mathcal{F}_i} \oplus \hat{\mathbf{x}}^{E_j}_{E_m} \end{bmatrix} \tag{9.23}$$

The covariance $\mathbf{P}^B_{\mathcal{F}+\varepsilon}$ of the joined map is obtained from the linearization of Equation 9.23, and is given by:

$$\begin{aligned} \mathbf{P}^B_{\mathcal{F}+\varepsilon} &= \mathbf{J}_{\mathcal{F}} \mathbf{P}^B_{\mathcal{F}} \mathbf{J}_{\mathcal{F}}{}^{\mathrm{T}} + \mathbf{J}_{\varepsilon} \mathbf{P}^{E_j}_{\varepsilon} \mathbf{J}_{\varepsilon}{}^{\mathrm{T}} \\ &= \begin{bmatrix} \mathbf{P}^B_{\mathcal{F}} & \mathbf{P}^B_{\mathcal{F}} \mathbf{J}_1^{\mathrm{T}} \\ \mathbf{J}_1 \mathbf{P}^B_{\mathcal{F}} & \mathbf{J}_1 \mathbf{P}^B_{\mathcal{F}} \mathbf{J}_1^{\mathrm{T}} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & \mathbf{J}_2 \mathbf{P}^{E_j}_{\varepsilon} \mathbf{J}_2^{\mathrm{T}} \end{bmatrix} \end{aligned} \tag{9.24}$$

where

$$\mathbf{J}_{\mathcal{F}} = \left.\frac{\partial \mathbf{x}^B_{\mathcal{F}+\varepsilon}}{\partial \mathbf{x}^B_{\mathcal{F}}}\right|_{(\hat{\mathbf{x}}^B_{\mathcal{F}}, \hat{\mathbf{x}}^{E_j}_{\varepsilon})} = \begin{bmatrix} \mathbf{I} \\ \mathbf{J}_1 \end{bmatrix}$$

$$\mathbf{J}_{\varepsilon} = \left.\frac{\partial \mathbf{x}^B_{\mathcal{F}+\varepsilon}}{\partial \mathbf{x}^{E_j}_{\varepsilon}}\right|_{(\hat{\mathbf{x}}^B_{\mathcal{F}}, \hat{\mathbf{x}}^{E_j}_{\varepsilon})} = \begin{bmatrix} 0 \\ \mathbf{J}_2 \end{bmatrix}$$

$$\mathbf{J}_1 = \begin{bmatrix} 0 & \dots & \mathbf{J}_{1\oplus}\left\{\hat{\mathbf{x}}^B_{F_i}, \hat{\mathbf{x}}^{E_j}_{E_0}\right\} & \dots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & \mathbf{J}_{1\oplus}\left\{\hat{\mathbf{x}}^B_{F_i}, \hat{\mathbf{x}}^{E_j}_{E_m}\right\} & \dots & 0 \end{bmatrix}$$

$$\mathbf{J}_2 = \begin{bmatrix} \mathbf{J}_{2\oplus}\left\{\hat{\mathbf{x}}^B_{F_i}, \hat{\mathbf{x}}^{E_j}_{E_0}\right\} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{J}_{2\oplus}\left\{\hat{\mathbf{x}}^B_{F_i}, \hat{\mathbf{x}}^{E_j}_{E_m}\right\} \end{bmatrix}$$

Obtaining vector $\hat{\mathbf{x}}^B_{\mathcal{F}+\varepsilon}$ with Equation 9.23 is an $O(m)$ operation. Given that the number of nonzero elements in $\mathbf{J}_1$ and $\mathbf{J}_2$ is $O(m)$, obtaining matrix $\mathbf{P}^B_{\mathcal{F}+\varepsilon}$ with Equation 9.24 is an $O(nm + m^2)$ operation. Thus when $n \gg m$, map joining is linear with n.

