VOL. 2, NO. 3
ARTICLE NO. 77-1071



MAY-JUNE 1979

# An Advanced Spacecraft Autopilot Concept

Edward V. Bergmann*
*Massachusetts Institute of Technology, Cambridge, Mass.*
and
S.R. Croopnick,† J.J. Turkovich,‡ and C.C. Work§
*The Charles Stark Draper Laboratory, Inc., Cambridge, Mass.*

**An autopilot is developed for rotation and translation control of a rigid spacecraft of arbitrary design, using reaction control jets as control effectors. The autopilot incorporates a six-dimensional phase space control law, and a linear programming algorithm for jet selection. The interaction of the control law and jet selection are investigated and a recommended configuration proposed. Simulations are performed to verify the performance of the new autopilot and comparisons are made with an existing spacecraft autopilot. The new autopilot is shown to require 35.4% fewer words of core memory, 20.5% less average CPU time, up to 65% fewer firings, and consume up to 25.7% less propellant for the cases tested. However, the cycle time required to perform the jet selection computations may render the new autopilot unsuitable for existing flight computer applications, without modifications. Finally, the new autopilot is shown to be capable of performing attitude control in the presence of a large number of jet failures.**

## Nomenclature

| | |
|---|---|
| $a_c$ | = control acceleration |
| $a_d$ | = disturbance acceleration |
| $a_i$ | = acceleration available from jet $i$ |
| $b$ | = separation between inner and outer phase spheres |
| $c$ | = convergence rate |
| $c_j$ | = net acceleration of cluster $j$ |
| $d$ | = number of equality constraints in linear jet select |
| $db^+, db^-$ | = upper and lower deadbands |
| $f_i$ | = fuel consumption rate of jet $i$ |
| $F_j$ | = thrust vector of jet $j$ |
| $I$ | = vehicle inertia tensor |
| $m$ | = vehicle mass |
| $r_j$ | = position of jet $j$ with respect to vehicle center of gravity |
| $r_{ps}$ | = phase sphere radius |
| $t_c$ | = autopilot cycle period |
| $t_i$ | = firing time of jet $i$ |
| $t_1$ | = time to drift from initial rate to rate reversal |
| $t_2$ | = time to drift from rate reversal to outside of deadzone |
| $x$ | = vehicle state |
| $\dot{x}$ | = time derivative of vehicle state |
| $\beta$ | = distance parallel to $a_d$, of rate reversal from tangent state |
| $\theta_r, \theta_\beta, \theta_y$ | = roll pitch and yaw attitude |
| $(\ )$ | = measured value of $(\ )$ |
| $(\ )_d$ | = desired value of $(\ )$ |
| $\Delta v_c$ | = rate change request |

Presented as Paper 77-1071 at the AIAA 1977 Guidance and Control Conference, Hollywood, Fla., Aug. 8-10, 1977; submitted Aug. 22, 1977; revision received Aug. 2, 1978. 

Index categories: Spacecraft Dynamics and Control; Spacecraft Systems.

*Graduate student, Department of Aeronautics and Astronautics. Student Member AIAA.

†Section Chief, Control and Flight Dynamics Division. Member AIAA.

‡Staff Engineer. Member AIAA.

§Staff Engineer.

## Introduction

SPACECRAFT control systems have generally required several simplifying assumptions about the spacecraft's behavior to make the design problem tractable. Rigorous control algorithms for only a limited class of spacecraft maneuvers exist, and are of little general use at this time.[1] Accordingly, spacecraft autopilots have typically been designed by assuming the dynamics to be decoupled by axis allowing independent control of each axis, as in Apollo. Such control laws use attitude and attitude rate feedback to generate rotational acceleration requests and, in many cases, open-loop computation of translational acceleration requests. Table lookup or dot product schemes are used to perform jet selection on a pass by pass basis.[2] Such systems, though suitable for vehicles with simple symmetry and jet configurations and unchanging inertia may be inadequate as control systems for more complex spacecraft.

For a rigid vehicle, the vehicle state propagates via Newton's law in translation and Euler's equations in rotation, described about principal axes. For many spacecraft the angular rates will be sufficiently small and/or the off diagonal elements of the inertia tensor sufficiently smaller than the diagonal elements so that the dynamic coupling terms in Euler's equations are many times smaller than the control torques. In such cases, the vehicle is assumed to act as a double integrator in rotation with disturbances due to dynamic coupling. For the Shuttle and Apollo the coupling terms were at least an order of magnitude less than the control torques at the typical angular rates of less than 2 deg/s. It has been the experience of Apollo that this is a justifiable approximation and offers a substantial simplification in autopilot design.

Control is applied by a set of fixed control effectors, such as jets, each of which can be switched on and off, characterized by its acceleration $a_i$, in general having components along each of the six axes so that no pure couples or forces can be expected from the jets.

Jets may fail, or vehicle mass and inertia properties change during a mission so that the autopilot must be capable of controlling the vehicle with differing sets of available jet accelerations. Hence, table lookup jet-select algorithms may become cumbersome or fail after some number of jet failures occur which were not planned for in advance.

Input commands may be in the form of a six-dimensional request and it is necessary for the autopilot to satisfy all

components of this request within a reasonable time. Many current autopilots either require a great deal of time to do so, or cannot simultaneously respond to all six components of a request. The proposed autopilot treats the vehicle control problem as a six-dimensional problem concurrently solving all six axes.

