8 *Kalman Filtering*

HERETOFORE, WE have assumed that the planner has access either to an exact geometric description of its environment, or to a suite of sensors (e.g., sonars) that provide accurate information about the environment. In this chapter, we begin to consider cases for which the robot's knowledge of the world derives from measurements provided by imperfect, noisy sensors.

The Kalman filter is one of the most useful estimation tools available today. Loosely speaking, Kalman filtering provides a recursive method of estimating the state of a dynamical system in the presence of noise [212, 309]. A key feature of the Kalman filter is that it simultaneously maintains estimates of both the state vector ($\hat{x}$) and the estimate error covariance matrix (P), which is equivalent to saying that the output of a Kalman filter is a Gaussian probability density function (PDF) with mean $\hat{x}$ and covariance P. In the context of localization, the Kalman filter output is then a distribution of likely robot positions instead of a single position estimate. As such, the Kalman filter is a specific example of a more general technique known as probabilistic estimation. Some of more general probabilistic estimation techniques are presented in chapter 9.

We begin this chapter by presenting a conceptual overview of probabilistic estimation in section 8.1. Section 8.2 carefully derives the Kalman filter for linear systems. The approach taken here begins with a simplified version of the problem, then gradually adds complexity until the full Kalman filtering equations are reached. Section 8.3 describes the extended Kalman filter (EKF), which is a Kalman filtering variant that can be used on nonlinear systems. Examples that use the EKF for mobile robot localization are presented and discussed. Section 8.4 concludes the chapter by introducing

the problem of simultaneous localization and mapping (SLAM) and showing how it can be solved using a Kalman filter.

8.1 Probabilistic Estimation

In this section, we introduce the concept of probabilistic estimation by considering the fundamental problem of estimating the location of a mobile robot. Probabilistic localization is a probabilistic algorithm: instead of maintaining a single hypothesis as to where in the world a robot might be, probabilistic localization maintains a *probability distribution* over the space of all such hypotheses. The probabilistic representation allows for the uncertainties that arise from uncertain motion models and noisy sensor readings to be accounted for in a principled way. The challenge is then to maintain a position probability density over all possible robot poses. Such a density can have arbitrary forms representing various kinds of information about the robot's position. For example, the robot can start with a uniform distribution representing that it is completely uncertain about its position, i.e., that the robot could be in any location with equal probability. It furthermore can contain multiple modes in the case of ambiguous situations. In the usual case, in which the robot is highly certain about its position, it consists of a unimodal distribution centered around the true position of the robot. This chapter discusses Kalman filtering, which is a form of probabilistic estimation where the estimate is assumed to be a Gaussian (unimodal) PDF. Other probabilistic estimation methods, such as Bayesian methods and particle filtering, can handle more general distributions and are discussed in chapter 9.

One crude method of mobile robot localization is achieved by simply integrating robot velocity commands from a known starting position. When the commands are executed perfectly and the robot starting position is perfectly known, this method gives a perfect estimate of position. Of course, perfect performance and knowledge are impossible to achieve in the real world. Errors between the velocity commands and the actual robot velocities will accumulate over time. In other words, as the robot moves, it is continuously losing information about its location. Eventually, the robot will lose so much information that the command integration estimate becomes meaningless for any practical purpose. A similar approach is to integrate robot velocity measurements reported by onboard odometry sensors. Figure 8.1 shows typical odometry data of a B21 robot as it is recorded with its odometry sensors. Note that the error in odometry quickly accumulates over time up to a rotational error of almost 45 degrees.

With just a few extensions, command integration can be thought of as a probabilistic estimation method. First, consider the robot starting location. Since, in the real world, the starting position cannot be perfectly known, it makes sense to represent the starting

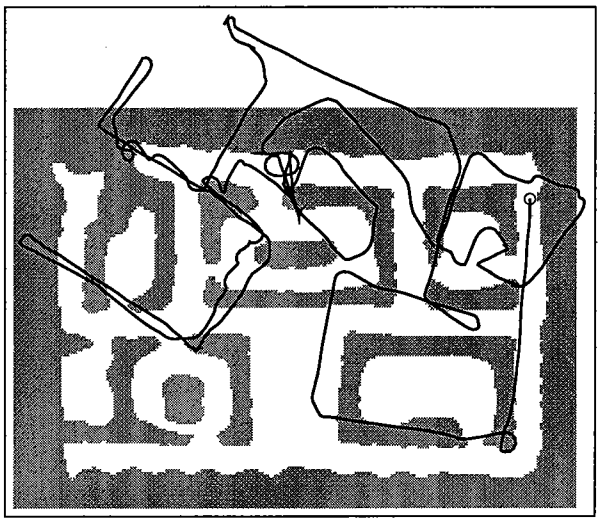

Figure 8.1 Odometry measurements of a B21 robot.

location as a PDF over the state space, i.e., the space of possible robot positions. If a good estimate of the starting position is available, the PDF will have a peak at that location, and the "sharpness" of the peak will represent the certainty of the initial estimate: the more certain the initial estimate, the higher and narrower the peak. Now the challenge is to propagate this PDF as the robot moves. The location of the peak is propagated by integrating the velocity commands just as before. However, since the commands are not executed perfectly, the estimate becomes more uncertain as time progresses and the peak of the PDF will become smaller and more spread out. The rate at which the peak spreads is determined by the amount of error (noise) in the velocity command: the greater the noise, the faster the peak spreads. Eventually, the peak will become so flat that the estimate provided by the PDF is meaningless.

It goes without saying that the goal of probabilistic estimation (or any estimation method, for that matter) is to provide a meaningful estimate of the system state, which in this case is the robot pose. The problem with the command integration method is that information is continually lost and no new information is ever gained. The solution to this is to inject new information into the system through the use of sensors that gather infomation about the environment. Consider, e.g., a robot equipped with a sensor capable of measuring the range and bearing to a landmark with a known location. Such a measurement adds new information to the system and can be used (at least partially) to compensate for the information that was lost by integrating. The new

information can be represented as a PDF in sensor space, usually with a peak at the value of the sensor reading. As we have already described, knowledge about the robot location prior to the sensor measurement is described by a PDF in the state space. The crux of the probabilistic estimation problem is to merge these two distributions in a meaningful way.

Generally, any probabilistic estimation method can be thought of as a two–step process of *prediction* and *update*. Given an estimate of the system state in the form of a PDF, the prediction propagates the PDF according to robot commands together with a motion model for the robot. The update step then corrects the prediction by merging the predicted PDF with information collected by the sensors. The "new" estimate is given by the updated PDF, and the process is iterated. Note that in each iteration, the prediction step accounts for the information lost due to errors in the motion model while the update step incorporates information gained by the sensors.

The following sections describe the well-known Kalman filter, which is a specific probabilistic estimation technique. In Kalman filtering, the motion model is assumed to be a linear function of the state variables and the inputs (commands). The quantities measured by the sensors are assumed to be linear functions of the state variables. Errors in both the motion model and the sensor model are assumed to be zero-mean white Gaussian noise. Because of this simple form, it is possible to derive closed-form equations to perform the prediction and update steps, making Kalman filter implementation a straightforward process.

8.2 Linear Kalman Filtering

One reason that Kalman filtering has become such a popular estimation method is that it is extremely easy to implement for linear systems. The equations in section 8.2.5 can be implemented directly with little understanding of the underlying theory. This feature makes Kalman filtering useful and accessible to a broad range of potential users, but it does not mean that the underlying theory is unimportant. In fact, most modern applications of Kalman filtering employ substantial modifications of the original equations. For example, modifications are necessary to address nonlinear sensor models or non-Gaussian noise models in robot localization and mapping problems. Other modifications are often used to reduce computational complexity.

This section is intended to provide the reader with an understanding of the fundamentals of Kalman filtering for linear systems. The approach taken here is intuitive and uses basic facts from geometry and linear algebra to reconstruct Kalman's equations. Some knowledge of multivariate Gaussian distributions is assumed (see the statistics primer in appendix I for an overview). We begin with a simplified version

of the Kalman filtering problem to illustrate the basic concept, then we incrementally add complexity until we arrive at the full Kalman equations. An example illustrating the application of the Kalman filter equations is presented, and the property of observability in linear systems is introduced. With the understanding provided here, the reader should be able to modify the Kalman filter to fit the needs of a specific estimation problem.

8.2.1 Overview

In order to apply Kalman filtering to the problem of robot localization, it is necessary to define equations that can be used to model the dynamics and sensors of the robot system. The vector x is used to denote the system (robot) state as it evolves through time. This chapter uses discrete time models, meaning that the continuously varying robot state is sampled at discrete, regularly spaced intervals of time to create the sequence $x(k)$, $k \in \{0, 1, 2, \ldots\}$. Specifically, if $x(0)$ is the value of the state at time $t = t_0$, then $x(k)$ is the value of the state at time $t_0 + Tk$, where T is defined to be the sampling time step.