9.4.3 Matching and Fusion After Map Joining

The map resulting from map joining may contain features that, coming from different local maps, correspond to the same environment feature. To eliminate such duplications and obtain a more precise map we need a data association algorithm to determine correspondences, and a feature fusion mechanism to update the global map. For determining correspondences we use the JCBB algorithm described in Section 9.3.2

Feature fusion is performed by a modified version of the EKF update equations, which consider a nonlinear measurement equation:

$$\mathbf{z}_{ij} = \mathbf{h}_{ij}(\mathbf{x}) = 0 \tag{9.25}$$

with null noise covariance matrix, which constraints the relative location between the duplicates F_i and F_j of an environment feature. Once the matching constraints have been applied, the corresponding matching features become fully correlated, with the same estimation and covariance. Thus, one of them can be eliminated.

The whole process of local map joining, matching, and fusion can be seen in the example of Figure 9.9.

9.4.4 Closing a Large Loop

To compare map joining with full EKF–SLAM we have performed a map building experiment, using a robotized wheelchair equipped with a SICK laser scanner. The vehicle was hand-driven along a loop of about 250 m in a populated indoor/outdoor environment in the Ada Byron building of our campus. The laser scans were segmented to obtain lines using the RANSAC technique. The global map obtained using the classical EKF–SLAM algorithm is shown in Figure 9.10a. At this point, the vehicle was very close to the initial starting position, closing the loop. The figure shows that the vehicle estimated location has some 10 m error and the corresponding 95% uncertainty ellipses are ridiculously small, giving an inconsistent estimation. Due to these small ellipsoids, the JCBB data association algorithm was unable to properly detect the loop closure. This corroborates the results obtained with simulations in Reference 9: in large environments the map obtained by EKF–SLAM quickly becomes inconsistent, due to linearization errors.

The same dataset was processed to obtain independent local maps at fixed intervals of about 10 m. The local maps were joined and fused obtaining the global map shown in Figure 9.10b. In this case the loop was correctly detected by JCBB and the map obtained seems to be consistent. Furthermore, the computational time was about 50 times shorter that the standard EKF approach.

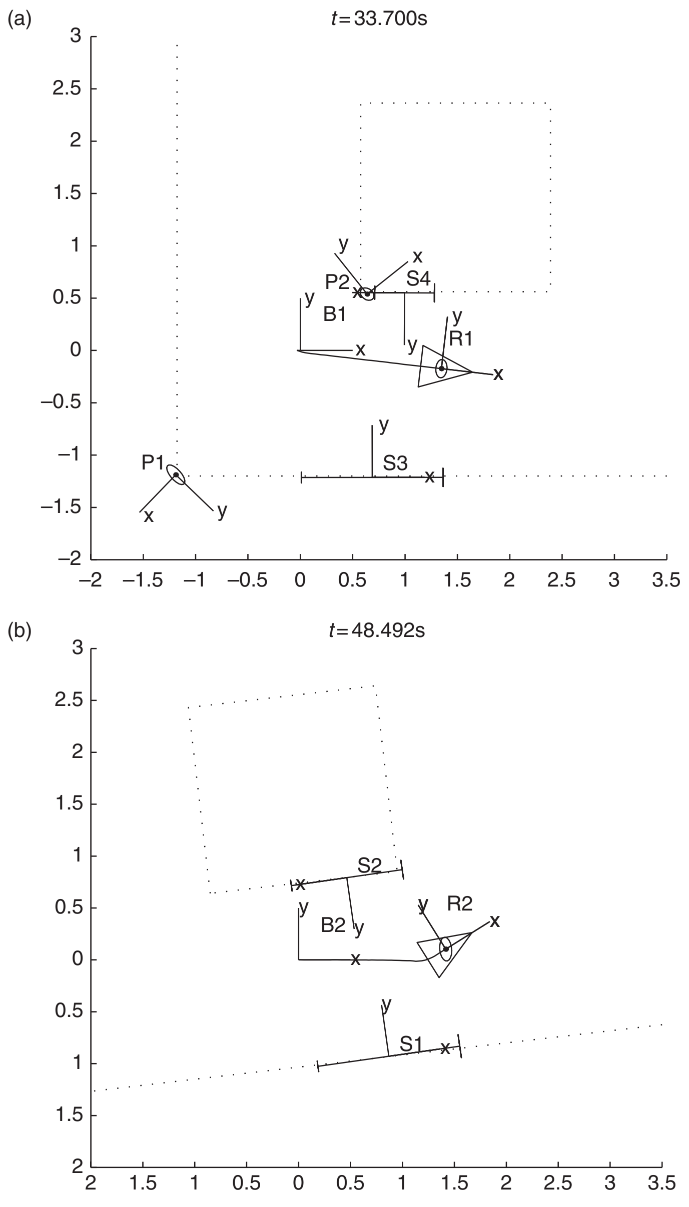
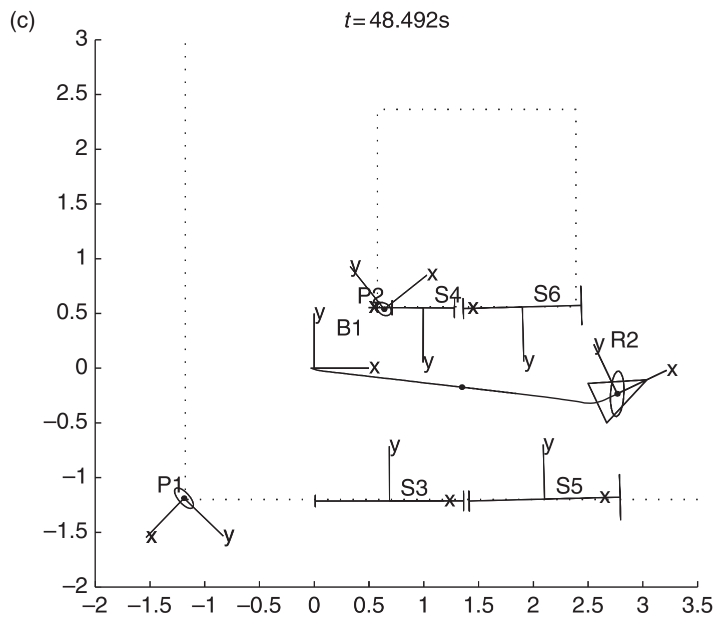


