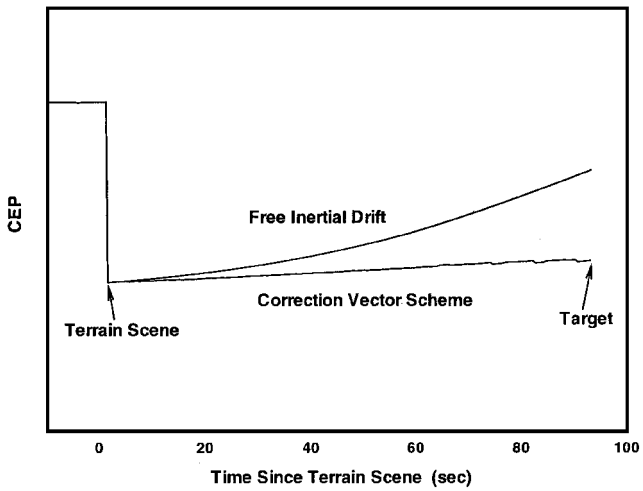


Fig. 2 Accuracy of correction vector scheme.

the covariance simulation incorporates the equations developed previously. Inasmuch as the Tomahawk missile is being considered, the terrain scene is a terminal area digital scene matching and area correlation (DSMAC) scene.

Shown in Fig. 2 is a typical CEP plot from the covariance simulation, which illustrates that the highly accurate positional solution obtained from the relative GPS correction vector scheme degrades only minimally between the DSMAC scene and the target. For target penetration considerations, the results presented here incorporate a terminal trajectory with a high-angle dive to impact the target. At the target, the navigational error contributes 2.6%, the steering error 10.2%, the scene center location error 20.9%, and the relative target location error 66.3%, which is consistent with the earlier discussion regarding relative error magnitudes. An examination of Fig. 2 also reveals that the correction vector scheme affords a substantial improvement over the situation where the Kalman filter accepts a high-quality DSMAC positional update and, subsequently, the missile flies in a free inertial manner to the target (the mode of operation for the current Block III Tomahawk missile). Because improvements in satellite imaging technology are expected to permit larger DSMAC scene to target separations while maintaining the same scene/target relative error, free inertial navigation to the target will clearly become less desirable, and relative GPS schemes such as the one proposed herein more desirable.

Summary

The advantage of employing the relative GPS correction vector scheme is basically threefold. First, through the virtual elimination of the pseudorange bias and absolute target location errors, implementation of this scheme results in accuracies at the target that are significantly better than those afforded by a standard GPS-only navigation solution. Second, the necessary modifications to existing navigation software are minimal, and thus the correction vector scheme is seen to offer affordability, especially when compared to the other primary alternative for achieving precision strike accuracies, i.e., weapons equipped with terminal guidance seekers. Continuing the comparison to terminal seekers, the proposed scheme also has an all-weather advantage because terminal seekers typically suffer degraded performance in certain environments, e.g., infrared seekers in low cloud ceilings. Finally, the proposed GPS bias elimination scheme permits significantly larger terrain scene to target separations and, hence, greater flexibility in mission planning.

References

[1]Schmidt, G., and Setterlund, R., "Precision Strike Concepts Exploiting Relative GPS Techniques," *Navigation, Journal of the Institute of Navigation*, Vol. 41, No. 1, 1994, pp. 101–103.

[2]Youhanaie, M., Hartman, D., Ziolkowski, F., and Pujara, N., "Relative GPS Guidance for Precision Weapon Delivery," Inst. of Navigation, GPS-95 Technical Meeting, Paper 7671, Sept. 1995.

[3]Phillips, R. G., and Schmidt, G. T., "Relative and Differential GPS," AGARD Lecture Series 207, System Implications and Innovative Applications of Satellite Navigation, June 1996, pp. 5-3–5-6.

Concatenated Approach to Trajectory Optimization

Hans Seywald* and Renjith R. Kumar*
Analytical Mechanics Associates, Inc., Hampton, Virginia 23666

I. Introduction

In this Note, a new method is presented for the numerical calculation of near-optimal solutions to trajectory optimization problems. The new approach, which is applicable to a wide class of problems, relies on a scheme of sequentially solving low-order discretized subproblems on short, partly overlapping subarcs. The technique requires very little computer memory and is ideally suited for parallel processing. The overall convergence rate is probably only linear, but the calculation of individual trajectories is very fast, which makes the overall algorithm competitive also in terms of speed. Furthermore, each iterate of the trajectory satisfies all physical constraints, i.e., represents a feasible trajectory, even before complete convergence to the optimal trajectory is achieved. That means the iteration process can be stopped at any time and the currently best obtained solution can be used as a feasible trajectory.

