A Rapidly Converging Recursive Method for Mobile Robot Localization

Daniel Boley

University of Minnesota
Minneapolis, MN 55455

Karen Sutherland

University of Wisconsin – La Crosse
La Crosse, WI 54601.

ABSTRACT

This paper proposes a simple method for estimating the position of a robot from relatively few sensor readings. Our algorithms are intended for applications where sensor readings are expensive or otherwise limited and the readings that are taken are subject to considerable errors or noise.

Our method is capable of converging to a position estimate with greater accuracy using fewer measurements than other methods often used for this type of application, such as the Kalman and extended Kalman filters. Our approach is validated using a mobile robot on which a camera is used to obtain bearing information with respect to landmarks in the environment.

[0]This research was supported in part by NSF grant CCR-9405380 and Minnesota Dept. of Transportation grant 71789-72996-173.

1. Introduction and Background

1.1. Introduction

It is often the case that the environment in which a mobile robot must navigate is not conducive to the taking of multiple sensor readings. One example is an outdoor, unstructured environment in which beacons are non-existent and the few existing natural landmarks are widely spaced and a significant distance from the viewpoint. We propose a simple method for successfully navigating in environments of this type, where sensor readings are expensive or otherwise limited and the readings that are taken are subject to considerable errors or noise.

The method we propose converges rapidly using only a few readings, would be exact if there were no errors in the data, and is relatively efficient. We use a variation on a least squares formulation to minimize a residual, and carry out the computations in a recursive manner. Our simple least squares formulation not only gives quite satisfactory results, but converges faster than the commonly used Kalman filter at approximately the same cost per step. In addition, it does not suffer from the problem of non-convergence often seen when the extended Kalman filter is used in this type of application.

To demonstrate the algorithms, we use a mobile robot platform on which is mounted a camera. The sensor readings are obtained by viewing one or more landmarks in the visual images obtained as the robot moves. The images yield bearings to the landmarks, which are then used to estimate the position of the robot. The bearings are subject to considerable noise, both from the coarseness of the image resolution and from odometry error in fixing the base line. In spite of the noise in the data, our algorithms were able to fix the location of the robot with relatively high accuracy.

We show how the task of estimating robot positions from bearing data can be formulated directly as a simple overdetermined system of linear equations, which is not a local linearized approximation, but is valid globally. We then show how the resulting system of linear equations can be solved in either a least squares sense or by using a total least squares approach, which has the advantage over least squares of admitting errors anywhere in the equations.

This paper is organized as follows. After the rest of this introduction, in which we describe the limitations of the Kalman filter for this particular application, we define in Section 2 the navigation problem we wish to address and show how it can be set up as a set of linear equations which would be exact if the data were error-free. In Section 3 we review the theory which guarantees the existence, uniqueness and sensitivity of the solution to the overdetermined system of equations set up in Section 2. In Section 4 we show how the system of linear equations can be solved efficiently in a recursive manner, in both a least squares (LS) and total least squares (TLS) sense. In Section 5 we illustrate the behavior of the method with some experiments, and we end in Section 6 with some concluding remarks.

1.2. The Kalman Filter and Extended Kalman Filter

The discrete Kalman filter (Kalman 1960), commonly used for prediction and detection of signals in communication and control problems, has more recently become a popular method of reducing uncertainty in robot navigation. One of the main advantages of using the filter is that it is recursive, eliminating the necessity for storing large amounts of data. The filter is basically a recursive weighted least squares estimator of the state of a dynamical system using a given transition rule. Suppose we have a discrete dynamical system $\mathbf{x}_i = F_{i-1}\mathbf{x}_{i-1} + e_{i-1}$, where $\mathbf{x}_i$ is the state vector, e_i is the noise vector, and F_{i-1} is the state transition matrix at time step i. We are given a sequence of measurements $\mathbf{b}_i$ obeying the model $\mathbf{b}_i = A_i\mathbf{x}_i + \epsilon_i$, where A_i is the given data matrix and ϵ_i is measurement noise. The Kalman filter is used to find an estimate of the state vector $\mathbf{x}_i$ from the measurement data that minimizes the noise in a least squares sense. The Kalman filter equations and a schematic diagram of the filter are in the Appendix. A complete description of the filter can be found in (Gelb 1974). It requires an initial estimate of the solution and assumes that noise is weighted white gaussian. The discrete Kalman filter is guaranteed to be optimal in that it is guaranteed to find the best solution in the least squares sense.

Although originally designed as an estimator for dynamical systems, the filter is used in many applications as a static state estimator (Smith & Cheeseman 1986). In the static problem, the state transition matrix F_{i-1} is the identity matrix I, so the problem is reduced to finding the state vector $\mathbf{x}$ minimizing the weighted Euclidean norm of the measurement noise

$$\|W\epsilon_i\|_2 = \|W(\mathbf{b}_i - A_i\mathbf{x}_i)\|_2,$$

where W is an optional weighting matrix (usually the inverse of the covariance matrix of measurement noise).

Also, due to the fact that functions are frequently non-linear, the extended Kalman filter (EKF) is used (Ayache & Faugeras 1989; Kosaka & Kak 1992). The EKF formalism linearizes the function by taking a first order Taylor expansion around the current estimate of the state vector (Gelb 1974). Assuming that the function is represented by a set of non-linear equations of the form $f_i(\mathbf{y}_i, \mathbf{x}) = 0$ where $\mathbf{x}$ is the state vector and $\mathbf{y}_i$ represents random parameters of f_i of which estimated measures, $\hat{\mathbf{y}}_i$, are taken, the first order Taylor expansion is given by:

$$f_i(\mathbf{y}_i, \mathbf{x}) = 0 \simeq f_i(\hat{\mathbf{y}}_i, \hat{\mathbf{x}}_{i-1}) + (\mathbf{y}_i - \hat{\mathbf{y}}_i)\frac{\partial \hat{f}_i}{\partial \mathbf{y}} + (\mathbf{x} - \hat{\mathbf{x}}_{i-1})\frac{\partial \hat{f}_i}{\partial \mathbf{x}}$$

where $\hat{\mathbf{x}}_i$ is the i-th estimate of the state vector and the derivatives are estimated at $(\hat{\mathbf{y}}_i, \hat{\mathbf{x}}_{i-1})$. This equation can be rewritten as:

$$\mathbf{b}_i = A_i\mathbf{x} + \epsilon_i$$

where:

$$\mathbf{b}_i = -f_i(\hat{\mathbf{y}}_i, \hat{\mathbf{x}}_{i-1}) + (\hat{\mathbf{x}}_{i-1})\frac{\partial \hat{f}_i}{\partial \mathbf{x}}$$

$$A_i = \frac{\partial \hat{f}_i}{\partial \mathbf{x}}$$

$$\epsilon_i = (\mathbf{y}_i - \hat{\mathbf{y}}_i)\frac{\partial \hat{f}_i}{\partial \mathbf{y}}$$

This linear approximation function is then used as the Kalman filter equation.

1.3. Limitations of the Kalman Filter

There are several basic problems which can occur when using either the Kalman or extended Kalman filter in robot navigation applications:

- The filter was developed for applications such as those in signal processing in which many measurements are taken (Kalman 1960). Sensing in robot navigation is often done using camera images. The gathering and processing of each image is a time consuming process so a successful method must make do with relatively few readings.

- The Kalman filter assumes a starting estimate is available. Convergence can be adversely affected by a poor starting estimate. The method we propose does not require any starting estimate, but rather is capable of producing one with relatively few readings.

- The Kalman filter implicitly assumes the errors in the data are Gaussian or approximate a Gaussian distribution. The accuracy can be degraded if this assumption is violated by, for example, the presence of systematic errors.

- An underlying assumption in least squares estimation is that the entries in the data matrix are error-free (Golub & Van Loan 1989). In many actual applications, the errors in the data matrix can be at least as great as the measurement errors. In such cases, a *Total Least Squares* (TLS) approach can give better results.

Two additional problems occur when using the EKF:

- The linearization process itself has the potential to introduce significant error into the problem.
- The EKF is not guaranteed to be optimal or to even converge (Sorenson 1970). It can easily fall into a local minimum when an initial estimate of the solution is poor, often the type of situation faced by robot navigators.