FIGURE 9.9 Example of local map joining. (a) Local map $\mathcal{M}^{B1}_{\mathcal{F}_1}$ with four point features, P1, P2, S3, and a segment S4, with respect to reference $B1$; (b) local map $\mathcal{M}^{B2}_{\mathcal{F}_2}$ with two features, S1 and S2, with respect to reference $B2$; (c) both maps are joined to obtain $\mathcal{M}^{B1}_{\mathcal{F}_1+\mathcal{F}_2}$; (d) map $\mathcal{M}^{B1}_{\mathcal{F}_{1:2}}$ after updating by fusing S3 with S5, and S4 with S6. Reprinted with permission [10]



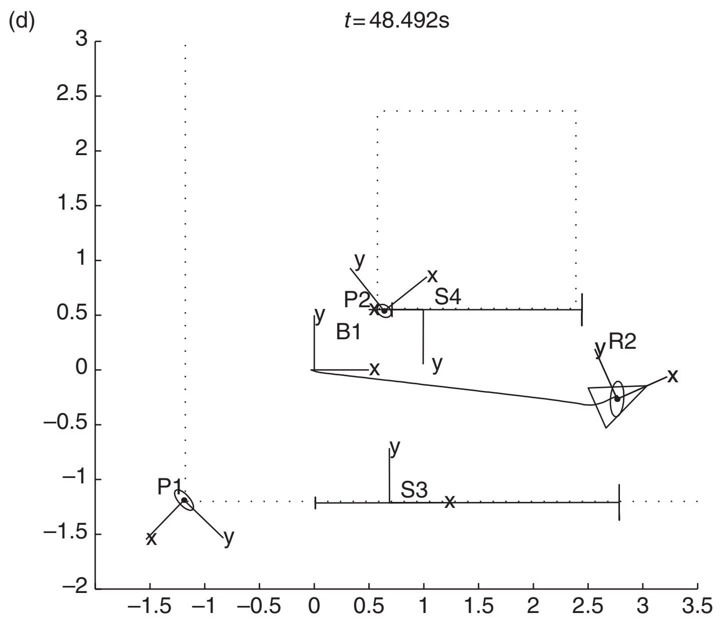


FIGURE 9.9 Continued

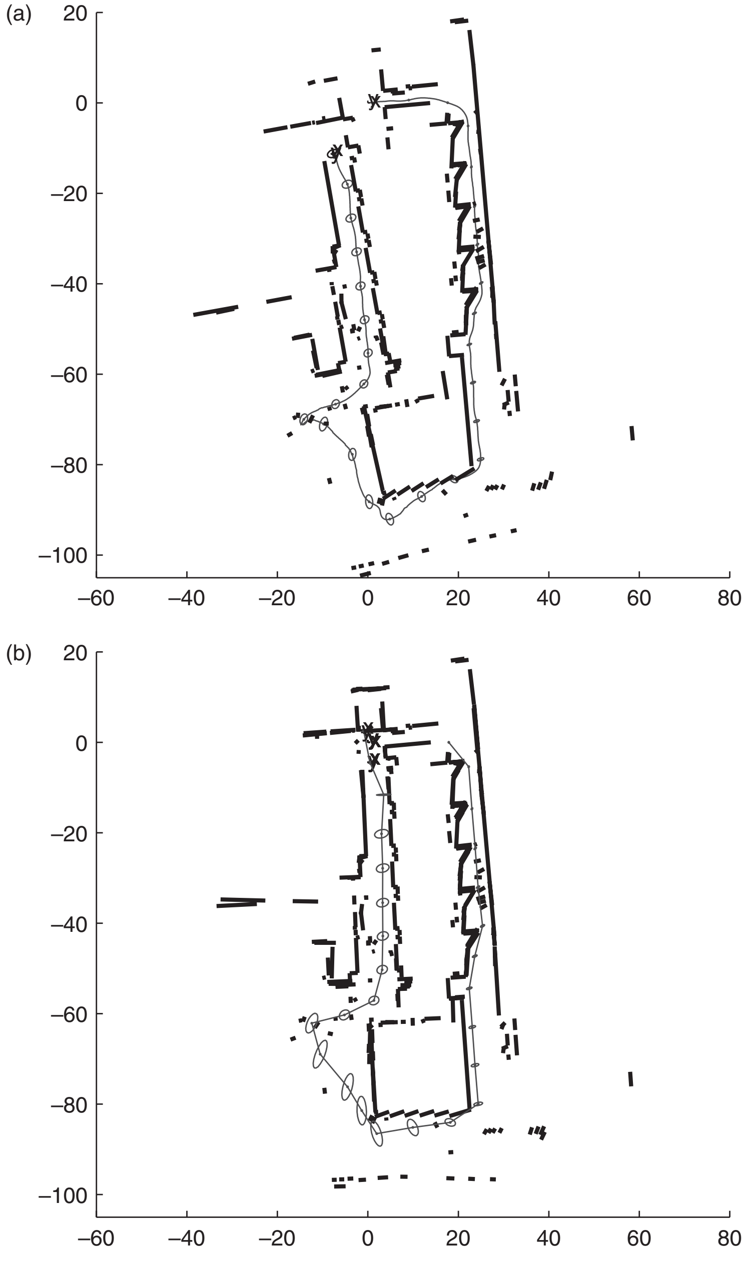


FIGURE 9.10 Global maps obtained using the standard EKF–SLAM algorithm (a) and local map joining (b)