For now we assume that the evolution of the robot state and the values measured by the robot sensors can be modeled as a linear dynamical discrete–time system:

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k) \tag{8.1}$$

$$y(k) = H(k)x(k) + w(k). \tag{8.2}$$

The vector $x(k) \in \mathbb{R}^n$ denotes the full system state. The vector $u(k) \in \mathbb{R}^m$ is used to represent the system input such as velocity commands, torques, or forces intentionally applied to the robot. The vector $y(k) \in \mathbb{R}^p$ is the system output and contains the values reported by the system sensors. The matrix $F(k) \in \mathbb{R}^{n \times n}$ encodes the dynamics of the system, and $G(k) \in \mathbb{R}^{n \times m}$ describes how the inputs drive the dynamics. The vector $v(k) \in \mathbb{R}^n$ is called the *process noise* and is assumed to be white Gaussian noise with zero mean and covariance matrix $V(k)$.[1] The process noise is used to account for unmodeled disturbances (such as slipping wheels) that affect the system dynamics. The matrix $H(k) \in \mathbb{R}^{p \times n}$ describes how state vectors are mapped into outputs. The *measurement noise* vector $w(k) \in \mathbb{R}^p$ is assumed to be white Gaussian noise with zero mean and covariance matrix $W(k)$. Here we assume that $H(k)$ is full row rank for all k, although it may not be square.

1. Here the term "white" means that the vector $v(k)$ is independent of $v(k-1)$ for all k. The properties of a Gaussian distribution, which is defined entirely by its mean vector and covariance matrix, is discussed in more detail later in this chapter.

The objective of Kalman filtering is to determine the "best" estimate of the state x at the kth time step given a previous estimate together with the known input $u(k)$ and output $y(k)$. In order to achieve this there are two separate difficulties that must be overcome. The first is the presence of the unknown and unmeasurable noise vectors $v(k)$ and $w(k)$. Hence, as its name implies, one task of the Kalman filter is to filter out these unwanted disturbances. The second difficulty is that the state in general cannot be directly observed from the outputs because $H(k)$ may not be invertible. This means that the state estimate must be reconstructed using the time history of the known signals $y(k)$ and $u(k)$ together with known parameters $F(k)$, $G(k)$, $H(k)$, $V(k)$, and $W(k)$.[2] A device that does this is called an observer. The Kalman filter is both an observer and a filter.

In this section we build up to Kalman's equations by first building an observer for a system with no measurement noise. Specifically, we derive the equations for a simple two-step observer using only a few simple facts from linear algebra. We then introduce the concept of using a multivariate Gaussian distribution as a state estimate, and we rederive the simple observer equations that use this kind of estimate. This leads naturally to the derivation of the Kalman filter equations for linear discrete time systems.

8.2.2 A Simple Observer

Here we consider a linear discrete time system with no noise:

$$x(k+1) = F(k)x(k) + G(k)u(k) \tag{8.3}$$

$$y(k) = H(k)x(k) \tag{8.4}$$

Here, $H(k)$ is assumed to be full row rank at every k. The objective is to build an observer for this system, i.e., we would like to find a set of equations that allows us to reconstruct the state x. The observer we build will be recursive[3]: it will take the most recent estimate together with the most recent input u and output y, and then return the next estimate. If the observer works (and the assumptions are valid), then the estimate will converge to the actual value of x over time.

Before we begin deriving the necessary equations, we first introduce some notation to make the job of keeping track of the estimate easier. Given two integers k_1 and k_2

2. For this to be possible the pair (F, H) must be observable, a property which is discussed briefly in section 8.2.7. Observability is also discussed in the overview of linear time invariant control systems in appendix J. A more thorough discussion can be found in any good linear systems theory textbook, e.g., [211].

3. Note that the definition of *recursive* is subtly different from what is commonly found in computer science.

with $k_1 \geq k_2$, we use $\hat{x}(k_1 \mid k_2)$ to denote the value of the state estimate at time k_1 given the value of the output at all times up to k_2. The symbol $\hat{x}(k_1 \mid k_2)$ is pronounced "x hat at k-one given k-two." This notation may seem cumbersome at first, but its usefulness will soon become apparent.

Now the observer follows an intuitive two-step process. Given the current state estimate $\hat{x}(k \mid k)$, we first generate a prediction $\hat{x}(k+1 \mid k)$ by propagating the prior estimate according to the system dynamics in equation (8.3). We then correct the prediction based on the output $y(k+1)$ to generate the next estimate $\hat{x}(k+1 \mid k+1)$. We call these two steps the prediction and update steps, respectively.

For the prediction step, we simply substitute $\hat{x}(k \mid k)$ into equation (8.3) to get

$$\hat{x}(k+1 \mid k) = F(k)\hat{x}(k \mid k) + G(k)u(k). \tag{8.5}$$

To perform the update, we first note that given the output $y(k+1)$, the system state is constrained to lie on the hyperplane

$$\Omega = \{x \in \mathbb{R}^n \mid H(k+1)x = y(k+1)\}.$$

Note that Ω is the set of states that are consistent with the measurement $y(k+1)$. For our simple observer, we choose the next estimate $\hat{x}(k+1 \mid k+1)$ to be the point in Ω that has the shortest distance to the prediction $\hat{x}(k+1 \mid k)$. This is an intuitive choice: we have some reason to believe that $\hat{x}(k+1 \mid k)$ is close to the actual state value, and we know that the actual state must be in Ω. So it makes sense to choose the update to be the point in Ω that is closest to $\hat{x}(k+1 \mid k)$. This choice of update is depicted graphically in figure 8.2. We can use algebra to find an expression for $\hat{x}(k+1 \mid k+1)$. Define the vector Δx to be the vector that points from $\hat{x}(k+1 \mid k)$ to $\hat{x}(k+1 \mid k+1)$, i.e.,

$$\Delta x = \hat{x}(k+1 \mid k+1) - \hat{x}(k+1 \mid k).$$

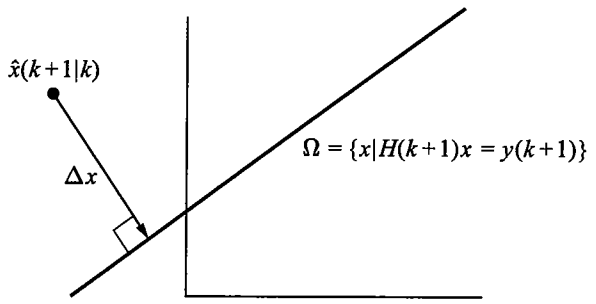

Figure 8.2 The set Ω corresponds to the states consistent with the current output $y(k+1)$. The corrected state lies in this set and is the state closest to the predicted estimate.

By our choice of $\hat{x}(k+1 \mid k+1)$, Δx is the shortest vector pointing from $\hat{x}(k+1 \mid k)$ to Ω. This means that Δx must be *orthogonal* to Ω by the standard inner product on $\mathbb{R}^n$, i.e., we must have $a^T \Delta x = 0$ for any a that is parallel to Ω.[4] Now we need two basic facts from linear algebra [392]:

1. A vector $a \in \mathbb{R}^n$ is parallel to Ω if and only if $H(k+1)a = 0$. The set of all such a is called the *null space* of $H(k+1)$ and is denoted by null$(H(k+1))$.

2. A vector $b \in \mathbb{R}^n$ is orthogonal to every vector in the space null$(H(k+1))$ if and only if b is in the *column space* of $H(k+1)^T$, where the column space of $H(k+1)^T$ is denoted column$(H(k+1)^T)$ and is defined to be the span of the columns of $H(k+1)^T$.

Note that any vector $b \in$ column$(H(k+1)^T)$ can be written as a weighted sum of the columns of $H(k+1)^T$, which is equivalent to saying that $b = H(k+1)^T \gamma$ for some $\gamma \in \mathbb{R}^p$. Combining these two facts, we see that in order to have Δx orthogonal to Ω, we must have

$$\Delta x = H(k+1)^T \gamma$$

for some vector γ in $\mathbb{R}^p$. Next, we will try to find γ.

Define the *innovation error* ν to be the difference between the actual output $y(k+1)$ and the predicted output $H(k+1)\hat{x}(k+1 \mid k)$. In other words, ν is the difference between what the sensors reported and what they would have reported if the prediction was correct. The larger the discrepancy between the actual and predicted measurements, the larger the necessary correction Δx will be. So for now we make the guess that γ can be written as a linear function of ν, i.e., $\gamma = K\nu$ for some $K \in \mathbb{R}^{p \times p}$. This yields the equation

$$\begin{aligned} \Delta x &= H(k+1)^T K \nu \\ &= H(k+1)^T K (y(k+1) - H(k+1)x(k+1 \mid k)). \end{aligned}$$

If we can find a K such that $H(k+1)\,(\hat{x}(k+1 \mid k) + \Delta x)$ agrees with the measurement $y(k+1)$ (i.e., $(\hat{x}(k+1 \mid k) + \Delta x) \in \Omega$), then our guess is correct and we have an expression for $\hat{x}(k+1 \mid k+1)$. To find K, we start with the requirement that

$$H(k+1)\,(\hat{x}(k+1 \mid k) + \Delta x) = y(k+1), \tag{8.6}$$

which implies that

$$H(k+1)\Delta x = y(k+1) - H(k+1)\hat{x}(k+1 \mid k) = \nu.$$

4. Technically, we must also define what we mean by "parallel." We say a vector a is *parallel* to a hyperplane Ω if $x + a \in \Omega$ for every $x \in \Omega$.

Substituting $\Delta x = H(k+1)^T K\nu$ yields

$$H(k+1)H(k+1)^T K\nu = \nu, \tag{8.7}$$

which implies that $K = (H(k+1)H(k+1)^T)^{-1}$. Note that the matrix $H(k+1)H(k+1)^T$ is guaranteed to be invertible by the assumption that $H(k+1)$ is full row rank for all k. We were able to find a K that solves equation (8.7) meaning that for the choice $\Delta x = K\nu$, equation (8.6) is satisfied. Our guess that Δx is a linear function of ν is then verified. As a result, we now have equations that fully express our simple two-step observer:

prediction:

$$\hat{x}(k+1\,|\,k) = F(k)\hat{x}(k\,|\,k) + G(k)u(k)$$

update:

$$\hat{x}(k+1\,|\,k+1) = \hat{x}(k+1\,|\,k) + H^T(HH^T)^{-1}(y(k+1) - H\hat{x}(k+1\,|\,k))$$

Note that in the update equation we have denoted $H(k+1)$ simply by H to keep the expression manageable.

