5.8 Autonomous Map Building

All of the localization strategies we have discussed require human effort to install the robot into a space. Artificial environmental modifications may be necessary. Even if this not be case, a map of the environment must be created for the robot. But a robot that localizes successfully has the right sensors for detecting the environment, and so the robot ought to build its own map. This ambition goes to the heart of autonomous mobile robotics. In prose, we can express our eventual goal as follows:

Starting from an arbitrary initial point, a mobile robot should be able to autonomously explore the environment with its on-board sensors, gain knowledge about it, interpret the scene, build an appropriate map, and localize itself relative to this map.

Accomplishing this goal robustly is probably years away, but an important subgoal is the invention of techniques for autonomous creation and modification of an environmental map. Of course a mobile robot's sensors have only a limited range, and so it must physically explore its environment to build such a map. So, the robot must not only create a map but it must do so while moving and localizing to explore the environment. In the robotics community, this is often called the simultaneous localization and mapping (SLAM) problem, arguably the most difficult problem specific to mobile robot systems.

The reason that SLAM is difficult is born precisely from the interaction between the robot's position updates as it localizes and its mapping actions. If a mobile robot updates its position based on an observation of an imprecisely known feature, the resulting position estimate becomes correlated with the feature location estimate. Similarly, the map becomes correlated with the position estimate if an observation taken from an imprecisely known position is used to update or add a feature to the map. The general problem of map-building is thus an example of the chicken-and-egg problem. For localization the robot needs to know where the features are, whereas for map-building the robot needs to know where it is on the map.

The only path to a complete and optimal solution to this joint problem is to consider all the correlations between position estimation and feature location estimation. Such cross-correlated maps are called *stochastic* maps, and we begin with a discussion of the theory behind this approach in the following section [55].

Unfortunately, implementing such an optimal solution is computationally prohibitive. In response a number of researchers have offered other solutions that have functioned well in limited circumstances. Section 5.8.2 characterizes these alternative partial solutions.

5.8.1 The stochastic map technique

Figure 5.38 shows a general schematic incorporating map building and maintenance into the standard localization loop depicted by figure 5.28 during the discussion of Kalman filter

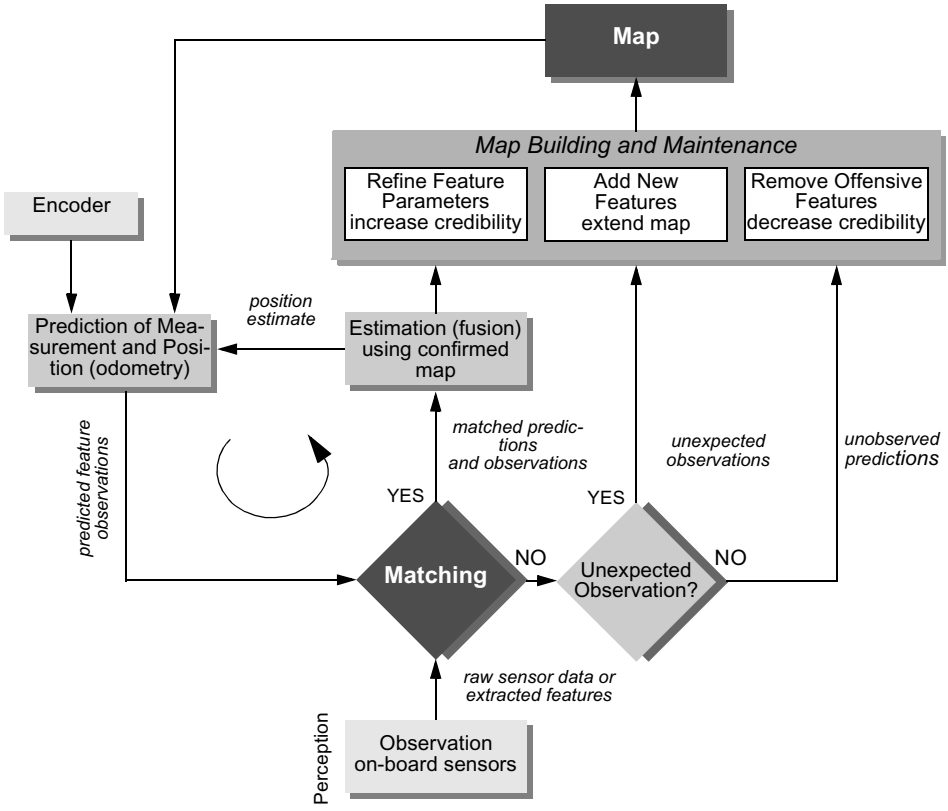

Figure 5.38
General schematic for concurrent localization and map building (see [23]).

localization [23]. The added arcs represent the additional flow of information that occurs when there is an imperfect match between observations and measurement predictions.

