Submitted to the International Conference on Robots and Systems (IROS 2004), Sendai Japan

Fast and Easy Systematic and Stochastic Odometry Calibration

ALONZO KELLY

Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
email: alonzo@ri.cmu.edu, url: http://www.frc.ri.cmu.edu/~alonzo

Abstract

A method of odometry calibration is proposed and validated which is designed to be as convenient as possible. Convenience is enhanced by reducing both the amount of software to be written and the amount of measurements to be made to a minimum. The path dependent nature of odometry can be exploited to reduce the amount of ground truth information to as little as a single known point. The odometry and covariance estimation functions themselves will be used to extract first order parameter variation. Linearization will be performed about the approximate available trajectory rather than the unknown ground truth one. While multiple trajectories will be required to calibrate variance models, it will not be required that they be the same or even similiar. The technique is general enough to apply to any form of odometry and it is general enough to be used for the extraction of unknown parameters of either systematic odometry models or stochastic error models. The derivation and experimental validation of the technique are presented.

1 Introduction

Let odometry be defined as dead reckoning from terrain relative velocity indications. Odometry calibration is important not only because it is the only way to determine pose between fixes which may be rare in some environments. It can be very important during the process of map-building because it can reduce the difficulty of detecting loop closure. It is also important simply because it represents an opportunity to improve performance significantly.

1.1 Prior Work

Calibration in practice often falls into two distinct categories. One approach is to drive specific trajectories in order excite error sources and determine the values of kinematic parameters [2][1] or random error model parameters [3] or both [6]. This approach requires the ability follow a predetermined path sufficiently well.

A more involved approach is to calibrate the system while it runs [4][8]. While this approach is likely to generate excellent results it also comes at the cost of high complexity. It still may not be adviseable to identify too many parameters, and it assumes that the rest of the sensors are sufficiently accurate to serve as references for calibration.

All previous work cited is based on closed form solutions for particular trajectories. Only a few works deal with stochastic error models, All are peculiar to a specific form of odometry and all require a second specialized algorithm to accomplish the actual calibration. By contrast, the technique presented here applies to any error model whether systematic or stochastic, regardless of complexity. It does not require specific trajectories or the ability to execute them and it applies to any form of odometry. The only external algorithm required is the solution to an overdetermined linear system.

1.2 Notational Conventions

It will be important to recognize that the functions $f(\ )$ and $g(\ )$ are used throughout to denote arbitrary functions. They may mean different things in each occurrence except where otherwise noted. This convention is used because it avoids the proliferation of dozens of distinct symbols in the paper which would have been used used only once.

2 Representing Odometry as a Nonlinear Dynamical System

In its most general form, we can write the equations of odometry as an abstract nonlinear dynamical system with observer thus:

$$\begin{aligned} \dot{\underline{x}}(t) &= \underline{f}(\underline{x}(t), \underline{u}(t), t) \\ \underline{z}(t) &= \underline{h}(\underline{x}(t), \underline{u}(t), t) \end{aligned} \tag{1}$$

Where:

- the first equation is the state equation and the second is the observer equation.
- $\underline{x}(t)$ is the state vector, normally consisting of position and orientation states.
- $\underline{u}(t)$ is the input vector consisting of the signals which comprise the forcing functions.

We will concentrate on the 2D case but 3D odometry can be accomplished in environments, such as the outdoors, which may require it.

We will find it convenient to explicitly represent the dependence on some assumed parameters, denoted $\underline{p}$. Also, we can often substitute the observer into the dynamics to eliminate the inputs and get the simpler form:

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t), \underline{z}(t), \underline{p}, t) \tag{2}$$

where the function $f(\ )$ is now different from in equation 1 and measurements now play the same role as the original inputs. This will be the standard form of system dynamics used in the sequel.