II. Problem Formulation

Let us consider the following class of optimal control problems:

$$\min_{u \in (PWC[t_0,\, t_f])^m} \int_{t_0}^{t_f} L(x, u, t)\, \mathrm{d}t \tag{1}$$

subject to the conditions

$$\dot{x} = f(x, u, t) \tag{2}$$

$$\psi_0[x(t_0), t_0] = 0 \tag{3}$$

$$\psi_f[x(t_f), t_f] = 0 \tag{4}$$

Here, $x(t) : \boldsymbol{R} \to \boldsymbol{R}^n$, $u(t) : \boldsymbol{R} \to \boldsymbol{R}^m$ are the state and control functions of time, respectively, and t is the time. In addition to the conditions (2–4), we may consider control constraints and state constraints of the general form

$$g(x, u, t) \leq 0 \tag{5}$$

$$h(x, t) \leq 0 \tag{6}$$

respectively. The smoothness of the functions

$$L : \boldsymbol{R}^{n+m+1} \to \boldsymbol{R}, \qquad f : \boldsymbol{R}^{n+m+1} \to \boldsymbol{R}^n$$

$$\psi_0 : \boldsymbol{R}^{n+1} \to \boldsymbol{R}^{k_0}, \qquad k_0 \leq n$$

$$\psi_f : \boldsymbol{R}^{n+1} \to \boldsymbol{R}^{k_f}, \qquad k_f \leq n$$

$$g : \boldsymbol{R}^{n+m+1} \to \boldsymbol{R}^p, \qquad h : \boldsymbol{R}^{n+1} \to \boldsymbol{R}^q$$

with respect to their arguments is assumed to be of whatever order is required. $(PWC[t_0, t_f])^m$ denotes the set of all piecewise continuous functions defined on the interval $[t_0, t_f]$ and mapping into $\boldsymbol{R}^m$. Conditions (2–4) represent the differential equations of the

Received Jan. 29, 1996; revision received Feb. 11, 1997; accepted for publication Feb. 12, 1997.

*Staff Scientist. Member AIAA.

underlying dynamical system, the initial conditions, and the final conditions, respectively. For conciseness, control constraints and state constraints are only considered in inequality form [see (5) and (6)]. The results obtained in this Note can be immediately extended to the case where state and control constraints are given as equality constraints or as a mixture of both, equality and inequality constraints.

Loosely speaking, the special feature that we require in the problem formulation (1–6) is that the cost state y defined by

$$\dot{y} = L(x, u, t), \qquad y(0) = 0$$

be controllable independently of the states x. More precisely, we require the following.

Condition 1. On any subinterval $[a, b] \subset [t_0, t_f]$, the contribution

$$\int_a^b L(x, u, t)\,\mathrm{d}t$$

to the cost function must not be completely determined by prescribing all the states x at a and b.

This additional condition enables us to improve a given nonoptimal trajectory iteratively by optimizing isolated subarcs subject to fixed boundary states. This concept, which is the central idea that leads to the concatenated approach to trajectory optimization (CATO) algorithm introduced in the present Note, is explained in more detail in the next section.

III. Principle of Optimality

The principle of optimality in its standard form applied to problem (1–6) can be stated as follows (see, for example, Ref. 1).

Theorem 1. Let $x^*(t)$, $u^*(t)$, $t \in [t_0, t_f]$, denote the optimal solution to problem (1–6), and for some $t_1 \in [t_0, t_f)$ let $x_1 = x^*(t_1)$. Then $x^*(t)$, $u^*(t)$, $t \in [t_1, t_f]$, represent an optimal solution to the problem

$$\min_{u \in (PWC[t_1, t_f])^m} \int_{t_1}^{t_f} L(x, u, t)\,\mathrm{d}t$$

subject to (2) and (4–6) and subject to the initial condition $x(t_1) = x_1$.

