Trajectory Optimization for Well-Conditioned Parameter Estimation

Andrew D. Wilson, *Student Member, IEEE,* Jarvis A. Schultz, *Student Member, IEEE,* and Todd D. Murphey, *Member, IEEE*

***Abstract*—When attempting to estimate parameters in a dynamical system, it is often beneficial to strategically design experimental trajectories that facilitate the estimation process. This paper presents an optimization algorithm which improves conditioning of estimation problems by modifying the experimental trajectory. An objective function which minimizes the condition number of the Hessian of the least-squares identification method is derived and a least-squares method is used to estimate parameters of the nonlinear system. A software-simulated example demonstrates that an arbitrarily designed trajectory can lead to an ill-conditioned least-squares estimation problem, which in turn leads to slower convergence to the best estimate and, in the presence of experimental uncertainties, may lead to no convergence at all. A physical experiment with a robot-controlled suspended mass also shows improved estimation results in practice in the presence of noise and uncertainty using the optimized trajectory.**

***Note to Practitioners*—This paper presents a software-automated method to design the time-varying control input for a dynamic system when attempting to better estimate model parameters. This type of scenario could include an automated system actively inspecting multi-body parts for elastic or damping coefficients or a robot attempting to better estimate its own inertias. The method requires known equations of motion of a system with unknown, constant parameters. We show—both in simulation and in experiment—that using the algorithm to optimize the control input results in better convergence of parameter estimates for simulated and experimental systems.**

***Index Terms*—optimal control, parameter estimation, iterative methods in optimization**

I. Introduction

In robotics and automation environments, it can be desirable to have automatic controllers capable of refining estimates of model parameters, such as the mass of an object, damping in a mechanical system, or geometric properties. To obtain better estimates, a model for the system is assumed and the system's trajectory is experimentally measured and compared to the expected trajectory in a least-squares sense [1]. First order gradient descent or second order Newton's method type optimization routines are utilized to determine values for the system parameters which minimize the least-squares error between the model-predicted system trajectories and the experimental data.

A. Wilson, J. Schultz, and T. Murphey are with the Department of Mechanical Engineering, Northwestern University, 2145 Sheridan Road, Evanston, IL 60208, USA. awilson@u.northwestern.edu, jschultz@u.northwestern.edu, and t-murphey@northwestern.edu

This is the accepted preprint version of the manuscript. The final published version is available at http://dx.doi.org/10.1109/TASE.2014.2323934.

One major choice in designing the experiment to obtain accurate estimates of the parameter values is the set of control inputs that will drive the experimental system. A change in the control inputs can have a significant impact on the ability to estimate the model parameters. In an extreme case, a trajectory could be chosen such that a zero eigenvalue exists in the least-squares Hessian, thus making any estimate of that parameter impossible.

A wide variety of work has been performed in the areas of experiment design, input design, and identifiability of parameters [2]–[8]. Highlights of a relevant subset of this literature include work by Armstrong on optimal "exciting" trajectories [9] and work on minimal parameter set methods by Gautier and Khalil [10]. These optimization methods synthesize trajectories for systems that have nonlinear dynamics with respect to the state but must be linear with respect to the parameters. Related work by Swevers also examines parameter estimation trajectories for the same class of systems but from the perspective of the Fisher information matrix [11]. In both cases, the analysis is performed on a discrete number of measurements taken from the trajectory, and a finite discretization of the dynamics is used with time-discretized inputs. Many other methods in the area of experiment design also rely on discrete optimizations which discretize the continuous dynamics [12]–[14].

Additional studies have been performed in areas such as computational biology [15], [16] and aerospace [17], with emphasis being placed on trajectory generation using a variety of basis functions or splines [18]–[21]. These methods allow one to optimize over a cost index, such as the condition number, using a finite dimensional optimization method over a fixed set of basis functions. While this allows the trajectory to be expressed in the continuous time domain, the space of allowable control signals is still finite.

A. Contribution and Related Work

This paper presents an extension and refinement of work originally published at the IEEE Conference on Automation Science and Engineering [22]. Using an infinite-dimensional, projection-based optimization method originally designed for Bolza-type trajectory tracking problems [23], the original work presents a formulation of the (non-Bolza) cost function to achieve conditioning optimization for parameter estimation. The cost function includes a term dependent on eigenvalues of the Hessian matrix of the parameter estimation problem which directly affects convergence rates of gradient descent based parameter optimization method.

Additionally, the formulation of the cost function requires additional states to be added to the trajectory optimization problem that are related to sensitivities of the trajectory with respect to the system parameters to be identified. In this optimization method, continuous-time dynamics are not discretized; rather, optimal variations of the input and trajectory are constructed locally and projected onto the trajectory manifold of feasible nonlinear executions of the dynamical system. This paper presents a compact version of the original algorithm which has been refined to include a dependence only on the first-order sensitivity as opposed to the first and second order sensitivities presented in the original work. The algorithm has also been extended to handle nonlinear output functions in addition to the nonlinear dynamic model. Finally, experimental results are presented in this paper to validate the theoretical claims of the original algorithm.

This paper is organized as follows: Section II covers the least-squares parameter optimization method; Section III presents the reformulated trajectory optimization method and derives the equations necessary to perform the optimization with respect to the condition number; and Sections IV and V present experimental results of condition number optimization for a cart and suspended mass system.

II. Nonlinear Least-Squares Estimation

For nonlinear dynamical systems, least-squares estimation methods can be used to estimate system parameters [24], [25]. We begin with a brief formulation of the estimation problem for completeness and to introduce notation. Given a nonlinear dynamical system, the system dynamics can be written as

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t), \theta) \\ \mathbf{y} &= \mathbf{g}(t, \mathbf{x}(t), \mathbf{u}(t), \theta)\end{aligned} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ defines the system states, $\mathbf{u} \in \mathbb{R}^r$ represents the control inputs, and $\theta \in \mathbb{R}^p$ is a vector of static model parameters to be estimated. The system output, $\mathbf{y} \in \mathbb{R}^h$ can also be a nonlinear function of the states, controls and parameters. To simplify the notation, the time arguments will be dropped though the input and trajectory remain time-varying.

