BROWN UNIVERSITY
Department of Computer Science
Master's Thesis
CS-90-M13

Localization with Extended Kalman Filtering

by
Jin Joo Lee

Localization with Extended Kalman Filtering

by

Jin Joo Lee

B.S., Ewha Womans University, 1988

Thesis

Submitted in partial fullfillment of the requirements for the degree of Master of Science in the Department of Computer Science at Brown University.

May 1991

This thesis by Jin Joo Lee
is accepted in its present form by the Department of Computer Science
as satisfying the thesis requirement for the degree of Master of
Science

August 20, 1990

Thomas L. Dean

Thomas L. Dean
Advisor

1 Introduction

In the area of autonomous mobile robot navigation, acquiring reasonably accurate position information can be vital to the overall success of many tasks. Due to slippage of the wheels and possible inaccuracies of the shaft encoders, one cannot solely rely on the robot's odometry to calculate its position. The need to overcome such uncertainty in position calculation directed researchers in robotics to search many disciplines for techniques that would accomplish such tasks. Here, we attempt to overcome such problems by applying a well known technique used in engineering that produces an *estimate* of a position: an estimate that minimizes the estimation error and thus is called the *optimal* estimate. This technique is known as extended Kalman filtering (EKF).

In this document, we first provide the mathematics of EKF along with some intuitions as to why this technique really works. We then describe, in detail, the actual mechanics involved in our application domain, namely, robotics. We also show how we were able to solve the problem of recognizing false sonar readings (due to multiple reflections) by applying a correspondence criteria which distinguishes between true and false readings by validating only those readings that were measured within some angular range.

We have implemented this approach on a real mobile robot. On top of a RWI (Real World Interface) base, we have mounted 8 Polaroid ultrasonic rangefinders. These sonars are our only means of sensing the outside world.

This project provides as a localization tool for various robotics projects involving planning, learning and control in a known[1] environment.

2 Kalman Filtering

Ever since the manual tracing of "blips" on radar and sonar systems evolved into computer controlled tracking algorithms, tracking of any kind of sensor data has continued to develop significantly.

Tracking "is the processing of measurements obtained from a target in order to maintain an estimate of its current state"[1].

A current *state* typically consists of:

[1] by *known*, we mean an environment where we know the positions of the objects respect to some global coordinate system

1. Kinematic components (position, velocity, acceleration, etc.).
2. Other components (spectral characteristics, "feature" information).
3. Constant or slowly-varying parameters.

Measurements are noise-corrupted observations related to the state of a target, such as

1. Direct estimate of position.
2. Range and/or azimuth (bearing) from a sensor.
3. Time of arrival difference between two sensors.
4. Frequency of narrow-band signal emitted by target.
5. Observed frequency difference between two sensors due to Doppler shift.
6. Signal strength.

The Kalman filter algorithm has proven to be quite successful in filtering sonar data [8]. The Kalman filter is an estimation algorithm that takes the current state, the control input, noisy observations and produces the optimal linear estimate of its current state along with the associated error variance. Kalman Filtering "attracted considerable attention because of its general validity, mathematical elegance and widespread technical application"[2]. Another striking feature of the Kalman Filter is the number of different ways the solution equations can be derived. The maximum likelihood method, the method of minimum variance and the least-squares method are derivation methods that were discovered following the initial system of differential equations derived by Bucy and Kalman (1961)[3].

The filtering algorithm that we have chosen for this project is the extended Kalman filter (EKF) [1, 5], which is a filter made for non-linear systems. In section 2.2 we describe the necessary equations for our tracking purposes. The techniques are discussed in more detail in [1, 5].

2.1 Review of underlying mathematics

Before we go on any further into the actual mathematics of EKF (Extended Kalman Filter), let us define and review some underlying mathematics. For more detailed descriptions, refer to [5].

Random Variables: a variable **X** which takes on values at random. It may also be thought of as a function of the outcomes of some random experiment. The *probability distribution function*

$$\mathbf{F(x)} = \Pr(\mathbf{X} \leq \mathbf{x})$$

and the *probability density function*

$$f(x) = \frac{dF(x)}{dx}$$

both specify the probability with which different values are taken by the random variable.

Expectation (or Mean) of a Random Variable: is defined as the sum of all values the random variable may take, each weighted by the probability with which the value is taken.

$$E[X] = \sum_i x_i p_i$$

Covariance of Random Variables: indicates the degree to which one variable is related to another, which is the expectation of the product of the deviations of two random variables from their means,

$$E[(X - E[X])(Y - E[Y])] = E[XY] - E[X]E[Y]$$

Normal Probability Distributions: is characterized by the *normal probability density function*

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

μ is the mean and σ is the standard deviation.

For n random variables, the *multidimensional normal distribution* is

$$f_n(x_1, x_2, \cdots, x_n) = \frac{1}{(2\pi)^{\frac{n}{2}} |P|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\underline{x} - \underline{\mu})' P^{-1} (\underline{x} - \underline{\mu})\right]$$

where $\underline{\mu} = E[\underline{x}]$ and $P = E[(\underline{x} - \underline{\mu})(\underline{x} - \underline{\mu})']$ are the mean and the covariance of the vector $\underline{x}' = (x_1, x_2, \cdots, x_n)$ respectively. The line under the letter x denotes the vector and $\underline{x}'$ is the transpose of the vector $\underline{x}$.

Stochastic (Random) Processes: a collection of functions of time, any one of which might be observed on any trial of an experiment. We shall denote the collection of functions by $\{x(t)\}$, and any observed member of such collection by $x(t)$.