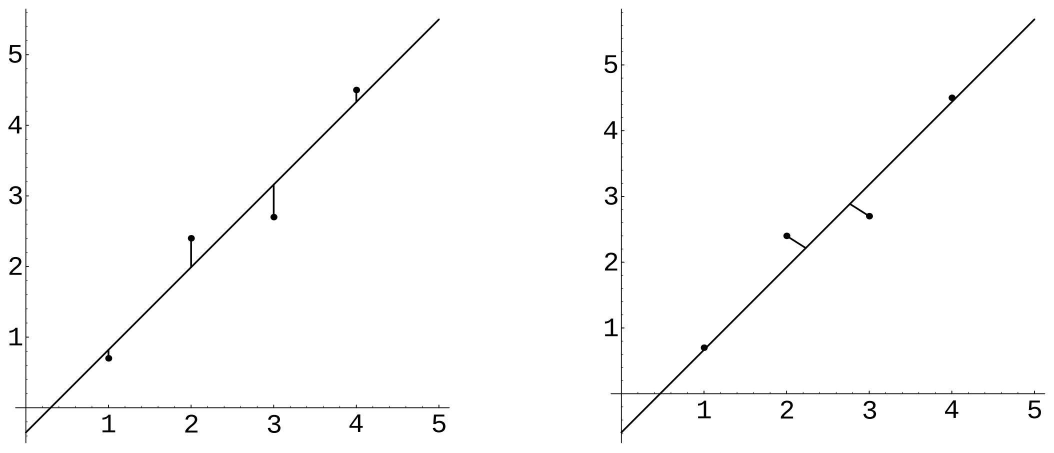

Figure 1: (a) In an LS solution, the sum of the squared vertical distances to the line of best fit is minimized. (b) In a TLS solution, the sum of the squared perpendicular distances to the line of best fit is minimized.

Although limited modifications can be made to the Kalman approach to improve robustness to noise (Schneiderman & Nashman 1994), our work in outdoor navigation (Sutherland & Thompson 1993), where measurements are expensive to obtain and have very significant nongaussian error inherent to the system, motivated us to look for another filtering method. Navigation algorithms such as those developed by (Hanebeck & Schmidt 1996) using a set theoretic framework are able to handle nongaussian noise. However, they require that a significant number of landmarks be available. We sought a method capable of converging with only a few measurements from as few as one or two landmarks.

As the interesting work by Mintz et. al. (Hager & Mintz 1991; McKendall & Mintz 1990) in robust estimation and modeling of sensor noise has demonstrated, the criterion of optimality depends critically on the specific model being used. Given two methods, the first may produce optimality in one sense but not do as well as the second in another sense. When error exists in both the measurement and the data matrix, the best solution in the *least squares* sense is often not as good as the best solution in the *eigenvector* sense, where the sum of the squares of the perpendicular distances from the points to the lines are minimized (Duda & Hart 1973) (Fig. 1). This second method is known in the statistical literature as *orthogonal regression* and in numerical analysis as *total least squares* (TLS) (Van Huffel & Vandewalle 1991).

2. Problem Formulation

2.1. Single Landmark

We first discuss the case where one uses bearings to a single landmark. In order to fix the position in a ground coordinate system, it is necessary to assume that we can obtain the direction of motion ("vehicle heading") from another sensor. This case is useful if, for example, the vehicle has a compass yielding directional information. It also serves to illustrate the general paradigm.

2.1.1. Equation for Single Landmark at Each Time Interval

We set up a virtual coordinate system with the landmark located at $(0, 0)$, and the x coordinate along the direction of motion of the robot, as illustrated in Figure 2.

At any step i, we have:

$$cot(\alpha_i) = \frac{x_0 + d_i}{y_0}, \tag{1}$$

where (x_0, y_0) is the unknown robot start position, α_i is the measured angle, and d_i is the distance traveled by the robot since starting at position (x_0, y_0). The distance d_i may be estimated from odometry or by integrating the velocity over time. The goal is to solve for the unknown coordinates (x_0, y_0). We can rewrite (1) as

$$sin(\alpha_i) \cdot x_0 - cos(\alpha_i) \cdot y_0 = -sin(\alpha_i) \cdot d_i. \tag{2}$$

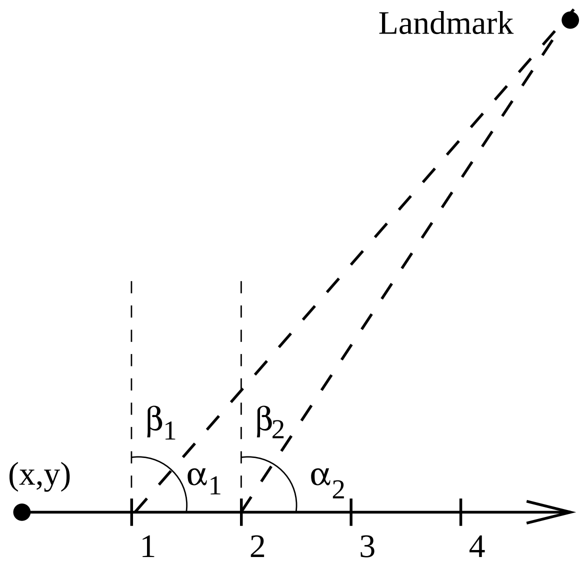

Figure 2: *Coordinate system for single landmark situation.*

2.1.2. Row by Row Collection into Overdetermined Matrix Equation $A\mathbf{x} = \mathbf{b}$.

Over several time intervals, we read new angles yielding new coefficients for the equation (2). We assemble all these equations into a single matrix equation of the form $A\mathbf{x} = \mathbf{b}$ where each row of A and $\mathbf{b}$ are respectively

$$\mathbf{a}_i^T = (\, sin(\alpha_i) \quad -cos(\alpha_i)\,)\,, \quad b_i = -sin(\alpha_i) \cdot d_i, \tag{3}$$

and $\mathbf{x} = (x_0, y_0)$ is the vector of unknowns. Since the angle readings are subject to noise, we must solve the equations $A\mathbf{x} = \mathbf{b}$ in a least squares sense. This is discussed below. It will be seen that we can encode this information in such as way that the amount of storage does not grow as more readings are entered.

Notice that equations $A\mathbf{x} = \mathbf{b}$ set up in this way are linear in the unknowns (x_0, y_0), but we have not made any kind of linearized approximation. If there were no error in the angles α_i or the distances d_i, the true start position would *exactly* satisfy the equations.

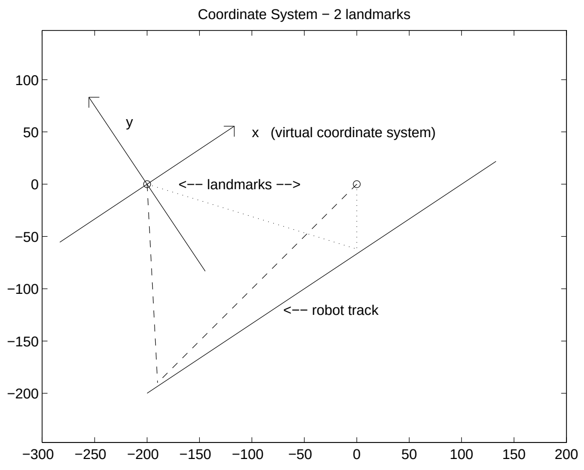

Figure 3: *Two Landmark Virtual Coordinate System.*

2.2. Two Landmarks

2.2.1. Equations Using Two Landmarks.

We show how to set up the equations analogous to (3), but for two landmarks. In general, two landmarks are sufficient to determine one's position without using any prior knowledge on direction of motion, once one has taken readings from at least two positions. In handling two landmarks, we set up a virtual coordinate system in which one landmark is placed at the origin as before, the x axis is placed parallel to the direction of motion, and the second landmark is placed at coordinates (l, m). The unknown quantities to be solved for are the starting robot position (x_0, y_0) and the position of the second landmark (l, m). The coordinates (l, m) are then used to map the virtual coordinate system to the ground coordinate system using the known ground coordinates of the two landmarks. The coordinate system is illustrated in Figure 3.

Applying the single landmark equations to each landmark here, we obtain the equations

$$sin(\alpha_{1i}) \cdot x_0 - cos(\alpha_{1i}) \cdot y_0 = -sin(\alpha_{1i}) \cdot d_i, \tag{4}$$

$$sin(\alpha_{2i}) \cdot x_0 - cos(\alpha_{2i}) \cdot y_0 - sin(\alpha_{2i}) \cdot l + cos(\alpha_{2i}) \cdot m = -sin(\alpha_{2i}) \cdot d_i, \tag{5}$$

2.2.2. Collection into Overdetermined Matrix Equation.

Equations (4) and (5) yield two rows for the matrix least squares problem $A\mathbf{x} = \mathbf{b}$ we have to solve, where the vector of unknowns is $\mathbf{x}^T = (x_0, y_0, l, m)$. We append the following two rows to the coefficient matrix A

$$A_i = \begin{pmatrix} \mathbf{a}_{1i}^T \\ \mathbf{a}_{2i}^T \end{pmatrix} = \begin{pmatrix} sin(\alpha_{1i}) & -cos(\alpha_{1i}) & 0 & 0 \\ sin(\alpha_{2i}) & -cos(\alpha_{2i}) & -sin(\alpha_{2i}) & cos(\alpha_{2i}) \end{pmatrix} \tag{6}$$

and two new entries to the right hand side vector $\mathbf{b}$

$$\mathbf{b}_i = \begin{pmatrix} b_{1i} \\ b_{2i} \end{pmatrix} = \begin{pmatrix} -sin(\alpha_{1i}) \cdot d_i \\ -sin(\alpha_{2i}) \cdot d_i \end{pmatrix}. \tag{7}$$

Notice we have again constructed a system of equations that are linear in the unknown quantities, but which are still exact in the sense that if the coefficients are obtained without error, the solution will be exact.