It turns out that there are some problems with this observer. Our choice of the update is naive. Since the update is always perpendicular to the set Ω, only the component of the state that directly affects the current sensor reading is updated. Estimate errors in the direction parallel to Ω are never corrected. As a result, the estimate $\hat{x}$ will not in general converge to x. However, what is important is that the intuitive notions of prediction and correction are the same as those used in the Kalman filter. In the following discussion we follow this intuition toward Kalman's equations, and in the process we fix the problems associated with our simple observer.

8.2.3 Observing with Probability Distributions

The estimate produced by the simple observer discussed in the previous section is a vector. In contrast, the estimate produced by a Kalman filter is a multivariate Gaussian probability distribution over the state space. In addition to providing a vector estimate $\hat{x}(k\,|\,k)$, a Kalman filter also provides an estimate of the error covariance $P(k\,|\,k)$ associated with $\hat{x}(k\,|\,k)$. In this section, we advance the simple observer from the previous section one step toward Kalman's filter by augmenting it to provide a covariance estimate.

First we review some basic facts about multivariate Gaussian distributions. A more detailed discussion can be found in the statistics primer in appendix I. For $x \in \mathbb{R}^n$, a

multivariate Gaussian distribution has a PDF of the form

$$p(x) = \frac{1}{\sqrt{(2\pi)^n |P|}} e^{-\frac{1}{2}(x-\bar{x})^T P^{-1}(x-\bar{x})}, \tag{8.8}$$

where $\bar{x}$ is a vector in $\mathbb{R}^n$ and P is a symmetric, positive definite $n \times n$ matrix. It is clear that $p(x)$ is entirely defined by $\bar{x}$ and P. Further, $E[x] = \bar{x}$ and $E[(x-\bar{x})(x-\bar{x})^T] = P$, so $\bar{x}$ and P are called the *mean vector* and *covariance matrix,* respectively. In the Kalman filter, we maintain a state estimate which will be the mean of a Gaussian distribution, so in the sequel we replace $\bar{x}$ with $\hat{x}$.

In this section, we consider linear discrete time systems with process noise but no measurement noise, i.e.,

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k) \tag{8.9}$$

$$y(k) = H(k)x(k). \tag{8.10}$$

As before, $v(k) \in \mathbb{R}^n$ is assumed to be white noise chosen from a zero-mean Gaussian distribution with covariance matrix $V(k)$ and the matrix $H(k)$ is assumed to be full row rank for all k.

Here we follow the same basic steps of prediction and update that were used for the simple observer. The main difference is that this time we must generate both a state vector estimate $\hat{x}(k \mid k)$ and a covariance matrix estimate $P(k \mid k)$. Hence the prediction step will generate $\hat{x}(k+1 \mid k)$ and $P(k+1 \mid k)$, and the update step will generate the next estimate given by $\hat{x}(k+1 \mid k+1)$ and $P(k+1 \mid k+1)$.

The state vector prediction $\hat{x}(k+1 \mid k)$ is found by substituting $\hat{x}(k \mid k)$ into equation (8.9). Since the expected value of $v(k)$ is zero, the resulting prediction is

$$\hat{x}(k+1 \mid k) = F(k)\hat{x}(k \mid k) + G(k)u(k). \tag{8.11}$$

To compute the predicted covariance matrix we start with the definition of the covariance matrix:

$$P(k+1 \mid k) = E\left[(x(k+1) - \hat{x}(k+1 \mid k))\,(x(k+1) - \hat{x}(k+1 \mid k))^T\right]$$

Substituting $x(k+1)$ from equation (8.9) and $\hat{x}(k+1 \mid k)$ from equation (8.11), then multiplying the terms inside the expectation, yields

$$\begin{aligned} P(k+1 \mid k) = E\big[&F(k)\,(x(k) - \hat{x}(k \mid k))\,(x(k) - \hat{x}(k \mid k))^T F(k)^T \\ &+ 2F(k)\,(x(k) - \hat{x}(k \mid k))\,v(k)^T + v(k)v(k)^T\big]. \end{aligned}$$

The fact that $v(k)$ is independent of both $x(k)$ and $\hat{x}(k \mid k)$ implies that $E[(x(k) - \hat{x}(k \mid k))v(k)] = E[x(k) - \hat{x}(k \mid k)]E[v(k)]$, which is zero due to the fact that $v(k)$ is assumed to be zero mean. Using this fact together with the linearity property of the

expectation yields

$$P(k+1\,|\,k) = F(k)E\left[(x(k)-\hat{x}(k\,|\,k))\,(x(k)-\hat{x}(k\,|\,k))^T\right]F(k)^T + E\left[v(k)v(k)^T\right].$$

The first expectation term in this equation matches the definition of the covariance matrix $P(k\,|\,k)$, while the second expectation term matches the definition of the covariance matrix $V(k)$. As a result we can write the prediction equation

$$P(k+1\,|\,k) = F(k)P(k\,|\,k)F(k)^T + V(k). \tag{8.12}$$

To perform the update step, we choose $\hat{x}(k+1\,|\,k+1)$ to be the most likely point x in the set

$$\Omega = \{x \in \mathbb{R}^n \mid H(k+1)x = y(k+1)\}.$$

Hence, we look for $x \in \Omega$ that maximizes the Gaussian distribution defined by $\hat{x}(k+1\,|\,k)$ and $P(k+1\,|\,k)$, i.e.,

$$p(x) = \frac{1}{\sqrt{(2\pi)^n\,|P(k+1\,|\,k)|}} e^{-\frac{1}{2}(x-\hat{x}(k+1\,|\,k))^T P(k+1\,|\,k)^{-1}(x-\hat{x}(k+1\,|\,k))}.$$

Because the exponential is monotonically increasing, $p(x)$ is maximized when $(x-\hat{x}(k+1\,|\,k))^T P(k+1\,|\,k)^{-1}(x-\hat{x}(k+1\,|\,k))$ is minimized. With this in mind, we introduce a new notion of distance with the norm[5]

$$\|x\|_M^2 = x^T P(k+1\,|\,k)^{-1}x,$$

which is derived from the new inner product on $\mathbb{R}^n$,

$$\langle x_1, x_2\rangle_M = x_1^T P(k+1\,|\,k)^{-1}x_2.$$

Define $\Delta x = \hat{x}(k+1\,|\,k+1) - \hat{x}(k+1\,|\,k)$. So we want to find $\hat{x}(k+1\,|\,k+1)$ such that

1. $\|\Delta x\|_M$ is minimized.

2. $(\hat{x}(k+1\,|\,k) + \Delta x) \in \Omega$.

The first condition means that the vector Δx is orthogonal to the hyperplane Ω with respect to the inner product $\langle\cdot,\cdot\rangle_M$. This notion is depicted graphically in figure 8.3. The ellipses in this figure represent sets of points that are equidistant to $\hat{x}(k+1\,|\,k)$ according to the $\|\cdot\|_M$ norm. With this notion of distance, choosing $\hat{x}(k+1\,|\,k+1)$ to be the closest point on Ω to $\hat{x}(k+1\,|\,k)$ is equivalent to choosing the point at which

5. $d_M(x,\hat{x}) = \|x-\hat{x}\|_M$ is called the *Mahalanobis distance* between x and $\hat{x}$. The Mahalanobis distance indicates how far away the point x is from the mean $\hat{x}$ in units of standard deviations.

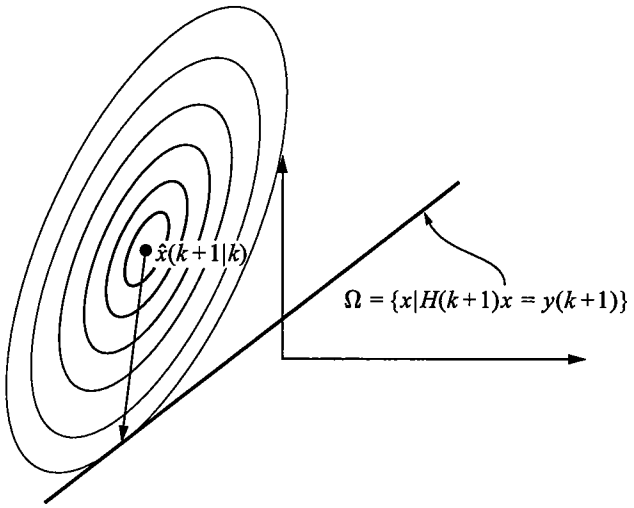

Figure 8.3 Correction determines the "closest" and "most likely" state on the set of states Ω.