Unexpected observations will effect the creation of new features in the map, whereas unobserved measurement predictions will effect the removal of features from the map. As discussed earlier, each specific prediction or observation has an unknown exact value and so it is represented by a distribution. The uncertainties of all of these quantities must be considered throughout this process.

The new type of map we are creating not only has features in it, as did previous maps, but it also has varying degrees of probability that each feature is indeed part of the environment. We represent this new map M with a set n of probabilistic feature locations $\hat{z}_t$, each

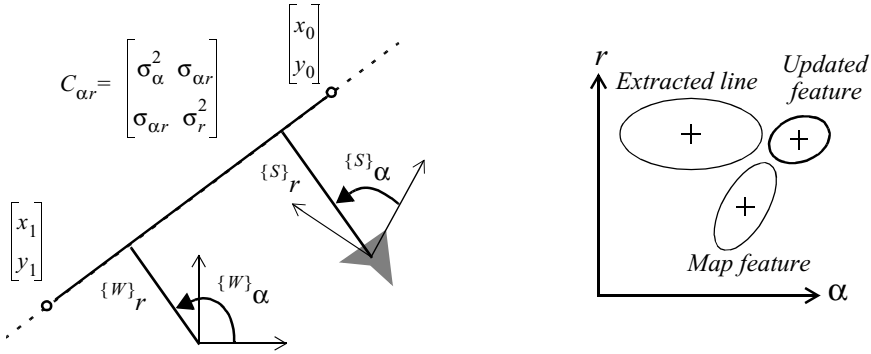

Figure 5.39
Uncertainties in the map.

with the covariance matrix Σ_t and an associated *credibility factor* c_t between 0 and 1 quantifying the belief in the existence of the feature in the environment (see figure 5.39):

$$M = \{\hat{z}_t, \Sigma_t, c_t | (1 \le t \le n)\} \tag{5.69}$$

In contrast to the map used for Kalman filter localization previously, the map M is not assumed to be precisely known because it will be created by an uncertain robot over time. This is why the features $\hat{z}_t$ are described with associated covariance matrices Σ_t.

Just as with Kalman filter localization, the matching step yields has three outcomes in regard to measurement predictions and observations: *matched prediction and observations*, *unexpected observations,* and *unobserved predictions*. Localization, or the position update of the robot, proceeds as before. However, the map is also updated now, using all three outcomes and complete propagation of all the correlated uncertainties (see [23] for more details).

An interesting variable is the credibility factor c_t, which governs the likelihood that the mapped feature is indeed in the environment. How should the robot's failure to match observed features to a particular map feature reduce that map feature's credibility? And also, how should the robot's success at matching a mapped feature increase the chance that the mapped feature is "correct?" In [23] the following function is proposed for calculating credibility:

$$c_t(k) = 1 - e^{-\left(\frac{n_s}{a} - \frac{n_u}{b}\right)} \tag{5.70}$$

where a and b define the learning and forgetting rate and n_s and n_u are the number of matched and unobserved predictions up to time k, respectively. The update of the covariance matrix Σ_t can be done similarly to the position update seen in the previous section. In map-building the feature positions and the robot's position are strongly correlated. This forces us to use a *stochastic map*, in which all cross-correlations must be updated in each cycle [55, 113, 136].

The stochastic map consists of a stacked system state vector:

$$X = \begin{bmatrix} x_r(k) & x_1(k) & x_2(k) & \dots & x_n(k) \end{bmatrix}^T \tag{5.71}$$

and a system state covariance matrix:

$$\Sigma = \begin{bmatrix} C_{rr} & C_{r1} & C_{r2} & \dots & C_{rn} \\ C_{1r} & C_{11} & \dots & \dots & C_{1n} \\ C_{2r} & \dots & \dots & \dots & C_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ C_{nr} & C_{n1} & C_{n2} & \dots & C_{nn} \end{bmatrix} \tag{5.72}$$

where the index *r* stands for the robot and the index $i = 1$ to *n* for the features in the map.

