Available online at www.sciencedirect.com
ScienceDirect
Automatica 43 (2007) 1265–1273
automatica
www.elsevier.com/locate/automatica

Brief paper

A bio-inspired pursuit strategy for optimal control with partially constrained final state ☆

D. Hristu-Varsakelis[a,*], C. Shao[b,1]

[a]Department of Applied Informatics, University of Macedonia, Thessaloniki 54006, Greece
[b]SAC Capital Advisor, LLC 540 Madison Avenue, New York, NY 10022, USA

Received 21 May 2005; received in revised form 20 September 2006; accepted 20 December 2006
Available online17 May 2007

Abstract

We discuss a biologically inspired cooperative control strategy which allows a group of autonomous systems to solve optimal control problems with free final time and partially constrained final state. The proposed strategy, termed “generalized sampled local pursuit” (GSLP), mimics the way in which ants optimize their foraging trails, and guides the group toward an optimal solution, starting from an initial feasible trajectory. Under GSLP, an optimal control problem is solved in many “short” segments, which are constructed by group members interacting locally with lower information, communication and storage requirements compared to when the problem is solved all at once. We include a series of simulations that illustrate our approach.

Keywords: Cooperative control; Agents; Optimization; Nonlinear systems; Optimal control; Autonomous swarms

1. Introduction

Biological systems have long been a source of inspiration for science and engineering, partly because of a desire to understand nature in the simplest possible terms, and because the successes of natural systems can often elucidate difficult problems in a wide variety of fields. In the area of systems and control in particular, the recent development of less-expensive small-scale air, water and ground-based autonomous vehicles has fueled interest in the study of animal groups, whose behavior demonstrates quite elegantly (Parrish & Hammer, 1997) how individuals' shortcomings (e.g., limited intelligence, memory and sensing range) can in many cases be overcome by effective cooperation. Examples include flocks of birds and schools of fish which move in tight formations and respond to predators almost as a single organism, worker bees which share information by “dancing” and distribute themselves optimally among nectar sources, and ants which use pheromone secretions to optimize their foraging trails (Camazine et al., 2001), to name a few.

Observations of natural groups suggest that self-organization may stem from a few simple rules of interaction, and that minor changes in those rules can result in a wide range of patterns. This idea has influenced the development of mathematical models linking individual with group behavior, including explanations of collective animal movement (Camazine et al., 2001; Jadbabaie, Lin, & Morse, 2003); it has also seeded a variety of research in cooperative approaches to robotics (Ögren, Fiorelli, & Leonard, 2002; Yamaguchi & Burdick, 1998), distributed covering and searching (Passino et al., 2002; Wagner, Lindenbaum, & Bruckstein, 1999), remote exploration (Brooks & Flynn, 1989), estimation (Kurazume & Hirose, 1998; Roumeliotis & Bekey, 2002) and optimization (Bruckstein, 1993; Dorigo, Maniezzo, & Colorni, 1996).

This work is a continuation of recent efforts (Hristu-Varsakelis & Shao, 2004, 2005; Shao & Hristu-Varsakelis,

☆ This paper was not presented at any IFAC meeting. This paper was recommended for publication in revised form by Associated Editor Kok Lay Teo under the direction of Editor Ian Petersen.

* Corresponding author. Tel.: +30 2310 891721; fax: +30 2310 891 290.
E-mail addresses: dcv@uom.gr (D. Hristu-Varsakelis), cshao@glue.umd.edu (C. Shao).

[1] Part of this work was done while the second author was with the Department of Mechanical Engineering, Institute for Systems Research, University of Maryland, College Park, MD, USA.

0005-1098/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.automatica.2006.12.023

2005b) to formulate cooperative optimal control strategies based on simple models of ant movement. By cooperating in large numbers, ants are able to find efficient (short) paths between their nest and a food source in complex environments, a task that appears too difficult for any individual to accomplish. It has been known that one way to model this process is to postulate that ants "pursue" one another (e.g., each pointing its velocity vector toward its predecessor in $\mathbb{R}^2$), thus producing progressively "straighter" trails. That idea, termed "local pursuit", was introduced by Bruckstein (1993) and was later extended (Hristu-Varsakelis & Shao, 2004, 2005) to cover nontrivial dynamics and environment geometries.

The contribution of this paper is to present a bio-inspired cooperative optimization strategy, of which the algorithms in Bruckstein (1993) and Hristu-Varsakelis and Shao (2004) are special cases, and to demonstrate its use in a series of challenging numerical optimal control problems. The focus on bio-inspiration is central to our approach, which attempts to demonstrate how a simple pursuit rule—when properly expressed—can lead to (locally) optimizing behavior in a group of engineered systems, in a way that mimics their natural counterparts. Compared to Bruckstein (1993) and Hristu-Varsakelis and Shao (2004), our algorithm addresses a larger and more useful class of optimal control problems with free final times and partially constrained final states, thus showing that the pursuit-based optimization works in a much broader setting than initially conceived.

In the next section we describe the optimal control problem to be addressed. Section 3 discusses an iterative optimization algorithm that is well suited to groups of cooperating dynamical systems. Section 4 contains the main results regarding the group's trajectories when its members evolve under the proposed strategy, followed by some key points concerning the algorithm's application. Section 5 tests our approach in a series of benchmark numerical optimal control examples.

2. Problem statement and notation

We are interested in the solution of optimal control problems using a group of cooperating "agents", where the term "agent" refers to a copy of a control system:

$$\dot{x}_k = f(x_k, u_k), \quad x_k(t) \in \mathbb{R}^n, \quad u_k(t) \in \Omega \subset \mathbb{R}^m \tag{1}$$

for $k = 0, 1, 2, \ldots$. Physically, each instance of (1) could stand for a robot, UAV, or other autonomous system.

Let x denote the state of a particular agent of (1). The problem under consideration is:

Problem 1. *Find a control $u^* \in \Omega$ and a final time Γ^* that minimize*

$$J(u, x, \Gamma) = \int_{t_0}^{t_0+\Gamma} g(x, u)\,\mathrm{d}t + G(x(t_0+\Gamma)) \tag{2}$$

with $x(t_0) = x_{\mathrm{I}}$ given, and subject to the dynamics (1) *and the final state constraint $Q(x(t_0+\Gamma)) = 0$, where $Q(\cdot) : \mathbb{R}^n \to \mathbb{R}^q$ is an algebraic function of the state. We will assume that $g(x(t), u(t)) \geqslant 0$, G is bounded below, and that g and G are otherwise such that Problem* 1 *has a solution* (*i.e.*, *the minimum exists*).[2]

Problem 1 involves optimal control with free final time and partially constrained final state. As the problem states, we are optimizing over the *control input*, u. Nevertheless, in the sequel it will sometimes be convenient to discuss the solution in terms of the optimal *trajectories* of (1). Of course, given an initial condition $x(t_0)$, the trajectory $x(t)$ of (1) is uniquely determined by u.