For this algorithm, we assume that measurements of the system output contain noise which is normally distributed and zero mean. The result of this assumption is that minimization of the least-square error in the system output is equivalent to maximum likelihood estimation of the system parameters [1]. Therefore, the least-squares cost function for the estimation algorithm is given by

$$J_p = \frac{1}{2} \int_{t_0}^{t_f} [\mathbf{g}(\mathbf{x}, \mathbf{u}, \theta) - \tilde{\mathbf{y}}]^T \cdot Q_\theta \cdot [\mathbf{g}(\mathbf{x}, \mathbf{u}, \theta) - \tilde{\mathbf{y}}] \, dt \tag{2}$$

where $\tilde{\mathbf{y}}$ denotes the measured output and Q_θ is an $h \times h$ positive semi-definite weighting matrix. Although not covered in this paper, estimates of the sensor covariance are commonly used as weights for Q_θ, which is analogous to estimating the covariances in continuous time filtering [26].

The minimization of (2) is treated as an unconstrained optimization problem with no bounds placed on the values of the parameters. A variety of optimization methods can be used to find an optimizer to this problem; however, this paper will focus on two approaches: gradient descent optimization and the Iterated Least-Squares method defined in [1]. The two approaches are based on the first and second derivatives of the cost function (2) with respect to θ.

A. Gradient Descent Optimization

Gradient descent techniques are commonly used in optimization problems due to their simplicity and because only the first derivative of the objective function needs to be calculated. The first derivative can be found by differentiating (2) with respect to θ. This results in

$$\begin{aligned} D_\theta J_p(\theta) = \int_{t_0}^{t_f} & [\mathbf{g}(\cdot) - \tilde{\mathbf{y}}]^T \cdot Q_\theta \cdot \\ & [D_x \mathbf{g}(\mathbf{x}, \mathbf{u}, \theta) \cdot D_\theta \mathbf{x}(\mathbf{x}, \mathbf{u}, \theta) + D_\theta \mathbf{g}(\mathbf{x}, \mathbf{u}, \theta)] \, dt. \end{aligned} \tag{3}$$

Note that the form $D_x J$ represents the derivative of J with respect to $\mathbf{x}$. Since $\mathbf{x}$ is constrained by the system dynamics through (1), the derivative $D_\theta \mathbf{x}$ is also constrained by a differential equation given by

$$\dot{\psi}(\mathbf{x}, \mathbf{u}, \theta) = D_x \mathbf{f}(\mathbf{x}, \mathbf{u}, \theta) \cdot \psi(\mathbf{x}, \mathbf{u}, \theta) + D_\theta \mathbf{f}(\mathbf{x}, \mathbf{u}, \theta) \tag{4}$$

where

$$\psi(\mathbf{x}, \mathbf{u}, \theta) = D_\theta \mathbf{x}(\mathbf{x}, \mathbf{u}, \theta)$$

with the initial condition at $t = 0$, $\psi(\mathbf{x}, \mathbf{u}, \theta) = \{0\}^{n \times p}$.

The estimation algorithm is applied as shown by Algorithm 1. An Armijo backtracking line-search is used at every iteration to ensure that the new parameter estimate sufficiently decreases the least-squares cost [27].

With respect to algorithm convergence, gradient descent optimization ensures convergence to a local minimizer; however, the rate of convergence can be extremely slow in practice. Given a strongly convex least-squares problem, it has been shown in prior literature [28] that the error of the parameter estimates follows

$$\frac{\|\theta_{i+1} - \theta^*\|}{\|\theta_i - \theta^*\|} \leq \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} \right),$$

where λ_n and λ_1 are the respective maximum and minimum eigenvalues of the Hessian of J_p. This result indicates that as the condition number $\kappa = \lambda_n / \lambda_1$ increases, the convergence rate of the gradient descent algorithm degrades.

Algorithm 1 Gradient Descent Parameter Estimation

```
Choose initial θ_0 ∈ ℝ^p, tolerance ϵ
while D_θ J_p(θ_i) > ϵ do
    d_i = −D_θ J_p(θ_i) from (3)
    Compute γ_i using Armijo backtracking search
    θ_{i+1} = θ_i + γ_i d_i
    i = i + 1
end while
```

B. Iterated Least-Squares Optimization

Since gradient descent-based optimization has slower convergence with increasingly ill-conditioned problems, Hessian-based methods may be used to improve convergence rates. The Iterated Least-Squares method uses an approximation of the Hessian to perform a quasi-Newton type optimization [1]. For the Iterated Least Squares method, the parameter estimate update is given by the following,

$$\theta_{i+1} = \theta_i + \left[\int_{t_0}^{t_f} \ell_p(\cdot)^T \cdot Q_\theta \cdot \ell_p(\cdot)\ dt \right]^{-1} \cdot D_\theta J_p(\theta_i) \tag{5}$$

where

$$\ell_p(\cdot) = D_x \mathbf{g}(\cdot) \cdot \psi(\cdot) + D_\theta \mathbf{g}(\cdot).$$

For compactness, arguments of previously defined functions have been omitted and are replaced by $(\cdot)$.

To show how one obtains the approximate Hessian used in (5), the full Hessian matrix will be derived in a similar manner to the first derivative. Taking the derivative of (3) results in the following:

$$D_p^2 J_p(\mathbf{x}, \mathbf{u}, \theta) = \int_{t_0}^{t_f} \ell_p(\cdot)^T \cdot Q_\theta \cdot \ell_p(\cdot) + [\mathbf{g}(\cdot) - \tilde{\mathbf{y}}]^T \cdot Q_\theta \cdot \\ \left[D_x \mathbf{g}(\cdot) \cdot D_\theta^2 \mathbf{x}(\cdot) + D_x^2 \mathbf{g}(\cdot) \cdot \psi(\cdot) + D_\theta^2 \mathbf{g}(\cdot) \right] dt.$$

To calculate the exact Hessian, a second differential equation for $D_\theta^2 \mathbf{x}$ must be calculated; however, near the optimal parameter set, $\hat{\theta}$, $(\mathbf{g}(\cdot) - \tilde{\mathbf{y}}) \approx 0$. Therefore, assuming that the estimated parameters are near the optimal set, the Hessian is approximated as

$$\overline{D_p^2 J_p}(\mathbf{x}, \mathbf{u}, \theta) = \int_{t_0}^{t_f} \ell_p(\cdot)^T \cdot Q_\theta \cdot \ell_p(\cdot)\ dt. \tag{6}$$

Although the Iterated Least-Squares algorithm improves convergence rates near the optimal parameter set compared to the gradient descent method, for highly ill-conditioned problems, undesirable convergence rates may exist. Convergence results for a robot and suspended mass system will be presented in the following sections.

III. Conditioning Optimization Algorithm

To improve the convergence rates using either gradient descent or Iterated Least-Squares estimation methods, an initial experimental trajectory having a Hessian with a high condition number can be optimized to produce an experimental trajectory with better Hessian conditioning. The theoretical framework for the optimization routine is largely based on an infinite dimensional LQR optimization with a nonlinear projection step. Details on the theoretical background can be found in [23], [29].

