The International Journal of
Robotics Research
32(8) 912–931

DOI: 10.1177/0278364913488635
ijr.sagepub.com

# Vehicle model identification by integrated prediction error minimization

**Neal Seegmiller, Forrest Rogers-Marcovitz, Greg Miller and Alonzo Kelly**

**Abstract**

*We present a highly effective approach for the calibration of vehicle models. The approach combines the output error technique of system identification theory and the convolution integral solutions of linear systems and stochastic calculus. Rather than calibrate the system differential equation directly for unknown parameters, we calibrate its first integral. This integrated prediction error minimization (IPEM) approach is advantageous because it requires only low-frequency observations of state, and produces unbiased parameter estimates that optimize simulation accuracy for the chosen time horizon. We address the calibration of models that describe both systematic and stochastic dynamics, such that uncertainties can be computed for model predictions. We resolve numerous implementation issues in the application of IPEM, such as the efficient linearization of the dynamics integral with respect to parameters, the treatment of uncertainty in initial conditions, and the formulation of stochastic measurements and measurement covariances. While the technique can be used for any dynamical system, we demonstrate its usefulness for the calibration of wheeled vehicle models used in control and estimation. Specifically we calibrate models of odometry, powertrain dynamics, and wheel slip as it affects body frame velocity. Experimental results are provided for a variety of indoor and outdoor platforms.*

**Keywords**

calibration, model identification, wheeled mobile robots, odometry, stochastic process

## 1. Introduction

Accurate dynamic models are essential for high-performance control and estimation systems. For example, in autonomous vehicle applications, the effectiveness of model-predictive planning (for path following, obstacle avoidance, lane change maneuvers, etc.) is directly correlated with model fidelity (Urmson et al., 2008). To illustrate, Figure 1 presents a classic case in which, even with closed-loop control, an inaccurate motion model causes failure. Likewise, in vehicle state estimation, dead reckoning in the absence of GPS or other position observations depends on a well-calibrated model of the platform's velocity kinematics.

In both control and estimation contexts, it is equally important to understand the stochastic error that remains after the deterministic model is calibrated. Such information can be a basis for assessing risk in stochastic control (Figure 2), or for modeling disturbances in optimal estimation.

We present an integrated prediction error minimization (IPEM) approach to identifying models of dynamical systems. We demonstrate its usefulness for the calibration of motion models for wheeled mobile robots (WMRs). Our objective is accurate multi-step prediction (or simulation) accuracy over a chosen time horizon. These models are of primary importance for *autonomous* WMRs, but may also be relevant to estimation on manned and teleoperated platforms.

Experimental results are provided for odometry calibration on an indoor mobile robot (Section 3) and slip calibration on off-road skid and Ackerman steered platforms (Section 5). In another publication, the authors validated the approach on an articulated rover (Seegmiller et al., 2012) (see also Section 6.4). The calibration approach may be useful on other platforms and terrain conditions not tested here, such as automobiles on paved roads.

### 1.1. Integrated prediction error formulation

In this section we describe the integrated prediction error formulation and its advantages at a high level. To enhance

Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA

**Corresponding author:**
Neal A. Seegmiller, Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA.
Email: nseegmiller@cmu.edu

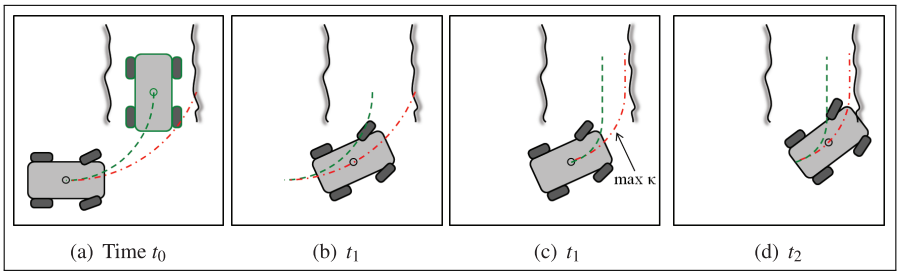


**Fig. 1.** This classic scenario encountered in mobile robot field tests motivates the need for accurate motion models. (a) The robot plans to make an aggressive turn into a narrow corridor following the dashed path; however, due to unmodeled understeering, the robot actually follows the dash-dot path. (b) The robot has deviated from the planned path. (c) Based on position feedback, the robot compensates by planning a sharper turn. (d) Once again, the robot fails to execute the planned path as its curvature exceeds the limits of the steering mechanism. The robot must now abandon the attempt and drive in reverse to avoid a collision. If the planner had an accurate model, it would have simply turned harder at $t_0$ when the turn was still feasible.

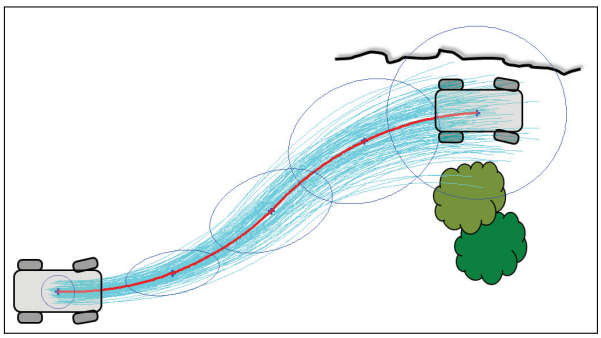

**Fig. 2.** This mobile robot scenario motivates the need for a *stochastic* motion model. Whatever the nature of the controller, its capacity to reject disturbances will not be perfect and this maneuver has little safety margin. The *mean* predicted path avoids the obstacles; however, the uncertainty ellipses (which account for both uncertainty about the initial pose and random disturbances) indicate that collision is still possible. The correct response may be to slow down or find an alternate route.

readability, we will use continuous time notation throughout the paper, but conversion to a discrete time is both straightforward and necessary in practical realizations.

System models are commonly known in the form of a (nonlinear) differential equation:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{p}) \tag{1}$$

where $\mathbf{x}$ is the state vector, $\mathbf{u}$ is the input vector, and $\mathbf{p}$ is the vector of parameters to be identified.

The traditional model identification approach uses the differential equation directly, which requires observations of $\dot{\mathbf{x}}$ and $\mathbf{x}$. Often $\dot{\mathbf{x}}$ can not be measured directly, so measurements of $\mathbf{x}$ are numerically differentiated with respect to time. For example, the *Springer Handbook of Robotics* explains the estimation of manipulator inertial parameters this way, which requires double differentiation of joint angle data (Siciliano and Khatib, 2008).

In contrast, when using IPEM we identify $\mathbf{p}$ from the integrated dynamics:

$$\mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \mathbf{p})\, d\tau \tag{2}$$

In effect, we integrate the prediction rather than differentiate the measurement. Despite its elegance, deriving the correct formulation to exploit this principle in practice requires significant effort. That formulation and its experimental validation constitutes most of the paper. Before embarking on the details, we first motivate the effort by outlining the significant benefits of this approach.

- IPEM reduces the required sensing frequency. Numerical differentiation of state measurements can be noisy and requires both accurate timestamps and high-frequency sensors. IPEM requires only measurements of relative changes in state at low frequency. The sensitivity to accurate timing is also much reduced. Timing errors result in state measurement errors; these may be large relative to the change in state over milliseconds (as observed when computing derivatives), but insignificant relative to the change in state over seconds.
- The amount of ground truth information needed can potentially be reduced. To compute state derivatives, state measurements must not only be high frequency (see the previous item) but available spatially throughout the trajectory. Alternatively, if using IPEM one could drive the system through a few known positions in space (measured at some previous time) and thereby avoid the need for online measurement entirely.
- The compounding effects of integrated error become a virtue. Integrated predictions can allow small errors to accumulate until they become observable and/or easier to disambiguate from other effects. On WMRs for example, an error in predicted angular velocity (due to a poor track width estimate) has no instantaneous effect on linear velocity, but it produces an observable position error after traveling some distance. A gyro can see the effect directly (if available), but even GPS can see it when IPEM is used.

- Predictive performance is optimized for a longer horizon ($t-t_0$). Parameter estimates calibrated to differential equation residuals may provide adequate predictions one time step ahead, but poor predictions when simulating forward many steps. Experimentally, this was particularly true when calibrating stochastic models (Section 6.2).

These advantages come at the cost of additional complexity. The integral observations of IPEM (that replace the numerical derivatives) are path-dependent functionals of inputs and disturbances rather than endpoint dependent functions. We use mathematics to describe how the predicted observation should depend on the trajectory and seek only to calibrate the underlying error-inducing processes that operate instantaneously. The result is a capacity to predict (and thus remove) error trajectories for arbitrary geometry accurately throughout state space, including during transients, based on adequate coverage during calibration. Such an observation formulation is key to enabling online identification, where one must be able to calibrate based on whatever the system is doing.

Other important implications of IPEM are that we must linearize an integral with respect to parameters (Section 2.1.2) and account for the integrated effect of uncertainty in the initial conditions (Section 2.1.4).

## 1.2. Related work

Choosing a name for our approach that correctly relates to prior work was challenging. Classical references on system identification include Ljung (1987) and Söderström and Stoica (1989). These references explain both "equation error" (EE) and "output error" (OE) models which are distinguished by their assumption of noise in the process equation or the output measurement, respectively. Our formulation assumes both, and characterizes the process noise.

Nicolao (1997) makes another distinction between EE and OE techniques. When forming residuals in EE identification, the output at time $k$ ($y_k$) is predicted using past values of the outputs ($y_{k-1},\ldots,y_{k-n}$) and inputs ($u_{k-1},\ldots,y_{k-n}$). In OE identification, $y_k$ is calculated by simulating the model forward multiple steps given only the inputs and initial conditions.

In our paper we do not always make such careful distinction between terms, but in Nicolao's paper and some others, "prediction" is restricted to mean one step-ahead prediction whereas predicting many steps ahead is called "simulation". Note that computing one step-ahead prediction residuals is equivalent to computing residuals of $\dot{\mathbf{x}}$ using numerical derivatives. In discrete time (1) can be rewritten:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{p})\,\Delta t \tag{3}$$

$$\frac{\mathbf{x}_k - \mathbf{x}_{k-1}}{\Delta t} = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{p}) \tag{4}$$

The EE/prediction approach is equivalent to what we call the "traditional" approach in Section 1.1, whereas our IPEM approach is more similar to OE/simulation.

The advantage to the EE/prediction method is that, provided that $f$ is linear in the parameters, then so are the residuals making them simple to estimate by linear least squares. In OE/simulation identification the residuals are nonlinear in the parameters which necessitates nonlinear optimization with all associated problems (convergence, initialization, local minima, etc.), but the identified model will be more suitable for simulation over a long horizon.

Farina and Piroddi (2008) compare several multi-step prediction error minimization (PEM) and simulation error minimization (SEM) techniques, which differ only by the chosen time horizon(s) for which residuals are computed. They relate these techniques to well-studied OE methods. They note that, while multi-step prediction techniques are computationally more expensive than one step-ahead techniques, they are better for unbiased estimation. Jørgensen and Jørgensen (2007) show that multi-step PEM is effective for identifying models for model predictive control (MPC), so long as the objective functions for identification and regulation are compatible. We chose to call our method "integrated" rather than "multi-step" PEM because we present mathematics in continuous time.

Our work on IPEM differs from related work in several ways, for example:

- We address the issue of computational cost by efficiently linearizing the integrated dynamics with respect to the parameters (Section 2.1.2).
- This linearization leads directly to a model of stochastic dynamics, which we calibrate simultaneously with the systematic dynamics. The literature on identifying stochastic differential equations is sparse, at least in robotics applications. Abbeel et al. (2005) present offline coordinate ascent methods for tuning Kalman filters automatically. In the statistics literature there are regression techniques for parameterized (or heteroscedastic) covariances (e.g. Hoff and Niu, 2012), but not in the context of a differential equation.
- We derive the correct expressions for covariance for both systematic and stochastic measurements, properly accounting for all sources of uncertainty. See Sections 2.1.4 and 2.2.3.
- We validate our approach experimentally for various applications in vehicle model identification. A discussion of related work for each specific application is included in its own section.

This paper unifies and elaborates on previous publications by the authors: Kelly (2004a) applied IPEM to 2D odometry calibration, Rogers-Marcovitz and Kelly (2010) applied IPEM to slip model calibration, Seegmiller et al. (2011) demonstrated slip model calibration in 3D on Crusher and introduced stochastic calibration, and later applied IPEM to powertrain model and sensor calibration

(Seegmiller et al., 2012). Note that the name "IPEM" was not used in previous papers, but was chosen for this journal paper. This journal paper provides a more thorough discussion of IPEM theory, full implementation details (including the calculation of stochastic measurement covariance), and more analysis of experimental results (including comparison with the traditional approach, analysis of the distribution of residuals, etc.)

### *1.3. Contents of the paper*

Section 2 explains the calculation of all necessary variables for model identification by IPEM. The close relationship between the systematic and stochastic models is made clear. Offline and online techniques are discussed, as well as a suggestion for active learning.

Next, we apply IPEM to vehicle model identification. We present experimental results for the calibration of 2D odometry (Section 3), powertrain dynamics (Section 4), and predictive models of wheel slip (Sections 5–6).