2.2.3. Collecting Rows when Only One Landmark is Visible at a Time.

Besides being linear in the unknowns, equation (7) also yields an interesting property when the landmarks are not visible at the same time. If only the first landmark is visible, it suffices to append only the first row of (6) and (7). When the second landmark is visible, it suffices to append only the second row of (6) and (7). Alternatively, a row of all zeroes can be used for the missing landmark. This comment also applies if the angles to each landmark are not read simultaneously. For example, if two different sensors are used, each devoted to its own landmark, and these two sensors are not synchronized, then we can append the appropriate row corresponding to a landmark whenever an angle to that landmark is obtained from the sensor, without regard to the other landmark. Each reading would include the correct current value for the distance d_i measured at the time the reading is taken. A unique least squares solution to $A\mathbf{x} = \mathbf{b}$ will exist as long as A has full column rank. In this case of two landmarks, this means that the rank of A should be 4, and this will occur when we have readings of at least two different angles from landmark 1 plus two different angles from landmark 2, regardless of the order in which the readings are obtained.

2.2.4. Mapping Back to Ground Coordinate System

Once we compute the estimated values for the unknown quantities x_0, y_0, l, m, the coordinates (l, m) can be used to map the virtual coordinate system back to the ground coordinate system. Suppose for simplicity that the ground coordinate system also has the origin at the first landmark, but with the directions of the axes defined a-priori (possible by a simple translation of the ground coordinate system). The distance between the landmarks in both coordinate systems must match in theory, and in practice a small mismatch in this inter-landmark distance can be used to correct scaling errors in the odometry used to obtain the distances d_i. As a result, it is not necessary to calibrate the odometry very accurately, as long as it is internally consistent. After this scaling, the two coordinate systems can be matched by rotation.

3. Least Squares and Total Least Squares: Theory

3.1. When Does Least Squares Problem Admit Unique Solution?

To account for noise in the data, we estimate the starting position using the best least squares fit. In this section we determine when the overdetermined system of equations we have constructed has a unique least squares solution by applying the general theory for least squares problems. When we state that we wish to solve the matrix equations $A\mathbf{x} = \mathbf{b}$ in a least squares sense, what we mean precisely is to minimize the residual $\min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2$. If there are no errors in the coefficients $A, \mathbf{b}$, then there should be a solution $\mathbf{x}$ for which the residual is zero (i.e. the equations are satisfied exactly). But in the presence of errors in the coefficients, the equations will not be satisfied exactly. The following is well known from the theory of linear least squares (Golub & Van Loan 1989, p222):

THEOREM T1. *The least squares problem*

$$\min_{\mathbf{x}} \|\mathbf{r}\| = \min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2 \tag{8}$$

admits a unique solution if and only if A has full column rank. If A is rank deficient, then there are multiple solutions. □

The issue we address here is under what conditions does the A matrix constructed by either (3) or (6) have full column rank.

In the single landmark case, if we are moving directly toward or away from the landmark, then the angles will be constant. Hence every row of A will be the same and the rank of A will be 1, not 2. On the other hand, it is simple matter to see that if we have at least two readings with at least two different angles, then the corresponding rows computed according to (3) will be linearly independent, and hence A will have full column rank 2.

In the two landmark case, a similar argument applies. If we have readings with at least two different angles to landmark 1 and two different angles to landmark 2, for a total of at least 4 readings, then the the 4 rows computed according to (6) will be linearly independent and the resulting matrix A will have full column rank 4. If, however, we are moving directly toward one of the landmarks in such a way that the angles remain constant, then the rows generated according to (6) corresponding to that landmark will all be the same. The rows corresponding to the other landmark taken alone yield a rank of at most 2, so the entire matrix A can have a rank of at most 3.

So we may conclude the following

THEOREM T2. *The least squares problems constructed from the single landmark case (3) or the two landmark case (6), (7) have unique solutions if and only if the robot vehicle is not moving directly toward or away from either landmark, and readings with at least two different angles are obtained.* $\square$

3.2. Sensitivity of Least Squares Solution

We examine the sensitivity of the solution to the Least Squares problem, assuming that it is a full rank problem satisfying the conditions of Theorems T1 and T2. All the algorithms in this paper are *backward stable* in the sense that they compute the exact solution for a problem that is a small perturbation of the problem originally presented to the algorithm. Thus there are two sources of perturbations to the "exact" set of equations to be solved: sensor errors and round-off errors. Both types of perturbations are subject to amplification due to the conditioning of the system of matrix equations, as shown by the following result (Golub & Van Loan 1989, p228):

THEOREM T3. *Consider the exact and perturbed least squares problem:*

$$\min_{\mathbf{x}} \|\mathbf{r}\| = \min_{\mathbf{x}} \|A\mathbf{x} - \mathbf{b}\|_2 \quad \textit{and} \quad \min_{\hat{\mathbf{x}}} \|\hat{\mathbf{r}}\| = \min_{\hat{\mathbf{x}}} \|(A + \Delta A)\hat{\mathbf{x}} - (\mathbf{b} + \mathbf{\Delta b})\|_2 \tag{11}$$

where A is $n \times p$ with $n \geq p$ with singular values $\sigma_1 \geq \cdots \geq \sigma_p > 0$. Assume that

$$\epsilon = \max\left\{ \frac{\|\Delta A\|}{\|A\|}, \frac{\|\mathbf{\Delta b}\|}{\|\mathbf{b}\|} \right\} < \frac{\sigma_p}{\sigma_1} \quad \textit{and} \quad \frac{\|\mathbf{r}\|}{\|\mathbf{b}\|} \equiv \sin\theta < 1. \tag{11a}$$

Then the error in the solution to the perturbed problem relative to the solution to the exact problem is

$$\frac{\|\hat{\mathbf{x}} - \mathbf{x}\|}{\|\mathbf{x}\|} \leq \left\{ \frac{2}{\cos\theta} \cdot \frac{\sigma_1}{\sigma_p} + \tan\theta \cdot \left(\frac{\sigma_1}{\sigma_p}\right)^2 \right\} \cdot \epsilon. \tag{11b}$$

$\square$

We interpret this result by noting first that the sensitivity depends on the inverse of the *condition number* $\kappa(A) \equiv \sigma_1/\sigma_p$. The smaller is this condition number, the less sensitive the least squares solution is to perturbations in the coefficients, whether from round-off errors or sensor errors. Thus, there are three factors giving rise to errors in the least squares solution: conditioning of the matrix, sensor errors yielding errors in the coefficients, and round-off arising during computations. Of these three factors, the round-off error is typically much less than the sensor errors, especially if normal floating point is used. The conditioning of the matrix can be controlled by increasing the range of angles over which the landmarks are visible. If the vehicle is moving almost but not exactly directly toward a landmark, a solution will exist in theory, but the matrix will be very ill-conditioned and the computed solution may be subject to much uncertainty. We note finally that whether or not any of these factors can be controlled, the residual vector obtained from an actual computed solution can be used to measure the impact of these factors in any particular case. Specifically, the computed residual yields the angle θ from (11a) which can be used to compute the upper bound to the relative error in (11b).

3.3. Total Least Squares – Application and Motivation

The ordinary least squares solution assumes all the error lies in the right hand side vector $\mathbf{b}$. The errors arise from both the distance data and the angle data. Hence the matrix A is subject to the same sources of errors as the right hand side. To allow for errors in the matrix A, we want to find the smallest perturbation $E, \mathbf{f}$ to both A and $\mathbf{b}$ so that the perturbed system $(A+E)\mathbf{x} = \mathbf{b} - \mathbf{f}$ has a solution. This is in contrast to ordinary least squares in which we find only the smallest $\mathbf{f}$ such that $A\mathbf{x} = \mathbf{b} - \mathbf{r}$ has a solution, the $\mathbf{r}$ being the residual. Admitting both perturbations $E, \mathbf{f}$ gives rise to the Total Least Squares (TLS) solution Specifically the TLS solution is obtained by finding the minimizing solution to

$$\min \|E, \mathbf{f}\|_2 \text{ such that } (A+E)\mathbf{x} = \mathbf{b} - \mathbf{f}. \tag{9}$$

In this way, we may admit errors anywhere in the system. Columns scaling may be used to weight errors in some columns more than others.

