

# 10 *Simultaneous Localization and Mapping*

## 10.1 Introduction

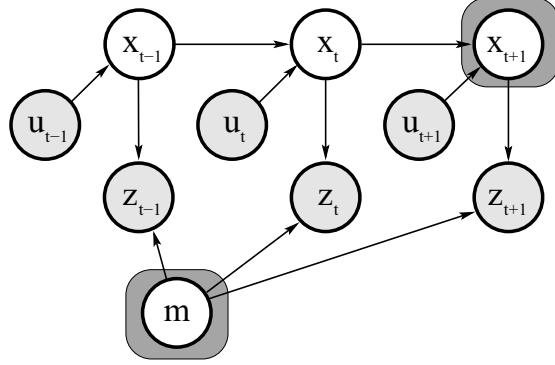
This and the following chapters address one of the most fundamental problems in robotics, the *simultaneous localization and mapping problem*. This problem is commonly abbreviated as *SLAM*, and is also known as *Concurrent Mapping and Localization*, or *CML*. SLAM problems arise when the robot does not have access to a map of the environment, nor does it know its own pose. Instead, all it is given are measurements  $z_{1:t}$  and controls  $u_{1:t}$ . The term “simultaneous localization and mapping” describes the resulting problem: In SLAM, the robot acquires a map of its environment while simultaneously localizing itself relative to this map. SLAM is significantly more difficult than all robotics problems discussed thus far. It is more difficult than localization in that the map is unknown and has to be estimated along the way. It is more difficult than mapping with known poses, since the poses are unknown and have to be estimated along the way.

ONLINE SLAM  
PROBLEM

From a probabilistic perspective, there are two main forms of the SLAM problem, which are both of equal practical importance. One is known as the *online SLAM problem*: It involves estimating the posterior over the momentary pose along with the map:

$$(10.1) \quad p(x_t, m \mid z_{1:t}, u_{1:t})$$

Here  $x_t$  is the pose at time  $t$ ,  $m$  is the map, and  $z_{1:t}$  and  $u_{1:t}$  are the measurements and controls, respectively. This problem is called the online SLAM problem since it only involves the estimation of variables that persist at time  $t$ . Many algorithms for the online SLAM problem are incremental: they discard past measurements and controls once they have been processed. The graphical model of online SLAM is depicted in Figure 10.1.



**Figure 10.1** Graphical model of the online SLAM problem. The goal of online SLAM is to estimate a posterior over the current robot pose along with the map.

#### FULL SLAM PROBLEM

The second SLAM problem is called the *full SLAM problem*. In full SLAM, we seek to calculate a posterior over the entire path  $x_{1:t}$  along with the map, instead of just the current pose  $x_t$  (see also Figure 10.2):

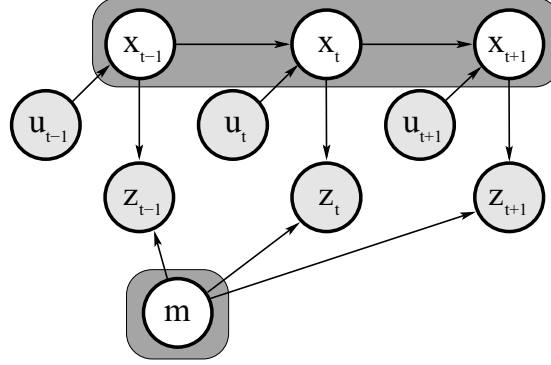
$$(10.2) \quad p(x_{1:t}, m \mid z_{1:t}, u_{1:t})$$

This subtle difference between online and full SLAM has ramifications in the type of algorithms that can be brought to bear. In particular, the online SLAM problem is the result of integrating out past poses from the full SLAM problem:

$$(10.3) \quad p(x_t, m \mid z_{1:t}, u_{1:t}) = \int \int \cdots \int p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) dx_1 dx_2 \dots dx_{t-1}$$

In online SLAM, these integrations are typically performed one-at-a-time. They cause interesting changes of the dependency structures in SLAM that we will fully explore in the next chapter.

A second key characteristic of the SLAM problem has to do with the nature of the estimation problem. SLAM problems possess a continuous and a discrete component. The continuous estimation problem pertains to the location of the objects in the map and the robot's own pose variables. Objects may be landmarks in feature-based representation, or they might be object patches detected by range finders. The discrete nature has to do with correspondence: When an object is detected, a SLAM algorithm must reason about the relation of this object to previously detected objects. This reasoning is typically discrete: Either the object is the same as a previously detected one, or it is not.



**Figure 10.2** Graphical model of the full SLAM problem. Here, we compute a joint posterior over the whole path of the robot and the map.

We already encountered similar continuous-discrete estimation problems in previous chapters. For example, EKF localization in Chapter 7.4 estimates the robot pose, which is continuous. But to do so it also estimates the correspondences of measurements and landmarks in the map, which are discrete. In this and the subsequent chapters, we will discuss a number of different techniques to deal with the continuous and the discrete aspects of the SLAM problem.

At times, it will be useful to make the correspondence variables explicit, as we did in Chapter 7 on localization. The online SLAM posterior is then given by

$$(10.4) \quad p(x_t, m, c_t \mid z_{1:t}, u_{1:t})$$

and the full SLAM posterior by

$$(10.5) \quad p(x_{1:t}, m, c_{1:t} \mid z_{1:t}, u_{1:t})$$

The online posterior is obtained from the full posterior by integrating out past robot poses and summing over all past correspondences:

$$(10.6) \quad \begin{aligned} p(x_t, m, c_t \mid z_{1:t}, u_{1:t}) \\ = \int \int \cdots \int \sum_{c_1} \sum_{c_2} \cdots \sum_{c_{t-1}} p(x_{1:t}, m, c_{1:t} \mid z_{1:t}, u_{1:t}) dx_1 dx_2 \cdots dx_{t-1} \end{aligned}$$

In both versions of the SLAM problems—online and full—estimating the full posterior (10.4) or (10.5) is the gold standard of SLAM. The full posterior captures all there is to be known about the map and the pose or the path.

In practice, calculating a full posterior is usually infeasible. Problems arise from two sources: (1) the high dimensionality of the continuous parameter space, and (2) the large number of discrete correspondence variables. Many state-of-the-art SLAM algorithms construct maps with tens of thousands of features, or more. Even under known correspondence, the posterior over those maps alone involves probability distributions over spaces with  $10^5$  or more dimensions. This is in stark contrast to localization problems, in which posteriors were estimated over three-dimensional continuous spaces. Further, in most applications the correspondences are unknown. The number of possible assignments to the vector of all correspondence variables  $c_{1:t}$  grows exponentially in the time  $t$ . Thus, practical SLAM algorithms that can cope with the correspondence problem must rely on approximations.

The SLAM problem will be discussed in a number of subsequent chapters. The remainder of this chapter develops an EKF algorithm for the on-line SLAM problem. Much of this material builds on Chapter 3.3, where the EKF was introduced, and Chapter 7.4, where we applied the EKF to the mobile robot localization problem. We will derive a progression of EKF algorithms that first apply EKFs to SLAM with known correspondences, and then progress to the more general case with unknown correspondences.

## 10.2 SLAM with Extended Kalman Filters

### 10.2.1 Setup and Assumptions

Historically the earliest—and perhaps the most influential—SLAM algorithm is based on the extended Kalman filter, or EKF. In a nutshell, the EKF SLAM algorithm applies the EKF to online SLAM using maximum likelihood data association. In doing so, EKF SLAM is subject to a number of approximations and limiting assumptions:

#### FEATURE-BASED MAPS

Maps, in EKF SLAM, are *feature-based*. They are composed of point landmarks. For computational reasons, the number of point landmarks is usually small (e.g., smaller than 1,000). Further, the EKF approach tends to work well the less ambiguous the landmarks are. For this reason, EKF SLAM requires significant engineering of feature detectors, sometimes using artificial beacons as features.

