JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 29, No. 4, July–August 2006

Real-Time Dynamic Optimization of Controllable Linear Systems

Martin Guay*
Queen's University, Kingston, Ontario K7L 3N6, Canada
and
J. Fraser Forbes†
University of Alberta, Edmonton, Alberta, T6G 2G6 Canada

A real-time optimization controller is developed to steer a linear time-varying control system to closed-loop trajectories that optimize a cost functional of interest. When advantage is taken of the differential flatness of the linear systems, a basis function approach is used to parameterize the open-loop trajectories of the system and to approximate the optimal solution of the finite time optimal control problem. An adaptive optimization method is used to formulate the real-time optimization scheme. The problem is posed as a real-time optimal trajectory generation problem in which the approximate optimal trajectories are computed using an extremum-seeking approach. The control algorithm provides tracking of the approximate optimal trajectories. Two optimal control problems are considered to demonstrate the application of the technique. It is shown that the technique can be successfully implemented in real time.

Introduction

DYNAMIC optimization is used to solve a wide range of offline and online problems. Linear time-varying systems constitute an important class of dynamic systems that arise naturally when tracking problems are considered. In a number of applications, it is desirable to design control systems that evolve on trajectories that provide optimal performance. There are several ways to solve such optimal control problems.

The most widely applied technique remains the linear quadratic optimal control problem. For linear time-varying systems, the linear quadratic optimal control problem can be addressed effectively by solving the Riccati differential equation. (See Ref. 1, Chap. 3, p. 124 for more details.)

For the solution of more general dynamic programming problems, numerical methods are usually required. Some suitable numerical methods include 1) the solution of the necessary conditions for optimality,[2] 2) dynamic programming,[3] 3) discretization,[4] 4) control parametrization.[5] These established numerical approaches have met with success. However, they often require considerable computational effort, which limits their implementation in real-time applications, for example, online trajectory planning or receding horizon estimation and control.

Some results have been published on control of differentially flat systems,[6,7] and a number of flatness-based dynamic optimization methods have been reported.[8–12] Differential flatness is a property of a special class of underdetermined dynamic control systems whose dynamics can be freely assigned without differential constraints. For such systems, all system trajectories can be trivially assigned by defining paths in the so-called flat outputs. The main advantage of flatness-based approaches is the elimination of numerical integration of the model in the solution of optimal control problems. This property has led to the development of efficient algorithms.[10,11]

Xu and Agrawal[13] suggested a procedure for optimization of linear time-varying dynamic systems by partitioning the time domain into intervals such that a transformation exists over each interval, but conditions for the existence of the transformation are not provided. Furthermore, like the conventional solution of this problem using Lagrange multipliers (see Ref. 2, pp. 42–168.), general algebraic constraints were not considered in their work.

Mahadevan et al.[11] proposed a flatness-based model predictive technique for fed-batch reactors. A real-time optimization scheme is proposed that implements a model-predictive approach where the nonlinear dynamic programming problem is solved at prespecified time intervals. At each subsequent interval, the dynamic programming problem is formulated and updated using the current state measurements and considering the optimization problem from the current time only. As in other flatness-based approaches, the dynamic optimization problem is transformed to a nonlinear optimization problem and solved using an appropriate nonlinear optimization problem code. The resulting sequential approach allows one to respond in real-time to the effect of uncertainties and disturbances.

In this paper, we develop a real-time optimization approach to solve dynamic optimization problems in finite dimensional linear time-varying systems. We consider the application to the solution of real-time dynamic optimization problems in differentially flat linear time-varying systems. The approach proposed provides a real-time optimization technique that can be used to compute and implement simultaneously optimal trajectories. The optimization technique is inspired by a Lyapunov-based optimization method proposed in Ref. 14, but applies, more generally, to dynamic optimization problems where one seeks optimal trajectories rather than optimal set points. As in Ref. 11, we exploit the knowledge of the model structure to parameterize the set of admissible trajectories to maximize the prescribed cost functional over a finite dimensional set of parameters. The main difference with existing approaches is that the parameters used in the approximation of the optimal solution are updated in real time. Standard nonlinear adaptive control techniques are used to develop a real-time optimization routine that operates exactly like an adaptive controller. A Lyapunov-based technique is employed that ensures the convergence of the real-time optimization scheme to a local optimum. This allows one to solve the problem as the process is evolving in time without the intervention of specialized optimization software. This contrasts with other approaches, such as the one proposed in Ref. 11, where the real-time optimization relies on the solution of a nonlinear optimization problem using some optimization software. The time required to solve the optimization problem is generally not taken into account in the problem formulation. Because the solution of such problems cannot be instantaneous, some input delay must be considered in its implementation. This input delay limits the applicability of the techniques to relatively slow systems.

Received 31 January 2005; revision received 4 October 2005; accepted for publication 18 October 2005. Copyright © 2006 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0731-5090/06 $10.00 in correspondence with the CCC.

*Associate Professor, Department of Chemical Engineering; guaym@chee.queensu.ca.

†Professor, Department of Materials and Chemical Engineering; Fraser.Forbes@ualberta.ca.

The paper is organized as follows. In the second section, we state and formulate the dynamic optimization problem. The optimization technique and the implementation of the control is discussed in the third section. Simulation results are presented in the fourth section, which is followed by brief conclusions in the fifth section.

Problem Statement

In this paper, we consider a controllable linear time-varying system of the form

$$\dot{x} = A(t)x + B(t)u(t) \tag{1}$$

where $A(t) \in \mathbb{R}^{n \times n}$ and $B(t) \in \mathbb{R}^{n \times p}$ are C^∞ matrix-valued function of time t, $x \in \mathbb{R}^n$ are the state variables and $u(t) = [u_1(t), \ldots, u_p(t)]^T \in \mathbb{R}^p$ is the vector of p input variables.

The control design objective is to steer the states of the linear system (1) to the trajectory that solves the following dynamic optimization problem:

$$\max_{u(t)} J = \int_0^T q[x(t), u(t)]\,\mathrm{d}t \tag{2}$$

subject to

$$\begin{aligned} \dot{x} &= A(t)x + B(t)u(t) \\ w[x(t), \boldsymbol{u}(t)] &\geq 0 \\ x(0) &= x_0 \\ x(T) &= x_f \end{aligned} \tag{3}$$

The variable T, assumed fixed, is taken as the length of the horizon considered for the cost functional. It is assumed that there exists a continuous control $u(t)$ that can steer the state variables of the control system from x_0 to x_f over the interval $t \in [0, T]$.

