Calhoun: The NPS Institutional Archive
DSpace Repository

Department of Mechanical and Aerospace Engineering (MAE) Faculty and Researchers' Publications

2003-12-01

**A Unified Computational Framework for
Real-Time Optimal Control , IEEE (42nd; 2003;
Maui, Hawaii)**

Ross, I. Michael; Fahroo, Fariba

IEEE

<https://hdl.handle.net/10945/29683>

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

A Unified Computational Framework for Real-Time Optimal Control

I. Michael Ross

Department of Mechanical and Astronautical
Engineering, Code: AA/Ro
Naval Postgraduate School
Monterey, CA 93943
imross@nps.navy.mil

Fariba Fahroo

Department of Applied Mathematics
Code: MA/Ff
Naval Postgraduate School
Monterey, CA 93943
ffahroo@nps.navy.mil

Abstract—The dynamics of each agent of a multi-agent controlled dynamical system can be formulated in several possible ways: differential inclusion, flatness parameterization, higher-order inclusions and so on. A plethora of techniques have been proposed for each of these formulations but they are typically not portable across equivalent mathematical formulations. Further complications arise as a result of path constraints such as those imposed by obstacle avoidance or control saturation. In this paper, we present a unified computational framework based on pseudospectral methods to handle the optimal control of dynamical systems where the description of the governing equations or that of the path constraint is not a limitation. We illustrate our ideas by way of multiple formulations of a flexible link manipulator problem that includes a differentially flat formulation subject to control saturation. A comparison of our approach to a recent method reveals that we get an almost 30% improvement in the cost. Our results also show that equivalent mathematical formulations can yield varying run times leading to some surprising questions on flatness parameterization for real-time computation.

I. INTRODUCTION

Rapid technological advances in nano- and micro-electro-mechanical systems have made possible the concept of multi-agent heterogeneous intelligent systems. The prime driver for such systems is the military as it sees these technologies as force multipliers. Future warfare is more likely to be driven by a coordinated command and control of a system of unmanned ground vehicles, unmanned air vehicles, space systems and other heterogeneous agents. Common tasks to be performed are motion planning, obstacle avoidance, target tracking, search operations and other high-level decisions. For the proper functioning of these systems, the outer-loop must be executed in real-time in addition to achieving optimal performance imposed by stringent systems requirements. While optimal control theory provides a framework for solving many of these complex problems, it is widely recognized that solving these problems is extremely difficult [18]; consequently, alternative paths are explored based on some assumptions of concept of operations [6] or differential-geometric properties of the dynamical system [10]. Although such assumptions may either limit or not be viable across the board for heterogeneous systems, the motivation for real-time optimal control is that it is an enabling technology for solving a vast array of complex control problems.

In this paper, we present a unified framework for solving nonlinear optimal control problems. We propose to exploit

the differential-geometric properties possessed by the dynamical system and explore various coordinatizations using the same computational framework. A unified framework is not only mathematically elegant but is also quite necessary from a practical point of view. If each agent of a multi-agent system has a common software, complex decisions can be quickly reprogrammed across the board. In addition, a common software reduces the enormous cost associated with software verification and management. Our approach is based on pseudospectral (PS) methods [3], [4], [11], [14]. To illustrate our ideas, we consider a flexible link manipulator problem and numerically solve different formulations of the same problem. Numerical experiments reveal that although higher-order formulations are computationally faster, flatness parameterization does not necessarily provide the most efficient computational scheme despite the fact that it generates the smallest number of optimization variables.