The special structure of problem (1–6) with Condition 1 allows a considerably stronger statement than the preceding one. The fact that the total cost is accumulated uniformly throughout the trajectory independently of the state values at the boundaries and that the current value of the cost does not have any effect on the state evolution and on the constraints implies that every subarc of an optimal solution provides an optimal solution to the cost criterion $\int L(x, u, t)\,\mathrm{d}t$ by itself. More precisely, we can state the following.

Lemma 1. Let $x^*(t)$, $u^*(t)$, $t \in [t_0, t_f]$, denote the optimal solution to problem (1–6), and assume that Condition 1 is satisfied. Furthermore, for some t_1, $t_2 \in [t_0, t_f]$ with $t_1 < t_2$, let $x_1 = x^*(t_1)$, $x_2 = x^*(t_2)$. Then $x^*(t)$, $u^*(t)$, $t \in [t_1, t_2]$, represent an optimal solution to the problem

$$\min_{u \in (PWC[t_1, t_2])^m} \int_{t_1}^{t_2} L(x, u, t)\,\mathrm{d}t$$

subject to (2), (5), and (6) and subject to the boundary conditions $x(t_1) = x_1$, $x(t_2) = x_2$.

Another statement that can be immediately derived from Lemma 1 is as follows.

Lemma 2. Let $\bar{x}(t)$, $\bar{u}(t)$, $t \in [t_0, t_f]$, denote a solution to problem (2–6) [no matter what the value of the cost $\int L(x, u, t)\,\mathrm{d}t$ is], and for some t_1, $t_2 \in [t_0, t_f]$ with $t_1 < t_2$, let $x_1 = \bar{x}(t_1)$, $x_2 = \bar{x}(t_2)$. Furthermore, let $x^*(t)$, $u^*(t)$, $t \in [t_1, t_2]$, denote a solution to problem (2), (5), and (6), subject to the boundary conditions $x(t_1) = x_1$, $x(t_2) = x_2$, such that

$$\int_{t_1}^{t_2} L(x^*, u^*, t)\,\mathrm{d}t \le \int_{t_1}^{t_2} L(\bar{x}, \bar{u}, t)\,\mathrm{d}t$$

Then the functions $\hat{x}(t)$, $\hat{u}(t)$, defined by

$$\hat{x}(t) = \begin{cases} \bar{x}(t) & \text{on} \quad [t_0, t_1] \cup [t_2, t_f] \\ x^*(t) & \text{on} \quad (t_1, t_2) \end{cases}$$

$$\hat{u}(t) = \begin{cases} \bar{u}(t) & \text{on} \quad [t_0, t_1] \cup [t_2, t_f] \\ u^*(t) & \text{on} \quad (t_1, t_2) \end{cases}$$

represent a solution to problem (2–6), and the cost associated with $\hat{x}$, $\hat{u}$ is better than or equal to the cost associated with $\bar{x}$, $\bar{u}$, i.e.,

$$\int_{t_0}^{t_f} L(\hat{x}, \hat{u}, t)\,\mathrm{d}t \le \int_{t_0}^{t_f} L(\bar{x}, \bar{u}, t)\,\mathrm{d}t$$

IV. New Numerical Method: CATO

In the following, the idea of Lemma 2 is exploited to design a new numerical algorithm. Assume we are given a direct optimization approach whose precision is controlled through the number of discretization nodes along the time axis. To be specific, we consider the differential inclusion approach, which requires a discretization only of the states. However, the basic idea of the CATO approach is applicable also in conjunction with pure control discretization schemes (only the controls are discretized) or with collocation schemes (both controls and states are discretized). For demonstration purposes, we assume that the computer capability is limited to 4 nodes (3 subintervals) and that the desired resolution can be achieved only with 13 nodes (12 subintervals).

The algorithm consists of the following four steps, the last one of which will be repeated until convergence is achieved (see Fig. 1).

Step 1. Generate a 4-node direct solution to problem (1–6).

Step 2. Generate a 13-node starting trajectory by linearly interpolating the 4-node solution obtained in step 1.

Step 3. Improve the 13-node starting trajectory by individually optimizing the four 4-node subarcs, [nodes 1–4], [nodes 4–7], [nodes 7–10], and [nodes 10–13]. Each subarc is optimized subject to fixed boundary states at its two ends, except at nodes 1 and 13, where the boundary conditions (3) and (4) are enforced, respectively.