Definition 1. Given the final state constraint $Q(x) = 0$, the *constraint set* of x is $S_Q \triangleq \{x \,|\, Q(x) = 0\}$.

We will take $\partial Q/\partial x$ to have constant non-zero rank in a neighborhood of the set S_Q, in order to ensure that first-order necessary conditions for optimality are satisfied at the optimal termination point on Q.

In the sequel we will make use of the following notation. For any pair of states $a, b \in \mathbb{D} \subset \mathbb{R}^n$, let $x^*(t)$ be the optimal trajectory of (1) from a to b with free final time. We will write $\Gamma^*(a, b)$ for the corresponding optimal final time. The cost of following x^* will be denoted by

$$\begin{aligned}\eta(a, b) &\triangleq \int_{t_0}^{t_0+\Gamma^*} g(x^*, u^*)\,\mathrm{d}t + G(x^*(t_0+\Gamma^*)) \\ &= \min_{u,\Gamma} J(u, x, \Gamma),\end{aligned} \tag{3}$$

where the minimum is taken subject to (1), with $x(t_0) = a$, $x(t_0+\Gamma) = b$.

Let $x^*(t)$ be the optimal trajectory from an initial state a to the constraint set S_Q, and let $\Gamma^*_Q(a, S_Q)$ be the corresponding optimal final time from a to S_Q. The cost of following x^* will be denoted by

$$\begin{aligned}\eta_Q(a) &\triangleq \int_{t_0}^{t_0+\Gamma^*_Q} g(x^*, u^*)\,\mathrm{d}t + G(x^*(t_0+\Gamma^*_Q)) \\ &= \min_{u,\Gamma_Q} J(u, x, \Gamma_Q),\end{aligned} \tag{4}$$

subject to $x(t_0) = a$, $Q(x(t_0+\Gamma_Q)) = 0$. We note that because the agent dynamics have no explicit dependence on time, $\eta(\cdot,\cdot)$, $\eta_Q(\cdot)$ and $J(\cdot,\cdot,\cdot)$, are independent of t_0.

The cost of following a trajectory $x(t)$ of (1) generated by a generic input u during $[t', t'+\sigma]$ will be denoted by

$$C(x, t', \sigma) \triangleq \int_{t'}^{t'+\sigma} g(x, u)\,\mathrm{d}t \tag{5}$$

with $x(t)$, $u(t)$ defined on an interval which contains $[t', t'+\sigma]$. The following can be derived easily from the properties of optimal trajectories and will be helpful in the sequel:

[2] We have assumed that $g \geqslant 0$ in order to avoid some obvious degeneracies; e.g., $g < 0$ on some region S and an agent can repeatedly move within S to decrease J without bound.

Fact 1. *Let η, η_Q and C as defined in* (3), (4) *and* (5) *and let x be a trajectory of* (1) *generated by u. Then*,

(1) $\eta(a,b) \leqslant C(x,t_0,\Gamma)$ *for any $x(\cdot)$ such that $x(t_0)=a$, and $x(t_0+\Gamma)=b$.*
(2) $\eta(a,c) \leqslant \eta(a,b)+\eta(b,c)$.
(3) $\eta_Q(a) \leqslant \eta(a,b)$ *for any $b \in S_Q$.*

3. A bio-inspired algorithm for optimal control

Assume that there is available an initial feasible (but suboptimal) control/trajectory pair $(u_{\text{feas}}(t), x_{\text{feas}}(t))$ for (1), obtained through a combination of a priori knowledge about the problem and/or random exploration. Following the idea in Bruckstein (1993) and Hristu-Varsakelis and Shao (2004), the agents x_k will leave the initial state x_{I} sequentially and pursue one another toward the target set S_Q, in a way which will be made precise shortly. The first agent follows x_{feas} to reach some point in S_Q. Each subsequent agent attempts to intercept its predecessor—along optimal trajectories defined by (3)—as long as the predecessor has not reached the constraint set S_Q. If that has already occurred, then the pursuer is to ignore the preceding agent and instead evolve along the optimal trajectory defined by (4). We will use the term *generalized sampled local pursuit* (GSLP) to describe the preceding strategy, in order to distinguish it from the special case (SLP) in Hristu-Varsakelis and Shao (2004). More precisely:

Algorithm 1 (*GSLP*). *Let $x_0(t), t \in [0,T_0]$ be an initial trajectory satisfying* (1) *with $x_0(0)=x_{\text{I}}$, $Q(x_0(T_0))=0$. Choose Δ such that $0<\Delta<T_0$.*

(1) DO $k=1,2,3,\ldots,$
 Let $t_k=k\Delta$ be the starting time of the kth agent, i.e., $x_k(t_k)=x_{\text{I}}$.
(2) DO $i=0,1,2,3,\ldots,$
(3) *define $t_k^i = t_k^{i-1}+\delta_i$, $t_k^0=t_k$ where $0<\delta_i<\min(\Delta,\Gamma^*_{i-1})$, and Γ^*_{i-1} is the optimal time ((3), (4)) from $x_k(t_k^{i-1})$ to $x_{k-1}(t_k^{i-1})$ or to S_Q (with $\Gamma^*_{-1} \triangleq \Delta$).*
(4) *Let $u^*_{t_k^i}(\tau)$ be a control that achieves*

$$\begin{cases} \eta(x_k(t_k^i), x_{k-1}(t_k^i)), & x_{k-1}(t_k^i) \notin S_Q, \\ \eta_Q(x_k(t_k^i)), & x_{k-1}(t_k^i) \in S_Q, \end{cases}$$

subject to (1), *where $\tau \in$*

$$\begin{cases} [t_k^i, t_k^i+\Gamma^*(x_k(t_k^i), x_{k-1}(t_k^i))], & x_{k-1}(t_k^i) \notin S_Q, \\ [t_k^i, t_k^i+\Gamma^*_Q(x_k(t_k^i), S_Q)], & x_{k-1}(t_k^i) \in S_Q. \end{cases}$$

(5) *Apply $u_k(t)=u^*_{t_k^i}(t)$ to the kth agent for $t \in [t_k^i, t_k^{i+1})$.*
(6) UNTIL *the kth agent reaches S_Q.*
(7) UNTIL $C(x_k,t_k,t_k+\Gamma_k) - C(x_{k+1},t_{k+1},t_k+\Gamma_{k+1})<\varepsilon$, *for some fixed (small) ε, where Γ_k is the time it took for the kth agent to reach S_Q.*

We will refer to Δ as the *pursuit interval* and to δ as the *updating interval*. For fixed final time problems, local pursuit processes have a fixed final time as well, i.e., $\delta_i=\delta$. To simplify the discussion, we will assume that for free final time problems the Γ^*_i are lower bounded, so that we may again take δ to be constant.