II. FUNDAMENTAL IDEAS

A general optimal control problem can be formulated as follows [18]: Determine the function, $[\tau_0, \tau_f] \ni \tau \mapsto x \in \mathbb{R}^{N_x}$ and possibly the clock times τ_0 and τ_f , that minimize the Mayer cost functional,

$$J[x(\cdot), \tau_0, \tau_f] = E(x(\tau_0), x(\tau_f), \tau_0, \tau_f) \quad (1)$$

subject to the dynamic constraints,

$$\dot{x}(\tau) \in \mathbb{F}(x(\tau), \tau) \quad a.e. \tau \in [\tau_0, \tau_f] \quad (2)$$

and end-point constraints,

$$(x(\tau_0), x(\tau_f), \tau_0, \tau_f) \in \mathbb{E} \quad (3)$$

where $E: \mathbb{E} \rightarrow \mathbb{R}$ is a given function, $\mathbb{F}: \mathbb{R}^{N_x} \times \mathbb{R} \rightarrow \mathbb{R}^{N_x}$ is a given multifunction and $\mathbb{E} \subset \mathbb{R}^{N_x} \times \mathbb{R}^{N_x} \times \mathbb{R} \times \mathbb{R}$ is a given set. See [18] for some underlying assumptions. Frequently, we are concerned with dynamical systems parameterized by a control, $u \in \mathbb{R}^{N_u}$, in which case, the multifunction, $\mathbb{F}$, is given explicitly by,

$$\mathbb{F}(x, \tau) = \{v : v = f(x, u, \tau), u \in \mathbb{U}(x, \tau)\} \quad (4)$$

where $\mathbb{U}(x, \tau) \subset \mathbb{R}^{N_u}$ is some *state-dependent* control set. Barring the loss of information on a passage to an inclusion from a vector-field formulation, it is apparent that the two systems are mathematically equivalent. However, as will be evident shortly, these mathematically equivalent systems are

computationally very different. It stands to reason, though, that the differential inclusion framework may be computationally preferable as there are N_u fewer variables to compute and hence offers a potential increase in computational speed. Following this path, a reduction in the number of variables for a Lagrangian system

$$\frac{d}{dt} \frac{\partial L(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} - \frac{\partial L(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}} = \mathbf{u} \quad (5)$$

is possible by way of a second-order differential inclusion,

$$\frac{d}{dt} \frac{\partial L(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} - \frac{\partial L(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}} \in \mathbb{U}(\mathbf{q}, \dot{\mathbf{q}}, \tau) \quad (6)$$

A recent approach to exploiting the properties of Lagrangian systems is the method of controlled Lagrangians [2], but note that these methods are currently directed at stabilizing mechanical systems, not computing optimal control. Another concept that can be exploited for the purpose of computing optimal control is to change coordinates, $\mathbf{x} \mapsto \mathbf{w} \in \mathbb{R}^{N_x}$, to the normal form, [8]

$$\dot{w}_1 = w_2, \quad \dot{w}_2 = w_3, \quad \dots, \quad (7)$$

$$\dot{w}_{N_x} = \tilde{f}(\mathbf{w}, \mathbf{u}) \quad (8)$$

so that the dynamics, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, can be “scalarized” to,

$$w_1^{(N_x)} = \tilde{f}(\mathbf{w}, \mathbf{u}) \quad (9)$$

This scalarization is possible under certain conditions (e.g. control-affine systems of relative degree N_x) but it is apparent that we may be able to eliminate $N_x - 1$ variables *provided higher-order derivatives can be accounted for in an explicit manner*. As in the Lagrangian system, a still further reduction in variables is possible by a passage to a higher-order differential inclusion,

$$w_1^{(N_x)} \in \tilde{f}(\mathbf{w}, \mathbb{U}(\mathbf{x}(\mathbf{w}), \tau)) \quad (10)$$

where $\tilde{f}(\mathbf{w}, \mathbb{U}(\mathbf{x}(\mathbf{w}), \tau))$ is a set-valued map obtained through a transformation akin to Eq.(4).