3 Systematic Odometry Calibration

The process of implementing odometry to determine state (pose) from measurements comes down to integration:

$$\underline{x}(t) = \underline{x}(0) + \int_0^t \underline{f}(\underline{x}(\tau), \underline{z}(\tau), \underline{p}, \tau) d\tau \quad (3)$$

In practice, the inevitable errors in the estimated parameter values and the partial falsehood of embedded assumptions (flat floors, no wheel slip) lead to errors $\delta\underline{x}(t)$ in the computed robot pose. The systematic model calibation problem is to determine the values of the parameters which cause the computed trajectory to agree with some externally provided ground truth trajectory as closely as possible. Equivalently, we could also seek to determine the errors in the parameters $\delta\underline{p}$ which are most consistent with the observed errors in the computed pose $\delta\underline{x}(t)$.

3.1 First Order Response to Parameter Variation

The equations to be calibrated are generally nonlinear and we will be required to linearize them and solve them iteratively. Hence, the first order response of the computed pose to errors in parameters is of central importance.

It has been shown [5] that the general solution to the perturbative dynamics of Equation (3) can be generated. This general solution can be written in the form of an integral whose precise form does not concern us here:

$$\delta\underline{x}(t) = \underline{f}[\underline{x}(t), \delta\underline{x}(0)] + \int_0^t \underline{g}(\underline{x}(\tau), \underline{z}(\tau), \underline{p}, \delta\underline{p}, \tau) d\tau \quad (4)$$

This solution relates errors in parameters $\delta\underline{p}$ to the resulting errors in computed pose $\delta\underline{x}(t)$ - for a given trajectory. Such an expression can be obtained by linearizing either the original nonlinear dynamics with respect to parameter errors or by linearizing the linear (with respect to input error) error dynamics again with respect to parameter variations. In any case, by Leibnitz rule, we must end up with an integral of a derivative to compute.

However, we will, in the interest of convenience, take the more straightforward approach of computing the first order response of equation (3) to parameter variation because:

- either equation will ultimately be integrated numerically so there may be little justification to prefer an iterative numerical solution to the exact linearized error dynamics (second equation) over an iterative numerical solution to the exact nonlinear dynamics (first equation).
- both equations (3) and (4) are integrals of nonlinear integrands but the second is more complex and the first must be implemented anyway - because it is the odometry solution itself.

In practice, odometry usually takes the form of a discrete time recursive version of equation (3):

$$\underline{x}_{k+1} = \underline{x}_0 + \sum_k \underline{f}(\underline{x}_k, \underline{z}_k, \underline{p}) \Delta t_k \quad (5)$$

Where $\underline{x}_k = \underline{x}(t_k)$ etc. This is usually implemented recursively:

$$\underline{x}_{k+1} = \underline{x}_k + \underline{f}(\underline{x}_k, \underline{z}_k, \underline{p}) \Delta t_k \quad (6)$$

3.2 Linearized Parameter Error Observer

Consider equation (3). If we use the notation $\underline{z}(\ )$ to mean the entire input function rather than its value at time t, we can suppress the integral notation and avoid the need to express the state in terms of itself. Equation (3) thus rewritten is:

$$\underline{x}(t) = \underline{g}(\underline{x}(0), \underline{z}(\ ), \underline{p}, t) \quad (7)$$

For a specified entire input time history $\underline{z}(\ )$ we can differentiate this with respect to the parameters by just running odometry (however it is implemented - for example, by equation (6)) twice for each parameter. One at a time, let a single element of the parameter vector be slightly adjusted to produce a numerical derivative of the form:

$$\frac{\partial \underline{g}}{\partial p_i} = \frac{\underline{g}(\underline{x}(0), \underline{z}(\ ), \underline{p} + \delta\underline{p}_i, t) - \underline{g}(\underline{x}(0), \underline{z}(\ ), \underline{p}, t)}{\delta p_i} \quad (8)$$

where the <u>vector</u> $\delta\underline{p}_i$ in the numerator is related to the scalar δp_i in the denominator:

$$\delta\underline{p}_i = \begin{bmatrix} 0 & 0 & \ldots & \delta p_i & \ldots & 0 & 0 \end{bmatrix} \quad (9)$$

ith position

That is, it is zero everywhere except for the occurrence of δp_i in the ith position. Also, this derivative depends on all of the arguments of $\underline{g}(\ )$ - the initial conditions, the input history, the parameters, and time.