Step 4 (iterated). Randomly pick a starting node n_0 such that 1) $n_0 \ge 1$ and n_0 is less than the number of nodes in each subarc, i.e., $1 \le n_0 < 4$ in our example, and 2) n_0 is different from the value picked in the previous iteration. (Here it is understood that $n_0 = 1$ was picked in step 3, i.e., in the iteration previous to the first iteration in step 4.) Then improve the latest 13-node trajectory by individually optimizing subsequent 4-node subarcs as shown in step 3, but this time starting at node number n_0. In our example in Fig. 1, with $n_0 = 2$, this leads to the optimization of the three individual 4-node subarcs, [nodes 2–5], [nodes 5–8], and [nodes 8–11].

V. Remarks

In the following, the CATO algorithm is discussed in some greater detail.

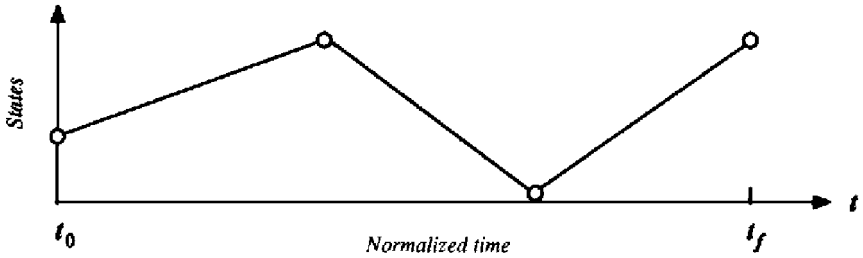
1) The purpose of steps 1 and 2 is to generate a 13-node starting trajectory under the assumption that the computer capability is limited to generating only a 4-node solution. By experience, steps 1 and 2 usually generate reasonable starting trajectories. In general, however, these starting trajectories may be far from optimal and need not even satisfy the (discretized versions of) constraints (2–6). Any method to generate a 13-node initial guess without exceeding the assumed 4-node computer capability could be used to replace steps 1 and 2.

2) Step 3 can be regarded as a special case of step 4, namely for the choice $n_0 = 1$.

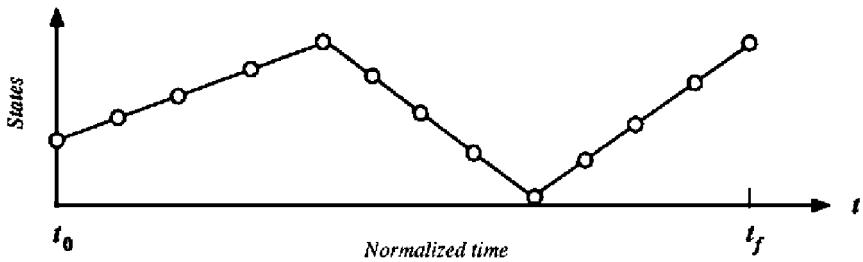
3) During the repetition of step 4, the choice of a starting node n_0 in the current iteration different from the starting node in the previous iteration guarantees the subsequent optimization of the trajectory at all nodes. Note that nodes that were fixed in the previous iteration are guaranteed to be optimized in one of the subsequent iterations.

4) In each step, the optimization of the individual subarcs is completely decoupled. Hence the CATO algorithm is well suited for parallel computing.

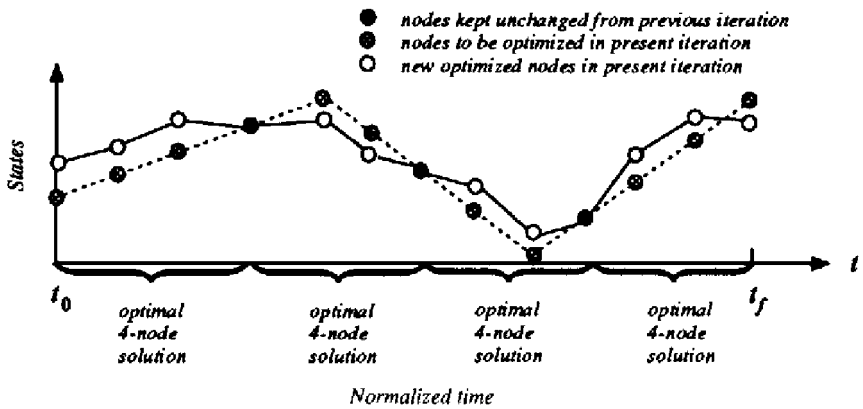
5) It is easy to verify that the CATO algorithm can also be applied to problems with nonequidistant node placement.