This paper introduces a new $n$-dimensional control system called the phase space control law, and applies it to the spacecraft control problem. A six-degree-of-freedom autopilot is developed, using the six-dimensional phase space control law and a linear programming algorithm for jet selection.[3] Results using Space Shuttle simulations demonstrate the capability of the new autopilot, and performance comparisons with existing spacecraft autopilots are presented.

## The New Autopilot

The autopilot which is under development incorporates the new phase space control law and the linear programming jet select. Its function is to receive input commands from a guidance system or flight crew as well as vehicle state measurements, and to issue firing commands to reaction control jets to satisfy the input commands. An overall block diagram of the vehicle with the new autopilot is shown in Fig. 1.

The new autopilot consists of four basic functions: a supervisor, an error computation routine, the phase space control law, and the linear jet select.

## The Supervisor

Along with other tasks, such as guidance, navigation, and system management, the new autopilot implemented as a program in a spacecraft flight computer is cycled at regular intervals, typically from 10 ms to 1-s duration. Error and control computation is performed in each of these cycles; however, new input commands from guidance or the crew are typically issued at longer, irregular intervals. The interaction between the control law and jet selection in the new autopilot differs from that of current autopilots. Many current control laws generate an acceleration request on a pass by pass basis, and jet-selection algorithms command jets from a table on a pass by pass basis. The control law of the new autopilot generates a rate change request on a pass by pass basis, and the jet select precomputes jet firing times to satisfy the request. A new jet selection is only to be made in response to a genuinely new rate change request as discussed below.

Certain tests are performed by the supervisor to determine if a request is new and need be passed to the jet select. These criteria determine if the control law has generated a genuinely new request, or an unmodeled disturbance has occurred, necessitating recomputation of the jet selection. Two sets of criteria exist, one for when the jets are on: that is, the jet select is implementing a new request, and one for when the jets are off and the vehicle is coasting. They are as follows.

If jets are firing and:

1) the guidance system or crew sets a flag indicating a new input command (i.e., change in commanded rate); or
2) the rate change prediction of the jet select disagrees with state measurement; or
3) the state error has crossed the deadzone limit in either direction; or
4) the jet fail status changes.

(Condition 2 is discussed in detail in the Appendix and condition 3 is covered below.)

Or, if no jets are firing and:

1) guidance system or crew sets a flag indicating a new input command; or
2) the magnitude of the state error is increasing and is not within the allowable deadbands; or
3) the state error has crossed the deadzone limit, going away from the desired state.

Therefore, the new autopilot typically performs jet selection much less frequently than it performs the error and control computations.

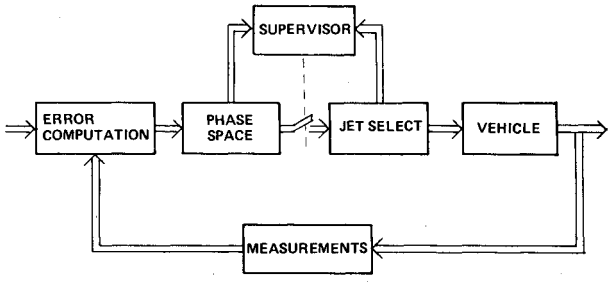


**Fig. 1 Overview of spacecraft/autopilot system.**

## The Phase Space Control Law

A new control law, called the phase space control law, is introduced. The initial concept of a three-dimensional attitude control law is due to S.R. Croopnick[4]; the present control is an outgrowth of his concept. The phase space control law uses a velocity to be gained principle to generate rate change requests, and certain transformations to achieve a spherical deadzone in state space. Control of all six spacecraft degrees of freedom is achieved by this one control law, whereas current systems use multiple iterations of their control laws, one iteration per axis.

Given a system with a six-dimensional state vector $\boldsymbol{x}$ which has an explicitly computable or measurable derivative $\dot{\boldsymbol{x}}$, or rate, it is desired to drive the system to a target state $\boldsymbol{x}_d$ and rate $\dot{\boldsymbol{x}}_d$. The rate change command is

$$\Delta \boldsymbol{v}_c = c \text{ unit}(\boldsymbol{x}_d - \boldsymbol{x}) + (\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}}) \tag{1}$$

The first term, $c$ unit $(\boldsymbol{x}_d - \boldsymbol{x})$, a component of rate which drives the state to the target, is called a convergent velocity and the scalar $c$ is called the convergence rate. The second term $(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}})$ is the correction to the current rate and its negative is called the relative rate.

In practical applications, measurements of state may be inaccurate, and a constant convergence rate may produce overshoots. Recognizing that it is not possible to instantaneously realize $\Delta \boldsymbol{v}_c$, future cycles through the velocity to be gained principle may produce a time varying $\Delta \boldsymbol{v}_c$. If, for example, the current rate is at some angle to the error vector, the error vector will rotate during the jet firings, causing $\Delta \boldsymbol{v}_c$ to rotate. While this rotation could be anticipated, and $\Delta \boldsymbol{v}_c$ modified to account for it, the new autopilot does not do so at present. This is equivalent to assuming that an instantaneous rate change is produced by the jets. In light of these considerations, it is useful to incorporate hysteresis into a control law using this principle.