## 2. Model identification using IPEM

In IPEM as in most identification techniques, we identify the parameters $\mathbf{p}$ by minimizing residuals, which are the difference between a measured and predicted output. To know how to change the parameters such that the residual is reduced, we linearize the prediction with respect to the parameters. As both the residual and parameters are vectors this produces a Jacobian matrix. We also compute the measurement covariance which accounts for all sources of uncertainty (including sensor noise in the observation of initial and final conditions and random disturbances).

In this section we describe the formation of all of these variables. First we describe the formation of predictions and measurements (or observations) for systematic calibration ($\mathbf{h}(\mathbf{p})$ and $\mathbf{z}_{sys}$). Next we describe the linearization of the system integral, which enables efficient computation of the Jacobian ($H_{sys} = \mathrm{d}\mathbf{h}(\mathbf{p})/\mathrm{d}\mathbf{p}$). Also, from the linearized systematic dynamics we can easily derive the stochastic dynamics. Using the stochastic dynamics, we can compute the measurement covariance, $R_{sys}$.

We then describe the formation of predictions, observations, Jacobians, and measurement covariance for calibration of the stochastic model ($\mathbf{h}(\mathbf{q})$, $\mathbf{z}_{stoch}$, $H_{stoch}$, and $R_{stoch}$). Note that $\mathbf{p}$ denotes the vector of systematic parameters and $\mathbf{q}$ the vector of stochastic parameters. We compare the stochastic variables to their systematic counterparts. Finally, we discuss offline and online calibration techniques and a suggestion for active learning.

### *2.1. Systematic model identification*

*2.1.1. Systematic predictions and observations* Given a measurement of state at some initial time $t_0$, time-indexed values of the driving signal $\mathbf{u}$ during the interval $(t_0, t)$, and our current estimate of the parameters $\mathbf{p}$, we *predict* the state at the final time $t$ by forward simulation, meaning solution of the system integral (2). State must also be measured at time $t$ for comparison with the prediction:

$$\mathbf{h}(\mathbf{p}) = \mathbf{x}(t)_{pred} \tag{5}$$

$$\mathbf{z}_{sys} = \mathbf{x}(t)_{meas} \tag{6}$$

The ideal prediction interval length $(t - t_0)$ for calibration is application dependent. If there is mismatch between the chosen model formulation and the true process (due to computational limitations or any other reason) it is possible that the calibrated model will not predict accurately for all intervals; however, accuracy will be optimized for the interval calibrated to. Accordingly, when calibrating a model for a model predictive controller, one might set the interval equal to the horizon used by the controller. Alternatively, Duong and Landau (1994) suggest a minimum necessary test horizon for model validation based on cross-correlation of inputs and outputs (assuming the model is linear and of finite dimension). Calibration based on overlapping intervals is permissible, but note that prediction errors for overlapping intervals will be correlated.

Note that, for simplicity, we assume that the state $\mathbf{x}$ is directly observable. With trivial modifications, our approach could also handle the common case in which the observable output is not state but is some function of state.

Note also that while only intermittent measurements of state ($\mathbf{x}$) are required, inputs ($\mathbf{u}$) must be sampled at a sufficiently high-frequency to capture any variations during the interval. Likewise, if $x(t)_{pred}$ is computed by numerical integration, the integration time step must be sufficiently small. Otherwise, integrated predictions will be poor even if the parameters are correct.

Finally, it is conceptually simplest to think of $t_0$ as being the present time and $t$ as being in the future. For example, a receding horizon controller predicts future state given current state and a planned sequence of inputs. However, in practice replanning often occurs before the planned trajectory is executed to completion. To fix this, in the identifier choose time $t$ to be the current time and time $t_0$ some past time prior to $t$. In this case, when predicting $\mathbf{x}(t)$ we use logged rather than planned inputs, and we withhold any state measurements that occurred during the interval.

*2.1.2. Linearization* A key aspect of the formulation is to linearize the error dynamics about the (known) trajectory prediction rather than the (unknown) ground truth trajectory. This is the same technique used by an extended Kalman filter (EKF). In the following, therefore, the reference trajectory corresponds to the system integral $\mathbf{h}(\mathbf{p})$ evaluated at the present parameter estimates $\mathbf{p}$.

Linearization of $\mathbf{h}(\mathbf{p})$ with respect to $\mathbf{p}$ can be accomplished in either of two ways. The first option is to differentiate the system integral (2) directly. A closed-form solution to the integral is almost never known, so this must be done numerically using finite differences as follows.

If we use the notation $\mathbf{u}(\ )$ to mean the entire input trajectory (from $t_0$ to $t$) we can suppress the integral notation in (2):

$$\mathbf{x}(t) = \mathbf{g}(\mathbf{x}(t_0), \mathbf{u}(\ ), \mathbf{p}) \tag{7}$$

The derivative with respect to the $i$th parameter ($p_i$) may be computed from the forward difference as follows:

$$\frac{\mathrm{d}\mathbf{g}}{\mathrm{d}p_i} \approx \frac{\mathbf{g}(\mathbf{x}(t_0), \mathbf{u}(\ ), \mathbf{p} + \delta\mathbf{p}_i) - \mathbf{g}(\mathbf{x}(t_0), \mathbf{u}(\ ), \mathbf{p})}{\varepsilon} \tag{8}$$

where $\delta\mathbf{p}_i$ is a vector of the same length as the parameter vector, in which all elements are zero except for the $i$th element which is $\varepsilon$, a small epsilon. We compute vectors $\mathrm{d}\mathbf{g}/\mathrm{d}p_i$ for all parameters then combine them as columns in the Jacobian matrix $H_{sys}$. This option is convenient but computationally expensive if there are many parameters.

The second option is to first linearize the system differential equation (1) with respect to state and inputs, then integrate to produce what we call the integrated perturbative dynamics (IPD). We then linearize the IPD with respect to the parameters, which can be done very efficiently by moving the derivative $\mathrm{d}/\mathrm{d}\mathbf{p}$ inside the integral. Here we explain the derivation of the IPD, step by step.

By differentiating (1), we obtain the linearized perturbative dynamics for deterministic (also known as *systematic*) error:

$$\delta\dot{\mathbf{x}}(t) = F(t)\,\delta\mathbf{x}(t) + G(t)\,\delta\mathbf{u}(t) \tag{9}$$

where $F$ and $G$ are Jacobian matrices:

$$F = \frac{\partial\mathbf{f}}{\partial\mathbf{x}}, \quad G = \frac{\partial\mathbf{f}}{\partial\mathbf{u}} \tag{10}$$

The differential equation (9) relates input perturbations $\delta\mathbf{u}$ to state error rates $\delta\dot{\mathbf{x}}(t)$; however, our observations are typically of state, not of rates. In order to process state observations directly, the differential equation must be solved. The key to its solution is the system transition matrix $\Phi$, which by definition satisfies:

$$\dot{\Phi}(t, \tau) = F(t)\,\Phi(t, \tau) \tag{11}$$

Consider the matrix integrating factor:

$$\Psi(t, \tau) = \int_{\tau}^{t} F(\zeta)\, d\zeta \tag{12}$$

It is well known in linear systems theory (see Brogan, 1991) that when the matrix exponential of $\Psi$ commutes with the system Jacobian $F$, then the matrix exponential of $\Psi$ is the transition matrix. In our case $\Psi$ will be nilpotent to degree two, so the matrix exponential will conveniently terminate at the linear term:

$$\Phi(t, \tau) = e^{\Psi(t,\tau)} = I + \Psi(t, \tau) \tag{13}$$

It will also be convenient to define what we will call the *input transition matrix*:

$$\Gamma(t, \tau) = \Phi(t, \tau)\,G(\tau) \tag{14}$$

Using these matrices, the solution to the first-order differential equation (9) is the following. We will call it the *vector convolution integral*:

$$\delta\mathbf{x}(t) = \Phi(t, t_0)\,\delta\mathbf{x}(t_0) + \int_{t_0}^{t} \Gamma(t, \tau)\,\delta\mathbf{u}(\tau)\, d\tau \tag{15}$$

*2.1.3. Jacobian of the systematic prediction* Here we derive the Jacobian of the prediction with respect to the parameters, using the IPD. According to (15) we attribute errors in our prediction of terminal state ($\delta\mathbf{x}(t)$) to errors or perturbations in the inputs ($\delta\mathbf{u}$), which in turn we attribute to errors in our parameter estimates ($\delta\mathbf{p}$). The Jacobian is

$$H_{sys} \approx \int_{t_0}^{t} \Gamma(t, \tau)\,\frac{d\mathbf{u}(\mathbf{p}, \tau)}{\mathrm{d}\mathbf{p}}\,\mathrm{d}\tau \tag{16}$$

By using the Leibniz integral rule to move the derivative inside the integral, computation is greatly reduced.

Note that this Jacobian equation assumes that the parameterization is limited to modulating the inputs; specifically we assume that (1) is of the form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}(\mathbf{p})) \tag{17}$$

For the general case in which the inputs are parameterized but parameters are not confined to modulating the inputs:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}(\mathbf{p}), \mathbf{p}) \tag{18}$$

we merely create new constant inputs $\mathbf{u}_p = \mathbf{p}$ that depend trivially on the parameters.

The reader may be concerned that we are linearizing twice when computing the Jacobian this way, so some discussion of limitations and potential problems is appropriate. First, IPEM is a nonlinear optimization problem, and most nonlinear optimization techniques require linearization with respect to the variables (or parameters) being solved for. If no closed-form solution to the system integral exists, this must be done numerically, either by finite differences or using the IPD. Both are only valid for small changes in the parameters ($\delta\mathbf{p}$).

When using the IPD, we are also linearizing the dynamics with respect to state and inputs. This means the Jacobian in (16) is only valid for small changes in the parameters and in the reference trajectory (i.e. the predicted trajectory). Note that both changes simultaneously approach zero as we get closer to the exact solution. So this double linearization does not affect the quality of our converged solution, but only the radius of convergence. In practice, we have not observed sensitivity of the converged solution to the initial parameter estimates, but this is possible. In fact, regardless of the accuracy of linearization (or even the avoidance of linearization), all nonlinear optimization techniques can potentially get stuck in local minima, depending on initialization.

*2.1.4. Systematic measurement covariance* By squaring and taking the expectation of the systematic error dynamics in (9) we obtain the *stochastic* error dynamics:

$$\dot{P}(t) = F(t)\,P(t) + P(t)\,F(t)^{\mathrm{T}} + G(t)\,Q(t)\,G(t)^{\mathrm{T}} \quad (19)$$

where $P$ is the second moment (covariance) of the error and $Q$ is the covariance of the input noise. The input noise $\delta\mathbf{u}(t)$ is assumed to be white.

$$P(t) = \mathrm{Exp}\left(\delta\mathbf{x}(t)\,\delta\mathbf{x}(t)^{\mathrm{T}}\right) \quad (20)$$

$$Q(t) = \mathrm{Exp}\left(\delta\mathbf{u}(t)\,\delta\mathbf{u}(t)^{\mathrm{T}}\right) \quad (21)$$

Under the whiteness assumption, the solution to the stochastic differential equation (19) is as follows. We will call it the *matrix convolution integral*:

$$P(t) = \Phi(t,t_0)\,P(t_0)\,\Phi(t,t_0)^{\mathrm{T}} + \int_{t_0}^{t} \Gamma(t,\tau)\,Q(\tau)\,\Gamma(t,\tau)^{\mathrm{T}}\,d\tau \quad (22)$$

Here we explain how the matrix convolution integral (22) helps us to compute the measurement covariance when calibrating the systematic model.

First, in (15), $\delta\mathbf{x}(t)$ represents the difference between the *actual* and predicted final state. In practice we do not know the actual final state, but only a noisy measurement of final state. If we instead define $\delta\mathbf{x}'(t)$ as the difference between the *measured* and predicted final state, the vector convolution integral becomes

$$\delta\mathbf{x}'(t) = \Phi(t,t_0)\left(\mathbf{x}(t_0)_{act} - \mathbf{x}(t_0)_{meas}\right) + \int_{t_0}^{t} \Gamma(t,\tau)\,\delta\mathbf{u}\left(\delta\mathbf{p},\tau\right)\,d\tau + \left(\mathbf{x}(t)_{meas} - \mathbf{x}(t)_{act}\right) \quad (23)$$

The subscripts *meas*, *pred*, and *act* in (23) denote measured, predicted, and actual values of state. Note that (23) is nearly identical to (15), except for one additional term that accounts for the error of the final pose measurement. We assume the errors in state measurements are drawn from a distribution with zero mean and known variance: $\Sigma_{x,meas}$.

The input perturbations in (23) have both a systematic and stochastic component. They are drawn from a distribution:

$$\delta\mathbf{u}(\delta\mathbf{p},\tau) \sim \mathcal{N}\left(\boldsymbol{\mu}_{\delta u}(\delta\mathbf{p}),\ Q\right) \quad (24)$$

The systematic component of $\delta\mathbf{u}$ (i.e. the mean of the distribution from which it is drawn, $\boldsymbol{\mu}_{\delta u}$) is attributed solely to errors in the parameter estimates, $\delta\mathbf{p}$. The stochastic component is characterized by the covariance matrix $Q$ which appears in the matrix convolution integral (22).