The constraint set $\Omega_c = \{x \in \mathbb{R}^n, u \in \mathbb{R}^p | w(x, u) \geq 0\}$ describes a convex subset of $\mathbb{R}^n \times \mathbb{R}^p$. It is assumed that the trajectories $x(\tau)$ evolve on a compact subset Ω of $\mathbb{R}^n$ and that the input trajectories take values in a convex set $\Omega_u \in \mathbb{R}^p$. The cost function $J : \Omega_p \to \mathbb{R}^+$ is assumed to be a convex and twice continuously differentiable function on ω_p. The function $q(x, u)$ is assumed to be positive definite and sufficiently smooth.

To solve this problem, we must consider some parametrization of the trajectories of the system (1) over the set Ω. In this paper, we focus on the situation where the model (1) is differentially flat. The approach consists of computing the optimal trajectories in the flat output space. The corresponding trajectories, computed in real time, are implemented by using a suitable tracking controller. This method leads to a two-degree-of-freedom real-time optimizing controller. In the next two subsections, we provide a brief introduction to the notion of flatness and the application of flatness in dynamic optimization.

Controllable Linear Systems and Flat Dynamical Systems

Differential flatness, a notion introduced in Ref. 15, refers to the existence of so-called flat or linearizing outputs that summarize the dynamics of a nonlinear system. It is closely related to the general ability to linearize a dynamic system by an appropriate choice of endogenous dynamic state feedback. Such feedback structures describe a special class of static or dynamic precompensators formed only of endogenous system variables, such as state variables, inputs, and a finite number of their time derivatives.

Formally, the system (1) is said to be differentially flat if there exist variables $y = [y_1, \ldots, y_p]^T$ given by an equation of the form

$$y = \boldsymbol{h}\big(x, u, \dot{u}, \ldots, u^{(\rho)}\big) \tag{4}$$

The variables $y = [y_1, \ldots, y_p]^T$ are referred to as the flat outputs.

The original system variables $x(t)$ and $u(t)$ are written as functions of these flat outputs [$y(t)$ or $y(\tau)$] and a finite number of their derivatives,

$$\begin{aligned} x(t) &= \alpha\big[y(t), y^{(1)}(t), \ldots, y^{(k)}(t)\big] \equiv \alpha[\bar{y}(t)] \\ u(t) &= \beta\big[y(t), y^{(1)}(t), \ldots, y^{(k)}(t)\big] \equiv \beta[\bar{y}(t)] \end{aligned} \tag{5}$$

for the case of differentially flat systems. Here, $y^{(i)}(t)$ stands for the ith derivative of $y(t)$ with respect to t, and k is the number of derivatives of $y(t)$ required to represent the system in the form given in Eq. (5). In addition, $\bar{y}(t)$ is a vector of derivatives of the flat output, that is,

$$\bar{y}(t) = \big[y(t), y^{(1)}(t), \ldots, y^{(k)}(t)\big]^T$$

For a linear time-varying system, the flatness property is equivalent to the controllability of the system. For the sake of completeness, we state this property formally in the following theorem, which can be found in Ref. 13.

Theorem: The linear time-varying system described by the matrices $[A(t), B(t)]$ is differentially flat if and only if it is controllable.

This results can be proven in a number of ways using either algebraic or geometric arguments.

Trajectory Generation by Dynamic Optimization

When flatness is used, the set of trajectories can be parameterized by simply choosing a suitable parametrization for the flat outputs. The resulting state and input trajectories can be computed directly from Eq. (5). This strategy has been employed in many studies. (See Refs. 9, 11, 12, 16–18, and references therein). The choice of parametrization depends entirely on the specific application. For example, one may enforce periodicity by parameterizing the flat output trajectories using Fourier series. In general, we parameterize the highest derivative of the flat outputs $y^k(t)$ as

$$y_j^{(k)}(t) = \sum_{i=1}^{N} \theta_i\, \Xi_{ij}(t), \qquad 1 \leq j \leq p \tag{6}$$

where $\theta = [\theta_1, \ldots, \theta_N]^T$ are the parameters to be assigned, and $\Xi_{ij}(t)$, $1 \leq i \leq N$ and $1 \leq j \leq p$ are the basis functions. The flat outputs and their first $k - 1$ derivatives are obtained by integrating Eq. (6) successively. The flat outputs are given by

$$y_j(t) = \sum_{i=1}^{N} \theta_i \int \cdots \int_k \Xi_{ij}(t)$$

for $1 \leq j \leq k$, or, in vector form,

$$y(t) = \theta^T \int \cdots \int_k \Xi(t)$$

where $\Xi(t)$ is the N by p matrix of basis functions.

The vector of derivatives of the flat outputs $\bar{y}(t)$ is written as

$$\bar{y}(t) = \theta^T \Phi(t)$$

where $\Phi(t) = [\int \cdots \int_k \Xi(t), \int \cdots \int_{k-1} \Xi(t), \ldots, \Xi(t)]$. When this parametrization is used, the cost function (2) is given by

$$J(\theta) = \frac{1}{T} \int_t^{t+T} q\{\alpha[\theta^T \Phi(\tau)], \quad \beta[\theta^T \Phi(\tau)]\}\,\mathrm{d}\tau \tag{7}$$

The constraints are reparameterized as follows:

$$w(\alpha(\theta(t)^T \Phi(t), \quad \beta(\theta(t)^T \Phi(t))) \geq 0 \tag{8}$$

We can then restate the parameterized form of the dynamic optimization problem as

$$\max_{\theta} J = \frac{1}{T} \int_t^{t+T} q\{\alpha[\theta^T \Phi(\tau)], \quad \beta[\theta^T \Phi(\tau)]\}\,\mathrm{d}\tau \tag{9}$$

subject to

$$w\{\alpha[\theta^T \Phi(\tau)], \quad \beta[\theta^T \Phi(\tau)]\} \geq 0$$

$$\alpha[\theta^T \Phi(0)] = x_0 \tag{10}$$

$$\alpha[\theta^T \Phi(T)] = x_f \tag{11}$$

Extremum-Seeking Dynamic Optimization

The objective of the controller design methodology is to steer the system to the optimal state and input trajectories, as functions of the parameters θ, that solve the optimization problem (9). In the remainder, we consider the cost functional (7) as a function of θ and $J(\theta)$.

