Real-Time Optimal Guidance for Orbital Maneuvering

AUDREY O. COHEN* AND KENNETH R. BROWN†
IBM Federal Systems Division, Burlington, Mass.

A new formulation for soft-constraint trajectory optimization is presented as a real-time optimal feedback guidance method for multiburn orbital maneuvers. Simulations show sizable improvements in end-condition accuracy and burn time. Control is always chosen to minimize burn time plus a quadratic penalty for end-condition errors, weighted such that early in the mission (when controllability is greatest) terminal errors are held negligible. Eventually, as controllability diminishes, the method partially relaxes but effectively still compensates perturbations in whatever subspace remains controllable. Although the soft-constraint concept is well-known in optimal control, the present formulation is novel in addressing the loss of controllability inherent in multiple burn orbital maneuvers. Moreover the necessary conditions usually obtained from a Bolza Formulation are modified here so that the fully hard constraint formulation is a numerically well-behaved subcase. As a result convergence properties have been greatly improved.

Introduction

OPTIMAL rocket steering in a real-time feedback guidance environment was first shown computationally feasible[2] in 1965–1967. This was done in the context of Saturn booster injection into orbit, and the guidance problem was conceived essentially as follows: achieve a desired set of orbital parameters (usually 5) with minimum or near-minimum propellant expenditure. The nonoptimal alternatives (e.g., IGM) that the then new "optimal" algorithms were compared against tacitly assumed the same problem statement. It was taken for granted that all guidance laws, in order to be acceptable at all, must be such as to result in achieving the desired terminal parameters. One might tolerate a little waste of burn time in order to get an explicit or near-explicit computational scheme, and this was the basis of the IGM[1] approach, but end conditions were part of mission requirements and could not be sacrificed.

This attitude was tenable for the Saturn booster ascent problem largely because the burns were so brief that no significant real-time perturbations could arise within a mission except those due to irregularities in the thrust acceleration vector. To a large extent, whatever errors are caused by thrust acceleration errors can be corrected by later thrust acceleration compensation. So the real-time perturbations generally left the guidance problem solvable until only a few seconds (say 5 or 10) remained in the mission. At this point feedback guidance could safely be turned off and the last few seconds could be flown open loop without any significant loss in end condition accuracy. Thus the feedback guidance algorithm, whether optimal or not, would never need to be exercised in an environment where the terminal conditions were unachievable.

Other guidance problems in the same Apollo program are not so simple. Long sequences of orbital maneuvers are needed to get from low-altitude Earth-orbit to a lunar orbit. During such sequences, perturbations can arise such that the remainder of the mission cannot be flown to the desired end conditions without introducing additional burn arcs not present in the optimal flight plan, in other words without a discontinuous departure from the optimal flight plan. The answer to this problem in the Apollo program was to permit such discontinuous departures from optimality in the form of midcourse corrections. For a given limited "perturbation budget" there exists a finite number of midcourse corrections that can be guaranteed in advance to be adequate in the sense of achieving end conditions to within defined tolerances. But this is the most that can be had—defined tolerances. Even a perfect guidance computation scheme could not achieve end conditions precisely, no matter how many midcourse corrections were used.

The same difficulty besets the sequences of orbital maneuvers for the future space shuttle. Perturbations can arise that cannot be compensated for within a short burn arc; either more burn arcs must be added or the perturbation cannot be fully compensated. Allowing in advance for midcourse correction burns does improve the situation, but the fact remains that the feedback guidance scheme eventually reaches a point where it must be turned off to prevent numerical blow up due to absence of exact solutions.

The basic problem here is loss of controllability as time-to-go becomes small. Missions requiring a full set of 6 or even 5 terminal constraints gradually become singular problems as $t \to t_f$. If we define sensitivities as the ratios of control changes to resulting changes in end conditions, then sensitivities become infinite as time-to-go approaches zero. Certain subspaces of the space of terminal conditions become uncontrollable later than others—velocity remains controllable longer than position—but eventually even velocity becomes difficult to vary except in the direction of the nominal thrust vector. The problem is mild enough in Saturn booster orbital injection to permit simple opening of the feedback guidance loop shortly before t_f. The problem is much more serious in Apollo orbital maneuvers, but the expedient of inserting midcourse corrections permits the part of the feedback guidance scheme that operates during burns to restrict itself to a 3-constraint problem tantamount to a velocity-only problem so that again the feedback loop can be closed throughout most of each burn.

There is nothing to prevent this same approach being applied to space shuttle orbital maneuvers with comparable results: namely, divergence is avoided, and end condition errors are kept within reasonable limits.

But the whole orientation of this approach is wrong from the start. It assumes the basic correctness of the original problem statement and tries to patch and adjust the context of this problem statement to give the illusion that it is being solved. What is needed instead is a basic revision of the original problem statement. It is nonsense to speak of minimizing burn time (or anything else) subject to hard constraints on terminal conditions, if the hard constraints are unachievable in the first place.

The necessary revision is not far to seek: clearly a Bolza formulation is called for. If the exact end conditions are un-

---

Presented as Paper 73-249 at the AIAA 11th Aerospace Sciences Meeting, Washington, D.C., January 10–12, 1973; submitted January 26, 1973. Research supported by NASA Marshall Space Flight Center, Huntsville under Contract NAS8-24018.

Index categories: Navigation, Control, and Guidance Theory; Spacecraft Navigation, Guidance, and Flight-Path Control Systems.

* Staff Engineer.

