JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 18, No. 1, January–February 1995

Genetic Algorithm Approach for Optimal Control Problems with Linearly Appearing Controls

H. Seywald* and R. R. Kumar*
Analytical Mechanics Associates, Inc., Hampton, Virginia 23666
and
S. M. Deshpande†
Booz, Allen & Hamilton, Inc., Los Angeles, California 90086

For optimal control problems in Mayer form with all controls appearing only linearly in the equations of motion, this paper presents a method for calculating the optimal solution without user-specified initial guesses and without a priori knowledge of the optimal switching structure. The solution is generated in a sequence of steps involving a genetic algorithm (GA), nonlinear programming, and (multiple) shooting. The centerpiece of this method is a variant of the GA that provides reliable initial guesses for the nonlinear programming method, even for large numbers of parameters. As a numerical example, minimum-time spacecraft reorientation trajectories are generated. The described procedure never failed to correctly determine the optimal solution.

Introduction

FINDING the solution to an optimal control problem is a difficult and time-consuming task. By employing Pontryagin's minimum principle in conjunction with simple or multiple shooting to solve the resulting boundary-value problem (BVP), this task becomes equivalent to finding the numerical values of the costates (Lagrange multipliers) associated with the physical states of the underlying dynamic system at discrete times. Thus, the problem of solving an optimal control problem can be reduced to solving a nonlinear system of equations. Usually, Newton-Raphson methods are well suited for this type of problem. However, due to the sensitivity of the state-costate dynamical system, the task of finding initial guesses that lie within the domain of convergence can become arbitrarily difficult. In addition, if a control appears only linearly in the equations of motion, the optimal solution is known to consist of a sequence of bang-bang and, possibly, singular subarcs. The switching structure, however, is not known in advance and has to be found by trial and error.

The present paper introduces a method for generating the optimal control solution for problems in which all controls appear only linearly in the equations of motion. In this method, the user need not provide initial guesses for the state history, the control history, the costate history, or the switching structure. Initial guesses that lie within the domain of convergence of a gradient search method are generated with a genetic algorithm (GA) using substring length 1 for each individual control parameter. A theoretical justification of the approach is given through hodograph analysis and convexity arguments. General convergence arguments pertaining to the GA are mainly heuristic and based on practical experience. Because of the probabilistic nature of GAs, this seems to be unavoidable.

Problem Formulation

We consider optimal control problems of the following general form:

$$\min_{u \in (PWC[t_0, t_f])^m} \Phi(x(t_f), t_f) \tag{1}$$

subject to the equations of motion

$$\dot{x}(t) = a(x(t), t) + \sum_{i=1}^{m} b_i(x(t), t) u_i(t) \tag{2}$$

the initial and boundary conditions

$$x(t_0) = x_0 \tag{3}$$

$$\Psi(x(t_f), t_f) = 0 \tag{4}$$

and the control constraints

$$u_i(t) \in [0, 1] \qquad \text{for} \qquad i = 1, \ldots, m \tag{5}$$

Here, PWC$[t_0, t_f]$ denotes the set of all real-valued piecewise continuous functions on the interval $[t_0, t_f]$. The terms $x(t) \in \boldsymbol{R}^n$ and $u(t) \in \boldsymbol{R}^m$ denote the state vector and the controls, respectively. All functions $\Phi: \boldsymbol{R}^{n+1} \to \boldsymbol{R}$, $a: \boldsymbol{R}^{n+1} \to \boldsymbol{R}^n$, $b_i: \boldsymbol{R}^{n+1} \to \boldsymbol{R}^n$, $i = 1, \ldots, m$, and $\Psi: \boldsymbol{R}^{n+1} \to \boldsymbol{R}^k$, $k \le n$, are assumed to be smooth of whatever order is required for this paper.

This represents a fairly general class of optimal control problems, the most significant restriction for this paper being that the controls appear only linearly in the equations of motion (2). In the following sections this assumption will be exploited to make an otherwise brute-force GA-based approach for calculating approximate solutions efficient and robust. Also note that, except for constraints (5), no further control or state constraints are considered in the problem formulation. Unless such constraints can be written in the form

$$u \le u_{\max}(x), \qquad u \ge u_{\min}(x)$$

where $u_{\max}(x)$, $u_{\min}(x)$ are smooth, a priori, known functions of states, this represents a nontrivial restriction that will not be resolved in this paper.