one of the equidistant ellipses tangentially intersects Ω. The resulting Δx must be orthogonal to Ω, but our notion of orthogonality is skewed by $P(k+1\,|\,k)^{-1}$. This means that we must have

$$aP(k+1\,|\,k)^{-1}(\Delta x) = 0$$

for all $a \in \text{null}(H(k+1))$. In the remainder of this section, we simply denote $H(k+1)$ by H for brevity. Using the linear algebra facts presented earlier, this expression can only be true if $\Delta x \in \text{column}(P(k+1\,|\,k)H^T)$, which means that

$$\Delta x = P(k+1\,|\,k)H^T\gamma$$

for some $\gamma \in \mathbb{R}^p$. As in the case of the simple observer, we guess that γ can be expressed as a linear function of the innovation error $v = y(k+1) - Hx(k+1\,|\,k)$, i.e.,

$$\Delta x = P(k+1\,|\,k)H^T K v \tag{8.13}$$

for some $K \in \mathbb{R}^{p\times p}$. Now we enforce $(\hat{x}(k+1\,|\,k) + \Delta x) \in \Omega$, i.e.,

$$H\,(\hat{x}(k+1\,|\,k) + \Delta x) = y(k+1),$$

which implies that $H\Delta x = v$. Substituting for Δx from equation (8.13) yields

$$HP(k+1\,|\,k)H^T K v = v,$$

which means that we must have

$$K = \left(HP(k+1\,|\,k)H^T\right)^{-1}.$$

The resulting update equation for the state vector estimate is

$$\hat{x}(k+1\,|\,k+1) = \hat{x}(k+1\,|\,k) + P(k+1\,|\,k)H^T\left(HP(k+1\,|\,k)H^T\right)^{-1}\nu. \tag{8.14}$$

To ease notation, we define

$$R = P(k+1\,|\,k)H^T\left(HP(k+1\,|\,k)H^T\right)^{-1}$$

so that the update equation can be written simply

$$\hat{x}(k+1\,|\,k+1) = \hat{x}(k+1\,|\,k) + R\nu.$$

To find the update equation for the covariance matrix estimate, we use the definition of the covariance matrix together with the update equation for the state vector estimate to get

$$P(k+1\,|\,k+1) = P(k+1\,|\,k) - RHP(k+1\,|\,k). \tag{8.15}$$

The details of this derivation are the subject of problem 7.

Summarizing the observer derived in this section:

prediction:

$$\hat{x}(k+1\,|\,k) = F(k)\hat{x}(k\,|\,k) + G(k)u(k) \tag{8.16}$$

$$P(k+1\,|\,k) = F(k)P(k\,|\,k)F(k)^T + V(k) \tag{8.17}$$

update:

$$\hat{x}(k+1\,|\,k+1) = \hat{x}(k+1\,|\,k) + R\nu \tag{8.18}$$

$$P(k+1\,|\,k+1) = P(k+1\,|\,k) - RHP(k+1\,|\,k) \tag{8.19}$$

where

$$\nu = y(k+1) - Hx(k+1\,|\,k) \tag{8.20}$$

$$R = P(k+1\,|\,k)H^T\left(HP(k+1\,|\,k)H^T\right)^{-1} \tag{8.21}$$

and H is shorthand for $H(k+1)$.

As in the case of the simple observer, this observer also has some problems. Because we assumed no sensor noise, the update equations will cause the covariance matrix estimate to become singular. This makes sense: noiseless measurements mean that the uncertainty in the directions associated with the sensor measurements will be zero. But the singular covariance makes the resulting notions of Gaussian distribution and Mahalanobis distance meaningless since they rely on the inverse of P. Still, the intuition behind using and propagating a Gaussian distribution as a state estimate is in line

with the intuition behind the Kalman filter in spite of this problem. In the next section, we advance this intuition one final step to derive the full Kalman filter equations.

8.2.4 The Kalman Filter

Consider the system described at the beginning of this chapter:

$$x(k+1) = F(k)x(k) + G(k)u(k) + v(k) \tag{8.22}$$

$$y(k) = H(k)x(k) + w(k) \tag{8.23}$$

The only difference between this system and the system in the previous section is that we have included the sensor noise term $w(k)$, a zero-mean white Gaussian random vector with covariance matrix $W(k)$.

Since the dynamic equation has not changed, the prediction step for the Kalman filter is identical to the prediction step for the observer defined in section 8.2.3. The addition of noise to the sensor equation significantly changes the update step, however. In the previous case, the output $y(k+1)$ constrained the next estimate $\hat{x}(k+1\,|\,k+1)$ to lie in the hyperplane Ω. We knew exactly what the output $y(k+1)$ had to be, and we chose $\hat{x}(k+1\,|\,k+1)$ to match it. As a result, we could use the algebraic equation $y(k+1) = H(k+1)\hat{x}(k+1\,|\,k+1)$ to find $\hat{x}(k+1\,|\,k+1)$. In the current case, there is no such algebraic constraint. We do not know exactly what the output should be; we only know that it is drawn from a Gaussian distribution in $\mathbb{R}^p$ with mean $y(k+1)$ and covariance matrix $W(k)$. Without this constraint, we cannot use the same algebraic approach to define $\hat{x}(k+1\,|\,k+1)$. Instead, we will first look for the most likely output y^* given the prediction $(\hat{x}(k+1\,|\,k), P(k+1\,|\,k))$ together with the measured output $y(k+1)$. Once we have y^*, we can introduce the algebraic constraint $y^* = H(k+1)\hat{x}(k+1\,|\,k+1)$ and proceed as before.

We begin to find y^* by projecting the prediction into output space. Using the output map $H(k+1)$ and the definition of covariance, we see that the state space distribution with mean $\hat{x}(k+1\,|\,k)$ and covariance matrix $P(k+1\,|\,k)$ projects into a Gaussian distribution in the output space ($\mathbb{R}^p$) with mean

$$\hat{y}(k+1) = H(k+1)\hat{x}(k+1\,|\,k)$$

and covariance matrix

$$\begin{aligned}\widehat{W} &= E\left[(\hat{y}(k+1) - y(k+1))\,(\hat{y}(k+1) - y(k+1))^T\right] \\ &= E\left[H(k+1)\big(\hat{x}(k+1\,|\,k) - x(k+1)\big)\big(\hat{x}(k+1) \right. \\ &\qquad \left. - x(k+1)\big)^T H(k+1)^T\right] \\ &= H(k+1)P(k+1\,|\,k)H(k+1)^T.\end{aligned}$$

The most likely output y^* is then defined to be the most likely point in the output space $\mathbb{R}^p$ given the Gaussian distribution that results from projecting the prediction and the Gaussian distribution that results from taking the measurement. The projected prediction and output distributions have mean-covariance pairs $(\hat{y}, \widehat{W})$ and $(y(k+1), W(k+1))$, respectively. Since these distributions are independent, y^* will be the peak of the function that results from taking their product. Fortunately the product of two Gaussian distributions is also Gaussian and the result can be obtained using a well-known formula [383]. We summarize the required result as a theorem:

THEOREM 8.2.1 (Product of Gaussians) *The product of two Gaussian distributions with mean-covariance pairs* (z_1, C_1) *and* (z_2, C_2) *is proportional to a third Gaussian with mean vector*

$$z_3 = z_1 + Q(z_2 - z_1)$$

and covariance matrix

$$C_3 = C_1 - QC_1,$$

where

$$Q = C_1(C_1 + C_2)^{-1}.$$

To sketch the proof of this theorem, we use the property that the product of two exponentials is the exponential of the sum of the exponents. A clever reordering of the terms in the resulting sum yields the result. See problem 8 for the details of the proof.

Applying theorem 8.2.1,

$$y^* = \hat{y} + \widehat{W}(k+1)(\widehat{W} + W(k+1))^{-1}(\hat{y} - y(k+1)).$$

Now that we have found the most likely output y^*, we can define $\Omega^* = \{x \in \mathbb{R}^n \,|\, H(k+1)x = y^*\}$ and, as in the previous section, proceed to find the $\Delta x = \hat{x}(k+1\,|\,k+1) - \hat{x}(k+1\,|\,k)$ that minimizes $\|\Delta x\|_M$ while satisfying $\hat{x}(k+1\,|\,k+1) \in \Omega^*$ (see figure 8.4). Using H to denote $H(k+1)$, we get the result

$$\begin{aligned}\Delta x &= P(k+1\,|\,k)H^T\left(HP(k+1\,|\,k)H^T\right)^{-1} \\ &\qquad (y^* - Hx(k+1\,|\,k)) \\ &= P(k+1\,|\,k)H^T\left(HP(k+1\,|\,k)H^T + W(k)\right)^{-1}\nu,\end{aligned}$$

where, as before, $\nu = y(k+1) - Hx(k+1\,|\,k)$ is the innovation error. Defining

$$R = P(k+1\,|\,k)H^T\left(HP(k+1\,|\,k)H^T + W(k+1)\right)^{-1},$$

we can write the state vector estimate update equation

$$\hat{x}(k+1\,|\,k+1) = \hat{x}(k+1\,|\,k) + R\nu. \tag{8.24}$$

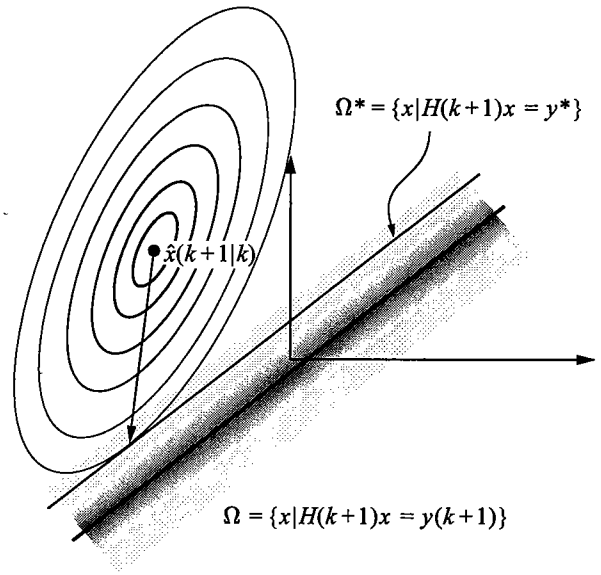

Figure 8.4 The sensor noise distribution is projected into the state space and is an extruded Gaussian centered on the states consistent with the current sensor reading. The most likely output y^* is determined by multiplying the Gaussian distribution that results from the measurement $y(k+1)$ with the Gaussian distribution that results from projecting the prediction into the output space. This then corresponds to a set of states Ω^*. The update is the point on Ω^* that is closest to the prediction $\hat{x}(k+1|k)$ in the sense of Mahalanobis distance.