† Development Engineer.

attainable, we obviously must settle for coming close in some sense to meeting them—the closer the better. And since burn time is still important too, we must tradeoff some kind of "distance" from the desired terminal conditions against burn time. If performance formerly was burn time J we need a new performance index J' given by

$$J' = J + \phi(x_f)$$

where ϕ is a function which is minimized when the terminal constraints are satisfied and which increases with some kind of "distance" from the terminal conditions.

The problem of defining ϕ is nontrivial and in fact arbitrary with respect to the original problem statement. That is why the original problem statement was inadequate. It did not recognize the possibility that constraints might be unachievable so it left open the question of which constraints are more important than others or how important burn time is in comparison to the constraints.

Even when no attempt is made to develop an "optimal" guidance algorithm, it is necessary to decide upon a ϕ function if only in order to meaningfully tune suboptimal schemes or compare the performance of alternative suboptimal schemes. It is not possible to compare two alternatives with respect to a vector of performance indices. A single scalar performance index is necessary for meaningful evaluation. Although some design efforts proceed without explicit designation of a single performance index, a designation is nonetheless implicit in the selection of one algorithm over its competitors and in the tuning of the free parameters of the algorithm chosen. Serious effort is required to define the performance index explicitly, but the effort is rewarded by a real gain in ability to effect design improvements.

There is no algorithmic approach to the problem of defining ϕ. We may of course use algorithms and experiments to obtain information about ϕ from other information that indirectly reflects on ϕ. For example, information to the effect that certain size errors in specific terminal conditions are for practical purposes negligible and that a certain level of increase in burn time is negligible, partially defines ϕ. This kind of information is important. It may be time consuming to extract, but unless an effort is made, there is no way to define ϕ and hence no way to obtain a truly adequate problem statement.

This report does not claim to have carried out the necessary effort to define ϕ, for Apollo missions, space shuttle missions, or any other specific case. Instead, it demonstrates the kind of improvements that can result from such an effort by experimenting with a single (reasonable) definition of ϕ for a representative coast-burn-coast-burn gross rendezvous mission for the space shuttle. The fact that the J' chosen for the experiments may not represent exactly what space shuttle mission designers would finally settle on as a valid performance index is irrelevant. The important point is that once a valid ϕ and hence J' are chosen, the guidance problem becomes much clearer and more manageable, and an optimal guidance approach emerges with better real-time performance than any scheme previously considered.

Multiarc Fuel-Optimal Guidance

The basic fuel-minimization problem for space vehicles is to use thrust direction $u/|u|$ and rate of mass loss $|\dot{m}|$ to satisfy mission objectives with a minimum expenditure of rocket fuel. The requirements for real-time fuel-optimal guidance are normally much more stringent than those for onboard optimal targeting or for optimization in a research laboratory. In addition to having reasonable storage requirements, an optimal guidance algorithm must have a short enough cycle time and a large enough region of near linear convergence to reconverge to the optimal control policy after perturbations in engine thrust and in knowledge of vehicle and target states. During each cycle, state must be propagated to final time and the effect of variations in control upon boundary conditions must be determined by either finite-difference or variational methods. Guidance performance toward the end of a mission is particularly critical because loss of control as $t \to t_f$ normally causes the near-linear region of convergence to shrink more rapidly than the cycle time. Thus, a guidance algorithm which is adequate during the first few arcs of a mission, may be unable to correct control policy rapidly enough to compensate for perturbations occurring during the final burn arc. In fact, unless precautions are taken to keep corrections to control in the near-linear region, guidance algorithms can diverge.

At least four design factors affect the real-time performance of an optimal guidance algorithm. First, the choice of an optimization method such as a neighboring extremal method, a gradient method, or a quasi-linearization method impacts both the speed and storage requirements of the algorithm. The dynamical formulation used to calculate final state and partial derivatives of boundary conditions is also important. The algebraic form used to express constraints and necessary conditions for optimality can significantly alter convergence. Finally, real-time performance can be significantly affected by the performance index and the hardness of constraints. Results showing the improvements in simulated real-time performance gained by considering this last design factor are presented in this paper, and the effect of the other design factors is reviewed to place these results in perspective.

Optimization Method

A "shooting" method of the neighboring extremal class of algorithms gives good results for real-time optimal guidance.[4] Storage requirements are minimal since initial costate and switching times rather than control $u(t)$ and $|\dot{m}(t)|$ throughout the mission are stored. At each guidance cycle, new information concerning engine performance and vehicle state are included to propagate state and costate to the estimated time t_0 for the next navigational update. Then state x and costate p are propagated to final time t_f and the effect of small variations in $p(t_0)$ and in switching times T upon boundary conditions are calculated. A Newton Raphson method can be used to calculate changes in $p(t_0)$ and T which zero errors in end conditions on a linearized basis. Control $u(t_0)$ and $|\dot{m}(t_0)|$ are determined from x and updated p and T.

$$u(t_0) = -s(t_0) \quad \text{where} \quad p^T = (q^T, s^T) \tag{1}$$

This method is faster than either gradient or quasi-linearization methods. Initialization is readily accomplished starting from an impulsive solution.