Problem Discretization: Step (i)

To calculate an approximate solution to the optimal control problem (1–5), the control functions of time $u_i(t) \in PWC[t_0, t_f]$ are discretized to piecewise constant functions $\hat{u}_i(t)$. To this end, let N be a user-chosen integer and let $t_0, t_1, \ldots, t_N = t_f$ be (for simplicity) an equidistant subdivision of the time interval $[t_0, t_f]$. Then, for $i = 1, \ldots, m$ and for $j = 1, \ldots, N$, we define

$$\hat{u}_i(t) = u_i^j \qquad \text{on} \qquad [t_{j-1}, t_j]. \tag{6}$$

Received Nov. 14, 1992; revision received Nov. 15, 1993; accepted for publication May 12, 1994.

*Supervising Engineer. Member AIAA.

†Associate.

For fixed parameters u_i^j, the state vector $\hat{x}(t)$ is the unique solution of the initial-value problem

$$\dot{\hat{x}}(t) = a(\hat{x}(t), t) + \sum_{i=1}^{m} b_i(\hat{x}(t), t)\hat{u}_i(t) \qquad (7)$$
$$\hat{x}(t_0) = x_0$$

It is well known[1–3] that $\hat{x}(t_f)$ is a smooth function of the parameters u_i^j, $i = 1, \dots, m$, $j = 1, \dots, N$, and the final time t_f. Hence, with the parameter vector U defined by

$$U = \left[u_1^1, \dots, u_m^1, \dots, u_1^N, \dots, u_m^N, t_f\right]^T \in \boldsymbol{R}^{mN+1} \qquad (8)$$

it is clear that

$$f(U) = \Phi(\hat{x}(t_f), t_f) \qquad (9)$$

$$g(U) = \Psi(\hat{x}(t_f), t_f) \qquad (10)$$

where Φ and Ψ are given by Eqs. (1) and (4) and $\hat{x}$ is defined by Eq. (7), respectively, represent smooth functions of U.

In the following sections we discuss a method to solve the problem

$$\min_{U \in \Omega} f(U) \qquad (11)$$

subject to

$$g(U) = 0 \qquad (12)$$

where Ω is the set of all $U \in \boldsymbol{R}^{mN+1}$, U as defined in Eq. (8), with the properties

$$0 \le u_i^j \le 1 \qquad \forall\, i = 1, \dots, m, \; j = 1, \dots, N \qquad (13)$$

and

$$t_f^{\min} \le t_f \le t_f^{\max} \qquad (14)$$

Here, $t_f^{\min}$, $t_f^{\max}$ are user-provided lower and upper bounds for the final time t_f, respectively.

The parameter optimization problem (11) and (12), resulting from the problem discretization presented above, is numerically very well behaved. Loosely speaking, each component u_i^j of the parameter vector U defined in Eq. (8) has locally an influence on the trajectory of about the same magnitude, and even relatively large changes in any single component u_i^j affect the trajectory only moderately. Even though controls applied "early" in the trajectory have a larger effect on the cost function (9) and the constraints (10) than the controls applied "later" in the trajectory, this discretization seems to be working well in conjunction with a shooting method.

Genetic Algorithm Approach for Generating Initial Guesses: Step (ii)

Cost Function

To make a gradient search method autonomous and robust, an algorithm is required that can provide a coarse approximation $\tilde{U}$ to the optimal solution U^*. For this task we apply a GA.[4,5] For use of the GA the constrained parameter optimization problem (11) and (12) is reformulated in a penalty function form. Explicitly, the cost function

$$F(U) := f(U) + c_1|g_1(U)| + \cdots + c_k|g_k(U)| \qquad (15)$$

is used. Here, f and g_i, $i = 1, \dots, k$, are the original cost function and constraint functions as given in Eqs. (9) and (10), respectively, and $c_1, \dots, c_k$ are user-specified positive weighting factors. Note that the penalty terms on the right-hand side of Eq. (15) are the absolute values of the magnitude by which the constraint components (12) are violated. This form of penalty function provides a much crisper measure for the violation of the constraints than error squares. The loss of differentiability is of no importance to the GA.

Using a GA of the general form described in the Goldberg's book,[4] we can now attempt to minimize the cost function $F(U)$ given in Eq. (15) subject to the constraints (13). With the dimension of the parameter vector U given by $mN+1$, where m and N denote the dimension of the control vector and the number of subintervals, respectively, it is clear that the complexity of the optimization problem increases rapidly with the number of the subintervals chosen by the user.