## The Phase Sphere Concept

Consider a system described by a $j$-dimensional state error vector, $\boldsymbol{x}_e$, given by

$$\boldsymbol{x}_e = \boldsymbol{x} - \boldsymbol{x}_d \tag{2}$$

It is necessary to control each element within its particular deadband, $db_i$, which may have different units as well as different numerical values. To decide whether an element $x_{e_i}$ is within its deadband, it is only necessary to compare two numbers, $|x_{e_i}|$ and $db_i$, if the deadbands are symmetric, i.e., the positive deadband has magnitude equal to that of the negative deadband. If not, the variable and its deadbands can be "biased" and a similar test performed.

By normalizing the vector space of $\boldsymbol{x}$, it is possible to replace this prism by a hypercube. The $i$th dimension of the

prism is twice $db_i$. All of these can be made numerically equal in a space containing $\tilde{x}_e$, with $\tilde{x}_{e_i}$ given by

$$\tilde{x}_{e_i}/r_{ps} = x_{e_i}/db_i \tag{3}$$

where $2r_{ps}$ is the size of a side of the hypercube. A slightly more restrictive, but simpler test for the tip of $x_e$ in this hypercube is to check if the tip lies within the inscribed hypersphere. This will be so if

$$\sqrt{\tilde{x}_e \cdot \tilde{x}_e} < r_{ps} \tag{4}$$

so that the variables are tested for an out-of-deadband condition by comparison of two scalars. The above-mentioned hypersphere is referred to as a "phase sphere" and is a simple means of providing deadbands and hysteresis.

The phase space control law brings together the velocity to be gained principle and the phase sphere concept into a simple control law. The primary interaction of the two is influenced by the determination of the convergence rate, $c$. Should the state lie within a phase sphere described by the deadbands, $c$ is set very small, or "close" to zero. Otherwise and depending upon performance requirement, $c$ may be set equal to a large value. Due to response time requirements, or other considerations, this value may be scheduled in any of several ways. For example, $c$ may be a function of $x_e$ selected to assure response time, or some other criterion.

In some applications, the numerical values of $db_i$ are greatly different. For example, in the spacecraft problem, an attitude deadband may be 0.1 deg, and a position deadband 50 ft. Generating the convergent velocity along $-$unit $(x_e)$ will direct the rate change along the numerically largest variable, rather than that furthest beyond its deadband. By generating this velocity along $-$unit $(\tilde{x}_e)$, the rate change is directed along the variable largest in relation to its deadband. Otherwise, serious overcontrolling of some variables (such as position above) and undercontrolling of others (attitude) may result.

A spacecraft can be driven to a desired state but will drift from that state due to disturbances, modeling errors and control errors. Control is applied to overcome these effects and to drive the state toward the desired state. A satisfactory result is to have the vehicle limit cycle about the desired state. When a system's state is within allowable bounds (i.e., $\tilde{x}_e$ lies within the phase sphere) but its relative rate is nonzero, it is prone to limit cycling. When one of the position state variables approaches its bound, it is necessary to reverse the component of relative rate driving that variable, i.e., to change the sign of its derivative. To do so at the exact point when that variable hits the limit is unlikely since the variable may overshoot the limit during the finite interval of time necessary to reverse the appropriate component of relative rate.

A second phase sphere provides a simple solution to this problem. If the original phase sphere is centered on $x_d$, with radius $r_{ps}$, the second is placed concentric to it, with radius $(r_{ps} - b)$. When the state of the vehicle drifts from the desired state, it intercepts the inner sphere before any of the state variables reaches its limit. The parameter $b$ is then chosen to assure that the appropriate component of relative rate can be reversed before any state variable reaches its deadband. Since the magnitude of the control is generally different along each of the state variables, $b$ actually is a function of the component of relative rate being reversed. As a result, the inner phase sphere would actually be a more complicated surface. By making it a sphere, some computational simplicity is gained, at the cost of more precise control than is required in some state variables.

The convergence rate, $c$, will be set to one of two values depending on the location of $\tilde{x}_e$ relative to the phase spheres. If it is outside the outer phase sphere, the value of $c$ will be such as to ensure sufficiently rapid convergence on the desired state. If the state is within the outer sphere, $c$ will be set to a reasonable limit cycle rate.