Dynamical Formulation

Expressing vehicle state in terms of geocentric, cartesian coordinates simplifies the state, costate, and variational equations for the problem. The equations of motion for state $x^T = (r^T, v^T)$ are

$$\begin{aligned} \dot{r} &= v \\ \dot{v} &= -F(r) + c|\dot{m}|u/m|u| \end{aligned} \tag{2}$$

where r is position, v is velocity, m is mass, c is exhaust velocity, and $F(r)$ is gravitational acceleration. Here, thrust acceleration is assumed proportional to $|\dot{m}|$, and $|\dot{m}|$ is bounded ($0 \leq |\dot{m}| \leq \beta$). Costate is propagated according to $\dot{p}^T = -p^T(\partial\dot{x}/\partial x)$ and variational equations for $(d/dt)(\partial x/\partial p_0, T)$ and $(d/dt)(\partial p/\partial p_0, T)$ are reasonably simple.[4] During burn arcs ($|\dot{m}| > 0$) these equations are readily integrated using standard numerical methods. During coast arcs ($|\dot{m}| = 0$), state, costate, and variational variables can be propagated analytically[3] if $F(r) = \mu r/r^3$. If higher order terms are included in gravitational acceleration, a variation of parameter method can be used to propagate state and costate efficiently through coast arcs.

Necessary Conditions

If the coasting motion is periodic and constraints on end conditions are independent of time, the number of coast arcs in a mission must be constrained for the optimization problem to be well-posed. Otherwise, the vehicle can coast through an orbit during the mission without changing either fuel consumption or end conditions. Normally, a sequence of coast arcs ($|\dot{m}| = 0$) and burn arcs ($|\dot{m}| > 0$) is specified in advance, and the switching times for transfering from one arc to another are optimized. Missions can begin with either coast or burn arcs, but they are assumed to end on burn arcs.

As shown in Ref. 1, the problem is equivalent to a boundary value problem in which six components of initial costate p and l switching times $T^T = (t_{7-l}, \ldots, t_6)$ are chosen such that $l+6$ boundary and necessary conditions are satisfied. Control u is chosen according to $u = -s$, and $|\dot{m}|$ is chosen to be 0 on coast arcs and at the maximum β on burn arcs.

Six of the necessary conditions for optimality are derived from the desired end conditions for the mission, while the other l conditions are related to the number of free coast and burn arcs. The necessary conditions which are associated with switching times are derived from the constancy of the Hamiltonian $H = L + p^T \dot{x}$ to insure that the switching function $S = 1 - w - c|s|/m$ is zero when going on or off a burn arc.[4] Here, w is costate adjoint to vehicle mass. These necessary conditions are

$$|s(t_6)| = m/c$$
$$T_v(t_6) - T_v(t_5) = 0$$
$$|s(t_5)| - |s(t_4)| = 0$$
$$T_v(t_4) - T_v(t_3) = 0$$
$$\vdots$$

where $T_v = q^T v - \mu(s^T r)/r^3$, t_6 is final time, the switching times are $T^T = (t_{7-l}, \ldots, t_6)$, and the last arc of the mission is assumed to be a burn arc. During a mission, as real time exceeds each switching time, it is eliminated from the control vector of switching times T and the corresponding necessary condition is dropped.

These l necessary conditions have two advantages for guidance: they have good convergence properties and they do not involve vehicle mass m explicitly. When m does not appear in the necessary conditions for optimality, it is unnecessary to propagate its adjoint costate w and $l+6$ variational variables through the mission.

Hard Constraints on Final State

When the fuel-minimization problem involves hard constraints on final state, a Lagrange formulation can be used. The cost functional

$$J = \int_{t_0}^{t_f} |\dot{m}|\, dt \tag{3}$$

is minimized subject to k hard constraints $C(x_f, t_f) = 0$ on final state. This minimization is equivalent to choosing initial costate $p(t_0)$ and l switching times T so that both the l necessary conditions described in the preceding section and following six conditions are satisfied:

k constraints

$$C(x_f, t_f) = 0$$

6-k transversality conditions (4)

$$A^T p_f = 0$$

The transversality conditions, which optimize J over any degree of freedom in end conditions, are constructed from the 6 by 6-k matrix A which is orthogonal to the gradient of the constraints

$$C_x A = 0$$

and which spans the 6-k degrees of freedom in state space orthogonal to C_x. For a rendezvous mission, Eq. (4) consists of six constraints and no transversality conditions. For an orbit injection mission in which five orbital constants are constrained, Eq. (4) consists of five constants and the transversality condition $A^T p_f = T_v(t_f) = 0$.

Soft Constraints on Final State

During a real-time guidance environment the exact satisfaction of constraints may be both undesirable and impractical. Normally the actual objective of a mission is to minimize mass loss while satisfying constraints within certain tolerances. $|C_i(x_f, t_f)| \leqq \varepsilon_i$ for $i = 1$ to k. However, the magnitude function has discontinuous first derivatives and is thus inappropriate for numerical guidance. The simplest function which both maintains the continuity of derivatives of $C(x_f, t_f)$ and permits soft constraints is $\phi = \frac{1}{2} C^T W C$ where W is a k by k matrix. A Bolza formulation can be used to implement these soft constraints with a cost functional

$$J' = \int_{t_0}^{t_f} |\dot{m}|\, dt + \tfrac{1}{2} C^T W C \tag{5}$$

where W is a positive, semidefinite matrix. Using this formulation, the six constraints on final costate are $p_f = (\phi_x)^T$ or

$$p_f = (C_x)^T W C \tag{6}$$

If the preceding six end conditions are implemented directly, the convergence characteristics of the algorithm can not necessarily be made to approach those for the Lagrange formulation as components of W are increased. In fact, numerical tests indicate that the direct implementation of Eq. (6) for a rendezvous mission with $W \propto I_{6\times6}$ seriously degrades the convergence properties of the guidance algorithm. An alternative formulation is available, however, which does not degrade convergence properties.