Substring Length 1 for Control Representation

In the GA approach each component U_i of the parameter vector U defined in Eq. (8) is approximated by a binary substring of user-specified length. Clearly, the length of these binary substrings is directly related to the resolution or the precision with which a solution can be obtained. But it is also crucially related to the family size required in the genetic algorithm and to the speed of convergence.[6] Given the finite central processing unit (CPU) time available for generating an acceptable approximation to the optimal solution, a large family size and slow convergence rate reduces the probability that convergence is obtained within the time available. The approach in this paper is to allow substring length 1 for each of the parameters U_i, $i = 1, \dots, mN$. That means, on every subinterval $[t_j, t_{j+1}]$, $j = 1, \dots, N$, every component u_k of the control vector is only allowed to have the values 0 or 1. Obviously, this approach keeps the string length to a minimum. A theoretical justification that this approach can yield meaningful results is given in the next section. In fact, even though the idea of using substring length 1 for the control representation may seem secondary at this point, it is the central idea in this paper. The problem discretization described in Eqs. (6–14) is tailored to implement this idea.

Theoretical Justification for Using Substring Length 1

From the theory of optimal control,[3,7,8] it is well known that the optimal solution to problem (1–5) "usually" is synthesized as a finite sequence of so-called bang-bang arcs, along which the optimal control rides identically on its upper or lower bound. In the not so common case where the control takes on intermediate values throughout an arc, the arc and the control along the arc are called singular.

Excluding the singular-control case for the moment, it is clear that the approximation proposed in the previous section, namely to allow only the control values 0 and 1 along each subinterval, is not a serious restriction. In fact, in the interior of each bang-bang arc, this discretization allows perfect representation of the optimal control. Inevitable discretization errors are introduced only in subintervals in the interior of which the optimal control has a switching point, i.e., switches from one extreme value to the other. Obviously, such discretization errors can be made arbitrarily small by increasing the number of subintervals [i.e., increasing the integer N defined in Eq. (6)]. However, as we intend to use the solution found by the GA only as an initial guess for a gradient search method, it is clear that the GA solution need not be very precise and the integer N need not be very large.

In the case of singular arcs, i.e., arcs along which the optimal control assumes intermediate values, it is clear that, at each instant of time, the control found by the GA may be considerably different from the optimal control. However, the effect of intermediate control on the evolution of the states can be approximated closely if the control is flipped back and forth between its maximum and minimum allowed value along subsequent subintervals. In fact, it can be shown that such chattering control[3] can approximate the evolution of the states arbitrarily closely if only the length of the subintervals is made sufficiently small, i.e., if N is made sufficiently large. Again, the argument is that the GA is only meant to produce an approximate solution to be used as an initial guess for a gradient search method, so the integer N can be kept reasonably small.

Nonlinear Programming Approach: Step (iii)

The smoothness of the representing functions f, g in Eq. (9) and (10) allows for the use of a gradient search method such as the NPSOL program,[9,10] which has become an industry standard for solving nonlinear programming problems. This code searches for a stationary point U^* at which the first-order necessary conditions for optimality, the well-known Kuhn-Tucker conditions,[7,10,11]

$$\left.\frac{\partial f}{\partial U}\right|_{U^*} + \lambda^T \left.\frac{\partial g}{\partial U}\right|_{U^*} = 0 \qquad (16)$$

$$g(U^*) = 0 \qquad (17)$$

are satisfied. The second-order necessary condition requires that

$$\Delta U^T \left(\frac{\partial^2 f}{\partial U^2} + \lambda^T \frac{\partial^2 g}{\partial U^2} \right) \bigg|_{U^*} \Delta U \geq 0 \tag{18}$$

for all ΔU that satisfy $\partial g / \partial U|_{U^*} \Delta U = 0$ and is used by the code to distinguish between relative maxima and minima.

The user has to provide an initial guess only for the parameter vector U, and once close enough to the optimal solution U^*, the code is guaranteed to have a superlinear to quadratic convergence rate. Furthermore, if a gradient search method reaches at the optimal solution, then it does so with very high precision.