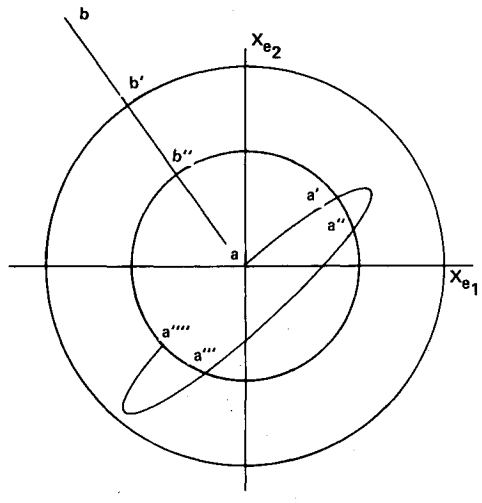


**Fig. 2 Two-dimensional trajectories.**

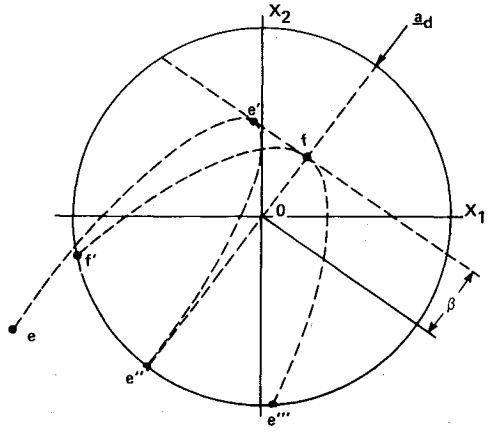


**Fig. 3 Desired disturbance trajectory.**

Figure 2 depicts a hypothetical two-dimensional limit cycle and a convergent trajectory. The limit cycle trajectory starts at $a$ with a small relative rate and coasts until it hits the inner phase sphere at $a'$. Control is then applied to reverse the relative rate, and the state reenters the inner phase sphere at $a''$. Again the state coasts until leaving the inner sphere at $a'''$, where control is again applied to reverse the relative rate, and the state reenters the inner sphere at $a''''$.

The convergent trajectory starts with a large state error at $b$. The state is driven at the chosen convergent rate toward the desired state. Upon intercepting the outer sphere at $b'$, the value of $c$ in the velocity-to-be-gained expression is changed to the limit cycle rate, and control applied to achieve this new rate at point $b''$, where a limit cycle is initiated.

## Disturbance Accelerations

In some instances, a disturbance may exist which will remain constant or nearly so for a large number of autopilot cycles. Such disturbances may be due to venting, gravity gradient torques, drag, or thrust vector offsets. An onboard state estimator will often be capable of identifying such disturbances, so that appropriate corrective action may be taken.

With no change to the current algorithm, a disturbance will cause an offset limit cycle. The state is driven away from the target by the disturbance until reaching the phase sphere. Depending on the magnitude of the disturbance relative to the available control forces or torques, the trajectory may be

temporarily reversed by the control effectors, establishing an offset limit cycle with frequent jet firings, or continuing away from the target, hence exhibiting loss of control.

When sufficient control authority exists, a more desirable limit cycle can be established. Consider the situation depicted in Fig. 3. In this section, all vectors will denote actual physical quantities, rather than normalized quantities. It is required to determine the convergent velocity, $\dot{x}_e$, such that the state follows trajectory $e\,e'\,e''$ in Fig. 3. This trajectory carries the state toward a point $e'$ selected so that the state will then coast to a point $e''$, described by the intersection of the $a_d$ vector through the origin, and the phase sphere.

For zero time average state error in the ensuing limit cycle trajectory, the displacement vector of point $e'$ from the target projected onto $a_d$ is given by[5]

$$\beta = \frac{r_{ps}}{3}\left[1 + \frac{|a_d|}{a_c \cdot \text{unit}(a_d)}\right] \tag{5}$$

In the case of different deadbands for each axis, $r_{ps}$ would become a function of the direction of $a_d$, depending on the relative values of the deadbands in each axis. The component of displacement of point $e'$ perpendicular to $a_d$ is selected to cause the state to arrive at point $e''$, rather than overshoot to point $e'''$. By so targeting the state, a limit cycle on or quite near the trajectory $e''fe''$ is obtained, using less propellant than one along $f'fe'''$ due to the lower component of rate perpendicular to $a_d$. To obtain the desired rate $\dot{x}_e$, one seeks that rate which causes the component parallel to $a_d$ to be zero at $e'$ and to coast back out to the edge of the phase sphere at the same time as the component perpendicular to $a_d$ has carried the state over to a line parallel to $a_d$ through the origin. In Fig. 3, this means we want to pick the rate error at point $e$ that causes the state to coast to $e'$, stop and coast back out to $e''$. In subsequent limit cycles, the state will be made to coast between $e''$ and $f$ by solving the same problem with $e$ at point $e''$.

Starting at point $e''$, which we can write as

$$x_{e''} = r_{ps}\,\text{unit}(a_d) \tag{6}$$

$$= x_{e'} + \dot{x}_{e'} t_2 + \tfrac{1}{2} a_d t_2^2 \tag{7}$$

point $x_{e'}$ is reached from point $x_e$ so that

$$x_{e'} = x_e + \dot{x}_e t_1 + \tfrac{1}{2} a_d t_1^2 \tag{8}$$

and

$$x_{e''} = x_e + \dot{x}_e t_1 + \tfrac{1}{2} a_d t_1^2 + \dot{x}_{e'} t_2 + \tfrac{1}{2} a_d t_2^2 \tag{9}$$

also

$$\dot{x}_{e'} = \dot{x}_e + a_d t_1 \tag{10}$$

so we can solve for $\dot{x}_e$,

$$\dot{x}_e = \frac{1}{t_1 + t_2}\left[r_{ps}\text{unit}(a_d) - x_e - a_d\left[\tfrac{1}{2}(t_1^2 + t_2^2) + t_1 t_2\right]\right] \tag{11}$$

where[3]

$$t_1 = \sqrt{\frac{2[x_e \cdot \text{unit}(a_d) + \beta]}{|a_d|}} \tag{12}$$

and

$$t_2 = \sqrt{\frac{2(r_{ps} + \beta)}{|a_d|}} \tag{13}$$

A disturbance limit cycle with zero time average state error may be obtained by replacing the convergent velocity in the velocity to be gained expression with $\dot{x}_e$.