The problem being optimized is unchanged if C and W in Eqs. (5) and (6) are replaced by a set of six constraints, augmenting a 6-k vector C' to C,

$$C'' = \begin{pmatrix} C \\ C' \end{pmatrix}$$

and a 6×6 weighting matrix

$$W'' = \begin{pmatrix} W & 0 \\ 0 & 0 \end{pmatrix}$$

respectively. If C' is chosen so that the row vectors of $(C'')_x$ span state space, then a matrix B^T exists which is the inverse of $[(C'')_x]^T$ and Eq. (6) can be replaced by

$$B^T p_f = W'' C'' \tag{7}$$

The B matrix can be interpreted both as the inverse of $(C'')_x$ and as the partial derivative of state with respect to the constraints. For diagonal weighting matrices, Eq. (7) is conveniently replaced by

$$(I - \theta) B^T p_f = \theta C'' \tag{8}$$

where

$$(I - \theta) W'' = \theta$$

The six conditions in Eq. (8) are clearly equivalent to those for the Lagrange formulation in Eq. (4) when θ is diagonal with 1's and 0's on the diagonal. Equation (8) is thus seen to represent a general formulation of the optimization problem which is valid and well-posed for both hard and soft constraints. As shown by the simulation results presented in this paper, use of Eq. (8) with an appropriate choice of θ can produce marked improvements in guidance performance.

Choice of Constraints

The choice of constraint functions C normally affects the convergence properties of a guidance algorithm. Ideally, components of C should be as linear as functions of initial costate and switching times as possible. Components of C and C_x should also be finite, well-behaved functions and the rows of C_x should be linearly independent.

Orbital constants are intuitively appealing candidates for components of C because they are linear functions of p_0 along coast arcs. However, when classical sets of orbital constants such as a, e, i, ω and Ω are used, C_x is not well-posed for certain values of state (i.e., circular orbits). However, the components of angular momentum per unit mass

$$h = r \times v \tag{9}$$

together with the two components of the eccentricity vector

$$e = -r/r - h \times v/\mu \tag{10}$$

which lie in the plane perpendicular to h, form a set of five orbital constants whose gradients are linearly independent and well-posed for all realistic values of state. Numerical tests have indicated that algorithms using these functions for C have good convergence properties. Moreover, coordinate orientations are available which decouple these five components in terms of controllability as $t \to t_f$. Since velocity is more readily controllable with thrust direction than position near the end of a mission, unit Cartesian vectors, $\hat{r}$, $\hat{h}$, and $\hat{k}$ oriented along nominal final position $r_f{}^0$, nominal angular momentum $h_f{}^0$, and $h_f{}^0 x r_f{}^0$ are intuitive choices for decoupling end conditions with respect to control. Moreover, with this choice of coordinates, the components of e lying perpendicular to h are approximately $\hat{e}^T r$ and $\hat{e}^T k$. Since differences in orbital energy α can cause two vehicles to drift further apart with each successive orbit, α is a desirable quantity to control. Replacing $h^T h$ by α does not significantly impact convergence properties. A constraint on phasing such as $\Delta r^T \hat{k}$ can be included to obtain a set of six constraints whose gradients span state space.

$$\begin{aligned} C_1 &= (\Delta h^T \hat{k})/h_f{}^0 \\ C_2 &= \Delta\alpha (r_f{}^0)^2/\mu \\ C_3 &= (\Delta e^T \hat{k})|r_f{}^0| \\ C_4 &= (\Delta h^T \hat{r})/h_f{}^0 \\ C_5 &= (\Delta e^T \hat{r})|r_f{}^0| \\ C_6 &= \Delta r^T \hat{k} \end{aligned} \tag{11}$$

Here, components of C have been scaled by constants $|r_f{}^0|$ and $|h_f{}^0|$ to convert them to units of length, $\hat{r} = r_f{}^0/|r_f{}^0|$, $\hat{k} = (h_f{}^0 \times r_f{}^0)/|(h_f{}^0 \times r_f{}^0)|$, and the operator Δ on a quantity indicates the actual value minus the desired value of the quantity.

A guidance algorithm using the constraints in Eqs. (11) has shown good convergence behavior for both rendezvous and orbit transfer missions. (For orbit transfer, the phasing constraint was relaxed by zeroing the last row and column in the weighting matrix θ.)

The gradients of the constraints in Eqs. (11) and the transversality vector $B^T p$ are listed in Appendix B.

For rendezvous missions, errors in position and velocity are a simpler but less meaningful choice for constraints than the ones in Eqs. (11)

$$C = \begin{pmatrix} r - r_{\text{target}} \\ v - v_{\text{target}} \end{pmatrix} \tag{12}$$

Constraints on r and v are less meaningful because orbital energy cannot be readily emphasized. Although the constraints in Eqs. (11) and (12) have about the same convergence properties for rendezvous missions, the state constraints in Eq. (12) are inappropriate for orbit transfer missions.

Simulation Results

A simulation program was written to analyze the effect of softening mission constraints upon guidance performance for a gross rendezvous mission. Results for the Lagrange cost functional in Eq. (3) were compared with those for the Bolza cost functional in Eq. (5), and the effects of abruptly relaxing hard-to-control constraints during the mission was also simulated. The constraints on orbital constants and phasing in Eqs. (11) were used for these comparisons. Two additional simulations were run with the Bolza cost functional in Eq. (5) and the state constraints in Eq. (12).