Gaussian Processes: a random process where its joint probability distribution functions of all orders are multidimensional normal distributions. If $\underline{x}(t)$ is an n-dimensional gaussian vector then the normal distribution is expressed by

$$f(\underline{x}, t) = \frac{1}{(2\pi)^{\frac{n}{2}} \mid P \mid^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\underline{x} - \underline{\mu})' P^{-1} (\underline{x} - \underline{\mu})\right]$$

where $\underline{\mu}$ is the mean and P is the covariance of the vector $\underline{x}$.

White Processes (Noise): a random process where the distribution of the expected value of the members of the collection of functions $x(t)$ (or power) over all frequency components in the full (theoretically infinite but practically wideband) range is uniform. This equal distribution has led to the name "white" in analogy with the white light, which has an approximately constant spectrum over the visible range. This is an idealized concept which does, however, serve as a very useful approximation where noise is wideband compared with the bandwidth of the system.

2.2 The EKF algorithm

Assume the robot's position is represented as a point in a global coordinate space by the vector $\mathbf{x}$ and we know this vector at time k. Then using the control input vector $\mathbf{u}$, we can calculate the trajectory and obtain the position vector for the next time step $k+1$. But since our world is not perfect and usually has some amount of noise, we must take into account some motion error. We model such error by vector v, which is a sequence of white Gaussian processes with zero-mean. In a short notation, this is expressed as $v(k) \sim N(0, Q(k))$, where $Q(k)$ is the variance.

Then, for a discrete-time, stochastic system, we can represent the system motion as follows:

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

where f is a state transition function that maps $\mathbf{x}(k)$ to $\mathbf{x}(k+1)$ accounting for the control action. We distinguish the system to be linear or non-linear according to the linearity of the function f. To obtain the next position that is as close (statistically) to the actual as possible, we have added the vector $v(k)$. k is the time argument indicating that the vector-valued function f is, in general, time-varying.

Observations are generally measurements that provide some information about the positions of targets. Given the robot's current position $\mathbf{x}(k)$, the position information of the target described by the vector p and the observation noise vector $w(k)$, we can easily calculate the observation $\mathbf{z}(k)$ by applying a simple distance measuring function h.
Observations are described by (in more general form)

$$\mathbf{z}(k) = h(p, \mathbf{x}(k)) + w(k) \tag{2}$$

where w is again assumed to be zero-mean, white, Gaussian; $w(k) \sim N(0, R_i(k))$, where $R_i(k)$ is the variance of the distribution.

There are three basic tracking steps in the Kalman filter: prediction, observation and estimation. In the following we describe them in more detail. Let us assume that one has the estimate of the vector $\mathbf{x}$ at time k, which we denote as $\hat{\mathbf{x}}$. An estimate $\hat{\mathbf{x}}$, is the computed value of a quantity, $\mathbf{x}$, based upon a set of observations. Depending on the criterion of optimality, there are several different ways and forms of estimation. The one we want to use here is one that minimizes the variance. That is to minimize the cost functional

$$J = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}(\hat{\mathbf{x}} - \mathbf{x})' S\,(\hat{\mathbf{x}} - \mathbf{x})\,p(\mathbf{x}|\mathbf{z})\,d\mathbf{x}_1 d\mathbf{x}_2 \ldots d\mathbf{x}_n$$

where S is an arbitrary, positive semidefinite matrix. Independent of S, when we set $\partial J/\partial \hat{\mathrm{x}} = 0$, we find that

$$\hat{\mathrm{x}} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} \mathrm{x}p(\mathrm{x}|\mathrm{z})\, d\mathrm{x}_1 d\mathrm{x}_2 \ldots d\mathrm{x}_n = E[\mathrm{x}||\mathrm{z}] \qquad (3)$$

which is the conditional mean estimate. In short, we write

$$\hat{\mathrm{x}}(k|k) \approx E[\mathrm{x}(k)|\mathrm{z}(1)\cdots\mathrm{z}(k)] \qquad (4)$$

where $\hat{\mathrm{x}}(k|k)$ denotes the conditional mean (or the expected value of $\mathrm{x}(k)$) given the first k observations.

From basic probability theory, we know that the variance of a random variable is $E((x-\mu)^2)$ where μ is the mean. Applying to our notation, we can derive the covariance matrix

$$P(k|k) = E\left[[\mathrm{x}(k) - \hat{\mathrm{x}}(k|k)][\mathrm{x}(k) - \hat{\mathrm{x}}(k|k)]' \mid \mathrm{z}^k\right] \qquad (5)$$

or

$$= E\left[[\tilde{\mathrm{x}}(k \mid k)][\tilde{\mathrm{x}}(k \mid k)]' \mid \mathrm{z}^k\right] \qquad (6)$$

using the notation $\tilde{\mathrm{x}}$ which represents the estimation error $\mathrm{x} - \hat{\mathrm{x}}$.

To obtain the predicted state at time $k+1$ based on information available up to time k, the nonlinear function in (1) is expanded in Taylor series around $\hat{\mathrm{x}}(k|k)$ with terms up to first or second order to yield the first- or second-order EKF, respectively. The expansion with first-order term is

$$\mathrm{x}(k+1) = f(\hat{\mathrm{x}}(k|k), u(k)) + f_{\mathrm{x}}(k)[\mathrm{x}(k) - \hat{\mathrm{x}}(k|k)] + v(k) \qquad (7)$$

$f_{\mathrm{x}}(k)$ is the Jacobian of f:

$$f_x(k) = [\nabla_{\mathrm{x}} f'(k, \mathrm{x})]'_{\mathrm{x}=\hat{\mathrm{x}}(k|k)}$$

where

$$\nabla_{\mathrm{x}} f'(k,\mathrm{x}) = \left[\begin{array}{ccc} \frac{\partial}{\partial \mathrm{x}_1} & \cdots & \frac{\partial}{\partial \mathrm{x}_n}\end{array}\right]' [f_1(k,\mathrm{x})\cdots f_n(k,\mathrm{x})]$$

evaluated at the latest estimate of the state. Thus, the prediction of the state to $k+1$ from k is obtained by taking the expectation of (7) conditioned on