## Jet Selection

The phase space control law requests rate changes to be produced by the control effectors. The logic involved in translating these requests into a sequence of jet firings, when the effectors are reaction jets, is called jet selection. Current spacecraft autopilots typically use either table lookup or dot product schemes. In his doctoral thesis,[6] Bard S. Crawford investigated linear programming to solve the optimal linear jet-selection problem. A practical algorithm for using linear programming for jet selection was developed by Craig Work.[3,7]

As shown by Crawford, the conditions for linear programming to provide a solution to the jet select problem are exactly the conditions for the vehicle to be controllable with the given jet configuration. These are, for $n$ degree of freedom control, that the vehicle have at least $n+1$ jets and that their acceleration vectors span a Euclidian $n$-space. Because the jet select problem is actually solved in real time, and depends on the rate change request, a cost function and the jet accelerations as parameters only, the new autopilot is capable of adapting, during operation, to changing vehicle configurations, such as jet failures or changes in vehicle inertia.

Hence the new autopilot is an adaptable autopilot which can be applied to any rigid controllable vehicle merely by adjusting certain parameters, if angular rates are kept small enough that the dynamic coupling terms are smaller than the control torques. Moreover, if it is at all possible to achieve the rate change requested by a linear combination of jet firings, the linear jet select will successfully do so, implying that this autopilot has the maximum possible fault tolerance.

Jet selection seeks to achieve a six-dimensional rate change requested by the control law. Posed as a linear programming problem, the jets are modeled by their activity vectors, six of which will form the basis. A jet activity vector, $a_i$, is defined to be the rate change obtained by firing jet $j$ for unit time. For jet $j$, $a_j$ is given by

$$a_j = \begin{bmatrix} F_j/m \\ \hline I^{-1}(r_j \times F_j) \end{bmatrix} \tag{14}$$

in vehicle coordinates. The vector $a_j$ is six-dimensional with three rotational components, and three translational components. Note that, for constant mass properties, $a_j$ is constant in vehicle coordinates. Any cost function which can be expressed as a linear combination of the free parameters may be used. In the new autopilot, fuel consumption is to be minimized, so that the cost function will be

$$\text{cost} = \sum_{i=1}^{\#\text{jets}} t_i f_i \tag{15}$$

Obviously, the jet select must be constrained to satisfy the request.

$$\Delta v_c = \sum_{i=1}^{\#\text{jets}} a_i t_i \tag{16}$$

and the firing times must be nonnegative

$$t_i > 0 \qquad \text{for all } i \tag{17}$$

To be soluble by linear programming: 1) the cost function must be linear and have a minimum value which satisfies the constraints; 2) the equality constraints must admit one or several solutions; and 3) the inequality constraints must hold for one or more of these solutions.

The first assumption insures that it is reasonable and possible to minimize the cost function via linear programming. The second requires that it is possible to provide the rate change with the control effectors. The third states that the sense of the activity vectors is correct (i.e., not all jets fire against the request). The reader is directed to Ref. 8 for an exposition on linear programming. In the following sections some practical modifications to the basic algorithm in the course of applying it to a real spacecraft are developed.

### Selection from a Large Number of Jets

Finding a solution to the linear programming jet-select problem involves testing each of the available jets not in the basis to determine which jet to bring into the basis. A cost reduction for each jet is computed and that jet which maximizes the savings is brought into the basis. For most spacecraft, 16 or fewer jets need be tested. Checking each of the jets would consume a large amount of time and effort each time a new selection is made. For a vehicle with a large number of jets, such as the Shuttle's 44, the computation involved in checking each jet for inclusion in the selection would be prohibitive. A great savings could be realized however, by using the fact that some arrangements of jets are often redundant, or nearly so, so that they can be represented by clusters of jets, where a cluster is defined as a group of jets with numerically similar activity vectors. A representative jet from each cluster is selected as the jet which maximizes the dot product of $a_i$ and $c_j$, where

$$c_j = \sum_{i=1}^{\#\text{ jets in cluster}} a_i \tag{18}$$

is the cluster activity vector. The selection is then performed on the cluster representatives, or alternatively on the $c_j$'s, significantly reducing the number of activity vectors to be checked. Implementation of the request is achieved by firing the representative or the cluster.

### Minimum Impulse Jet Select

When the spacecraft is limit cycling, it is desired to maintain the longest limit cycle period, hence the lowest firing rates, to conserve fuel. The linear programming scheme very often selects two or three jets to fire or fails to meet a minimum on-time requirement when the request is on the order of the jet firing interval granularity. This granularity is due to a minimum firing time, on the order of a few milliseconds, imposed on the jets by startup and tailoff transients. The total impulse of this minimum duration burn is the jet's minimum impulse.

When the linear programming scheme generates jet on-times shorter than this minimum impulse time, on-times are rounded up to the minimum on-time, or down to zero. In the interest of minimum jet cycling, the algorithm is constrained to selecting one jet when requests are on the order of jet granularity. To do so, the starting value of the cost function is taken as $10^6$ times the request, a six-element vector. The first jet will be selected to maximize the saving in cost, by having the largest impact on the maximum component of the request. This demands that the jet activity vector be as closely coaligned with the request as possible, that is, the linear jet select becomes a dot product jet select. The jet so chosen is then fired for the minimum on-time.