All the above methods to eliminate variables have been based on parameterizing a differential inclusion by a control variable. Another approach to parameterizing a dynamical system is by way of flat outputs. A controlled dynamical system, $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$, is differentially flat [5] if there exists a variable $\mathbf{y} \in \mathbb{R}^{N_u}$, called the flat output, and a function $\mathbf{c}(\cdot)$,

$$\mathbf{y} = \mathbf{c}(\mathbf{x}, \mathbf{u}, \dot{\mathbf{u}}, \dots, \mathbf{u}^{(\alpha)}) \quad (11)$$

such that

$$\mathbf{x} = \mathbf{a}(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\beta)}) \quad (12)$$

$$\mathbf{u} = \mathbf{b}(\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(\beta+1)}) \quad (13)$$

where α and β are finite positive integers that denote the number of derivatives of the respective variables. Evidently, smoothness, particularly of the control variable, is presumed. For a differentially flat system, the optimal control problem

reduces to a standard problem of the calculus of variations but one that involves higher-order derivatives [14]. The main computational advantage of a flat parameterization is the complete elimination of the dynamical constraints. However, in the presence of control constraints (particularly, state-dependent control constraints), flatness parameterization implies a path constraint on the flat output resulting from a complex transformation of the control region, $\mathbb{U}(\mathbf{x}, \tau)$. That is, the constraint, $\mathbf{u} \in \mathbb{U}(\mathbf{x}, \tau)$ transforms to,

$$\mathbf{z} \in \tilde{\mathbb{U}}(\tau) \quad (14)$$

where $\mathbf{z} = [\mathbf{y}, \dot{\mathbf{y}}, \dots, \mathbf{y}^{(s)}]^T$, $s = \beta + 1$ and $\tilde{\mathbb{U}}$ denotes the transformed control space. Thus, a purportedly unified computational framework which has the additional ability to exploit flatness parameterization must be capable of handling such constraints. As noted in the previous section, designing different computational methods for different dynamical systems, or alternative coordinatization of the same dynamical system, is an expensive proposition that we seek to avoid. A unified framework also requires that it takes into account a cost function that may be given in a generalized Bolza form, [9]

$$J[\mathbf{x}(\cdot), \mathbf{u}(\cdot), \tau_0, \tau_f] = E(\mathbf{x}(\tau_0), \mathbf{x}(\tau_f), \tau_0, \tau_f) + \int_{\tau_0}^{\tau_f} F(\mathbf{x}(\tau), \dot{\mathbf{x}}(\tau), \mathbf{u}(\tau), \tau) d\tau \quad (15)$$

where $F : \mathbb{R}^{N_x} \times \mathbb{R}^{N_x} \times \mathbb{R}^{N_u} \times \mathbb{R} \rightarrow \mathbb{R}$ is a given (measurable) function. Note the functional dependence of the running cost on the velocity variable. In higher-order formulations, the running cost will be functionally dependent on higher-order derivatives that should be explicitly accounted for in a unified computational framework. It is well-known that a Bolza problem can be reformulated as a Mayer problem by the process of introducing a new state variable x_{N_x+1} and the dynamic constraint,

$$\dot{x}_{N_x+1} = F(\mathbf{x}(\tau), \dot{\mathbf{x}}(\tau), \mathbf{u}(\tau), \tau) \quad (16)$$

However, there are significant computational differences between the two mathematically equivalent forms. In the first place, Eq.(16), introduces a new variable (and thus, a potential increase in computational time). Further, the new variable must satisfy an equality constraint over the entire interval $[\tau_0, \tau_f]$ which is a more stringent computational requirement than the approximation of an integral. Finally, integration is a smoothing process (preferred computationally) while differentiation is an anti-smoothing process. These problems get exacerbated if the problem has isoperimetric constraints. Thus, a good unified framework must take into account such efficiencies. We will now show how the pseudospectral method described below can accommodate all these notions in an efficient manner.

III. UNIFIED COMPUTATIONAL FRAMEWORK

Our unified framework is based on pseudospectral (PS) methods. Our ideas apply across all PS methods but we limit our discussion to Legendre PS methods for clarity. From a differential-geometric framework, PS methods are quite different from standard numerical methods (like Runge-Kutta) because they separate the accurate discretization of the tangent bundle from the vector field. Hence, they have wide applicability as evident from recent applications to systems governed by differential inclusions [4], differentially flat systems [14], higher-order systems [13] and standard state-space systems [15]. In addition, PS methods generally treat the approximation of integrals by Gaussian quadratures which facilitates an efficient computational framework.