The notorious drawbacks of gradient search methods are well known. First, for highly nonlinear problems, which are typically encountered in trajectory optimization, it may be quite nontrivial to find an initial guess U from which a gradient search method can converge at all. Second, gradient search methods are vulnerable to getting caught in local minima. In practice, this often means that intuitive, physically feasible, suboptimal solutions that are consistent with the constraints (3) and (4) may not be good enough initial guesses. However, with the approximate solution $\bar{U}$, obtained by the GA, used as an initial guess, this algorithm becomes autonomous and robust.

The benefit of the solutions obtained from the discretization scheme proposed so far is twofold. First, these solutions themselves provide acceptable engineering solutions to optimal control problems. Second, the solutions can be used to provide the necessary information to start high-fidelity methods that employ a more problem-specific discretization. For example, if the optimal switching structure is identified to be void of singular arcs, then a nonlinear programming approach in which the location structural of the switching points are the parameters to be optimized can be employed. If singular-control arcs are part of the optimal solution, then, in most cases, the user has to resort to solving the first-order necessary conditions of optimal control. But exceptions are possible, e.g., in the cases where application of the Pontryagin minimum principle[3,7,8] yields the singular control in state feedback form.

Identification of Optimal Switching Structure: Step (iv)

The solutions found by solving the discretized problem as described in the previous sections can be used to identify the correct switching structure associated with the optimal control solution. This means that the sequence of arcs along which the optimal control takes on its upper bound, its lower bound, or intermediate values can be determined by simple inspection of the control functions of time obtained from parameter optimization. If only the user-chosen number of subintervals N [see step (i)] is sufficiently large and if the GA succeeded in generating an approximate solution close to the global minimum of the cost function (15), it can be expected that the switching structure suggested by the parameter optimization solution is identical to the switching structure of the optimal control solution. Again, by simple inspection of the parameter optimization solution, values can be also obtained for the approximate location of the switching points, i.e., the times at which the controls switch between different control logics.

In the next section a method is presented for generating the optimal control solution to problem (1–5) by employing the results obtained in steps (iii) and (iv).

Synthesis of Optimal Control Solutions: Step (v)

The first-order necessary conditions of optimal control associated with problem (1–5) are given by

$$\dot{\lambda}^T = -\frac{\partial H}{\partial x} \tag{19}$$

$$\lambda^T(t_f) - \frac{\partial \Phi}{\partial x(t_f)} - \nu^T \frac{\partial \Psi}{\partial x(t_f)} = 0 \tag{20}$$

$$H\big|_{t_f} + \frac{\partial \Phi}{\partial t_f} + \nu^T \frac{\partial \Psi}{\partial t_f} = 0 \tag{21}$$

Here, $\lambda(t) \in \boldsymbol{R}^n$ and $\nu \in \boldsymbol{R}^k$ are unknown Lagrange multiplier vectors and the Hamiltonian H is defined by

$$H(x, u, \lambda, t) = \lambda^T \left(a(x, t) + \sum_{i=1}^{m} b_i(x, t) u_i \right) \tag{22}$$

At every instant of time the optimal control is determined from the Pontryagin minimum principle,[3,7,8] which requires that the Hamiltonian (22) be minimized subject to all control constraints (22), i.e.,

$$u_i^* = \begin{cases} 0 & \text{if} \quad S_i > 0 \\ 1 & \text{if} \quad S_i < 0 \\ \text{singular} & \text{if} \quad S_i \equiv 0 \end{cases} \tag{23}$$

Here,

$$S_i = \frac{\partial H}{\partial u_i} \tag{24}$$

is referred to as the switching function associated with control u_i. In the singular-control case, where $S_i \equiv 0$ on a nonzero subinterval of the total time interval $[t_0, t_f]$, an explicit expression for the optimal control has to be determined from successive differentiation of the identity $S_i \equiv 0$ (see Ref. 12). Furthermore, at each switching point t_s, where a control component, say u_i, switches between any of the three possible control logics defined in Eq. (23), the associated switching function has to be zero, i.e.,

$$S_i(t_s) = 0 \tag{25}$$

Let us assume now that, for a particular problem, the methodology presented in steps (i)–(iv) above leads to the tentative identification of the optimal switching structure. Assuming also, for simplicity, that no singular control is involved, the set of optimality conditions is given by the differential equations (2) and (19), the initial conditions (3), the final conditions (4), (20), and (21), and at each point of control discontinuity, a switching condition of the form (24). This represents a multipoint BVP. The unknown variables can be considered to be the initial value of the time-varying Lagrange multiplier vector $\lambda(t)$, the constant multiplier vector ν, and the location of the switching times t_s.