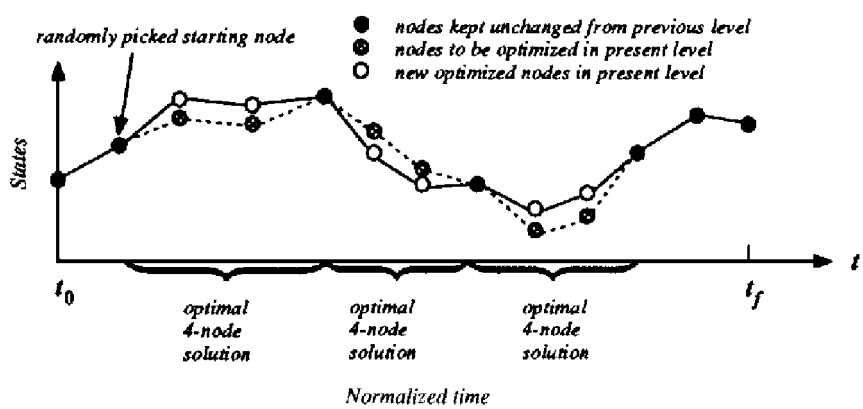
a) Step 1: solve the 4-node discretized problem



b) Step 2: linearly interpolate 4-node solution and obtain 13 node guess



c) Step 3: solve concatenated 4-node sub-problems



d) Step 4: randomly choose starting node n_0 with $0 < n_0 < 4$, say $n_0 = 2$, and re-solve concatenated 4-node sub-problems as in step 3

Fig. 1 Steps involved in the concatenated trajectory optimization approach for the example of a 13-node solution generated from 4-node subarcs.

6) A formal convergence proof is not given in this Note. It is observed, however, that each iterate of the CATO method, starting with the trajectory obtained in step 3, satisfies the conditions (2–6) within the discretization of a 13-node solution. Furthermore, the cost [in the sense of problem (1)] of subsequent iterates of the CATO method is nonincreasing. Finally, at least for fixed-final-time problems, the optimal solution to the overall problem furnishes a stationary point for the CATO iterations.

7) By experience it can be expected that the robustness of any given direct optimization method reduces when the number of design parameters is increased. As the CATO algorithm is based on the idea of solving a big problem by breaking it into a sequence of smaller problems, it can be expected that the overall robustness of the CATO algorithm is an improvement over solving the problem in a single run with a large number of parameters.

8) Stop criterion: The CATO algorithm is stopped once a stationary point is reached, i.e., once subsequent iterations do not lead to a significant change in the trajectory.

For engineering applications a more pragmatic strategy may be to simply stop the iteration after a certain time limit is reached and accept as the optimal solution the best iterate that has been generated. For example, in on-line robot path planning, a new maneuver could be shaped and iteratively improved until the robot has completed the current maneuver. This strategy is sensible as all iterates after step 2

of the CATO algorithm represent feasible solutions that satisfy all physical constraints even before complete convergence (in terms of cost) is achieved.

VI. Classes of Solvable Problems

An important class of optimal control problems that fits into the pattern described by (1–6) and Condition 1 are minimum-energy problems, i.e., problems with a cost function of the form with $L(x, u, t) = x^T Qx + u^T Ru$.

Another important class of problems suited for the CATO approach are autonomous minimum-time problems. Intuitively, this is clear, as for any two points A and B along a given (nonoptimal) trajectory, the final time is guaranteed to decrease if only the time that it takes to go from A to B is decreased. Clearly, in the practical application of the CATO algorithm to autonomous minimum-time problems, each subproblem in steps 3 and 4 of Fig. 1 would simply become an autonomous minimum-time problem in itself. Note that for this type of problem application of the CATO iterations, in general, leads to a nonequidistant node distribution even if the starting grid (in step 2 of Fig. 1) is equidistant.