In the Legendre PS method, the states and controls are approximated by N th order Lagrange polynomials which interpolate the functions at the Legendre-Gauss-Lobatto (LGL) points. The LGL nodes $t_l, l = 0, \dots, N$, are distributed on the interval $[-1, 1]$ and defined as:

$$t_0 = -1, \quad t_N = 1$$

and for $1 \leq l \leq N-1$, t_l are the zeros of $\dot{L}_N$, the derivative of the Legendre polynomial, L_N . The discretization process begins by approximating the continuous state and control variables, $\{x, u\}(t) \approx \{x^N, u^N\}(t)$ where

$$x^N(t) = \sum_{l=0}^N x_l \phi_l(t) \quad u^N(t) = \sum_{l=0}^N u_l \phi_l(t) \quad (17)$$

where, for $l = 0, 1, \dots, N$

$$\phi_l(t) = \frac{1}{N(N+1)L_N(t_l)} \frac{(t^2-1)\dot{L}_N(t)}{t-t_l} = \begin{cases} 1 & \text{if } l = k \\ 0 & \text{if } l \neq k \end{cases} \quad (18)$$

are the Lagrange interpolating polynomials of order N , and $x_l = x^N(t_l)$, $u_l = u^N(t_l)$. The derivative terms are approximated from Eq.(17) by differentiating the approximation,

$$\dot{x}(t) \approx \dot{x}^N(t) = \sum_{l=0}^N x_l \dot{\phi}_l(t) \quad (19)$$

and then evaluating the expression at the LGL nodes. This process gives rise to the $(N+1) \times (N+1)$ differentiation matrix D_1 with entries $D_{1,kl} = \dot{\phi}_l(t_k)$ or

$$D_1 := [D_{1,kl}] := \begin{cases} \frac{L_N(t_k)}{L_N(t_l)} \cdot \frac{1}{t_k - t_l} & k \neq l \\ -\frac{N(N+1)}{4} & k = l = 0 \\ \frac{N(N+1)}{4} & k = l = N \\ 0 & \text{otherwise} \end{cases} \quad (20)$$

which operates over the discretization, $X = (x_0, x_1, \dots, x_N)$, to generate a discrete derivative $\dot{X} = D_1 * X = [X][D_1]^T = (\dot{x}_0, \dot{x}_1, \dots, \dot{x}_N)$. In

general, commuting the operations of differentiation and approximation is not desirable due to the large errors resulting from the Runge phenomenon [16]; however, PS methods offer an accurate way to commute these operations by way of choosing nodes based on minimal error norms. In fact, it can be shown that PS methods offer exponential convergence (with respect to the number of nodes) to the derivative of a smooth function [16]. Letting $D_1 \equiv D$, the higher order derivatives can be obtained from matrix powers: $D_i = D^i$. Thus, D_2 is obtained by simply squaring D_1 , while $D_3 = D^3$ and so on. Discretization of the different dynamical constraints is based on imposing the constraints only at the LGL nodes. In this manner, the functions are replaced by a vector of their values at the points and derivative operators are replaced by differentiation matrices. Therefore, discretization of the differential inclusion can be written as,

$$\begin{aligned} \dot{x}(\tau_i) \simeq \dot{x}^N(\tau_i) &= \sum_{j=0}^N D_{1,ij} x_j \in \frac{\tau_f - \tau_0}{2} \mathbb{F}(x_i, \tau_i) \\ i &= 0, 1, \dots, N \end{aligned} \quad (21)$$

where the factor $\frac{\tau_f - \tau_0}{2}$ comes from an affine transformation of the time-domain from $[\tau_0, \tau_f]$ to $[-1, 1]$, and τ_i are the shifted LGL nodes. Similarly, the state-space constraint