To find the update equation for the covariance matrix estimate, we use the definition of the covariance matrix together with the update equation for the state vector estimate to get

$$P(k+1|k+1) = P(k+1|k) - RHP(k+1|k). \tag{8.25}$$

8.2.5 Kalman Filter Summary

The Kalman filter equations are summarized as follows:

prediction:

$$\hat{x}(k+1|k) = F(k)\hat{x}(k|k) + G(k)u(k) \tag{8.26}$$

$$P(k+1|k) = F(k)P(k|k)F(k)^T + V(k) \tag{8.27}$$

update:

$$\hat{x}(k+1\,|\,k+1) = \hat{x}(k+1\,|\,k) + Rv \tag{8.28}$$

$$P(k+1\,|\,k+1) = P(k+1\,|\,k) - RH(k+1)P(k+1\,|\,k) \tag{8.29}$$

where

$$v = y(k+1) - H(k+1)x(k+1\,|\,k)) \tag{8.30}$$

$$S = H(k+1)P(k+1\,|\,k)H(k+1)^T + W(k+1) \tag{8.31}$$

$$R = P(k+1\,|\,k)H(k+1)^T S^{-1}. \tag{8.32}$$

These equations provide the optimal estimate of x in the sense that the expected value of the error between $x(k)$ and $\hat{x}(k\,|\,k)$ is minimized at every k. One can view R as the weighting factor that takes into account the relationship between the accuracy of the predicted estimate and the measurement noise. If R is "large," then the sensor readings are more believable than the prediction and the Kalman filter weights the sensor reading highly when computing the updated estimate. If R is "small," then the sensor readings are not as believable and, as a result, they do not have as much influnce in the update step.

In this chapter we have chosen to present the derivation of the Kalman filter equations as an optimization problem because we believe that to be an intuitive approach. It is important to note, however, that the state and covariance estimates that result from the use of these equations are not only the "best" estimates, they are also the "correct" estimates. If the estimate at time k is Gaussian and described by $(\hat{x}(k\,|\,k), P(k\,|\,k))$, then the correct distribution at time $k+1$ (i.e., the *posterior* distribution) is in fact also Gaussian and is described by $(\hat{x}(k+1\,|\,k+1), P(k+1\,|\,k+1))$.

If we allow the noise terms $v(k)$ and $w(k)$ to have non-Gaussian distributions, then these equations still provide the best *linear* estimator, but there may be nonlinear estimators that do a better job.

8.2.6 Example: Kalman Filter for Dead Reckoning

In mobile robotics, the term *dead reckoning* typically refers to a position estimate achieved by integrating odometry measurements. Here we present an example of a more sophisticated form of dead reckoning where a Kalman filter is used to fuse the robot commands (inputs) with measurements from odometry sensors.

Consider a mobile robot constrained to move along a straight line. The robot state is defined to be $x = [x_r, v_r]^T$ where x_r and v_r are the robot position and velocity, respectively. The input u is a real–valued force applied to the robot. Newton's law states that $\frac{dv_r}{dt} = \frac{u}{m}$, where m is the mass of the robot. This can be approximated by

the discrete time equation

$$\frac{v_r(k+1) - v_r(k)}{T} = \frac{u(k)}{m},$$

where T is the sampling rate (in seconds) of the discretization. So then the discrete time state equation can be written as

$$\begin{aligned} x(k+1) &= \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ \frac{T}{m} \end{bmatrix} u(k) + v(k) \\ &\triangleq Fx(k) + Gu(k) + v(k), \end{aligned} \tag{8.33}$$

where the process noise term $v(k)$ is used to account for errors that arise from unmodeled sources such as discretization and friction. The vector $v(k)$ is assumed to be zero-mean white Gaussian noise with covariance matrix V.

We assume that the robot is equipped with a sensor that measures velocity. We also assume that the error in this measurement is well modeled as zero-mean white Gaussian noise with known variance W. Then the output $y(k)$ can be written

$$\begin{aligned} y(k+1) &= \begin{bmatrix} 0 & 1 \end{bmatrix} x(k) + w(k) \\ &\triangleq Gx(k) + w(k), \end{aligned} \tag{8.34}$$

where w is the noise term.

Now the Kalman filter can be applied using the sequence of equations listed in section 8.2.5. We simulated this example in MATLAB using the parameters $m = 1$, $W = .5$, $T = 0.5$, and

$$V = \begin{bmatrix} 0.2 & 0.05 \\ 0.05 & 0.1 \end{bmatrix}.$$

Assume that the input at time k is known to be $u(k) = 0$, and assume an initial state estimate of $\hat{x}(k \mid k) = [2,\ 4]^T$ and an initial covariance estimate of

$$P(k \mid k) = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}.$$

Further, assume that the (unknown) value of the actual state is $x(k+1) = [1.8,\ 2]^T$. The sequence of prediction, combining prediction with measurement in output space, and update are depicted graphically in figures 8.5, 8.6, and 8.7. Here, the two-dimensional Gaussian distributions that result from $\hat{x}$ and P are represented by confidence ellipses. Specifically, these ellipses are chosen so that the probability that the actual value of the state x is contained within the ellipse is 0.95.

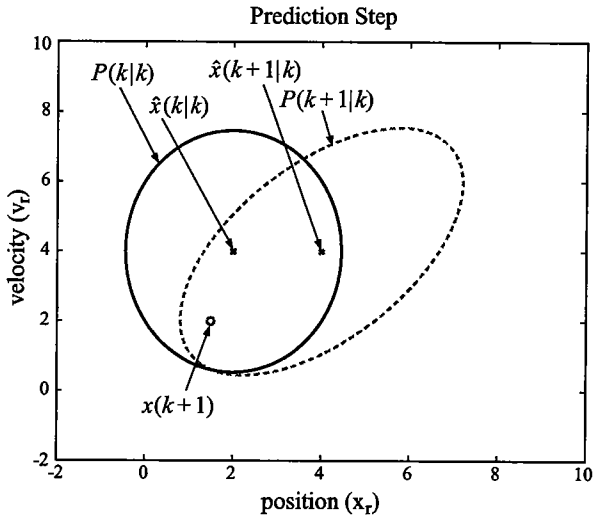

Figure 8.5 The initial estimate $\hat{x}(k \mid k)$ has an uncertainty $P(k \mid k)$ which grows to $P(k+1 \mid k)$ when the robot moves, reflecting the increase in uncertainty. This increase in uncertainty is depicted by plotting the 0.95 confidence ellipses.

8.2.7 Observability in Linear Systems

Note that in the update step of the previous example, the updated ellipse is "squished" significantly in the vertical direction. This squishing corresponds to the information gained from the velocity measurement. For this particular example, each iteration of the Kalman filter will reflect a gain of information in the velocity direction and a loss of information in the position direction. As a result, the expected error on the position estimate will grow monotonically without bound. This failure is not the fault of the Kalman filter, which is guaranteed to provide the best possible estimate. The problem instead lies with the system itself; specifically, the system dynamics and output equations do not interact in a way that allows the state to be recovered from the available outputs. In other words, the system in the example fails to be observable.

Loosely speaking, a system is said to be *observable* if the full state can be reconstructed by observing the input u and the output y over some period of time (see appendix J for a discussion of observability in linear systems.) For linear systems where the system matrices $F(k)$ and $H(k)$ do not vary with k, there is a simple test to determine observability:

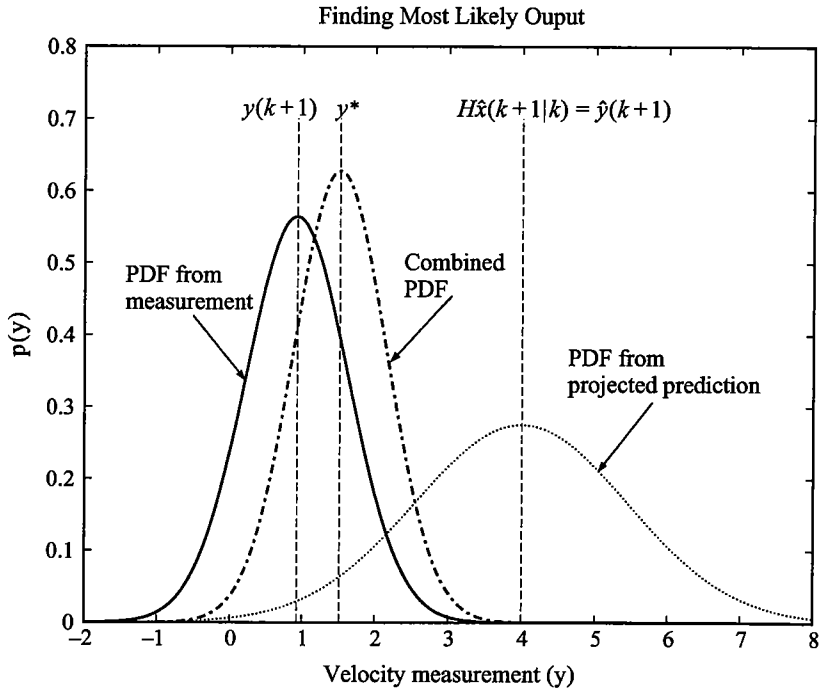

Figure 8.6 Measurements and predictions are then merged. The PDF plotted with the dot-dashed line results from the combination of the measurement PDF and the PDF of the prediction projected into output space, where the combination is computed using theorem 8.2.1. The most likely output y^* is the value at which this combined distribution reaches its peak.