We propose to solve the optimization problem (9) using an interior point method with penalty functions to incorporate the constraints and the boundary conditions as proposed in Ref. 19 (Chap. 5, p. 72). The constraints described by the vector-valued inequality (8) are enforced by an interior point method using a log-barrier function that is incorporated in the cost. (See Ref. 20, Chap. 17, p. 490 for further details.) The boundary conditions (10) and (11) are incorporated using a penalty function to the cost. This leads to a modified cost function given by

$$J_{ip} = \int_0^T \left(q(\alpha(\theta^T\Phi(\tau)), \beta(\theta^T\Phi(\tau))) - \sum_{i=1}^{m} \mu_i \log\left(w_i(\alpha(\theta^T\Phi(\tau)), \beta(\theta^T\Phi(\tau))) - \epsilon_i\right)\right) \mathrm{d}\tau + M\left(\alpha(\theta^T\Phi(0)) - x_0\right)^2 + M\left(\alpha(\theta^T\Phi(T)) - x_f\right)^2 \tag{12}$$

where $\mu_i > 0$ and $\epsilon_i > 0$ for $i = 1, \ldots, m$ and $M > 0$ are positive constants that are the tuning parameters of the modified cost functions. In general, μ_i and ϵ_i are taken as small as possible, and M is taken as large as possible. We first make the following assumption.

Assumption: The constraint set described by Eq. (3), which is convex of the set $\Omega \subset \Re^n$, remains convex over a set Υ in the parameter space in its parameterized form Eq. (8).

The assumption guarantees that the unconstrained optimization of the modified cost J_{ip} leads to the constrained optimum of $J(\theta)$ as the tuning constants $\mu_i \to 0$. Although this assumption can be restrictive in practice, most applications can be adequately solved using the proposed technique through a suitable a priori analysis of the problem. One technique, proposed in this paper, is to solve the optimization problem using an update law that constrains the parameters to remain in a convex set.

Real-Time Optimization Technique

The basic approach is to formulate the optimization of $J(\theta)$ using a Lyapunov-based approach. Given that the functional is convex with respect to θ over a prescribed region Υ, we can rely on the first-order conditions for optimality given by

$$\nabla_\theta J_{ip}(\theta^*) = 0 \tag{13}$$

where $\nabla_\theta J_{ip}(\theta^*)$ is the gradient of J_{ip} with respect to θ evaluated at the minimizer θ^*. As in Ref. 14, we propose the following Lyapunov function,

$$V = \tfrac{1}{2}\|\nabla_\theta J_{ip}(\theta)\|^2 \tag{14}$$

Note that the gradient of J_{ip} is now a function of a time-varying set of parameters $\theta(t)$ given by the expression

$$\nabla_\theta J_{ip}(\theta)^T = \nabla_\theta J(\theta)^T + \int_0^T \sum_{i=1}^{m} \frac{\mu_i}{\{w_i[\alpha(\bar{y}), \beta(\bar{y})] - \epsilon_i\}} \times \left(\frac{\partial w_i}{\partial x}\frac{\partial \alpha}{\partial \theta} + \frac{\partial w_i}{\partial u}\frac{\partial \beta}{\partial \theta}\right)\mathrm{d}\tau$$
$$M\left\{\alpha[\theta^T\Phi(0)] - x_0\right\}\frac{\partial \alpha}{\partial \theta}(0,\theta) + M\left\{\alpha[\theta^T\Phi(T)] - x_f\right\}\frac{\partial \alpha}{\partial \theta}(T,\theta) \tag{15}$$

where

$$\nabla_\theta J(\theta)^T = \int_0^T \left[\frac{\partial q}{\partial x}\frac{\partial \alpha}{\partial \theta}(\tau,\theta) + \frac{\partial q}{\partial u}\frac{\partial \beta}{\partial \theta}(\tau,\theta)\right]\mathrm{d}\tau \tag{16}$$

$$\frac{\partial \alpha}{\partial \theta}(t,\theta) = \frac{\partial \alpha}{\partial \bar{y}(t)}\frac{\partial \bar{y}(t)}{\partial \theta}, \qquad \frac{\partial \beta}{\partial \theta}(t,\theta) = \frac{\partial \beta}{\partial \bar{y}(t)}\frac{\partial \bar{y}(t)}{\partial \theta} \tag{17}$$

$$\frac{\partial \bar{y}(t)}{\partial \theta} = \left[\phi(t), \ldots, \phi(t)^{(k)}\right]$$

The time derivative of V is

$$\dot{V} = \nabla_\theta J_{ip}[\theta(t)]\left\{\nabla_\theta^2 J_{ip}[\theta(t)]\dot{\theta}\right\}$$

where $\nabla_\theta^2 J_{ip}[\theta(t)]$ is the Hessian of J_{ip} evaluated at $\theta(t)$ given by

$$\nabla_\theta^2 J_{ip}(\theta)^T = \nabla_\theta^2 J(\theta)^T + \int_0^T \left(\sum_{i=1}^{m} - \frac{\mu_i}{\{r_i[\alpha(\bar{y})] - \epsilon_i\}^2}\frac{\partial r_i}{\partial x}\frac{\partial \alpha}{\partial \theta} + \frac{\mu_i}{\{r_i[\alpha(\bar{y})] - \epsilon_i\}}\frac{\partial \alpha^T}{\partial \theta}\frac{\partial^2 r_i}{\partial x \partial x^T}\frac{\partial \alpha}{\partial \theta} + \frac{\mu_i}{\{r_i[\alpha(y)] - \epsilon_i\}}\frac{\partial r_i}{\partial x}\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T}\right)\mathrm{d}\tau$$
$$+ M\left\{\alpha[\theta^T\Phi(0)] - x_0\right\}\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T}(0,\theta) + M\frac{\partial \alpha^T}{\partial \theta}(0,\theta)\frac{\partial \alpha}{\partial \theta}(0,\theta) + M\left\{\alpha[\theta^T\Phi(T)] - x_f\right\}\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T}(T,\theta) + M\frac{\partial \alpha^T}{\partial \theta}(T,\theta)\frac{\partial \alpha}{\partial \theta}(T,\theta) \tag{18}$$

with

$$\nabla_\theta^2 J[\theta(t)] = \int_0^T \left(\frac{\partial q}{\partial x}\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T} + \frac{\partial \alpha^T}{\partial \theta}\frac{\partial^2 q}{\partial x \partial x^T}\frac{\partial \alpha}{\partial \theta} + \frac{\partial q}{\partial u}\frac{\partial^2 \beta}{\partial \theta \partial \theta^T} + \frac{\partial \beta^T}{\partial \theta}\frac{\partial^2 q}{\partial u \partial u^T}\frac{\partial \beta}{\partial \theta} + 2\frac{\partial \beta^T}{\partial \theta}\frac{\partial^2 q}{\partial u \partial x^T}\frac{\partial \alpha}{\partial \theta}\right)\mathrm{d}\tau \tag{19}$$