Considering the state and control functions of time, say, $\hat{x}(t)$ and $\hat{u}(t)$, obtained from the parameter optimization approach discussed in steps (i)–(iv) above as fixed functions of time, the costate equation (19) reduces to the linear time-varying ordinary differential equation

$$\dot{\lambda}^T(t) = -\lambda^T(t) \left(\frac{\partial a(\hat{x}(t), t)}{\partial x} + \sum_{i=0}^{m} \frac{\partial b_i(\hat{x}(t), t)}{\partial x} u_i(t) \right) \tag{26}$$

It can be easily verified now that, for fixed switching times $\hat{t}_s$ and fixed final time $\hat{t}_f$, the left-hand sides of conditions (20), (21), and (24) depend linearly on the initial values $\lambda(t_0)$ and the multiplier vector ν. Assuming the nondegenerate case where the number of switching points is sufficiently large, such that all unknown parameters $\lambda(t_0)$ and ν are completely determined by conditions (20), (21), and (24), we can apply a linear least-squares approach to solve for $\lambda(t_0)$ and ν. Note that it may not be possible to satisfy all of the conditions (20), (21), and (24) exactly, as $\hat{x}(t)$, $\hat{u}(t)$, t_f, and the switching times t_s may be different from the respective values associated with the optimal control solution. However, the continuous dependence of the left-hand sides of conditions (20), (21), and (24) on the states and control functions of time (with respect to the L_2 norm), the final time and the switching times, guarantees that the linear least-squares solution $\lambda(t_0)$, ν is close to the respective quantities associated with the optimal control solution as long as the parameter optimization solution obtained in steps (i)–(iv) is close to the optimal control solution. Hence it can be expected that, by generating initial guesses as described above, a multipoint BVP solver converges without further user interaction.

Numerical Example: Minimum-Time Spacecraft Reorientation

To illustrate the performance of the proposed approach, we consider the following minimum-time spacecraft reorientation (MTSR) problem[13,14]

$$\min_u t_f \tag{27}$$

subject to the dynamic system

$$\dot{\omega} = u \tag{28}$$

$$\dot{q} = \tfrac{1}{2}\Omega q \tag{29}$$

the initial conditions

$$\omega(0) = [0, 0, 0]^T, \qquad q(0) = [1, 0, 0, 0]^T \tag{30}$$

the final conditions

$$\omega(t_f) = [0, 0, 0]^T, \qquad q(t_f) = [0, 0, 0, 1]^T \tag{31}$$

the control constraints

$$u_i \in [-1, +1], \qquad i = 1, 2, 3 \tag{32}$$

and free final time t_f. Here, $\omega = [\omega_1, \omega_2, \omega_3]^T$ and $q = [q_0, q_1, q_2, q_3]^T$ represent the vector of angular velocities and the quaternion vector, respectively. The matrix Ω is given by

$$\Omega = \begin{bmatrix} 0 & -\omega_1 & -\omega_2 & -\omega_3 \\ \omega_1 & 0 & \omega_3 & -\omega_2 \\ \omega_2 & -\omega_3 & 0 & \omega_1 \\ \omega_3 & \omega_2 & -\omega_1 & 0 \end{bmatrix} \tag{33}$$

The equations of motion (28) and (29) refer to an inertially symmetric spacecraft with orthogonal control axes and the initial and final conditions (30) and (31) prescribe a 180-deg rest-to-rest reorientation.

In Ref. 13 the authors applied time-consuming continuation methods to obtain a first solution of problem (27–33). With the methods described in the present paper the solution-finding process is nearly automized.

For the GA approach, the free final time is represented by a 20-digit binary substring. All controls are represented by one-digit binary substrings. Hence, with the total time interval divided into N subintervals, the string length used in GA is $3N + 20$ and the total number of parameters to be optimized by the nonlinear programming code is $3N + 1$. In all GA calculations a population of 30 was used. The probabilities of mutation and crossover were set 0.001 and 0.3, respectively. The cost function to be minimized by GA is

$$F(U) = ct_f + |\omega_1(t_f)| + |\omega_2(t_f)| + |\omega_3(t_f)| + |q_0(t_f)| + |q_1(t_f)| + |q_2(t_f)| + |q_3(t_f) - 1| \tag{34}$$

Here, $c > 0$ is a user-chosen real number and U is defined as in Eq. (8) with $m = 3$. The initial conditions (30) can be satisfied directly by starting the integration at the appropriate initial states and need not be considered in the cost function (34).