#### GAUSSIAN NOISE ASSUMPTION

As any EKF algorithm, EKF SLAM makes a *Gaussian noise assumption* for robot motion and perception. The amount of uncertainty in the posterior must be relatively small, since otherwise the linearization in EKFs tend to introduce intolerable errors.

POSITIVE  
INFORMATION

The EKF SLAM algorithm, just like the EKF localizer discussed in Chapter 7.4, can only process *positive* sightings of landmarks. It cannot process negative information that arises from the absence of landmarks in sensor measurements. This is a direct consequence of the Gaussian belief representation and was already discussed in Chapter 7.4.

### 10.2.2 SLAM with Known Correspondence

The SLAM algorithm for the case with known correspondence addresses the continuous portion of the SLAM problem only. Its development is in many ways parallel to the derivation of the EKF localization algorithm in Chapter 7.4, but with one key difference: In addition to estimating the robot pose  $x_t$ , the EKF SLAM algorithm also estimates the coordinates of all landmarks encountered along the way. This makes it necessary to include the landmark coordinates into the state vector.

COMBINED STATE  
VECTOR

For convenience, let us call the state vector comprising robot pose and the map the *combined state vector*, and denote this vector  $y_t$ . The combined vector is given by

$$(10.7) \quad y_t = \begin{pmatrix} x_t \\ m \end{pmatrix} = (x \ y \ \theta \ m_{1,x} \ m_{1,y} \ s_1 \ m_{2,x} \ m_{2,y} \ s_2 \ \dots \ m_{N,x} \ m_{N,y} \ s_N)^T$$

Here  $x$ ,  $y$ , and  $\theta$  denote the robot's coordinates at time  $t$  (not to be confused with the state variables  $x_t$  and  $y_t$ ),  $m_{i,x}, m_{i,y}$  are the coordinates of the  $i$ -th landmark, for  $i = 1, \dots, N$ , and  $s_i$  is its signature. The dimension of this state vector is  $3N + 3$ , where  $N$  denotes the number of landmarks in the map. Clearly, for any reasonable number of  $N$ , this vector is significantly larger than the pose vector that is being estimated in Chapter 7.4, which introduced the EKF localization algorithm. EKF SLAM calculates the online posterior  $p(y_t \mid z_{1:t}, u_{1:t})$ .

The *EKF SLAM algorithm* is depicted in Table 10.1—notice the similarity to the EKF localization algorithm in Table 7.2. Lines 2 through 5 apply the motion update, whereas lines 6 through 20 incorporate the measurement vector.

Lines 3 and 5 manipulate the mean and covariance of the belief in accordance to the motion model. This manipulation only affects those elements of the belief distribution concerned with the robot pose. It leaves all mean and covariance variables for the map unchanged, along with the pose-map covariances. Lines 7 through 20 iterate through all measurements. The test

```

1:  Algorithm EKF_SLAM_known_correspondences( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, c_t$ ):
2:       $F_x = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & \underbrace{0 \cdots 0}_{3N} \end{pmatrix}$ 
3:       $\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} - \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$ 
4:       $G_t = I + F_x^T \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$ 
5:       $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + F_x^T R_t F_x$ 
6:       $Q_t = \begin{pmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\phi^2 & 0 \\ 0 & 0 & \sigma_s^2 \end{pmatrix}$ 
7:      for all observed features  $z_t^i = (r_t^i \ \phi_t^i \ s_t^i)^T$  do
8:           $j = c_t^i$ 
9:          if landmark  $j$  never seen before
10:              $\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \\ \bar{\mu}_{j,s} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \\ s_t^i \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \\ 0 \end{pmatrix}$ 
11:          endif
12:           $\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$ 
13:           $q = \delta^T \delta$ 
14:           $\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ \text{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \\ \bar{\mu}_{j,s} \end{pmatrix}$ 
15:           $F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & \underbrace{0 \cdots 0}_{3j-3} & 0 & 0 & 1 & \underbrace{0 \cdots 0}_{3N-3j} \end{pmatrix}$ 
16:           $H_t^i = \frac{1}{q} \begin{pmatrix} -\sqrt{q} \delta_x & -\sqrt{q} \delta_y & 0 & +\sqrt{q} \delta_x & \sqrt{q} \delta_y & 0 \\ \delta_y & -\delta_x & -q & -\delta_y & +\delta_x & 0 \\ 0 & 0 & 0 & 0 & 0 & q \end{pmatrix} F_{x,j}$ 
17:           $K_t^i = \bar{\Sigma}_t H_t^{iT} (H_t^i \bar{\Sigma}_t H_t^{iT} + Q_t)^{-1}$ 
18:           $\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i)$ 
19:           $\bar{\Sigma}_t = (I - K_t^i H_t^i) \bar{\Sigma}_t$ 
20:      endfor
21:       $\mu_t = \bar{\mu}_t$ 
22:       $\Sigma_t = \bar{\Sigma}_t$ 
23:      return  $\mu_t, \Sigma_t$ 

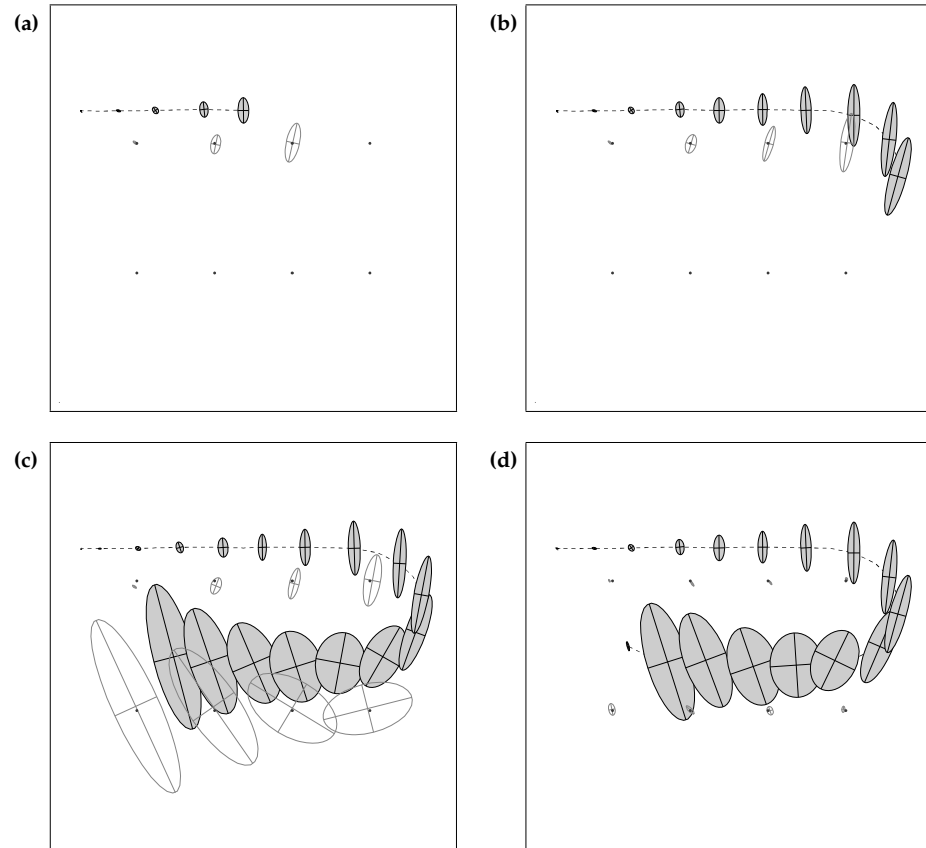
```

**Table 10.1** The EKF algorithm for the SLAM problem (with known correspondences).

in line 9 returns true only for landmarks for which we have no initial location estimate. For those, line 10 initializes the location of such a landmark by the projected location obtained from the corresponding range and bearing measurement. As we shall discuss below, this step is important for the linearization in EKFs; it would not be needed in linear Kalman filters. For each measurement, an “expected” measurement is computed in line 14, and the corresponding *Kalman gain* is computed in line 17. Notice that the Kalman gain is a matrix of size 3 by  $3N + 3$ . This matrix is usually not sparse. Information is propagated through the entire state estimate. The filter update then occurs in lines 18 and 19, where the innovation is folded back into the robot’s belief.