The measurement covariance, $R_{sys}$, for systematic calibration is the expected residual covariance assuming certainty in our parameter estimates. It is computed using the matrix convolution integral:

$$R_{sys} = P'(t) = \Phi(t,t_0)\,\Sigma_{x,meas}(t_0)\,\Phi(t,t_0)^{\mathrm{T}} + \int_{t_0}^{t} \Gamma(t,\tau)\,Q(\tau)\,\Gamma(t,\tau)^{\mathrm{T}}\,d\tau + \Sigma_{x,meas}(t) \quad (25)$$

Again, note that $P'(t)$ in (25) is almost identical to $P(t)$ in (22) except for an additional term which accounts for noise in the terminal state measurement.

Note that $R_{sys}$ differs from the common $R$ matrix in a state estimation Kalman filter (i.e. that estimates $\mathbf{x}(t)$ rather than $\mathbf{p}$). In a state estimation filter the measurement covariance only accounts for sensor noise at time $t$. We are not trying to measure state, but rather the error in our predictions of state. Even when our parameters are perfectly calibrated, we do not expect to predict $\mathbf{x}(t)$ perfectly, because of random disturbances and uncertainty in the initial conditions. Accordingly, these must be accounted for in our measurement covariance.

Finally, note that we are not concerned with the absolute accuracy of state measurements, but with the accuracy of the measured relative change in state from $t_0$ to $t$. For example, let state be position measured via GPS. In this case, poor positional accuracy in latitude and longitude is irrelevant so long as errors are correlated such that the change in position from $t_0$ to $t$ can be measured accurately. Accordingly, let $\Sigma_{x,meas}$ be the covariance of measurement error with the correlated component subtracted out.

## 2.2. Stochastic model identification

Due to random disturbances and uncertainty in initial conditions, our predictions of future state can never be perfect. A model of stochastic dynamics enables us to, at least, bound the error on our predictions.

In this section we discuss the calibration of the stochastic dynamics. First we discuss the fundamental challenges of calibrating stochastic dynamics without control of the reference trajectory. Next we explain the formation of predictions and observations, just as we did for the systematic dynamics. We derive the Jacobian for a particular parameterization of the input covariance $Q$ and finally, explain the calculation of stochastic measurement covariance.

One standard technique to quantify stochasticity in a process is to repeat the same trajectory numerous times, then compute a sample covariance from the outcomes. However, when passively calibrating online, we do not have the luxury of controlling the system such that trajectories are repeated. Even if we could repeat trajectory $A$ multiple times, the question becomes: how does our characterization of uncertainty for trajectory $A$ help us predict uncertainty for trajectory $B$?

In this light, the path dependence of $\Gamma(t,\tau)$ in (25) is revealed as a virtue. We can use observations of $P'(t)$ for *any trajectory* in order to calibrate the driving noises represented by $Q$. Once $Q$ is calibrated, it can be used to predict uncertainty for any other trajectory.

We must accept at the outset that online stochastic calibration requires some assumptions. First, we assume the mean error of the residuals is zero, which is true only after the systematic parameters are accurately calibrated. In practice we calibrate the systematic and stochastic models

simultaneously online. We must therefore accept that initial estimates of the $Q$ matrix will be overly large due to some systematic error in the residuals not yet calibrated out. Of course even after convergence, the systematic parameters may change to adapt to new conditions. We therefore, assume that the systematic parameters are changing relatively slowly compared with the time period required to estimate variance. Finally, the true process of uncertainty propagation for the system may not be linear; however, a linear approximation can be evaluated efficiently and may be the only practical option online.

*2.2.1. Stochastic predictions and observations* Predicted variance is computed using the matrix convolution integral (25) and our current estimate of the stochastic parameters $\mathbf{q}$.

$$\mathbf{h}(\mathbf{q}) = P'(t) = R_{sys} \tag{26}$$

We also require a measurement of the variance of residuals. Because in general trajectories are not repeated, we will formulate our measurement as a scatter matrix computed from a *single* residual; this is simply the outer product:

$$\mathbf{z}_{stoch} = \mathbf{r}(\mathbf{x})\,\mathbf{r}(\mathbf{x})^{\mathrm{T}} \tag{27}$$

$$\mathbf{r}(\mathbf{x}) = \mathbf{x}(t)_{meas} - \mathbf{x}(t)_{pred} \tag{28}$$

Note that $\mathbf{r}(\mathbf{x})$ is exactly the residual from the systematic calibration.

*2.2.2. Jacobian of the stochastic prediction* Here we compute the Jacobian of the stochastic prediction with respect to the stochastic parameters, $H_{stoch} = \mathrm{d}\mathbf{h}(\mathbf{q})/\mathrm{d}\mathbf{q}$. In the simplest case, the $Q$ matrix is constant (meaning not state or input dependent) and symmetric. The stochastic parameters $\mathbf{q}$ are simply the independent elements of $Q$. In this case, the Jacobian with respect to the element $q_{ij}$ is

$$J_{q_{ij}} = \int_{t_0}^{t} \gamma_{*i}(t,\tau)\,\gamma_{*j}(t,\tau)^{\mathrm{T}}\,d\tau \tag{29}$$

where $\gamma_{*i}(t,\tau)$ is the $i$th column of $\Gamma(t,\tau)$.

In order to ensure that $Q$ is positive-definite, an alternative parameterization may be used consisting of the elements of $L$, the lower-triangular matrix in the Cholesky decomposition of $Q$. For example, if $Q$ were $3\times3$:

$$Q = LL^{\mathrm{T}}, L = \begin{bmatrix} l_{11} & & \\ l_{21} & l_{22} & \\ l_{31} & l_{32} & l_{33} \end{bmatrix}, \mathbf{q} = \begin{bmatrix} l_{11} & l_{21} & l_{22} & l_{31} & l_{32} & l_{33} \end{bmatrix}^{\mathrm{T}} \tag{30}$$

Of course, this also changes the measurement Jacobian:

$$J_{l_{ij}} = \int_{t_0}^{t} \Gamma(t,\tau)\,\frac{\mathrm{d}LL^{\mathrm{T}}}{\mathrm{d}l_{ij}}\,\Gamma(t,\tau)^{\mathrm{T}}\,d\tau \tag{31}$$

Note that the prediction and observation in the stochastic dynamics are both matrices. To avoid tensors when computing the Jacobian, it is convenient to reshape the unique elements of the prediction and observation matrices into vectors.

*2.2.3. Stochastic measurement covariance* Next we derive the measurement covariance $R_{stoch}$. In practice we found that stochastic calibration performance depends critically on calculating $R_{stoch}$ correctly.

Recall that the stochastic measurement $\mathbf{z}_{stoch}$ is just the outer product of the residual $\mathbf{r}(\mathbf{x})$, reshaped into a vector. If the systematic parameters $\mathbf{p}$ are accurately calibrated, we can reasonably assume that the mean residual is zero. In this case, we can consider $\mathbf{z}_{stoch}$ to be a sample covariance for a single sample. Accordingly, the measurement covariance $R_{stoch}$ must be the *covariance of a sample covariance*.

To clarify we will express terms in full for a specific example. If the residual contains three elements $(x, y, \theta)$, then its covariance $R_{sys}$ is a $3 \times 3$ matrix:

$$\mathbf{r}(\mathbf{x}) = \begin{bmatrix} x & y & \theta \end{bmatrix}^{\mathrm{T}} \tag{32}$$

$$R_{sys} = \begin{bmatrix} V(x) & C(x,y) & C(x,\theta) \\ & V(y) & C(y,\theta) \\ & & V(\theta) \end{bmatrix} \tag{33}$$

where $V$ and $C$ are abbreviations for variance and covariance, respectively.

Because $\mathbf{z}_{stoch}$ is a function of the random vector $\mathbf{r}(\mathbf{x})$, it too is a random vector with covariance $R_{stoch}$:

$$\mathbf{z}_{stoch} = \begin{bmatrix} xx & xy & x\theta & yy & y\theta & \theta\theta \end{bmatrix}^{\mathrm{T}} \tag{34}$$

$$R_{stoch} = \begin{bmatrix} V(xx) & C(xx,xy) & C(xx,x\theta) & \dots \\ & V(xy) & C(xy,x\theta) & \dots \\ & & V(x\theta) & \dots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \tag{35}$$

Each element of $\mathbf{z}_{stoch}$ is a product of random variables. The variance and covariance of products of random variables (i.e. the elements of $R_{stoch}$) can be calculated using the following two rules (from Bohrnstedt and Goldberger, 1969):

$$V(ab) = V(a)\,V(b) + C^2(a,b) \tag{36}$$

$$C(ab,cd) = C(a,c)\,C(b,d) + C(a,d)\,C(b,c) \tag{37}$$

The variance and covariance terms required by the rules (36) and (37) are obtained from $R_{sys}$ in (33). These rules assume all random variables $(a, b, c, d)$ have an expected value of zero.

## 2.3. Comparison of systematic and stochastic calibration

Table 1 lists necessary variables for systematic and stochastic calibration, making the relationship between the two clear. Note that the stochastic prediction $z_{stoch} = P'(t)$ is equivalent to the systematic measurement covariance ($R_{sys}$).

**Table 1.** Comparison of systematic and stochastic calibration variables.

| | Systematic | Stochastic |
|---|---|---|
| Parameters | $\mathbf{p}$ | $\mathbf{q}$ |
| Measurement $\mathbf{z}$ | $\mathbf{x}(t)_{meas}$ | $\mathbf{r}(\mathbf{x})\,\mathbf{r}(\mathbf{x})^{T*}$ |
| Prediction $\mathbf{h}$ | $\mathbf{x}(t)_{pred}$ | $P'(t)$ |
| Measurement cov. $R$ | $P'(t)$ | $\mathrm{Cov}(\mathbf{r}(\mathbf{x})\,\mathbf{r}(\mathbf{x})^{\mathrm{T}})$ |

$^{*}\mathbf{r}(\mathbf{x}) = \mathbf{x}(t)_{meas} - \mathbf{x}(t)_{pred}$

## 2.4. Offline and online techniques

Both offline and online calibration methods are possible using the IPEM framework. Online calibration makes for additional challenges, such as weighing past experience with new data, and allows opportunities such as active learning.

While the convolution integrals depend on input histories (functionals), they are merely functions of the parameters. So while the generation of predictions and observations is somewhat complicated, the calibration techniques are quite standard.

*2.4.1. (Offline) Batch least-squares* Both systematic and stochastic parameters may be calibrated using an offline least-squares technique. Each executed trajectory generates equations of the form:

$$H_{sys}\Delta\mathbf{p} = \mathbf{z}_{sys} - \mathbf{h}(\mathbf{p}) \tag{38}$$

$$H_{stoch}\Delta\mathbf{q} = \mathbf{z}_{stoch} - \mathbf{h}(\mathbf{q}) \tag{39}$$

Equations for each trajectory are stacked into overconstrained matrix equations, then least-squares solutions for the errors in the parameter estimates ($\Delta\mathbf{p}$, $\Delta\mathbf{q}$) are obtained using the pseudoinverse. For example, the error in the systematic parameters is computed as follows:

$$\Delta\mathbf{p} = \begin{bmatrix} H^1_{sys} \\ \vdots \\ H^N_{sys} \end{bmatrix}^{+} \begin{bmatrix} \mathbf{z}^1_{sys} - \mathbf{h}(\mathbf{p})^1 \\ \vdots \\ \mathbf{z}^N_{sys} - \mathbf{h}(\mathbf{p})^N \end{bmatrix} \tag{40}$$

where the superscripts denote the trajectory number (1 to $N$) and $^{+}$ denotes the pseudoinverse. Provided a sufficiency of unique trajectories is executed, there will be enough linearly independent rows in the stacked matrix of Jacobians to overconstrain the system.

If measurement covariances for each trajectory vary, weighting rows of the matrix equation accordingly is recommended. Also, the Jacobians in (38) and (39) are based on a linearization of the integrated dynamics about the current parameter estimates; if $\mathbf{h}(\mathbf{p})$ or $\mathbf{h}(\mathbf{q})$ is nonlinear in the parameters, iteration is required. An example of this batch least-squares approach is presented in Section 3.

*2.4.2. (Online) Extended Kalman filter* Kalman filters are an excellent means of online calibration using the IPEM formulation. They make utilizing the measurement covariances derived in Sections 2.1.4 and 2.2.3 straightforward. Furthermore, the $\mathcal{Q}$ matrix in Kalman filters provides explicit control over the relative weight of history and the present measurement. The state $\chi$ is simply the vector of parameters to be identified ($\mathbf{p}$ or $\mathbf{q}$). The process update simply adds uncertainty without changing the parameter estimates:

$$\chi_{k|k-1} = \chi_{k-1|k-1} \tag{41}$$

$$\mathcal{P}_{k|k-1} = \mathcal{P}_{k-1|k-1} + \mathcal{Q}_k \tag{42}$$

The measurement update equations in an EKF are

$$\mathbf{r}_k = \mathbf{z}_k - \mathbf{h}(\chi_{k|k-1}) \tag{43}$$

$$S_k = H_k\mathcal{P}_{k|k-1}H_k^{\mathrm{T}} + R_k \tag{44}$$

$$K_k = \mathcal{P}_{k|k-1}H_k^{\mathrm{T}}S_k^{-1} \tag{45}$$

$$\chi_{k|k} = \chi_{k|k-1} + K_k\mathbf{r}_k \tag{46}$$

$$\mathcal{P}_{k|k} = (I - K_kH_k)\,\mathcal{P}_{k|k-1} \tag{47}$$

where $S$ is the covariance of the residual (or innovation) $\mathbf{r}$, $K$ is the Kalman gain, and $\mathcal{P}$ is the state estimate covariance.

