Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Real-time Dynamic Optimization of Nonlinear Systems: A Flatness-based Approach

M. Guay

Abstract— **In this paper, a real-time optimizing controller is developed to steer a differentially flat nonlinear control system to closed-loop trajectories that optimize a cost functional of interest. An interior point optimization method with penalty function is used to formulate a real-time optimization scheme. The problem is posed as a real-time optimal trajectory generation problem in which the optimal trajectories are computed using an adaptive extremum-seeking approach. A fed-batch bioreactor simulation example is used to demonstrate the application of the technique to a finite time dynamic optimization problems that arise in batch process control.**

I. Introduction

The design of optimal controller for batch processes has been an area of active research of the last twenty years. In most applications, the optimization task consists of two distinct steps. First, a reference trajectory is generated. The trajectory is generally computed by solving a dynamic optimization problem. Second, a suitable tracking controller is designed to regulate the process about its nominal optimal trajectory. The main drawback of this approach is that the optimal trajectory, computed from a detailed process model, cannot be formulated to incorporate uncertainties and disturbances. Online optimization techniques have been proposed to provide sequential real-time updates of the optimal trajectory ([1],[2]). A leading approach employs a standard real-time optimization approach. In real-time optimization, process measurements are used to update process model parameters in real-time. The updated model is then used in the formulation of a dynamic optimization problem that is solved sequentially. The resulting control system can be used to update system trajectories at intervals that allow the solution of the dynamic optimization problem. In a number of applications, a cascade optimization structure consisting of a high level real-time optimization layer and a low-level tracking controller is used to provide regulation of the optimal trajectory ([3],[4]). In [5], a feedback-based implementation was proposed to incorporate feedback components in a cascade real-time optimization framework. In this approach, process parameter variations are taken into account in the formulation of the dynamic optimization problem. This integration provides some feedback compensation for the model changes which can effectively improve the performance of existing real-time optimization schemes. Despite significant interest and some convincing results, the main caveat of these techniques is the inherent delay associated with the solution of the dynamic optimization problem. It remains very difficult to provide an optimization technique that can perform in real-time.

One approach that has received considerable interest in the literature is the use of differential flatness in the formulation of dynamic optimization problems. Flat dynamical systems [6] belong to a special class of systems that are characterized by the existence of so-called flat outputs. The flat output space provides a reduced-dimensional space where the system trajectories can be defined freely in the absence of differential constraints. The ability to assign trajectories freely allows one to parameterize the process state variable trajectories to transform the dynamic optimization problem to a standard nonlinear optimization problem. Some results have been published on control of flat systems (e.g., [7]; [8]) and a number of flatness based dynamic optimization methods have been reported (e.g., [9]; [10]; [11]; [12]). One of the advantages of flatness-based approaches is the elimination of the numerical integration of the dynamical model and sensitivity equations in the computation of the optimal solution. Therefore, such approaches are very attractive for real-time applications ([11]).

In this paper, we develop a real-time optimization approach to solve dynamic-optimization problems arising in batch process control. The approach proposed provides a real-time optimization technique that can be used to simultaneously compute and implement optimal trajectories. The optimization technique uses a Lyapunov-based optimization method originally proposed in [13] for the solution of adaptive extremum-seeking control problems. In order to solve the problem, 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 paper is organized as follows. In Section 2, we state and formulate the dynamic optimization problem. The optimization technique and the implementation of the controller is discussed in Section 3. Simulation results are presented in Section 4 which is followed by brief conclusions in Section 5.

II. Problem Statement

In this paper, we consider a class of nonlinear control-affine dynamical system of the form:

$$\dot{x} = f(x) + \sum_{i=1}^{P} g_i(x)u_i(t) \tag{1}$$

Department of Chemical Engineering, Queen's University, Kingston, Ontario, Canada K7L 3N6 guaym@chee.queensu.ca

The authors would like to acknowledge the financial support of NSERC.

0-7803-9568-9/05/$20.00 ©2005 IEEE

where $x \in \mathbb{R}^n$ are the state variables and $\mathbf{u}(t) = [u_1(t), \ldots, u_p(t)]^T \in \mathbb{R}^p$ is the vector of p input variables, $f(x) : \mathbb{R}^n \to \mathbb{R}^n$ and $g_i(x) : \mathbb{R}^n \to \mathbb{R}^n$, for $i = 1, \ldots, p$, are C^∞ vector-valued function of the state variables x.