When discussing pairs of agents during pursuit, the $(k-1)$st and kth agents will be termed "*leader*" and "*follower*", respectively. As Step 4 of the algorithm indicates, there are two types of follower movement, which could be termed loosely as "catching up" and "free running", depending on whether the leader has reached the final constraint set S_Q. The former lets agents "learn" from their leaders, while the "free running" stage enables them to optimize the final state within S_Q.

Regarding the controllability of (1), we will assume that the state solution, as well as the solution of the optimal control problem solved by each follower (Step 4 of GSLP), exist in an open region containing the leader and follower states and is reachable for $u \in \Omega$, starting from the follower's state, and that similar conditions hold for the overall solution from x_{I} to S_Q.

Finally, without loss of generality, we assume that followers do not intercept their leaders (in problems with free final time). If an interception does occur, one can simply prescribe that the follower "join" its leader by reproducing the leader's trajectory after the time of interception. Because Δ is finite, there will be a finite number of such events.

4. Main results

This section discusses the behavior of the group (1) under GSLP. Because of space limitations, we include only sketches of the proofs of the results that follow. The complete arguments, along with helpful illustrations, can be found in Shao and Hristu-Varsakelis (2005a). We begin by considering the sequence of trajectories $\{x_k(t)\}$ produced by GSLP; we will first discuss the convergence of the corresponding cost sequence, and then that of the trajectories themselves. In the subsequent discussion, it will be convenient to distinguish between the *planned trajectory*, denoted by $\hat{x}_{k,t_i}(\tau)$, that a follower computes at time t_i in order to reach its leader's state, and the *realized trajectory*, denoted by $x_k(t)$, along which the follower actually evolves, and which may differ from $\hat{x}_{k,t_i}(\tau)$ because of subsequent trajectory updates (in Steps 2–6 of GSLP).

Lemma 1. *Consider a leader–follower pair evolving under GSLP with pursuit interval Δ. Let the leader's trajectory be $x_{k-1}(t)$ $(t \in [t_{k-1}, t_{k-1}+T_{k-1}])$, where T_{k-1} is the time to reach S_Q, and fix $\lambda \in [0, T_{k-1})$. Suppose the follower updates its trajectory only once, as follows*:

- *If $\lambda < T_{k-1}-\Delta$, the follower moves along the optimal trajectory (in the sense of* (3)*) joining $x_k(t_k+\lambda)$ and $x_{k-1}(t_k+\lambda)$, with optimal final time $\Gamma=\Gamma^*(x_k(t_k+\lambda), x_{k-1}(t_k+\lambda))$. During other times, the follower reproduces the leader's trajectory, i.e.,*

$$x_k(t)=\begin{cases} x_{k-1}(t-\Delta), & t \in [t_k, t_k+\lambda], \\ x_{k-1}(t-\Gamma), & t \in [t_k+\lambda+\Gamma, t_k+T_k], \end{cases}$$

where T_k is the time it takes the follower to reach the final set S_Q.

- *If $\lambda \geqslant T_{k-1} - \Delta$, the follower chooses to evolve along the optimal trajectory (in the sense of (4)) from $x_k(t_k+\lambda)$ to the constraint set S_Q. During other times $x_k(t) = x_{k-1}(t-\Delta)$, $t \in [t_k, t_k+\lambda]$.*

Then the cost along the follower's trajectory will be no greater than the cost along the trajectory of its leader.

Proof. First, consider the case $\lambda < T_{k-1} - \Delta$. For $t \in [t_k+\lambda, t_k+\lambda+\Gamma]$, the follower moves on the locally optimal trajectory $x_k(t)$. The cost along x_k satisfies

$$\begin{aligned} &C(x_k, t_k, T_k) \\ &\quad = C(x_k, t_k, \lambda) + \eta(x_k(t_k+\lambda), x_{k-1}(t_k+\lambda)) \\ &\qquad + C(x_k, t_k+\lambda+\Gamma, T_k-\lambda-\Gamma) \\ &\quad \leqslant C(x_{k-1}, t_{k-1}, \lambda) + C(x_{k-1}, t_{k-1}+\lambda, \Delta) \\ &\qquad + C(x_{k-1}, t_{k-1}+\lambda+\Delta, T_{k-1}-\lambda-\Delta) \\ &\quad = C(x_{k-1}, t_{k-1}, T_{k-1}). \end{aligned} \tag{6}$$

If $\lambda \geqslant T_{k-1} - \Delta$, then the cost along x_k is

$$\begin{aligned} C(x_k, t_k, T_k) &= C(x_k, t_k, \lambda) + \eta_Q(x_k(t_k+\lambda)) \\ &\leqslant C(x_{k-1}, t_{k-1}, \lambda) + C(x_{k-1}, t_{k-1}+\lambda, T_{k-1}-\lambda) \\ &= C(x_{k-1}, t_{k-1}, T_{k-1}), \end{aligned}$$

which is equivalent to the desired statement. □

The next Lemma shows that the cost of the iterative trajectories converges under GSLP:

Lemma 2. *If the agents* (1) *evolve under GSLP, the cost of the iterated trajectories converges.*

Proof (*Sketch*). Let C_{k-1} be the cost along the leader's trajectory, $x_{k-1}(t)$ $(t \in [t_{k-1}, t_{k-1}+T_{k-1}])$. Let x_k^i and C_k^i, $i = 0, 1, 2, \ldots$, be the trajectory and corresponding cost for an agent that pursues x_k^{i-1} (with $x_k^0(t) = x_{k-1}(t)$) by performing only a *single trajectory update*, as described in Lemma 1, with $\lambda = (i-1)\delta$, $\delta > 0$. There will be a finite integer j such that C_k^j is the cost incurred by the follower x_k as he pursues the leader under GSLP. Using Lemma 1, $C_k = C_k^j \leqslant C_{k-1}$. Because the sequence $\{C_k\}$ is non-increasing and bounded below (there exists a minimum for (2)), it has a limit. □

Before stating the main result, we will require that small changes to the endpoints of a trajectory have a small effect on the optimal cost of (2).