As a result, we obtain the following expression for the derivative of V

$$\dot{V} = \nabla_\theta J_{ip}[\theta(t)]\left\{\nabla_\theta^2 J_{ip}[\theta(t)]\dot{\theta}\right\} \tag{20}$$

We propose the following parameter update law:

$$\dot{\theta} = -k\Gamma^{-1}\nabla_\theta J_{ip}[\theta(t)]^T \tag{21}$$

where $\Gamma = \{\nabla_\theta^2 J_{ip}[\theta(t)] - \rho I\}$ and $\rho = \|\nabla_\theta^2 J_{ip}[\theta(t)]\|_F$ is the Frobenius norm of the Hessian matrix. Clearly, the matrix Γ is by construction negative definite such that

$$\dot{J}_{ip} = -\nabla_\theta J_{ip}[\theta(t)]\Gamma^{-1}\nabla_\theta J_{ip}[\theta(t)]^T \geq 0 \tag{22}$$

The cost functional is constrained to increase as long as the gradient is nonzero. Note that the rate of change of the Lyapunov function V becomes

$$\dot{V} = -k\nabla_\theta J_{ip}[\theta(t)]\nabla_\theta^2 J_{ip}[\theta(t)]\Gamma^{-1}\nabla_\theta J_{ip}[\theta(t)]^T$$

The cost functional is constrained to increase as long as the gradient is nonzero. Note that the rate of change of the Lyapunov function V becomes

$$\dot{V} = -k\nabla_\theta J_{ip}[\theta(t)]\nabla_\theta^2 J_{ip}[\theta(t)]\Gamma^{-1}\nabla_\theta J_{ip}[\theta(t)]^T$$

Because the Hessian $\nabla_\theta^2 J_{ip}[\theta(t)]$ is not necessarily negative definite (and because the corrected Hessian Γ is, by construction, negative definite), the strict negativity of $\dot{V}$ cannot be ensured. As a result, the norm of the gradient $\nabla_\theta J_{ip}$ can potentially become unbounded, leading to the divergence of the scheme. To avoid the divergence of the scheme, one must provide a mechanism to constrain the magnitude of the norm of the gradient of J_{ip}. In this paper, this is achieved by constraining the value of the parameters to a convex set of the form

$$\Omega_W = \left\{\theta \in \mathbb{R}^N \,|\, \|\theta\| \le w_m\right\}$$

for some $w_m > 0$. If one can provide a parameter update law that can restrict the values of the parameters to remain in Ω_w, then by continuous differentiability of J_{ip} one can ensure that the gradient of J_{ip} remains bounded.

Constraints on the parameters can be effectively enforced by using a projection algorithm. This algorithm is given by

$$\dot{\boldsymbol{\theta}} = \mathrm{Proj}\{\theta, \Psi\} =$$

$$\begin{cases} \Psi, & \text{if } \|\boldsymbol{\theta}\| < w_m \\ & \text{or } [\|\boldsymbol{\theta}\| = w_m \text{ and } \nabla\mathcal{P}(\boldsymbol{\theta})\Psi \le 0] \\ \Psi - \Psi \dfrac{\gamma\nabla\mathcal{P}(\boldsymbol{\theta})\nabla\mathcal{P}(\boldsymbol{\theta})^T}{\|\nabla\mathcal{P}(\boldsymbol{\theta})\|^2_\gamma}, & \text{otherwise} \end{cases}$$

where $\Psi = -k\Gamma^{-1}\nabla_\theta J[\theta(t)]^T$ and $\mathcal{P}(\boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{\theta} - w_m \le 0$, $\boldsymbol{\theta}$ is the vector of parameter estimates, γ is a positive definite symmetric matrix, and w_m is chosen such that $\|\boldsymbol{\theta}\| \le w_m$.

The relevant properties of the projection operator Proj $\{\boldsymbol{\theta}, \Psi\}$ are given in Ref. 21, Appendix E, p. 511. The purpose of the projection algorithm is to prevent the divergence of the optimization scheme. Although this can be achieved, it remains to check whether the maximization of the cost J can still proceed when a projection algorithm is employed.

By the properties of the projection algorithm, the parameters are guaranteed to remain in the convex set Ω_W. Furthermore, it is also guaranteed that the rate of change of J_{ip}, subject to the projection algorithm (23), given by

$$\dot{J}_{ip}(t) = \nabla_\theta J_{ip}[\boldsymbol{\theta}(t)]\,\mathrm{Proj}\{\boldsymbol{\theta}, \Psi\} \tag{23}$$

is such that

$$\dot{J}_{ip}(t) \ge 0$$

$\forall\theta \in \Omega_W$. Thus, the projection algorithm plays the role of a trust-region algorithm that limits the domain of the trajectories prescribed by Eq. (21) while ensuring the progress of the optimization algorithm. The restricted update law ensures that a local maximum of J can always be achieved over a convex set in the parameter space.

Note that by the smoothness of the cost J with respect to the decision variables θ, it is guaranteed that there exists an upper bound on the magnitude of the gradient and Hessian of J over the convex Ω_w.

Note that the optimality of the trajectories obtained is dependent on the choice of parameterization. The trajectories resulting from such finite dimensional approximations can only lead to approximate solutions of the dynamic optimization problem considered. As an illustration, it is generally known that the dynamic optimization problem can often lead to discontinuities that dissect different types of arcs in the optimization problem. The parameterization of the flat output trajectories forces the trajectories to be smooth and, thus, provides only approximate solutions of the true optimal trajectories. This problem, shared by many numerical methods for the solution of optimal control problems, has been documented in Ref. 9 for similar flatness-based approaches.

The purpose of the optimization strategy is to generate in real time a trajectory of the nonlinear system (1) that maximizes the cost functional J_{ip}. The approximate optimal trajectory provides a reference trajectory that is implemented by the control system.

Implementation

In this section, we propose a tracking controller that drives the state variables of the system to track the optimal state and input trajectories generated in real time. When flatness is used, it is straightforward to implement a tracking controller that provides asymptotic trajectory tracking.