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

$$\max_{x(t),u(t)} \quad J = \int_0^T q(x(t), u(t))dt \tag{2}$$

$$\begin{aligned} \text{subject to:} \quad & \dot{x} = f(x) + \sum_{i=1}^{P} g_i(x)u_i(t) \\ & w(x(t), \mathbf{u}(t)) \geq 0 \\ & x(0) = x_0 \\ & x(T) = x_f \end{aligned} \tag{3}$$

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 $\mathbb{R}^n \times \mathbb{R}^p$. The constraint functions $w : \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^c$ in (3) is a C^∞ vector valued function of the x and u taking value in $\mathbb{R}^c$. It is assumed that the trajectories $x(\tau)$ evolve on a compact subset Ω of $\mathbb{R}^n$, the input trajectories take value in $\mathbb{R}^p$. The cost functional $J : \Omega \times \mathbb{R}^p \to \mathbb{R}^+$ is assumed to be a convex and continuously differentiable function on Ω. 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.

A. Flat Dynamical Systems

Differential flatness, a notion introduced in [6], 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 dynamical system by an appropriate choice of endogenous dynamic state feedback transformations. Such feedback structures describe a special class of static or dynamic pre-compensators formed only of endogenous system variables, such as state variables, inputs and a finite number of their time derivatives.

The system (1) is said to be differentially flat if there exists variables $y = [y_1, ..., y_p]^T$ given by an equation of the form:

$$y = \mathbf{h}(x, u, \dot{u}, ..., u^{(\rho)}) \tag{4}$$

where $\mathbf{h}$ is a C^∞ vector-valued function. The variables $y = [y_1, ..., y_p]^T$ are referred to as the flat outputs.

The original system variables $\mathbf{x}(t)$ and $\mathbf{u}(t)$ are written as functions of these *flat outputs* ($\mathbf{y}(t)$) and a finite number of their derivatives:

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

The functions $\alpha(\dot{)}$ and $\beta(\dot{)}$ are C^∞ vector-valued functions. Here $\mathbf{y}^{(i)}(t)$ stands for the i^{th} derivative of $\mathbf{y}(t)$ with respect to t and k is the number of derivatives of $\mathbf{y}(t)$ required to represent the system in the form given in (5). In addition, $\bar{\mathbf{y}}(t)$ is a vector of derivatives of the flat output, that is,

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

B. Dynamic Optimization using Flatness

Using flatness, 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 (5). This strategy has been employed in many studies (see, [14], [11], [15], [10], [16], [6] and references therein). The choice of parametrization depends entirely on the specific application. In general, we parameterize the highest derivative of the flat outputs, $y^{(k)}(t)$, as

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

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

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

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

$$\mathbf{y}(t) = \theta^T \int \ldots \int_k \Xi(t)$$

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

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

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

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

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

The constraints are re-parameterized as follows:

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

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

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

subject to: (9)

$$w(\alpha(\theta^T\Phi(\tau)),\beta(\theta^T\Phi(\tau))) \geq 0$$

$$\alpha(\theta^T\Phi(0)) = x_0 \quad (10)$$

$$\alpha(\theta^T\Phi(T)) = x_f \quad (11)$$

III. Extremum-Seeking Dynamic Optimization

The objective of the controller design methodology is to steer the system to the parameterized optimal state and input trajectories that solves the nonlinear optimization problem (9). In the remainder, we consider the cost functional (7) as a function of θ, $J(\theta)$. We propose to solve the optimization problem (9) using an interior/exterior point method to incorporate the constraints and the boundary conditions. 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. The boundary conditions (10)-(11) are incorporated using an exterior point method by adding a penalty function to the cost. This leads to a modified cost function given by

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

where $\mu_i > 0$, $\epsilon_i > 0$ for $i = 1,\ldots,m$ and $M > 0$ are positive constants that are the tuning parameters for the interior/exterior 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 3.1: The constraint set described by equation (3), which is convex of the set $\Omega \subset \mathbb{R}^n$, remains convex over a set Υ in the parameter space in its parameterized form (8).

Assumption 3.1 guarantees that the unconstrained optimization of the modified cost J_{ip} leads to the constrained optimum of $J(\theta)$ as the tuning constants $\mu_i \to 0$ and $M \to \infty$. 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.