The fact that the Kalman gain is fully populated for all state variables—and not just the observed landmark and the robot pose—is important. In SLAM, observing a landmark does not just improve the position estimate of this very landmark, but that of other landmarks as well. This effect is mediated by the robot pose: Observing a landmark improves the robot pose estimate, and as a result it eliminates some of the uncertainty of landmarks previously seen by the same robot. The amazing effect here is that we do not have to model past poses explicitly—which would put us into the realm of the full SLAM problem and make the EKF a non-realtime algorithm. Instead, this dependence is captured in the Gaussian posterior, more specifically, in the off-diagonal covariance elements of the matrix  $\Sigma_t$ .

Figure 10.3 illustrates the EKF SLAM algorithm for an artificial example. The robot navigates from a start pose that serves as the origin of its coordinate system. As it moves, its own pose uncertainty increases, as indicated by uncertainty ellipses of growing diameter. It also senses nearby landmarks and maps them with an uncertainty that combines the fixed measurement uncertainty with the increasing pose uncertainty. As a result, the uncertainty in the landmark locations grows over time. In fact, it parallels that of the pose uncertainty at the time a landmark is observed. The interesting transition happens in Figure 10.3d: Here the robot observes the landmark it saw in the very beginning of mapping, and whose location is relatively well known. Through this observation, the robot’s pose error is reduced, as indicated in Figure 10.3d—notice the very small error ellipse for the final robot pose! This observation also reduces the uncertainty for other landmarks in the map. This phenomenon arises from a correlation that is expressed in the covariance matrix of the Gaussian posterior. Since most of the uncertainty in earlier landmarks is caused by the robot pose, and since this very uncertainty persists over time, the location estimates of those landmarks are



**Figure 10.3** EKF applied to the online SLAM problem. The robot's path is a dotted line, and its estimates of its own position are shaded ellipses. Eight distinguishable landmarks of unknown location are shown as small dots, and their location estimates are shown as white ellipses. In (a)–(c) the robot's positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In (d) the robot senses the first landmark again, and the uncertainty of *all* landmarks decreases, as does the uncertainty of its current pose. Image courtesy of Michael Montemerlo, Stanford University.

correlated. When gaining information on the robot's pose, this information spreads to previously observed landmarks. This effect is probably the most important characteristic of the SLAM posterior. Information that helps localize the robot is propagated through map, and as a result improves the



localization of other landmarks in the map.

### 10.2.3 Mathematical Derivation of EKF SLAM

The derivation of the EKF SLAM algorithm for the case with known correspondences largely parallels that of the EKF localizer in Chapter 7.4. The key difference is the augmented state vector, which now includes the locations of all landmarks in addition to the robot pose.

In SLAM, the initial pose is taken to be the origin of the coordinate system. This definition is somewhat arbitrary, in that it can be replaced by any coordinate. None of the landmark locations are known initially. The following initial mean and covariance express this belief:

$$(10.8) \quad \mu_0 = (0 \ 0 \ 0 \ \dots \ 0)^T$$

$$(10.9) \quad \Sigma_0 = \begin{pmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \infty & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \infty \end{pmatrix}$$

The covariance matrix is of size  $(3N + 3) \times (3N + 3)$ . It is composed of a small  $3 \times 3$  matrix of zeros for the robot pose variables. All other covariance values are infinite.

As the robot moves, the state vector changes according to the standard noise-free velocity model (see Equations (5.13) and (7.4)). In SLAM, this motion model is extended to the augmented state vector:

$$(10.10) \quad y_t = y_{t-1} + \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t + \gamma_t \Delta t \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

The variables  $x$ ,  $y$ , and  $\theta$  denote the robot pose in  $y_{t-1}$ . Because the motion only affects the robot's pose and all landmarks remain where they are, only the first three elements in the update are non-zero. This enables us to write

the same equation more compactly:

$$(10.11) \quad y_t = y_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t + \gamma_t \Delta t \end{pmatrix}$$

Here  $F_x$  is a matrix that maps the 3-dimensional state vector into a vector of dimension  $3N + 3$ .

$$(10.12) \quad F_x = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & \underbrace{0 \dots 0}_{3N \text{ columns}} \end{pmatrix}$$

The full motion model with noise is then as follows

$$(10.13) \quad y_t = \underbrace{y_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \theta + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \theta - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}}_{g(u_t, y_{t-1})} + \mathcal{N}(0, F_x^T R_t F_x)$$

where  $F_x^T R_t F_x$  extends the covariance matrix to the dimension of the full state vector squared.

As usual in EKF, the motion function  $g$  is approximated using a first degree Taylor expansion

$$(10.14) \quad g(u_t, y_{t-1}) \approx g(u_t, \mu_{t-1}) + G_t (y_{t-1} - \mu_{t-1})$$

where the Jacobian  $G_t = g'(u_t, \mu_{t-1})$  is the derivative of  $g$  with respect to  $y_{t-1}$  at  $u_t$  and  $\mu_{t-1}$ , as in Equation (7.7).

Obviously, the additive form in (10.13) enables us to decompose this Jacobian into an identity matrix of dimension  $(3N + 3) \times (3N + 3)$  (the derivative of  $y_{t-1}$ ) plus a low-dimensional Jacobian  $g_t$  that characterizes the change of the robot pose:

$$(10.15) \quad G_t = I + F_x^T g_t F_x$$

with

$$(10.16) \quad g_t = \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix}$$

Plugging these approximations into the standard EKF algorithm gives us lines 2 through 5 of Table 10.1. Obviously, several of the matrices multiplied

in line 5 are sparse, which should be exploited when implementing this algorithm. The result of this update are the mean  $\bar{\mu}_t$  and the covariance  $\bar{\Sigma}_t$  of the estimate at time  $t$  after updating the filter with the control  $u_t$ , but before integrating the measurement  $z_t$ .

The derivation of the measurement update is similar to the one in Chapter 7.4. In particular, we are given the following measurement model

$$(10.17) \quad z_t^i = \underbrace{\begin{pmatrix} \sqrt{(m_{j,x} - x)^2 + (m_{j,y} - y)^2} \\ \text{atan2}(m_{j,y} - y, m_{j,x} - x) - \theta \\ m_{j,s} \end{pmatrix}}_{h(y_t, j)} + \mathcal{N}\left(0, \underbrace{\begin{pmatrix} \sigma_r & 0 & 0 \\ 0 & \sigma_\phi & 0 \\ 0 & 0 & \sigma_s \end{pmatrix}}_{Q_t}\right)$$

Here  $x, y$ , and  $\theta$  denotes the pose of the robot,  $i$  is the index of an individual landmark observation in  $z_t$ , and  $j = c_t^i$  is the index of the observed landmark at time  $t$ . The variable  $r$  denotes the range to a landmark,  $\phi$  is the bearing to a landmark, and  $s$  the landmark signature; the terms  $\sigma_r, \sigma_\phi$ , and  $\sigma_s$  are the corresponding measurement noise covariances.