A. Cost Function

The cost function, which is minimized by the trajectory optimization algorithm, directly incorporates the condition number of the parameter estimation Hessian, a control cost, and an optional cost on the trajectory itself. The trajectory optimization algorithm uses the current best estimate of the parameter set. Therefore, it is assumed that the parameter set is close to the true value; thus the Hessian is approximated by (6).

Using this formulation, the condition number of the Hessian can be calculated. Assuming that the Euclidean norm is used and the Hessian is symmetric, the condition number is equal to

$$\kappa(\overline{D_p^2 J_p}(\mathbf{x}, \mathbf{u}, \theta)) = \left| \frac{\lambda_{max}}{\lambda_{min}} \right|,$$

where λ is the set of eigenvalues of the matrix $D_\theta^2 J_p(\cdot)$.

With the definition of the condition number, the cost function for the trajectory optimization problem is given by

$$J_\tau = Q_p \left(\frac{\lambda_{max}}{\lambda_{min}} - 1 \right) + \frac{1}{2} \int_{t_0}^{t_f} \left[(\mathbf{x}(t) - \mathbf{x_d}(t))^T \cdot Q_\tau \cdot \right. \\ \left. (\mathbf{x}(t) - \mathbf{x_d}(t)) + \mathbf{u}(t)^T \cdot R_\tau \cdot \mathbf{u}(t) \right] dt \tag{7}$$

where Q_p is a scalar weight on the condition number minimization, Q_τ is a $n \times n$ weighting matrix on the trajectory tracking cost, and R_τ is a $r \times r$ weighing matrix on the control inputs.

Although the cost function appears to include a terminal condition on the eigenvalues of $\overline{D_\theta^2 J_p}(\cdot)$, since the least-squares estimator is defined over the entire time horizon, the eigenvalues themselves are functions of each point along the trajectory. This property will allow the apparent non-Bolza form of the optimal control problem to be cast into a Bolza form for the iterative LQR step described in the next section. This Bolza form of the problem only holds locally for perturbations to the trajectory required for the iterative LQR optimization.

The overall optimization problem can therefore be written as

$$\boxed{\begin{array}{c} \min_{\mathbf{u}}\ J_\tau \\ \\ s.t.\ (1) \text{ and } (4) \text{ are satisfied.} \end{array}}$$

B. Extended Dynamics Constraints

The trajectory optimization algorithm requires that the objective function be an explicit function of the system states [30]. However, given that the Hessian defined by (6) is a function of the differential equation given by (4), the cost also depends on $\psi(\mathbf{x}, \mathbf{u}, \theta)$.

Since the objective is to minimize a norm that includes the derivative of the states with respect to the parameters, $\psi(\cdot)$ from (4) must be appended as an additional state. Appending $\psi(\cdot)$ to the state vector as an additional dynamic constraint allows for variations in $\psi(\cdot)$ in the optimization algorithm. For convenience, $\bar{\mathbf{x}}(t) = (\mathbf{x}(t), \psi(\cdot))$ will define the extended states and $\eta(t) = (\bar{\mathbf{x}}(t), \mathbf{u}(t))$ defines a feasible curve for the dynamics of the extended state.

C. Optimization Routine

The optimal control problem is solved using an iterative descent technique shown in Algorithm 2. Since directly solving the constrained optimization problem is unlikely to be feasible for general nonlinear systems, iterative steps are computed which incrementally improve the cost until a minimizer is obtained. Additionally, a projection operator $P(\xi_i(t))$ is used so that the optimization can be reformulated as an unconstrained problem of the form

$$\arg\min_{\xi_i(t)} \quad J_\tau(P(\xi_i(t))).$$

This allows variations of the trajectory to be calculated free of the constraint of maintaining feasible dynamics; however, the solution is projected to a feasible trajectory at each iteration of the optimization algorithm.

For each iteration of the algorithm, a descent direction $\zeta_i(t) = (\bar{\mathbf{z}}, \mathbf{v})$ is computed from a time-varying linearization operating about each point in time of the current trajectory. The descent direction calculation can be cast as an unconstrained LQR problem [30] given by

$$\zeta_i(t) = \arg\min_{\zeta_i(t)} DJ_\tau(P(\xi_i(t))) \circ \zeta_i(t) + \frac{1}{2}\langle \zeta_i(t), \zeta_i(t)\rangle \quad (8)$$

such that

$$\dot{\bar{\mathbf{z}}} = A\bar{\mathbf{z}} + B\mathbf{v}$$

where $\zeta_i(t) \in T_{\eta_i}\mathcal{T}$, i.e., the descent direction lies in the tangent space of the trajectory manifold at the current iteration. The components of the descent direction, $\zeta_i = (\bar{\mathbf{z}}(t), \mathbf{v}(\mathbf{t}))$ are defined by $\bar{\mathbf{z}}(t)$, the perturbation to the extended state and $\mathbf{v}(t)$, the perturbation to the control. Matrices A and B are linearizations of the system dynamics, which are formulated in the following section. Since (8) is a quadratic function of ζ_i with linear constraints, the descent direction can be computed using LQR techniques described in detail in the following section.

Using the projection operator $P(\xi_i(t))$, the unconstrained, or infeasible solution $\xi_i(t)$ is projected onto the dynamics constraints at each iteration as detailed in [23]. The projection operator is a stabilizing feedback law used to map an infeasible or feasible trajectory, defined by $\xi(t) = (\bar{\alpha}(t), \mu(t))$ to a dynamically feasible trajectory, $\eta(t) = (\bar{\mathbf{x}}(t), \mathbf{u}(t))$.

The projection operator used in this paper is given by

$$P(\xi(t)) : \begin{cases} \mathbf{u}(t) = \mu(t) + K(t)(\bar{\alpha}(t) - \bar{\mathbf{x}}(t)) \\ \dot{x}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \\ \dot{\psi}(t) = D_x f(\mathbf{x}, \mathbf{u}, \theta)^T \psi(t) + D_\theta f(\mathbf{x}, \mathbf{u}, \theta)^T \end{cases}$$

where the domain of the operator is an unconstrained trajectory, $\xi(t) = (\bar{\alpha}(t), \mu(t))$ computed from the iterative step $\xi_i(t) = \eta_i(t) + \gamma_i\zeta_i$, and the output is a feasible trajectory, $\eta(t) = (\bar{\mathbf{x}}(t), \mathbf{u}(t))$. The feedback gain $K(t)$ can be optimized as well by solving an additional linear quadratic regulation problem. Details of the optimal gain problem can be found in [23], but any feedback synthesis technique may be used.