In a first approach the optimal final time is assumed to lie between 0 and 5. Discretizing the problem by dividing the total time interval into only 10 subintervals quickly yields a solution satisfying all boundary conditions with a final time less than 3.3. It is not expected that the control functions of time obtained in this solution can provide insight into the structure of the optimal control, but it provides valuable and easily obtained information about the upper bound on the optimal final time.

Next the problem is discretized by dividing the total time interval into 200 subintervals. The free final time is again represented by a 20-digit substring, this time restricting the final time between 0 and 3.3. The state histories obtained from GA after 1000, 4000, and 14,000 iterations are shown in Figs. 1–3. To keep the algorithm from setting the final time equal zero, the final time is added to the total cost function with a very small weighting factor $c = 0.0001$ during the first 1000 iterations. In all further GA iterations this weighting factor is increased to 0.01.

Using the results shown in Figs. 1 and 2 as initial guesses for a gradient search method yields the results shown in Figs. 4 and 5, respectively. Proceeding with these two parameter optimization solutions as discussed in steps (iv) and (v) above yields two different optimal control solutions with exactly the same cost. The associated state functions of time are indistinguishable from those given in Figs. 4 and 5, respectively. The final time associated with the parameter optimization solution is less than 2% larger than the minimum possible maneuver time found by optimal control. The fact that at least two different solutions with the same absolute minimum final time are obtained can be easily explained through spatially symmetric reorientation maneuvers.[15] It can be expected that all possible spatially symmetric optimal solutions are obtained if initial guesses for the nonlinear programming approach are generated using the GA with different seeds in the random-number generator.

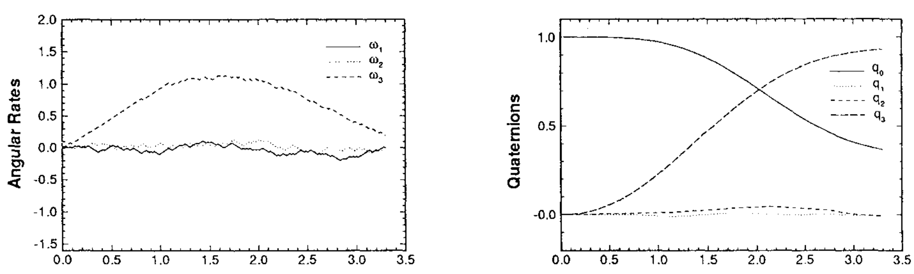

Fig. 1 Solution obtained after 1000 GA iterations.

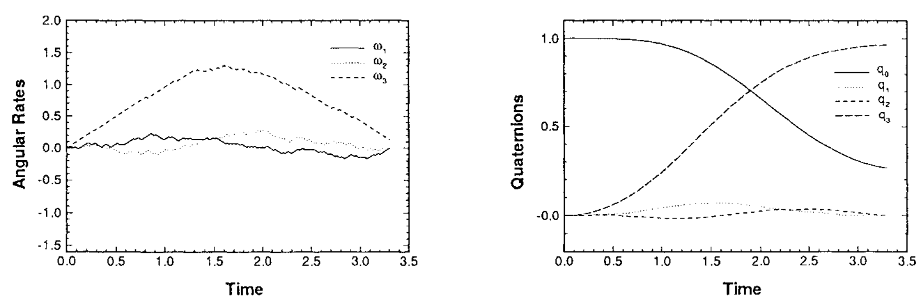

Fig. 2 Solution obtained after 4000 GA iterations.

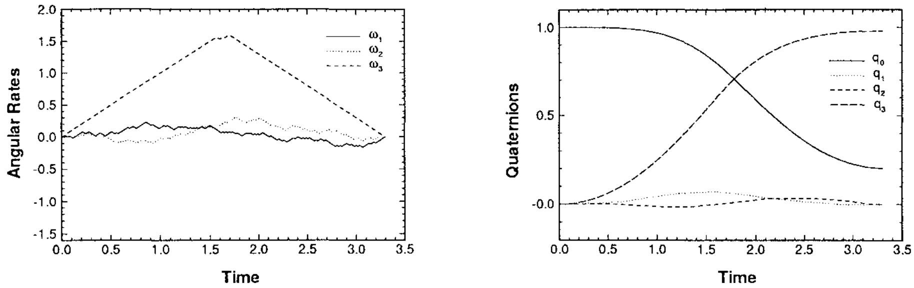

Fig. 3 Solution obtained after 14,000 GA iterations.