Equation (9) is a numerical approximation for the ith column of the Jacobian matrix $\partial \underline{g} / \partial \underline{p}$. If we collect all of these columns together, we can express how a change in computed pose depends linearly on the associated change in parameters:

$$\delta\underline{x}(t) = \frac{\partial \underline{g}}{\partial \underline{p}} \delta\underline{p} = J\delta\underline{p} \quad (10)$$

If we are willing to further assume that some observed errors are caused solely by such a set of parameter errors, this system can be solved for the magnitudes of the errors. Assuming enough constraints are present, the solution can be obtained from the pseudoinverse:

$$\delta\underline{p} = [J^T J]^{-1} J^T \delta\underline{x}(t) \quad (11)$$

3.3 Benefit of Path Dependence of Odometry

While it is often a source of difficulty that odometry is a path dependent (integration) process, this property can be used to great advantage in calibration problems. Indeed, because the above Jacobian is itself path ($\underline{z}(\ )$ or more precisely the associated $\underline{x}(t)$) dependent, each independent path to the same terminal point leads to an independent observation of the effect of parameter errors on terminal computed pose. Indeed, even two separated points along the same path generates independent observations - a technique exploited by all on-line identification methods.

In the end, only the conditioning of the Jacobian itself matters in determining whether the system can be solved so we can use just a few, or even just one known point to calibrate any number of unknown parameters provided the paths used are sufficiently different:

$$\delta \underline{x}(t_f) = \frac{\partial \underline{g}}{\partial \underline{p}} \delta \underline{p} = J \delta \underline{p} \tag{12}$$

3.4 Benefit of Using Computed Trajectory as Reference Trajectory

In practice, another important principle is critical to making odometry calibration convenient. While we often view perturbation in terms of how the correct answer is corrupted to produce the incorrect one, it is equally valid to view perturbation as the process of adjusting the incorrect answer to produce the correct one. Indeed, this is the basic difference between the linearized and extended Kalman filters [7].

The last section precludes the need to have ground truth all along test trajectories in order to compute observed errors. Likewise, the evaluation of the parameter Jacobian with respect to the computed trajectory precludes the need for any ground truth in the linearization.

When the arguments of both this section and the previous are invoked, it is clear that equation (12) amounts to a completely general way to calibrate any number of parameters from observed errors at any number of points along any number of trajectories. Only the rank and the conditioning of the stacked Jacobian matters.

4 Stochastic Odometry Calibration

Of course, once the best fit parameters are determined from the data set, there is still likely to be a certain amount of error remaining. This remaining error after systematic calibration can often be profitably modelled as if it was random.

Parameters of a stochastic error model may be determined in like fashion to the preceeding systematic derivation. Any number of moments of a probability distribution can be determined, in principle, using modifications of the following technique but we will concentrate on the second moment, or variance.

The response of equation (2) in its nonlinear form to random parameter variations requires either Monte Carlo analysis or significant analytic complexity. We will therefore restrict ourselves to linearized models of stochastic error propagation. It is well known [9] that the response of equation [2] to random input variations is given, to first order, by the linear variance equation:

$$\dot{P}(t) = F(t)P(t) + P(t)F(t)^T + L(t)Q(t)L(t)^T \tag{13}$$

where, with reference to equation (2), we define the particular expectation:

$$P = Exp(\delta \underline{x}(t) \delta \underline{x}(t)^T) \tag{14}$$

as the state covariance and:

$$Q = Exp(\delta \underline{z}(t) \delta \underline{z}(t)^T) \tag{15}$$

as the measurement covariance. The Jacobians are those of the nonlinear system dynamics with respect to the state and the inputs:

$$F(t) = \frac{\partial}{\partial \underline{x}} \underline{f} \bigg|_{\underline{x}, \underline{z}} \qquad L(t) = \frac{\partial}{\partial \underline{u}} \underline{f} \bigg|_{\underline{x}, \underline{z}} \tag{16}$$