Algorithm 2 Trajectory Optimization

Initialize $\eta_0 \in \mathcal{T}$, tolerance ϵ
while $DJ_\tau(\eta_i(t)) \circ \zeta_i > \epsilon$ **do**
  Calculate descent, ζ_i:
    $\zeta_i = \arg\min_{\zeta_i(t)} DJ_\tau(P(\xi_i(t))) \circ \zeta_i + \frac{1}{2}\langle \zeta_i, \zeta_i\rangle$
  Compute γ_i with Armijo backtracking search
  Calculate infeasible trajectory:
    $\xi_i(t) = \eta_i(t) + \gamma_i\zeta_i$
  Project trajectory onto dynamics constraints:
    $\eta_{i+1}(t) = P(\xi_i(t))$
  $i = i + 1$
end while

Given the descent direction, ζ_i, a backtracking line-search of the projection, $P(\eta_i(t) + \gamma_i\zeta_i)$ provides a feasible trajectory solution assuming that the step size γ_i satisfies the Armijo sufficient decrease condition. This new feasible trajectory then becomes the trajectory for the next iteration of the optimal control algorithm.

D. Descent Direction Calculation

At each iteration of the conditioning optimization algorithm, the LQR problem given by (8) must be solved. The descent direction depends on the linearization of the cost function, $DJ(P(\xi_i(t)))$ and the local quadratic model, $\frac{1}{2}\langle \zeta_i(t), \zeta_i(t)\rangle$. By rewriting the linearization in terms of extended states and controls and evaluating the local quadratic model using a weighed norm on the states and controls, the equation for the descent direction (8) becomes

$$\arg\min_{\zeta_i(t)} = \int_{t_0}^{t_f} \mathbf{a}(t)^T\bar{\mathbf{z}}(t) + \mathbf{b}(t)^T\mathbf{v}(t) + \frac{1}{2}\bar{\mathbf{z}}(t)^T Q_n \bar{\mathbf{z}}(t) + \frac{1}{2}\mathbf{v}(\mathbf{t})^T R_n \mathbf{v}(\mathbf{t})\ dt, \quad (9)$$

such that

$$\dot{\bar{\mathbf{z}}} = A\bar{\mathbf{z}} + B\mathbf{v}$$

where $\mathbf{a}(t)$ and $\mathbf{b}(t)$ are the linearizations of the cost function with respect to $\bar{\mathbf{x}}$ and $\mathbf{u}$, A and B are the linearizations of the extended states, and Q_n and R_n are weighting matrices for the local quadratic model approximation. Design of these weighting matrices can lead to faster convergence of the optimal control algorithm depending on the specific problem. In the following subsections, the formulations of the linearizations will be presented.

1) Cost Function Linearization: The first term of (9) is computed as the linearization of the cost function with respect to the extended states, $\bar{\mathbf{x}}(t)$ and the controls, $\mathbf{u}(t)$. To begin, we apply the quotient rule to the cost function given in (7). Taking the first derivative results in the following equation for $\mathbf{a}(t)$:

$$\mathbf{a}(t) = \frac{\partial J_\tau}{\partial \bar{\mathbf{x}}} = Q_p \left(\frac{1}{\lambda_{min}} \frac{\delta \lambda_{max}}{\delta \bar{\mathbf{x}}} - \frac{\lambda_{max}}{\lambda_{min}^2} \frac{\delta \lambda_{min}}{\delta \bar{\mathbf{x}}} \right) + \int_{t_0}^{t_f} \left[(\mathbf{x}(t) - \mathbf{x}_d(t))^T \cdot Q_\tau \right] dt. \tag{10}$$

It is clear from this result that the derivative of the eigenvalues, $\frac{\delta \lambda_{max}}{\delta \bar{\mathbf{x}}}$ is needed. Fortunately, eigenvalue perturbation theory permits such differentiation, and the resulting form is relatively compact. From [31], the derivative of one eigenvalue of some matrix A is given by

$$D_i \lambda_k = y_k^T \cdot D_i A \cdot x_k, \tag{11}$$

where λ_k is the k^{th} eigenvalue of A, x_k is the associated left eigenvector, y_k is the associated right eigenvector of A, and $D_i A$ and $D_i \lambda_k$ are the partial derivatives of A and λ_k with respect to some argument i.

Given (11), the derivative of λ_i, for $i \in \{min, max\}$, with respect to $\bar{\mathbf{x}}$ is written as

$$\frac{\delta \lambda_i}{\delta \bar{\mathbf{x}}} = \int_{t_0}^{t_f} \mathbf{w}^{*T} \frac{\delta}{\delta \bar{\mathbf{x}}} \left[\ell_p(\cdot)^T \cdot Q_p \cdot \ell_p(\cdot) \right] \mathbf{w}^* \, dt, \tag{12}$$

where $\mathbf{w}^*$ is the eigenvector associated with λ_i.

Lastly, differentiating the terms of the Hessian yields,

$$\frac{\partial}{\partial \bar{x}} \left(\ell_p(\cdot)^T \cdot Q_p \cdot \ell_p(\cdot) \right) = \begin{bmatrix} 2\, \ell_p^T \cdot Q_p \cdot \left(D_x^2 \mathbf{g}(\cdot) \cdot \psi(\cdot) + D_x D_\theta \mathbf{g}(\cdot) \right) \\ 2\, \ell_p^T \cdot Q_p \cdot D_x \mathbf{g}(\cdot) \cdot E \end{bmatrix} \tag{13}$$

where E is a tensor of the form

$$E_{i,j,k,l} = \delta_{i,k} \delta_{j,l}$$

with $\delta_{\cdot,\cdot}$ as the Kronecker delta function.