$$\dot{x}(\tau) = f(x(\tau), u(\tau), \tau)$$

is approximated by

$$\dot{x}^N(t_k) = \sum_{l=0}^N x_l \dot{\phi}_l(t_k) = \sum_{l=0}^N D_{kl} x_l = \frac{\tau_f - \tau_0}{2} f(x_k, u_k, \tau_k) \quad (22)$$

while a general higher order differential constraint of the form

$$\tilde{f}(x, \dot{x}, \dots, x^{(n)}, u) = 0 \quad (23)$$

can be discretized as

$$\tilde{f}(X, D_1 * X, \dots, D_n * X, U) = 0 \quad (24)$$

where $U = (u_0, u_1, \dots, u_N)$. It is thus, apparent that a flat output and its derivatives can also be easily discretized as [14],

$$Y_i = [y_0, y_1, \dots, y_N][D^i]^T \quad (25)$$

where Y_i is an equivalent representation of the vector-valued polynomials, $y^{(i)}$, $i = 1 \dots, s$. As noted in [14], for a flat system, this representation need not be an approximation.

For discretizing the generalized Bolza cost function, Eq.(15), the Gauss-Lobatto integration rule yields

$$\begin{aligned} J^N[X, U, \tau_0, \tau_f] &= E(x_0, x_N, \tau_0, \tau_f) \\ &+ \frac{\tau_f - \tau_0}{2} \sum_{k=0}^N F(x_k, \sum_{j=0}^N D_{1,kj} x_j, u_k, \tau_k) w_k \end{aligned} \quad (26)$$

where w_k are the LGL weights given by

$$w_k := \frac{2}{N(N+1)} \frac{1}{[L_N(t_k)]^2} \quad k = 0, 1, \dots, N.$$

This formulation can easily be adapted to the cases where the cost functional is a function of higher-order derivatives of the states or outputs as well (as in the case of differentially flat system). In these cases, the derivatives of the functions are replaced by discrete approximations resulting from a repetitive use of the differentiation matrix to generate derivatives of higher order. The end result is that the optimal control problem for all of the different formulations discussed in the previous section can be easily and accurately approximated to a nonlinear programming (NLP) problem. One important aspect of the NLP obtained by PS methods is that it preserves the structure of the original optimal control problem which is of significant consequence to the dualization of the problem [11] and convergence of the discretization [12].

All these ideas are further illustrated in the next section by way of a benchmark nonlinear control problem where six different formulations are considered.

IV. EXAMPLE PROBLEM

In order to demonstrate our ideas in a concise manner, we choose the simple one-link flexible robot arm discussed in [1], [8], [17]. Although this system is static feedback linearizable and is hence trivially differentially flat, we include this formulation among the array of possibilities mostly to demonstrate that the transformation of the control constraint to the flat space can be easily handled by our PS method.

A. Lagrangian Formulation

From the Lagrangian of this system, the equations of motion are,

$$I_1 \ddot{q}_1 + m_1 g l \sin q_1 + k(q_1 - q_2) = 0 \quad (27)$$

$$I_2 \ddot{q}_2 - k(q_1 - q_2) = u \quad (28)$$

where $I_1 = I_2 = 1.0$, $k = 1.0$, $g = 9.8$, $m_1 = 0.01$, $l = 0.5$. The optimal control problem is to minimize,

$$J = \int_{\tau_0}^{\tau_f} u^2 d\tau \quad (29)$$

while steering the system from the initial configuration $[q_1(\tau_0), q_2(\tau_0), \dot{q}_1(\tau_0), \dot{q}_2(\tau_0)] = [0.03, 0.01, 0.04, 0.05]$ to the final state $[q_1(\tau_f), q_2(\tau_f), \dot{q}_1(\tau_f), \dot{q}_2(\tau_f)] = [0.06, 0.02, 0.08, 0.02]$. The constraint on the control is $-15 \leq u \leq 15$. Discretization of this problem by the PS method yields an NLP variable of dimension $3(N+1)$ where N is the degree of approximation and $N+1$ is the number of nodes.