$\mathbf{z}^{k\dagger}$ and neglecting the higher-order terms. The first-order term in (7) does not appear in (8) because it is approximately zero-mean and since (4).

$$\hat{\mathrm{x}}(k+1|k) = f(\hat{\mathrm{x}}(k|k), \mathbf{u}(k)) \tag{8}$$

The covariance associated with the predicted state is obtained by first subtracting (8) from (7) to yield

$$\mathrm{x}(k+1) - \hat{\mathrm{x}}(k+1|k) = f_{\mathrm{x}}(k)[\mathrm{x}(k) - \hat{\mathrm{x}}(k|k)] + v(k)$$

rewritten in a shorter notation,

$$\tilde{\mathrm{x}}(k+1|k) = f_{\mathrm{x}}\tilde{\mathrm{x}}(k|k) + v(k) \tag{9}$$

The higher-order terms have been dropped already. Multiplying the above by its transpose

$$\begin{aligned} [\tilde{\mathrm{x}}(k+1|k)][\tilde{\mathrm{x}}(k+1|k)]' &= f_x\tilde{\mathrm{x}}(k|k)f_x'\tilde{\mathrm{x}}'(k|k) + f_x\tilde{\mathrm{x}}(k|k)v(k) \\ &\quad + f_x'\tilde{\mathrm{x}}'(k|k)v(k) + Q(k) \end{aligned} \tag{10}$$

now, taking the expectation of (10) conditioned on $\mathbf{z}^k$ and using (6) yields

$$P(k+1|k) = f_x P(k|k) f_x' + Q(k) \tag{11}$$

where $Q(k)$ is the motion noise variance $(v(k))^2$. Thus, $P(k+1|k)$ is just the sum of this noise variance and the propagated error $f_x P(k|k) f_x'$.

Similarly, to predict what we expect to observe at time step $k+1$, we first obtain $\mathbf{z}(k+1)$ by substituting k to $k+1$ in (2) producing

$$\mathbf{z}(k+1) = h(p, \mathrm{x}(k+1)) + w(k+1)$$

Then, we expand the above in Taylor series around $\hat{\mathrm{x}}(k+1|k)$ with terms up to first order only, as in (7). Then we take expectations conditioned on $\mathbf{z}^k$ and obtain the estimate of the predicted target observation as follows:

$$\hat{\mathbf{z}}(k+1|k) \approx h(p, \hat{\mathrm{x}}(k+1|k)) \tag{12}$$

which is just the calculated observation at the expected position $\hat{\mathrm{x}}(k+1|k)$.

[†] $\mathbf{z}^k = \{\mathbf{z}(j), j = 1, \ldots, k\}$

Now, we take the actual observation (the sonar range of the target) $\mathbf{z}(k+1|k)$, and compare this with the prediction.

$$\nu(k+1) = [\mathbf{z}(k+1|k) - \hat{\mathbf{z}}(k+1|k)] \tag{13}$$

The associated variance of the *innovation* vector $\nu(k+1)$ is obtained in the same way as the covariance

$$S(k+1) = E[\nu(k+1)\nu(k+1)^{'}] = h_{\mathbf{x}}P(k+1|k)h_{\mathbf{x}}^{'} + R(k+1) \tag{14}$$

where the Jacobian of h is

$$h_{\mathbf{x}}(k+1) = [\nabla_{\mathbf{x}}h'(k+1|k)]^{'}_{\mathbf{x}=\hat{\mathbf{x}}(k+1|k)}$$

If the observation corresponds to the prediction, we are now ready to compute the *filter gain* which is called the Kalman Gain Matrix, written as:

$$W = P(k+1|k)h_{\mathbf{x}}^{'}(k+1)S^{-1}(k+1) \tag{15}$$

and thus the optimal linear estimate is

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + W(k+1)\nu(k+1) \tag{16}$$

where $\nu(k+1)$ is as in (13).

The optimality of the Kalman filter is contained in its structure and in the specification of the gain matrices. Let us assume for the moment that h is the identity matrix. In such a case P and S^{-1} are both $n \times n$ matrices. If S^{-1} is a diagonal matrix (no cross-correlation between terms), W is just the product of the error covariance P and the inverse of the mean square observation noise. Essentially, each element of this Kalman Gain Matrix is the ratio between statistical measures of the uncertainty in the state estimate and the uncertainty in the observation.

Thus, the gain matrix is *proportional* to the uncertainty of the estimate and *inversely proportional* to the observation noise. In (16), we can see that the innovation vector ν is multiplied by W either to increase or decrease the amount of correction added to the linear estimate. For example, if the uncertainty of the estimate was large and the variance of the observation noise was small, it would mean that the quantity ν contained considerable information about the error estimates and therefore, should be magnified by W. Conversely, if the uncertainty of the estimate was small and the observation noise was large, the quantity ν was due mainly because of the noise and therefore, should affect the linear estimate very little.

Consequently, the difference between actual and predicted observations will be used to provide basis for corrections to the estimate. Intuitively, it is not difficult to see why the Kalman Gain Matrix will improve the estimates.

Finally, the covariance of the state at time $k+1$ is

$$P(k+1|k+1) = P(k+1|k) - W(k+1)S(k+1)W^{'}(k+1) \qquad (17)$$

The effect of measurement noise on the error covariance of the discrete filter can be best observed from the following matrix inversion relationship which can be obtained after some manipulations. [5] discusses the derivation of the following equation in more detail.

$$P^{-1}(k+1|k+1) = P^{-1}(k+1|k) + h_{\mathbf{x}}(k+1)S^{-1}(k+1)h_{\mathbf{x}}(k+1)$$

When measurement noise is large ($S^{-1}(k+1)$ is small), there is only a small increase in the inverse of the error covariance (a small *decrease* in the error covariance). Which means the associated measurements contribute little to the reduction of estimation errors. On the other hand, when the measurement noise is small ($S^{-1}(k+1)$ is large), the inverse of the error covariance decreases in large amounts (a considerable decrease in the error covariance). And thus the contribution of the associated measurements to the reduction of estimation errors becomes significant.