Combining equations (10), (12), (13), results in the complete linearization $\mathbf{a}(t)$ in (9). The linearization with respect to the controls $\mathbf{u}(t)$ is simply given by

$$\mathbf{b}(t) = \mathbf{u}(t)^T \cdot R_\tau.$$

2) Dynamics Linearization: The two other quantities needed to compute the descent direction are $A(t)$ and $B(t)$ – the linearizations of the dynamics. The descent direction, ζ_i, will satisfy the linear constraint ODE given by

$$\dot{\bar{\mathbf{z}}}_i(t) = A(t) \bar{\mathbf{z}}_i(t) + B(t) \mathbf{v_i}(t),$$

where $A(t)$ is the linearization of the nonlinear dynamics given by (1) and (4) with respect to $\bar{\mathbf{x}}(t)$, and $B(t)$ is the linearization with respect to $\mathbf{u}(t)$. The linearization $A(t)$, of the dynamics with respect to the extended state $\bar{\mathbf{x}}(t)$, is given by

$$A(t) = \begin{bmatrix} \frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{x}} & \frac{\partial \dot{\mathbf{x}}}{\partial \psi} \\ \frac{\partial \dot{\psi}}{\partial \mathbf{x}} & \frac{\partial \dot{\psi}}{\partial \psi} \end{bmatrix} = \begin{bmatrix} D_x f(\cdot) & \{0\}^{n \times n \times p} \\ D_x^2 f(\cdot) \cdot \psi(t) + D_x D_\theta f(\cdot) & D_x f(\cdot) \cdot E \end{bmatrix}.$$

Additionally, the linearization of the dynamics with respect to the control input $\mathbf{u}(t)$, is required. This linearization matrix $B(t)$, is given by

$$B(t) = \begin{bmatrix} \frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{u}} \\ \frac{\partial \dot{\psi}}{\partial \mathbf{u}} \end{bmatrix} = \begin{bmatrix} D_u f(\cdot) \\ D_u D_x f(\cdot) \cdot \psi(t) + D_u D_\theta f(\cdot) \end{bmatrix}.$$

We now have all the data that (9) requires to compute a descent direction $\zeta_i(t)$. Since this descent direction is based on the linearized dynamics, the projection operator must be applied to $\eta_i(t) + \gamma_i \zeta_i$ to ensure the dynamics constraints are satisfied. This process is iteratively repeated until convergence is achieved as shown in Algorithm 2.

IV. Experimental Setup

To illustrate the use of conditioning optimization on a non-linear, dynamic system, a simulation and experimental test of a two dimensional robot and suspended mass system is analyzed. The system has two configuration variables, $q = (x(t), \phi(t))$, where x is the horizontal displacement of the cart and ϕ is the rotational angle of the link as seen in Fig. 1. For this example, the robot's horizontal acceleration will be directly controlled, therefore, $u(t) = \ddot{x}(t)$. A load cell provides the magnitude of string tension force as the sole output of the system.

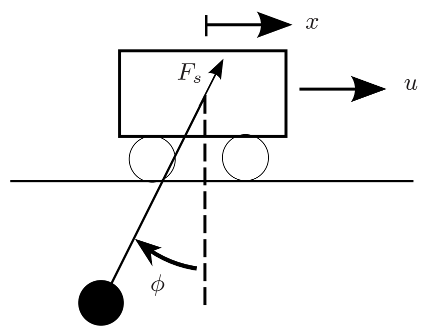


Fig. 1. Diagram of the robot and suspended mass system.

A. System Model

The equations of motion for the system are given by the following

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ u \\ \dot{\phi} \\ \frac{u}{\ell} \cos\phi - \frac{g}{\ell} \sin\phi \end{bmatrix}$$

where u is the cart acceleration control input, m is the mass suspended from the robot, and ℓ is the length of the string. Additionally, the equation for the force output F_s is

$$\mathbf{y} = F_s = mg\cos\phi - m\ell\dot{\phi}^2 - u\sin\phi. \tag{14}$$

It is assumed that the trajectories will maintain tension in the string; therefore, a fixed distance between the robot and mass can be assumed.

The goal of the experiment is to estimate the value of the mass attached to the string and the length of the string, $\theta = \{m, \ell\}$. The choice of force as the system output highlights the effect of the trajectory on the conditioning of the estimation problem. Static or nearly static trajectories, such as the initial trajectory shown in Fig. 3a, provide valid estimates of the mass value; however, the string length cannot be resolved. On the other hand, dynamic trajectories provide fluctuating force measurements which can be used in conjunction with the predicted model to simultaneously estimate the length and mass values.

B. Experimental Testbed Setup

The experimental setup shown in Fig. 2 consists of a differential drive mobile robot with magnetic wheels moving in a plane. The robot's driving surface is provided by a smooth steel plate mounted above and parallel to the ground. Two 24V DC motors drive the robot's magnetic wheels. PID loops running at 1500 Hz close the loop around motor velocity using optical encoders for feedback. Two 12V lithium iron phosphate batteries provide the required power. Desired velocity commands are sent to the robot wirelessly using Digi XBee® modules at 50 Hz, and a 32-bit Microchip PIC microprocessor handles all on-board processing, motor control, and communication.

The motors are powerful relative to the inertias of the robot, the suspended mass, and the wheels. Thus they are capable of accurately tracking aggressive trajectories up to a maximum rotational velocity. Since the input to the system is the direct acceleration of the cart, PID loops on the motor velocity allow the robot to accurately track a given velocity profile. A load cell is attached to the robot and the mass is suspended from its measurement point to enable collection of force data using an instrumentation amplifier and 10-bit, on-board ADC. The load cell output has been fit to a linear model using six known masses ranging from 0.05kg to 0.30kg. The Robot Operating System (ROS) is used for interpreting and transmitting desired trajectories and for collecting and processing all of the experimental data [32].

C. Identification Procedure

The procedure for completing the optimizations and taking measurement data from the experimental system is performed in the following order.

1) Perform the conditioning optimization using the current best estimate of the parameters which yields a new set of control inputs for the system.
2) Run the experiment using the newly generated inputs and take measurement of the output along the trajectory.
3) Perform the least-squares parameter optimization using the optimized trajectory to determine a new estimate of the parameter set.

V. Results

A. Overview

The following section presents the results of the optimization study in both simulation and experimental trials. Given an initial trajectory and set of unknown model parameters, the conditioning algorithm is used to synthesize an optimized trajectory. The key results are as follows:

- The condition number of the parameter optimization Hessian decreases from 1.73×10^8 using measurements from the initial trajectory to 19.3 using optimized trajectory measurements.
- In simulation, 27 gradient descent steps are needed to converge to the actual parameter values using optimized trajectory measurements compared to over 1000 using the initial trajectory.
- Using the experimental system, the error in predicted string length decreases to 2.4% using optimized trajectory measurements compared to 122.% using the initial trajectory data.

Detailed results and analysis of the data is provided in the sections that follow.