A. 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,

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

where $\nabla_\theta J_{ip}(\theta^*)$ is the gradient of J_{ip} with respect to θ evaluated at the minimizer θ^*. As in [13], we propose the following Lyapunov function,

$$V = \frac{1}{2}\left\|\nabla_\theta J_{ip}(\theta)\right\|^2 \quad (14)$$

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

$$\begin{aligned} \nabla_\theta J_{ip}(\theta)^T &= \nabla_\theta J(\theta)^T \\ &+ \int_0^T \sum_{i=1}^m \frac{\mu_i}{(w_i(\alpha(\bar{\mathbf{y}}),\beta(\bar{\mathbf{y}})) - \epsilon_i)}\left(\frac{\partial w_i}{\partial x}\frac{\partial \alpha}{\partial \theta} + \frac{\partial w_i}{\partial u}\frac{\partial \beta}{\partial \theta}\right)d\tau \\ &+ M(\alpha(\theta^T\Phi(0)) - x_0)\frac{\partial \alpha}{\partial \theta}(0,\theta) + M(\alpha(\theta^T\Phi(T)) - x_f)\frac{\partial \alpha}{\partial \theta}(T,\theta) \end{aligned} \quad (15)$$

where

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

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

and

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

The time derivative of V is

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

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

$$\begin{aligned} \nabla^2_\theta J_{ip}(\theta)^T &= \nabla^2_\theta J(\theta)^T + \int_0^T \Bigg(\sum_{i=1}^m -\frac{\mu_i}{(w_i(\alpha(\bar{\mathbf{y}})) - \epsilon_i)^2}\frac{\partial w_i}{\partial x}\frac{\partial \alpha}{\partial \theta} \\ &+ \frac{\mu_i}{(w_i(\alpha(\bar{\mathbf{y}})) - \epsilon_i)}\frac{\partial \alpha^T}{\partial \theta}\frac{\partial^2 w_i}{\partial x \partial x^T}\frac{\partial \alpha}{\partial \theta} + \frac{\mu_i}{(r_i(\alpha(\bar{\mathbf{y}})) - \epsilon_i)}\frac{\partial r_i}{\partial x}\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T}\Bigg)d\tau \\ &+ M(\alpha(\theta^T\Phi(0)) - x_0)\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T}(0,\theta) + M\frac{\partial \alpha^T}{\partial \theta}(0,\theta)\frac{\partial \alpha}{\partial \theta}(0,\theta) \\ &+ M(\alpha(\theta^T\Phi(T)) - x_f)\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T}(T,\theta) + M\frac{\partial \alpha^T}{\partial \theta}(T,\theta)\frac{\partial \alpha}{\partial \theta}(T,\theta) \end{aligned} \quad (18)$$

with

$$\begin{aligned} \nabla_\theta{}^2 J(\theta(t)) &= \int_0^T \Bigg(\frac{\partial q}{\partial x}\frac{\partial^2 \alpha}{\partial \theta \partial \theta^T} + \frac{\partial \alpha^T}{\partial \theta}\frac{\partial^2 q}{\partial x \partial x^T}\frac{\partial \alpha}{\partial \theta} + \frac{\partial q}{\partial u}\frac{\partial^2 \beta}{\partial \theta \partial \theta^T} \\ &+ \frac{\partial \beta^T}{\partial \theta}\frac{\partial^2 q}{\partial u \partial u^T}\frac{\partial \beta}{\partial \theta} + 2\frac{\partial \beta^T}{\partial \theta}\frac{\partial^2 q}{\partial u \partial x^T}\frac{\partial \alpha}{\partial \theta}\Bigg)d\tau. \end{aligned} \quad (19)$$

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

$$\dot{V} = \nabla_\theta J_{ip}(\theta(t))\left(\nabla_\theta{}^2 J_{ip}(\theta(t))\dot{\theta}\right). \quad (20)$$

We propose the following parameter update law:

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

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

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

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 correction of the Hessian renders the rate of change of V indefinite. In order to avoid divergence of the scheme, the value of the parameters is constrained to the convex set

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

for some $w_m > 0$ through the use of a projection algorithm. This algorithm is given by

$$\dot{\theta} = = \begin{cases} \Psi, \text{ if } \|\theta\| < w_m \\ \text{or } (\|\theta\| = w_m \text{ and } \nabla\mathcal{P}(\theta)\Psi \leq 0) \\ \Psi - \Psi\frac{\gamma\nabla\mathcal{P}(\theta)\nabla\mathcal{P}(\theta)^T}{\|\nabla\mathcal{P}(\theta)\|^2_\gamma}, \text{otherwise} \end{cases}$$

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

The relevant properties of the projection operator, (23), are given in [17]. 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

$$\dot{J}_{ip}(t) = \nabla_\theta J_{ip}(\theta(t))Proj\left\{\theta, \Psi\right\} \tag{23}$$

is such that

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

$\forall\theta \in \Omega_W$. Thus, the projection algorithm plays the role of a trust-region algorithm which limits the domain of the trajectories prescribed by equation (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 in the magnitude of the gradient and Hessian of J over the convex set Ω_W.