In contrast to localization based on an a priori accurate map, in the case of a stochastic map the cross-correlations must be maintained and updated as the robot is performing automatic map-building. During each localization cycle, the cross-correlations robot-to-feature and feature-to-robot are also updated. In short, this optimal approach requires every value in the map to depend on every other value, and therein lies the reason that such a complete solution to the automatic mapping problem is beyond the reach of even today's computational resources.

5.8.2 Other mapping techniques

The mobile robotics research community has spent significant research effort on the problem of automatic mapping, and has demonstrated working systems in many environments without having solved the complete stochastic map problem described earlier. This field of mobile robotics research is extremely large, and this text will not present a comprehensive survey of the field. Instead, we present below two key considerations associated with automatic mapping, together with brief discussions of the approaches taken by several automatic mapping solutions to overcome these challenges.

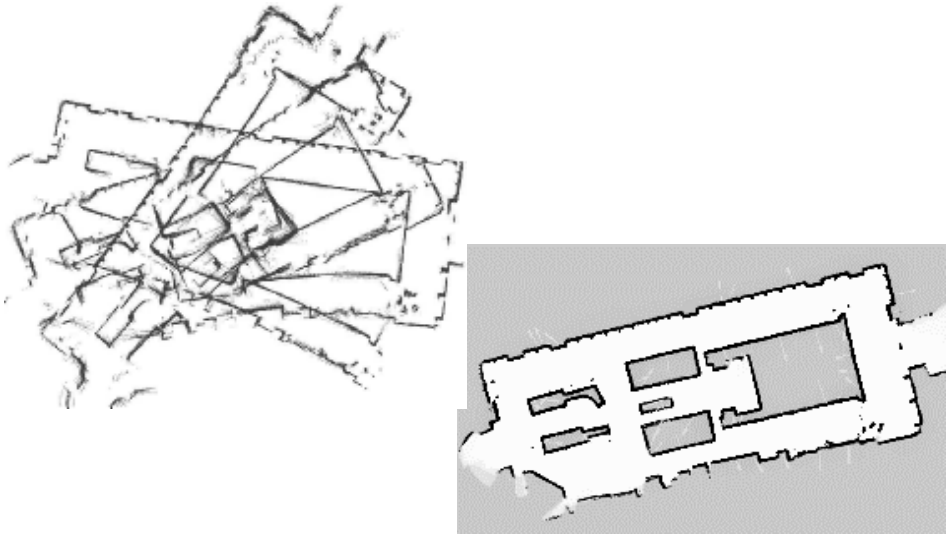

Figure 5.40
Cyclic environments: A naive, local mapping strategy with small local error leads to global maps that have a significant error, as demonstrated by this real-world run on the left. By applying topological correction, the grid map on the right is extracted (courtesy of S. Thrun [142]).

5.8.2.1 Cyclic environments

Possibly the single hardest challenge for automatic mapping to be conquered is to correctly map cyclic environments. The problem is simple: given an environment that has one or more loops or cycles (e.g., four hallways that intersect to form a rectangle), create a globally consistent map for the whole environment.

This problem is hard because of the fundamental behavior of automatic mapping systems: the maps they create are not perfect. And, given any local imperfection, accumulating such imperfections over time can lead to arbitrarily large *global* errors between a map, at the macrolevel, and the real world, as shown in figure 5.40. Such global error is usually irrelevant to mobile robot localization and navigation. After all, a warped map will still serve the robot perfectly well so long as the local error is bounded. However, an extremely large loop still eventually returns to the same spot, and the robot must be able to note this fact in its map. Therefore, global error does indeed matter in the case of cycles.

In some of the earliest work attempting to solve the cyclic environment problem, Kuipers and Byun [94] used a purely topological representation of the environment, reasoning that the topological representation only captures the most abstract, most important

features and avoids a great deal of irrelevant detail. When the robot arrives at a topological node that could be the same as a previously visited and mapped node (e.g., similar distinguishing features), then the robot postulates that it has indeed returned to the same node. To check this hypothesis, the robot explicitly plans and moves to adjacent nodes to see if its perceptual readings are consistent with the cycle hypothesis.