In an overview, EKF **predicts**:

a. the state estimate
b. the target observation estimate

Then it **observes**:

c. the state with additive noise
d. the target with additive noise

producing:

- from a and c, the covariance
- from b and d, the innovation

Finally,

- the optimal linear estimate for time $k+1$ is produced by adding the weighted innovation to the predicted state estimate. Innovation is weighted accordingly with the degree of uncertainty in the position estimate and the observation or put simply, the size of the covariance and innovation.

2.3 Applying EKF to Robot Navigation

In this section, we describe how we applied EKF to our particular problem of robot navigation.

We have chosen to represent the robot's position with respect to a fixed global frame at time k as a vector

$$\mathbf{x}(k) = \begin{bmatrix} x(k) \\ y(k) \\ \theta(k) \end{bmatrix} \tag{18}$$

where $x(k)$ and $y(k)$ describe the cartesian location and $\theta(k)$ represents the heading measured counterclockwise from the positive x-axis.

The robot's motion is controlled by the control input

$$\mathbf{u}(k) = \begin{bmatrix} T(k) \\ \Delta\theta(k) \end{bmatrix} \tag{19}$$

where $T(k)$ is the speed per time step and $\Delta\theta(k)$ is the rotational value per time step.

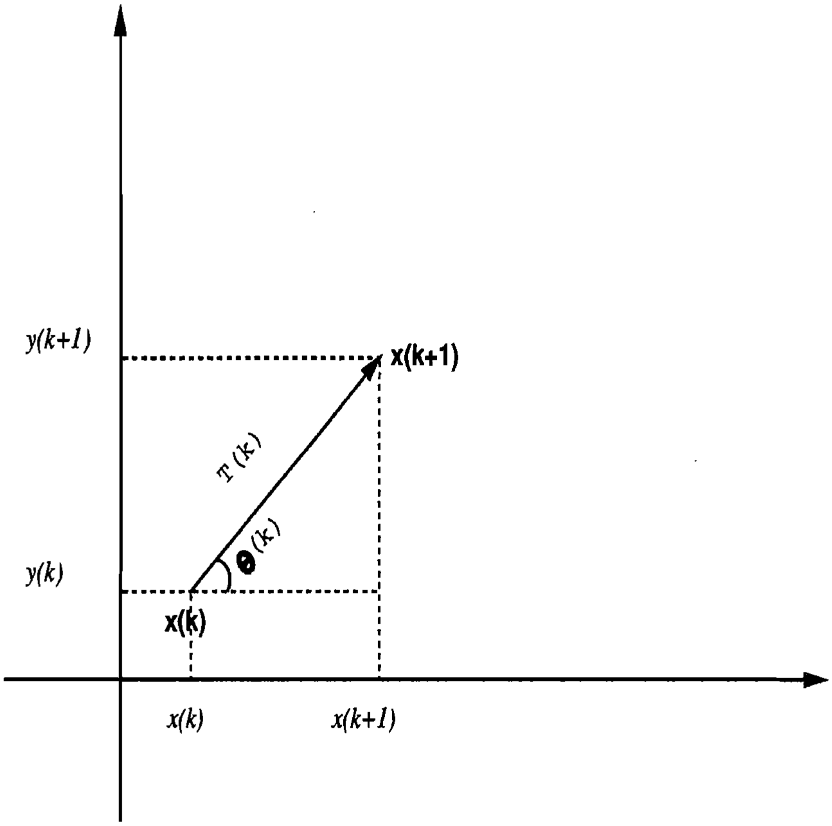

Figure 1: calculating positions

To obtain the coordinates of the position at time $k+1$, given that we know the position at time k, we merely need to add the Δ terms of each variable as follows:

$$\mathrm{x}(k+1) = \begin{bmatrix} x(k) + \Delta x(k) \\ y(k) + \Delta y(k) \\ \theta(k) + \Delta\theta(k) \end{bmatrix} \tag{20}$$

we can derive the $\Delta^{'}$s using some trigonometric functions.
Thus, the state transition function f,

$$f(\mathrm{x}(k), \mathbf{u}(k)) = \begin{bmatrix} x(k) + T(k)\cos\theta(k) \\ y(k) + T(k)\sin\theta(k) \\ \theta(k) + \Delta\theta(k) \end{bmatrix} \tag{21}$$

As it is quite clear from (21), the function f is a non-linear function. It is because of this property that we have to use a filtering technique for a non-linear system.

In the previous section we defined one single observation equation and so assumed we had only one target. In actuality, it is most often the case that we will have more than one target of different types. Thus, we modify (2) as follows:

$$\mathbf{z}_i(k) = h_i(p_i, \mathrm{x}(k)) + w_i(k) \tag{22}$$

where $\mathbf{z}_i(k)$ is the observation of target p_i at time k. The non-linear measurement function $h_i(p_i, \mathrm{x}(k))$ is different for each type of target. (refer to section 4.3 for the exact observation equations)

Since we know the exact formulation of our transition function f, we can now compute the Jacobian

$$f_x(k) = \begin{bmatrix} 1 & 0 & -T(k)\sin\theta(k) \\ 0 & 1 & T(k)\cos\theta(k) \\ 0 & 0 & 1 \end{bmatrix} \tag{23}$$

As we have modified (2) to handle more than one target of more than one types, we modify the estimate of the predicted target observation.

$$\hat{\mathbf{z}}(k+1|k) \approx h_i(p_i, \hat{\mathrm{x}}(k+1|k)) \tag{24}$$

Since each observation is independent of each other, we should have a separate filter gain for each target.

$$W_j = P(k+1|k)\nabla' h_j S_j^{-1}(k+1) \tag{25}$$

where ∇h is another way to denote the Jacobian.