THEOREM 8.2.2 (Observability Test) *The linear time-invariant discrete time system*

$$x(k+1) = Fx(k) + Gu(k) + v(k)$$
$$y(k) = Hx(k) + w(k)$$

is observable if and only if the observability matrix

$$Q = \begin{bmatrix} H \\ HF \\ HF^2 \\ \vdots \\ HF^{(n-1)} \end{bmatrix}$$

has rank n.

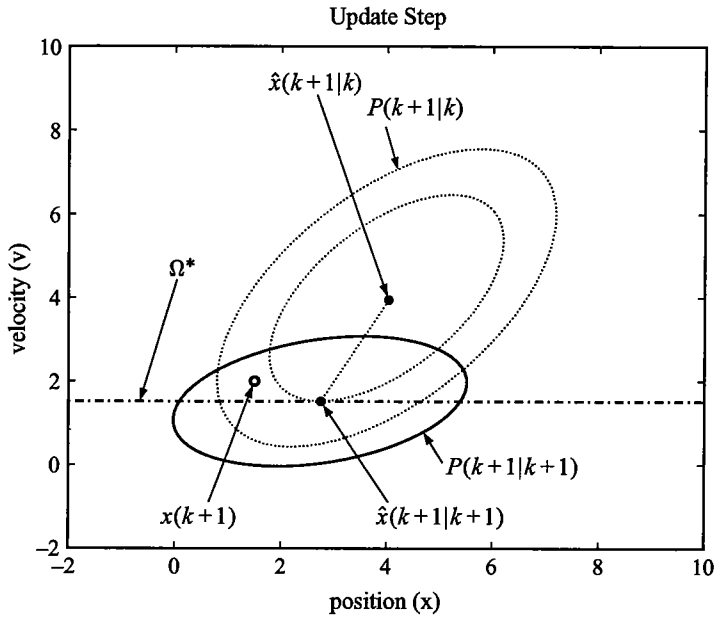

Figure 8.7 The updated estimate $\hat{x}(k+1 \mid k+1)$ is the point on Ω^* that is closest to $\hat{x}(k+1 \mid k)$ in terms of Mahalanobis distance. The smaller of the two dotted ellipses is the smallest ellipse that intersects Ω^*, hence $\hat{x}(k+1 \mid k+1)$ is defined by this intersection. Note that the line Ω^* represents the set of states that are consistent with the most likely output y^* that was found in figure 8.6.

For any observable linear system, the estimate provided by the Kalman filter is guaranteed to converge in the sense that the expected error between the actual and estimated state will be bounded for all time.

8.3 Extended Kalman Filter

The Kalman filter is a powerful tool for linear systems, but many systems encountered in practice are nonlinear. Consider the system

$$x(k+1) = f(x(k), u(k), k) + v(k) \tag{8.35}$$

$$y(k) = h(x(k), k) + w(k), \tag{8.36}$$

where x, y, u, v, and w are as before and

$$f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{Z}^+ \to \mathbb{R}^n$$

and

$$h : \mathbb{R}^n \times \mathbb{Z}^+ \rightarrow \mathbb{R}^p$$

are both continuously differentiable in $x(k)$. One approach to state estimation for systems of this type is to linearize the equations about the current estimate and then apply Kalman's equations using the resulting approximation. This formulation is called the extended Kalman filtering. The EKF equations are:

prediction:

$$\hat{x}(k+1 \mid k) = f(\hat{x}(k \mid k), u(k), k) \tag{8.37}$$

$$P(k+1 \mid k) = F(k)P(k \mid k)F(k)^T + V(k) \tag{8.38}$$

where

$$F(k) = \left.\frac{\partial f}{\partial x}\right|_{x=\hat{x}(k \mid k)} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{x=\hat{x}(k \mid k)} . \tag{8.39}$$

update:

$$\hat{x}(k+1 \mid k+1) = \hat{x}(k+1 \mid k) + Rv \tag{8.40}$$

$$P(k+1 \mid k+1) = P(k+1 \mid k) - RH(k+1)P(k+1 \mid k) \tag{8.41}$$

where

$$v = y(k+1) - h(x(k+1 \mid k), k+1) \tag{8.42}$$

$$S = H(k+1)P(k+1 \mid k)H(k+1)^T + W(k+1) \tag{8.43}$$

$$R = P(k+1 \mid k)H(k+1)^T S^{-1} \tag{8.44}$$

and

$$H(k+1) = \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}(k+1 \mid k)} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \frac{\partial h_1}{\partial x_2} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \frac{\partial h_2}{\partial x_1} & \frac{\partial h_2}{\partial x_2} & \cdots & \frac{\partial h_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_p}{\partial x_1} & \frac{\partial h_p}{\partial x_2} & \cdots & \frac{\partial h_p}{\partial x_n} \end{bmatrix}_{x=\hat{x}(k+1 \mid k)} . \tag{8.45}$$

8.3.1 EKF for Range and Bearing Localization

The EKF is well suited to the problem of localizing a mobile robot equipped with sensors that can detect range and bearing to previously mapped landmarks in the

environment [275]. Consider a robot whose state at time k is given by $x(k) = [x_r(k), y_r(k), \theta_r(k)]^T$, where $(x_r(k), y_r(k))$ denotes its position in the plane and $\theta(k)$ denotes its orientation. The input is $u(k) = [u_1(k), u_2(k)]^T$, where $u_1(k)$ and $u_2(k)$ denote the forward and angular velocities of the robot, respectively. The process model for this robot nonlinear, i.e.,

$$x(k+1) = \begin{bmatrix} \cos\theta_r(k)u_1(k) + x_r(k) \\ \sin\theta_r(k)u_1(k) + y_r(k) \\ u_2(k) + \theta_r(k) \end{bmatrix} + v(k),$$

where $v(k)$ is a random vector from a Gaussian distribution whose mean is zero and covariance is $V(k)$.

The robot is equipped with sensors that can measure the range and bearing to certain landmarks in the environment. Assume that the free space is populated with n_ℓ landmarks whose locations are known to be $(x_{\ell i}, y_{\ell i}), i = 1, 2, \ldots, n_\ell$. At any time k, the robot can only see the subset of landmarks that is within the range of its sensors, so the number of measurements taken varies with k. The number of measurements taken at the kth timestep is denoted by $p(k)$. Each measurement has two components, a range component and a bearing component. We also assume for now that for each measurement, we somehow know which landmark was observed. We introduce the association map a: $\{1, 2, \ldots, p(k)\} \rightarrow \{1, 2, \ldots, n_\ell\}$ which is defined such that the ith measurement at time k corresponds to the $a(i)$th landmark. The output equation for this system is then given as

$$y(k) = \begin{bmatrix} h_1(x(k), a(1)) \\ h_2(x(k), a(2)) \\ \vdots \\ h_{p(k)}(x(k), a(p(k))) \end{bmatrix} + \begin{bmatrix} w_1(k) \\ w_2(k) \\ \vdots \\ w_{p(k)}(k) \end{bmatrix},$$

where, for $i = 1, 2, \ldots, p(k)$,

$$h_j(x(k), j) = \begin{bmatrix} \sqrt{(x_r(k) - x_{\ell j})^2 + (y_r(k) - y_{\ell j})^2} \\ \text{atan2}(y_r(k) - y_{\ell j}, x_r(k) - x_{\ell j}) - \theta_r(k) \end{bmatrix}$$

and $w_i(k) \in \mathbb{R}^2$ is a random vector taken from a Gaussian distribution with zero mean and covariance matrix $W_i(k)$.

In order to linearize, we differentiate the process and sensor models with

$$F(k) = \left.\frac{\partial f}{\partial x}\right|_{x=\hat{x}(k|k)}$$

and

$$H(k+1) = \begin{bmatrix} H_1(k+1, a(1)) \\ H_2(k+1, a(2)) \\ \vdots \\ H_{p(k+1)}(k+1, a(p(k+1))) \end{bmatrix} = \begin{bmatrix} \frac{\partial h_1}{\partial x}\Big|_{x=\hat{x}(k+1|k)} \\ \frac{\partial h_2}{\partial x}\Big|_{x=\hat{x}(k+1|k)} \\ \vdots \\ \frac{\partial h_{p(k+1)}}{\partial x}\Big|_{x=\hat{x}(k+1|k)} \end{bmatrix}$$

resulting in

$$F = \begin{bmatrix} 1 & 0 & -\sin\theta_r(k)u_1(k) \\ 0 & 1 & \cos\theta_r(k)u_1(k) \\ 0 & 0 & 1 \end{bmatrix}$$

and

$$H_i(k+1, j) = \begin{bmatrix} \frac{(\hat{x}_r(k+1|k)-x_{\ell j})}{\sqrt{(\hat{x}_r(k+1|k)-x_{\ell j})^2+(\hat{y}_r(k+1|k)-y_{\ell j})^2}} & \frac{(\hat{y}_r(k+1|k)-y_{\ell j})}{\sqrt{(\hat{x}_r(k+1|k)-x_{\ell j})^2+(\hat{y}_r(k+1|k)-y_{\ell j})^2}} & 0 \\ \frac{-(\hat{y}_r(k+1|k)-y_{\ell j})}{1+\left(\frac{\hat{y}_r(k+1|k)-y_{\ell j}}{\hat{x}_r(k+1|k)-x_{\ell j}}\right)^2(\hat{x}_r(k+1|k)-x_{\ell j})^2} & \frac{1}{1+\left(\frac{\hat{y}_r(k+1|k)-y_{\ell j}}{\hat{x}_r(k+1|k)-x_{\ell j}}\right)^2(\hat{x}_r(k+1|k)-x_{\ell j})} & -1 \end{bmatrix}.$$

With these matrices in hand, we can use the linearized Kalman filter equations to estimate the robot state.