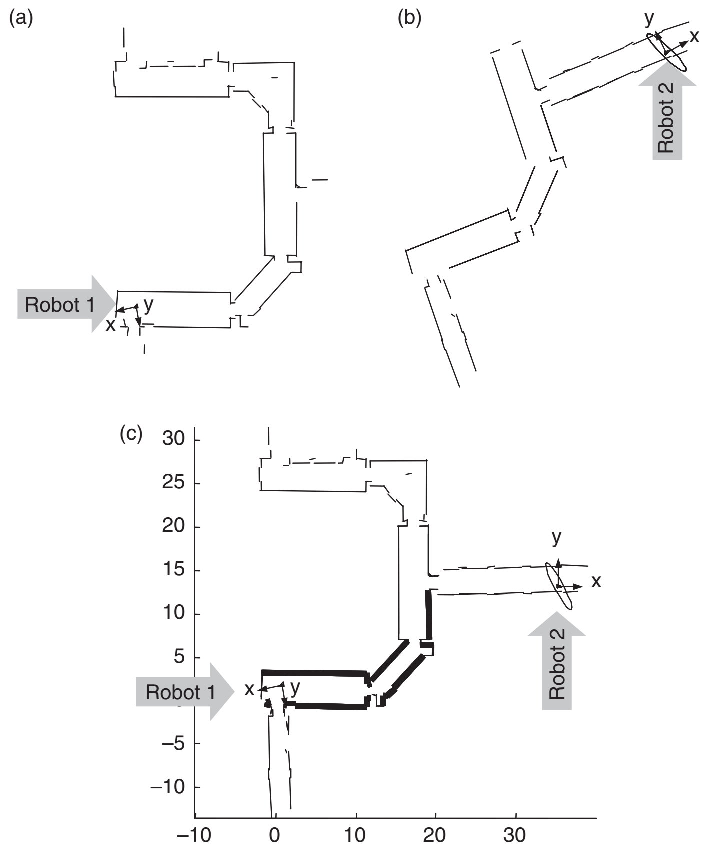


FIGURE 9.11 Maps build by two independent robots (a, b) and global map obtained joining them (c)

9.4.5 Multi-robot SLAM

The techniques explained above can be applied to obtain global maps of large environments using several mobile robots. In Figure 9.11a and b, we can see the maps built by two independent robots that have traversed a common area. In this case, the relative location between the robots is unknown. The process for obtaining a common global map is as follows:

- Choose at random one feature on the first map, pick its set of covisible features and search for matchings in the second map using the RS relocation algorithm. Repeat the process until a good matching is found, for a fixed maximum number of tries.

- When a match is found, choose a common reference in both maps. In this case the reference is built in the intersection of two nonparallel walls that have been matched by RS. This gives the relative location between both maps. Change the base of the second map to be the common reference using the technique detailed in Reference 10.
- Join both maps (Section 9.4.2), search for more matchings using JCBB, and fuse both maps in a global map (Section 9.4.3) that contains the location of all features and both robots, relative to the base of the first map.

The global map obtained is shown in Figure 9.11c. The bold lines axe the covisibility set used to match both maps. After that point, both robots can continue exploring the environment, building new independent local maps that can be joined and fused with the global map.

9.5 Conclusions

The EKF approach to SLAM dates back to the seminal work reported in Reference 2 where the idea of representing the structure of the navigation area in a discrete-time state-space framework was originally presented. Nowadays the basic properties and limitations of this approach are quite well understood. Three important convergence properties were proven in Reference 26: (1) the determinant of any submatrix of the map covariance matrix decreases monotonically as observations are successively made, (2) in the limit, as the number of observations increases, the landmark estimates become fully correlated, and (3) in the limit, the covariance associated with any single landmark location estimate reaches a lower bound determined only by the initial covariance in the vehicle location estimate at the time of the first sighting of the first landmark.

It is important to note that these theoretical results only refer to the evolution of the covariance matrices computed by the EKF in the ideal linear case. They overlook the fact that, given that SLAM is a nonlinear problem, there is no guarantee that the computed covariances will match the actual estimation errors, which is the true SLAM consistency issue first pointed out in Reference 38. In a recent paper [9], we showed with simulations that linearization errors lead to inconsistent estimates well before the computational problems arise. In Section 9.4 we have presented experimental evidence that methods like map joining, based on building independent local maps, effectively reduce linearization errors, improving the estimator consistency.

The main open challenges in SLAM include efficient mapping of large environments, modeling complex and dynamic environments, multi-vehicle SLAM, and full 3D SLAM. Most of these challenges will require scalable representations, robust data association algorithms, consistent estimation techniques, and different sensor modalities. In particular, solving SLAM with monocular or stereo vision is a crucial research goal for addressing many real life applications.