Note that $\chi$, $\mathcal{P}$, and $\mathcal{Q}$ in script font refer to variables in an *identification* Kalman filter. These are not to be confused with the analogous variables $\mathbf{x}$, $P$, and $Q$ in the vector and matrix convolution integrals ((15), (22)) of the system model being calibrated. The stochastic model we calibrate (with parameter $Q$) could be used as the uncertainty propagation model in an *estimation* Kalman filter, but that would be an entirely distinct filter from the one in this section.

## 2.5. Active learning

As mentioned previously, the rate of convergence of the parameter estimates depends on the coverage of input trajectories. An active learning algorithm may be used to speed up convergence by choosing trajectories that provide the most information about the parameters.

According to Kruschke (2008), information gain for a Kalman filter is obtained from the entropy (or "uncertainty") of the probability distribution of the parameter estimates. For a normal distribution $f$ the differential entropy $h(f)$ is specified completely by the $n \times n$ covariance matrix $\Sigma$:

$$h(f) = \frac{1}{2}\ln\{(2\pi e)^n\,|\Sigma|\} \tag{48}$$

In our case, $\Sigma$ is $\mathcal{P}$, the parameter estimate covariance. Information gain is the change (reduction) in entropy following a measurement update. Note that the update equation for $\mathcal{P}$ in (47) does not depend on the residual, so the information gain for planned trajectories can be computed without actually executing them. The exact active learning algorithm will depend on application requirements; but, in general, input trajectories should be chosen which yield the greatest reduction in differential entropy while balancing other cost considerations (obstacle avoidance, energy expenditure, etc.).

## 3. Application: calibrating odometry models offline

This section presents a very straightforward application of IPEM to the calibration of 2D odometry for a WMR.

### 3.1. Related work on odometry calibration

Calibration of wheel odometry has long been an important research topic in robotics. An early example of automated odometry calibration is the UMBmark test in which track width and wheel diameter are identified as the WMR traverses a preprogrammed square trajectory (Borenstein and Feng, 1996). Chong and Kleeman (1997) then demonstrated how a model of *non-systematic* odometry error (i.e. covariance propagation) may be identified from repeated traversals of a preprogrammed trajectory.

A more ambitious approach is to calibrate to arbitrary trajectories while the system runs. Recent examples include Durrant-Whyte (1996), Roy and Thrun (1998), and Rudolph (2003). Some approaches including Martinelli (2002) and Antonelli et al. (2005) are related to ours in that they require only measurements of initial and final pose. Some have combined odometry calibration with other tasks. Censi et al. (2008) demonstrated simultaneous calibration of odometry parameters for a mobile robot and the pose of its onboard laser scanner. Kümmerle et al. (2012) calibrate odometry while performing simultaneous localization and mapping (SLAM).

Our IPEM approach to vehicle model identification differs from this prior work on odometry calibration in several ways, but not all are demonstrated by the simple example in this section. Many are not made clear until Section 5 on slip calibration. Much of the math derived for slip calibration applies to odometry calibration and both may be calibrated simultaneously (Section 6.4).

Most of the above prior work applies only to differentially steered mobile robots moving in 2D. Our vehicle modeling approach can accommodate numerous vehicle configurations and 3D terrain. Prior work by Antonelli et al. (2005) and Censi et al. (2008) calibrates the odometry parameters that affect rotation to heading measurements alone, whereas we derive the system *transition matrix* which enables calibration of these parameters to both heading and position measurements. Very little prior work attempts to characterize the non-systematic odometry error for arbitrary trajectories, one exception being Martinelli et al. (2007). Our stochastic model is somewhat more general than Martinelli's, which does not consider lateral disturbances.

### 3.2. Odometry calibration example

Kelly (one of the authors of this paper) published a "fast and easy" method of 2D odometry calibration using an IPEM formulation (Kelly, 2004a). We refer you to that work for details on the procedure and experimental results, but a summary is presented here.

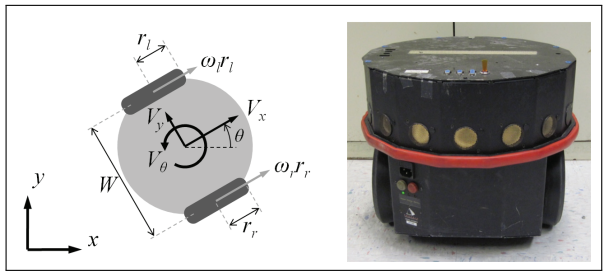


**Fig. 3.** (Left) A diagram of the 2D differential heading odometry parameters, including track width $W$ and left and right wheel radii: $r_l$ and $r_r$. Pose is measured with respect to a ground-fixed frame. (Right) A picture of a differentially steered Nomad Scout robot; a similar platform was used in the experiment.

In this example we calibrate systematic parameters for differential heading odometry. The system differential equation is

$$\dot{\rho}(t) = \mathbf{f}(\boldsymbol{\rho}(t), \mathbf{u}(t), \mathbf{p})$$
$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \cos(\theta(t)) & 0 \\ \sin(\theta(t)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{r_l}{2} & \frac{r_r}{2} \\ -\frac{r_l}{W} & \frac{r_r}{W} \end{bmatrix} \begin{bmatrix} \omega_l(t) \\ \omega_r(t) \end{bmatrix} \quad (49)$$

The state $\boldsymbol{\rho}(t)$ is the pose (or position and heading) of the vehicle. The inputs are angular velocities of the left and right wheels, $\mathbf{u} = [\omega_l(t) \quad \omega_r(t)]^{\mathrm{T}}$. The vector of parameters to be identified includes the track width and the left and right wheel radii, $\mathbf{p} = [W \quad r_l \quad r_r]^{\mathrm{T}}$. These dimensions are depicted in Figure 3.

Given the pose at some initial time $t_0$ and timestamped encoder measurements of wheel velocities, the predicted pose at time $t$ is calculated by integration of (49). For convenience, the Jacobian of each prediction of $\boldsymbol{\rho}(t)$ using the forward difference as explained in Section 2.1.2. Equations for all trajectories are then stacked and the parameters are solved for using the pseudoinverse as explained in Section 2.4.1.

This technique was validated experimentally on a custom platform similar to a Nomad Scout robot (see Figure 3).

Owing to the path-dependent nature of odometry, using IPEM reduces the amount of ground truth information required to just the initial and final pose. These poses were measured by constructing a triangular reference layout. The initial and final position were marked on the floor making two vertices; the third vertex was marked some distance along the $x$-axis, which was aligned with the initial heading. Calibration is beneficial as long as the measurement process is significantly more accurate than the odometry system being calibrated. We were able to measure positions to a repeatability of 2 mm.

A total of 28 different trajectories were executed from the same start point to nearly the same endpoint, approximately 12 m apart. A laser pointer was mounted horizontally to indicate points on distant walls in order to repeatably set the initial heading. As shown in Figure 4, systematic error

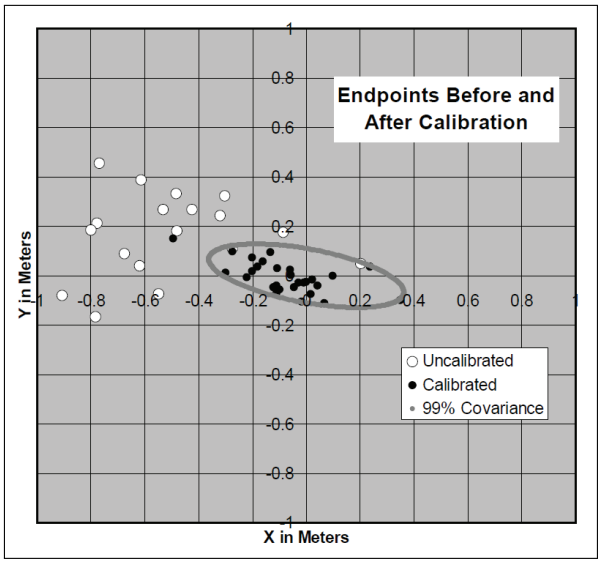


**Fig. 4.** Residuals before and after least-squares calibration of the odometry parameters. Residuals are the difference between the measured endpoint using the triangle layout, and the estimated endpoint using the odometry integral.

is reduced by about 75% in the worst case, and the corrected data also has less variance. Of course, the degree of improvement is reduced when the initial calibration is already quite accurate.

## 4. Application: identifying powertrain dynamics

In this section, we explain how IPEM can be used for online identification of a first-order vehicle powertrain model. Forward kinematic WMR motion models require angular wheel velocities ($\omega$) to compute vehicle velocity. In an estimation context, wheel velocities are measured using encoders, but in a MPC context, future wheel velocities must be predicted according to the planned commands and a dynamic model of the powertrain.

No powertrain can instantaneously drive wheels from one velocity to another; to do so would require infinite torque. When velocity commands change, the response of many powertrains can be modeled adequately by a time delay and a first-order transient response. In Seegmiller et al. (2012), the authors present how the IPEM formulation can be used to identify the time delay and time constant. A brief summary is presented here, but please refer to the original paper for a full discussion of results and related work.

Most WMR motion models in related work omit powertrain dynamics, assuming they operate at low or constant speeds where transient effects are negligible. Those who do model powertrain dynamics (e.g. Yu et al., 2010) do not attempt online identification. Others have proposed time delay identification methods in the context of sensor calibration (Kelly and Sukhatme, 2010; Sheehan et al., 2010).

For our model, the system differential equation is

$$\dot{\omega}(t) = \frac{1}{\tau_c}\left(\tilde{\omega}(t-\tau_d) - \omega(t)\right) \tag{50}$$

where $\omega$ denotes the angular velocity of the wheel. We present a model for an individual wheel, but common models can also be learned for multiple wheels. $\tilde{\omega}$ denotes the *commanded* velocity, $\tau_c$ the time constant, and $\tau_d$ the time delay. Given the angular velocity at some initial time $t_0$ the velocity at the future time $t$ is given by the integral:

$$\omega(t) = \omega(t_0) + \int_{t_0}^{t} \frac{1}{\tau_c}\left(\tilde{\omega}(\tau-\tau_d) - \omega(\tau)\right) d\tau \tag{51}$$

In practice, this integral is computed numerically using the recursive discrete-time relation:

$$\omega[i+1] = \omega[i] + \frac{1}{\tau_c}\left(\tilde{\omega}[i - \frac{\tau_d}{\Delta t}] - \omega[i]\right)\Delta t \tag{52}$$

where the integer in brackets [ ] denotes the time index and $\Delta t$ denotes the time step duration. If $\tau_d/\Delta t$ is not an integer, rounding or interpolation between commands is necessary.

The time constant $\tau_c$ and time delay $\tau_d$ can be estimated online using an EKF as explained in Section 2.4.2. The state is the vector of parameters to be identified: $\chi = \mathbf{p} = [\tau_c \;\; \tau_d]^{\mathrm{T}}$. The measurement $\mathbf{z}$ is the wheel encoder velocity measurement at time $t$ and the predicted measurement $\mathbf{h}(\mathbf{p})$ is the predicted wheel velocity, which is computed according to (51) using the current parameter estimates.

The measurement Jacobian is $H = [\mathrm{d}\omega(t)/\mathrm{d}\tau_c \;\; \mathrm{d}\omega(t)/\mathrm{d}\tau_d]$. These derivatives may both be computed using finite differences. Alternatively the derivative with respect to $\tau_c$ may be computed by solving the following integral:

$$\frac{\mathrm{d}\omega(t)}{\mathrm{d}\tau_c} = \int_{t_0}^{t} \frac{-1}{\tau_c^2}\left(\tilde{\omega}(\tau-\tau_d) - \omega(\tau)\right) - \frac{1}{\tau_c}\frac{\mathrm{d}\omega(\tau)}{\mathrm{d}\tau_c}\, d\tau \tag{53}$$

This integral was derived using the Leibniz and chain rules, and it may be solved numerically using a recursive, discrete-time relation analogous to (52).

We used the online IPEM approach to calibrate a powertrain model for a skid-steered, hydraulic drive "LandTamer" vehicle as it drove in circles. As seen in Figure 5, the wheel velocities predicted by the calibrated powertrain model closely match the actual wheel velocities measured by the encoders. IPEM calibration was more accurate and robust to noise than calibration to $\dot{\omega}$ residuals.

Note that all results on the Crusher and LandTamer platforms presented in Section 6 were generated using powertrain models, calibrated as described here, which were instrumental in removing large prediction errors associated with transients. A powertrain model was not calibrated for the RecBot only because commands were not logged; instead, wheel encoder velocities were used directly.