Condition 1. $\forall a, b_1, b_2 \in \mathbb{D}, \Omega > 0$ *and for any $x_1(t)$, with $x_1(0) = a$, $x_1(T) = b_1$, $\exists \varepsilon > 0$ and $x_2(t)$, a trajectory of* (1) *with $x_2(0) = a$, $x_2(T) = b_2$, such that the costs of x_1 and x_2 satisfy*

$$\|b_1 - b_2\|_\infty < \varepsilon \Rightarrow \|C(x_1, 0, T) - C(x_2, 0, T)\|_\infty < \mathscr{L}\Omega$$

for some constant $\mathscr{L}$, independent of Ω.

Condition 1 implies that optimal trajectories of (1) which "overlap" (to be made precise below), are locally optimal:

Lemma 3. *Let $x^*(t)$ be a trajectory of* (1) *such that*:

(i) *$x^*(t)$ $(t \in [0, t_1+\Delta_1])$ is optimal (in the sense of* (3)) *from $x^*(0)$ to $x^*(t_1+\Delta_1)$, and*
(ii) *$x^*(t)$ $(t \in [t_1, T^*])$ is optimal (in the sense of* (4)) *from $x^*(t_1)$ to the constraint set S_Q. Assume also that Condition 1 is satisfied, and $0 < t_1 < t_1 + \Delta_1 < T^*$.*

Then, $x^*(t)$ $(t \in [0, T^*])$ is a local minimum of (4).

Proof (*Sketch*). In the special case where the condition for optimality can be expressed in terms of a differential equation (e.g., optimal trajectories which must satisfy the Euler–Lagrange equations), then $x^*(t)$ $(t \in [0, T^*])$ is a local minimum because it clearly must satisfy the same differential equation on $[0, T^*]$. For a proof of the general case (following the proof by contradiction in Hristu-Varsakelis & Shao (2004)), see Shao and Hristu-Varsakelis (2005a). □

The next two Lemmas discuss the convergence of the optimal trajectories.

Lemma 4. *If at all times during GSLP, the locally optimal trajectory from follower to leader (or to S_Q) is unique, then GSLP converges to a limiting trajectory $x_\infty(t)$.*

Proof (*Sketch*). Suppose that the trajectory costs converge but that there exist more than one limiting trajectory. Let $x_1(t)$, $t \in [0, T_1]$) and $x_2(t)$, $t \in [0, T_2]$) be two such possibilities. Let $t_1 \in [0, T_1]$ be the earliest time that $x_1(t)$ differs from $x_2(t)$. From Lemma 2, x_1 and x_2 must have the same cost, otherwise convergence of the cost is contradicted. Suppose now that a leader, $x_{k-1}(t)$, travels along $x_1(t)$, while its follower, $x_k(t)$, travels along $x_2(t)$. Choose $\delta > 0$ small, and suppose that the follower is to perform a series of updates to its trajectory, every δ time units. At some time at or after t_1 (suppose it is at t_1), the follower measures x_k and continues to evolve along $x_2(t)$, $t \in [t_1, t_1+\delta)$. This means that the trajectory composed of: (i) $x_2(t)$, $t \in [t_1, t_1+\delta)$ and (ii) the optimal trajectory from $x_2(t_1+\delta)$ to $x_1(t_1+\Delta)$, either has a lower cost than $x_1(t)$, $t \in [t_1, t_1+\Delta)$, or it has the same cost as $x_1(t)$, $t \in [t_1, t_1+\Delta)$. The first possibility (after considering the remaining updates to the follower's trajectory) contradicts the convergence of cost under GSLP, since it leads to the construction of a trajectory with lower cost than x_1 and x_2. The latter possibility contradicts the assumption that the locally optimal trajectory from follower to leader is unique. □

Lemma 5. *Let $\hat{x}_{k,t_i}(t)$, $t \in [t_i, t_i + \Gamma(t_i)]$, be the family of planned trajectories that the follower x_k computes at $t_i = k\Delta + i\delta$, $i = 0, 1, \ldots, i_{max}$, in order to reach $x_{k-1}(t)$ (or S_Q) optimally. If during GSLP: (i) the locally optimal trajectory from follower to leader (or to S_Q) is unique, and (ii) $x_{k-1} = x_\infty$ (Lemma 4), then $\hat{x}_{k,t_i}(t) = x_k(t)$, i.e., along the limiting trajectory produced under GSLP, the planned and realized trajectories overlap. Furthermore, if the locally optimal trajectories obtained at every updating time are smooth (differentiable), then the limiting trajectory is also smooth.*

Proof (*Sketch*). Suppose that a leader, x_{k-1} evolves along the limiting trajectory x_∞. Then (Lemma 4) $x_k(t + \Delta) = x_{k-1}(t)$ $\forall t \in [t_k, t_k + T_k]$. Suppose also that at some time, say t_1, the follower's planned trajectory is $\hat{x}_{k,t_1}(t)$, $t \in [t_1, t_1 + \hat{\Gamma}(t_1))$, and that starting at some time $t' \geqslant t_1$, $\hat{x}_{k,t_1}$ differs from the realized trajectory $x_k(t)$, $t \in [t_1, t_1 + \Gamma(t_1))$ (where $\hat{\Gamma}(t_1)$ and $\Gamma(t_1)$ are optimal, and $\hat{x}_{k,t_1}(t_1 + \hat{\Gamma}(t_1)) = x_k(t_1 + \Gamma(t_1))$). If $\hat{x}_{k,t_i}(t) \neq x_k(t)$, then we can either construct a new trajectory for the follower to take with lower cost than x_k (which contradicts the assumed convergence to a limiting cost), or the uniqueness of the optimal trajectories from follower to leader is violated.