4.1 First Order Response to Parameter Variation

The general solution to equation (13) can be written in the form of an integral whose precise form does not concern us here:

$$P(t) = \underline{f}[P(0)] + \int_0^t \underline{g}(P(\tau), Q(\tau), \underline{p}, \tau) d\tau \tag{17}$$

These expressions also depend in general on the state, and possibly, the inputs. The stochastic model calibation problem is to determine the parameters $\underline{p}$ which are most consistent with the observed variance $\overline{P(t)}$ of some data set.

The first order response of the computed variance to errors in parameters is of central importance. The computed variance $P(t)$ is already linear in the measurement variance $Q(t)$, so if $Q(t)$ is linear in the parameters, the state covariance will also be linear in the parameters and they can be solved in a single iteration. In any case, we can once again proceed by seeking an expedient path to parameter linearization based on software which is likely to be written already.

In analogy to the systematic case, we might have a numerical integration of equation (13) available or a quadrature of the general solution in equation (17). However, the most likely situation is that equation (17) has been discretized to be of the form:

$$P_{k+1} = P_0 + \sum_i \underline{f}(P_k, Q_k(\Delta t_k), \underline{p})$$

and then implemented recursively in the form of the system model covariance update in a Kalman filter:

$$P^-_{k+1} = \Phi_k P_k \Phi_k^T + \Gamma_k Q_k \Gamma_k^T \tag{18}$$

We assume that, in this case, the forcing covariance Q_k represents the errors in all odometric sensing - which is to say that the filter has been implemented so that the system model is forced rather than autonomous. In the autonomous case, the odometry measurements would also have to be processed using the measurement update:

$$P_k = [I - K_k H_k] P^-_k \tag{19}$$

In either case, it only matters that the resulting computed covariance represent the estimate of error which is to be calibrated - however it is computed. As we did in the systematic case for Jacobians, we will use the computed trajectory for computing the covaraince s.

4.2 Linearized Parameter Error Observer

Consider equation (17). If we use the notation $Q(\,)$ to mean the entire input error function rather than its value at time t, we can suppress the integral notation and avoid the need to express the state covariance in terms of itself. Equation (17) thus rewritten is:

$$\underline{P}_{pred}(t) = \underline{g}(P(0), Q(\ ), \underline{p}, t) \tag{20}$$

For a specified entire input error time history $Q(\ )$ we can differentiate this with respect to the parameters by just running the covariance update (however it is implemented - for example, by equation (18)) twice for each parameter as was done in the systematic case.

Although covariance is a matrix, the above function $\underline{g}(\ )$ was considered to be vector-valued because it is useful to avoid redundant computation by accounting for the symmetric nature of $P(t)$. In 2D we would map the P matrix onto a vector thus:

$$\begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{x\theta} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{y\theta} \\ \sigma_{x\theta} & \sigma_{y\theta} & \sigma_{\theta\theta} \end{bmatrix} \rightarrow \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \\ \sigma_{x\theta} \\ \sigma_{y\theta} \\ \sigma_{\theta\theta} \end{bmatrix} = \underline{P}_{pred}(t) \tag{21}$$

4.3 Observations of Variance

Unlike for the systematic case, it is not a straightforward matter to make observations of actual state covariance. To observe the parameters of a random process we will undoubtedly have to execute multiple paths and use the variance of a point or points along the path to measure actual variance. To simply execute the same path multiple times would introduce the requirement that we ensure somehow that it is physically the same so that the spread in computed poses at corresponding points would be solely due to the measurement error we are trying to calibrate.

However, in the absence of fixtures to physically guide the robot, being able to execute repetitive motion is tantamount to repetitive position estimation. Unless there is a separate ground truth position estimation system, repetitive position estimation is contrary to the original assumption that there is random error to be calibrated.