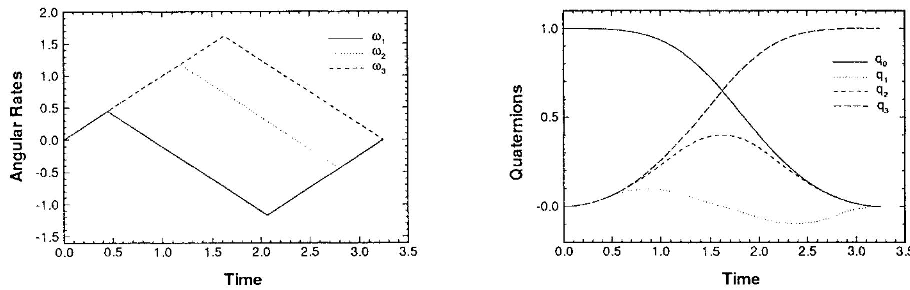

Fig. 4 Solution obtained from gradient search method.

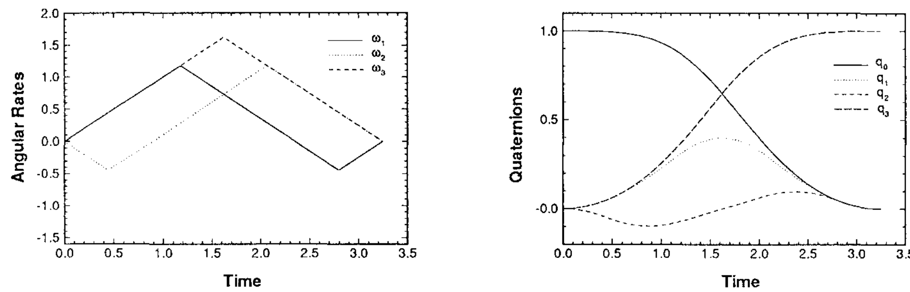

Fig. 5 Solution obtained from gradient search method.

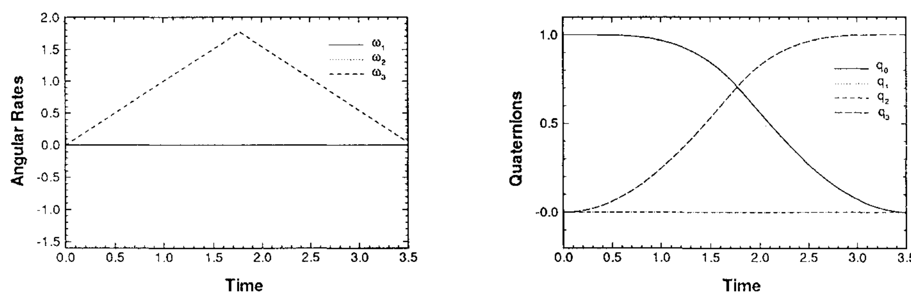

Fig. 6 Solution obtained from principal-axis rotation.

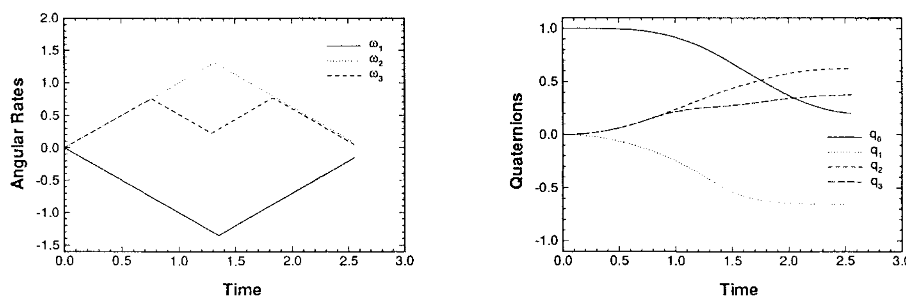

Fig. 7 Another example without physically intuitive initial guess.