The smoothness of the limiting trajectory follows from the fact that it is composed of overlapping segments which agree on intervals of size $\Delta - \delta > 0$ and are each smooth by assumption. □

The next theorem is follows from Lemmas 1 to 5:

Theorem 1. *Suppose that the group of agents* (1) *evolves under GSLP, that at all times t, the locally optimal trajectories from follower to leader are unique, and that Condition* 1 *holds. Then, the limiting trajectory, x_∞, is unique and locally optimal. It is also smooth, if the locally optimal trajectories calculated at every updating time are smooth.*

Proof. The uniqueness of x_∞ follows from Lemma 4. Also, if $x_{k-1}(t) = x_\infty(t - t_{k-1})$, then $x_{k-1}(t - \Delta) = x_k(t)$. Choose δ_1, δ_2 such that $0 < \delta_1 < \delta_2 < \Gamma$ for all optimal final times Γ of the planned trajectories $\hat{x}_k$ generated during GSLP. Then, x_∞ is piecewise smooth and locally optimal on $[t_k + i\delta_1, t_k + i\delta_1 + \delta_2]$, $i = 0, 1, 2 \ldots$, because it coincides with the planned trajectories, $\hat{x}_k(t)$, on those intervals. Because the intervals in question are overlapping, $x_k(t)$ (and thus x_∞) is optimal, as it is composed of overlapping locally optimal trajectories (using Lemma 3 with S_Q being a single point in the state space). Smoothness of x_∞ is proved in similar piecewise fashion. □

4.1. Remarks

The convergence results for GSLP assume an infinite sequence of agents. However, we have observed in our numerical experiments (including those to be described in Section 5) that in free final time problems, optimality is achieved in a finite number of iterations. See also Hristu-Varsakelis and Shao (2004) for a technique which "recycles" a finite collection of agents by reversing the roles of S_Q and x_{I}.

Because agents choose their controls based on local information, GSLP is not guaranteed to converge to the global optimum. The choice of agent separation Δ can affect whether the limiting trajectory is a local or a global optimum. Some interesting special cases involving spaces with "holes" or "obstacles" are discussed in Hristu-Varsakelis and Shao (2004), as is a version of GSLP where agents adjust their trajectories continuously ($\delta \to 0$).

In remote exploration settings, GSLP could be useful for a group of autonomous vehicles, because of its reduced information and sensing range requirements. Indeed, agents need not have access to the global geometry; each of them may operate using their own, ad hoc coordinate system which may be unknown to other members. In fact, agents may not even know the coordinates of the target state (or set), having only a set of instructions for reaching it (the initial feasible control). Under such constraints, solving the optimal control problem may not be possible for a single vehicle. With local pursuit on the other hand, it is sufficient for each vehicle to operate on its own local coordinate patch, calculating optimal trajectories from its own state to that of its nearby leader.

In general, there will not be analytical solutions available for the follower-to-leader optimal trajectories; thus, local pursuit requires an existing numerical optimization algorithm as a "subroutine", so that each agent can compute optimal trajectories to its leader (Step 2 of GSLP). In Section 5, we used the RIOTS (Chen & Schwartz, 2002) optimization software for this purpose, although other numerical optimization codes could be used just as easily. Although a discussion of GSLP as a numerical method is beyond the scope of this paper, it is worth noting that GSLP offers a trade-off between execution time and memory savings. For example, RIOTS is built upon the NPSOL optimization package (Gill, Murray, Saunderss, & Wright, 1998), which (as is typical for many numerical optimization methods), has an execution time and memory requirement which are quadratic ($O(n^2N^2)$) in the number of states n and in the number of knot points N used to sample the trajectory of (1). Thus, it may be advantageous to solve an optimal control problem in many small pieces (as GSLP does), as opposed to all at once, making it possible to solve larger problems with a given memory size. See Hristu-Varsakelis, Shao, and Samaras (2007) for additional details on the numerical performance of GSLP.

5. Simulations and experiments

In this section, we describe a series of simulations that illustrate the performance of local pursuit. All codes were written in MATLAB and ran on a 1.6 GHz Pentium-M PC.

5.1. Cooperative trail optimization

Consider the problem of finding shortest 3-D Euclidean paths in an environment consisting of a plane with two right cones (of radii 700 and 1000 units of length, centered at $(0, 0)$ and $(2600, 0)$, respectively); a partial top view is shown in Fig. 1. Each object (the plane and each cone) was parametrized with its own set of coordinate functions. A series of agents,

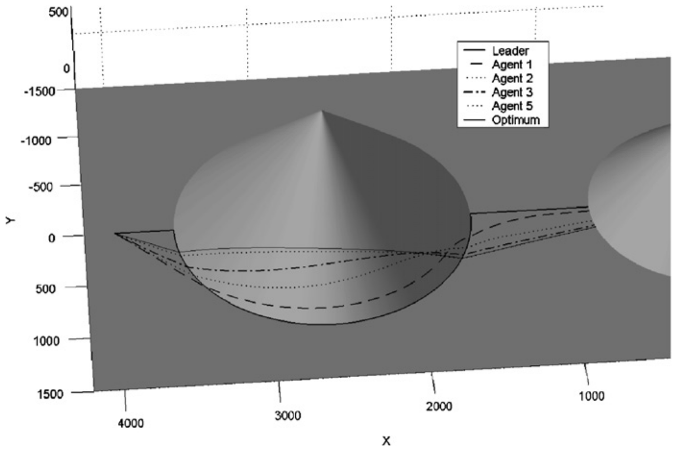

Fig. 1. Trajectories produced by GSLP (two-cone example), with $\delta = 1$, $\Delta = 868$, $T_0 = 4340$. The first agent proceeds along the plane-cone boundary. The target set S_Q was the boundary of the rightmost cone.