In the interest of maximal convenience, we would prefer not to have to make any ground truth measurements and not to use any special fixturing. In fact, it would be ideal if the remaining errors after the systematic calibration process is is applied to the original data set could be used.

To accomplish this, we must define a new random process which is the selection of multiple randomly perturbed trajectories where each is selected from its own ensemble. Within each ensemble, the reference trajectory being perturbed is the same but the reference trajectories of different ensembles are not necessarily the same. Figure 1 illustrates this new random process.

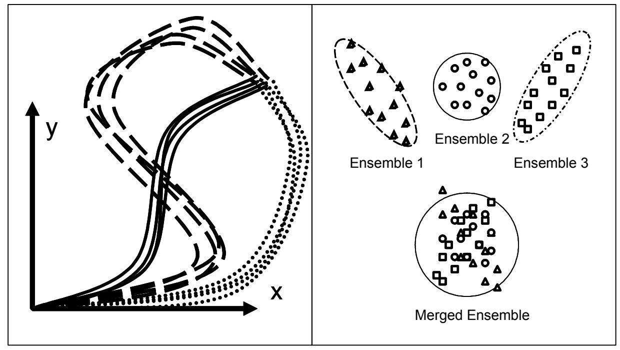

Figure 1: Visualization of Calibration Random Process Ensembles. Three ensemles are illustrated and only one of the trajectories in each is randomly selected. The reference trajectories of all ensembles must have one or more points in common to be used for calibration. This is only true of the endpoints in the figure.

Let T_i denote the event that a trajectory from ensemble i is selected. Let $\delta\underline{x}_k$ denote the event that the vehicle state has a particular error value at time step k. Then, by Bayes rule we have for n equally likely ensembles:

$$p(\delta\underline{x}_k) = \sum_i p(\delta\underline{x}_k|\ T_i)p(T_i) = \frac{1}{n}\sum_i p(\delta\underline{x}_k|\ T_i) \tag{22}$$

The covariance of this distribution is:

$$P_k = Exp(\delta\underline{x}_k \delta\underline{x}_k^T) = \frac{1}{n}\sum_i P_k(T_i) \tag{23}$$

where $P_k(T_i)$ is the covariance of pose error at time step k for ensemble i. In plain terms, the covariance of pose error for the compound process is just the weighted sum (average) of the individual covariances. As the figure illustrates, the merged covariance of a highly positive correlated a highly negative correlated and a decorrelated ensemble produces a decorrelated merged ensemble.

Multiple ensembles are introduced to relieve the need to execute repetitive motions. If two trajectories are the same or almost the same, our technique of using the computed trajectory to compute covariance will automatically account (to first order) for the degree of change in the trajectory.

To use this technique the covariance estimate is computed for each test trajectory and the average of all of these covariances is used as the predicted covariance of the model based on the present parameters values. This average result is to be used in equation (20).

Once the systematic biases are removed from the data set, the observed covariance in the data is computed from the scatter matrix of the n points used:

$$S = \frac{1}{n-1}\Sigma \begin{bmatrix} \delta x_i \delta x_i & \delta x_i \delta y_i & \delta x_i \delta\theta_i \\ \delta x_i \delta y_i & \delta y_i \delta y_i & \delta y_i \delta\theta_i \\ \delta x_i \delta\theta_i & \delta y_i \delta\theta_i & \delta\theta_i \delta\theta_i \end{bmatrix} \tag{24}$$

This computation provides the 6 measured values $\underline{P}_{meas}$ of covariance. The differences between these measured values and the values computed in equation (20) is denoted

$$\delta P = \underline{P}_{meas} - \underline{P}_{pred} \tag{25}$$

The model is then linearized numerically as in the systematic case:

$$\frac{\partial \underline{g}}{\partial p_i} = \frac{\underline{g}(P(0), Q(\ ), \underline{p} + \delta \underline{p}_i, t) - \underline{g}(P(0), Q(\ ), \underline{p}, t)}{\delta p_i}$$

It may be more convenient to compute each of these parameter partials as matrices. In any case, we then solve the square system of six simultaneous equations for the required parameter changes $\delta \underline{p}$:

$$\delta P = \left(\frac{\partial \underline{g}}{\partial p_i}\right) \delta \underline{p} \tag{26}$$

For convenience, it is possible to reuse almost all of the parameter solution algorithm for the systematic case in the stochastic case. As mentioned earlier, the linear variance equation is linear in $Q(\ )$ which is iteself often linear in the parameters, so the above solution may not need to be iterated.