Any flat system can be transformed via a dynamic feedback transformation to a Brunovsky form given by

$$\dot{z}_{i1} = z_{i2}, \ldots, \dot{z}_{i\nu_i - 1}$$

$$= z_{i\nu_i}, \dot{z}_{i\nu_i} = \alpha_i\left(z_{11}, \ldots, z_{1\nu_1}, \ldots, z_{p1}, \ldots, z_{p\nu_p}\right)$$

or

$$\dot{z}_i = A_i z_i + B_i\alpha_i\left(z_{11}, \ldots, z_{1\nu_1}, \ldots, z_{p1}, \ldots, z_{p\nu_p}\right)$$

where $z_i = [z_{i1}, \ldots, z_{i\nu_i}]^T$.

$$A_i = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ 0 & 0 & 0 & \ldots & 0 \end{bmatrix}, \qquad B_i = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

for $i = 1, \ldots, p$, where ν_i are the Kronecker indices of the Brunovsky form, and the flat outputs are represented by $y_i = z_{1i}$ for $i = 1, \ldots, p$. The dynamics of the flat system can be summarized as follows:

$$\dot{z} = Az + Bv \tag{24}$$

where $A = \text{BlockDiag}\,[A_1, A_2, \ldots, A_p]$, $B = \text{BlockDiag}[B_1, B_2, \ldots, B_p]$, $z = [z_1^T, z_2^T, \ldots, z_p^T$, $v = [\alpha_1, \alpha_2, \ldots, \alpha_p])$. The functions α_i are time-varying functions of the z_1 and u and their derivatives.

When the dynamic optimization strategy just highlighted is followed, the parameters $\theta(t)$ (evaluated at time t) encode the reference trajectory for the highest derivative of the flat outputs, written as

$$\xi^{(\nu)} = \left[\xi_1^{(\nu_1)}, \xi_2^{(\nu_2)}, \ldots, \xi_p^{(\nu_p)}\right]^T = \theta(t)^T\phi(t) \tag{25}$$

where $\phi(t)$ is an N by p matrix of basis functions chosen to parameterize the system dynamics. The lower-order derivatives are obtained by successively integrating the elements of Eq. (25) over the interval $[0, T]$.

Given the reference trajectories for the flat outputs, $\xi_1(t), \ldots, \xi_p(t)$, and their derivatives, we define the tracking errors

$$e_{i1} = z_{i1} - \xi_i(t), \ldots, e_{i\nu_i} = z_{i\nu_i} - \xi_i^{(\nu_i - 1)}(t)$$

for $i = 1, \ldots, p$. The tracking error dynamics are given by

$$\dot{e}_{i1} = z_{i2} - \xi_i^{(2)}(t), \ldots, \dot{e}_{i\nu_i}$$

$$= \alpha_i\left(z_{11}, \ldots, z_{1\nu_1}, \ldots, z_{p1}, \ldots, z_{p\nu_p}\right) - \xi_i^{(\nu_i)} \tag{26}$$

where $i = 1, \ldots, p$. The linear term α_i is a time-varying function of the state variables, the input variables, and their derivatives. Let the highest order of differentiation of the input variable u_i be given by ρ_i. The p derivatives are summarized in vector form as

$$\boldsymbol{u}^{(\rho)} = \left[u_1^{(\rho_1)}, \ldots, u_o^{(\rho_p)}\right]^T$$

By the controllability property of the linear system, the vector-valued function $\alpha = [\alpha_1, \ldots, \alpha_p]^T$ is such that rank $[\partial\alpha/\partial\boldsymbol{u}^{(\rho)}] = p$. Following a standard argument, it is possible to develop a tracking controller that provides asymptotic tracking. For the class of linear time-varying control systems, the function α takes the form

$$\alpha = \tilde{A}(t)\bar{y}(t) + \tilde{B}(t)u^\rho$$

The error dynamics are given by

$$\dot{e} = Ae + B\left(\alpha - \xi^{(\nu)}\right)$$

where $e = [e_1^T, \ldots, e_p^T]^T$ with $e_i = [e_{i1}, \ldots, e_{i\nu_i}]^T$.

By assumption, we have that the matrix $\tilde{B}(t)$ is nonsingular for all time $t \in [0, T]$. A suitable tracking controller is then given by

$$u^{\rho} = -[\tilde{B}(t)]^{-1}\left\{\tilde{A}(t)\bar{y}(t) - Ke + \begin{bmatrix} \xi_1^{(\nu_1)} \\ \vdots \\ \xi_p^{\nu_p} \end{bmatrix}\right\}$$

The gain matrix $K \in \Re^{p\times n}$ is chosen such that the matrix $A - BK$ is Hurwitz where A and B are in Brunovsky form.

The preceding leads to the closed-loop error dynamics

$$\dot{e} = (A - BK)e \tag{27}$$

which guarantees that the origin of the error dynamics is globally asymptotically stable. Because, by construction, the tracking dynamics arebounded, it follows that the trajectories of the state variables and the input variables are bounded and converge to the reference trajectory generated by the real-time optimization system.

In the next section, we present two simulation examples that show the effectiveness of the proposed method for real-time optimization of systems.

Simulation Results

Dynamic Optimization of Two-State-System

In the first example, we consider a simple two-state linear time-varying system. The matrices $\boldsymbol{A}(t)$ and $\boldsymbol{B}(t)$ are given by

$$\boldsymbol{A} = \begin{bmatrix} e^{-t} & 1 \\ 0 & e^{t} \end{bmatrix}, \qquad \boldsymbol{B} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

To demonstrate the constraint handling capability of the technique, we consider the following dynamic optimization problem:

$$\min_{u} \frac{1}{2}\int_0^1 \left[\frac{1}{2}x_1(t)^2 + \frac{1}{2}x_2(t)^2\right] \mathrm{d}t$$

so that

$$\dot{\boldsymbol{x}} = \boldsymbol{A}(t)\boldsymbol{x} + \boldsymbol{B}(t)\boldsymbol{u}, \qquad t \in [0, 1]$$
$$|u_1(t)| \le 10$$
$$\boldsymbol{x}(0) = [2 \quad 0]^T$$
$$\boldsymbol{x}(1) = [0 \quad 0]^T$$

This dynamic system is differentially flat with flat output $y = x_1$. The derivatives of the flat output are given by

$$\dot{y} = e^{-t}x_1 + x_2$$
$$\ddot{y} = e^{-t}x_1 + e^{-2t}x_1 + e^{-t}x_2 + e^{t}x_2 + 2u_1$$

They provide a transformation of the state and input space. Using flatness, we parameterize $\ddot{y}$ as a fourth-order polynomial. A tracking controller is designed as shown in the preceding section with controller gain $K = [-500, -160]$. The control action is saturated such that $|u_1(t)| \le 10\,\forall t \in [0, 1]$.