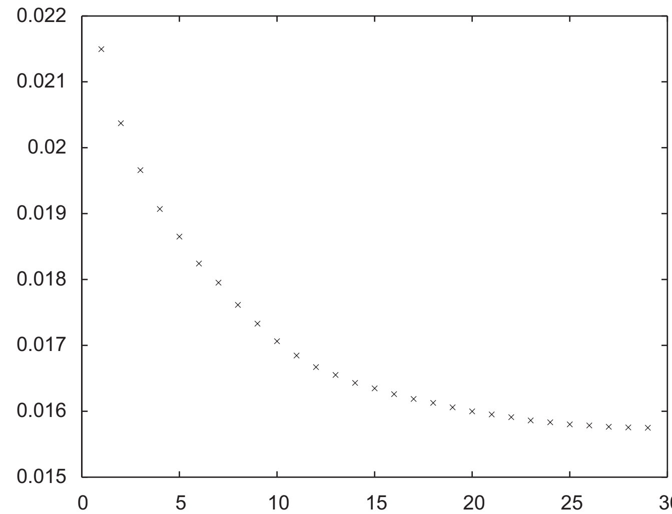

Fig. 2. Cost of each agent's trajectory (container crane).

governed by $\dot{x}_k = u_k$, $\|u_k\| = 1$, were required to travel from $x_I = (4000, 0, 0)$ to any point on the second cone, making this a problem with partially constrained final state. Fig. 1 shows the iterated trajectories generated by the agents under GSLP with $T_0 = 4340$, $\Delta = 0.2T_0$, $\delta = 1$. The lead agent's trajectory was restricted to the plane, following the boundary of the first cone. Starting from x_I, each agent advanced by solving a short-range optimal control problem with boundary conditions given by its own state and that of its leader (or the boundary of the second cone). With a termination threshold of $\varepsilon = 1$, GLSP completed in eight iterations. The trajectory of the eighth agent had a length of 3600.3 units, compared to 3600 for the minimum.

The use of local pursuit reduced the computational complexity of the problem, because the globally optimal trajectory crosses multiple coordinate patches from the plane to the cone(s) and vice versa. When a leader–follower pair were on the same patch, finding an analytic solution for the optimal trajectory was straightforward. In other cases, optimal trajectory segments crossed at most two coordinate patches, so that agents had to select from a one-parameter family of curves. On the other hand, computing the globally optimal trajectory all at once required searching a four-parameter family of curves (there are a total of four "crossings" between coordinate patches).

5.2. Optimal control of a container crane

Next, we simulated a series of agents whose dynamics were given by the container crane model from (Sakawa & Shindo, 1982; Teo, Jennings, Lee, & Rehbock, 1999):

$$\dot{x}_1 = x_4, \quad \dot{x}_2 = x_5, \quad \dot{x}_3 = x_6,$$

$$\dot{x}_4 = u_1 + 17.2656x_3, \quad \dot{x}_5 = u_2,$$

$$\dot{x}_6 = -(x_1 + 27.0756x_3 + 2x_5x_6)/x_2, \tag{7}$$

subject to $|u_1(t)| \leqslant 2.83374$, $-0.80865 \leqslant u_2(t) \leqslant 0.71265$, and state constraints $|x_4(t)| \leqslant 2.5$, $|x_5(t)| \leqslant 1.0$, where by slight abuse of notation we have used $x_1, \ldots, x_6$, to indicate the components of the state vector. The initial state was $x_I = [0, 22, 0, 0, -1, 0]^T$. The motion to be optimized was similar to that in Teo et al. (1999), where the crane is initially moving downwards and transitions to a state $x_f = [4, 19, 0, 2, 0, 0]$, at $t = 9$ s, where it has been translated and moving horizontally. The cost to be minimized was $J = \int_0^9 x_3^2 + x_6^2 \, dt$. To generate the initial feasible trajectory, the first agent moved optimally to an intermediate state $x_m = [0, 19, 0, 0, 0, 0]^T$ (a downward movement to a stop) at $t = 4.5$ s, and then optimally again to the desired final state, x_f, at $t = 9$ s, at a total cost of 0.02149. Subsequent agents applied GSLP with $\delta = 0.5$, $\Delta = 4$, in order to find the overall optimal control/trajectory pair from x_I to x_f. In this case the optimal control problem had no known analytical solution; each follower computed the optimal control to its leader numerically by calling RIOTS (Chen & Schwartz, 2002), and applied that control for δ time units, before repeating the procedure.

With a termination threshold of $\varepsilon = 10^{-6}$, GSLP performed 29 iterations. Fig. 2 shows the iterated costs. The trajectory of the 29th agent (Fig. 3) had a cost of 0.015719, which was within 10^{-5} of the optimum, 0.015714. The optimal trajectory and inputs agreed with those obtained using RIOTS to solve the problem "in one piece". During computation, each follower-to-leader trajectory segment was sampled at 20 knot points. Each iteration of GSLP (i.e., each agent's trajectory) took an average 450 s to compute, and required the corresponding agent to solve 10 versions of the optimal control problem with a time horizon of 3.5 s, using approximately 380 kb of memory. This should be compared with 193 s and approximately 1840 kb of memory which were required to compute the optimal trajectory all at once, using the same optimization routine and knot point density.

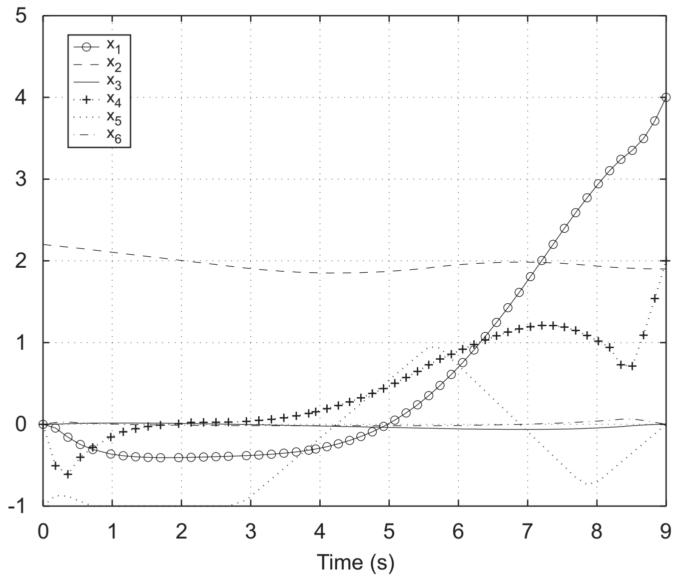

Fig. 3. Trajectory of the 29th agent in the container crane example. The second state has been scaled by (1/10).