Details of Real-Time Guidance Simulation

The mission simulated was a shuttle vehicle with $m = 2.6 \times 10^5 \times 46.253$ kg. $\beta = |\dot{m}| = 45.045 \times 46.253$ kg/sec, and $c|\dot{m}| = 2 \times 10^4 \times 0.45359$ km kg/sec^2 in a 100 by 150 naut mile elliptical orbit maneuvering to gross rendezvous with a target in a coplanar 200 naut mile circular orbit. The initial phasing of the shuttle and target vehicles was such that an impulsive two-burn Hohman transfer would have led to an exact rendezvous. A modified version of the GUIDE 71/6 program was used for the simulation. The guidance algorithm was initialized by using a Hohman transfer solution for $p(t_0)$ and $T^T = (t_3, t_4, t_5, t_6)$ and iterating three times. Here, t_3 and t_4 were switching times at the beginning and end of the first burn arc, and t_5 and t_6 were switching times at the beginning and end of the final burn arc. Variations in engine performance and real-time errors in knowledge of shuttle and target state were simulated by introducing random perturbations in the velocities of the shuttle and target at different points throughout the mission. The random perturbations were introduced in both shuttle and target velocity by adding random accelerations multiplied by the length of time between perturbations. During both coast and burn arcs, random accelerations represented effects such as higher order terms in the geopotential and had a standard deviation of 5×10^{-5} g's. During burn arcs random accelerations with $\sigma = 0.05 \times (c|\dot{m}|/m)$ were included in the perturbations to represent engine fluctuations. Each time a set of random perturbations was introduced, the results from a single iteration of the guidance algorithm were used to update control. The random perturbations were introduced every 100 sec during coast arcs, every 20 sec during burn arcs, 10 sec before t_3 and t_5, 10 sec after t_4, 2 sec after t_3 and t_5, and 2 sec before t_4 and t_6. The single iteration of the guidance program taken after each perturbation required considerably less than one sec of cpu time for execution, and was thus consistent with the minimum interval between calls to the guidance routine. The execution time per guidance cycle became shorter toward the end of each mission simulation both because of the reduction in number of switching times affecting end conditions and because of the shorter time interval for Runge Kutta integration.

Some of the perturbations were large enough to be well outside the linear region of convergence for the algorithms using hard constraints. Under these conditions, if the full change in costate Δp and switching times ΔT calculated by the guidance algorithms was accepted, shuttle state projected to t_6 would diverge from target state and rendezvous would not be achieved. In order to prevent the guidance algorithm from actually degrading end conditions when the perturbations were large, only that fraction ε of Δp and ΔT was accepted which was consistent with a simple empirically derived rule for judging linearity: the largest value of ε between 0 and 1 was selected such that each component of s changed by no more than $0.1|s|$ and such that all changes in switching times were ≤ 2 sec. Experience had shown that allowing either large changes in $|s|$ or in the switching times tended to degrade end conditions rather than improve them. The size of ε appeared to be a reasonable indication of the ability of the guidance algorithms to cope with perturbations.

After each perturbation, the error in end conditions was calculated before and after updating $p(t_0)$ and T by $\varepsilon\Delta p(t_0)$ and $\varepsilon\Delta T$. Feedback delays were ignored in calculating end condition errors so that these errors would be zero if end conditions were linear functions of $p(t_0)$ and T. Prior studies have shown this makes no appreciable difference in final errors if the guidance cycle (feedback delay time) is less than a few seconds long. Since all the guidance algorithms being compared had essentially the same execution time, including feedback delay would have had about the same effect on each of them. Moreover, omitting complicating factors such as feedback delay, emphasizes any tendencies of a guidance algorithm to be unable to control end conditions when given perfect information about state.

Comparison of Simulated Guidance Performance

Five different simulation runs were made. In simulation A fuel was optimized subject to six hard constraints on final state. The Lagrange formulation with orbital and phasing constraints

Table 1 Comparison of simulation results

Run	Time, sec	t_4-t_3, sec	t_6-t_5, sec	ε	C_1	C_2	C_3	C_4	C_5	C_6
A	Initial	70.935	34.703	1.0	0	-10^{-12}	-10^{-13}	0	-10^{-12}	10^{-12}
B	Initial	70.935	34.703	1.0	0	10^{-12}	10^{-13}	0	10^{-12}	10^{-12}
C	Initial	70.924	34.691	1.0	-10^{-17}	15	-10^{-7}	-10^{-17}	10^{-5}	10^{-9}
D	Initial	70.930	34.671	1.0	0	26	-10^{-8}	0	38	-10^{-8}
E	Initial	70.932	34.682	1.0	0	16	-10^{-6}	0	24	−29
A	t_3-10	70.852	34.720	1.0	−25	1	−0.2	−0.6	−1	−1
B	t_3-10	70.852	34.720	1.0	−25	1	−0.2	−0.6	−1	−1
C	t_3-10	70.782	34.702	1.0	2	15	0	51	−0.3	0.2
D	t_3-10	70.814	34.683	1.0	−3	26	0	17	38	0.4
E	t_3-10	70.790	34.693	1.0	2	16	0	51	24	−28
A	t_4-2	70.591	34.741	0.21	1200	−480	550	26	980	2600
B	t_4-2	70.746	34.729	0.097	1500	−590	640	33	1200	3300
C	t_4-2	69.748	34.725	0.41	99	63	76	88	−67	−740
D	t_4-2	70.421	34.713	0.31	940	−370	350	43	850	−2500
E	t_4-2	69.740	34.718	0.33	140	73	99	88	−73	860
A	t_5-10	70.591	34.855	0.005	1300	−550	840	−57	1100	2400
B	t_5-10	70.746	33.858	1.0	−0.1	0.03	−0.2	−20	2700	4000
C	t_5-10	69.748	34.712	1.0	59	12	−53	30	140	−97
D	t_5-10	70.421	34.892	0.024	990	−510	500	−50	790	−2200
E	t_5-10	69.740	34.735	0.076	200	−17	320	17	61	580
A	t_6-2	70.591	34.855	10^{-8}	3100	−1100	900	−36	−100	2400
B	t_6-2	70.746	33.362	0.14	830	−130	52	−7	2400	4000
C	t_6-2	69.748	33.935	0.36	860	−35	4	43	190	−100
D	t_6-2	70.421	35.037	0.05	2500	−1200	270	−30	−520	−2200
E	t_6-2	69.740	34.192	0.35	890	−62	390	29	−42	580