5 Example: Differential Heading Odometry

Further suppose that we have a differentially steered vehicle whose two wheels are instrumented with encoders as shown in Figure2.

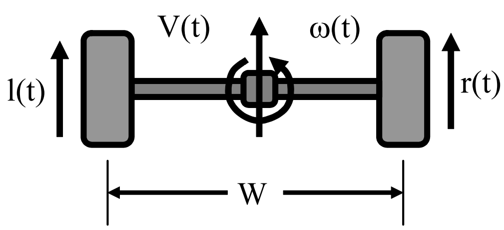

Figure 2: Differential Heading Odometry. Velocity measurements of both wheels are used to determine the linear and angular velocity of the control point.

We can consider that the encoders measure wheel velocities rather than distance by invoking the justified assumption of an available measurement of time. Let the inputs be defined as the angular velocities of the wheels:

$$\underline{u}(t) = \left[\omega_r(t)\ \omega_l(t)\right]^T \tag{27}$$

The state vector will be the position and orientation of the vehicle thus:

$$\underline{x}(t) = \left[x(t)\ y(t)\ \theta(t)\right]^T \tag{28}$$

Further, let the vector of linear and angular velocty be denoted thus:

$$\underline{v}(t) = \left[V(t)\ \omega(t)\right]^T \tag{29}$$

The left $l(t)$ and right $r(t)$ wheel velocities are related to the linear and angular velocities thus:

$$\begin{aligned} r(t) &= V(t) + \frac{W}{2}\omega(t) \\ l(t) &= V(t) - \frac{W}{2}\omega(t) \end{aligned} \tag{30}$$

We can also represent the fact that the encoders actually measure angular velocity and introduce the two assumed wheel radii:

$$\begin{aligned} \omega_r(t) &= \frac{r(t)}{r_r} = \frac{1}{r_r}\left[V(t) + \frac{W}{2}\omega(t)\right] \\ \omega_l(t) &= \frac{l(t)}{r_l} = \frac{1}{r_l}\left[V(t) - \frac{W}{2}\omega(t)\right] \end{aligned} \tag{31}$$

Any amount of additional unknown parameters can be introduced at the discretion of the modeller. For example, we can permit wheel radius to vary as a function of speed and/or curvature as a first order model of wheel slip. However, let this be the final form of the observer for our illustrative purpose here. We therefore have the parameter vector:

$$\underline{p} = \left[W\ r_l\ r_r\right] \tag{32}$$

5.1 Odometry Equations

The observer can be substituted in to the dynamics to produce the odometry equations. First, inverting equation (31):

$$\begin{aligned} V(t) &= \frac{1}{2}(r_r\omega_r(t) + r_l\omega_l(t)) \\ \omega(t) &= \frac{1}{W}(r_r\omega_r(t) - r_l\omega_l(t)) \end{aligned} \tag{33}$$

Then, the odometry equations are simply:

$$\begin{aligned} x_{k+1} &= x_k + V_k\cos(\theta_k)\Delta t_k \\ y_{k+1} &= y_k + V_k\sin(\theta_k)\Delta t_k \\ \theta_{k+1} &= \theta_k + \omega_k\Delta t_k \end{aligned} \tag{34}$$