B. Lagrangian Inclusion

In this formulation the controls are eliminated from the Euler-Lagrange formulation resulting in Eq.(27), the second-order differential inclusion,

$$-15 \leq I_2 \ddot{q}_2 - k(q_1 - q_2) \leq 15 \quad (30)$$

and a cost function formulation given by,

$$J = \int_{\tau_0}^{\tau_f} (I_2 \ddot{q}_2 - k(q_1 - q_2))^2 d\tau \quad (31)$$

Therefore, the resulting NLP variable has dimension $2(N+1)$.

C. State-Space Formulation

The generalized coordinates, q_1 and q_2 , of the Lagrangian formulation is easily converted to the state-space formulation in terms of four state variables $x_1 = q_1, x_2 = \dot{q}_1, x_3 = q_2, x_4 = \dot{q}_2$:

$$\dot{x}_1 = x_3 \quad \dot{x}_2 = x_4 \quad (32)$$

$$\dot{x}_3 = \frac{m_1 g l \sin x_1 + k(x_1 - x_2)}{I_1} \quad (33)$$

$$\dot{x}_4 = \frac{k(x_1 - x_2) + u}{I_2} \quad (34)$$

The related NLP variable is of dimension $5(N+1)$, and is the largest among these formulations.

D. Differential Inclusion

In this formulation, we use the bounds on u and Eq.(34) to rewrite the last dynamic constraint as an inequality. So the differential constraints become Eqs.(32)-(33) and,

$$-15 \leq I_2 \dot{x}_4 - k(x_1 - x_2) \leq 15 \quad (35)$$

whereas the cost function is written as,

$$J = \int_{\tau_0}^{\tau_f} (I_2 \dot{x}_4 - k(x_1 - x_2))^2 d\tau \quad (36)$$

As in the case of the Lagrangian Inclusion the control variable is eliminated from this formulation resulting in an NLP variable of dimension $4(N+1)$.

E. Normal Form Formulation

Since this system has relative degree 4, the dynamics can be scalarized to a fourth-order system. This formulation of the state dynamics can be easily achieved by solving for q_2 from Eq.(27) in terms of q_1 and $\dot{q}_1$ and substituting the expression in Eq.(28). We obtain a 4th-order differential equation in terms of $x_1 \equiv w_1$

$$w_1^{(4)} = \beta_1 u + (\beta_2 \cos w_1 + \beta_3) \dot{w}_1 + \beta_4 \dot{w}_1^2 \sin w_1 + \beta_5 \sin w_1 \quad (37)$$

where the constants β_i are defined as [17] $\beta_1 = \frac{k}{I_1 I_2}$, $\beta_2 = \frac{-m_1 g l}{I_1}$, $\beta_3 = \frac{-k(I_1 + I_2)}{I_1 I_2}$, $\beta_4 = -\beta_2$, $\beta_5 = \frac{-m_1 k g l}{I_1 I_2}$. The size of the NLP variable for this formulation is $2(N+1)$.

F. Constrained Differentially-Flat Formulation

The system is static feedback linearizable and is hence trivially differentially flat with $y = x_1$ as the flat output. However, the control constraint imposes a constraint on the flat output so that the optimal control problem in flat space is path-constrained and can be formulated as finding the function $t \mapsto y$ that minimizes the cost function,

$$J = \frac{1}{\beta_1^2} \int_{\tau_0}^{\tau_f} (y^{(4)} - (\beta_2 \cos y + \beta_3) \ddot{y} - \beta_4 \dot{y}^2 \sin y - \beta_5 \sin y)^2 d\tau \quad (38)$$

subject to the constraint

$$\frac{1}{\beta_1} |y^{(4)} - (\beta_2 \cos y + \beta_3) \ddot{y} - \beta_4 \dot{y}^2 \sin y - \beta_5 \sin y| \leq 15 \quad (39)$$

Additional constraints on the problem are the transformed nontrivial end-point constraints. The size of the NLP variable for this formulation is $N+1$ and is the smallest among these formulations.