from Eq. (11) was used. (Using state constraints from Eq. (12) did not change the results.) Simulation B was identical to simulation A during initialization and the first coast arc in the mission. The constraint on phasing C_6 was relaxed at the beginning of the first burn arc, and the constraints on C_4 and C_5 (components of h and e parallel to r_f^0) were relaxed at the end of the first burn arc. Thus, during the last two arcs of simulation B, only the first three orbital constants in Eq. (11) were constrained. Simulation runs C, D, and E used the Bolza formulation with performance index $J = \int |\dot{m}|\, dt + \phi$. The orbital constants in Eq. (11) were used to formulate ϕ for run C

$$\phi_C = \frac{\beta}{2}\left(C_1^{\,2} + 10^2 C_2^{\,2} + \sum_{i=3}^{6} C_i^{\,2}\right)$$

Here, the C_i are in units of km and β = maximum $|\dot{m}|$. Orbital energy was weighted more heavily because an error in it causes the relative vehicle positions to change from orbit to orbit. Thus, a 0.005-sec decrease in burn time was judged to be worth a 100 m error in C_3, but only worth a 10 m error in C_2. In simulation runs D and E the state constraints Δr and Δv in Eq. (12) were used to construct constraint functions similar to ϕ_C. For run D, all components of position were weighted equally and these weights were roughly adjusted to imitate the initial tradeoff between energy and burn time in run C. In run E those components of position and velocity which were contributing more to final energy were weighted more heavily. Constraint functions for runs D and E were

$$\phi_D = (\beta/2)(10^2 \Delta r^2 + 10^8 \Delta v^2)$$

and

$$\phi_E = \frac{\beta}{2}(\Delta r_1^{\,2} + \Delta r_2^{\,2} + 10^2 \Delta r_3^{\,2}) + \frac{\beta |r|^2}{2|v|^2}(\Delta v_2^{\,2} + \Delta v_3^{\,2} + 10^2 \Delta v_1^{\,2})$$

Since C_x^{-1} was not diagonal, a nondiagonal weighting matrix would have been required to formulate a constraint function from Δr and Δv which was equivalent to ϕ_C.

Table 1 illustrates guidance performance for simulation runs A–E at five times during the mission. t_4-t_3 and t_6-t_5 are the proposed (or past) lengths of the first and second burn arcs at each of these times, while ε is the empirical scaling factor which is used to keep control updates within the region of linear convergence. $C_1, \ldots, C_6$ are the errors in end conditions at final time which would occur if no further control updates or perturbations in state occurred. As indicated in Eqs. (11), C_1 and C_4 are related to errors in the orientation of the orbital plane while C_2 is related to the error in orbital energy. C_3 and C_5 are related to the ellipticity and orientation of perigee in the orbital plane, and C_6 is the phasing difference between shuttle and target vehicles. All errors in end conditions are scaled so that they have units of meters.

As shown by the results for run A in Table 1, all six constraints can be satisfied exactly during the initialization phase of the mission, but they cannot be satisfied on a real-time basis during the entire mission. By the end of the first burn arc (t_4-2), noticeable errors in end conditions have appeared for run A, while by the end of the second coast arc (t_5-10), the algorithm with hard constraints on all six end conditions is completely unable to cope with perturbations. Abruptly relaxing the phasing constraint (run B) at the end of the first coast appears to create a transient and harm rather than improve guidance behavior during the first burn arc. However, once the transients in run B due to changing mission objectives have settled out, the guidance algorithm manages to constrain the first three orbital constants throughout the rest of the mission.

The Bolza formulation used for runs C, D, and E significantly improves guidance behavior when weights are appropriately chosen. Initially, small errors in end conditions are accepted to save fuel, while during the mission only corrections to control which do not significantly increase fuel consumption are accepted. The results for run C in Table 1 show particularly good guidance performance throughout the entire mission. The poorer guidance behavior shown on run D appears to be due to an inappropriate choice of weights rather than to any poor convergence properties of the constraint formulation in Eq. (12). Changing weights significantly alters guidance performance within pairs of runs such as A and C with constraints from Eq. (12). Runs such as C and E with different constraints but comparable weights have comparable guidance behavior.