5.2 Covariance Propagation Equations

The transition matrix for this system is:

$$\Phi_k \approx I + F_k\Delta t_k = \begin{bmatrix} 1 & 0 & -V_k\sin(\theta_k)\Delta t_k \\ 0 & 1 & V_k\cos(\theta_k)\Delta t_k \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

where F_k is the system Jacobian. The input noise distribution matrix Γ is:

$$\Gamma = \frac{\partial \dot{\underline{x}}}{\partial \underline{u}} = \frac{\partial \dot{\underline{x}}}{\partial \underline{v}}\frac{\partial \underline{v}}{\partial \underline{u}} = \begin{bmatrix} \cos(\theta_k) & 0 \\ \sin(\theta_k) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{r_r}{2} & \frac{r_l}{2} \\ \frac{r_r}{W} & -\frac{r_l}{W} \end{bmatrix} \tag{36}$$

The covariance propagation equations are therefore:

$$P^{-}_{k+1} = \Phi_k P_k \Phi_k^T + \Gamma_k Q_k \Gamma_k^T \tag{37}$$

We need Q_k to depende on some parameters which are to be calibrated. For illustrative purposes, let the variances of each wheel encoder be assumed to grow linearly with distance travelled. Thus their time derivative Q is proportional to velocity and is given by:

$$Q = \begin{bmatrix} \sigma_{rr} & \sigma_{rl} \\ \sigma_{rl} & \sigma_{ll} \end{bmatrix} = \begin{bmatrix} \alpha_{rr} V_r & \alpha_{rl} V \\ \alpha_{rl} V & \alpha_{ll} V_l \end{bmatrix} \tag{38}$$

and then:

$$Q_k = Q \Delta t_k \tag{39}$$

The parameters to be solved for are:

$$\underline{p} = \begin{bmatrix} \alpha_{rr} & \alpha_{ll} & \alpha_{rl} \end{bmatrix}^T \tag{40}$$

This effective variance accounts also for aspects of terrain/wheel interaction. Some correlation of the errors observed at the two wheels is to be expected. Having no basis to prefer one wheel velocity to the other, this correlation is modelled to grow based on the mean velocity of both wheels.

6 Results

This above technique was implemented for the case of differential heading odometry. A reference layout was constructed as follows. Three points were marked on the floor and the lengths of the lines between them were measured. The resulting triangle was then solved to determine the locations of the second and third points with respect to the first when the line from the first to the second was established as the x axis. As long as the layout process is significantly more accurate than the odometry system being calibrated, it is accurate enough. We were able to repeatedly measure the triangle to a repeatability of 2 mm.

The data set used in [5] is reused here although the calibration techniqe presented here is quite different. A total of 28 different trajectories were executed from physically the same start point to physically the same endpoint. A laser pointer was mounted horizontally to indicate points on distant walls in order to verify heading repeatability.

Six of the 28 trajectories used are shown in figure Figure 3:

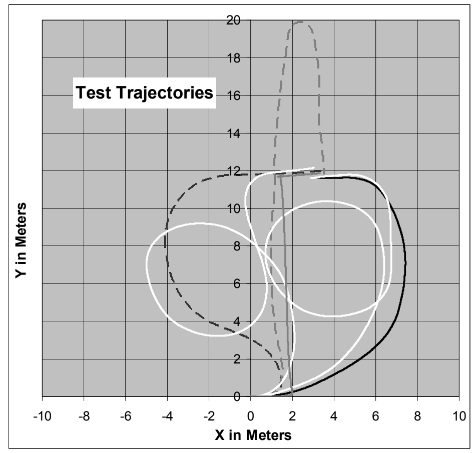

Figure 3: Calibration Test Trajectories. All start at the origin. All terminate near the point (3,12) with arbitrary heading..

Figure 4: is a zoomed-in view of the correct endpoint.