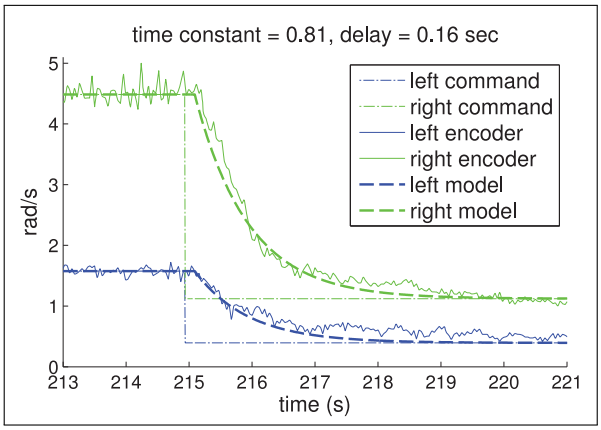


**Fig. 5.** Plot of the left and right wheel angular velocities as the LandTamer decelerates. The calibrated powertrain model predicts wheel velocities that closely match the encoder data.

## 5. Application: identifying models of slip

In this section we apply the IPEM formulation to the identification of wheel slip models. After discussing related work, we present our generalized vehicle model. The model presented here is more sophisticated than in Section 3 because it allows predictions on 3D terrain and includes a parameterized model of slip velocities. Both the systematic and stochastic dynamics of the vehicle are identified online. Extensive experimental results are presented in Section 6. The authors previously published slip modeling results in Rogers-Marcovitz and Kelly (2010) and Seegmiller et al. (2011), but a more complete analysis is presented here.

### 5.1. Related work on wheel slip

Most publications on wheel slip are concerned with the real-time estimation of slip velocities (or dimensionless slip ratios and angles) for feedback control. They do not attempt to predict slip but to observe and compensate for it. Proposed slip observers differ in the type of vehicle assumed and the sensors required. Song et al. (2008) estimated slip from optical flow. Burke (2012) estimated slip for tracked vehicles using only a gyroscope. Grip et al. (2008) and Lenain et al. (2010) estimated sideslip angle for automotive applications using dynamic observers. Yi et al. (2009) used an EKF and a low-cost inertial measurement unit (IMU) to estimate slip. Low and Wang (2007) used an EKF, inertial sensors, and real-time kinematic (RTK) GPS. Lucet et al. (2008) proposed a robust path-following controller that makes use of slip estimates.

Other publications are primarily concerned with the relationship between wheel slip and forces at the wheels. Some terramechanics-based models require knowledge of numerous tire constants and soil parameters. These models have been applied to detect immobilization (Ward and Iagnemma, 2008), and to investigate steering maneuvers for planetary rovers on loose soil (Ishigami et al., 2007). Terry and Minor (2008) and Setterfield and Ellery (2013) estimated force at the wheels in real-time to maintain traction.

Other researchers have addressed the problem of model identification for ground robots. For example, Seneviratne et al. (2009) provided an algorithm to learn soil parameters. Angelova et al. (2006) learned a mapping between slip ratios (measured by visual odometry) and terrain features (appearance and geometry) extracted from stereo imagery. However, there is little precedent in the literature for the calibration of *predictive* models despite the fact that they are fundamental to virtually every decision that a mobile robot makes. The only precedent known to us is Bode (2007) where our colleague constructed an artificial neural network that was trained offline. Our method learns a predictive model online. Note that, unlike some related work, we are not concerned with precisely estimating slip variations (as is required for traction control). Instead we are estimating the gross characteristics of slip (i.e. mean and variance) to improve the predictive accuracy seconds into the future.

### 5.2. Vehicle model design

We made two important choices in designing our vehicle motion models. First, we use a *velocity kinematic* model instead of a full dynamics model. In a full dynamics model, inputs to vehicle motion have dimensions of power and force; however, we are interested in a predictive model of the vehicle under the influence of its own powertrain control system. In other words, the system boundary encloses this control system, which receives velocity commands as input. Accordingly, we judged a velocity driven model to be most appropriate. Second, we simplify vehicle motion to instantaneously planar motion of the body frame. The remainder of this section explains this choice.

Forward velocity kinematics refers to the problem, relevant to simulation and estimation, of computing the vehicle body frame velocity from wheel velocities. These forward kinematics can be formulated for any vehicle configuration as a linear system of "wheel equations" of the form

$$H_v(\theta)\,\mathbf{V} = \mathbf{v}_c - H_\theta(\theta)\,\dot{\theta} \tag{54}$$

In (54), the vector $\mathbf{V}$ is the velocity of the vehicle body frame (both linear and angular), $\mathbf{v}_c$ is the linear velocity of wheel contact points with respect to the ground, and $\theta$ is the position of articulated joints. $\dot{\theta}$ is the time derivative of $\theta$ and includes steering rates and angular wheel velocities. Modification may be required if some joints are passively articulated, so that the corresponding elements of $\dot{\theta}$ are solved for simultaneously with $\mathbf{V}$. For additional explanation, see Kelly and Seegmiller (2012).

In a full 3D model, one computes the six-degree-of-freedom (6-DOF) motion of the body frame (and passive articulation rates if applicable) by solving the linear system in (54). For simplicity and computational efficiency, we only solve for the velocity of the body frame in a

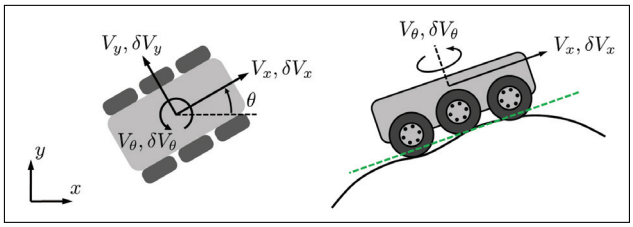


**Fig. 6.** The velocities $V_x$, $V_y$, and $V_\theta$ represent the 3-DOF motion of the body frame in a plane tangent to the terrain surface. Disturbances ($\delta$) are also expressed in the body frame. Suspension deflections are ignored.

plane tangent to the terrain surface. This 3-DOF motion is represented by the velocities $V_x$, $V_y$, and $V_\theta$ (Figure 6).

In effect, we transform complex configuration-dependent velocity kinematic models to a simpler "unicycle" model. While the true inputs to the configuration-dependent model are steering rates and angular wheel velocities, the inputs ($\mathbf{u}$) to the unicycle model are the three body frame velocities. We then have the following kinematic differential equation for the time derivative of pose ($\boldsymbol{\rho}$) with respect to a ground-fixed reference frame:

$$\dot{\rho} = T(\gamma, \beta, \theta)\,[\mathbf{u}] \tag{55}$$

or, expressed in full,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} c\theta c\beta & c\theta s\beta s\gamma - s\theta c\gamma & 0 \\ s\theta c\beta & s\theta s\beta s\gamma + c\theta c\gamma & 0 \\ 0 & 0 & \frac{c\gamma}{c\beta} \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_\theta \end{bmatrix} \tag{56}$$

$$c = \cos(\ ),\ s = \sin(\ ),\ \gamma = \text{roll},\ \beta = \text{pitch},\ \theta = \text{yaw}$$

Note that the differential equation (56) is nonlinear because the matrix $T$ contains the yaw state. Note $\theta$ is used here to denote yaw and is not to be confused with the vector $\theta$ in (54).

We assume the total velocity $\mathbf{u}$ is composed of nominal and slip velocities:

$$\mathbf{u} = \mathbf{u}_n + \mathbf{u}_s \tag{57}$$

$$\begin{bmatrix} V_x \\ V_y \\ V_\theta \end{bmatrix} = \begin{bmatrix} V_{n,x} \\ V_{n,y} \\ V_{n,\theta} \end{bmatrix} + \begin{bmatrix} V_{s,x} \\ V_{s,y} \\ V_{s,\theta} \end{bmatrix} \tag{58}$$

The slip component $\mathbf{u}_s$ is discussed in Section 5.3. The nominal (or kinematic) vehicle velocity $\mathbf{u}_n$ is computed assuming all wheels *roll without slipping*. In other words, we solve the system of wheel equations in (54) assuming the velocity of wheel contact points with respect to the ground ($\mathbf{v}_c$) is zero. Often it is convenient to precompute a closed-form pseudoinverse solution to the system. For example, for a skid-steered vehicle the nominal velocity would be

$$\begin{bmatrix} V_{n,x} \\ V_{n,\theta} \end{bmatrix} = \begin{bmatrix} \frac{r_l}{2} & \frac{r_r}{2} \\ -\frac{r_l}{W} & \frac{r_r}{W} \end{bmatrix} \begin{bmatrix} \omega_l \\ \omega_r \end{bmatrix} \tag{59}$$

$$V_{n,y} = 0$$

This equation already appeared in (49). The track width ($W$) and left and right wheel radii ($r_l$, $r_r$) are kinematic parameters. In a control context, wheel angular velocities ($\omega_l$, $\omega_r$) are predicted from planned velocity commands and a model of the powertrain dynamics (see Section 4). In an estimation context, wheel encoder measurements may be used directly.

Note that simplifications to the vehicle model do not necessarily limit functionality. Even though *instantaneous* motion is restricted to a tangent plane, full 3D motion is still predicted. At each time step roll, pitch, and elevation are computed by fitting the tangent plane to the terrain map at the vehicle's $(x, y, \theta)$ location. The nominal motion model does not need to be perfect as we are characterizing deviations from this model (slip). The model merely needs to be parameterized such that a good fit to experimental data can be obtained, as determined by the minimization of pose prediction residuals.

### 5.3. Parameterization

Here we describe our chosen parameterization of slip. Note that we do not predict the common dimensionless variables slip ratio ($s$) and slip angle ($\alpha$), which quantify slip at the individual wheels. Instead we predict the additional velocity of the *body frame* caused by wheel slip, relative to the body frame velocity that would occur if all wheels rolled without slipping.

Of course, slip is not constant but depends on the commanded trajectory and terrain geometry. Accordingly, we parameterize the systematic component of $\mathbf{u}_s$ over commanded velocities, accelerations, and components of the gravity vector as follows:

$$\mathbf{u}_s = \begin{bmatrix} V_{s,x} \\ V_{s,y} \\ V_{s,\theta} \end{bmatrix} = C\alpha \tag{60}$$

where $\alpha$ is the vector of parameters to be identified, and the coefficient matrix $C$ is

$$C = \begin{bmatrix} \mathbf{c}_x & 0 & 0 \\ 0 & \mathbf{c}_y & 0 \\ 0 & 0 & \mathbf{c}_\theta \end{bmatrix} \tag{61}$$

$$\mathbf{c}_x = \begin{bmatrix} V_{n,x} & |V_{n,\theta}| & (V_{n,x}|V_{n,\theta}|) & g_x \end{bmatrix}$$

$$\mathbf{c}_y = \begin{bmatrix} V_{n,x} & V_{n,\theta} & (V_{n,x}V_{n,\theta}) & g_y \end{bmatrix}$$

$$\mathbf{c}_\theta = \begin{bmatrix} V_{n,x} & V_{n,\theta} & (V_{n,x}V_{n,\theta}) & g_x & g_y \end{bmatrix}$$

where $C$ is a $3 \times 13$ block diagonal matrix (off-diagonal blocks are row vectors of zeros). The use of absolute values in the parameterization of $V_{s,x}$ makes forward slip an even function of angular velocity, which we observed experimentally.

This parameterization works well in practice but also makes intuitive sense. Wheel slip is fundamentally caused by forces acting on the vehicle. The velocity terms $V_{n,x}$ and

$V_{n,\theta}$ are included because frictional contact forces are proportional to them. Centripetal acceleration ($V_{n,x}V_{n,\theta}$) and the components of the gravity vector ($g_x$, $g_y$) represent the net applied non-contact forces. Centripetal acceleration is more commonly expressed as in the form $v^2/r$ where the radius $r$ is the distance to the center of rotation. Our form is equivalent given that $v = V_{n,x}$ and $r = V_{n,x}/V_{n,\theta}$ when $V_{n,y}$ is zero. Including forward acceleration ($\dot{V}_{n,x}$) in the parameterization of forward slip is also reasonable, however, did not help for our datasets.

### 5.4. Linearized Vehicle Dynamics

Here we provide the transition matrix for the IPD vehicle model. The perturbative dynamics are used to compute the Jacobians and model the stochastic dynamics as explained previously (beginning in Section 2.1.2). The matrix integrating factor is

$$\Psi(t,\tau) = \int_\tau^t F(\zeta)\, d\zeta = \begin{bmatrix} 0 & 0 & -\Delta y \\ 0 & 0 & \Delta x \\ 0 & 0 & 0 \end{bmatrix} \tag{62}$$

The *history point displacements* are defined as $\Delta x(t,\tau) = x(t) - x(\tau)$ and $\Delta y(t,\tau) = y(t) - y(\tau)$.

The transition matrix is

$$\Phi(t,\tau) = e^{\Psi(t,\tau)} = I + \Psi(t,\tau) = \begin{bmatrix} 1 & 0 & -\Delta y \\ 0 & 1 & \Delta x \\ 0 & 0 & 1 \end{bmatrix} \tag{63}$$

This matrix encodes the relationship between position and heading error at time $\tau$ and position and heading error at the future time $t$.

The *input transition matrix* is

$$\Gamma(t,\tau) = \Phi(t,\tau)\, G(\tau) = \Phi(t,\tau)\, T(\gamma,\beta,\theta) \tag{64}$$

The $T$ matrix is expressed in full in (56). For a detailed derivation of the systematic and stochastic vehicle error dynamics refer to Kelly (2004b).