This expression is approximated by the linear function

$$(10.18) \quad h(y_t, j) \approx h(\bar{\mu}_t, j) + H_t^i (y_t - \bar{\mu}_t)$$

Here  $H_t^i$  is the derivative of  $h$  with respect to the full state vector  $y_t$ . Since  $h$  depends only on two elements of that state vector, the robot pose  $x_t$  and the location of the  $j$ -th landmark  $m_j$ , the derivative factors into a low-dimensional Jacobian  $h_t^i$  and a matrix  $F_{x,j}$ , which maps  $h_t^i$  into a matrix of the dimension of the full state vector:

$$(10.19) \quad H_t^i = h_t^i F_{x,j}$$

Here  $h_t^i$  is the Jacobian of the function  $h(y_t, j)$  at  $\bar{\mu}_t$ , calculated with respect to the state variables  $x_t$  and  $m_j$ :

$$(10.20) \quad h_t^i = \begin{pmatrix} \frac{\bar{\mu}_{t,x} - \bar{\mu}_{j,x}}{q_t} & \frac{\bar{\mu}_{t,y} - \bar{\mu}_{j,y}}{q_t} & 0 & \frac{\bar{\mu}_{j,x} - \bar{\mu}_{t,x}}{q_t} & \frac{\bar{\mu}_{j,y} - \bar{\mu}_{t,y}}{q_t} & 0 \\ \frac{\sqrt{q_t}}{q_t} & \frac{\sqrt{q_t}}{q_t} & -1 & \frac{\sqrt{q_t}}{q_t} & \frac{\sqrt{q_t}}{q_t} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The scalar  $q_t = (\bar{\mu}_{j,x} - \bar{\mu}_{t,x})^2 + (\bar{\mu}_{j,y} - \bar{\mu}_{t,y})^2$ , and as before,  $j = c_t^i$  is the landmark that corresponds to the measurement  $z_t^i$ . The matrix  $F_{x,j}$  is of dimension  $6 \times (3N + 3)$ . It maps the low-dimensional matrix  $h_t^i$  into a matrix

of dimension  $3 \times (3N + 3)$ :

$$(10.21) \quad F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & \underbrace{0 \cdots 0}_{3j-3} & 0 & 0 & 1 & \underbrace{0 \cdots 0}_{3N-3j} \end{pmatrix}$$

These expressions make up for the gist of the Kalman gain calculation in lines 8 through 17 in our EKF SLAM algorithm in Table 10.1, with one important extension. When a landmark is observed for the first time, its initial pose estimate in Equation (10.8) leads to a poor linearization. This is because with the default initialization in (10.8), the point about which  $h$  is being linearized is  $(\hat{\mu}_{j,x} \ \hat{\mu}_{j,y} \ \hat{\mu}_{j,s})^T = (0 \ 0 \ 0)^T$ , which is a poor estimator of the actual landmark location. A better landmark estimator is given in line 10 of Table 10.1. Here we initialize the landmark estimate  $(\hat{\mu}_{j,x} \ \hat{\mu}_{j,y} \ \hat{\mu}_{j,s})^T$  with the expected position. This expected position is derived from the expected robot pose and the measurement variables for this landmark

$$(10.22) \quad \begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \\ \bar{\mu}_{j,s} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \\ s_t^i \end{pmatrix} + \begin{pmatrix} r_t^i \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \\ 0 \end{pmatrix}$$

We note that this initialization is only possible because the measurement function  $h$  is bijective. Measurements are two-dimensional, as are landmark locations. In cases where a measurement is of lower dimensionality than the coordinates of a landmark,  $h$  is a true projection and it is impossible to calculate a meaningful expectation for  $(\bar{\mu}_{j,x} \ \bar{\mu}_{j,y} \ \bar{\mu}_{j,s})^T$  from a single measurement only. This is, for example, the case in computer vision implementations of SLAM, since cameras often calculate the angle to a landmark but not the range. SLAM is then usually performed by integrating multiple sightings and applying triangulation to determine an appropriate initial location estimate. In the SLAM literature, such a problem is known as *bearing only SLAM* and will be further discussed in one of the exercises (page 334).

BEARING ONLY SLAM

Finally, we note that the EKF algorithm requires memory that is quadratic in  $N$ , the number of landmarks in the map. Its update time is also quadratic in  $N$ . The quadratic update complexity stems from the matrix multiplications that take place at various locations in the EKF.

1: **Algorithm EKF\_SLAM**( $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, N_{t-1}$ ):

2:  $N_t = N_{t-1}$

$$3: F_x = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$$

$$4: \bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} - \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$$

$$5: G_t = I + F_x^T \begin{pmatrix} 0 & 0 & -\frac{v_t}{\omega_t} \cos \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \cos(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & -\frac{v_t}{\omega_t} \sin \mu_{t-1, \theta} + \frac{v_t}{\omega_t} \sin(\mu_{t-1, \theta} + \omega_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$$

$$6: \bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + F_x^T R_t F_x$$

$$7: Q_t = \begin{pmatrix} \sigma_r & 0 & 0 \\ 0 & \sigma_\phi & 0 \\ 0 & 0 & \sigma_s \end{pmatrix}$$

8: *for all observed features*  $z_t^i = (r_t^i \ \phi_t^i \ s_t^i)^T$  *do*

$$9: \begin{pmatrix} \bar{\mu}_{N_t+1, x} \\ \bar{\mu}_{N_t+1, y} \\ \bar{\mu}_{N_t+1, s} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t, x} \\ \bar{\mu}_{t, y} \\ s_t^i \end{pmatrix} + r_t^i \begin{pmatrix} \cos(\phi_t^i + \bar{\mu}_{t, \theta}) \\ \sin(\phi_t^i + \bar{\mu}_{t, \theta}) \\ 0 \end{pmatrix}$$

10: *for*  $k = 1$  *to*  $N_t+1$  *do*

$$11: \delta_k = \begin{pmatrix} \delta_{k, x} \\ \delta_{k, y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{k, x} - \bar{\mu}_{t, x} \\ \bar{\mu}_{k, y} - \bar{\mu}_{t, y} \end{pmatrix}$$

$$12: q_k = \delta_k^T \delta_k$$

see next page for continuation

*continued from the previous page*

13:	$\hat{z}_t^k = \begin{pmatrix} \sqrt{q_k} \\ \text{atan2}(\delta_{k,y}, \delta_{k,x}) - \bar{\mu}_{t,\theta} \\ \bar{\mu}_{k,s} \end{pmatrix}$
14:	$F_{x,k} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$
15:	$H_t^k = \frac{1}{q_k} \begin{pmatrix} -\sqrt{q_k} \delta_{k,x} & -\sqrt{q_k} \delta_{k,y} & 0 & \sqrt{q_k} \delta_{k,x} & \sqrt{q_k} \delta_{k,y} & 0 \\ \delta_{k,y} & -\delta_{k,x} & -1 & -\delta_{k,y} & \delta_{k,x} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} F_{x,k}$
16:	$\Psi_k = H_t^k \bar{\Sigma}_t [H_t^k]^T + Q_t$
17:	$\pi_k = (z_t^i - \hat{z}_t^k)^T \Psi_k^{-1} (z_t^i - \hat{z}_t^k)$
18:	<i>endfor</i>
19:	$\pi_{N_t+1} = \alpha$
20:	$j(i) = \underset{k}{\text{argmin}} \pi_k$
21:	$N_t = \max\{N_t, j(i)\}$
22:	$K_t^i = \bar{\Sigma}_t [H_t^{j(i)}]^T \Psi_{j(i)}^{-1}$
23:	$\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^{j(i)})$
24:	$\bar{\Sigma}_t = (I - K_t^i H_t^{j(i)}) \bar{\Sigma}_t$
25:	<i>endfor</i>
26:	$\mu_t = \bar{\mu}_t$
27:	$\Sigma_t = \bar{\Sigma}_t$
28:	<i>return</i> $\mu_t, \Sigma_t$

**Table 10.2** The EKF SLAM algorithm with ML correspondences, shown here with outlier rejection.

## 10.3 EKF SLAM with Unknown Correspondences

### 10.3.1 The General EKF SLAM Algorithm

#### MAXIMUM LIKELIHOOD CORRESPONDENCE

The EKF SLAM algorithm with known correspondences is now extended into the general EKF SLAM algorithm, which uses an incremental *maximum likelihood* (ML) estimator to determine correspondences. Table 10.2 depicts the algorithm for unknown correspondences.