The easiest way to solve (9) is to use the Singular Value Decomposition (SVD). The SVD of an $n \times (p+1)$ matrix M is the factorization $U\Sigma V^T = M$ where U, V are orthogonal matrices and Σ is non-negative diagonal with the diagonal entries in descending order. For simplicity in exposition, assume without loss of generality that $n \geq (p+1)$. The diagonals $\sigma_1 > \cdots > \sigma_{p+1} \geq 0$ of Σ are the singular values, and the columns of $V = (\mathbf{v}_1, \ldots, \mathbf{v}_{p+1})$ are called the corresponding singular vectors.

3.4. When Does Total Least Squares Admit Unique Solution?

Regarding existence and uniqueness of a solution to (9), the theory of TLS (Bjorck 1996, p178) provides us with this theorem

THEOREM T4. *The system (9) admits a minimizing solution* $\mathbf{x}$ *if and only if the last component* $v_{p+1,p+1}$ *of the right singular vector* $\mathbf{v}_{p+1}$ *corresponding to the smallest singular value* $\sigma_{\min} = \sigma_{p+1}$ *of* $M = (A, -\mathbf{b})$ *is nonzero. The solution is unique if and only if this singular value has multiplicity 1. Then the TLS discrepancy is* $(E, \mathbf{f}) = -\sigma_{p+1}\mathbf{u}_{p+1}\mathbf{v}_{p+1}^T$ *and the* TLS solution *is* $\mathbf{x} = (v_{1,p+1}, \ldots, v_{p,p+1})^T / (v_{p+1,p+1})$. *If the smallest singular value is multiple, then the corresponding singular vectors are not uniquely defined, and any choice of those corresponding singular vectors will yield a TLS solution.* $\square$

The previous theorem gives very general technical conditions for the existence and uniqueness of solutions to the TLS problem based on quantities appearing during the computation of the TLS solution itself. It is more difficult to obtain such conditions in terms of quantities more easily available from the data, especially in the context of the robot localization problem we are considering. However, a sufficient condition for the existence and uniqueness of a TLS solution can be expressed in terms of the solution and residual, $\mathbf{x}, \mathbf{r}$ from the *ordinary least squares* problem (8).

THEOREM T5. *If the least squares residual* $\mathbf{r}$ *from (8) is sufficiently small to satisfy* $\|r\|/\|(x^T, -1)\| < \sigma_{\min}(A)$*, then the TLS solution is guaranteed to exist and be unique, where* $\mathbf{x}$ *is the corresponding least squares solution and* $\sigma_{\min}(A)$ *denotes the smallest singular value of* A*.*

PROOF: Let $\sigma_i, i = 1 \ldots, p$ and $\bar{\sigma}_i, i = 1, \ldots, p+1$ be the singular values of the matrix A and the augmented matrix $(A, \mathbf{b})$, respectively. From the interlacing property of the singular values $(\bar{\sigma}_i \geq \sigma_i \geq \sigma_{i+1})$,

$$\bar{\sigma}_p \geq \sigma_p > \left\| \begin{pmatrix} A & \mathbf{b} \end{pmatrix} \cdot \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \right\| \geq \bar{\sigma}_{p+1} \cdot \left\| \begin{pmatrix} \mathbf{x} \\ -1 \end{pmatrix} \right\|,$$

hence $\bar{\sigma}_{p+1}$ is strictly less than σ_p. This sufficient to guarantee a unique solution to the TLS problem (Bjorck 1996, p179). $\square$

To interpret this result, if the matrix A of coefficients is well conditioned enough (i.e. $\sigma_{\min}(A)$ is large enough) and the least squares residual (from (8)) is small enough (i.e. the effective noise introduced into the right hand side vector $\mathbf{b}$ is small enough), then the TLS solution exists and is unique. Here "small enough" means the least squares residual is small relative to the smallest singular value of A.

4. Efficient Solution of Overdetermined Matrix Problem

The least squares problem (8) can be solved by a variety of methods. The two most popular methods are based on, respectively, the Normal Equations and the QR Decomposition. We very briefly sketch the methods in their traditional form, and then later indicate how these methods may be adapted to recursive computation. Details can be found in (Bjorck 1996).

The Normal Equations arising from (8) is the $p \times p$ matrix equation

$$A^T A\mathbf{x} = A^T\mathbf{b}. \tag{12}$$

The solution of this system of equations yields the least squares solution to (8), assuming A has full column rank. This method has the disadvantage that the conditioning of $A^T A$ is the square of that of A.

The QR Decomposition of A is the factorization $QR = A$ where Q is an $n \times p$ matrix with orthonormal columns and R is a $p \times p$ upper triangular matrix. This can be computed by standard algorithms (Bjorck 1996, sec. 2.3–2.4). Once computed, the least squares solution can be obtained by solving the square, triangular system

$$R\mathbf{x} = Q^T\mathbf{b}. \tag{13}$$

This method has the advantage that R has the same conditioning as A, but it is slightly more expensive if A has many rows.

4.1. Recursive Solution of Least Squares Problem

When determining the robot vehicle's position, each set of angles gives rise to a new row in the matrix A. We note that the systems (12) or (13) to be solved to obtain the positions are all fixed size $p \times p$ as new rows are added to A. The key to the efficient solution to the least squares

step	*cost* *(non-recursive)*	*cost* *(recursive update)*
Assemble (12)	$np^2 + np$	
Update RHS in (12)		$2p$
Cholesky Factorization	$p^3/3 + p^2/2 + p/6$	$2p^2 + 10p$
two triangular solves	$2p^2 + 4p$	$2p^2 + 4p$
Totals	$np^2 + np + p^3/3 + 5p^2/2 + p/6$	$4p^2 + 16p$
when $p = 2$	$6n + 13$	48
when $p = 4$	$20n + 62$	128

Table 1: *Costs for Normal Equations in Floating Point Operations (Flops).*

problem is to show how to *update* the matrices and right hand sides in (12) and (13) without recomputing them from scratch.

4.1.1. Normal Equations, Updating $A^T A$ and $A^T\mathbf{b}$ at Each Step.

Let $A, \mathbf{b}$ be the matrix and right hand side obtained after several readings, and suppose we must now incorporate a new set of readings which form a new row $\mathbf{a}^T$ for A and a new entry b for $\mathbf{b}$. Observe that the updated matrix for the Normal equations,

$$\begin{pmatrix} A^T & \mathbf{a} \end{pmatrix} \begin{pmatrix} A \\ \mathbf{a}^T \end{pmatrix} = A^T A + \mathbf{a}\mathbf{a}^T, \tag{14}$$

is a rank one update on the original matrix for the Normal Equations. Likewise, the new right hand side consists of just the old right hand side plus a simple correction:

$$\begin{pmatrix} A^T & \mathbf{a} \end{pmatrix} \begin{pmatrix} \mathbf{b} \\ b \end{pmatrix} = A^T \mathbf{b} + \mathbf{a}b \tag{15}$$

Since the Normal Equations are a symmetric positive definite system, they are solved via the Cholesky factorization (Bjorck 1996, sec. 2.2.2), $A = LL^T$, which is essentially a symmetric version of Gaussian Elimination when pivoting is necessary, where L is lower triangular. Instead of updating the matrix $A^T A$, we would really like to update its Cholesky factor, and the cost of this is p^2 floating point operations (multiplies and adds). We refer the reader to (Bjorck 1996, sec. 3.2) for the details. Updating the right hand side costs $2p$ operations. Solving the new Normal Equations, given the new Cholesky factorization, requires two triangular solves costing a total of $2p^2$ operations. So the total cost for each update is approximately $3p^2 + 2p$ floating point operations. We summarize the steps in Table 1.