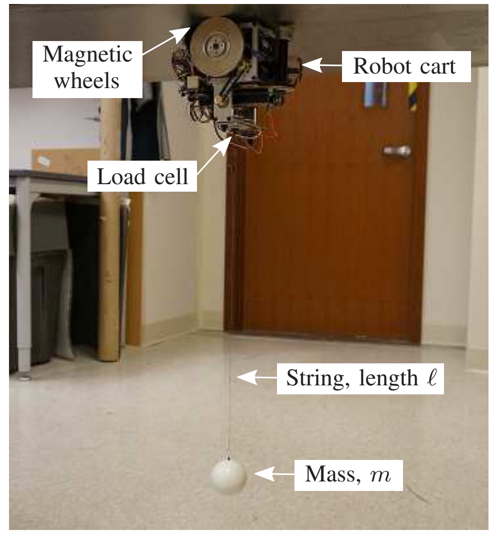


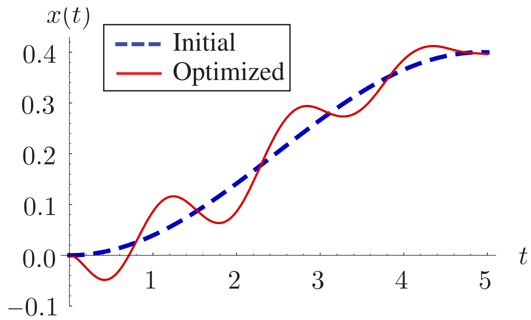
Fig. 2. Image showing the robot and suspended mass experimental system with a diagram of main components.

B. Optimization Results

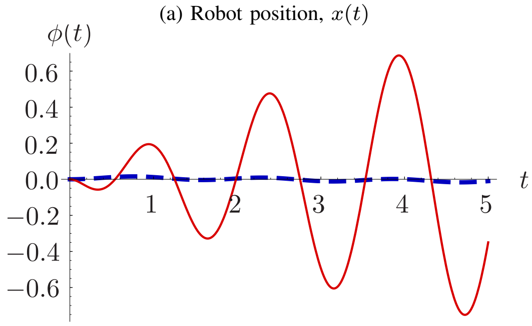
The two parameters that will be estimated are suspended mass value and the string length. An initial, and intentionally incorrect, estimate of the parameters was chosen for both the gradient descent and the Iterated Least-Squares estimation algorithms. The actual values of the parameters were independently measured to provide a benchmark for algorithm verification. These values are as follows,

$$\text{Initial estimate}: m = 0.10 \text{ kg}, \quad \ell = 0.60 \text{ m}$$
$$\text{Actual values}: m = 0.12 \text{ kg}, \quad \ell = 0.50 \text{ m}$$

A minimal control trajectory is chosen as the initialization for the conditioning optimization algorithm as shown in Fig. 3. Using the initial estimate of the parameter set and initial trajectory, the optimization algorithm was run until a convergence criterion of $(J(\eta_{i+1}(t)) - J(\eta_i(t))) < 10^{-1}$ was satisfied. The comparison of initial and optimized trajectories can be seen in Fig. 3.



(a) Robot position, $x(t)$



(b) Angle of suspended mass with respect to the robot, $\phi(t)$

Fig. 3. Plots showing the robot (a) and suspended mass (b) trajectory before and after conditioning optimization.

TABLE I
OPTIMIZATION RESULTS

	Initial	Optimal
First Eigenvalue, λ_1:	3.18×10^6	3.63×10^6
Second Eigenvalue, λ_2:	1.84×10^{-2}	1.88×10^5
Condition Number, κ:	1.73×10^8	1.93×10^1
Optimization Cost, J_τ:	8.63×10^4	1.02×10^2

TABLE II
NUMBER OF ITERATIONS TO CONVERGE TO $|J_p| < 10^{-6}$

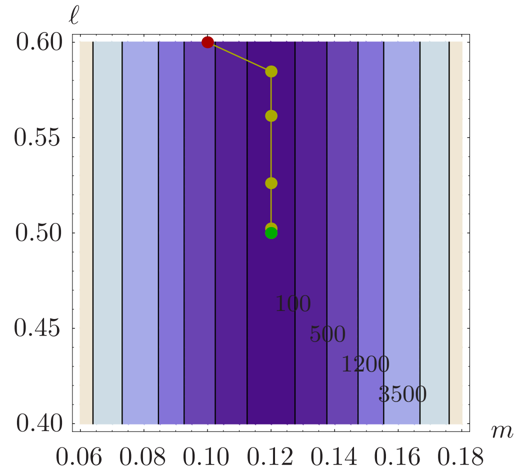
	Initial	Optimal
Gradient Descent:	> 1000*	27
ILS:	5	3

*Estimation was stopped after 1000 iterations with no significant improvement in J_p

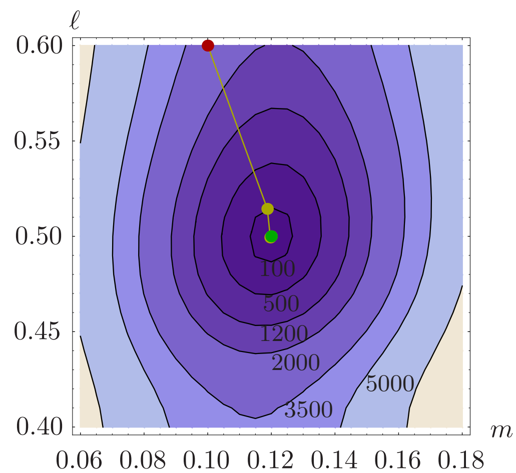
The eigenvalues of $\overline{D_p^2 J_p}$, the condition number κ, and the cost J_τ for the initial trajectories and the optimized trajectories are listed in Table I. The results show that the condition number, κ, decreases drastically from 1.73×10^8 to 19.3. Thus a significant improvement in the convergence rate of the parameter estimation algorithms, especially the gradient descent method, can be expected.

The plots of the optimized trajectory show that the oscillation of the suspended mass improved the simultaneous estimation conditioning involving the suspended mass and length parameters. While the load cell can correctly estimate the mass given a static or near static system, the length can only be estimated given a dynamic motion.

In simulation, the gradient descent and Iterated Least-Squares parameter estimation algorithms were executed using the initial and optimal trajectories. The convergence results appear in Table II. As the results indicate, second-order information in the Iterated Least-Squares method greatly improves convergence over the gradient descent method, even for the initial, ill-conditioned problem. The benefits of conditioning optimization are evident in the significantly improved convergence of the gradient descent algorithm. The optimal trajectory does provide a convergence improvement for the Iterated Least-Squares method; however, it is less significant.