Since the correspondence is unknown, the input to the algorithm **EKF\_SLAM** lacks a correspondence variable  $c_t$ . Instead, it includes the momentary size of the map,  $N_{t-1}$ . The motion update in lines 3 through 6 is equivalent to the one in **EKF\_SLAM\_known\_correspondences** in Table 10.1. The measurement update loop, however, is different. Starting in line 8, it first creates the hypothesis of a new landmark with index  $N_t + 1$ ; this index is one larger than the landmarks in the map at this point in time. The new landmark's location is initialized in line 9, by calculating its expected location given the estimate of the robot pose and the range and bearing in the measurement. Line 9 also assigns the observed signature value to this new landmark. Next, various update quantities are then computed in lines 10 through 18 for all  $N_t + 1$  possible landmarks, including the “new” landmark. Line 19 sets the threshold for the creation of a new landmark: A new landmark is created if the Mahalanobis distance to all existing landmarks in the map exceeds the value  $\alpha$ . The ML correspondence is then selected in line 20. If the measurement is associated with a previously unseen landmark, the landmark counter is incremented in line 21, and various vectors and matrices are enlarged accordingly—this somewhat tedious step is not made explicit in Table 10.2. The update of the EKF finally takes place in lines 23 and 24. The algorithm **EKF\_SLAM** returns the new number of landmarks  $N_t$  along with the mean  $\mu_t$  and the covariance  $\Sigma_t$ .

The derivation of this EKF SLAM follows directly from previous derivations. In particular, the initialization in line 9 is identical to the one in line 10 in **EKF\_SLAM\_known\_correspondences**, Table 10.1. Lines 10 through 18 parallel lines 12 through 17 in **EKF\_SLAM\_known\_correspondences**, with the added variable  $\pi_k$  needed for calculating the ML correspondence. The selection of the ML correspondence in line 20, and the definition of the Mahalanobis distance in line 17, is analogous to the ML correspondence discussed in Chapter 7.5; in particular, the algorithm **EKF\_localization** in Table 7.3 on page 217 used an analogous equation to determine the most likely landmark (line 16). The measurement updates in lines 23 and 24 of Table 10.2 are also

analogous to those in the EKF algorithm with known correspondences, assuming that the participating vectors and matrices are of the appropriate dimension in case the map has just been extended.

Our example implementation of **EKF\_SLAM** can be made more efficient by restricting the landmarks considered in lines 10 through 18 to those that are near the robot. Further, many of the values and matrices calculated in this inner loop can safely be cached away when looping through more than one feature measurement vector  $z_t^i$ . In practice, a good management of features in the map and a tight optimization of this loop can greatly reduce the running time.

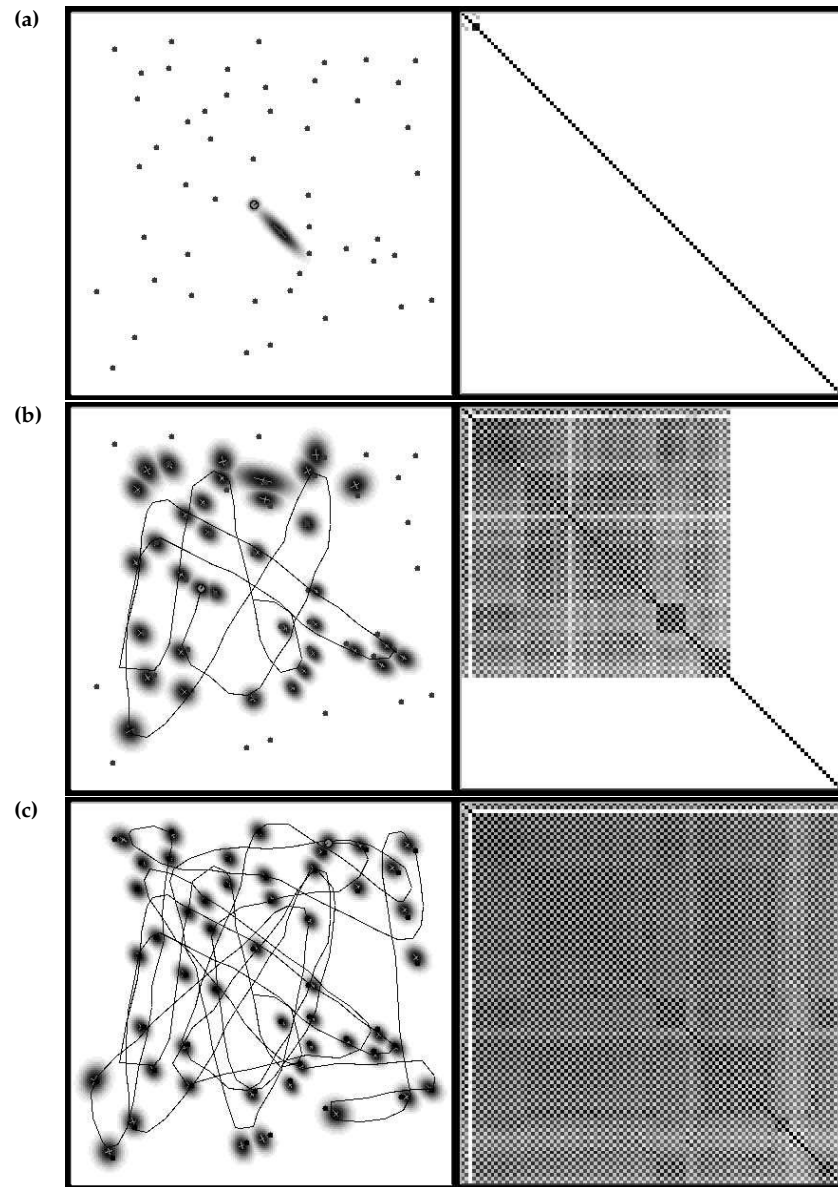
### 10.3.2 Examples

Figure 10.4 shows the EKF SLAM algorithm—here with known correspondence—applied in simulation. The left panel of each of the three diagrams plots the posterior distributions, marginalized for the individual landmarks and the robot pose. The right side depicts the correlation matrix for the augmented state vector  $y_t$ ; the correlation is the normalized covariance. As is easily seen from the result in Figure 10.4c, over time all  $x$ - and  $y$ -coordinate estimates become fully correlated. This means the map becomes known in relative terms, up to an uncertain global location that cannot be reconciled. This highlights an important characteristic of the SLAM problem: The absolute coordinates of a map relative to the coordinate system defined by the initial robot pose can only be determined in approximation, whereas the relative coordinates can be determined asymptotically with certainty.

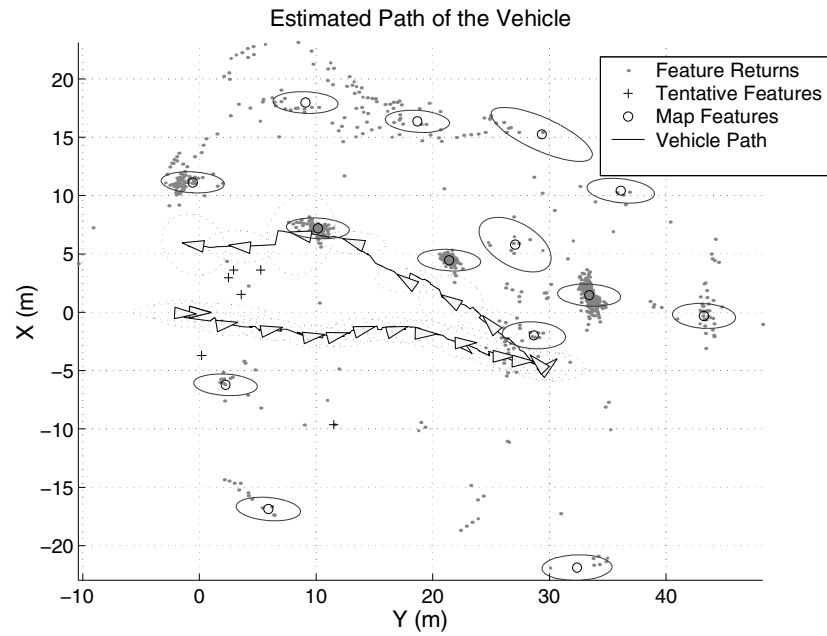
In practice, EKF SLAM has been applied successfully to a large range of navigation problems, involving airborne, underwater, indoor, and various other vehicles. Figure 10.5 shows an example result obtained using the underwater robot Oberon, developed at the University of Sydney, Australia, and shown in Figure 10.6. This vehicle is equipped with a pencil sonar, a sonar that can scan at very high resolutions and detect obstacles up to 50 meters away. To facilitate the mapping problem, researchers have deposited long, small vertical objects in the water, which can be extracted from the sonar scans with relative ease. In this specific experiment, there is a row of such objects, spaced approximately 10 meters apart. In addition, a more distant cliff offers additional point features that can be detected using the pencil sonar.