Thus, to obtain the optimal estimate we simply add all the Kalman Gain Matrices of different targets.

$$\hat{\mathrm{x}}(k+1|k+1) = \hat{\mathrm{x}}(k+1|k) + \sum_j W_j(k+1)[\mathbf{z}_j(k+1|k) - h_j(p_j, \hat{\mathrm{x}}(k+1|k))] \tag{26}$$

Similarly for the covariance,

$$P(k+1|k+1) = P(k+1|k) - \sum_j W_j(k+1)S_j(k+1)W_j'(k+1) \tag{27}$$

3 Louie the robot

Here we describe in detail the physical aspects of our robot.

3.1 Base

Louie has the Real World Interface three-wheeled circular robot base as its actuator. This base is 12 inches in diameter and is 7 inches in height. The wheels are built in such a way that the robot can turn in place.

3.2 Sensors

Louie has two sensors: the actuator sensors which returns the distance the wheels have traveled and the rotational degrees when inquired. For the range sensors we have installed 8 Polaroid Ultrasonic Rangefinders in an arrangement as shown in Figure (2). We have divided the 8 sonars into 4 pairs of 2 sonars each, where each pair is placed 18 cm apart in the same plane. Hence forth we shall call such a pair, a *sonar plane.* Consequently, we have four sonar planes, each plane covering one of the four directions in the robot's coordinate system: $0°$,$90°$, $180°$, and $-90°$.

4 The sonar model

As pointed out in [4], the key to interpreting sensor information correctly, is to have a good model of sensor behavior. A perfect sonar model would be able to exactly predict what sensor data can be observed from any given configuration of objects. A good model should be able to predict expected targets and explain unexpected targets. It should also suggest what information should be extracted from the sensor data, how it relates to physical features and how it should be used to construct and maintain a map of the environment.

The peculiarities of sonars have been discussed in several papers [6, 7]. According to the theory of acoustics, the angle of reflection from a specular surface is equal to the angle of incidence. Thus, as shown in Figure 3a, only sonar beams with near perpendicular incidence angles receive enough sound energy reflected directly from target to produce correct distance readings.

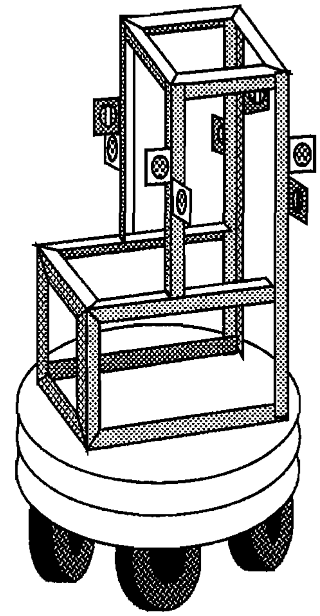

Figure 2:

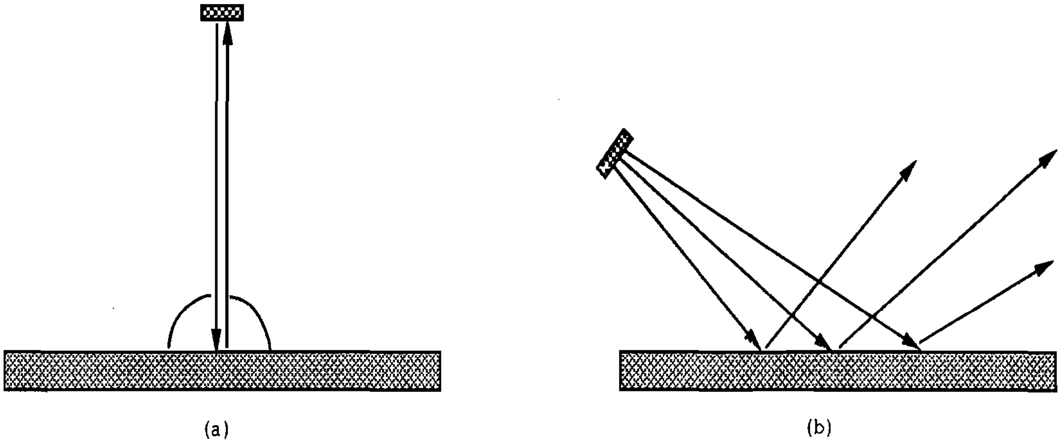

Figure 3:

As the angle of incidence changes to more oblique angles (as shown in Figure 3b) most of the beams are reflected away from the surface and hence the amount of sound energy reflected directly back from the target to the sonar transducer decreases. Thus, a much longer than actual distance reading is produced since the majority of the sonar energy the transducer detects at this point are the beams that were reflected off more than one surface. It is important that we discern true readings from these false readings produced by multiple reflections. Figure 4 is a typical example exhibiting such properties. The robot was placed at one end of the hallway and was rotated 360° to accumulate 120 readings acquired at intervals of 3°s. It can be easily observed that at areas where the sonar and the wall comprised an angle that was not anywhere close to perpendicular (angles less than $90° - 15°$ or greater than $90° - 15°$), false readings were produced. This is due to specular reflections as it has been previously mentioned.