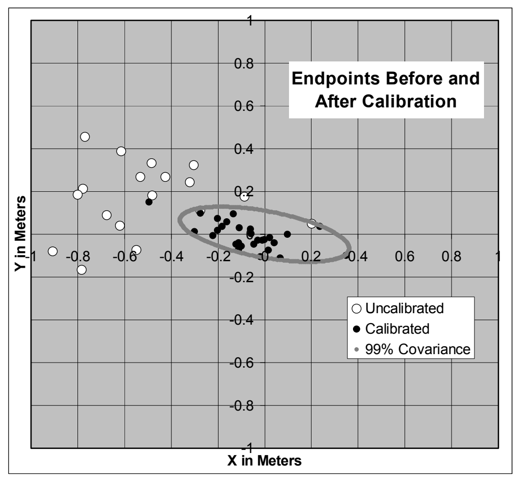

Figure 4: Residuals before and after calibration. The black squares are transformed into the white squares by systematic model calibration. The grey ellipse captures the remaining variance.

The white circles represent the uncalibrated results and the black ones are the calibrated results. Systematic error has been reduced by about 75% in the worst case and the corrected data also has less variance. Of course, if a better systematic model were used, it would probably reduce the variance more. In the systematic model calibration, a highly overdetermined system of equations was solved with 84 constraints on 3 parameters.

The grey ellipse represents a 99% probability ellipse characterizing the variance in the calibrated data. There were only three parameters to be found and we elected to calibrate only the translational elements of covariance. Hence, three linear equations were solved for three unknowns exactly in a single iteration and the scatter matrix and predicted covariance after calibration are identical:

$$S = P = \begin{bmatrix} 0.033 & -0.006 \\ -0.006 & 0.004 \end{bmatrix}$$

7 Summary and Conclusions

A general purpose method of odometry calibration has been presented which applies to both systematic and stochastic models and which is designed for maximal operational convenience.

The derivation for systematic and stochastic model parameter calibration took slightly different paths due to the algorithms most likely to be available for reuse. However, the technique applies to any algorithms for odometry and covariance propagation. It is, of course, a major advantage to be able use the algorithm to be calibrated in its own calibration.

The fact that odometry is highly sensitive to small parameter variations is the essence of both the problem and its solution. This sensitivity arise because odometry is an integration process. We exploit the path dependence of odometry to maximize sensitivity to parameter variation which also requiring a minimum amount of ground truth data.

While it is possible to augment the unknowns by the initial conditions and solve for these in addition to the parameters, we chose not to because being unable to enforce initial conditions normally implies being unable to enforce terminal conditions - which would invalidate the whole approach.

8 Acknowledgements

The author gratefully acknowledges the efforts of Wei Wei in performing the experiments and implementing the calibration algorithms.

9 References

[1] G. Antonelli, S. Chiaverini, and G. Fusco, "An odometry calibration method for mobile robots based on the least-squares technique", 2003 American Control Conference, Denver, CO, pp. 3429--3434, 2003.

[2] J. Borenstein and L. Feng, "Measurement and Correction of Systematic Odometry Error in Mobile Robots", *IEEE Transactions on Robotics and Automation*, 12(6), pp 869-880. 1996.

[3] K. Chong, and L. Kleeman, "Accurate Odometry and Error Modelling for a Mobile Robot", In Proc 1997 Int. Conf. on Robotics and Automation, pp 2783-2788, Albuquerque, NM, May 1997.

[4] H. F. Durrant-Whyte, "An Autonomous Guided Vehicle for Cargo Handing Applications", *International Journal of Robotics Research*, 15(5), 407-440.

[5] A. Kelly "Linearized Error Propaation in Odometry" *International Journal of Robotics Research*, February 2004.

[6] A. Martinelli, "The Odometry Error of a Mobile Robot with a Synchronous Drive System", IEEE Trans on Robotics and Automation, 18(3):399-405,2002.

[7] P. S. Maybeck, *Stochastic Models, Estimation, and Control*, Academic Press, New York, 1982.

[8] A. Rudolph, "Quantization and Estimation of Differential Odometry Errors in Mobile Robotics with Redundant Sensor Information" *International Journal of Robotics Research*,

[9] R. Stengel, *Optimal Control and Estimation*, Dover, New York, 1994.