In the experiment shown in Figure 10.5, the robot moves by these land-





**Figure 10.4** EKF SLAM with known data association in a simulated environment. The map is shown on the left, with the gray-level corresponding to the uncertainty of each landmark. The matrix on the right is the correlation matrix, which is the normalized covariance matrix of the posterior estimate. After some time, all  $x$ - and all  $y$ -coordinate estimates become fully correlated.



**Figure 10.5** Example of Kalman filter estimation of the map and the vehicle pose. Image courtesy of Stefan Williams and Hugh Durrant-Whyte, Australian Centre for Field Robotics.

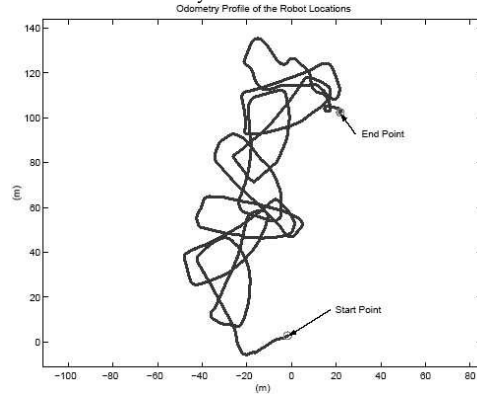


**Figure 10.6** Underwater vehicle Oberon, developed at the University of Sydney. Image courtesy of Stefan Williams and Hugh Durrant-Whyte, Australian Centre for Field Robotics.

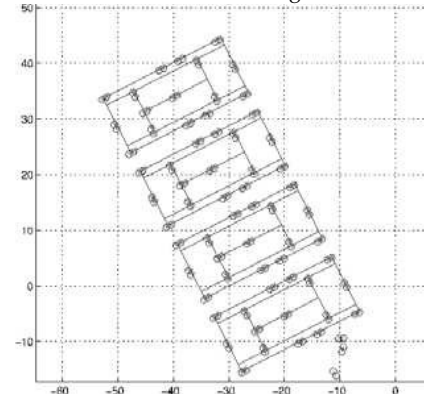
(a) RWI B21 Mobile robot and testing environment



(b) Raw odometry



(c) Result of EKF SLAM with ground truth



**Figure 10.7** (a) The MIT B21 mobile robot in a calibrated testing facility. (b) Raw odometry of the robot, as it is manually driven through the environment. (c) The result of EKF SLAM is a highly accurate map. The image shows the estimated map overlaid on a manually constructed map. All images and results are courtesy of John Leonard and Matthew Walter, MIT.

marks, then turns around and moves back. While doing so, it measures and integrates landmarks into its map using the EKF SLAM algorithm described in this chapter.

The map shown in Figure 10.5 shows the robot's path, marked by the triangles connected by a line. Around each triangle one can see an ellipse, which corresponds to the covariance matrix of the Kalman filter estimate projected into the robot's  $x$ - $y$  position. The ellipse shows the variance; the larger it is, the less certain the robot is about its current pose. Various small dots in

Figure 10.5 show landmark sightings, obtained by searching the sonar scan for small and highly reflective objects. The majority of these sightings are rejected, using a mechanism described in the next section. However, some are believed to correspond to a landmark and are added to the map. At the end of the run, the robot has classified 14 such objects as landmarks, each of which is plotted with the projected uncertainty ellipse in Figure 10.5. These landmarks include the artificial landmarks put out by the researchers, but they also include various other terrain features in the vicinity of the robot. The residual pose uncertainty is small.

Figure 10.7 shows the result of another EKF SLAM implementation. Panel (a) shows MIT's RWI B21 mobile robot, situated in a testing environment. The testing environment is a tennis court; obstacles are hurdles whose position was measured manually with centimeter accuracy for evaluation purposes. Panel (b) of Figure 10.7 shows the raw odometry path. The result of EKF SLAM is shown in Panel (c), overlaid with the manually constructed map. The reader should verify that this is indeed an accurate map.

### 10.3.3 Feature Selection and Map Management

Making EKF SLAM robust in practice often requires additional techniques for *map management*. Many of them pertain to the fact that the Gaussian noise assumption is unrealistic, and many spurious measurements occur in the far tail end of the noise distribution. Such spurious measurements can cause the creation of fake landmarks in the map which, in turn, negatively affect the localization of the robot.

#### OUTLIERS

Many state-of-the-art techniques possess mechanisms to deal with *outliers* in the measurement space. Such outliers are defined as spurious landmark sightings outside the uncertainty range of any landmark in the map. The most simple technique to reject such outliers is to maintain a *provisional landmark list*. Instead of augmenting the map by a new landmark once a measurement indicates the existence of a new landmark, such a new landmark is first added to a provisional list of landmarks. This list is just like the map, but landmarks on this list are *not* used to adjust the robot pose (the corresponding gradients in the measurement equations are set to zero). Once a landmark has consistently been observed and its uncertainty ellipse has shrunk, it is transitioned into the regular map.

#### PROVISIONAL LANDMARK LIST

In practical implementations, this mechanism tends to reduce the number of landmarks in the map by a significant factor, while still retaining all physical landmarks with high probability. A further step, also commonly found

LANDMARK EXISTENCE  
PROBABILITY

in state-of-the-art implementations, is to maintain a *landmark existence probability*. Such a posterior probability may be implemented as log odds ratio and be denoted  $o_j$  for the  $j$ -th landmark in the map. Whenever the  $j$ -th landmark  $m_j$  is observed,  $o_j$  is incremented by a fixed value. Not observing  $m_j$  when it would be in the perceptual range of the robot's sensors leads to a decrement of  $o_j$ . Since it can never be known with certainty whether a landmark is within a robot's perceptual range, the decrement may factor in the probability of such an event. Landmarks are removed from the map when the value  $o_j$  drops below a threshold. Such techniques lead to much leaner maps in the face of non-Gaussian measurement noise.

NUMERICAL  
INSTABILITY OF EKF  
SLAM

When initializing the estimate for a new landmark starting with a covariance with very large elements—as suggested in Equation (10.9)—may induce *numerical instabilities*. This is because the very first covariance update step will change this value by several orders of magnitude, too many perhaps for generating a matrix that is still positive definite. A better strategy involves an explicit initialization step for any feature that has not been observed before. In particular, such a step would initialize the covariance  $\Sigma_t$  directly with the actual landmark uncertainty, instead of executing line 24 in Table 10.2 (same with the mean in line 23).

As noted previously, the maximum likelihood approach to data association has a clear limitation, which arises from the fact that the maximum likelihood approach deviates from the idea of full posterior estimation in probabilistic robotics. Instead of maintaining a joint posterior over augmented states and data associations, it reduces the data association problem to a deterministic determination, which is treated as if the maximum likelihood association was always correct. This limitation makes EKF brittle with regards to landmark confusion, which in turn can lead to wrong results. In practice, researchers often remedy the problem by choosing one of the following two methods, both of which reduce the chances of confusing landmarks:

- **Spatial arrangement.** The further apart landmarks are, the smaller the chance to accidentally confuse them. It is therefore common practice to choose landmarks that are sufficiently far away from each other so that the probability of confusing one with another becomes small. This introduces an interesting trade-off: a large number of landmarks increases the danger of confusing them. Too few landmarks makes it more difficult to localize the robot, which in turn also increases the chances of confusing landmarks. Little is currently known about the optimal density of landmarks, and researchers often use intuition when selecting specific land-

marks.