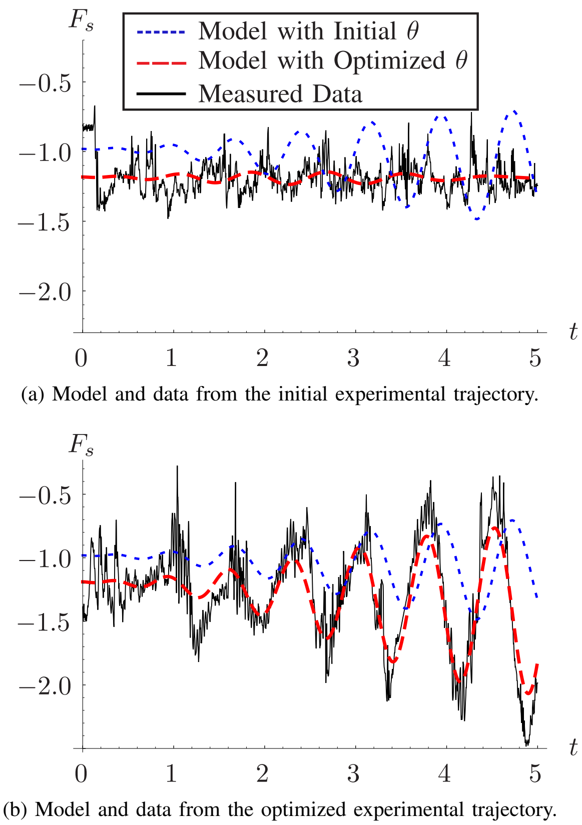
(a) Contour plot showing convergence in simulation using the initial trajectory.



(b) Contour plot showing convergence in simulation using the optimized trajectory.

Fig. 4. Contour plots showing the parameter estimation cost J_p across the parametric space with the convergence path shown by the colored dots. The red dot indicates the initial parameter estimate, the yellow dots show intermediate iteration points, and the green dot indicates the converged point.

The convergence path of the Iterated Least-Squares algorithm is shown on contour plots of the parametric space in Fig. 4. In the initial trajectory, the lack of information about the length parameter leads to the ill-conditioning of the problem. The algorithm quickly converges to the correct mass yet takes many extra steps to converge to both parameters. After optimization, the optimal trajectory yields improved conditioning of the cost basin which allows the algorithm to more directly converge on both parameters simultaneously.



(a) Model and data from the initial experimental trajectory.

(b) Model and data from the optimized experimental trajectory.

Fig. 5. Plots showing measured force data collected from the experimental trial and predicted force data using both initial and optimized parameter values.

In simulations with no measurement or model uncertainty, the optimal trajectory only provides slight improvement in the Iterated Least-Squares convergence rate; however, in practice, use on experimental systems introduces measurement noise and model uncertainty which greatly impacts the estimates using the two trajectories. The following section provides experimental results illustrating this point.

C. Experimental Results

Using the same initial and optimized control inputs shown in Fig. 3, each trajectory was run on the experimental platform described in Sec. IV-B. During each trial, force data was collected at 300Hz and interpolated to provide continuous data for the estimation algorithm. Following the data collection, the Iterated Least-Squares algorithm was run on each set of data to attempt to find a best estimate of the parameter set using the measured output.

The results of the experiment are shown in Fig. 5 and Table III. The impact of an ill-conditioned problem is accentuated in the experimental results. While both trajectories provide estimates that are within 2.4% of the mass value, the length parameter estimate from the initial trajectory has a 122% error compared to only 2.0% with the optimized trajectory.

Fig. 5 shows the predicted force output using the initial parameter estimate with the blue dotted line before running the Iterated Least-Squares estimation algorithm. After running the estimation algorithm on both trajectories, the model predicted

TABLE III
EXPERIMENTAL RESULTS

	m (kg)	ℓ (m)
Initial Estimate:	0.100	0.60
Measured Baseline Value:	0.124	0.50
Estimate from Initial Trajectory:	0.121	1.11
% Error from Baseline Value:	2.4	122.
Optimal Trajectory Estimate:	0.121	0.51
% Error from Baseline Value:	2.4	2.0

force output for the optimized parameters is shown by the red dashed line. The force data collected by the load cell on the robot platform is shown by the solid black line.

Examining this data, the initial trajectory results in small angle displacements of the mass. Since the displacement is small compared to the noise, the model is fit mostly to noise, providing an inaccurate estimate of the length as seen in Table III. As the mass is purely a function of the mean of the signal, the signal from both trajectories is sufficient to provide a close estimate of the suspended mass. Synthesizing a new trajectory with better conditioning results in oscillations of the mass, which provide greater sensitivity in the measurement output with respect to the length parameter, resulting in better estimate convergence.

VI. Conclusion

It is important to consider the design of the trajectory when attempting to estimate model parameters in a nonlinear dynamic system. Using optimization as a means of synthesizing these trajectories results in better conditioning of least-squares parameter estimation, as shown in the experimental results. In the case of gradient descent approaches, orders of magnitude reduction in the number of iterations can be achieved by conditioning the Hessian of the estimation problem. While second-order techniques show modest improvement in theory, in practice, measurement noise and unbiased uncertainties highlight the improvement in the estimate convergence.

Using the condition number cost balanced with appropriate weights of control effort and tracking error results in an improved trajectory while allowing users to assign importance to each component. Different scenarios will require varying levels of constraint on control effort and tracking error while attempting to estimate system parameters.

While least-squares based estimation methods are relatively robust to unbiased noise such as Gaussian measurement noise, sources of biased error such as unmodeled dynamics may be problematic for arbitrarily chosen experimental trajectories. Future work into creating more robust algorithms may allow for an expanded class of noise and error to be included in theoretical error and convergence bounds. Additionally, there is great potential for further approximations and dimensionality reduction of the extended state dynamics, ψ, which may allow for faster and more efficient synthesis of the experimental trajectory.

Acknowledgment

This material is based upon work supported by the United States Department of Defense through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program and by the National Science Foundation under Grant CMMI 1334609. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. John Wiley & Sons, 2004.

[2] L. Ljung, *System Identification: Theory for the User (2nd Edition)*. Prentice Hall, 1999.

[3] K. Hsu, K. Poolla, and T. L. Vincent, "Identification of Structured Nonlinear Systems," *IEEE Transactions on Automatic Control*, vol. 53, pp. 2497–2513, Dec. 2008.

[4] S. K. Pradhan and B. Subudhi, "Real-Time Adaptive Control of a Flexible Manipulator Using Reinforcement Learning," *IEEE Transactions on Automation Science and Engineering*, vol. 9, pp. 237–249, Apr. 2012.

[5] H. Hjalmarsson and J. Martensson, "Optimal Input Design for Identification of Non-linear Systems: Learning From the Linear Case," in *2007 American Control Conference*, pp. 1572–1576, July 2007.

[6] H. Jansson and H. Hjalmarsson, "Input design via LMIs admitting frequency-wise model specifications in confidence regions," *IEEE Transactions on Automatic Control*, vol. 50, pp. 1534–1549, Oct. 2005.