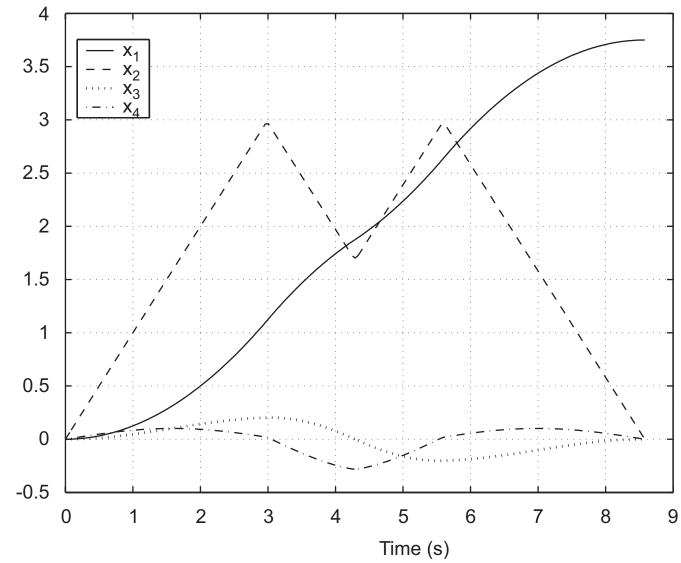

Fig. 4. Trajectory of the 10th agent (optimal) for the bridge crane example. The first state is scaled by (1/4).

5.3. Minimum-time control of a bridge crane

Next, we simulated a series of agents whose dynamics were given by the bridge crane system used in Luus (2000) and Moon and VanLandingham (1997)

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = u, \quad \dot{x}_3 = x_4,$$
$$\dot{x}_4 = -0.98x_3 + 0.1u, \tag{8}$$

with $x_I = 0$ and $|u(t)| \leqslant 1$. To generate an initial trajectory, we had the first agent proceed to the intermediate state $x_m = [5.0554, 0.1862, 0.1444, -0.0307]^T$ using the control $u(t) = \sin(2\pi t/4)$, $t \in [0, 7.5]$, and then on to the final state $x_f = [15, 0, 0, 0]^T$ optimally from x_m. Subsequent agents applied GSLP with $\delta = 1$ and $\Delta = 5$ in order to find the minimum-time trajectory from x_I to x_f. We transformed the minimum time problem into a partially constrained final state problem by defining a new state, x_5, with $\dot{x}_5 = 1$, and minimizing x_5 at the time when the state satisfies $[x_1, \ldots, x_4]^T = x_f$.

During pursuit, each follower called RIOTS to compute the minimum-time control to its leader, and applied that control for δ time units, before re-adjusting its trajectory. The initial trajectory reached x_f in 14.5748 s; subsequent agents arrived in 13.9376, 12.5651, 12.0216, 11.6930, 11.1640, 9.3250, 8.5663, and 8.5810 s. The trajectory of the 10th agent was optimal (8.5808 s) (Fig. 4). The optimal cost, trajectory and control, all agreed with those given by RIOTS when solving the problem in its entirety, as well as with the values reported in Luus (2000). RIOTS used 50 sampling points for each follower-to-leader trajectory segment. Each GSLP iteration took an average 9.81 s to complete, and used 128 kb of memory. This should be compared with 2.94 s and 488 kb of memory which were required to compute the optimal trajectory all at once using the same optimization routines and trajectory sampling density.

5.4. Minimum-time control of a quadruple integrator

Finally, we let the agent dynamics be given by the quadruple integrator (Chung & Wu, 1992, Luus, 2000):

$$\frac{d^4 z}{dt^4} = u, \quad |u(t)| \leqslant 1, \tag{9}$$

where $x \triangleq [z, dz/dt, d^2z/dt^2, d^3z/dt^3]^T$, and $x_I = [0.1, 0.2, 0.3, 0]^T$. We let the first agent evolve first to $x_m = [-1.7585, -2.4903, -0.9726, 0]^T$ using the control $u(t) = -\sin(2\pi t/4)/4$, $t \in [0, 8]$, and then optimally to the final state $x_f = 0$, in a total of 17.952 s. Subsequent agents applied GSLP ($\delta = 2$, $\Delta = 4$) in order to find the minimum-time trajectory from x_I to the origin. We used RIOTS to compute the minimum-time control from each follower to its leader, and applied that control for δ time units, before repeating the procedure. The second-through-fifth agents reached the target state in 13.1455, 12.4043, 6.6219, and 4.8352 s, respectively. The trajectory of the sixth agent was optimal, with a duration of 4.8320 s. The optimal time, trajectory (Fig. 5) and control (the well-known four-stage bang-bang policy), all agreed with those given by RIOTS when solving the problem in its entirety, as well as with the values reported in Luus (2000) (4.8319 s for the minimum time). Calls to RIOTS used 30 sampling points for each follower-to-leader trajectory segment. Each iteration of GSLP took an average of 3.05 s, and used 54 kb of memory, compared with 2.4 s and 608 kb of memory which were required to compute the optimal trajectory all at once using the same routines and sampling density.

6. Conclusions and ongoing work

We discussed a bio-inspired control strategy which enables a group of control systems to cooperatively find optimal

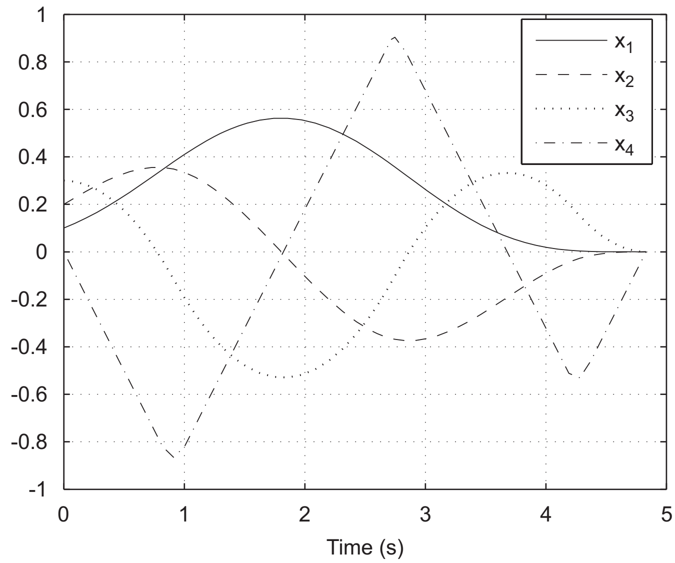

Fig. 5. Trajectory of the sixth agent (optimal) for the quadruple integrator example.

trajectories, starting from an initial feasible control. Our algorithm, mimics the way in which ants optimize their foraging trails and requires limited, short-range interactions between members of the group; it applies to systems with nonlinear dynamics and can be used to solve optimal control problems with free final time and partially constrained final state. The local nature of GSLP limits the amount of information (e.g., global geometry of the environment) that members must possess. Furthermore, it allows a potentially complex control problem to be solved in a series of smaller segments, trading off computation time for lower storage requirements. In each of the examples presented here, GSLP converged to the global optimum, although in general only local optimality is guaranteed. The algorithm's numerical performance and convergence rate, as well as a rigorous description of the class of optimal control problems for which GSLP converges in a finite number of iterations, are the subject of ongoing work.