### Deletion of Certain Jet Firings

The linear programming algorithm for jet selection is constrained to exactly satisfy the rate change request. Since the control law is based on approximations to the equations governing spacecraft behavior, this request is inexact, particularly when state errors are large. It may be unnecessary to exactly satisfy the request, and a benefit obtained from not doing so.

Often, the firing is dominated by one or two jets with long firing times which closely approximate the request. In such a case, it is practical to delete short jet firings.

### Ignore Option

In certain situations it is desirable to precisely control the spacecraft about some axes, but not others. For example, it may be required to rotate a spacecraft at a precise rate for pointing, but large amounts of translation drift are acceptable. The jet-selection problem, then, is one of fewer than six dimensions. Means of reformulating the problem in these cases to one of fewer than six dimensions are desirable to save unnecessary computational effort.

A method employed by the new autopilot is to assume rotational control is always desired, but translational control is optional. The jet selection problem is then one of satisfying a three-dimensional request at times when rotation only is controlled, and a six-dimensional request when translation is also to be controlled. A substantial saving in computational burden is obtained, as the three-dimensional problem is much more quickly solved than the six-dimensional problem. Fuel savings also follow as three or fewer jets are used for the three-dimensional problem, whereas up to six may be used for the six-dimensional problem, particularly in the presence of jet coupling, where some jets are in the basis purely to cancel coupling.

A more sophisticated approach is to ignore those axes for which no specific request is made (i.e., rate is correct or can be controlled), and recast the problem with only the remaining axes. The jet select problem then can have as few as one dimension or up to six, but computations should be performed only for those axes for which they are necessary.

### Testing

The new autopilot was implemented as a set of computer programs and used to "fly" simulations of the Space Shuttle orbiter. It must be noted that the programs used represented a development version of the new autopilot, and a preliminary version of the Shuttle autopilot. Both are under development so that their final computer burden and core requirements should be somewhat less than presented here.

The simulator used for the Space Shuttle orbiter is the C.S. Draper Laboratory Statement Level Simulator (SLS), described in Ref. 9. Basically, the SLS is a set of digital computer programs which accepts an autopilot program and numerically integrates the orbiter equations of motion using a fourth-order Runge-Kutta integrator. Translational motion is modeled by the usual Newtonian equations of orbital mechanics including the response to the control effectors, while rotation is modeled by Euler's rotational equations for rigid bodies.

Fuel usage and firings were tabulated by the SLS during simulations, which included a fuel penalty of 0.0226 kg per firing to represent incremental fuel loss in jet turn on and turn off. The results are shown in Table 1.

Performing the constant rate maneuver, the new autopilot commanded three firings, for a total on-time of 4.048 s, using 5.5878 kg of propellant. The phase plane autopilot commanded eight firings for a total on-time of 3.967 s using 5.5928 kg of propellant to attain the same rate. The new autopilot performed comparably with the conventional autopilot for this case.

For the 20-deg yaw maneuver with no jets failed, the new autopilot commanded 14 jet firings, for a total on-time of 8.318 s using 11.674 kg of propellant. The phase plane autopilot performing the same maneuver commanded 18

**Table 1 Comparison of fuel consumption**

| Maneuver | Phase plane autopilot, kg | New autopilot, kg | Savings, kg |
|---|---|---|---|
| 2-deg/s constant rate | 5.5928 | 5.5878 | 0.005 |
| 20-deg yaw | 15.703 | 11.674 | 4.029 |
| 20-deg yaw aft jets failed | Unacceptable roll coupling | 21.05258 | . . . |
| 1.5-m translation along $Z$ axis | Failed | 15.722 | . . . |
| 2-deg maneuver to track | 22.37 | 14.5 | 7.87 |

**Table 2 Core requirements for the phase plane autopilot**

| Component | Program words | Data words |
|---|---|---|
| Jet selection | 1836 | 855 |
| Phase plane controller | 436 | 36 |
| Jet firing interface | 20 | 6 |
| Error computation | 34 | 7 |
| Automatic translation control (rate only) | 44 | 15 |
| Total | 2370 | 919 |

**Table 3 Core requirements for the new autopilot**

| Component | Program words | Data words |
|---|---|---|
| Linear jet selection | 831 | 645 |
| Jet firing supervisor | 94 | 40 |
| Rate-change prediction | 66 | 10 |
| Phase space control law | 82 | 6 |
| Error computation | 179 | 26 |
| Control law sequencer | 87 | 59 |
| Total | 1339 | 786 |

**Table 4 Worst-case execution time**

| Component | Time, ms |
|---|---|
| Phase plane controller (three passes) | 3.84 |
| Jet selection | 4.663 |
| Total | 8.503 |

**Table 5 New autopilot requirements**

| Component | Time, ms |
|---|---|
| Phase space control law | 0.7802 |
| Control law supervisor | 1.68 |
| Linear jet select | 31.69 |
| Total | 34.1502 |

firings, for a total on-time of 11.202 s using 15.703 kg of propellant. The new autopilot thus commanded four fewer firings, and saved 4.029 kg of propellant. In the 2-deg maneuver test, the vehicle was rotated about the vector (0.7071, 0, 0.07071) in body coordinates, and commanded to track a 0.45 deg/s pitch rate. The new autopilot required 35 jet firings for a total on time of 10.3 s and consumed 14.516 kg of propellant. The phase plane autopilot commanded 16 firings using 22.37 kg of fuel.