For the real-time optimization scheme, all parameters are initiated at zero. The parameter update law was implemented with a gain $k = 100$. The projection algorithm ensures that the parameters are restricted to evolve within a ball of radius 10. The log-barrier function parameters were set to $\mu = 1$ and $\epsilon = 0.0001$ for both constraints $\{u_1 \ge -10\}$ and $\{u_1 \le 10\}$. The initial and final state constraints are implement with the parameter $M = 10{,}000$.

The result of the simulation are shown in Figs. 1–3. The value of the extended cost J_{ip} is shown in Fig. 1. The results demonstrate that the real-time optimization scheme converges quickly to the local optimum $J_{ip} = 2.61$. The closed-loop state trajectories and the corresponding approximate optimal trajectories to be tracked are shown in Fig. 2. The tracking controller implements the approximate optimal trajectory effectively for this simple problem. The corresponding input trajectories are shown in Fig. 3. The approximate optimal input trajectory demonstrates that the constraints are active at the optimum. The tracking controller is shown to deviate

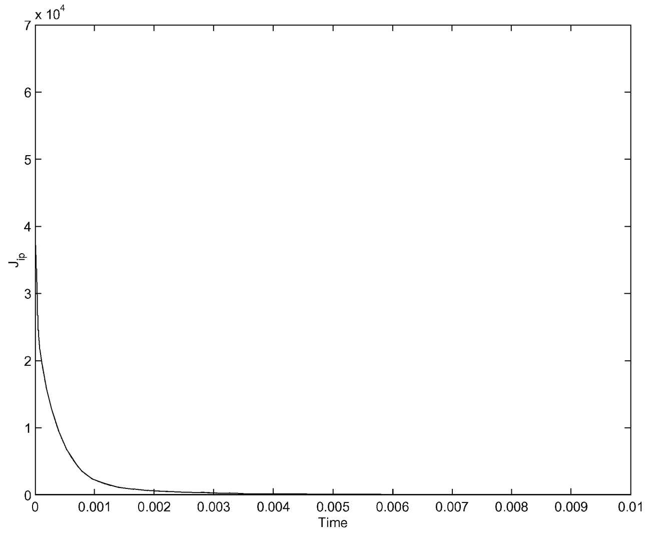


Fig. 1 Cost functional J for real-time optimization scheme.

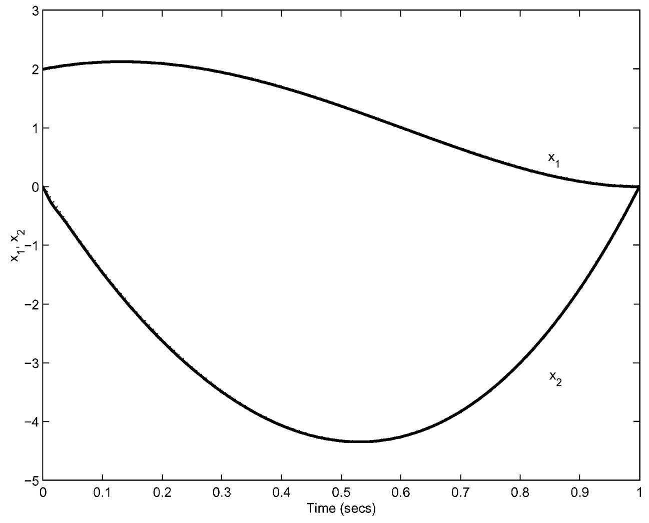


Fig. 2 Closed-loop ——, state variable trajectories and - - - -, optimal state trajectories for real-time optimization scheme.

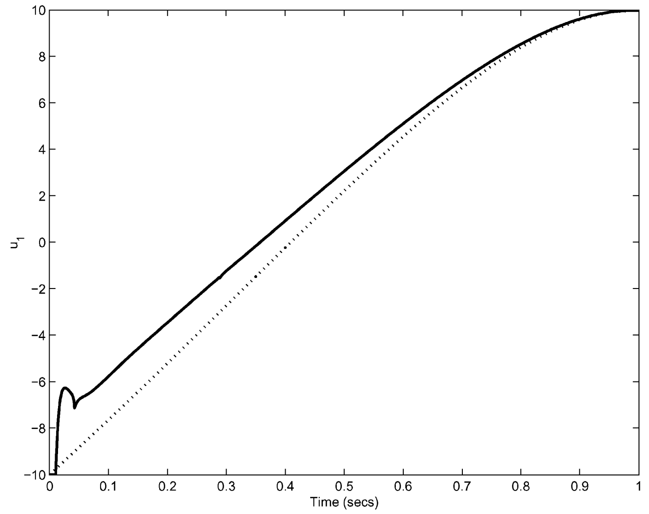


Fig. 3 Closed-loop ——, input variable trajectories and - - - -, optimal input trajectories for real-time optimization scheme.

somewhat for the optimal trajectory. This was a result of the initial adaptation that occurs at the beginning of the test run. Overall, the simulation results demonstrate that the real-time optimizing controller performs well.

Spring–Mass–Damper System

The second example is the spring–mass–damper system treated in Ref. 13. The system consists of four state variables $x_1 = \dot{q}_1, x_2 = \dot{q}_2$, $x_3 = q_1$, and $x_4 = q_2$ and two inputs u'_1 and u'_2. The matrices $\boldsymbol{A}(t)$ and $\boldsymbol{B}(t)$ are

$$\boldsymbol{A} = \begin{bmatrix} -\dfrac{c_1 + c_2}{m_1} & \dfrac{c_2}{m_1} & -\dfrac{k_1 + k_2}{m_1} & \dfrac{k_2}{m_1} \\ \dfrac{c_1}{m_2} & -\dfrac{c_2 + c_3}{m_2} & \dfrac{k_2}{m_2} & -\dfrac{k_2 + k_3}{m_2} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} \dfrac{1}{m_1} & 0 \\ 0 & \dfrac{1}{m_2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

where m_1 and m_2 are the masses; k_1, k_2, and k_3 are the spring stiffness; and c_1, c_2, and c_3 are the damping coefficients. Following Ref. 13, we let $m_1 = m_2 = 1$, $c_1 = c_2 = c_3 = 1$, $k_1 = 3t$, $k_2 = k_3 = 3$, and $u'_1 = e^{-t}u_1, u'_2 = u_2$, and $\boldsymbol{u} = [u_1, u_2]^T$. The boundary conditions are $\boldsymbol{x}(0) = [0\ \ 0\ \ 10\ \ 20]^T$ and $\boldsymbol{x}(1) = [0\ \ 0\ \ 0\ \ 0]^T$. The dynamic optimization problem is

$$\min_u \frac{1}{2} \int_0^1 \boldsymbol{u}^T \boldsymbol{u}\, \mathrm{d}t$$

so that

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}(t)\boldsymbol{x} + \boldsymbol{B}(t)\boldsymbol{u}, \qquad t \in [0, 1] \\ \boldsymbol{x}(0) &= [0\quad 0\quad 10\quad 20]^T \\ \boldsymbol{x}(1) &= [0\quad 0\quad 0\quad 0]^T \end{aligned}$$