VII. Numerical Example: F-15 Minimum Time to Climb

As stated in an earlier section, the results of this Note are generated using the direct method called trajectory optimization via differential inclusion (TODI)[2] to perform the optimization of short subarcs of the trajectory as required by the CATO algorithm. To solve the resulting nonlinear programming problems, the code NPSOL version 4.05 (Ref. 3) is used.

The precision of the problem discretization associated with the TODI algorithm is controlled through the user-chosen integer N, which represents the total number of nodes along the subarc to be optimized. In between neighboring nodes, the states are approximated by linear functions, and the controls (even though they do not appear explicitly in the TODI formulation) are considered constant. Higher-order discretizations are possible with the TODI approach, but in problems with discontinuous control jumps and complicated switching structures they suffer the same disadvantages as in collocation approaches.

For application of the CATO approach, M nodes are placed equidistantly along the total time interval associated with the full problem (with the first node at t_0 and the last node at t_f). Subsequently, in the framework of the CATO approach, subarcs of N ($N < M$) nodes are optimized. In the present study, we have chosen M and N such that $M-1$ is an integer multiple of $N-1$. Note that for minimum-time problems the CATO iterations cause the node placement, in general, to become nonequidistant even when the starting node placement is equidistant.

All calculations are performed on a SPARC 10 workstation. The CPU time requirements reported are satisfied without making use of the warm-start option in NPSOL. Through this option, the user can provide initial guesses for the active constraint set and for the associated Lagrange multipliers in addition to the initial guesses for the design parameters. It is expected that, especially in the later CATO iterations, when initial guesses are very good, the warm-start option would speed up the iterations significantly, but this has not been tested numerically. Further CPU time savings could be achieved, of course, through parallel processing.

We consider the problem of steering an aircraft from prescribed initial states at ground level to the dashpoint (the rightmost point on the level-flight envelope) in minimum time. Explicitly, the problem can be stated as follows:

$$\min t_f \tag{7}$$

subject to the equations of motion

$$\dot{E} = (\eta T - D)(v/W), \qquad \dot{h} = v \sin\gamma$$
$$\dot{\gamma} = (g/v)(n - \cos\gamma) \tag{8}$$

the control constraint

$$0 \le \eta \le 1 \tag{9}$$

and the boundary conditions

$$E(0) = 2668\,\text{m} \tag{10a}$$
$$h(0) = 5\,\text{m} \tag{10b}$$
$$\gamma(0) = 0\,\text{rad} \tag{10c}$$
$$E(t_f) = 38{,}029.207\,\text{m} \tag{10d}$$
$$h(t_f) = 12{,}119.324\,\text{m} \tag{10e}$$
$$\gamma(t_f) = 0\,\text{rad} \tag{10f}$$

The specific energy E (replacing the velocity v), the altitude h, and the flight-path angle γ are the state variables. The load factor n and the power setting η are the control variables. The variables m, g, $W = mg$, $T = T(E, h)$, and $D = D(E, h, n)$ are the vehicle mass, the Earth's gravitational acceleration, and the vehicle weight, thrust, and drag, respectively. The precise definitions of all quantities are given in Ref. 4.

The aim is to generate a 101-node CATO solution by iteratively optimizing 11-node subarcs, i.e., in terms of the definitions of the previous section, we choose $M = 101$ and $N = 11$. To generate an initial guess for step 1 of the CATO algorithm, we interpolate linearly the prescribed initial and final states defined in Eqs. (10). For estimates of the free final time between 10 and 1000 s the 11-node TODI approach always converges in fewer than 30 iterations and the required CPU time is always less than 2 s. The subsequent 11-node TODI runs in steps 3 and 4 of the CATO algorithm typically converge in fewer than 10 iterations and take less than 1 CPU second. For example, 10, 20, 40, and 100 CATO iterations are performed in 90, 170, 305, and 575 CPU seconds, respectively. The associated optimal final times are 268.94, 268.26, 268.04, and 268.02 s, respectively. Each 11-node TODI run involves solving a nonlinear parameter optimization problem with 35 parameters and 68 constraints. For this problem NPSOL version 4.05 requires a work array of dimension 9338.