Since the phase plane autopilot failed to perform the 20-deg yaw with all the aft jets failed, no comparison may be made on that test. Similar results were obtained in a case where all the forward jets were failed. In performing the maneuver, the new autopilot commanded nine firings, for a total of 11.383 s, consuming 15.722 kg of propellant. No similar capability was designed into the phase space autopilot.

No detailed testing was performed to determine fuel usage during long-term attitude hold. With the minimum-impulse option, the fuel consumption should generally be equal to or less than that of the conventional autopilot for the same task. The new autopilot uses only one jet firing for a minimum duration to reverse the limit cycle rate, while the conventional autopilot uses two or three, requiring more fuel. Limit cycle periods are thus typically longer for the new autopilot, resulting in a lower jet firing frequency.

## Computer Burden

The computer burden of the new autopilot and the phase plane autopilot are based on HAL/F compilations for sizing and HAL/S compilations for timing for operation in the IBM AP101 computer.

The core requirements for the phase plane autopilot in AP101 wholewords are shown in Table 2 for a total of 3289 words of core memory. The core requirements of the new autopilot in AP101 wholewords are shown in Table 3 for a total of 2125 words of core storage. The new autopilot saves 1164 wholewords of core burden over the phase plane autopilot.

The worst-case execution time required for specific parts of the phase plane autopilot is shown in Table 4. Both must be cycled at 25 Hz, giving a 21.3% CPU burden.

The new autopilot requirements are shown in Table 5. Here, it is necessary to cycle the phase space control law and the control law supervisor at 25 Hz. However, the linear jet select is only performed under certain conditions determined by the supervisor. An estimate of its CPU burden can be determined by noting that the linear jet selection was performed six times over the 24 s of the attitude maneuver test. This gives a CPU burden of 0.79%. Combining this with a 6.15% CPU burden for the phase space control law and supervisor gives a 6.94% total CPU burden for the new autopilot.

## Jet Failures

To compare the response of the new autopilot and the phase space autopilot to jet failures, the 20-deg yaw was repeated with the aft jets failed. The phase plane autopilot yawed 20 deg, but accumulated an unacceptable −9.6-deg roll error. The new autopilot was able to perform the 20-deg yaw and maintain roll and pitch error within the allowable deadband.

**Table 6 Comparison of final attitude errors (roll, pitch, yaw), deg**

| Maneuver | Phase plane | New autopilot |
|---|---|---|
| 20-deg yaw | (0.5313, 0.2719, 0.810) | (−0.766, −0.256, 0.914) |
| 20-deg yaw aft jets failed | (−9.607, −0.651, 0.03) | (0.076, −0.196, 0.512) |
| 2-deg maneuver to track | (−0.0899, −0.0158, −0.4295) | (−0.3481, −0.1216, 0.0218) |

## Accuracy

Neither autopilot brought the vehicle state within the deadzone at the end of the simulation of the 20-deg yaw, though both were converging at similar rates. Each was within 0.2 deg of the deadband, and coasting toward the commanded attitude at the termination of the simulation. The final errors in degrees are shown in Table 6. When the maneuver was repeated with aft jets failed the new autopilot drove the attitude to within the deadbands, but the phase plane autopilot accumulated a −9.6-deg roll error. The errors are shown in Table 6. For the 2 deg/s roll rate command, the new autopilot drove the rate error below 0.01 deg/s in all three axes, as did the phase plane autopilot. In the 2-deg maneuver to track, both autopilots maintained attitudes to within 0.5 deg in each axis. Finally, in performing the position maneuver, the new autopilot drove the position error within 0.06 m in all translation axes, maintaining attitude within the 2.5-deg deadband in all rotation axes.

The two autopilots show roughly equal accuracy in general, but the phase plane autopilot fails when all the aft jets are failed, whereas the new autopilot retains its capability.

## Conclusions

A new spacecraft autopilot, capable of controlling vehicles of arbitrary design with changing mass properties and jet failure status, has been developed. The phase space control law concept forms the basis of the control law, and a linear programming algorithm is used for jet selection. This autopilot has undergone preliminary testing with a Space Shuttle orbiter simulation, controlling both translation and rotation with the reaction jets as control effectors.

Preliminary tests comparing the new autopilot with an early development version of the Space Shuttle autopilot indicate that some software benefits and fuel savings may be realized with comparable performance for the cases studied herein. The new autopilot requires 35.4% fewer words of core storage, and places an average 20.5% lower CPU burden on the flight computer; however, since a single pass through jet selection takes approximately 31 ms, the new autopilot lacks in response time. For some maneuvers tested, a 25.6% savings in RCS propellant is realized.

The new autopilot incorporates greater flexibility than current autopilots, incorporating adaptive logic to adjust to changing vehicle mass properties and jet configurations. Thus, the new autopilot should be capable of controlling a spacecraft over the range of its inertia properties (due to payload deployment, fuel usage, etc.) and in the presence of jet failures. Specifically, failure of 28 of the 44 jets on the Shuttle does not prevent the new autopilot from performing certain maneuvers, but disables a conventional autopilot.