4.1 Regions of Constant Depth

The concept of Regions of Constant Depth (RCD) was introduced in [8]. When a sonar scan is performed (as shown in figure 4), one can clearly observe that there are regions which have the constant distance readings (constant depth). Not all such regions have correct readings, however. What qualifies a region as a valid RCD is the width. The wider the width, the more reliable the range readings are. Therefore, we treat only those RCDs which have width that is wider than some threshold. RCDs that do not pass the threshold should be ignored. Henceforth, we shall denote only those regions that pass the threshold as RCDs. We have found that these regions tend to occur when sonars scan areas that have good and wide reflective property (e.g. walls and other similar flat objects with surface having reasonable reflective property). When the sonar scan produces an unstable reading (i.e. regions that do not have constant depth) or have very narrow RCD, it is either because it is aiming at a target at a *bad*[‡] angle, producing a 2nd or higher-order reflections[§] resulting in a false range reading, or because it is aiming at a weak target such as a convex corner where a diffracted echo is

[‡] angle where the sonar and the target are no longer close to perpendicular; producing higher-order reflections

[§] the order of a reflection is the number of surfaces the sonar beam has reflected off before returning to the transducer

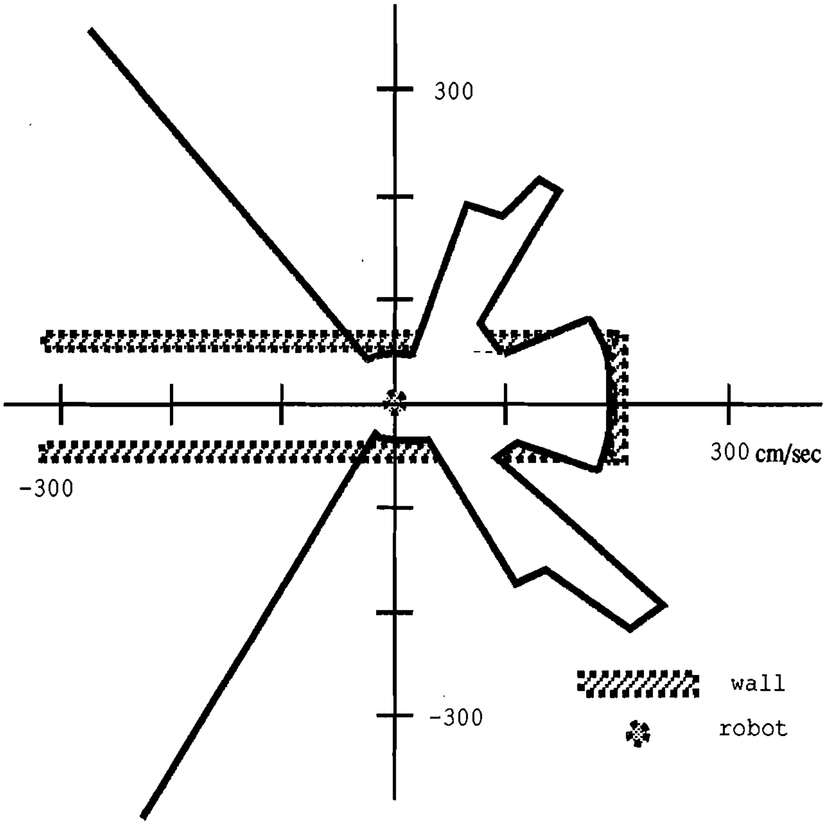

Figure 4: a sonar scan of a hallway

returned. It is hard to distinguish these different cases, since when a sonar reading is made, all that is returned is just a number indicating the distance. But we do know that robust targets provide wide RCDs and therefore makes them well observable.

If we take advantage of this fact and restrict ourselves to treat only such robust features as targets, we are less likely to misinterpret the sensor data. But using only valid RCDs is not enough to ensure correct interpretation. There are still cases, especially with lower-order reflection, when the order of the reflection is greater than 1 but makes a valid RCD. In such a case, we must somehow distinguish between 1st-order and other higher-order reflections. For, only 1st-order reflections provide an accurate, correct range of the target. In a later section, we discuss how we overcome this problem.

There is one possible disadvantage about using RCD's. It is the problem of having to acquire a complete scan of the environment. A scan of at least 90° is necessary to acquire enough data to generate a RCD. Having a ring of sonars is one possible solution. But even with such a ring of 8, 16 or more sonars, it is usually the case that never more than half the number of sonars are fired in the same step in fear of the sonars interfering each other's beam. Since our robot system has sonars in a rectangular format, a rotating scan is unavoidable. This can be quite costly in terms of both time and data processing.

Using the layout of our sonar configuration, it is possible to partly overcome this inefficiency. Instead of performing a rotating scan everytime, we use the pairs of sonars in each plane to imitate the effect of rotating. If for some contiguous time step, both sonars in the same plane observe the same target with reasonably equivalent distance readings, we can conclude that the target is at least the width of the displacement between the sonars and thus have an RCD of at least that width.

4.2 Target Classification and Measurement

Given that we are navigating in an indoor environment, we can basically classify targets into two types: WALLS and CORNERS. Because walls are quite stationary and unchanging (unless you have a building where the walls are movable), they are a good source of reliable information when it comes to localization. Thus, for the current implementation, we limit ourselves to targets of Wall types.

First of all, it is important that we recognize the right type of target since the distance measuring functions h_i is different for each type. Secondly, it is crucial that we do indeed calculate the correct distance to the target.

Distinguishing Target Types:
We consider a target to be a wall when the width of the RCD is approximately 30°. In which case we say the *observable angle* β_i is 30°. This part is not absolutely necessary since from our global map, we already know what type of target we are looking at.

Measuring minimum distance to wall: If we consider WALL as a line in a 2-D coordinate system, it can be defined as a vector in Hessian normal form (P_R, P_θ). P_R is the minimum distance from the (infinite) line to the origin of global coordinate frame. P_θ is the angle (counterclockwise) between the positive x-axis and the orthogonal line drawn from the wall to the origin (See figure 5).