[7] M. Casini, A. Garulli, and A. Vicino, "Time complexity and input design in worst-case identification using binary sensors," in *2007 46th IEEE Conference on Decision and Control*, pp. 5528–5533, 2007.

[8] H. Kim, B. Park, J. Lee, Y. J. Park, and W. K. Chung, "A parameter estimation method for the bilateral teleoperation framework for an O2 lance manipulator," in *2012 IEEE International Conference on Automation Science and Engineering*, pp. 564–568, Aug. 2012.

[9] B. Armstrong, "On Finding Exciting Trajectories for Identification Experiments Involving Systems with Nonlinear Dynamics," *The International Journal of Robotics Research*, vol. 8, pp. 28–48, Dec. 1989.

[10] M. Gautier and W. Khalil, "Direct calculation of minimum set of inertial parameters of serial robots," *IEEE Transactions on Robotics and Automation*, vol. 6, pp. 368–373, June 1990.

[11] J. Swevers, C. Ganseman, D. Tukel, J. de Schutter, and H. Van Brussel, "Optimal robot excitation and identification," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 5, pp. 730–740, 1997.

[12] R. Mehra, "Optimal input signals for parameter estimation in dynamic systems–Survey and new results," *IEEE Transactions on Automatic Control*, vol. 19, pp. 753–768, Dec. 1974.

[13] T. L. Vincent, C. Novara, K. Hsu, and K. Poolla, "Input design for structured nonlinear system identification," *Automatica*, vol. 46, pp. 990–998, June 2010.

[14] K. Fang and A. T. Shenton, "Constrained Optimal Test Signal Design for Improved Prediction Error," *IEEE Transactions on Automation Science and Engineering*, vol. PP, no. 99, pp. 1–12, 2013.

[15] M. Baltes, R. Schneider, C. Sturm, and M. Reuss, "Optimal experimental design for parameter estimation in unstructured growth models," *Biotechnology Progress*, vol. 10, pp. 480–488, Sept. 1994.

[16] D. Faller, U. Klingmuller, and J. Timmer, "Simulation Methods for Optimal Experimental Design in Systems Biology," *SIMULATION*, vol. 79, pp. 717–725, Dec. 2003.

[17] C. Jauberthie, L. Denis-Vidal, P. Coton, and G. Joly-Blanchard, "An optimal input design procedure," *Automatica*, vol. 42, no. 5, pp. 881–884, 2006.

[18] W. Wu, S. Zhu, X. Wang, and H. Liu, "Closed-loop Dynamic Parameter Identification of Robot Manipulators Using Modified Fourier Series," *International Journal of Advanced Robotic Systems*, p. 1, May 2012.

[19] W. Rackl, R. Lampariello, and G. Hirzinger, "Robot excitation trajectories for dynamic parameter estimation using optimized B-splines," in *2012 IEEE International Conference on Robotics and Automation*, pp. 2042–2047, May 2012.

[20] K. Radkhah, D. Kulic, and E. Croft, "Dynamic parameter identification for the CRS A460 robot," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3842–3847, Oct. 2007.

[21] J. Mårtensson, C. R. Rojas, and H. Hjalmarsson, "Conditions when minimum variance control is the optimal experiment for identifying a minimum variance controller," *Automatica*, vol. 47, pp. 578–583, Mar. 2011.

[22] A. D. Wilson and T. D. Murphey, "Optimal trajectory design for well-conditioned parameter estimation," in *2013 IEEE International Conference on Automation Science and Engineering*, pp. 13–19, Aug. 2013.

[23] J. Hauser, "A projection operator approach to the optimization of trajectory functionals," in *World Congress*, vol. 15, p. 310, July 2002.

[24] D. B. Özyurt and P. I. Barton, "Cheap Second Order Directional Derivatives of Stiff ODE Embedded Functionals," *SIAM Journal on Scientific Computing*, vol. 26, pp. 1725–1743, Jan. 2005.

[25] L. M. Miller and T. D. Murphey, "Simultaneous optimal estimation of mode transition times and parameters applied to simple traction models," *IEEE Transactions on Robotics*, vol. 29, pp. 1496–1503, Dec 2013.

[26] R. E. Kalman and R. S. Bucy, "New Results in Linear Filtering and Prediction Theory," *Journal of Basic Engineering*, vol. 83, no. 1, p. 95, 1961.

[27] L. Armijo, "Minimization of functions having Lipschitz continuous first partial derivatives.," *Pacific Journal of Mathematics*, vol. 16, no. 1, pp. 1–3, 1966.

[28] J. Nocedal and S. Wright, *Numerical Optimization*. Springer, 2006.

[29] B. D. O. Anderson and J. B. Moore, *Optimal control: linear quadratic methods*. Englewood Cliffs, NJ: Prentice-Hall, Jan. 1990.

[30] J. Hauser and D. Meyer, "The trajectory manifold of a nonlinear control system," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171)*, vol. 1, pp. 1034–1039, 1998.

[31] R. B. Nelson, "Simplified calculation of eigenvector derivatives," *AIAA Journal*, vol. 14, pp. 1201–1205, Sept. 1976.

[32] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA workshop on open source software*, vol. 3, 2009.

Andrew D. Wilson received the B.S. degree in mechanical engineering from Penn State University, University Park, PA, USA, in 2010. He is currently a Ph.D. candidate in mechanical engineering with the Neuroscience and Robotics Laboratory at Northwestern University, Evanston, IL, USA.

In 2010, he received the National Defense Science and Engineering Graduate Fellowship (NDSEG) from the U.S. Department of Defense. His research interests include modeling and control of dynamic systems using informative learning methods.

Jarvis A. Schultz received both a B.S. and M.S. degree in mechanical engineering from the University of Wyoming. As a Ph.D. candidate in mechanical engineering working in the Neuroscience and Robotics Lab at Northwestern University, his research interests are focused on embedded control for complex nonlinear systems.

Todd D. Murphey received his B.S. degree in mathematics from the University of Arizona and the Ph.D. degree in Control and Dynamical Systems from the California Institute of Technology. He is an Associate Professor of Mechanical Engineering at Northwestern University. His laboratory is part of the Neuroscience and Robotics Laboratory, and his research interests include robotics, control, computational methods for biomechanical systems, and computational neuroscience. Honors include the National Science Foundation CAREER award in 2006, membership in the 2014-2015 DARPA/IDA Defense Science Study Group, and Northwestern's Professorship of Teaching Excellence. He is a Senior Editor of the IEEE Transactions on Robotics.