V. NUMERICAL RESULTS

The various NLPs discussed in the previous section were solved using SNOPT [7]. Figure 1 compares the control

Fig. 1. Control, u , by various methods.

profile obtained by our methods with that of [17]. Our cost of approximately 109 is nearly 30% less than the cost of approximately 154 obtained in [17]. The value of this cost is not reported in [17], but was obtained by us by computations of the piecewise constant control assumed in [17]. For the purpose of completeness, the profiles of the generalized coordinates, q_1 and q_2 , are shown in Fig. 2.

The plots displayed in Figs. 1 and 2 are representative of the converged results obtained from a few hundred random runs where the random parameters were the initial guesses required for the sequential quadratic programming (SQP) method of SNOPT. The purpose of the random runs was to investigate the numerical performances of the various formulations in an equal setting. Table I summarizes the results for 32 nodes. The different formulations are ordered

Fig. 2. Optimized q_1 and q_2 .

in the decreasing number of NLP variables. The run time (see the fourth column) bears no correlation to the size of the NLP. The fastest run time was the Normal Form formulation with Euler-Lagrange being a close second. The various formulations were coded in MATLAB on a Pentium 4, 2.4 GHz PC with 512MB of RAM. The gradient and Jacobians required for the SQP method were computed within SNOPT by automatic finite differencing. It is possible to reduce the run times for all formulations by using analytic Jacobians. A conservative estimate [15] in the reduction in run-time by the use of analytic Jacobians is a factor of 5. Thus, we expect the Normal Form formulation to run in about 0.06 seconds or about 17 Hz, and obviously much faster on a dedicated processor comparable to the Pentium 4 and/or faster machines using hardware and software intended for real-time systems. In any case, the last two columns of Table I are quite interesting. The average run times for the 100 randomized initial guesses roughly follows the same trend as that for the linear guess, albeit with different run times. Convergence was declared if the cost was approximately near 109 (column 3) and SNOPT returned an "inform"

Method	NLP Vars	Cost ¹	Run Time ¹ (sec)	Avg. Run Time ² (sec)	% Conver- gence ⁴
State Space	160	109.6	0.99	1.40	100
Diff Incl	128	109.6	1.77	2.36	100
Euler-Lagrange	96	109.6	0.45	0.82	100
Lagrangian Incl	64	109.6	1.02	0.99	100
Normal Form	64	109.6	0.30	0.81	100
Diff-Flat	32	109.7	17.03	15.97 ³	89 ³

TABLE I

NUMERICAL PERFORMANCE OF VARIOUS FORMULATIONS.

¹ From linear guess. ² For 100 random runs averaged over converged solutions using default tolerances. ³ No converged solutions were obtained for default tolerances; optimality tolerance had to be loosened from a default value of 10^{-6} to 10^{-5} and iteration limits increased to obtain convergence. ⁴ See text for a discussion of convergence criteria.

value [7] of zero. The only exception was the differentially flat formulation. In this case, the optimality tolerance had to be increased in addition to increasing the iteration limits. These observations are roughly in accordance with our prior work on flatness parameterization [14] and the independent conclusions of [17].

VI. CONCLUSIONS AND FURTHER WORK

Advancements in nonlinear control theory can be quickly exploited by PS methods. One central reason for the versatility of the PS methods is purely geometric: PS methods separate the discretization of the tangent bundle from that of the vector field, and the accuracy of the approximation of the former is largely independent of the latter. Furthermore higher-order derivatives can be easily obtained by way of elementary matrix multiplications. Hence the relative degree of a dynamical system can be easily exploited. For similar reasons, differentially flat systems with control constraints or even state-dependent control constraints can be easily handled. Although flatness parameterization appears to be attractive in offering the least number of variables, numerical experiments reveal that it is not necessarily the most computationally efficient approach for real-time optimal control. Apparently, reduction in the number of optimization variables and constraints does not necessarily lead to faster computations. There is a significant interplay at the junction where differential-geometric concepts are exploited for computational advantage and the numerical properties of the resulting optimization problem. This is an area that promises significant breakthroughs in control theory as advances in large-scale numerical methods remain heretofore unexplored within a differential-geometric setting. A number of these ideas will be explored in future papers, but it is evident that a combination of PS methods with recent advances in nonlinear control theory and optimization can be combined to effectively solve some challenging control problems.