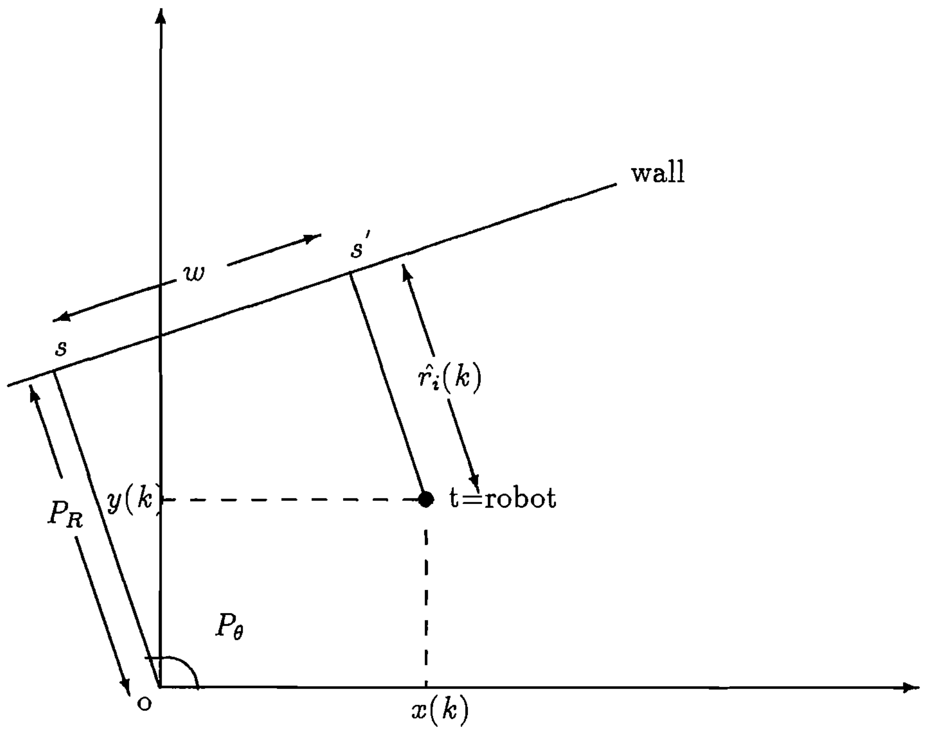

Figure 5:

If we represent os,ss', ot,ts' as vectors, using vector addition we can derive the following equality:

$$\vec{os} + \vec{ss'} = \vec{ot} + \vec{ts'}$$

Representing each of the vectors in polar coordinates we can rewrite this as follows:

$$P_R \begin{pmatrix} -\cos P_\theta \\ sin P_\theta \end{pmatrix} + w \begin{pmatrix} \sin P_\theta \\ \cos P_\theta \end{pmatrix} = \begin{pmatrix} x(k) \\ y(k) \end{pmatrix} + \hat{r}_i(k) \begin{pmatrix} -\cos P_\theta \\ \sin P_\theta \end{pmatrix} \tag{28}$$

From the 1st row of 28, we are able to derive

$$\hat{r}_i(k) = P_R - w\frac{\sin P_\theta}{\cos P_\theta} + \frac{x(k)}{\cos P_\theta} \tag{29}$$

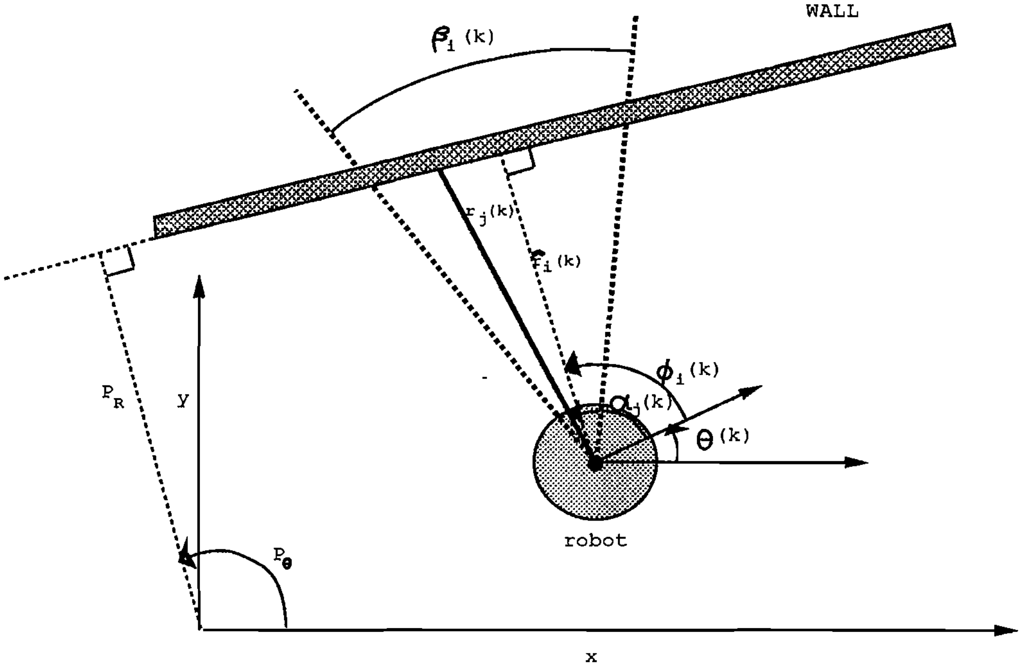

Figure 6:

From the 2nd row, we have

$$\hat{r}_i(k) = P_R + w\frac{\cos P_\theta}{\sin P_\theta} - \frac{y(k)}{\sin P_\theta} \tag{30}$$