This linear time-varying system is controllable and, hence, differentially flat. The flat outputs are simply $y_1 = x_3$ and $y_2 = x_4$. The state and input of the system are given by

$$\begin{aligned} &y_1 = x_3, \qquad \dot{y}_1 = x_1 \\ &\ddot{y}_1 = -[(c_1 + c_2)/m_1]y_1 + (c_2/m_1)\dot{y}_2 - [(k_1 + k_2)/m_1]y_1 \\ &\qquad + (k_2/m_1)y_2 + (1/m_1)\exp(-t)u_1 \\ &y_2 = x_4, \qquad \dot{y}_2 = x_2 \\ &\ddot{y}_2 = (c_2/m_2)\dot{y}_1 - [(c_2 + c_3)/m_2]y_2 + (k_2/m_2)y_1 \\ &\qquad - [(k_2 + k_3)/m_2]y_2 + (1/m_2)u_2 \end{aligned}$$

For all of the simulations, we consider polynomial basis functions with $N = 4$ such that $\chi_{i,j} = t^{j-1}$ for $i = 1, 2$. The initial parameter estimates are $\theta(0) = [0, 0, 0, 0, 0, 0, 0, 0]$. The parameters were constrained to the ball of radius 100 centered at the initial parameter estimates.

The following tracking controller was shown to have satisfactory performance,

$$v(t) = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = K_c e - \begin{bmatrix} \ddot{y}_1 \\ \ddot{y}_2 \end{bmatrix} \tag{28}$$

where

$$K_c = \begin{bmatrix} -100 & 0 & -100 & 0 \\ 0 & -100 & 0 & -100 \end{bmatrix} \tag{29}$$

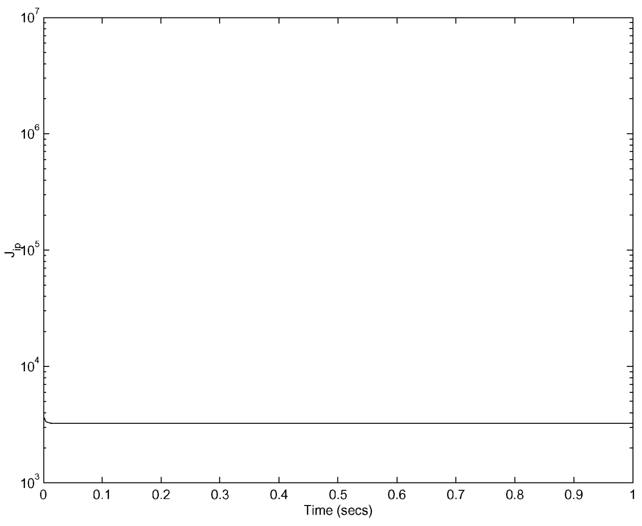
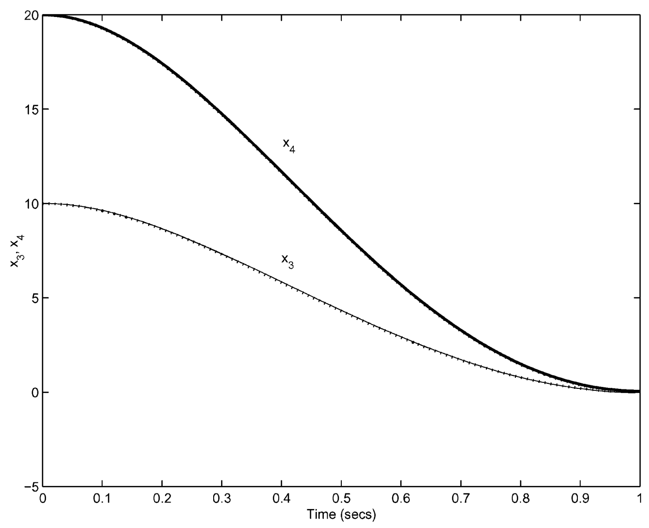
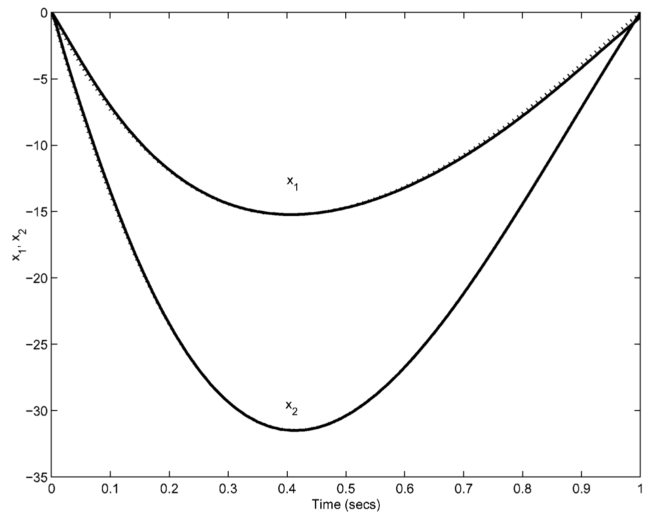


Fig. 4 Cost functional J for real-time optimization scheme.



a)



b)

Fig. 5 Closed-loop ——, state variable trajectories and - - - -, optimal state trajectories for real-time optimization scheme.

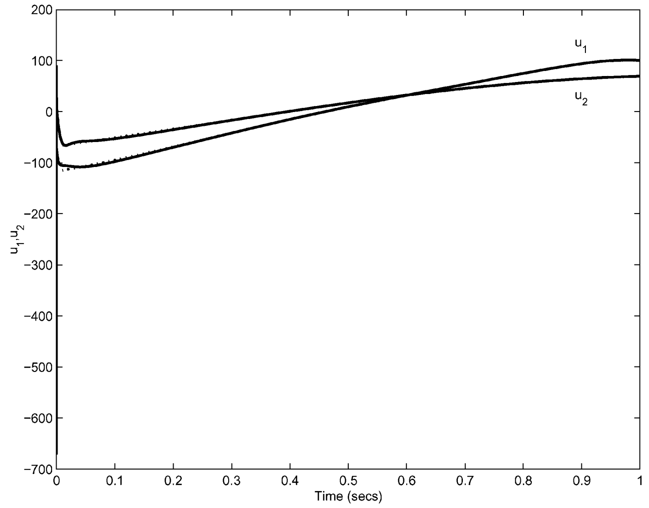


Fig. 6 Closed-loop ——, input variable trajectories and - - - -, optimal input trajectories for real-time optimization scheme.