4.1.2. QR Method, Updating Q and R at Each Step.

The update process for the QR Decomposition is similar to that for the normal equations. Let $\widehat{Q}, \widehat{R}$ be the updated QR factors. Then

$$\widehat{Q}\widehat{R} = \begin{pmatrix} A \\ \mathbf{a}^T \end{pmatrix} = \begin{pmatrix} QR \\ \mathbf{a}^T \end{pmatrix} = \begin{pmatrix} Q & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R \\ \mathbf{a}^T \end{pmatrix}. \tag{16}$$

Moving Q to the left hand side we get a QR Decomposition of a $(p+1) \times p$ matrix:

$$\left\{ \begin{pmatrix} Q^T & 0 \\ 0 & 1 \end{pmatrix} \widehat{Q} \right\} \widehat{R} = \widetilde{Q}\widehat{R} = \begin{pmatrix} R \\ \mathbf{a}^T \end{pmatrix}. \tag{17}$$

The $\widetilde{Q}\widehat{R}$ factorization of (17) can be computed fast $(4p^3/3)$, independent of the number of samples taken), and in fact one can speed it up to $2p^2$ operations by taking advantage of the fact that R is already upper triangular (Bjorck 1996, p136). The right hand side is then updated by applying the updating transformation $\widetilde{Q}$ from (17) to the old right hand side $Q^T\mathbf{b}$:

$$\widehat{Q}^T \begin{pmatrix} \mathbf{b} \\ b \end{pmatrix} = \left\{ \widetilde{Q}^T \begin{pmatrix} Q^T & 0 \\ 0 & 1 \end{pmatrix} \right\} \begin{pmatrix} \mathbf{b} \\ b \end{pmatrix} = \left\{ \widetilde{Q}^T \begin{pmatrix} Q^T\mathbf{b} \\ b \end{pmatrix} \right\}. \tag{18}$$

If $\widetilde{Q}$ is stored as a sequence of the p elementary rotations arising from the QR Decomposition, this costs about $4p$ operations, so the total cost (including the one triangular solve) is approximately $3p^2 + 16p$ operations. We summarize the steps in Table 2.

4.1.3. Theorem: Fast Solution of Exact Least Squares Problem.

We note that the updating formulas (15)-(18) are all exact. Hence we have the following simple result, which we state formally in order to contrast these methods with other approximate methods. Both the recursive and non-recursive methods involve the computation of a factorization, in one case from scratch and in the other by updating using elementary transformations. The recursive algorithms yield the same exact factorizations as the non-recursive algorithms, except that the transformations are accumulated in a different way. So the results will differ only within rounding errors.

THEOREM T6. *The solutions to the recursively updated least squares problems obtained using formulas (15) through (18) are identical to the solutions obtained by solving the full least squares*

step	*cost* *(non-recursive)*	*cost* *(recursive update)*
Get QR Decomp	$2np^2 - 2p^3/3 + O(l.o.t)$	
Update (17)		$2p^2 + 10p$
get right hand side of (13)	$2np$	
Update (18)		$4p$
one triangular solve	$p^2 + 2p$	$p^2 + 2p$
Totals	$2np^2 - 2p^3/3 + O(l.o.t)$	$3p^2 + 16p$
when $p = 2$	approx $8n - 16/3$	44
when $p = 4$	approx $32n - 128/3$	112

Table 2: *Costs for QR Decomposition in Floating Point Operations (Flops).*

problem (8) by the corresponding non-recursive method (whether Normal Equations or QR Decomposition), within rounding errors. $\square$

4.2. Recursive Total Least Squares

As previously mentioned, errors are present in the matrix A of coefficients as well as the right hand side of the least squares problem (8) arising from the robot navigation problem. To account for this, a Total Least Squares (TLS) solution is desirable. In this section we sketch the steps necessary to update a TLS solution as new readings are obtained without having to repeat the computation from scratch. The paradigm is similar to the recursive least squares solution in the sense that we use a fast update of a suitable decomposition, which is then used to obtain the new solution.

4.2.1. SVD Method, Updating only Σ and V in $U\Sigma V^T = (A, -\mathbf{b})$.

The primary method for obtaining the TLS solution is via the SVD: $U\Sigma V^T = (A, -\mathbf{b})$, where A is $n \times p$ with $n >> p$. In this case, only the $(p+1) \times (p+1)$ matrices Σ and V are needed; hence the $n \times (p+1)$ matrix U may be omitted from the computation. The updating problem is then the following, using the notation similar to that in the previous section. Given the SVD of

$(A, -\mathbf{b})$, compute the SVD of the updated matrix

$$(\widehat{A}, \widehat{-\mathbf{b}}) = \left(\begin{pmatrix} A \\ \mathbf{a}^T \end{pmatrix} \quad \begin{pmatrix} -\mathbf{b} \\ -b \end{pmatrix} \right). \tag{19}$$

We have the following identities

$$\widehat{U}\widehat{\Sigma}\widehat{V}^T = (\widehat{A}, -\widehat{\mathbf{b}}) = \begin{pmatrix} A & -\mathbf{b} \\ \mathbf{a}^T & -b \end{pmatrix} = \begin{pmatrix} U\Sigma V^T \\ (\mathbf{a}^T, b) \end{pmatrix} = \begin{pmatrix} U & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \Sigma \\ (\mathbf{a}^T, b)V \end{pmatrix} V^T. \tag{20}$$

Since the U matrix is not needed, the new $\widehat{\Sigma}, \widehat{V}$ can be computed by finding the SVD of the $p+1 \times p$ matrix $\begin{pmatrix} \Sigma \\ (\mathbf{a}^T, b)V \end{pmatrix}$ and using the result to update the V matrix. Unfortunately, it is not obvious how to compute this SVD in less than $O(p^3)$ operations, so this algorithm is practical only for small values of p (as encountered here), depending on the power of the underlying computer.

4.2.2. ULV Method, an SVD Approximation.

In cases such as the applications considered in this paper where the exact TLS solution is still corrupted by external effects such as noise, it suffices to obtain an approximate TLS solution at much less cost. We seek a method that can obtain a good approximation to the TLS solution, but which admits rapid updating as new data samples arrive. One such method is the so-called ULV Decomposition, first introduced by Stewart (Stewart 1993) as a means to obtain an approximate SVD which can be easily updated as new data arrives, without making any a priori assumptions about the overall distribution of the singular values. The ULV Decomposition of a real $n \times (p+1)$ matrix M (where $n \geq (p+1)$) is a triple of 3 matrices U, L, V plus a rank index r, where $M = ULV^T$, V is $(p+1) \times (p+1)$ and orthogonal, L is $(p+1) \times (p+1)$ and lower triangular, U has the same shape as M with orthonormal columns, and where L has the form

$$L = \begin{pmatrix} C & 0 \\ E & F \end{pmatrix} \tag{21}$$

where C $(r \times r)$ encapsulates the "large" singular values of M, (E, F) $((p+1-r) \times (p+1))$ approximately encapsulate the $p+1-r$ smallest singular values of M, and the last $p+1-r$ columns of V encapsulate the corresponding trailing right singular vectors. To solve the TLS problem, the U matrix is not required, hence we need to carry only L, V, and the effective rank r. Therefore, a given ULV Decomposition can be represented just by the triple $[L, V, r]$.

The feature that makes this decomposition of interest is the fact that, when a new row of coefficients is appended to the M matrix, the L, V and r can be updated in only $O(p^2)$ operations, with L restored to the standard form above, as opposed to the $O(p^3)$ cost for an

step	*cost (recursive update)*
Compute $\mathbf{a}^T V$	$2p^2$
`Absorb_One`	$8p^2 + 20p + 6$
`Extract_Info`	$2p^2 + 4p$
`Deflate_One`	$8p^2 + 20p$
Totals	$20p^2 + 44p + 6$
when $p = 2$	174
when $p = 4$	502

Table 3: *Estimated Costs for ULV Update in Floating Point Operations (Flops).*

SVD (though still more than the least squares updates (12) through (18)). In this way, it is possible to track the leading r-dimensional "signal subspace" or the remaining "noise subspace" relatively cheaply. Details on the updating process can be found in (Stewart 1993; Hosur, Tewfik, & Boley 1995). Briefly, the steps are as follows: `Absorb_One` to absorb the new row $(\mathbf{a}^T, b)$ into L, restoring the triangular form, `Extract_Info` to estimate the smallest singular value and corresponding singular vector of the new L, and `Deflate_One` to decouple the singular value just estimated by a change in bases (i.e. expose the blocks E, F of small entries in (21)). The respective costs are given in Table 3.