## 6. Experimental results: identifying models of slip

In this section we validate the use of IPEM to identify wheel slip parameters online. We first present experimental results obtained on Crusher, a six-wheeled skid-steered vehicle with an advanced active suspension. Crusher is capable of autonomously driving through deserts, mountains, forests, wetlands, and other extreme environments. Results obtained using the IPEM formulation are compared with those obtained using the more traditional approach of minimizing state derivative residuals (Section 6.2).

In order to show applicability to other platforms, Section 6.3 shows additional test results on the LandTamer (skid-steered, hydraulic-drive) and RecBot (Ackerman-steered, electric). In all cases, a high-end IMU and differential GPS unit were used for ground truth position measurement. Our method should work just as well with visual odometry or any other system that measures relative motion, on the scale of a few seconds, with error significantly less than the prediction errors being resolved. Slip model calibration should be achievable on all types of terrain, but note that the most dramatic improvements in motion prediction accuracy will be obtained on low-traction surfaces.

**Fig. 7.** Test vehicles. From left to right: Crusher, LandTamer, and RecBot

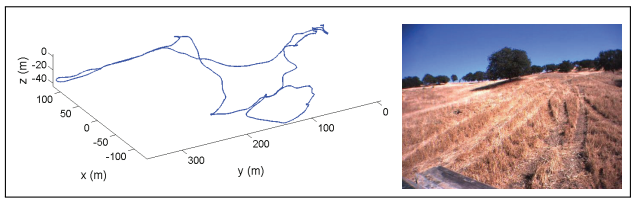


**Fig. 8.** (Left) The path of the Crusher vehicle in the Camp Roberts test. During the test Crusher traverses steep slopes; roll angles range from $-28^\circ$ to $29^\circ$ and pitch ranges from $-22^\circ$ to $17^\circ$. (Right) Image captured by one of Crusher's cameras; Crusher traverses both a dirt road and tall dry grass.

### 6.1. Crusher at Camp Roberts

In our first presented test, Crusher drove over rough, grassy terrain at Camp Roberts in California. Crusher traversed steep slopes (see Figure 8) which enabled the identifier to learn the dependence of slip on gravity, in addition to commanded velocities and accelerations. Speeds up to 6 m/s and angular velocities up to 4 rad/s were commanded. Crusher experienced lateral accelerations of up to 0.5 g as a result of slopes, and 0.3 g as a result of aggressive maneuvers.

Scatter plots of pose prediction errors (i.e. the residuals $\boldsymbol{\rho}(t)_{meas} - \boldsymbol{\rho}(t)_{pred}$) for the Camp Roberts test are shown in Figure 9. Slip parameters were calibrated using the online EKF method (Section 2.4.2) to residuals for overlapping 2-second intervals. Each dot in Figure 9 represents the along-track and cross-track error of a single pose prediction; all data was processed but for legibility only 1000 data points are plotted. These data points are equally spaced in time and span the entire 13 minute experiment.

In Figure 9(a) pose predictions are computed assuming no slip, meaning $\mathbf{u}_s$ is assumed to be zero so $\mathbf{u} = \mathbf{u}_n$, computed according to (59). This is the default skid-steer motion model commonly used when nothing is known about wheel slip. These kinematic predictions are quite poor under conditions of steep slopes, high speeds, and persistent understeer. Figure 9(b) shows prediction error *during*

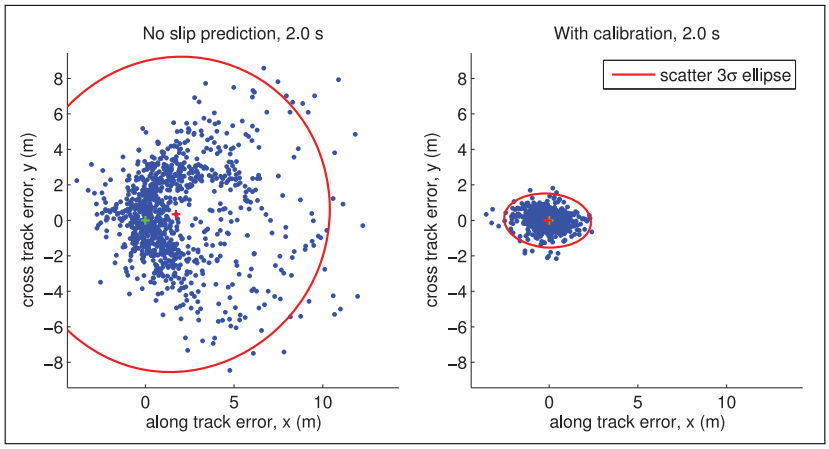


**Fig. 9.** Scatter plots of along-track and cross-track error for the Crusher vehicle at Camp Roberts. Each point represents predicted pose error for a two-second path segment. (a) Predicted pose error assuming no slip. (b) Prediction error *during online calibration*. The three standard deviation error ellipse of the points is also shown.

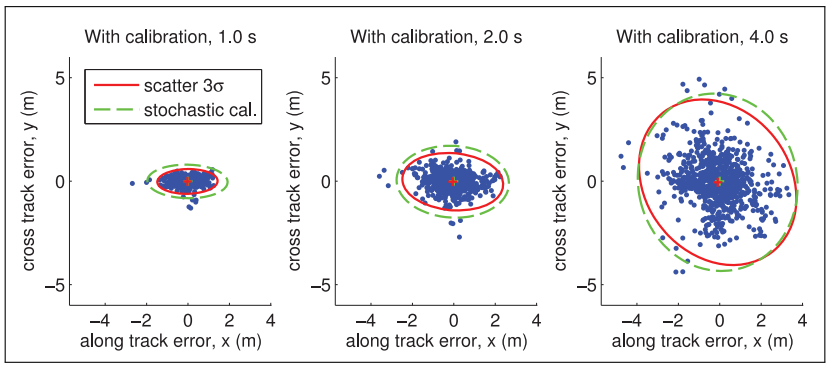


**Fig. 10.** Scatter plots of pose prediction error for 1-, 2-, and 4-second path segments. The calibrated stochastic model predicts a unique pose error covariance for each path segment. The dashed ellipses denote the average of these predictions, which accurately model the average growth and deformation of pose uncertainty in time.

*online calibration* with an initial estimate of zero for all slip parameters.

In both cases, the roll and pitch angles required at each time step during the prediction interval are computed by fitting wheel geometry to an elevation map at the predicted $(x, y, \theta)$ coordinates. The elevation map is generated using all LADAR data collected until time $t_0$ at the start of the interval. In an estimation context, logged IMU values for roll and pitch could be used instead.

Pose predictions during online calibration are significantly more accurate than no-slip predictions. Note that the mean error is reduced from 1.8 m to a few centimeters, and the standard deviation of along track error and cross track error are reduced by 72% and 83%, respectively. The standard deviation of predicted heading error is reduced by 90%.

The error that is not removed by the systematic calibration is accurately characterized by the calibrated stochastic model, as seen in Figure 10. These are scatter plots of pose prediction error just like Figure 9, but for 1-, 2-, and 4-second path segments. Each dot represents a path segment of unique shape for which a unique pose error covariance $P'(t)$ is predicted by the calibrated stochastic model according to (25). The average of these predicted covariances (denoted by the dashed ellipse) is precisely the calibrated estimate of the sample covariance of the *set* of trajectories. The sample covariance (denoted by the solid ellipse) is computed offline for comparison.

Note that the calibrated stochastic model accurately predicts the changing size and shape of predicted pose uncertainty in time. The elongation in the cross-track direction is caused by the increasing effect of heading error (caused by angular velocity perturbations) with distance traveled.

We may also evaluate the stochastic model by the Mahalanobis distances of pose residuals, which are computed as follows:

$$D_M\big(\mathbf{r}(\boldsymbol{\rho})\big) = \sqrt{\mathbf{r}(\boldsymbol{\rho})\ P'(t)^{-1}\ \mathbf{r}(\boldsymbol{\rho})^{\mathrm{T}}} \qquad (65)$$
$$\mathbf{r}(\boldsymbol{\rho}) = \boldsymbol{\rho}(t)_{meas} - \boldsymbol{\rho}(t)_{pred}$$

If the pose residuals were normally distributed with zero mean and the covariances predicted by our stochastic model

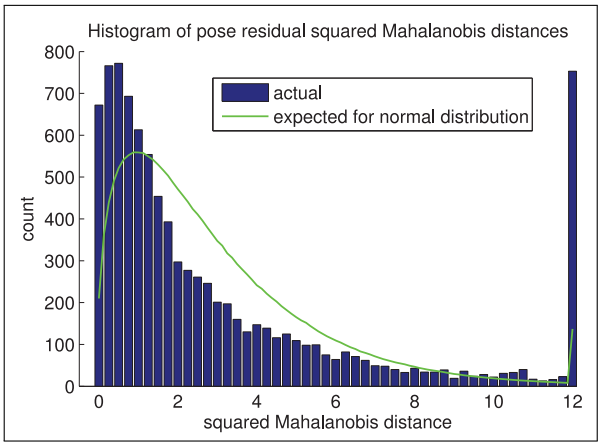


**Fig. 11.** Histogram of pose residual squared Mahalanobis distances for 2-second path segments. The bars indicate the experimental distribution. The line indicates the expected histogram if pose residuals were normally distributed about zero and the covariances predicted by the stochastic model were correct (a chi-squared distribution with 3 DOFs.) The rightmost bar counts $D_M^2$ values greater than or equal to 12.

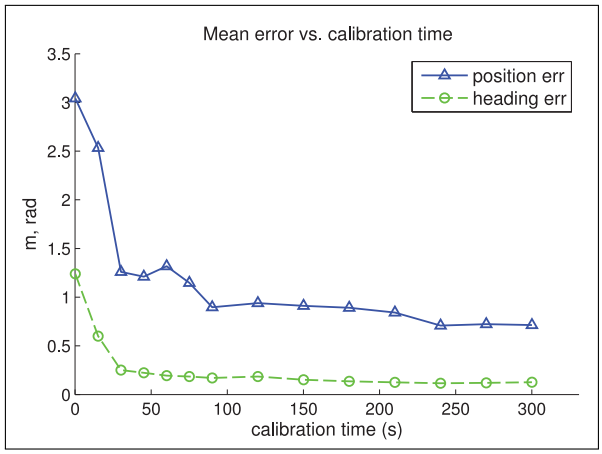


**Fig. 12.** Plots of the mean pose prediction error versus calibration time on the Camp Roberts dataset. The model is calibrated for 0–330 seconds of driving then evaluated on holdout data (the remainder of the 13 minute data log). The solid line denotes the mean Euclidean distance error of the predicted $(x, y)$ position for a 2-second path segment. The dashed line denotes the mean of the absolute value of predicted heading error. These errors do not converge to zero because there is remaining stochastic error, and the chosen metrics are always positive.

$(P'(t))$, we would expect the distribution of *squared* Mahalanobis distances $(D_M^2)$ to match a chi-squared distribution (denoted by the line in Figure 11). The observed distribution of Mahalanobis distances (denoted by histogram bars) nearly matches the expected one, but has a slightly smaller mean and more outliers.

The time to converge depends primarily on the initial parameter estimates and the diversity of input trajectories. For example, parameters associated with yaw rate cannot be calibrated until the vehicle turns, and parameters associated with gravity cannot be calibrated until the vehicle drives on an incline. Antonelli et al. (2005) and Kelly (2004b) provide some insight on trajectory shapes that are good and bad for odometry calibration (and by extension for slip calibration). For example, closed loops are a bad choice because some systematic errors cancel out.

Figure 12 shows the pose prediction performance on holdout Camp Roberts data after calibrating for limited periods ranging from 0 to 330 seconds. Based on pose prediction error, the filter nearly converges within 1.5 minutes of driving when starting from the uncalibrated case (i.e. all slip parameters initialized to zero). In practice, one could use the parameters calibrated for the same vehicle on different terrain as a better initial guess, thereby decreasing the convergence time.

Note that, even though the filter has not fully converged during the first 1.5 minutes, the *online* pose prediction accuracy during that period is still greatly improved over the no-slip predictions, as seen in Figure 9.

The slip surfaces learned by the filter on the Camp Roberts dataset are shown in Figure 13. The directions of the forward, lateral, and angular slip velocities of the body frame correspond to the directions of $V_x$, $V_y$, and $V_\theta$ in Figure 6. Note that forward slip is negatively correlated with the absolute value of angular velocity, $|V_\theta|$. Lateral slip depends primarily on centripetal acceleration $(V_x V_\theta)$. For this skid-steered platform, angular slip is predominantly a linear function of angular velocity. As expected, the filter learned a positive correlation between forward slip and the $x$ component of the gravity vector $(V_{s,x} = \ldots + 0.307 g_x)$ and between lateral slip and the $y$ component $(V_{s,y} = \ldots + 0.064 g_y)$.

Note that our raw residuals cannot be plotted with the surfaces in Figure 13. Instantaneous velocity residuals could be plotted as points, but our residuals are of poses. We predict pose over an interval of seconds during which the velocity is free to vary and is not measured.

Figure 14 shows examples of extreme cases that are predicted well in the Crusher datasets. First, it is important to note that Crusher's effective turn rate is only a third of the commanded rate. This is partly due to the fact that four of six wheels are slipping sideways and resisting motion in a tight turn. Furthermore, when turning on a hill, the wheels on the high side carry reduced load and are therefore unable to generate as much traction as on level ground. The result is that turning becomes impossible and turn commands cause more or less translational motion perpendicular to the terrain gradient. These effects and others depend on the orientation of the terrain gradient in the body frame, just as our model is formulated to learn them.