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

B. Implementation

In this section, we propose a tracking controller that drives the state variables of the system to track the approximate optimal trajectory generated in real-time. Using flatness, 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

$$\begin{aligned} \dot{z}_{i1} &= z_{i2}, \ldots, \dot{z}_{i\nu_i - 1} = z_{i\nu_i}, \\ \dot{z}_{i\nu_i} &= \alpha_i(z_{11}, \ldots, z_{1\nu_1}, \ldots, z_{p1}, \ldots, z_{p\nu_p}) \end{aligned}$$

or

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

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

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

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 $A = \text{BlockDiag}\left[A_1, A_2, \ldots, A_p\right]$, $B = \text{BlockDiag}\left[B_1, B_2, \ldots, B_p\right]$, $z = \left[z_1^T, z_2^T, \ldots, z_p^T\right]$, $v = [\alpha_1, \alpha_2, \ldots, \alpha_p]$). The functions α_i are understood to be smooth nonlinear time-varying functions of the flat outputs z_{i1} ,$(i = 1, \ldots, p)$, u and their derivatives.

Following the dynamic optimization strategy highlighted above, 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 (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

$$\begin{aligned} \dot{e}_{i1} &= z_{i2} - \xi_i^{(2)}(t), \; \ldots, \\ \dot{e}_{i\nu_i} &= \alpha_i(z_{11}, \ldots, z_{1\nu_1}, \ldots, z_{p1}, \ldots, z_{p\nu_p}) - \xi_i^{(\nu_i)} \end{aligned} \tag{26}$$

where $i = 1, \ldots, p$. 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

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

By the flatness (and hence the controllability) property, the vector-valued function $\alpha = [\alpha_1, \ldots, \alpha_p]^T$ is such that rank $\left[\frac{\partial\alpha}{\partial\mathbf{u}^{(\rho)}}\right] = p$. Following a standard argument, it is possible to develop a tracking controller that provides asymptotic

tracking. For the class of control-affine nonlinear systems, the function α takes the form

$$\alpha = F(x, u(t), \ldots, u^{(\rho-1)}) + G(x, u(t), \ldots, u^{(\rho-1)})u^{\rho}$$

By assumption, we have that the matrix $G(x, u(t), ..., u^{(\rho-1)})$ is nonsingular over the set Ω. A suitable tracking controller is then given by

$$\begin{aligned} u^{\rho} &= -[G(x, u(t), \ldots, u^{(\rho-1)})]^{-1} \\ &\times \left(F(x, u(t), \ldots, u^{(\rho-1)}) - Ke + \begin{bmatrix} \xi_1^{(\nu_1)} \\ \vdots \\ \xi_p^{\nu_p} \end{bmatrix} \right) \end{aligned} \quad (27)$$

The matrix $K \in \Re^{p \times n}$ is chosen such that the matrix $A-BK$ is Hurwitz . The implementation of the tracking controller (27) leads to the closed-loop error dynamics

$$\dot{e} = (A - BK)e \quad (28)$$

which guarantees that the origin of the error dynamics is globally asymptotically stable. Since by construction the tracking dynamics are bounded, 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.

IV. Dynamic Optimization of a Fed-batch Bioreactor

In this example, adapted from [5], a bioreactor is used to produce large quantities of a drug P. The dynamics of the bioreactor are given by:

$$\begin{aligned} \dot{x_1} &= \frac{\alpha_m x_1 x_2}{x_2 + K_l x_1} - D x_1 \quad (29) \\ \dot{x_2} &= -\frac{1}{Y_{xs}} \frac{\alpha_m x_1 x_2}{x_2 + K_l x_1} - \frac{1}{Y_p} \frac{\theta_m x_1 x_2}{K_p + x_2 + x_2^2/K_I} \\ &\quad - M_x x_1 - D x_2 + D u_1 \end{aligned}$$

where $x_1(g/l)$ is the biomass concentration, $x_2(g/l)$ is the substrate concentration, $u_1(g/l)$ is the inlet substrate concentration and $D(hr^{-1})$ is the dilution rate. We assume that the dilution rate remains fixed at 0.040 (hr^{-1}). The values of the model parameters are listed in Table IV.