Acknowledgments

The authors would like to thank Prof. Y. Chen and Dr. A. Schwartz for providing a copy of the RIOTS software, as well as Prof. R. Luus for helpful discussions on the examples of Section 5. The authors also thank the anonymous reviewers for their detailed and helpful feedback.

References

Brooks, R. A., & Flynn, A. M. (1989). Fast, cheap and out of control: A robot invasion of the solar system. *Journal of the British Interplanetary Society*, *42*, 478–485.

Bruckstein, A. M. (1993). Why the ant trails look so straight and nice. *The Mathematical Intelligencer*, *15*(2), 59–62.

Camazine, S., Deneubourg, J.-L., Franks, N. R., Sneyd, J., Theraulaz, G., & Bonabeau, E. (2001). *Self-organization in biological systems*. Princeton, NJ: Princeton University Press.

Chen, Y., & Schwartz, A. L. (2002). Riots95—-A Matlab toolbox for solving general optimal control problems and its applications to chemical processes. In R. Luus (Ed.), *Recent developments in optimization and optimal control in chemical engineering* (pp. 229–252). Transworld Research Publications.

Chung, T. S., & Wu, C. J. (1992). A computationally efficient numerical algorithm, for the minimum-time control problem of continuous systems. *Automatica*, *72*, 841–847.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, *26*(1), 29–41.

Gill, P. E., Murray, W., Saunderss, M. A., & Wright, M. H. (1998). *User's guide for NPSOL (v.5.0): A FORTRAN package for nonlinear programming*. Technical Report 98-2, Department of Mathematics, U.C. San Diego.

Hristu-Varsakelis, D., & Shao, C. (2004). Biologically-inspired optimal control: Learning from social insects. *International Journal of Control*, *77*(18), 1545–1566.

Hristu-Varsakelis, D., & Shao, C. (2005). Corrections to biologically-inspired optimal control: Learning from social insects. *International Journal of Control*, *78*(2), 157.

Hristu-Varsakelis, D., Shao, C., & Samaras, N. (2007), Local pursuit as a bio-inspired computational optimal control tool. In *Proceedings of the European Control Conference*, July 2007, to appear.

Jadbabaie, A., Lin, J., & Morse, A. S. (2003). Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, *48*(6),

Kurazume, R., & Hirose, S. (1998). Study on cooperative positioning system: Optimum moving strategies for cps-iii. In *Proceedings of the 1998 IEEE international conference in robotics and automation*, Vol. 4, Leuven, Belgium, pp. 2896–2903.

Luus, R. (2000). *Iterative dynamic programming. Monographs and surveys in pure and applied mathematics*. London: Chapman & Hall/CRC.

Moon, S. M., & VanLandingham, H. F. (1997). On the variational approach in a time optimal control problem. In *Proceedings of the IASTED international conference of CONTROL'97*, pp. 290–293.

Ögren, P., Fiorelli, E., & Leonard, N. E. (2002). Formation with a mission: Stable coordination of vehicle group maneuvers. In *Proceedings of the 15th international symposium of MTNS*, Notre Dame, IL.

Parrish, J. K., & Hammer, W. M. (1997). *Animal groups in three dimensions*. UK: Cambridge.

Passino, K., Polycarpou, M., Jacques, D., Pachter, M., Liu, Y., Yang, Y., et al., (2002). Cooperative control for autonomous air vehicles. In: R. Murphey, & P. M. Pardalos (Eds.), *Cooperative control and optimization* (pp. 233–272). Dordrecht: Kluwer Academic Publishers.

Roumeliotis, S. I., & Bekey, G. A. (2002). Distributed multi-robot localization. *IEEE Transactions on Robotics and Automation*, *18*(5), 781–795.

Sakawa, Y., & Shindo, Y. (1982). Optimal control of container cranes. *Automatica*, *18*, 257–266.

Shao, C., & Hristu-Varsakelis, D. (2005a). *Bio-inspired cooperative optimal control with partially-constrained final state*. Technical Report 2005-76, ISR, University of Maryland, College Park, MD. Available online at: ⟨http://techreports.isr.umd.edu/reports/2005/TR_2005-76.pdf⟩.

Shao, C., & Hristu-Varsakelis, D. (2005b). Optimal control through biologically inspired pursuit. In *Proceedings of the 16th IFAC world congress*, Prague.

Teo, K. L., Jennings, L. S., Lee, H. W. J., & Rehbock, V. (1999). The control parameterization enhancing transform for constrained optimal control problems. *Journal of Australasian Mathematical Society Series B*, *40*, 314–335.

Wagner, I. A., Lindenbaum, M., & Bruckstein, A. M. (1999). Distributed covering by ant-robots using evaporating traces. *IEEE Transactions on Robotics and Automation*, *15*(5), 918–933.

Yamaguchi, H., & Burdick, J. W. (1998). Asymptotic stabilization of multiple nonholonomic mobile robots forming group formations. In *Proceedings of IEEE international conference in robotics and automation*, Vol. 4, pp. 3573–3580.

Dimitrios Hristu-Varsakelis is a faculty member in the Department of Applied Informatics at the University of Macedonia, Greece. He received his Ph.D. in Engineering Sciences (1999) and M.S. in Applied Mathematics (1997) from the Division of Engineering and Applied Sciences at Harvard University. During 2000–2005 he was an Assistant Professor in the Department of Mechanical Engineering and held a joint appointment with the Institute for Systems Research from 2002 to 2005. His research interests include control under limited communication, bio-inspired decision-making, and modeling of socio-economic systems. Dr. Hristu-Varsakelis is a senior member of the IEEE and a member of SIAM. He is a co-recipient of the 1999 Eliahu Jury award from the Division of Engineering and Applied Sciences, Harvard University, and a co-recipient of the 2005 IFAC Young Author Prize.

Cheng Shao received his Ph.D. in Mechanical Engineering in 2005 from the University of Maryland, College Park, MD, USA. Prior to that, he received the M.S. degree (2001) in Mechatronics, and the B.S. (1998) in Machine Design and Manufacturing, both from Tsinghua University, Beijing, China. He is currently an Analyst with SAC Capital Management, New York, NY, USA.