For comparison, if a 101-node solution to problem (7–10) is generated directly using TODI (i.e., without the CATO approach), then the total number of parameters is 305 and the number of constraints is 608. The work array required by NPSOL version 4.05 is then of dimension 575,798. Using again as initial guess a linear interpolant of the boundary states defined in Eqs. (10), the convergence behavior is significantly less robust than in the 11-node case. Three test runs were attempted, with the initial guess $t_{f,\,\text{guess}}$ for the final time t_f set at 1) 200 s, 2) 268.104 s (which is the optimal final time), and 3) 300 s. Cases 2 and 3 did not converge. In case 1, convergence was achieved after 96 iterations and nearly 2033 CPU seconds. The obtained optimal final time is then 268.104 s. The variational solution of problem (7–10) yields an optimal final time of 268.198 s.

Figures showing the time histories of selected states can be found in Ref. 5.

VIII. Summary and Conclusions

A method was introduced to generate high-resolution, near-optimal, direct solutions to trajectory optimization problems without the usual requirement for high computing power. The method is based on the iterative improvement of a low-accuracy, possibly infeasible, starting trajectory on short, partly overlapping subintervals using standard direct optimization techniques. By experience, the overall rate of convergence to the optimal cost is only linear, but each CATO iteration is fast, and each iterate represents a feasible solution by itself, even before complete convergence in terms of cost is achieved. The class of problems that can be solved with this approach includes minimum-energy problems and autonomous minimum-time problems.

Acknowledgment

This work was supported by NASA Langley Research Center under Contract NAS1-20405.

References

[1]Lee, E. B., and Markus, L., *Foundations of Optimal Control Theory*, Krieger, Malabar, FL, 1986, p. 424.

[2]Seywald, H., "Trajectory Optimization Based on Differential Inclusion," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 3, 1994, pp. 480–487.

[3]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 Lab., TR SOL 86-2, Dept. of Operations Research, Stanford Univ., Stanford, CA, Jan. 1986.

[4]Seywald, H., "Optimal and Suboptimal Minimum Time-To-Climb Trajectories," AIAA Paper 94-3554, 1994.

[5]Seywald, H., and Kumar, R. R., "Concatenated Approach to Trajectory Optimization (CATO)," *Proceedings of the European Control Conference ECC 95* (Rome), 1995, pp. 2100–2105.

Life-Extending Control of Reusable Rocket Engines

Michael Holmes,* Sekhar Tangirala,† and Asok Ray‡
Pennsylvania State University,
University Park, Pennsylvania 16802

Introduction

THE goal of life-extending control is to achieve a tradeoff between structural durability and dynamic performance of complex mechanical systems such as aircraft, spacecraft, and energy conversion systems. Typically, a controller synthesis procedure guarantees the stability of a closed-loop system within specified uncertainty bounds and, at the same time, ensures that the performance specifications are satisfied. Performance is usually defined in terms of command signal tracking, disturbance rejection, and/or control effort minimization. However, in general, reduction of damage occurring in critical plant components that are vulnerable to structural stresses is not explicitly taken into account in synthesis of the control law. Work reported by Ray et al.[1] shows that, using optimized open-loop feedforward control sequences, it is possible to substantially reduce the damage rate and accumulation in critical plant components with no significant loss in dynamic performance. However, such a damage-mitigation procedure is based on extensive off-line optimization and is applicable only when the operational maneuver is known a priori. Tangirala[2] presented a combined feedforward/feedback controller synthesis methodology applied to a laboratory test bed that achieves life extension and enhanced performance by using optimal open-loop input sequences along with a damage-mitigating output feedback controller.

This Engineering Note presents the results of ongoing research in life-extending control and is a continuation of the results presented in previous publications. Specifically, this Note presents a procedure for synthesis of life-extending feedback controllers and explores the feasibility of its application to a reusable rocket engine such as the Space Shuttle main engine. However, for brevity, the dynamic model of fatigue damage, discussed in detail in a previous publication,[1] is not presented. As an example of life-extending control synthesis, the results reported focus on 1) reduction of fatigue damage in the blades of the oxidizer (O_2) and fuel (H_2) turbines and 2) enhancement of engine performance by simultaneously minimizing the turbine torques and the output tracking errors of the thrust chamber pressure and the O_2/H_2 mixture ratio. To realize different levels of performance/damage tradeoff, one performance controller and three damage-mitigating controllers are synthesized by using an induced L_2-norm method[3] applicable to sampled data systems.

Received Sept. 19, 1996; revision received Feb. 18, 1997; accepted for publication Feb. 19, 1997.