8.3.2 Data Association

The solution presented in the previous section glosses over one very important aspect of localization: it assumes that each measurement is automatically associated with the correct landmark. In practice, landmarks have similar properties which make them good features but often make them difficult to distinguish one from another. When this happens, we must address the problem of *data association,* which is the question of which landmark corresponds to a particular measurement. This is equivalent to finding the association map used in the previous section.

The basic idea used for data association is as follows. Consider the ith measurement $y_i(k+1)$. For each landmark in the map, we compute the innovation ν_{ij} which is defined to be the difference between the actual measurement $y_i(k+1)$ and the measurement that we would expect if $y_i(k+1)$ corresponded to the jth landmark and the prediction $\hat{x}(k+1\,|\,k)$ was correct. This means that

$$\nu_{ij} \triangleq y_i(k+1) - h_i(\hat{x}(k+1\,|\,k), j).$$

The smaller the innovation ν_{ij}, the more likely it is that the ith measurement corresponds to the jth landmark. We then can make a good guess of which landmark corresponds to the measurement by choosing the landmark that yields the smallest innovation. However, the notion of size must be weighted by the uncertainties in the predictions and measurements. Fortunately, these uncertainties are encoded in the matrix S from the Kalman filter update equation (8.31), and S can be used to create a Mahalanobis norm for ν_{ij} which indicates the size of the innovation in units of standard deviations. We write this measure of the innovation as

$$\chi_{ij}^2 = \nu_{ij} S_{ij}^{-1} \nu_{ij}^T,$$

where

$$S_{ij} = H_i(k+1, j) P(k+1 \mid k) H_i(k+1, j)^T + W_i(k+1).$$

We then can build the data association function $a()$ by setting $a(i)$ equal to the value of j that minimizes χ_{ij}.

Figure 8.8 contains an example of localization using a Kalman filtering. The actual path, i.e., ground truth, is displayed along with estimates of the robot's location and its uncertainty of that estimate as the robot moves along the path. Note that the estimate converges to the actual path as the robot moves along the path and as more measurements are taken. Also, note that the uncertainty of the estimate considerably decreases as well.

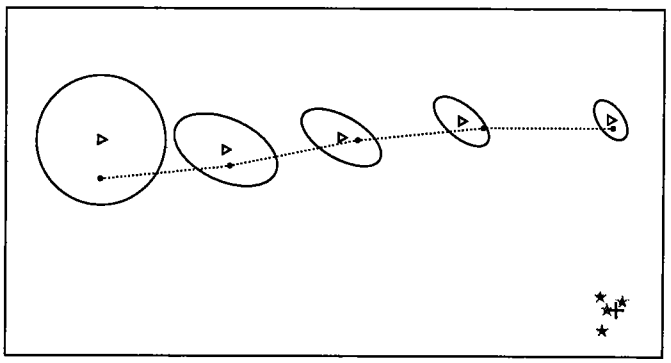

Figure 8.8 The dotted line displays the actual path of the Nomad Scout mobile robot. The triangles and ellipses correspond to the estimated location and variance, respectively, of the robot's location. The stars correspond to the measured location of the beacon, which also has an associated variance (due to noise) which is not displayed.

8.3.3 EKF for Range-Only Localization

The EKF solution to the problem of localization using range-only sensors is a trivial extension to the range and bearing case. The only difference is that the output equation will not contain any bearing information, so we simply remove the rows from h and H that correspond to the bearing measurements. The EKF equations are then applied in the usual manner.

8.4 Kalman Filter for SLAM

In this section we introduce the use of the Kalman filter to solve the problem SLAM which has been an active topic of research in recent years (see, e.g., [96, 125,316,384]). We begin with a very simple case where the robot is able to measure the relative displacement between itself and a number of fixed landmarks. The simple example also assumes that each sensor reading is automatically associated with the correct landmark so that the robot does not have to determine which landmark corresponds to any given measurement. After using this simple example to demonstrate the basic concept of Kalman filter-based SLAM, we present a more realistic example where the robot measures range and bearing to fixed landmarks and the data association problem is not automatically solved.

8.4.1 Simple SLAM

One common approach to solving the SLAM problem is to use a Kalman filter to simultaneously estimate the position of a moving vehicle along with the positions of landmarks seen by the vehicle. This technique was originally suggested by Smith, Self, and Cheeseman [384]. Here we present the most basic example of this technique: we assume an omnidirectional motion model for the vehicle and we assume that the vehicle has sensors capable of uniquely identifying each landmark and providing a measurement of the relative displacement between the vehicle and the landmark. We assume that the vehicle's sensor can see every landmark at every instant of time.

We first define the state to be the location of the vehicle (x_r, y_r) together with the locations of each of the landmarks, $(x_{\ell i}, y_{\ell i})$, $i = 1, 2, \ldots, n_\ell$, where n_ℓ is the total number of landmarks. In other words,

$$x = [x_r \quad y_r \quad x_{\ell 1} \quad y_{\ell 1} \quad x_{\ell 2} \quad y_{\ell 2} \quad \cdots \quad x_{\ell n_\ell} \quad y_{\ell n_\ell}]^T.$$

We assume that the control inputs are u_x and u_y, the vehicle velocities in the x- and y-directions, respectively. We model the errors associated with this motion with the

random vector $v_r(k) = [v_{rx}(k), v_{ry}(k)]^T$, which is zero-mean white Gaussian noise with covariance matrix $V_r(k)$. The landmarks do not move, so the resulting dynamic equations for the system are

$$x(k+1) = x(k) + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_x(k) \\ u_y(k) \end{bmatrix} + \begin{bmatrix} v_{rx}(k) \\ v_{ry}(k) \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

This equation can clearly be written in the form

$$x(k+1) = Fx(k) + Gu(k) + v(k),$$

where $v(k)$ is a zero-mean white Gaussian noise with covariance matrix

$$V(k) = \begin{bmatrix} V_r(k) & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}.$$

The measurement to the ith landmark is the position of the landmark relative to the vehicle plus some noise, i.e.,

$$y_i(k) = \begin{bmatrix} x_{\ell i}(k) - x_r(k) \\ y_{\ell i}(k) - y_r(k) \end{bmatrix} + w_i(k),$$

where $w_i(k)$ is an independently distributed Gaussian random vector with covariance matrix $W_i(k)$. Note the $y_i(k)$ is a linear function of the system state $x(k)$. Specifically, we can write

$$y_i(k) = H_i x(k) + w_i(k),$$

where

$$H_i = \begin{bmatrix} -1 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 \end{bmatrix}.$$

The first row of H has a -1 in the first column that to corresponds x_r and a 1 in the $(2i+1)$th column that corresponds to $x_{\ell i}$, and zeros everywhere else. Similarly, the second row is all zeros except for a -1 in the second column and a 1 in the $(2i+2)$th column.

With this notation, we can stack all of the measurements together to create one big measurement vector $y = [y_1, y_2, \ldots, y_{n_\ell}]^T$ which gives the measurement equation

$$y(k) = Hx(k) + w(k),$$

where

$$H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_{n_\ell} \end{bmatrix}, \quad \text{and} \quad w(k) = \begin{bmatrix} w_1(k) \\ w_2(k) \\ \vdots \\ w_{n_\ell}(k) \end{bmatrix},$$

and the covariance matrix associated with $w(k)$ is

$$W(k) = \begin{bmatrix} W_1(k) & 0 & \cdots & 0 \\ 0 & W_2(k) & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & W_{n_\ell} \end{bmatrix},$$

where $W_i(k)$ is the covariance matrix associated with $w_i(k)$. The problem has now been put into a form suitable for the Kalman filtering equations in section 8.2.5. Kalman estimates of the system state x provide estimates of both vehicle and landmark locations, hence solving the SLAM problem.

8.4.2 Range and Bearing SLAM

Now we consider the SLAM problem for a mobile whose inputs are forward velocity and angular velocity and whose measurements are range and bearing readings. In a sense, we are combining the range-bearing localization approach from section 8.3.1 with the SLAM approach described above in section 8.4.1. The difference is that the number of columns in the H matrix is the same as the number of rows in the state vector. Moreover, the H matrix now contains partial derivatives of the measurement equations with respect to the state.