## 6.2. Comparison with traditional approach

Here we compare results obtained using the IPEM approach to those obtained using the more traditional approach, in

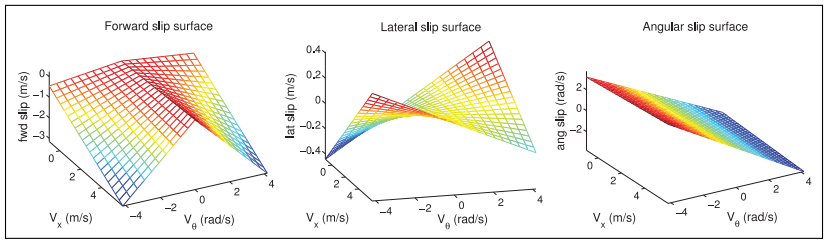


**Fig. 13.** Plots of the slip surfaces for the Crusher Camp Roberts dataset. These surfaces show the *predicted* forward, lateral, and angular slip of the body frame as a function of the nominal (or commanded) forward and angular velocity ($V_x$ and $V_\theta$), according to (60). These surfaces correspond to zero $x$ and $y$ components of the gravity vector (i.e. driving on level ground).

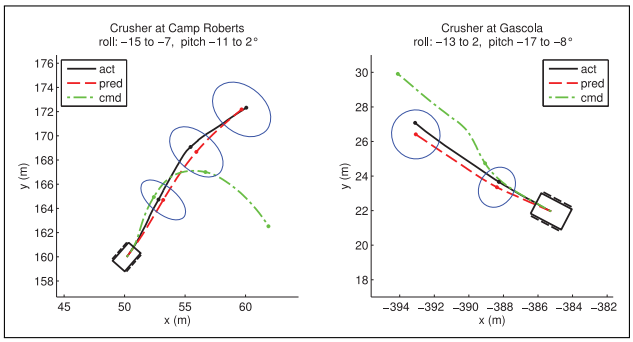


**Fig. 14.** Aerial views of predicted paths without slip modeling (dash-dot line), with slip modeling (dashed line), and ground truth paths (solid line) under different conditions for Crusher. The paths are 6 and 4 seconds long; the ellipses are $1\sigma$ estimates of uncertainty at two second intervals. (Left) Crusher driving on a hillside at Camp Roberts with a roll angle up to -15°. Commanded turns in either direction are correctly predicted to be impossible. (Right) Crusher driving uphill at Gascola on a 17° slope. A 25% longitudinal wheel slip is correctly predicted.

which we minimize state derivative residuals. Because we chose slip velocity $\mathbf{u}_s$ to be a linear function of the slip parameters $\alpha$, we find the least-squares solution for $\alpha$ by solving the following system of linear equations:

$$\begin{bmatrix} C^1 \\ \vdots \\ C^N \end{bmatrix} \alpha = \begin{bmatrix} \mathbf{u}^1_{meas} - \mathbf{u}^1_n \\ \vdots \\ \mathbf{u}^N_{meas} - \mathbf{u}^N_n \end{bmatrix} \tag{66}$$

where the superscripts denote the time step (1 to $N$), $C$ denotes the coefficient matrix from (61), $\mathbf{u}_n$ the nominal vehicle velocity, and $\mathbf{u}_{meas}$ the measured velocity. here $\mathbf{u}_{meas}$ is obtained by numerically differentiating the pose measurements using the central difference:

$$\mathbf{u}^k_{meas} = T(\gamma, \beta, \theta)^{-1} \frac{\boldsymbol{\rho}^{k+1}_{meas} - \boldsymbol{\rho}^{k-1}_{meas}}{t^{k+1} - t^{k-1}} \tag{67}$$

The transform matrix $T$ in (67) is the inverse of the matrix in (55).

Figure 15 compares pose prediction accuracy for slip models calibrated using IPEM and this traditional approach. Note that when the time between measurements is small, the performance of the two approaches is comparable; however, with longer delays between measurements, the traditional approach suffers because the numerical derivatives are increasingly poor approximations of the actual velocity. In contrast, the IPEM approach performs well even with long delays between pose measurements. This makes IPEM ideal for situations in which high-frequency RTK GPS is unavailable, and pose fixes can only be obtained intermittently (e.g. from sparse landmarks). Note that Crusher is an extremely well-engineered vehicle that accurately timestamps all sensor data using GPS time tags. On a typical WMR timestamps are less precise and, as a result, numerical derivatives would be less accurate.

The traditional approach to stochastic calibration uses (21) directly. The maximum-likelihood estimate of the input noise $Q$ is the sample covariance of the measured velocity perturbations (with the systematic error removed):

$$Q = \frac{1}{N} \sum_{i=1}^{N} (\delta \mathbf{u}^i)(\delta \mathbf{u}^i)^{\mathrm{T}} \tag{68}$$
$$\delta \mathbf{u}^i = \mathbf{u}^i_{meas} - (\mathbf{u}^i_n + C^i \alpha)$$

where $\alpha$ is identified using (66).

Even with accurate velocity measurements (obtained using the highest available pose measurement frequency), calibrating $Q$ in this way does not accurately model the propagation of pose uncertainty. In Figure 16 this approach severely underestimates the uncertainty of future poses. This is because our stochastic model assumes white noise, but the input noise is clearly correlated in time as seen in Figure 17. Even though our assumption of white noise is not correct, the IPEM approach still fits a model that closely approximates the observed pose uncertainty propagation (see Figures 10 and 11).

## 6.3. *Additional datasets*

To show applicability to other terrain conditions and vehicle configurations, this section presents results on additional datasets. The quantitative performance of the systematic calibration for each dataset is summarized in Table 2. Note that the results shown are not the training error but the test error *during online calibration* of the vehicle model. Pose predictions are made using estimates of the slip parameters

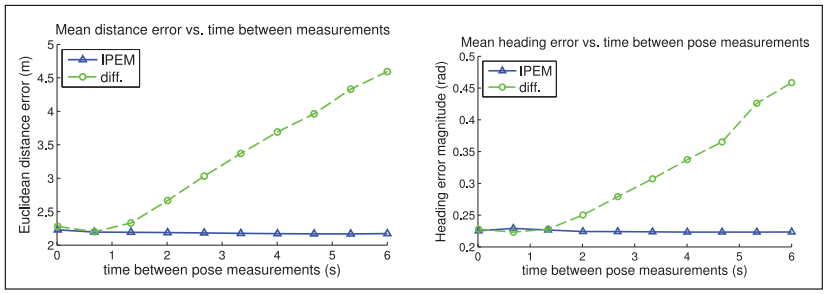


**Fig. 15.** Calibration performance versus time between pose measurements. Reported errors are for 6-second path segments. Owing to the limited data, path segments overlap. Performance degrades with longer delays between measurements using the traditional approach (diff.), which requires numerical differentiation of measurements to form state derivative residuals. Performance remains constant for the IPEM approach.

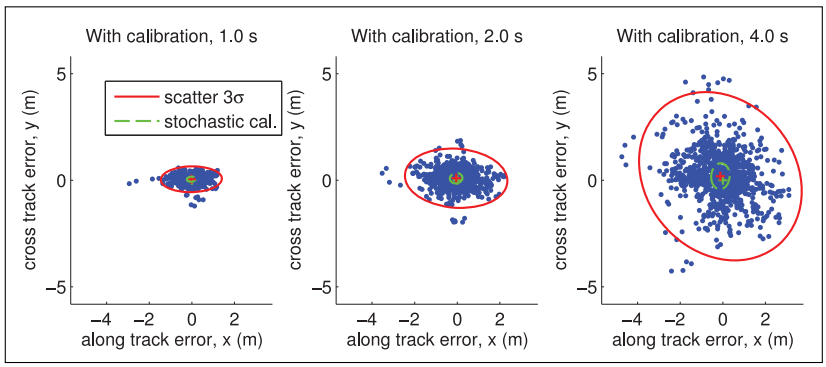


**Fig. 16.** Pose prediction errors for 1-, 2-, and 4-second path segments for the Crusher at Camp Roberts dataset. Dashed line ellipses represent pose error covariance predictions for a stochastic model calibrated to state derivative residuals (i.e. velocity residuals). When calibrated using this traditional approach, the stochastic model severely underestimates uncertainty.

**Table 2.** Improvements in pose prediction accuracy during online calibration in multiple tests.

| Dataset | Reduction in pose prediction error[1] (along-track, cross-track, heading) | |
|---|---|---|
| | $\mu$ | $\sigma$ |
| Crusher-Camp Roberts | 95%, 97%, 92% | 72%, 83%, 90% |
| Crusher-Gascola | 99%, 0%[2], 89% | 73%, 82%, 86% |
| LandTamer-circles | 88%, 99%, 99% | 57%, 88%, 84% |
| RecBot-yard | 93%, 0%[2], 64% | 47%, 65%, 16% |

[1]Compared with the no-slip model

[2]No improvement, but no-slip prediction error was already small (<0.03 m)

based only on observations prior to the time at which the prediction is made ($t_0$).

In the "Crusher-Gascola" test, Crusher was driven on muddy slopes during a rainstorm at Gascola, Pennsylvania. In the "LandTamer-circles" test, data was collected as the LandTamer drove in circles at various speeds and curvatures (0.25–1.0 m/s and 0.4–0.6 $m^{-1}$) in a muddy gravel lot after a heavy rain. Finally, in the "RecBot-yard" test, data was collected as the RecBot drove randomly on a grass lawn for just over five and half minutes at speeds up to 4.8 m/s. The grass was mostly level and flat, except for large tractor treads that added variance to the slip rates. The RecBot is a medium size drive-by-wire UGV and is Ackerman-steered in contrast to the previous skid-steered datasets. The RecBot results are less pronounced than those generated on skid-steer platforms because its angular slip is less significant. A longer prediction interval was chosen for the RecBot (6 versus 2 seconds) to emphasize the improved performance of the calibrated model.

## 6.4. Combined slip and odometry calibration

Here we describe how IPEM can be used to calibrate slip and odometry parameters *simultaneously*. This is a straightforward extension to the slip calibration algorithm already

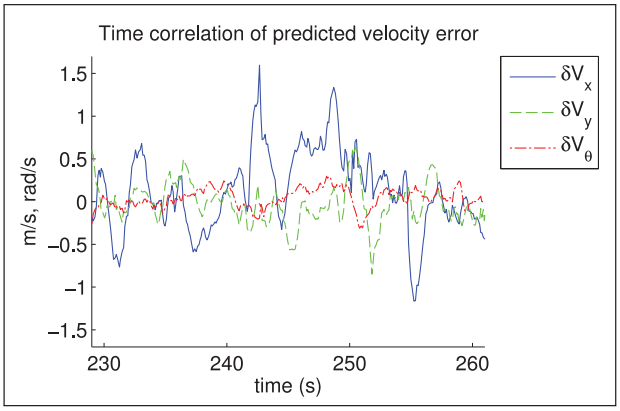


**Fig. 17.** Time correlation of predicted velocity error during a representative 30-second window of the Crusher at Camp Roberts dataset.

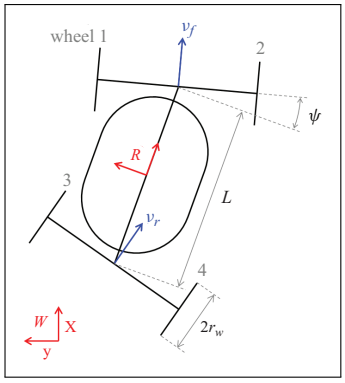


**Fig. 18.** Diagram of the Zoë rover showing the world and robot frames, front and rear axle velocities $v_f$ and $v_r$, steer angle $\psi$, and dimensions.

presented. Instead of attributing systematic velocity errors $\delta\mathbf{u}$ to errors in the slip velocity estimates $\mathbf{u}_s(\alpha)$, we also attribute them to errors in the nominal velocity $\mathbf{u}_n(\beta)$. In Seegmiller et al. (2012), the authors demonstrated the use of IPEM to identify odometry parameters $\beta$ for the passively steered Zoë rover. We present a summary here.