Parameter	Value	Parameter	Value
α_m	0.11 (hr^{-1})	K_l	1
Y_{xs}	0.47	Y_p	1.2
K_p	0.0001 (g/l)	K_I	0.1 (g/l)
M_x	0.029 (hr^{-1})	θ_m	0.004 (hr^{-1})

Table IV
Model Parameters for the Bioreactor

The dynamic optimization problem is given as follows:

$$\begin{aligned} \min_{\mathbf{u}} \quad & -\int_0^{t_f} \frac{\theta_m x_1 x_2}{K_p + x_2 + x_2^2/K_I} dt \\ s.t. \quad & \text{eq.(29)}, \ \forall t \in [0, 1], \\ & x_1(0) = 1 \ (g/l), \ x_2(0) = 0.2 \ (g/l), \\ & 0 \le x_1(t) \le 40(g/l) \\ & 0 \le x_2(t) \le 0.5(g/l) \ \forall t \in [0, 1] \\ & 0 \le u_1(t) \le 400(g/l) \ . \end{aligned}$$

If one uses the inlet substrate concentration, u_1, as the input, this system is differential flat with the biomass concentration, $y = x_1$ as a flat output. Differentiating y twice with respect to t provides the required parametrization of the state and input trajectories.

A sixth order polynomial is used to approximate the system's trajectories. The real-time optimization gain k is set 0.1. Six log barrier functions with $\mu = 1$ and $\epsilon_1 = 0.001$ are required to enforce the constraints $0 \le x_1(t) \le 40$, $0 \le x_2(t) \le 0.5$ and $0 \le u_1(t) \ge 400$. The penalty function parameter is to $M = 1000$. The parameter estimates are initialized at $\hat{\theta}(0) = [1, \ 0, \ 0, \ 0, \ 0, \ 0, \ 0]$. The tracking controller gain is given by $K = [-0.01, -0.01]$.

The result of the simulation are shown in Figures 1-5. The value of the extended cost J_{ip} is shown in Figure 1. The results demonstrate that the real-time optimization scheme converges quickly to the local optimum $J_{ip} = -1823$. The cumulative value of the nominal cost functional $J(t)$, which is indicative of the productivity of the batch reactor, is shown in figure 2. The trajectory and the corresponding optimal trajectory of the flat output x_1 are shown in Figure 3. The tracking controller implements the optimal trajectory effectively for this simple problem. The substrate concentration, x_2, is shown in Figure 4. The corresponding input trajectories are shown in 5. The tracking controller deviates slightly from the optimal trajectory to compensate for the effect of the initial adaptation that occurs at the beginning of the time interval.

V. Conclusions

In this paper, we proposed and solved an extremum-seeking control for the design of real-time dynamic optimization problems for a class of nonlinear dynamical control systems. The approach provides a real-time trajectory generation system that computes the optimal system trajectories while a suitable tracking controller is used to regulate the process. A simulation example considered to demonstrate the effectiveness of the real-time optimization controller. Although the current results is restricted to flat dynamical systems, the technique developed in this paper is applicable to general nonlinear systems assuming that a suitable parametrization of the systems' trajectories is available.

References

[1] J. Eaton and J. Rawlings, "Feedback control of nonlinear processes using on-line optimization techniques," *Computers Chem. Engng.*, vol. 14, pp. 469–479, 1990.