With the recent popularity of metric maps, such as fixed decomposition grid representations, the cycle detection strategy is not as straightforward. Two important features are found in most autonomous mapping systems that claim to solve the cycle detection problem. First, as with many recent systems, these mobile robots tend to accumulate recent perceptual history to create small-scale local *submaps* [51, 74, 157]. Each submap is treated as a single sensor during the robot's position update. The advantage of this approach is twofold. Because odometry is relatively accurate over small distances, the relative registration of features and raw sensor strikes in a local submap will be quite accurate. In addition to this, the robot will have created a virtual sensor system with a significantly larger horizon than its actual sensor system's range. In a sense, this strategy at the very least defers the problem of very large cyclic environments by increasing the map scale that can be handled well by the robot.

The second recent technique for dealing with cycle environments is in fact a return to the topological representation. Some recent automatic mapping systems will attempt to identify cycles by associating a topology with the set of metric submaps, explicitly identifying the loops first at the topological level. In the case of [51], for example, the topological level loop is identified by a human who pushes a button at a known landmark position. In the case of [74], the topological level loop is determined by performing correspondence tests between submaps, postulating that two submaps represent the same place in the environment when the correspondence is good.

One could certainly imagine other augmentations based on known topological methods. For example, the globally unique localization methods described in section 5.7 could be used to identify topological correctness. It is notable that the automatic mapping research of the present has, in many ways, returned to the basic topological correctness question that was at the heart of some of the earliest automatic mapping research in mobile robotics more than a decade ago. Of course, unlike that early work, today's automatic mapping results boast correct cycle detection combined with high-fidelity geometric maps of the environment.

5.8.2.2 Dynamic environments

A second challenge extends not just to existing autonomous mapping solutions but to the basic formulation of the stochastic map approach. All of these strategies tend to assume that the environment is either unchanging or changes in ways that are virtually insignificant. Such assumptions are certainly valid with respect to some environments, such as, for example, the computer science department of a university at 3 AM. However, in a great many

cases this assumption is incorrect. In the case of wide-open spaces that are popular gathering places for humans, there is rapid change in the free space and a vast majority of sensor strikes represent detection of transient humans rather than fixed surfaces such as the perimeter wall. Another class of dynamic environments are spaces such as factory floors and warehouses, where the objects being stored redefine the topology of the pathways on a day-to-day basis as shipments are moved in and out.

In all such dynamic environments, an automatic mapping system should capture the *salient* objects detected by its sensors and, furthermore, the robot should have the flexibility to modify its map as to the positions of these salient objects changes. The subject of *continuous mapping*, or mapping of dynamic environments, is to some degree a direct outgrowth of successful strategies for automatic mapping of unfamiliar environments. For example, in the case of stochastic mapping using the credibility factor c_t mechanism, the credibility equation can continue to provide feedback regarding the probability of the existence of various mapped features after the initial map creation process is ostensibly complete. Thus, a mapping system can become a map-modifying system by simply continuing to operate. This is most effective, of course, if the mapping system is real-time and incremental. If map construction requires off-line global optimization, then the desire to make small-grained, incremental adjustments to the map is more difficult to satisfy.

Earlier we stated that a mapping system should capture only the *salient* objects detected by its sensors. One common argument for handling the detection of, for instance, humans in the environment is that mechanisms such as c_t can take care of all features that did not deserve to be mapped in the first place. For example, in [157] the authors develop a system based on a set of local occupancy grids (called *evidence grids*) and a global occupancy grid. Each time the robot's most recent local evidence grid is used to update a region of the global occupancy grid, extraneous occupied cells in the global occupancy grid are freed if the local occupancy grid detected no objects (with high confidence) at those same positions.

The general solution to the problem of detecting salient features, however, begs a solution to the perception problem in general. When a robot's sensor system can reliably detect the difference between a wall and a human, using, for example, a vision system, then the problem of mapping in dynamic environments will become significantly more straightforward.

We have discussed just two important considerations for automatic mapping. There is still a great deal of research activity focusing on the general map-building and localization problem [22, 23, 55, 63, 80, 134, 147, 156]. However, there are few groups working on the general problem of probabilistic map-building (i.e., stochastic maps) and, so far, a consistent and absolutely general solution is yet to be found. This field is certain to produce significant new results in the next several years, and as the perceptual power of robots improves we expect the payoff to be greatest here.