Further development of certain aspects of the new autopilot are under study. Among these are the implications and limitations of selecting the values of several parameters such as limit cycle rate, buffer region size, and convergence rate. Finally, the approach used in the new autopilot is based on a linear approximation to vehicle dynamics. Extensions to nonlinear cases are under consideration.

## Appendix: Condition 2 for Performing a New Jet Selection When Jets are Firing

Consider a single-component rate-change request $\Delta v_c$ and the actual rate $\dot{x}$. The desired rate is $\dot{x}+\Delta v_c$. At time $t_0$, the jet select turns on a set of jets to implement the rate-change request, with firings lasting through time $t_f$. Assume that during the interval $[t_0, t_f]$ the state has not crossed either of the phase spheres, so that the value of the convergence rate $c$ in the velocity-to-be-gained principle

$$\Delta v_c = c(x_d - x) + (\dot{x}_d - \dot{x}) \tag{A1}$$

is constant, so that $\Delta v_c$ changes as a function of relative rate only. Clearly, the relative rate is decreasing in magnitude with time as the request is implemented This observation applies similarly to the magnitude of $n$-dimensional vectors, so that in controlling any system, the request changes during its implementation.

Since the jet activity vectors in general do not coalign with the request, and vehicle dynamics are nonlinear, the state trajectory during a set of jet firings will not be a straight line. That is, the net acceleration of the jets firing changes during implementation of a request as the jets are shut off one by one and is generally not parallel to $\Delta v_c$. The combination of jets selected to fire often includes one or two jets nearly aligned with the request, and one to four others fired to bring the net rate change in line with the request. Since all the jets are turned on at once, the rate change achieved when the first few jets are shut off will not be aligned with the request, but the additional rate change of the remaining jets added to this will cause the net rate change again to coalign with and satisfy the request.

Actual rate changes due to jet firings must be monitored and compared to some anticipated rate change. Unfortunately, since the rate vector rotates with time during a firing, checking that the acceleration coaligns with the request will not work. Rather, it must be assumed that the solution of the linear jet select problem will produce a rate change ultimately satisfying the request. Prediction of the rate change on a per cycle basis, as

$$\Delta \dot{x} = t_c \sum_i a_i \tag{A2}$$

for all the jets $i$ that are firing where $t_c$ is the autopilot cycle period, is performed and compared with actual measurements of the vehicle rate change. The comparison is based on whether

$$t_c \sum_i a_i \cdot (\dot{x}_{\text{now}} - \dot{x}_{\text{last}}) < G \left| t_c \sum_i a_i \right|^2 \tag{A3}$$

where $\dot{x}_{\text{now}}$ is the current estimate of vehicle rate, $\dot{x}_{\text{last}}$ is the estimate of vehicle rate from the last cycle, and $G$ is a gain. If this condition is true, the actual rate change is either not closely coaligned with the prediction or significantly smaller than the prediction. Both cases indicate that a new selection and possibly a recomputation of jet activity vectors is required. Hence, the jet select is called.

If the condition is false, the actual rate change is coaligned with the prediction or greater in magnitude than the prediction. The first is desirable, indicating the selection is being correctly implemented. The second is undesirable, resulting in either overcontrol or divergence of the errors. Either case will be detected by the control law, and corrective action taken.

## Acknowledgments

This paper was prepared under Contract NAS 9-13809 with the National Aeronautics and Space Administration. Publication of this paper does not constitute approval by NASA of the findings or conclusions contained herein.

## References

[1] Dixon, M.V., "Fuel-Time Optimal Spacecraft Reorientation," C.S. Draper Laboratory, Cambridge, Mass., T-502, May 1968.

[2] Croopnick, S.R., et al., "Space Shuttle Orbiter Orbital Flight Test LEVEL C Functional Subsystem Software Requirements Guidance Navigation and Control," C.S. Draper Laboratory, Cambridge, Mass., R-942, July 1976.

[3]Bergmann, E.V., "A New Spacecraft Autopilot," C.S. Draper Laboratory, Cambridge, Mass., T628, May 1976.

[4]Croopnick, S.R., "Three Dimensional Attitude Control Law," Seminar given at the C.S. Draper Laboratory, Cambridge, Mass., Nov. 1974.

[5]Croopnick, S.R., Keene, D.W., and Turkovich, J.J., "On Orbit Phase," Section 6 of "Integrated Digital Flight Control System for the Space Shuttle Orbiter," C.S. Draper Laboratory, Cambridge, Mass., E-2736, Jan. 1975.

[6]Crawford, B.S., "Operational Design of Multi-Jet Spcecraft Control Systems," C.S. Draper Laboratory, Cambridge, Mass., T-509, Sept. 1968.

[7]Jones, J.E., Weissman, P.S., Work, C.C., and Kubiak, E.T., "DFCS Input and Output," Section 2 of "Integrated Digital Flight Control System for the Space Shuttle Orbiter," C.S. Draper Laboratory, Cambridge, Mass., E-2736, Jan. 1974.

[8]Chung, A., *Linear Programming*, Charles E. Merrill Books, Inc., Columbus, Ohio, 1966.

[9]"User's Guide to the C.S. Draper Laboratory Statement Level Simulator," C.S. Draper Laboratory, Cambridge, Mass., R-799 Aug. 1976.