The real-time optimization scheme is implemented with a gain of $k = 100$. Figure 4 shows the variation in the modified cost functional J_{ip}. As expected, the real-time optimization routine provides a quick decrease of the cost function to its minimum value. Figure 5 shows the resulting approximate optimal state trajectories along with the closed-loop state trajectories of the tracking controller. Figure 6 shows the corresponding input trajectories. The results first demonstrate that the real-time optimization technique is able to compute the approximate optimal state trajectories correctly as given in Ref. 13. Second, the results show that the tracking controller designed is able to implement the approximate optimal trajectories.

The simulation study was performed on a Pentium M 2.0-GHz Intel processor using a standard SIMULINK computation platform. The total computation time for the optimization problem including simulation of the linear system was 0.844 CPU s. A total of 147 time steps were computed yielding an average of 5.7 μs per iteration. In the current case, a final time of 1 s was assumed for the completed task. The total optimization steps as the time for iteration were, therefore, sufficient for practical implementation of the real-time dynamic optimization routine. Thus, the real-time optimization objective is achieved. Note that the current implementation performance could be drastically improved by choosing a more suitable real-time platform. Further work is ongoing in this area to develop the full potential of the technique described in this work.

Conclusions

In this paper, we proposed and solved an extremum-seeking control problem for a class of linear time-varying dynamical control systems. The system provides a real-time optimal trajectory generation system that optimizes cost functionals evaluated over a fixed timed horizon. Two examples were considered to demonstrate that the extremum-seeking controller can compute and regulate a nonlinear dynamical system to an optimal trajectory generated in real-time.

Acknowledgment

The authors would like to acknowledge the financial support of the Natural Sciences and Engineering Research Council.

References

[1]Brockett, R., *Finite Dimensional Linear Systems*, Wiley, New York, 1970, Chap. 3, p. 124.

[2]Bryson, A., and Ho, Y., *Applied Optimal Control*, Hemisphere, Washington, DC, 1975.

[3]Luus, R., "Optimal Control by Dynamic Programming Using Systematic Reduction in Grid Size," *International Journal of Control*, Vol. 35, No. 5, 1990, pp. 995–1013.

[4]Cuthrell, J., and Biegler, L., "On the Optimization of Differential Algebraic Process Systems," *AIChE Journal*, Vol. 33, No. 8, 1987, pp. 1257–1270.

[5]Binder, T., Cruse, T. A., and Villar, C. C., "Dynamic Optimization Using a Wavelet Based Adaptive Control Vector Parameterization Strategy," *Computers and Chemical Engineering*, Vol. 4, No. 2, 2000, pp. 1201–1207.

[6]Delaleau, E., and Rudolph, J., "Control of Flat Systems by Quasi-Static Feedback of Generalized States," *International Journal of Control*, Vol. 71, No. 5, 1998, pp. 745–765.

[7]van Nieuwstadt, M., and Murray, R., "Real-Time Trajectory Generation for Differentially Flat Systems," *International Journal of Robust and Nonlinear Control*, Vol. 8, No. 11, 1998, pp. 995–1020.

[8]Faiz, N., Agrawal, S., and Murray, R., "Differential Flat Systems with Inequality Constraints: An Approach to Real-Time Feasible Trajectory Generation," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, 2000, pp. 219–227.

[9]Oldenburg, J., and Marquardt, W., "Dynamic Optimization Based on Higher Order Differential Model Representations," *Computers and Chemical Engineering*, Vol. 26, No. 3, 2002, pp. 385–400.

[10]Mahadevan, R., Agrawal, S., and Doyle, F. J. III, "A Flatness Based Approach to Optimization in Fed-Batch Reactors," *Proceedings of the International Federation of Automatic Control 2000*, International Federation of Automatic Control, Laxenburg, Austria, 2000, pp. 111–116.

[11]Mahadevan, R., Agrawal, S., and Doyle, F. J. III, "Differential Flatness-Based Nonlinear Model Predictive Control of Fed-Batch Reactors," *Control Engineering Practice*, Vol. 9, No. 8, 2001, pp. 889–899.

[12]Guay, M., Kansal, S., and Forbes, J., "Dynamic Optimization of Differential Flat Nonlinear Systems," *Industrial Engineering Chemistry Research*, Vol. 40, No. 9, 2001, pp. 2089–2102.

[13]Xu, X., and Agrawal, S., "Linear Time-Varying Dynamic System Optimization via Higher-Order Method: A Sub-Domain Approach," *Journal of Vibration and Acoustics*, Vol. 122, No. 1, 2000, pp. 31–35.

[14]Guay, M., and Zhang, T., "Adaptive Extremum-Seeking Control of Nonlinear Systems with Parametric Uncertainties," *Automatica*, Vol. 30, No. 7, 2003, pp. 1283–1293.

[15]Rouchon, P., Fliess, M., Lévine, J., and Martin, P., "Flatness and Defect of Nonlinear Systems: Introductory Theory and Examples," *International Journal of Control*, Vol. 61, 1995, pp. 1327–1361.

[16]Agrawal, S., Faiz, N., and Murray, R., "Feasible Trajectories of Linear Dynamic with Inequality Constraints Using Higher-Order Representations," *Proceedings of the International Federation of Automatic Control 2000*, International Federation of Automatic Control, Laxenburg, Austria, 1999.

[17]Murray, R., Rathinam, M., and Sluis, W., "Differential Flatness of Mechanical Control Systems," *Proceedings of the 1995 ASME International Mechanical Engineering Congress*, Vol. 57-1, American Society of Mechanical Engineers, New York, 1995.

[18]Rothfuss, R., Rudolph, R., and Zeitz, M., "Flatness Based Control of a Nonlinear Chemical Reactor Model," *Automatica*, Vol. 32, No. 10, 1996, pp. 1433–1439.

[19]Fiacco, A., and McCormick, G., *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, Society for Industrial and Applied Mathematics, Philadelphia, 1990, Chap. 5, p. 72.

[20]Nocedal, J., and Wright, S., *Numerical Optimization*, Springer-Verlag, New York, 1999, Chap. p. 49.

[21]Krstić, M., Kanellakopoulos, I., and Kokotović, P., *Nonlinear and Adaptive Control Design*, Wiley, New York, 1995, Appendix E, p. 511.