- **Signatures.** When selecting appropriate landmarks, it is essential to maximize the perceptual distinctiveness of landmarks. For example, doors might possess different colors, or corridors might have different widths. The resulting signatures are essential for successful SLAM.

With these additions, the EKF SLAM algorithm has indeed been applied successfully to a wide range of practical mapping problems, involving robotic vehicles in the air, on the ground, and underwater.

A key limitation of EKF SLAM lies in the necessity to select appropriate landmarks. By reducing the sensor stream to the presence and absence of landmarks, a lot of sensor data is usually discarded. This results in an information loss relative to a SLAM algorithm that can exploit sensors without extensive pre-filtering. Further, the quadratic update time of the EKF limits this algorithm to relatively scarce maps with less than 1,000 features. In practice, one often seeks maps with  $10^6$  features or more, in which case the EKF ceases to be applicable.

The relatively low dimensionality of the map tends to create a harder data association problem. This is easily verified: When you open your eyes and look at the full room you are in, you probably have no difficulty in recognizing where you are! However, if you are only told the location of a small number of landmarks—e.g., the location of all light sources—the decision is much harder. As a result, data association in EKF SLAM is more difficult than in some of the SLAM algorithms discussed in subsequent chapters, and capable of handling orders of magnitude more features. This culminates into the *fundamental dilemma of EKF SLAM*: While incremental maximum likelihood data association might work well with dense maps with hundreds of millions of features, it tends to be brittle with scarce maps. However, EKFs require sparse maps because of the quadratic update complexity. In subsequent chapters, we will discuss SLAM algorithms that are more efficient and can handle much larger maps. We will also discuss more robust data association techniques. For its many limitations, the value of the EKF SLAM algorithm presented in this chapter is mostly historical.

FUNDAMENTAL  
DILEMMA EKF SLAM

## 10.4 Summary

This chapter described the general SLAM problem and introduced the EKF approach.

- The SLAM problem is defined as a concurrent localization and mapping problem, in which a robot seeks to acquire a map of the environment while simultaneously seeking to localize itself relative to this map.
- The SLAM problem comes in two versions: online and global. Both problems involve the estimation of the map. The online problem seeks to estimate the momentary robot pose, whereas the global problem seeks to determine all poses. Both problems are of equal importance in practice, and have found equal coverage in the literature.
- The EKF SLAM algorithm is arguably the earliest SLAM algorithm. It applies the extended Kalman filter to the online SLAM problem. With known correspondences, the resulting algorithm is incremental. Updates require time quadratic in the number of landmarks in the map.
- When correspondences are unknown, the EKF SLAM algorithm applies an incremental maximum likelihood estimator to the correspondence problem. The resulting algorithm works well when landmarks are sufficiently distinct.
- Additional techniques were discussed for managing maps. Two common strategies for identifying outliers include a provisional list for landmarks that are not yet observed sufficiently often, and a landmark evidence counter that calculates the posterior evidence of the existence of a landmark.
- EKF SLAM has been applied with considerable success in a number of robotic mapping problems. Its main drawback is the need for sufficiently distinct landmarks, and the computational complexity required for updating the filter.

In practice, EKF SLAM has been applied with some success. When landmarks are sufficiently distinct, the approach approximates the posterior well. The advantage of calculating a full posterior are manifold: It captures all residual uncertainty and enables the robot to reason about its control taking its true uncertainty into account. However, the EKF SLAM algorithm suffers from its enormous update complexity, and the limitation to sparse maps. This, in turn, makes the data association problem a difficult one, and EKF SLAM tends to work poorly in situations where landmarks are highly ambiguous. Further brittleness is due to the fact that the EKF SLAM algorithm relies on an incremental maximum likelihood data association tech-

nique. This technique makes it impossible to revise past data associations, and can induce failure when the ML data association is incorrect.

The EKF SLAM algorithm applies to the online SLAM problem; it is inapplicable to the full SLAM problem. In the full SLAM problem, the addition of a new pose to the state vector at each time step would make both the state vector and the covariance grow without bounds. Updating the covariance would therefore require an ever-increasing amount of time, and the approach would quickly run out of computational time no matter how fast the processor.

## 10.5 Bibliographical Remarks

The problem of SLAM predates the invention of modern robots by many centuries. The problem of modeling a physical structure from a moving sensor platform is at the core of a number of fields, such as *geosciences*, *photogrammetry*, and *computer vision*. Many of the mathematical techniques that form the core of the SLAM work today were first developed for calculating planetary orbits. For example, the least squares method can be traced back to Johann Carl Friedrich Gauss (1809). SLAM is essentially a geographic surveying problem. Teleported to a robot it creates challenges that human surveyors rarely face, such as the correspondence problem and the problem of finding appropriate features.

In robotics, the EKF to the SLAM problem was introduced through a series of seminal papers by Cheeseman and Smith (1986); Smith and Cheeseman (1986); Smith et al. (1990). These papers were the first to describe the EKF approach discussed in this chapter. Just like us in this book, Smith et al. discussed the EKF in the context of feature-based mapping with point landmarks, and known data association. The first implementations of EKF SLAM were due to Moutarlier and Chatila (1989a,b) and Leonard and Durrant-Whyte (1991), some using artificial beacons as landmarks. The EKF became fashionable at a time when many authors investigated alternative techniques for maintaining accurate pose estimates during mapping (Cox 1991). Early work by Dickmanns and Graefe (1988) on estimation road curvature in autonomous cars is highly related; see (Dickmanns 2002) for a survey.

SLAM is a highly active field of research, as a recent workshop indicates (Leonard et al. 2002b). An extensive literature for the field of SLAM—or CML for *concurrent mapping and localization* as Leonard and Durrant-Whyte (1991) call it—can be found in Thrun (2002). The importance of maintaining correlations in the map was pointed out by Csorba (1997), who in his Ph.D. thesis, who also established some basic convergence results. Since then, a number of authors have extended the basic paradigm in many different ways. The feature management techniques described in this chapter are due to Dissanayake et al. (2001, 2002); see also Bailey (2002). Williams et al. (2001) developed the idea of provisional feature lists in SLAM, to reduce the effect of feature detection errors. Feature initialization is discussed in Leonard et al. (2002a), who explicitly maintains an estimate of previous poses to accommodate sensors that provide incomplete data on feature coordinates. A representation that avoids singularities by explicitly factoring “perturbations” out of the posterior was devised by Castellanos et al. (1999), who reports improved numerical stability over the vanilla EKF. Jensfelt et al. (2002) found significant improvements in indoor SLAM when utilizing basic geometric constraints, such as the fact that most walls are parallel or orthogonal. Early work on SLAM with sonars goes back to Rencken



(1993); a state-of-the-art system for SLAM with sonar sensors can be found in Tardós et al. (2002). Castellanos et al. (2004) provided a critical consistency analysis for the EKF. An empirical comparison of multiple algorithms can be found in Vaganay et al. (2004). A few open questions are discussed by Salichs and Moreno (2000). Research on the important data association problem will be reviewed in a later chapter (see page 481).