*Research Assistant, Department of Mechanical Engineering.

†Research Associate, Department of Mechanical Engineering.

‡Professor, Department of Mechanical Engineering, E-mail: axr2@psu.edu. Associate Fellow AIAA.

Synthesis of Life-Extending Feedback Controllers

This section summarizes a synthesis procedure for sampled-data output feedback life-extending control systems based on a linearized continuous-time model of a reusable rocket engine. A description of the rocket engine and derivation of the plant model are reported by Ray and Dai.[4] The plant model has 20 states and the two control inputs are the command signals to oxidizer valve actuators of the oxidizer (O_2) preburner and fuel (H_2) preburner. The inputs to the controller are the error in the thrust chamber pressure and the O_2/H_2 mixture ratio.

The main objective of the control synthesis is to achieve a tradeoff between high system performance and low fatigue damage rate and accumulation in critical plant components. This is accomplished by appropriate selection of the (frequency-dependent) weighting functions used for controller synthesis. This section presents the design of four different controller modules that yield different levels of performance/damage tradeoff. The performance controller (PC) is designed without taking damage into consideration. The three damage-mitigating controllers (DMC1, DMC2, and DMC3) demonstrate three levels of tradeoff between performance and structural damage.

The nominal plant model used for the controller design is obtained by linearizing the nonlinear model of a reusable rocket engine[4] at a chamber pressure of 17.58 MPa (2550 psi), which represents the midpoint of the pressure ramp-up range from 14.48 (2100 psi) to 20.69 MPa (3000 psi) considered here. Referring to Fig. 1, the design goal is to find a stabilizing discrete-time controller so that the induced L_2-norm of the transfer matrix from w to $[z_1 \;\; z_2 \;\; z_3]^T$ is minimized. We use the method of Bamieh and Pearson[3] for L_2-induced synthesis for sampled-data systems, which makes use of a lifting technique to take intersample behavior into account in the controller synthesis procedure. The sampled-data problem is recast in terms of an equivalent discrete-time H_∞ synthesis problem. The scheme for the sampled-data controller design is shown in Fig. 1. The sampling interval is chosen as 0.002 s, which is fast enough to capture the thermal-hydraulic dynamics of the rocket engine without any appreciable aliasing in the output signal.

The performance weighting functions for penalizing the deviations of the actual thrust chamber pressure and O_2/H_2 mixture ratio from their reference trajectories are selected as

$$W_{\text{press}}(s) = \frac{k_p(s+\alpha_p)}{s+\beta_p} \qquad \text{and} \qquad W_{O_2/H_2}(s) = k_0 \tag{1}$$

where for DMC1, $k_p = 2000.0$, $\alpha_p = 7.5$, $\beta_p = 30.0$, and $k_0 = 1.0 \times 10^5$; for DMC2, $k_p = 4000.0$, $\alpha_p = 7.5$, $\beta_p = 30.0$, and $k_0 = 1.0 \times 10^6$; for DMC3, $k_p = 8000.0$, $\alpha_p = 7.5$, $\beta_p = 30.0$, and $k_0 = 2.0 \times 10^6$; and for PC, $k_p = 500.0$, $\alpha_p = 0.5$, $\beta_p = 1.0$, and $k_0 = 2.0 \times 10^5$.

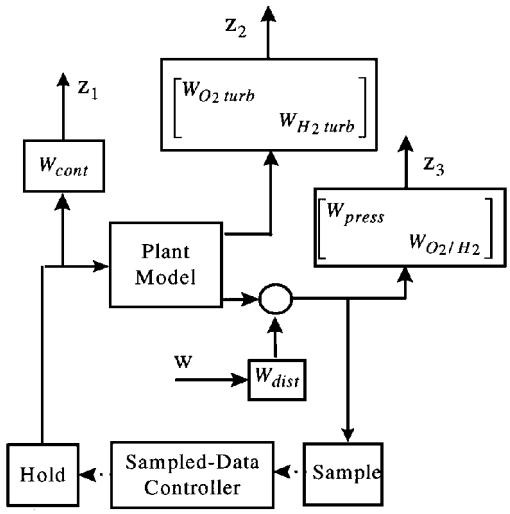


Fig. 1 Scheme for synthesis of the sampled-data controller.