We can adapt the ULV Decomposition to solve the Total Least Squares (TLS) problem $A\mathbf{x} \approx \mathbf{b}$, where new measurements b are continually being added, as proposed in (Boley, Steinmetz, & Sutherland 1995). The adaptation of the ULV to the TLS problem has also been analyzed independently in great detail in (Van Huffel & Zha 1993), though the recursive updating process was not discussed at length. For our specific purposes, let A be an $n \times p$ matrix and $\mathbf{b}$ be an n-vector, where p is fixed and n is growing as new measurements arrive. Then given a ULV Decomposition of the matrix $(A, \mathbf{b})$ and an approximate TLS solution to $A\mathbf{x} \approx \mathbf{b}$, our task is to find a TLS solution $\widehat{\mathbf{x}}$ to the augmented system $\widehat{A}\widehat{\mathbf{x}} \approx \widehat{\mathbf{b}}$, where

$$\widehat{A} = \begin{pmatrix} \lambda A \\ \mathbf{a}^T \end{pmatrix} \text{ and } \widehat{\mathbf{b}} = \begin{pmatrix} \lambda \mathbf{b} \\ b \end{pmatrix},$$

and λ is an optional exponential forgetting factor (Haykin 1991).

The RTLS Algorithm:

- Start with $[L, V, r]$, the ULV Decomposition of $(A, \mathbf{b})$, and the coefficients $\mathbf{a}^T, b$ for the new incoming equation $\mathbf{a}^T\mathbf{x} = b$.
- Compute the updated ULV Decomposition for the system augmented with the new incoming equation. Denote the new decomposition by $[\widehat{L}, \widehat{V}, \widehat{r}]$.
- Partition
$$\widehat{V} = \begin{pmatrix} \widehat{V}_{11} & \widehat{V}_{12} \\ \widehat{V}_{21} & \widehat{V}_{22} \end{pmatrix},$$
where $\widehat{V}_{22}$ is $1 \times (p + 1 - \widehat{r})$.

  If $\|\widehat{V}_{22}\|$ is too close to zero (according to a user supplied tolerance), then we can adjust the rank boundary $\widehat{r}$ down to obtain a more robust, but approximate solution (Boley, Steinmetz, & Sutherland 1995; Hosur, Tewfik, & Boley 1995).
- Find an orthogonal matrix Q such that $\widehat{V}_{22}Q = (0, \ldots, 0, \alpha)$, and let $\mathbf{v}$ be the last column of $\widehat{V}_{12}Q$. Then compute the new approximate TLS solution according to the formula $\widehat{\mathbf{x}} = -\mathbf{v}/\alpha$.

This RTLS Algorithm makes very few assumptions about the underlying system, though the user must supply a zero tolerance and a gap tolerance for $\|\widehat{V}_{22}\|$. For the application here, it sufficed to set the zero tolerance to zero and depend on just the gap tolerance of 1.5. Under the conditions of Theorem T5, the block $\widehat{V}_{22}$ above is just a scalar, and $Q = 1$.

4.3. Pros and Cons for various methods.

The theory we have developed so far has assumed that the least squares problem to be solved has full rank. If the rank is deficient, then at most only a partial solution can be obtained. In general multiple solutions exist to the mathematical problem and geometric or physical considerations must be used to determine which of the many solutions to correct. However, an alternative is to reject the solution if the system is rank deficient, and furthermore to use the distance to rank deficiency as a measure of the well-posedness of the geometric problem to be solved. The relative distance to rank deficiency is simply the ratio of the smallest to the largest singular value, which is exactly the reciprocal of the condition number of the underlying matrix (the matrix A in the case of the LS problem and the augmented matrix $(A, \mathbf{b})$ for TLS). For the least squares problem, we have already seen that the sensitivity of the result is closely related to exactly this condition number (11b). Hence not only do the schemes proposed here provide fast estimates of the robot position, they also provide estimates of the error in the computed position.

The recursive TLS method based on the recursive update of the SVD is computing exactly the same solution as the nonrecursive TLS method, since we are computing the same partial SVD in both cases. When the ULV approximation is used, we are computing only an approximate solution, but we note that the computed L and V satisfy the relation $ULV = (A, \mathbf{b})$ exactly at every stage. Hence if the exact solution is ever desired, it can be recovered at any stage by taking the SVD of the L matrix, which would yield the exact SVD of the full A matrix. Hence even with the approximate ULV decomposition, we have not lost any information.

The choice of method to use in any particular situation depends on the size of the errors in the data obtained from the sensors, the conditioning of the equations which depends on the geometry of the system, and the computing power available on board the vehicle. If the errors are modest and the system geometry leads to a reasonably well-conditioned system, then any of the methods we have proposed will yield reasonable position estimates, so the choice can be made entirely on cost. However, if the sensors have low resolution or high noise, then a TLS solution may better capture the effect of such errors throughout the whole system of equations. The improvement from TLS may be more noticeable when the geometry of the system leads to moderately ill-conditioned systems of equations. But if the conditioning of the system becomes significant compared to the precision of the underlying arithmetic (because the underlying navigation problem becomes ill-posed), then none of the methods will yield useful position estimates. But estimates of the condition number can be obtained from the SVD for free and from the triangular factors in all the other methods with low additional cost (Higham 1987), and these condition number estimates are reliable indicators of ill-posedness of the underlying problem.

5. Experiments.

To compare the performance of the Kalman filter and RTLS in practice, we ran three sets of experiments, the first with a physical mobile robot and camera and a single landmark, the second in simulation with two landmarks, and the third with a physical robot and two landmarks not visible simultaneously.

5.1. One Landmark.

In this experiment, a physical robot viewed a single landmark with a camera, and used the bearing information derived from the images to compute its location. The setup in the first set was modeled after the problems faced by an actual mobile robot (Ayache & Faugeras 1989; Durrant-White, Bell, & Avery 1995; Kosaka & Kak 1992). The robot did not know its own position on the map, but did know the location of a single landmark. The robot moved in a straight line taking a series of images. Its task was to find the landmark in each image, and use the results to determine its start position relative to the landmark.

A Panasonic WV-BL202 camera was mounted on a TRC Labmate at an angle of 90° to robot bearing, so that each image yields an angle β_i, as shown in Figure 2. Horizontal field of view was $50°44'$, limiting the angles β to the range $\pm 25°22'$. "Landmarks" were mini Maglite high intensity flashlight candles. The angular position of the landmark was measured in a sequence of images taken while the robot moved across the room at a constant velocity. In addition to the error in angle measure, error also occurred in velocity, robot bearing and in the times at which the images were taken. It is not possible to predict and model these errors. For example, velocity was set at 20mm/second, but average true velocity across runs ranged from 21.4mm/second to 22.5mm/second. In addition, the supposed constant velocity was not constant throughout a single run, varying in an unpredictable manner. It would be unrealistic to assume any of these errors or their combined result to have a gaussian distribution.

The Kalman filter was given an arbitrary start position of (0,0) (the location of the landmark), so that the deviation at time 0 for the Kalman filter is the initial distance from the robot to the landmark.

Figure 4 shows a comparison of four of the robot runs. The robot velocity was set to 20mm/sec. Five images were grabbed 12 seconds apart. The robot start position relative to the landmark used for localization was different in each run. The deviations d of the estimate of start location from actual start location at each 12 second time interval t are compared. The RTLS filter converged faster and to more accuracy than did the Kalman, often requiring only 2 or 3 steps to achieve full accuracy.

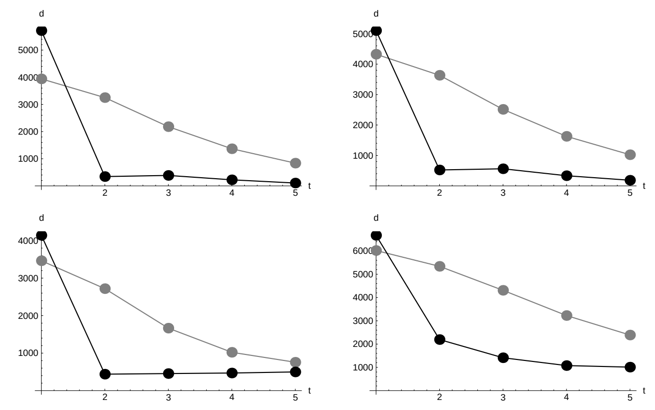

Figure 4: *Performance of RTLS (black) and Kalman filter (grey) on runs using the TRC Labmate starting with 4 different landmark locations. Images were grabbed at time intervals t (horizontal axis) 12 seconds apart. The vertical axis gives the deviation of the estimated start position from the actual start position in millimeters.*

5.2. Two Landmark Simulation.

The second set of experiments was run in simulation, but used two landmarks. We assumed that the robot has no instrument such as a compass which could be used to register its heading. Such instruments can give varying, incorrect readings in outdoor, unstructured environments (Sutherland & Thompson 1994), so it is useful to design and evaluate methods to obtain heading information from external sources. If such heading information was available, it could be used independently or as a correction to estimates from internal sources. The robot knows the location of the two landmarks on a map (ground coordinate system). A coordinate system is arbitrarily centered at one landmark. The goal is to determine the robot start position plus the location of the second landmark. Knowing which landmark is which in the view will allow the robot to uniquely determine its starting position from multiple readings along a baseline of unknown direction, except for certain degenerate configurations. Even if the robot does not know the order of the two landmarks in its view, it can limit its start position to only two possible locations in the ground coordinate system, symmetrically located on either side of the line joining the landmarks, without any *a priori* knowledge of direction.