As noted, a key limitation of the EKF solution to the SLAM problem lies in the quadratic nature of the covariance matrix. This “flaw” has not remained unnoticed. In the past few years, a number of researchers have proposed EKF SLAM algorithms that gain remarkable scalability through decomposing the map into submaps, for which covariances are maintained separately. Some of the original work in this field is by Leonard and Feder (1999), Guivant and Nebot (2001), and Williams (2001). Leonard and Feder’s (1999) *decoupled stochastic mapping* algorithm decomposes the map into collections of smaller, more manageable submaps. This approach is computationally efficient, but does not provide a mechanism to propagate information through the network of local maps (Leonard and Feder 2001). Guivant and Nebot (2001, 2002), in contrast, provided an approximate factorization of the covariance matrix which reduced the actual complexity of EKF updating by a significant factor. Williams (2001) and Williams et al. (2002) proposed the *constrained local submap filter (CLSF)*, which relies on creating independent local submaps of the features in the immediate vicinity of the vehicle. Williams et al. (2002) provides results for underwater mapping (see Figure 10.5 for some of his early work). The *sequential map joining techniques* described in Tardós et al. (2002) is a related decomposition. Bailey (2002) devises a similar technique for representing SLAM maps hierarchically. Folkesson and Christensen (2003) describes a technique by which frequent updates are limited to a small region near the robot, whereas the remainder of the map is updated at much lower frequencies. All these techniques achieve the same rate of convergence as the full EKF solution, but incur an  $O(n^2)$  computational burden. However, they scale much better to large problems with tens of thousands of features.

A number of researchers have developed hybrid SLAM techniques, which combine EKF-style SLAM techniques with volumetric techniques, such as occupancy grid maps. The *hybrid metric map (HYMM)* by Guivant et al. (2004) and Nieto et al. (2004) decomposes maps into triangular regions (LTRs) using volumetric maps such as occupancy grid maps as a basic representation for those regions. These local maps are combined using EKFs. Burgard et al. (1999b) also decomposes maps into local occupancy grid maps, but uses the *expectation maximization (EM)* algorithm (see Dempster et al. (1977)) for combining local maps into a joint global map. The work by Betgé-Brezetz et al. (1995, 1996) integrated two types of representations into a SLAM framework: bitmaps for representing outdoor terrain, and object representations for sparse outdoor objects.

Extensions of SLAM to dynamic environments can be found in Wang et al. (2003), Hähnel et al. (2003c), and Wolf and Sukhatme (2004). Wang et al. (2003) developed an algorithm called *SLAM with DATMO*, short for SLAM with the detection and tracking of moving objects. Their approach is based on the EKF, but it allows for the possible motion of features. Hähnel et al. (2003c) studied the problem of performing SLAM in environments with many moving objects. They successfully employed the EM algorithm for filtering out measurements that likely correspond to moving objects. By doing so, they were able to acquire maps in environments where conventional SLAM techniques failed. The approach in Wolf and Sukhatme (2004) maintains two coupled occupancy grids of the environment, one for the static map, and one for moving objects. SLAM-style localization is achieved by a regular landmark-based SLAM algorithm.

SLAM systems have been brought to bear in a number of deployed systems. Rikoski et al. (2004) applied SLAM to sonar odometry of submarine, providing a new approach for “auditory odometry.” SLAM in abandoned mines is described in Nüchter et al. (2004), who extended the

paradigm to full 6-D pose estimation. Extensions to the multi-robot SLAM problem have been proposed by a number of researchers. Some of the earlier work is by Nettleton et al. (2000), who developed a technique by which vehicles maintain local EKF maps, but fuse them using the information representation of the posterior. An alternative technique is due to Rekleitis et al. (2001a), who use a team of stationary and moving robots to reduce the localization error when performing SLAM. Fenwick et al. (2002) provides a comprehensive theoretical investigation of multi vehicle map fusion, specifically for landmark based SLAM. Techniques for fusing scans were developed in Konolige et al. (1999); Thrun et al. (2000b); Thrun (2001).

#### STRUCTURE FROM MOTION

A number of researchers have developed SLAM systems for specific sensor types. An important sensor is a camera; however, cameras only provide bearing to features. This problem is well-studied in the computer vision literature as *structure from motion* (SFM) (Tomasi and Kanade 1992; Soatto and Brockett 1998; Dellaert et al. 2003), and in the field of photogrammetry (Konecny 2002). Within SLAM, the seminal work on bearing only SLAM is due to Deans and Hebert (2000, 2002). Their approach recursively estimates features of the environment that are invariant to the robot pose, so as to decouple the pose error from the map error. A great number of researchers has applied SLAM using cameras as the primary sensor (Neira et al. 1997; Cid et al. 2002; Davison 2003). Davison (1998) provides active vision techniques in the context of SLAM. Dudek and Jegessur's (2000) work relies on place recognition based on appearance, whereas Hayet et al. (2002) and Bouguet and Perona (1995) use visual landmarks. Diebel et al. (2004) developed a filter for SLAM with an active stereo sensor that accounts for the nonlinear noise distribution of a stereo range finder. Sensor fusion techniques for SLAM were developed by Devy and Bulata (1996). Castellanos et al. (2001) found empirically that fusing laser and camera outperformed each sensor modality in isolation.

SLAM has also been extended to the problem of building dense 3-D models. Early systems for acquiring 3-D models with indoor mobile robots can be found in Reed and Allen (1997); Iocchi et al. (2000); Thrun et al. (2004b). Devy and Parra (1998) acquire 3-D models using parametric curves. Zhao and Shibasaki (2001), Teller et al. (2001), and Frueh and Zakhor (2003) have developed impressive systems for building large textured 3-D maps of urban environments. Neither of these systems addresses the full SLAM problem due to the availability of outdoor GPS, but they are highly related to the mathematical basis of SLAM. These techniques blend smoothly with a rich body of work on aerial reconstruction of urban environments (Jung and Lacroix 2003; Thrun et al. 2003).

The following chapters discuss alternatives to the plain EKFs. The techniques described there share many of the intuitions with the extensions discussed here, as the boundary between different types of filters has become almost impossible to draw. The literature review will be continued after the next chapter, when discussing a SLAM algorithm using information-theoretic representations.

## 10.6 Exercises

#### BEARING ONLY SLAM

1. What is the computational complexity of the motion update in EKF SLAM? Use the  $O(\ )$  notation. Compare this with the worst case complexity for EKFs over a feature vector of the same size.
2. *Bearing only SLAM* refers to the SLAM problem when the sensors can only measure the bearing of a landmark but not its range. As noted, bearing

only SLAM is closely related to Structure from Motion (SFM) in Computer Vision. One problem in bearing only SLAM with EKF's concerns the initialization of landmark location estimates, even if the correspondences are known. Discuss why, and devise a technique for initializing the landmark location estimates (means and covariances) that can be applied in bearing only SLAM.

3. On page 329, we remarked that the EKF algorithm in Table 10.2 can become numerically instable. Devise a method for setting  $\mu_t$  and  $\Sigma_t$  directly when a new feature is observed for the first time. Such a technique would not require initializing the covariance with very large values. Show that the result is mathematically equivalent to lines 23 and 24 when the covariance is initialized as in Equation (10.9).
4. The text suggests using a binary Bayes filter to compute the probability that a landmark represented in the posterior actually exists in the physical world.
  - (a) In the first part of this exercise, you are asked to design such a binary Bayes filter.
  - (b) Now extend your filter to a situation in which landmarks sporadically disappear with a probability  $p^*$ .
  - (c) Picture a situation where for a well-established landmark, no information is received for a long time with regards to its existence (no positive and no negative information). To what value will your filter converge? Prove your answer.
5. The EKF SLAM algorithm, as presented here, is unable to cope with the data association problem in a sound statistical way. Lay out an algorithm (and a statistical framework) for posterior estimation with unknown data association that represents the posterior by mixtures of Gaussians, and characterize its advantages and disadvantages. How does the complexity of the posterior grow over time?
6. Based on the previous problem, develop an approximate method for posterior estimation with unknown data association where the time needed for each incremental update step does not grow over time (assuming a fixed number of landmarks).
7. Develop a Kalman filter algorithm that uses local occupancy grid maps as its basic components, instead of landmarks. Among other things, prob-

lems that have to be solved are how to relate local grids to each other, and how to deal with the ever-growing number of local grids.