APPENDIX: TRANSFORMATIONS IN 2D

Two basic operations used in stochastic mapping are transformation inversion and composition, which were represented by Reference 2 using operators $\ominus$ and $\oplus$:

$$\hat{\mathbf{x}}_A^B = \ominus\hat{\mathbf{x}}_B^A$$
$$\hat{\mathbf{x}}_C^A = \hat{\mathbf{x}}_B^A \oplus \hat{\mathbf{x}}_C^B$$

In this chapter, we generalize the $\oplus$ operator to also represent the composition of transformations with feature location vectors, which results in the change of base reference of the feature. The Jacobians of these operations are defined as:

$$\mathbf{J}_{\ominus}\{\hat{\mathbf{x}}_B^A\} = \left.\frac{\partial(\ominus\mathbf{x}_B^A)}{\partial\mathbf{x}_B^A}\right|_{(\hat{\mathbf{x}}_B^A)}$$

$$\mathbf{J}_{1\oplus}\{\hat{\mathbf{x}}_B^A, \hat{\mathbf{x}}_C^B\} = \left.\frac{\partial(\mathbf{x}_B^A \oplus \mathbf{x}_C^B)}{\partial\mathbf{x}_B^A}\right|_{(\hat{\mathbf{x}}_B^A, \hat{\mathbf{x}}_C^B)}$$

$$\mathbf{J}_{2\oplus}\{\hat{\mathbf{x}}_B^A, \hat{\mathbf{x}}_C^B\} = \left.\frac{\partial(\mathbf{x}_B^A \oplus \mathbf{x}_C^B)}{\partial\mathbf{x}_C^B}\right|_{(\hat{\mathbf{x}}_B^A, \hat{\mathbf{x}}_C^B)}$$

In 2D, the location of a reference B relative to a reference A (or transformation from A to B) can be expressed using a vector with three d.o.f.: $\mathbf{x}_B^A = [x_1, y_1, \phi_1]^{\mathrm{T}}$. The location of A relative to B is computed using the inversion operation:

$$\mathbf{x}_A^B = \ominus\mathbf{x}_B^A = \begin{bmatrix} -x_1\cos\phi_1 - y_1\sin\phi_1 \\ x_1\sin\phi_1 - y_1\cos\phi_1 \\ -\phi_1 \end{bmatrix}$$

The Jacobian of transformation inversion is:

$$\mathbf{J}_{\ominus}\{\mathbf{x}_B^A\} = \begin{bmatrix} -\cos\phi_1 & -\sin\phi_1 & -x_1\sin\phi_1 - y_1\cos\phi_1 \\ \sin\phi_1 & -\cos\phi_1 & x_1\cos\phi_1 + y_1\sin\phi_1 \\ 0 & 0 & -1 \end{bmatrix}$$

Let $\mathbf{x}_C^B = [x_2, y_2, \phi_2]^T$ be a second transformation. The location of reference C relative to A is obtained by the composition of transformations $\mathbf{x}_B^A$ and $\mathbf{x}_C^B$:

$$\mathbf{x}_C^A = \mathbf{x}_B^A \oplus \mathbf{x}_C^B = \begin{bmatrix} x_1 + x_2 \cos\phi_1 - y_2 \sin\phi_1 \\ y_1 + x_2 \sin\phi_1 + y_2 \cos\phi_1 \\ \phi_1 + \phi_2 \end{bmatrix}$$

The Jacobians of transformation composition are:

$$\mathbf{J}_{1\oplus}\{\mathbf{x}_B^A, \mathbf{x}_C^B\} = \begin{bmatrix} 1 & 0 & -x_2 \sin\phi_1 - y_2 \cos\phi_1 \\ 0 & 1 & x_2 \cos\phi_1 - y_2 \sin\phi_1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{J}_{2\oplus}\{\mathbf{x}_B^A, \mathbf{x}_C^B\} = \begin{bmatrix} \cos\phi_1 & -\sin\phi_1 & 0 \\ \sin\phi_1 & \cos\phi_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Acknowledgment

This research has been funded in part by the Dirección General de Investigación of Spain under project DPI2003-07986.