Figure 5 summarizes the results in an example where the two landmarks and the robot were placed at positions $(-200, 0)$, $(0, 0)$, and $(-200, -200)$, respectively, in the ground coordinate system.

When the angle error is negligible, the TLS method provides uniformly good estimates. When the angle error is moderate, the error from TLS method suffers from an initial jump, but quickly recovers because it needs no initial estimate. Furthermore, in the regions where the RTLS error exceeds the Kalman filter error, neither filter yields any accuracy at all, since both errors are larger than the values being estimated.

5.3. Two Landmark (but One at a Time) Experiment.

This experiment was modeled after the simulation in the previous section, but we assumed that only one landmark was visible at a time. The experiment was carried out with the same hardware as in the first experiment. In this case, the image processing was carried out an a 486-based PC-104 system. By using a more efficient image processing algorithm (Fischer & Gini 1996), we were able to use landmarks consisting of a large letter `T` on a white 8.5 by 11 inch sheet of paper, under ordinary room light conditions. The overall layout is illustrated in Figure 6.

In this experiment, we used the two landmark algorithm as described in this paper, but inserted rows of zeroes corresponding to the landmark not in the field of view. As the results in Figure 7 show, the algorithm fails to deliver even a rough estimate of the location as long as only one landmark is visible. This is because the system is rank deficient, and in the current version of the algorithm, we accept solutions only when the rank is full. As soon as the second landmark is sighted, the rank of the system immediately becomes full and we obtain valid solution estimates. After only 3 or 4 readings of the second landmark, these solution estimates have converged.

6. Conclusions

We have shown how the navigation problem can be set up as an overdetermined system of linear equations which can be solved at each step in a time independent of the number of readings. Our method would be exact if the the data were error-free, so using higher resolution vision equipment would yield a corresponding improvement in the accuracy. The recursive methods we propose are guaranteed to converge to the nonrecursive LS or TLS solution, respectively, computed “off-line” after all the data has been collected. This is in contrast to a method in which the model being manipulated is itself a linear approximation of the true system, or in which there is a possibility of divergence, as occurs in the EKF. The cost of our recursive LS method is comparable to that of a Kalman filter, and the method also yields error estimates dynamically. These error estimates could be used to decide when to re-initialize the iteration or flag an "ill-posedness" error. The RTLS methods can be applied with only a modest increase in costs over the recursive LS methods.

In practice, the Kalman filter is also used when the item being estimated is being updated itself, such as the current robot position, as proposed in (Bonnifait & Garcia 1996). But even in this case, the Kalman filter requires a good initial estimate of the robot's position, and the methods proposed in here would be more suitable to producing such an estimate with no a-priori estimate than using a pure Kalman filter. Such a hybrid filter, using the LS or TLS filter to initialize and the Kalman filter to continue, would be more successful than a pure Kalman filter.

In addition, since this filter converges rapidly, another option is to restart from scratch periodically. A restart could be done also as the landmarks change. In fact, since the computation is so inexpensive, one simple way to handle multiple landmarks visible one at a time in sequence is to carry out the computation with the first two landmarks as outlined above. When the second landmark comes into view, one starts a new computation to be used with the the second and third landmark while continuing the computation with the first two landmarks. The computation using landmarks 1 and 2 will yield a position while that using landmarks 2 and 3 will be ready to yield a position as soon as landmark 3 comes into view. Upon this latter event, the first computation is terminated, and the resources used to start a new computation for landmarks 3 and 4. In this way two computations are always active, one yielding a currently valid position estimate. This scenario is just one example of the flexibility possible with a simple, rapidly converging position estimator.

References

Ayache, N., and Faugeras, O. D. 1989. Maintaining representations of the environment of a mobile robot. *IEEE Transactions on Robotics and Automation* 5(6):804–819.

Bjorck, A. 1996. *Numerical Methods for Least Squares Problems.* SIAM, Philadelphia.

Boley, D. L.; Steinmetz, E. S.; and Sutherland, K. T. 1995. Recursive total least squares: An alternative to using the discrete Kalman filter in robot navigation. In Dorst, L.; van Lambalgen, M.; and Voorbraak, F., eds., *Lecture Notes in Artificial Intelligence 1093 - Reasoning with Uncertainty in Robotics*, 221–234. Springer Verlag.

Bonnifait, P., and Garcia, G. 1996. A multisensor localization algorithm for mobile robots and its real-time experimental validation. In *Proc. 1996 Int'l. Conf. on Robotics and Automation*, 1395–1400. IEEE.

Duda, R. O., and Hart, P. E. 1973. *Pattern Classification and Scene Analysis.* John Wiley and Sons, Inc., 1st edition.

Durrant-White, H.; Bell, E.; and Avery, P. 1995. The design of a radar-based navigation system for large outdoor vehicles. In *Proceedings 1995 Intl. Conf. on Robot. & Auto.*, 764–769. IEEE.

Fischer, J., and Gini, M. 1996. Vision-based mini-robots. *The Robotics Practitioner* 2(2):40–46.

Gelb, A. 1974. *Applied Optimal Estimation.* The M. I. T. Press, 1st edition.

Golub, G. H., and Van Loan, C. F. 1989. *Matrix Computations.* Johns Hopkins, 2nd edition.

Hager, G., and Mintz, M. 1991. Computational methods for task-directed sensor data fusion and sensor planning. *The International Journal of Robotics Research* 10(4):285–313.

Hanebeck, U. D., and Schmidt, G. 1996. Set theoretic localization of fast mobile robots using an angle measurement technique. In *Proc. 1996 Int'l. Conf. on Robotics and Automation*, 1387–1394. IEEE.

Haykin, S. 1991. *Adaptive Filter Theory.* Prentice Hall, 2nd edition.

Higham, N. J. 1987. A survey of condition number estimators for triangular matrices. *SIAM Rev.* 29:575–596.

Hosur, S.; Tewfik, A. H.; and Boley, D. 1995. Multiple subspace ULV algorithm and LMS tracking. In Moonen, M., and Moor, B. D., eds., *3rd Int'l Workshop on SVD and Signal Processing*, 295–302. Elsevier. Leuven, Belgium.

Kalman, R. E. 1960. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering* 35–45.

Kosaka, A., and Kak, A. C. 1992. Fast vision-guided mobile robot navigation using model-based reasoning and prediction of uncertainties. *CVGIP: Image Understanding* 56(3):271–329.

McKendall, R., and Mintz, M. 1990. Sensor-fusion with statistical decision theory: A prospectus of research in the grasp lab. Technical Report MS-CIS-90-68, University of Pennsylvania.

Schneiderman, H., and Nashman, M. 1994. A discriminating feature tracker for vision-based autonomous driving. *IEEE Trans. Robot. & Auto.* 10(6):769–775.

Smith, R. C., and Cheeseman, P. 1986. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research* 5(4):56–68.

Sorenson, H. W. 1970. Least-squares estimation: from gauss to kalman. *IEEE Spectrum* 63–68.

Stewart, G. W. 1993. Updating a rank-revealing ULV decomposition. *SIAM J. Matrix Analysis* 14(2).

Sutherland, K. T., and Thompson, W. B. 1993. Inexact navigation. In *Proceedings 1993 International Conference on Robotics and Automation*. IEEE.

Sutherland, K. T., and Thompson, W. B. 1994. Localizing in unstructured environments: Dealing with the errors. *IEEE Trans. Robot. & Auto.* 10(6):740–754.

Van Huffel, S., and Vandewalle, J. 1991. *The Total Least Squares Problem - Computational Aspects and Analysis.* SIAM, Philadelphia.

Van Huffel, S., and Zha, H. 1993. An efficient total least squares algorithm based on a rank-revealing two-sided orthogonal decomposition. *Numerical Algorithms* 4(1-2):101–133.