VII. ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for many valuable comments that helped the revision of this paper.

VIII. REFERENCES

- [1] Agrawal, S. K., and N. Faiz, "A new efficient method for optimization of a class of nonlinear systems without Lagrange multipliers," *JOTA*, Vol 97, No. 1, 1998, pp. 11-28.
- [2] Bloch, A. M., N. E. Leonard and J. E. Marsden, "Controlled Lagrangians and the stabilization of mechanical systems II: Potential shaping," *IEEE TAC*, Vol.46, No.10, 2001, pp. 1556-1571.
- [3] Elnagar, J., M. A. Kazemi and M. Razzaghi, "The pseudospectral Legendre method for discretizing optimal control problems," *IEEE TAC*, Vol. 40, No. 10, 1995, pp. 1793-1796.
- [4] Fahroo, F., and I. M. Ross, "Second look at approximating differential inclusions," *AIAA JGCD*, Vol. 24, No. 1, 2001, pp. 131-133.
- [5] Fliess, M., J. Lévine, P. Martin and P. Rouchon, "Flatness and defect of nonlinear systems: Introductory theory and examples," *Inter. J. Contr.*, Vol. 61, No. 6, 1995, pp. 1327-1361.
- [6] Frazzoli, E., M.A. Dahleh, and E. Feron, "Real-time motion planning for agile autonomous vehicles," *AIAA JGCD*, Vol. 25, No. 1, 2002, pp. 116-129.
- [7] Gill, P.E., W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM J. of Opt.*, Vol.12, No. 4, 2002, pp. 979-1006.
- [8] Isidori, A., *Nonlinear Control Systems*, Springer-Verlag, New York, NY, 1989.
- [9] Lowen, P. D. and R. T. Rockafellar, "New necessary conditions for the generalized problem of Bolza," *SIAM J. of Control & Opt.*, Vol.34, 1996, pp. 1496-1511.
- [10] Milam, M. B., K. Mushambi and R. M. Murray, "A new computational approach to real-time trajectory generation for constrained mechanical systems," *Proc. IEEE CDC*, Sydney, Australia, Dec. 2000.
- [11] Ross, I. M., and F. Fahroo, "A pseudospectral transformation of the covectors of optimal control systems," *Proc. IFAC-Symp. System, Structure and Control*, Prague, Aug. 2001.
- [12] Ross, I. M., and F. Fahroo, "Convergence of pseudospectral discretizations of optimal control problems," *Proc. IEEE CDC*, Orlando, FL, Dec. 2001.
- [13] Ross, I. M., J. Rea and F. Fahroo, "Exploiting higher-order derivatives in computational optimal control," *Proc. Med. Conf. Control and Automation*, Lisbon, Portugal, July 2002.
- [14] Ross, I. M., and F. Fahroo, "Pseudospectral methods for optimal motion planning of differentially flat systems," *Proc. IEEE CDC*, Las Vegas, NV, Dec. 2002.
- [15] Strizzi, J., Ross, I. M and Fahroo, F., "Towards real-time computation of optimal controls for nonlinear systems," *Proc. AIAA GNC Conf.*, Monterey, CA, Aug. 2002, AIAA Paper No. 2002-4945.
- [16] Trefethen, L. N., *Spectral Methods in MATLAB*, SIAM, Philadelphia, PA, 2000.
- [17] Veeraklaew, T., and S. K. Agrawal, "New computational framework for trajectory optimization of higher-order dynamic systems," *AIAA JGCD*, Vol. 24, No. 2, 2001, pp. 228-236.
- [18] Vinter, R. B., *Optimal Control*, Birkhäuser, Boston, MA, 2000.