The measurement equations are the same as in the range and bearing localization example, i.e.,

$$y_i(k) = \begin{bmatrix} \sqrt{(x_{\ell i}(k) - x_r(k))^2 + (y_{\ell i}(k) - y_r(k))^2} \\ \text{atan2}((y_{\ell i}(k) - y_r(k)), (x_{\ell i}(k) - x_r(k)) - \theta_r(k) \end{bmatrix} + w_i(k). \tag{8.46}$$

The first three columns of the H matrix will be fairly dense since the planar location of the robot is part of both measurement equations. The columns to the right will be

sparse as in the last example of EKF SLAM since the measurement of each landmark is only a function of the robot position and that landmark's position.

$$H_i = \begin{bmatrix} \frac{\partial y_i}{\partial x_r} \\ \frac{\partial y_i}{\partial y_r} \end{bmatrix} \tag{8.47}$$

$$= \begin{bmatrix} \frac{-x_{\ell i}(k)+x_r(k)}{\rho_i} & \frac{-y_{\ell i}(k)+y_r(k)}{\rho_i} & 0 & \dots & \frac{x_{\ell i}(k)-x_r(k)}{\rho_i} & \frac{y_{\ell i}(k)-y_r(k)}{\rho_i} & \dots \\ \frac{y_{\ell i}(k)-y_r(k)}{\rho_i^2} & \frac{-x_{\ell i}(k)+x_r(k)}{\rho_i^2} & -1 & \dots & \frac{-y_{\ell i}(k)+y_r(k)}{\rho_i^2} & \frac{x_{\ell i}(k)-x_r(k)}{\rho_i^2} & \dots \end{bmatrix}$$

where ρ_i is the range of the landmark as given in the measurement equation. Now, we substitute the modified H matrix into the previously defined framework for Kalman filter SLAM.

Again, we have the problem of data association, i.e., we must determine which landmark corresponds to each measurement. We also have to determine when a new landmark has been encountered. Once again, we use the Mahalanobis distance metric to compare the ith measurement with the measurement prediction for the jth landmark, i.e.,

$$\chi_{ij}^2 = \big(y(k)_i - h(k)_j\big)^T S_{ij} \big(y(k)_i - h(k)_j\big). \tag{8.48}$$

Once the χ_{ij} has been calculated for each combination of landmarks and measurements, the minimum is checked against an acceptance threshold to assure that the match is likely enough. If the minimum χ is above a high threshold, then the measurement is not likely to have come from any existing landmark. Therefore, we have an indication that a new landmark should be initialized and added to the map.

Problems

1. The methods presented in this chapter generally assume that the noise is modeled as a zero-mean white Gaussian random process and that the noise enters the system dynamics and measurement equations through addition. It is important to understand that this is always an approximation; real systems never contain such nice noise. For each of the noise sources listed below, briefly describe how the zero-mean white Gaussian noise assumption used in the Kalman filter fails:
   (a) distance measurements;
   (b) bearing measurements;
   (c) odometry error in a differentially steered wheeled robot due to a mismatch in wheel size;
   (d) odometry error in a wheeled robot due to wheel slippage;
   (e) sonar errors due to multipath reflections;
   (f) temperature dependent drift in a rate gyro.

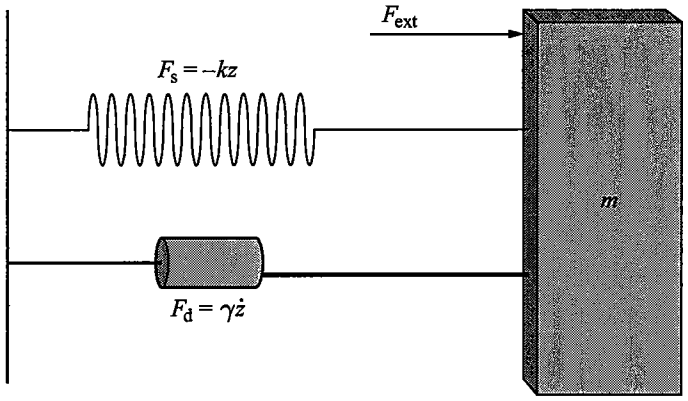

Figure 8.9 Mass–spring–damper.

2. The mass–spring–damper system shown in figure 8.9 with mass m, spring constant k, and damping coefficient γ can be modeled by the second order differential equation

   $$m\ddot{z} + \gamma\dot{z} + kz = 0.$$

   Define the system state to be $x = [z \ \dot{z}]^T$. In discrete time with sampling time step T, the derivative $\frac{d}{dt}$ can be approximated as

   $$\dot{x}(t)\Big|_{t=kT} \approx \frac{x(k+1) - x(k)}{T}.$$

   Use this approximation to write the mass–spring–damper system as a linear discrete time system in state space.

3. Consider the linear time invariant, noise-free, zero-input, single-output discrete time system

   $$x(k+1) = Fx(k); \quad y(k) = Hx(k),$$

   where $x(k) \in \mathbb{R}^2$.

   (a) Describe the set of states $x(k)$ that are possible given the measurement $y(k)$.
   (b) Describe the set of states $x(k+1)$ that are possible given the measurement $y(k)$ (but not $y(k+1)$).
   (c) Given the measurement $y(k+1)$, under what condition will the set of possible $x(k+1)$ be a single point? How does this relate to the result in theorem 8.2.2?

4. Let $\hat{x} \in \mathbb{R}^n$, $y \in \mathbb{R}^p$, and $H \in \mathbb{R}^{p \times n}$. Define the hyperplane $\Omega = \{x \in \mathbb{R}^n \mid Hx = y\}$. Let P be a positive definite matrix and define the norm $\|x\|_P = x^T Px$. Show that, with respect to this norm, the shortest vector Δx that satisfies $H(x + \Delta x) = y$ must be orthogonal to Ω, i.e., $\Delta x^T Pa$ must be zero for every a that is parallel to Ω.

5. Using the definitions from problem4, show that null(H) is parallel to Ω.

6. Show that for a matrix A, null(A) is orthogonal to column(A^T).

7. The solutions to the following sequence of problems combine to form the derivation of equation (8.15). To make the expressions more manageable, we introduce the notation $x_k = x(k)$, $\hat{x}_k = \hat{x}(k \mid k)$, $\hat{x}_{k+1}^- = \hat{x}(k+1 \mid k)$, $P_k = P(k \mid k)$, and $P_{k+1}^- = P(k+1 \mid k)$. We also denote the innovation as $\nu = \nu(k+1) = y(k+1) - H(k)x_k$.

   (a) Starting with the definition of P_k,

   $$P_k = E\left[(x_{k+1} - \hat{x}_{k+1})(x_{k+1} - \hat{x}_{k+1})^T\right],$$

   show that

   $$P_k = E\left[(x_{k+1} - \hat{x}_{k+1}^-)(x_{k+1} - \hat{x}_{k+1}^-)^T - 2R\nu(x_{k+1} - \hat{x}_{k+1}^-)^T - R\nu(R\nu)^T\right].$$

   (b) Continue to show that

   $$P_k = P_{k+1}^- - 2RHP_{k+1}^- + RHP_{k+1}^- H^T R^T.$$

   (c) Next show that

   $$P_k = P_{k+1}^- - 2RHP_{k+1}^- H^T R^T + RHP_{k+1}^- H^T R^T.$$

   Equation (8.15) follows trivially from this last expression.

8. The solutions to the following sequence of problems combine to form the proof of theorem 8.2.1. Consider two multivariate Gaussian distributions with mean-covariance pairs (z_1, C_1) and (z_2, C_2).

   (a) Show that the product of these two Gaussian distributions is proportional to

   $$e^{-\frac{1}{2}(z^T(C_1^{-1}+C_2^{-1})z - 2z^T(C_1^{-1}z_1 + C_2^{-1}z_2) + z_1^T C_1^{-1} z_1 + z_2^T C_2^{-1} z_2)}.$$

   (b) Consider the term in the exponential of a Gaussian with mean-covariance pair (z_3, C_3). By equating the terms that are quadratic in z, show that

   $$C_3 = C_1 - QC_1,$$

   where $Q = C_1(C_1 + C_2)^{-1}$.

   (c) By equating the terms that are linear in z, show that

   $$z_3 = z_1 + Q(z_2 - z_1).$$

9. Consider the nonlinear discrete time system

$$x(k+1) = \begin{bmatrix} x_1(k) + Tu_1(k)cos(x_3(k)) \\ x_2(k) + Tu_1(k)sin(x_3(k)) \\ x_3(k) + Tu_2(k) \end{bmatrix}; \qquad y(k) = x_1(k) + w(k),$$

where $v(k)$ is Gaussian white noise with zero mean and variance 0.5. Suppose the estimate $\hat{x}(1|1) = [1\ 0.5\ \frac{\pi}{4}]^T$, the input $u(1) = [3\ \pi]$, and the time step $T = 0.25$. Also suppose that the covariance estimate $P(1|1)$ is the 3×3 identity matrix. Using the extended Kalman filter formulation,

(a) compute the predicted estimate and covariance, $\hat{x}(2|1)$ and $P(2|1)$.
(b) given the measurement $y(2) = 1.7$, compute $\hat{x}(2|2)$ and $P(2|2)$.

10. Consider the system of problem 9 with noise added to the inputs: $u_1(k)$ is replaced by $u_1(k) + s_1(k)$ and $u_2(k)$ is replaced by $u_2(k) + s_2(k)$, where s_1 and s_2 are Gaussian white noise with variances σ_1^2 and σ_2^2 respectively. Given an estimate $\hat{x}(k \mid k)$, state the equation used to find the estimate covariance prediction $P(k+1 \mid k)$.

11. Consider the system given by equations (8.1) and (8.2) with all of the standard assumptions. Show that if the initial estimate $\hat{x}(0|0)$ is such that the expected value of the intitial estimate error $E[x(0) - \hat{x}(0|0)] = 0$, then the expected value of the error of the estimate provided by the Kalman filter remains zero for all k.