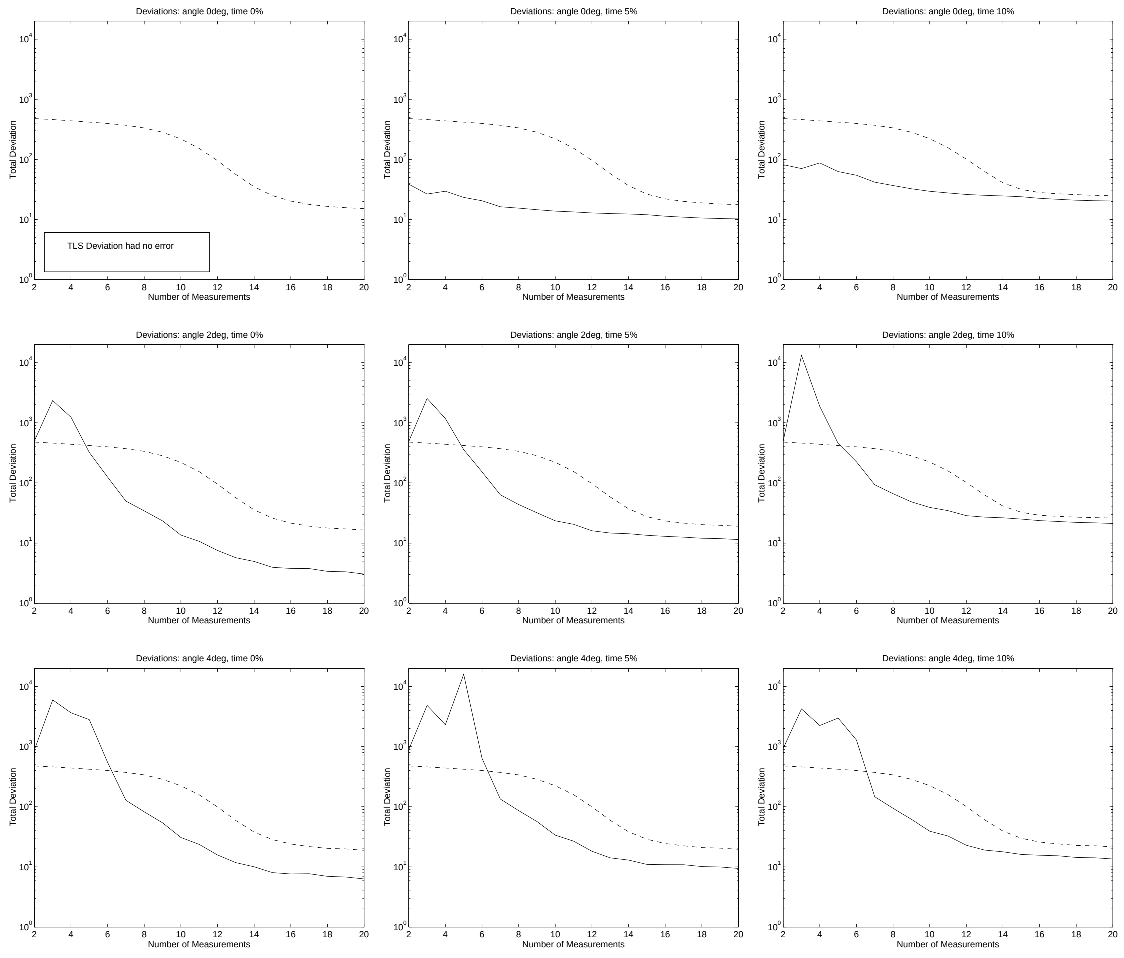

Figure 5: *Mean deviations (d on vertical axis) between estimated and actual start positions, versus time steps (t on horizontal axis). Each row of plots shows the results with uniform errors in the angles of* $0, \pm 2°, \pm 4°$*, respectively, and each column shows the results with normally distributed errors in t with standard deviations* $0, 5\%, 10\%$*, respectively. TLS shown with solid line, Kalman with dashes.*

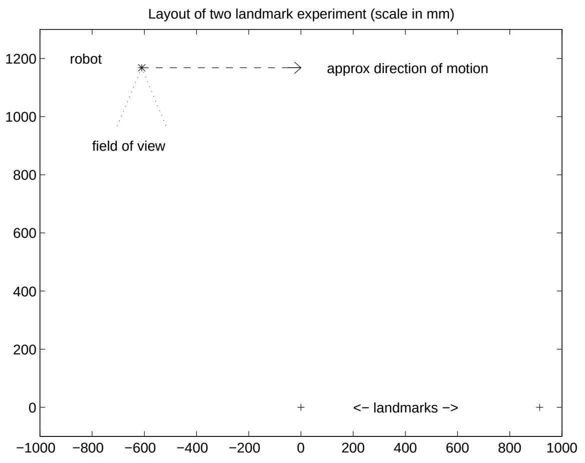

Figure 6: Layout of experiment with two landmarks where only one was visible at a time. The dotted lines show the field of view of the camera.

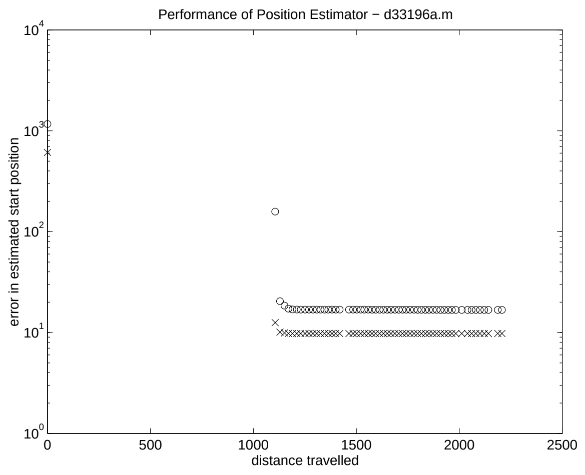

Figure 7: Typical results with two landmarks visible only one at a time. Notice convergence takes place within 3 or 4 readings of the second landmark.

Appendix – Kalman Filter Equations

We sketch the Kalman filter equations with a rough estimate of the costs, assuming the weighting matrices U_i, V_i are diagonal. By estimating the costs in this simple way, we do not intend to claim that our methods are faster, only that the flop counts involved are comparable with those of the Kalman filter.

item	*formula*	*approx. cost (flops)*
System Model	$\mathbf{x}_i = F_{i-1}\mathbf{x}_{i-1} + e_{i-1}$	
Measurement Model	$\mathbf{b}_i = A_i\mathbf{x}_i + \epsilon_i$	
Initial Conditions	$\mathbf{x}_0 = (A_0^T V_0^{-1} A_0)^{-1} A_0^T V_0^{-1} \mathbf{b}_0$	
	$P_0 = (A_0^T V_0^{-1} A_0)^{-1}$	
State Estimate Extrapolation	$\mathbf{x}_i(-) = F_{i-1}\mathbf{x}_{i-1}(+)$	0
Error Covariance Extrapolation	$P_i(-) = F_{i-1}P_{i-1}(+)F_{i-1}^T + U_{i-1}$	p
State Estimate Update	$\mathbf{x}_i(+) = \mathbf{x}_i(-) + K_i[\mathbf{b}_i - A_i\mathbf{x}_i(-)]$	$4p^2 + 2p$
Error Covariance Update	$P_i^{-1}(+) = P_i^{-1}(-) + A_i^T V_i^{-1} A_i$	$2p^3 + 2p^2$
Kalman Gain Matrix	$K_i = P_i(+)A_i^T V_i^{-1}$	$2p^3 + 2p^2$
Total		$4p^3 + 8p^2 + 3p$
when $p = 2$		70
when $p = 4$		396

Table 4: Discrete Kalman Filter Equations where B is the measurement vector, A is the data matrix, F is the state transition matrix, $\mathbf{x}$ is the state vector, and $e_i \sim N(0, U_i)$, $\epsilon_i \sim N(0, V_i)$

.

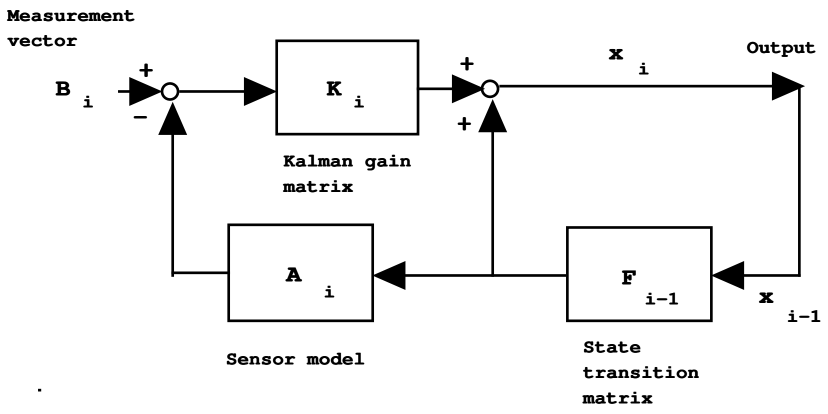

Figure 8: Schematic diagram of Kalman filter