The forward kinematics of the Zoë rover are as follows. The nominal vehicle velocity (assuming no wheel slip) $\mathbf{u}_n$ is a function of the odometry parameters $\beta$ and the angular velocities of the four wheels $\omega$:

$$\begin{bmatrix} V_{n,x} \\ V_{n,y} \\ V_{n,\theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & \frac{1}{L} & 0 & -\frac{1}{L} \end{bmatrix} \left( \frac{r_w}{2} \begin{bmatrix} (\omega_1+\omega_2)\cdot\cos\psi_f \\ (\omega_1+\omega_2)\cdot\sin\psi_f \\ (\omega_3+\omega_4)\cdot\cos\psi_r \\ (\omega_3+\omega_4)\cdot\sin\psi_r \end{bmatrix} \right) \tag{69}$$

where measured steer angle ($\psi$) is a linear function of the potentiometer voltage ($V$):

$$\psi_f = m_f V_f + b_f, \quad \psi_r = m_r V_r + b_r \tag{70}$$

The vector of odometry parameters $\beta$ is

$$\beta = \begin{bmatrix} r_w & m_f & b_f & m_r & b_r \end{bmatrix}^{\mathrm{T}} \tag{71}$$

Because we attribute systematic error to both the slip and odometry parameters, the state in the identification filter $\chi$ is the combined vector of parameters:

$$\mathbf{p} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{72}$$

Based on (16) The measurement Jacobian is

$$H_{sys} \approx \int_{t_0}^{t} \Gamma(t,\tau) \left( \frac{\mathrm{d}\mathbf{u}_n(\mathbf{p},\tau)}{\mathrm{d}\mathbf{p}} + \frac{\mathbf{du}_s(\mathbf{p},\tau)}{\mathrm{d}\mathbf{p}} \right) d\tau \tag{73}$$

According to Section 5.3, the nominal velocity $\mathbf{u}_n$ is only a function of $\beta$; however, the slip velocity $\mathbf{u}_s$ is a function of both $\alpha$ and $\beta$ (because $\mathbf{u}_s$ is a function of $\mathbf{u}_n(\beta)$). In our experiments with Zoë, the parameter values calibrated online using IPEM were more conveniently obtained than manually calibrated values, and produced more accurate simulations.

Note that observability issues can arise when calibrating odometry and slip parameters simultaneously. Depending on the parameterization of slip, some parameters may be indistinguishable. For example, an incorrect wheel radius is impossible to distinguish from longitudinal wheel slip that is proportional to velocity. Parameters can not be independently distinguished if they produce linearly dependent columns in the measurement Jacobian, $H$. In practice, it may not matter if the error is attributed to wheel slip or an odometry parameter as long as the calibrated model can accurately predict trajectories.

## 7. Conclusions

In this paper we have presented our IPEM approach to dynamic model identification. The IPEM formulation is completely general; it works for any differential equation, and we have shown that it can be used for arbitrary parameterizations including parameterized disturbance inputs. Although the technique is general, we have concentrated on its application to vehicle models and thereby highlighted some of its advantages for the robotics community. Our interest in this problem of model identification stems from complete system failures in motion planning and control due to inaccurate models, as well as a realization that our capacity to simulate and even invert dynamic vehicle models was developed far beyond our capacity to produce a useful one. We were also dissatisfied with our incapacity to characterize the uncertainty in the predictions of the model in a way that is calibrated to real experience.

While the core idea of integrating the prediction is simple enough, its realization in a working system raises numerous formulation questions that we address in the paper. One fundamental issue is the path dependence of errors. We use linear systems theory to derive the IPD of the system, which express this dependence in *closed form* using the vector and matrix convolution integrals. Then the errors of interest are dependent, at most, on the instantaneous state and inputs, just as they would be in a more traditional approach (based

on one-step-ahead prediction or state derivative residuals). We can therefore calibrate to and simulate trajectories of arbitrary shape.

Another issue is the high cost of computing the Jacobian of integral predictions with respect to the parameters. We show how this Jacobian can be computed very efficiently using the IPD and Leibniz's rule to move differentiation inside the integral. Derivation of the IPD solves two other problems. First, it provides a transition matrix that is exactly the transformation needed to account for the effect of error in the initial conditions on error at the trajectory endpoint. Second, with little effort it provides a model of the stochastic dynamics. We explain the calculation of scatter matrix observations and their covariance necessary to calibrate the stochastic model.

The benefits of the IPEM approach are made clear by our results in vehicle model identification. We showed how IPEM performs better than the traditional approach of calibrating to state derivative residuals under certain conditions, such as when state measurements are significantly time-separated or when disturbances are time-correlated. We showed how IPEM can accurately fit our stochastic model to real data, even when the true process of uncertainty propagation may not be linear and noise may not be white as assumed. Although IPEM may appear to be more computationally expensive than traditional techniques, we have found it to be lightweight. In all tests our IPEM calibration code ran much faster than real time. Furthermore, residuals are formed from predictions and measurements that may already be computed by autonomous control and estimation systems.

Our unified approach to the identification of systematic and stochastic motion models enables WMRs to be much more informed about their own mobility. This could greatly benefit both WMR estimation and planning/control systems. By modeling the highly predictable component of wheel slip, WMRs can accurately dead-reckon their position in the absence of GPS measurements. Furthermore they can plan feasible paths on steep hillsides and use calibrated uncertainty models to assess the risk of candidate paths.

### 7.1. Future work

In this paper, we have chosen to use a simplified 3D vehicle model in which *body frame* slip is parameterized as though for a unicycle. While this works well experimentally in many applications, in future work we will investigate the identification of full 3D velocity kinematic models using IPEM, in which suspension deflections are accounted for and slip is parameterized at the individual wheels. This greater fidelity is important for highly articulated vehicles on uneven terrain.

Because our vehicle model is velocity-based and not force-based, skidding with locked brakes and certain other slip scenarios are not predicted, but will be investigated in future work. One possibility is to identify full dynamic models that account for vehicle inertia, but simulation of these models would be computationally more expensive.

In related work, we studied the coupling of slip models learned via IPEM with inertial navigation (Rogers-Marcovitz et al., 2012). We are also investigating the generation of maps of spatial variations in terrain traction properties at test sites based on slip observations. Experimentally, we also hope to evaluate our approach on an even broader variety of platforms and terrain types. Ideally terrain-dependent effects can be factored out so that information can be shared between platforms.

**Funding**

This research was made with US Government support awarded by the Army Research Office (grant number W911NF-09-1-0557), the Army Research Laboratory (grant number W911NF-10-2-0016), and by the DoD, Air Force Office of Scientific Research, and a National Defense Science and Engineering Graduate (NDSEG) Fellowship (32 CFR 168a).

**References**

Abbeel P, Coates A, Montemerlo M, Ng AY and Thrun S (2005) Discriminative training of Kalman filters. In: *Proceedings of Robotics: Science and Systems*, Cambridge, MA.

Angelova A, Matthies L, Helmick D, Sibley G and Perona P (2006) Learning to predict slip for ground robots. In: *Proceedings IEEE International Conference on Robotics and Automation*.

Antonelli G, Chiaverini S and Fusco G (2005) A calibration method for odometry of mobile robots based on the least-squares technique: theory and experimental validation. *IEEE Transactions on Robotics* 21(5): 994–1004.

Bode M (2007) *Learning the Forward Predictive Model for an Off-road Skid-steer Vehicle*. Technical Report CMU-RI-TR-07-32, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Bohrnstedt G and Goldberger A (1969) On the exact covariance of products of random variables. *Journal of the American Statistical Association* 64(328): 1439–1442.

Borenstein J and Feng L (1996) Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation* 12: 869–880.

Brogan WL (1991) *Modern Control Theory*. Englewood Cliffs, NJ: Prentice-Hall.

Burke M (2012) Path-following control of a velocity constrained tracked vehicle incorporating adaptive slip estimation. In: *Proceedings IEEE International Conference on Robotics and Automation*.

Censi A, Marchionni L and Oriolo G (2008) Simultaneous maximum-likelihood calibration of odometry and sensor parameters. In: *Proceedings IEEE International Conference on Robotics and Automation*.

Chong KS and Kleeman L (1997) Accurate odometry and error modeling for a mobile robot. In: *Proceedings IEEE International Conference on Robotics and Automation*.

Duong HN and Landau ID (1994) On test horizon for model validation by output error. *IEEE Transactions of Automatic Control* 39(1): 102–106.

Durrant-Whyte HF (1996) An autonomous guided vehicle for cargo handling applications. *The International Journal of Robotics Research* 15(5): 407–440.

Farina M and Piroddi L (2008) Some convergence properties of multi-step prediction error minimization identification criteria. In: *Proceedings IEEE Conference on Decision and Control*.

Grip HF, Imsland L, Johansen TA, Fossen TI, Kalkkuhl JC and Suissa A (2008) Nonlinear vehicle side-slip estimation with friction adaptation. *Automatica* 44(3): 611–622.

Hoff P and Niu X (2012) A covariance regression model. *Statistica Sinica* 22: 729–753.

Ishigami G, Miwa A, Nagatani K and Yoshida K (2007) Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil. *Journal of Field Robotics* 24(3): 233–250.

Jørgensen JB and Jørgensen SB (2007) MPC-relevant prediction-error identification. In: *Proceedings of the American Control Conference*.

Kelly A (2004a) Fast and easy systematic and stochastic odometry calibration. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

Kelly A (2004b) Linearized error propagation in odometry. *The International Journal of Robotics Research* 23(1): 179–218.

Kelly A and Seegmiller N (2012) A vector algebra formulation of mobile robot velocity kinematics. In: *Proceedings of the Field and Service Robotics*.

Kelly J and Sukhatme GS (2010) A general framework for temporal calibration of multiple proprioceptive and exteroceptive sensors. In: *Proceedings of the International Symposium on Experimental Robotics*, New Delhi, India.

Kruschke JK (2008) Bayesian approaches to associative learning: From passive to active learning. *Learning and Behavior* 36(3): 210–226.

Kümmerle R, Grisetti G and Burgard W (2012) Simultaneous parameter calibration, localization, and mapping. *Advanced Robotics* 26(17): 2021–2041.

Lenain R, Thuilot B, Cariou C and Martinet P (2010) Mixed kinematic and dynamic sideslip angle observer for accurate control of fast off-road mobile robots. *Journal of Field Robotics* 27(2): 181–196.

Ljung L (1987) *System Identification - Theory for the User*. Englewood Cliffs, NJ: Prentice-Hall.

Low CB and Wang D (2007) Integrated estimation for wheeled mobile robot posture, velocities, and wheel skidding perturbations. In: *Proceedings IEEE Intertnational Conference on Robotics and Automation*.

Lucet E, Grand C, Sallé D and Bidaud P (2008) Dynamic sliding mode control of a four-wheel skid-steering vehicle in presence of sliding. In: *Proceedings of the RoManSy*, Tokyo, Japan.

Martinelli A (2002) The odometry error of a mobile robot with a synchronous drive system. *IEEE Transactions on Robotics and Automation* 18(3): 399–405.

Martinelli A, Tomatis N and Siegwart R (2007) Simultaneous localization and odometry self calibration for mobile robot. *Autonomous Robots* 22(1): 75–85.

Nicolao GD (1997) System identification: Problems and perspectives. In: *12th Workshop on Qualitative Reasoning*.

Rogers-Marcovitz F, George M, Seegmiller N and Kelly A (2012) Aiding off-road inertial navigation with high performance models of wheel slip. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

Rogers-Marcovitz F and Kelly A (2010) On-line mobile robot model identification using integrated perturbative dynamics. In: *Proceedings of the International Symposium on Experimental Robotics*.

Roy N and Thrun S (1998) Online self-calibration for mobile robots. In: *Proceedings IEEE International Conference on Robotics and Automation*.

Rudolph A (2003) Quantification and estimation of differential odometry errors in mobile robotics with redundant sensor information. *The International Journal of Robotics Research* 22(2): 117–128.

Seegmiller N, Rogers-Marcovitz F and Kelly A (2012) Online calibration of vehicle powertrain and pose estimation parameters using integrated dynamics. In: *Proceedings of the International Conference on Robotics and Automation*.

Seegmiller N, Rogers-Marcovitz F, Miller G and Kelly A (2011) A unified perturbative dynamics approach to vehicle model identification. In: *Proceedings of the International Symposium on Robotics Research*.

Seneviratne L, Zweiri Y, Hutangkabodee S, et al. (2009) The modeling and estimation of driving forces for unmanned ground vehicles in outdoor terrain. *International Journal of Modeling, Identification, and Control* 6(1): 40–50.

Setterfield TP and Ellery A (2013) Terrain response estimation using and instrumented rocker-bogie mobility system. *IEEE Transactions on Robotics* 29(1): 172–188.

Sheehan M, Harrison A and Newman P (2010) Automatic self-calibration of a full field-of-view 3D n-laser scanner. In: *Proceedings of the International Symposium on Experimental Robotics*, New Delhi, India.

Siciliano B and Khatib O (eds.) (2008) *Springer Handbook of Robotics*, chapter 14. Berlin: Springer.

Söderström T and Stoica P (1989) *System Identification*. Englewood Cliffs, NJ: Prentice-Hall.

Song X, Song Z, Seneviratne LD and Althoefer K (2008) Optical flow-based slip and velocity estimation technique for unmanned skid-steered vehicles. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

Terry JD and Minor MA (2008) Traction estimation and control for mobile robots using the wheel slip velocity. In: *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

Urmson C, Anhalt J, Bae H, Bagnell JA, Baker CR, Bittner RE et al. (2008) Autonomous driving in urban environments: Boss and the grand challenge. *Journal of Field Robotics* 25(8): 425–466.

Ward C and Iagnemma K (2008) A dynamic-model-based wheel slip detector for mobile robots on outdoor terrain. *IEEE Transactions on Robotics* 24(4): 821–831.

Yi J, Wang H, Zhang J, Song D, Jayasuriya S and Liu J (2009) Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation. *IEEE Transactions on Robotics* 25: 1087–1097.

Yu W, Chuy O, Collins E and Hollis P (2010) Analysis and experimental verification for dynamic modeling of a skid-steered wheeled vehicle. *IEEE Transactions on Robotics* 26(2): 340–353.