References

1. J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós. THE SPmap: a probabilistic framework for simultaneous localization and map building. *IEEE Transactions on Robotics Automation*, 15: 948–953, 1999.
2. R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In Faugeras O. and Giralt G., editors, *Robotics Research, The Fourth International Symposium*, pp. 467–474. The MIT Press, Cambridge, MA, 1988.
3. J. E. Guivant and E. M. Nebot. Optimization of the simultaneous localization and map-building algorithm for real-time implementation. *IEEE Transactions on Robotics and Automation*, 17: 242–257, 2001.
4. J. J. Leonard, R. Carpenter and H. J. S. Feder. Stochatic mapping using forward look sonar. *Robotica*, 19, 467–480
5. J. H. Kim and S. Sukkarieh. Airborne simultaneous localisation and map building. In *IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 2003.
6. J. Knight, A. Davison, and I. Reid. Towards constant time SLAM using postponement. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 406–412, Maui, Hawaii, 2001.

7. S. J. Julier and J. K. Uhlmann. Building a million beacon map. In Paul S. Schenker, and Gerard T. McKee (eds), *SPIE Int. Soc. Opt. Eng.*, vol. 4571, pp. 10-21, Washington DC, 2001.
8. Y. Liu and S. Thrun. Results for outdoor-SLAM using sparse extended information filters. In *IEEE International Conference on Robotics and Automation*, pp.1227–1233, Taipei, Taiwan, 2003.
9. J. A. Castellanos, J. Neira, and J. D. Tardós. Limits to the consistency of EKF-based SLAM. In *5th IFAC Symposium on Intelligent Autonomous Vehicles*, Lisbon, Portugal, 2004.
10. J. D. Tardós, J. Neira, P. Newman, and J. Leonard. Robust mapping and localization in indoor environments using sonar data. *International Journal of Robotics Research*, 21: 311–330, 2002.
11. S. B. Williams, G. Dissanayake, and H. Durrant-Whyte. An efficient approach to the simultaneous localisation and mapping problem. In *IEEE International Conference on Robotics and Automation, ICRA*, vol. 1, pp. 406–411, Washington DC, 2002.
12. S. B. Williams. *Efficient Solutions to Autonomous Mapping and Navigation Problems*. Australian Centre for Field Robotics, University of Sydney, September 2001. http://www.acfr.usyd.edu.au/
13. M. Bosse, P. Newman, J. Leonard, M. Soika, W. Feiten, and S. Teller. An atlas framework for scalable mapping. In *IEEE International Conference on Robotics and Automation*, pp. 1899–1906, Taipei, Taiwan, 2003.
14. T. Bailey *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*, Australian Centre for Field Robotics, University of Sydney, August 2002, http://www.acfr.usyd.edu.au/
15. J. J. Leonard and P. M. Newman. Consistent, Convergent and Constant-Time SLAM. In *International Joint Conference on Artificial Intelligence*, Acapulco, Mexico, August 2003.
16. C. Estrada, J. Neira, and J. D. Tardós. Hierarchical SLAM: real-time accurate mapping of large environments. *IEEE Transactions on Robotics and Automation*, 2005 (to appear).
17. M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Edmonton, Canada, 2002, AAAI.
18. J. S. Gutmann and K. Konolige. Incremental Mapping of Large Cyclic Environments. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA*, pp. 318–325, 1999.
19. F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4: 333–349, 1997.
20. S. Thrun, D. Hähnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker. A system for volumetric robotic mapping of abandoned mines. In *IEEE International Conference on Robotics and Automation*, pp. 4270–4275, Taipei, Taiwan, 2003.
21. J. A. Castellanos and J. D. Tardós *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers, Boston, MA, 1999.

AQ: Please provide publisher name for reference [7].

AQ: Please update reference [16].

AQ: Please provide place of publisher for reference [18].