[2] S. Palanki, C. Kravaris, and H. Wang, "Synthesis of state feedback laws for end-point optimization in batch processes," *Chem. Eng. Science*, vol. 48, no. 1, pp. 135–152, 1993.
[3] B. Srinivasan, E. Visser, and D. Bonvin, "Optimization-based control with imposed feedback structures," in *Proc. IFAC ADCHEM'97*, Banff, Canada, 1997, pp. 635–640.
[4] M. Krothapally and S. Palanki, "A neural network strategy for batch process optimization," *Computers Chem. Engng.*, vol. 21, pp. 463–468, 1997.
[5] E. Visser, B. Srinivasan, S. Palanki, and D. Bonvin, "A feedback-based implementation scheme for batch process optimization," *J. Proc. Control*, vol. 10, pp. 399–410, 2000.
[6] P. Rouchon, M. Fliess, J. Lévine, and P. Martin, "Flatness and defect of nonlinear systems: Introductory theory and examples," *Int. J. Control*, vol. 61, no. 6, pp. 1327–1361, 1995.
[7] E. Delaleau and J. Rudolph, "Control of flat systems by quasi-static feedback of generalized states," *International Journal of Control*, vol. 71, no. 5, pp. 745–765, 1998.
[8] M. van Nieuwstadt and R. Murray, "Real-time trajectory generation for differentially flat systems."
[9] N. Faiz, S. Agrawal, and R. Murray, "Differential flat systems with inequality constraints: an approach to real-time feasible trajectory generation," *AIAA Journal Guidance, Navigation and Control*, vol. 24, no. 2, pp. 219–227, 2000.
[10] J. Oldenburg and W. Marquardt, "Dynamic optimization based on higher order differential model representations," in *Proc. ADCHEM 2000*, Pisa, Italy, 2000, pp. 809–814.
[11] R. Mahadevan, S. Agrawal, and F. D. III, "A flatness based approach to optimization in fed-batch reactors," in *Proc. ADCHEM 2000*.
[12] M. Guay, S. Kansal, and J. Forbes, "Dynamic optimization of differential flat nonlinear systems," *Ind. Eng. Chem. Res.*, vol. 40, no. 9, pp. 2089–2102, 2001.
[13] M. Guay and T. Zhang, "Adaptive extremum-seeking control of nonlinear systems with parametric uncertainties," *Automatica*, vol. 30, pp. 1283–1293, 2003.
[14] N. F. S.K. Agrawal and R. Murray, "Feasible trajectories of linear dynamic with inequality constraints using higher-order representations," in *Proc. IFAC World Congress*, Beijing,China, 1999.
[15] R. Murray, M. Rathinam, and W. Sluis, "Differential flatness of mechanical control systems," in *Proc. ASME Intern. Cong.and Exp.*
[16] R. Rothfuss, R. Rudolph, and M. Zeitz, "Flatness based control of a nonlinear chemical reactor model," *Automatica*, vol. 32, pp. 1433–1439, 1996.
[17] M. Krstić, I. Kanellakopoulos, and P. Kokotović, *Nonlinear and Adaptive Control Design*. New York: John Wiley and Sons, 1995.

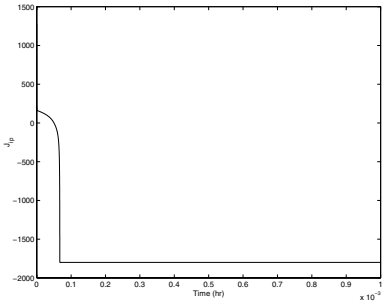

Fig. 1. Cost functional J_{ip} for the the real-time optimization scheme

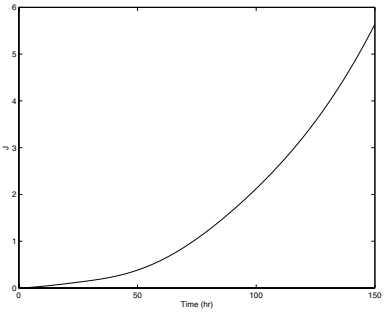

Fig. 2. Cost functional J for the the real-time optimization scheme

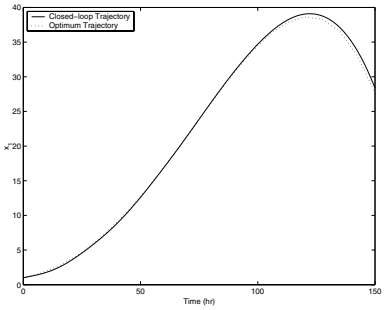

Fig. 3. Closed-loop state variable x_1 trajectories (full lines) and optimal state trajectories (dashed lines) for the real-time optimization scheme.

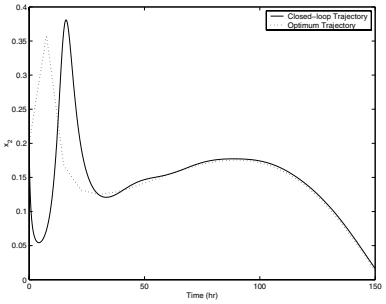

Fig. 4. Closed-loop state variable x_2 trajectories for the real-time optimization scheme.

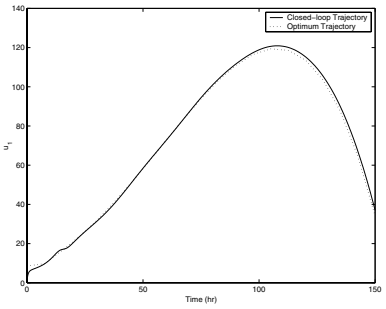

Fig. 5. Closed-loop input variable trajectories (full lines) and optimal input trajectories (dashed lines) for the the real-time optimization scheme