Table 2 Performance comparison

Simulation run	Final burn time + ϕ_C/β in sec	$(C^TC)^{1/2}$ in km at end of mission	Fuel saved over run A, kg
A	174.1	4.2	0
B	116.1	4.7	2780
C	104.1	0.88	3670
D	183.3	3.6	−25
E	104.8	1.13	3160

Table 2 summarizes different aspects of the guidance performance of the algorithms using hard, step-wise-relaxed, and soft constraints for the simulated mission. Run C which used soft constraints from Eqs. (11) with an emphasis on energy errors clearly out performed the hard and step-wise-relaxed constraint algorithms both in terms of fuel savings and errors in end conditions. Run D used more fuel during the simulation because its performance index placed too great an emphasis on achieving all components of desired r and v, and thus encountered controllability problems earlier in the mission. The soft constraint algorithms used in runs C and E achieved better guidance performance because their performance indices kept control updates within the linear region for a greater part of the mission.

Conclusions

The simulation results presented in the preceding section show the importance of a realistic problem definition for real-time optimal guidance. When gross rendezvous was defined as a fuel minimization problem with hard constraints, the guidance algorithm was unable to cope with simulated perturbations during the last 25 min of the mission. A systematic relaxation of uncontrollable constraints improved guidance behavior, but still did not solve the basic problem. It was nonsense to minimize burn time subject to hard constraints when the constraints were unachievable in the presence of perturbations. When the gross rendezvous problem was redefined as one in which burn time plus a function of errors in terminal end conditions was minimized, a significant improvement in guidance performance was obtained.

The penalty function formulation derived in this paper can be used for a variety of rendezvous and orbit transfer problems. The values of components of the weighting matrix θ for this formulation determine the hardness of the constraints, the nature of the mission, and the real-time guidance performance of the algorithm. Thus, for example, setting θ equal to the identity matrix leads to a rendezvous mission with hard constraints, while zeroing the weight on phasing produces an algorithm for orbit injection. Since the hardness of mission constraints profoundly affects guidance performance, simulations of real-time missions should be used to tune the weighting matrix. If an algorithm is to be used for real-time optimal guidance, it must try to achieve an objective which is reachable in the presence of perturbations. With a properly tuned θ matrix, real-time fuel-optimal guidance is feasible for all but the last few seconds of space missions.

Appendix A: Transversality Condition Theorem

If constraints and transversality conditions are constants of the motion along coast arcs, then only the change in thrust level affects them, and computer code can be simplified by treating derivatives of C and B^Tp with respect to intermediate and final times uniformly. It is therefore desirable to know whether or not transversality conditions such as those in Eqs. (4, 7, and 8) are constants of the motion. The following theorem indicates how transversality conditions depend upon time.

Theorem

Consider the n dimensional state and costate equations

$$\dot{x} = f \quad \text{and} \quad \dot{p}^T = -p^T f_x \tag{A1}$$

and an n dimensional set of constraints $C(x, t)$ whose gradients span state space. If C_x and f are differentiable and C_x has a differentiable inverse $C_x^{-1} = B$, then

$$(d/dt)(p^TB) = -(p^TB)\{\partial\dot{C}/\partial x\}B \tag{A2}$$

Proof

We have

$$d(p^TB)/dt = p^T\dot{B} + \dot{p}^TB = p^T(\dot{B} - f_xB) \tag{A3}$$

from Eq. (A1).

In order to simplify this expression, recall that $B = C_x^{-1}$ so that

$$C_xB = I \tag{A4}$$

and

$$[(d/dt)(C_x)]B + C_x\dot{B} = 0$$

Hence,

$$\dot{B} = -(C_x)^{-1}\left[\frac{dC_x}{dt}\right]B = -(C_x)^{-1}\left[C_{xx}f + \frac{\partial C_x}{\partial t}\right]B \tag{A5}$$

Now, in order to replace the term $C_{xx}f + C_{xt}$ in Eq. (A5), consider that

$$(\dot{C})_x = (\partial/\partial x)\{C_xf + C_t\} = C_{xx}f + C_xf_x + C_{tx}$$

Thus, since $C_{tx} = C_{xt}$,

$$C_{xx}f + C_{xt} = (\dot{C})_x - C_xf_x \tag{A6}$$

Combining Eqs. (A5) and (A6)

$$\dot{B} = f_xB - (C_x)^{-1}(\dot{C})_xB$$

When this expression is substituted into (A3) and $(C_x)^{-1}$ is replaced by B, one obtains (A2). Q.E.D.

Applications

When all components of $C(x, t)$ are constants of the motion, Eq. (A2) shows that the transversality conditions (p^TB) are also constants of the motion. All but the last component of the constraint vector in Eqs. (11) are constants of the motion. For this set of constraints $C^T = (0, 0, 0, 0, 0, \Delta v^T\hat{k})$ and

$$d(p^TB)/dt = -|r|[T_v(t_6)/h](0^T, \hat{k}^T)B \tag{A7}$$

where $|r|T_v(t_6)/h$ is the sixth component of B^Tp at t_6. Thus, as $T_v(t_6) \to 0$ the transversality conditions corresponding to the constraints in Eqs. (11) become constants of the motion. For orbit transfer missions using this formulation, $T_v(t_6) \to 0$ and the components of (p^TB) become constants of the motion as a solution is approached. Moreover, as seen from the structure of B in Appendix B, the first and fourth components of B^Tp are always constants of the motion when $C(x, t)$ from Eqs. (11) is used. For near circular orbits, $(\hat{k}^Tr) \simeq 0$ and the third and fifth components of p^TB are practically constants of the motion.