AQ: Please provide place of publisher for references [11,17]

22. P. Newman, J. Leonard, J. D. Tardós, and J. Neira. Explore and return: Experimental validation of real-time concurrent mapping and localization. In *IEEE International Conference on Robotics and Automation*, pp. 1802–1809. IEEE, 2002.
23. J.A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós Sensor influence in the performance of simultaneous mobile robot localization and map building. In Corke Peter and Trevelyan James (eds), *Experimental Robotics VI. Lecture Notes in Control and Information Sciences*, vol. 250, pp. 287–296. Springer-Verlag, Heidelberg, 2000.
24. A.J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *Proceedings of International Conference on Computer Vision, Nice*, October 2003.
25. J. A. Castellanos, J. Neira, and J. D. Tardós. Multisensor fusion for simultaneous localization and map building. *IEEE Transactions on Robotics and Automation*, 17: 908–914, 2001.
26. M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17: 229–241, 2001.
27. A. H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, New York, 1970.
28. T. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, New York, 1988.
29. Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation.* John Wiley & Sons, New York, 2001.
30. W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints*. The MIT Press, Cambridge, MA, 1990.
31. J. M. M Montiel and L. Montano. Efficient validation of matching hypotheses using mahalanobis distance. *Engineering Applications of Artificial Ingelligence*, 11: 439–448, 1998.
32. J. Neira and J. D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on Robotics and Automation*, 17: 890–897, 2001.
33. J. Neira, J. D. Tardós, and J. A. Castellanos. Linear time vehicle relocation in SLAM. In *IEEE International Conference on Robotics and Automation*, pp. 427–433, Taipei, Taiwan, September 2003.
34. T. Bailey, E. M. Nebot, J. K. Rosenblatt, and H. F. Durrant-Whyte. Data association for mobile robot navigation: a graph theoretic approach. In *IEEE International Conference Robotics and Automation*, pp. 2512–2517, San Francisco, California, 2000.
35. M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. Assoc. Comp. Mach.*, 24: 381–395, 1981.
36. R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, U.K., 2000.
37. Thrun Sebastian, Liu Yufeng, Koller Daphne, Y. Ng Andrew, Ghahramani Zoubin, and Durrant-Whyte Hugh. Simultaneous Localization and

Mapping with Sparse Extended Information Filters. *The International Journal of Robotics Research*, 23: 693–716, 2004.

38. S. J. Julier and J. K. Uhlmann A counter example to the theory of simultaneous localization and map building. In *2001 IEEE International Conference on Robotics and Automation*, pp. 4238–4243, Seoul, Korea, 2001.

Biographies

José A. Castellanos was born in Zaragoza, Spain, in 1969. He received the M.S. and Ph.D. degrees in Industrial-Electrical Engineering from the University of Zaragoza, Spain, in 1994 and 1998, respectively. He is an Associate Professor with the Departamento de Informática e Ingeniería de Sistemas, University of Zaragoza, where he is in charge of courses in SLAM, Automatic Control Systems and Computer Modelling and Simulation. His current research interest include multisensor fusion and integration, Bayesian estimation in nonlinear systems and simultaneous localization and mapping.

José Neira was born in Bogotá, Colombia, in 1963. He received the M.S. degree in Computer Science from the Universidad de los Andes, Colombia, in 1986, and the Ph.D. degree in Computer Science from the University of Zaragoza, Spain, in 1993. He is an Associate Professor with the Departamento de Informática e Ingeniería de Sistemas, University of Zaragoza, where he is in charge of courses in Compiler Theory, Computer Vision and Mobile Robotics. His current research interests include autonomous robots, data association, and environment modeling.

Juan D. Tardós was born in Huesca, Spain, in 1961. He received the M.S. and Ph.D. degrees in Industrial-Electrical Engineering from the University of Zaragoza, Spain, in 1985 and 1991, respectively. He is an Associate Professor with the Departamento de Informática e Ingeniería de Sistemas, University of Zaragoza, where he is in charge of courses in Real Time Systems, Computer Vision and Artificial Intelligence. His current research interests include perception and mobile robotics.