In Fig. 6, the state histories associated with a principal-axis rotation are shown. This suboptimal control logic also provides a good enough initial guess for a gradient search method to converge and leads to the solution shown in Fig. 5. However, for many problems of practical interest such a physically motivated initial guess may not be available. This is demonstrated in a second example, where the same MTSR problem (27–30) is solved for the arbitrarily chosen final conditions

$$\omega(t_f) = [-0.15,\ 0.08,\ 0.05]^T$$
$$q(t_f) = [0.20,\ -0.66,\ 0.62,\ 0.37]^T \tag{35}$$

In terms of the usual Euler angles Ψ, θ, Φ, the above boundary conditions on the quaternion vector q refer to an angular reorientation

$\Delta\Psi = -94.17$ deg, $\Delta\theta = +47.89$ deg, $\Delta\Phi = +163.64$ deg. For this example no physically intuitive initial guess is available. Mechanically applying steps (i)–(iv) of the method described above leads to an optimal control solution satisfying all the first-order necessary conditions, without investing any significant expertise in optimal control. The state histories of the associated nonlinear programming solution are presented in Fig. 7.

Summary and Conclusions

For optimal control problems where all controls appear only linearly in the equations of motion a method is introduced to calculate the optimal control solution without significant user interaction. First the problem is discretized to a finite-dimensional parameter optimization problem by allowing only piecewise constant control functions of time. Initial guesses for a gradient search method are generated with a genetic algorithm. The nonlinear programming results are used to identify the optimal switching structure and to calculate reasonable initial guesses for the Lagrange multipliers. These data are used as initial guesses to solve the multipoint boundary value problem associated with the first-order necessary conditions of optimal control. Using as a practical example the nonlinear problem of minimum time spacecraft reorientation, it is shown that the proposed algorithm enjoys outstanding robustness and stability. It is interesting to note that in earlier publications authors resorted to tedious continuation methods to obtain solutions for a similar problem.

References

[1]Bartle, R. G., *The Elements of Real Analysis*, Wiley, New York, 1975.

[2]Taylor, A. E., *General Theory of Functions and Integration*, Blaidel, 1965, p. 419.

[3]Lee, E. B., and Markus, L., *Foundations of Optimal Control Theory*, Krieger, Malabar, FL, 1986.

[4]Goldberg, D. E., *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[5]Holland, J., *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor, MI, 1975.

[6]Goldberg, D. E., Deb, K., and Clark, J. H., "Genetic Algorithms, Noise, Undersizing of Populations," *Complex Systems*, Vol. 6, 1992, pp. 333–362.

[7]Bryson, A. E., and Ho, Y. C., *Applied Optimal Control*, Hemisphere, New York, 1975.

[8]Neustadt, L. W., *Optimization: A Theory of Necessary Conditions*, Princeton Univ. Press, Princeton, NJ, 1976.

[9]Gill, P. E., Murray, W., Saunders, M. A., and Wright, M. H., "User's Guide for NPSOL (Version 4.0): A Fortran Package for Nonlinear Programming," Systems Optimization Laboratory, Department of Operations Research, Stanford Univ., Stanford, CA.

[10]Gill, P. E., Murray, W., and Wright, M. H., *Practical Optimization*, Academic, New York, 1981.

[11]Wouk, A., *A Course of Applied Functional Analysis*, Wiley, New York, 1979.

[12]Bell, D. J., and Jacobson, D. H., *Singular Optimal Control Problems*, Mathematics in Science and Engineering, Academic, New York, 1975.

[13]Bilimoria, K. D., and Wie, B., "Minimum-Time Large-Angle Reorientation of a Rigid Spacecraft," Aug. 1990.

[14]Seywald, H., and Kumar, R. R., "Singular Control in Minimum Time Spacecraft Reorientation," *Proceedings of the AIAA Guidance, Navigation and Control Conference* (New Orleans, LA), AIAA, Washington, DC, 1991, pp. 432–442 (AIAA Paper 90-3486).

[15]Bocvarov, S., Lutze, F. H., and Cliff, E. M., "Extremal Spacecraft-Reorientation Maneuvers: Symmetries and Group Properties," *Proceedings of the AIAA Guidance, Navigation, and Control Conference* (New Orleans, LA), AIAA, Washington, DC, 1991, pp. 936–948.

[16]Chowdhry, R. S., and Cliff, E. M., "Optimal Rigid Body Reorientation Problem," AIAA Paper 90-3485, Aug. 1990.