Appendix B: Equations for C_x and C_x^{-1}

Scaling factors were introduced in Eqs. (11) so that all components of $C(x, x_f^0)$ had units of length. $C_i(x, x_f^0) = M_i(x_f^0)Z_i(x, x_f^0)$ for $i = 1, \ldots, 6$ where $M_i(x_f^0)$ is the ith-scaling factor and $Z_i(x, x_f^0)$ is the ith component of the unscaled constraint vector

$$Z(x, x_f^0)^T = [\Delta h^T(h_f^0 \times r_f^0),\ \Delta\alpha,\ \Delta e^T(h_f^0 \times r_f^0),\ \Delta h^Tr_f^0,\ \Delta e^Tr_f^0,\ \Delta r^T(h_f^0 \times r_f^0)]$$

When the constraints in Eqs. (11) are used, the necessary conditions in Eq. (7) are evaluated at $x = x_f^0$. Thus, $\partial C(x, x_f^0)/\partial x|_{x=x_f^0}$ can be used both to determine the effect of variations in final state upon constraints and to determine the inverse matrix $B(x, x_f^0)|_{x=x_f^0}$. It is unnecessary to determine the general expression for $B(x, x_f^0)$ because partial derivatives of transversality conditions with respect to x_f can be calculated from the identity

$$\left[\frac{\partial B(x, x_f^0)^T p}{\partial x}\right]\Bigg|_{x=x_f^0} = - B(x_f^0, x_f^0)^T \frac{\partial}{\partial x}\left[\frac{\partial C(x, x_f^0)}{\partial x}\right]^T\Bigg|_{x=x_f^0} B(x_f^0, x_f^0)^T p$$

This identity can be verified by differentiating

$$[\partial C(x, x_f^0)/\partial x]^T B(x, x_f^0)^T p = Ip = p$$

with respect to x, left multiplying by $B(x, x_f^0)^T$, and evaluating at $x = x_f^0$.

The gradient of constraints in Eq. (11) evaluated at $x = x_f^0$ is

$$\left[\frac{\partial C_i(x, x_f^0)}{\partial x_j}\right]\Bigg|_{x=x_f^0} = M_i(x_f^0)\left[\frac{\partial Z_i(x, x_f^0)}{\partial x_j}\right]\Bigg|_{x=x_f^0}$$

where $[\partial Z_i(x, x_f^0)/\partial x_j]|_{x=x_f^0}$ are components of the matrix

$$\begin{matrix} \gamma h^T, & -r^2 h^T \\ \mu r^T/r^3, & v^T \\ -(\zeta v^T + \gamma E r^T)/\mu, & -(r^2 \gamma v^T + \xi r^T)/\mu \\ -h^T, & 0^T \\ (v^2 r^T - \gamma v^T)/\mu, & 2(r^2 v^T - \gamma r^T)/\mu \\ (h \times r)^T, & 0^T \end{matrix}$$

Here, $\gamma = (r^T v)$, $\zeta = \mu|r| - \gamma^2$, $\xi = h^2 - \gamma^2$, $E = v^2 - \mu/r$, and the subscript f and superscript 0 have been omitted from all state components in the matrix.

The components of the inverse matrix evaluated at $x = x_f^0$ are

$$\{B(x, x_f^0)^T\}_{ij}|_{x=x_f^0} = \frac{1}{M_i(x_f^0)}\left[\left(\frac{\partial Z(x, x_f^0)}{\partial x}\right)^T\right]_{ij}^{-1}\Bigg|_{x=x_f^0}$$

where

$$\left[\left(\frac{\partial Z(x, x_f^0)}{\partial x}\right)^T\right]^{-1}\Bigg|_{x=x_f^0}$$

is the matrix

$$\begin{matrix} 0^T, & -h^T/(r^2 h^2) \\ -r^T/\alpha, & v^T/2\alpha \\ -\mu\gamma r^T/(r^2 h^2 \alpha), & (\mu\gamma v^T - 2\mu\alpha r^T)/(2r^2 h^2 \alpha) \\ -h^T/h^2, & -\gamma h^T/(r^2 h^2) \\ \mu\xi r^T/(2r^2 h^2 \alpha), & -(2\mu\gamma\alpha r^T + \mu\zeta v^T)/(2r^2 h^2 \alpha) \\ v^T/h^2, & -\mu r^T/(r^3 h^2) \end{matrix}$$

where superscripts and subscripts have been omitted from all state components.

References

[1] Cherry, G. W., "A General Explicit Optimizing Guidance Law for Rocket Propelled Spaceflight," AIAA Paper 64-638, Los Angeles, Calif., 1964.

[2] Brown, K. R. and Johnson, G. W., "Real-Time Optimal Guidance," *IEEE Transactions on Automatic Control*, Vol. AC-12, Oct. 1967.

[3] Goodyear, W. H., "Completely General Closed-Form Solution for Coordinates and Partial Derivatives of the Two-Body Problem," *The Astronomical Journal*, Vol. 70, April 1965, pp. 189–192.

[4] Brown, K. R., Harrold, E. F., and Johnson, G. W., "Rapid Optimization of Multiple-Burn Rocket Flights," Rept. CR-1430, Sept. 1969, NASA.

[5] Cohen, A. O. and Brown, K. R., "Simulation Results for a Modified Version of the GUIDE 71/6 Program for Optimal Ascent and Orbital Guidance," Final Report for NAS8-24018, April 1972, IBM Federal Systems Div., Burlington, Mass.