Solving for $\hat{r}_i(k)$ from 29 and 30,

$$\hat{r}_i(k) = \mid P_R - x(k)\cos(P_\theta) - y(k)\sin(P_\theta) \mid \tag{31}$$

where (also see Figure 6)

$$P_\theta = \phi_i(k) + \theta(k) \tag{32}$$

4.3 The Correspondence Problem

As discussed in an earlier section, problem arises when sonars aim at targets at bad angles. This is when reflections of higher order occur. For such a reading, it is best not to believe the data at all. We can achieve this by checking at each step whether the sonar reading corresponds to the real target or not. We know the reading does not correspond to the actual target range when the angle of $r_j(k)$ does not fall in the β_i region. More formally this is possible, if we know

- the width of the angle during which the target is observable by a sonar. We shall call this the *observable* angle β_i.
- the true bearing to the target from the robot's direction of travel, at which point the minimum distance to the target is measured. We denote this angle as $\phi_i(k)$. Since we have prior knowledge about the initial position (including the orientation $\theta(k)$ of the robot relative to the global map) and the angle P_θ of the target can be easily calculated using 32.

Now a sonar reading is in correspondence with the target if the sonar's actual bearing, which we denote as $\alpha_j(k)$, is within the observable area. More formally,

$$\phi_i(k) - \frac{\beta_i}{2} \leq \alpha_j(k) \leq \phi_i(k) + \frac{\beta_i}{2} \tag{33}$$

where $\phi_i(k)$ and β_i is are defined above.

Because our sonars are fixed by hardware, as mentioned in section 3.2, it is possible to know the value of $\alpha_j(k)$ (the orientation of each sonar plane) a priori. From our sonar model, we also know β_i a priori, as long as we know what type of target we are looking at. $\phi_i(k)$ can be easily obtained from 32. Thus, we are able to determine at each step in time the correspondence of a sonar reading to a target.

5 Implementation

Our tracking program has two phases; the Setup phase and the Tracking phase. The setup phase is executed only once in the beginning of the navigation but the tracking phase continue as long as the robot is traveling. In the setup phase, we perform a 90° scan. During the tracking phase, we take advantage of the fact that we have two sonars in the same plane placed apart by some distance and achieve the same effect of a rotating scan by using the sonar pairs to validate continuous observation of the RCD of a target.

The Setup Phase

During this initial phase, the program collects all the information that is needed for tracking and initializes various parameters accordingly. In our implementation, we perform a minimum of 90° scan once during this phase.

- ACQUIRING SONAR DATA

  All the sonar data that is acquired during the initial 90° scan is stored for further analysis. Typically, a scan at the speed of 3° per second produces 30 readings in total for each of the eight sonars. It is up to the user to choose the speed of rotation and the rotation amount but a minimum rotation of 90° is necessary to ensure correct target classification. Since we have the global map a priori and we know the position of the robot, it is possible to omit this stage completely and only rely on the a priori knowledge. But in practice, it is not easy to provide accurate position and distance information. Thus, here we left it to the robot to determine the exact position and minimum distance P_R of the intial set of targets. We do give to the robot approximate information of where the targets are and for how much longer the targets will be visible.

- CATEGORIZATION

  For each single sonar, the 30 readings are sorted into five buckets. The boundaries of these five buckets are chosen differently for each trial depending on the minimum range read during the scan.

  Shown in Figure 7 is an example of a real run showing the results of such sort.

  By default, the first (or lowest) boundary is set to the minimum distance encountered during the scan. Then, the next four boundaries are calculated as offsets of 10, 20, 30, 35 from the lowest boundary.

own sonar model using experimental results.

7 Future work

If we were to point a downfall in the EKF technique, it would be the fact that the algorithm relies quite heavily on a priori accurate knowledge of global position information. We tried to overcome in part by inserting the Setup Phase in which we try to make the robot figure out for itself exactly where and how far the targets are from the robot. But in order to do this, we need the minimum distance from the robot to the target. And to extract the minimum distance, we have to perform a rotating scan of at least 90°. We can probably overcome this either by changing the sonars to *active* sensors or by adding another type of sensor to provide some local support.

When computing the optimal estimate and the covariance, we could apply some weighting function to each Kalman Gain Matrix of a target as we gather the sum. Depending on what type of target the Kalman Gain Matrix was obtained from or how confident the robot is about a particular reading, the significance of the filter gain could vary. For stable targets like walls, we could increase the weight. For unstable targets such as convex corners, we could decrease the gain to allow this particular observation to affect the overall estimate as little as possible.

Basically, we must think of ways where we will need less a priori knowledge.

References

[1] Bar-Shalom, Y. and Fortmann, T. E. *Tracking and Data Association.* Academic Press, 1988.

[2] Brammer, Karl and Siffling, Gerhard. *Kalman-Bucy Filters.* Norwood, MA., Artech House, Inc., 1989.

[3] Kalman, R.E. and Bucy, R.S. New results in linear filtering and prediction theory, *Trans. ASME, Ser. D, J. Basic Eng.,* 83: 95-108, 1961.

[4] H.F. Durrant-Whyte. Sensor models and multi-sensor integration. *International Journal of Robotics Research,* 7(6): 97-113, 1988.

[5] Gelb, Arthur. *Applied Optimal Estimation.* Cambridge, Mass., MIT Press, 1974.

[6] R. Kuc and B. Barshan. Navigating vehicles through an unstructured environment with sonar. In *Proc. IEEE Int. Conf. Robotics and Automation,* 1422-1426, May 1989.

[7] R. Kuc and M. W. Siegel. Physically based simulation model for acoustic sensor robot navigation. *IEEE Trans. on Pattern analysis and Machine Intelligence,* PAMI-9(6):766-778, November 1987.

[8] Leonard, John J. and Durrant-Whyte, Hugh F., "A Unified Approach to Mobile-Robot Navigation", *Technical Report,* Department of Engineering Science, University of Oxford, September 1989.