JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 24, No. 3, May–June 2001

Computationally Efficient Control Allocation

Wayne C. Durham*
Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061

The details of a computationally efficient method for calculating near-optimal solutions to the three-objective, linear, control-allocation problem are described. The control-allocation problem is that of distributing the effort of redundant control effectors to achieve some desired set of objectives. The optimal solution is that which exploits the collective maximum capability of the effectors within their individual physical limits. Computational efficiency is measured by the number of floating-point operations required for solution. The method presented returned optimal solutions in more than 90% of the cases examined; nonoptimal solutions returned by the method were typically much less than 1% different from optimal. The computational requirements of the method presented varied linearly with increasing numbers of controls and were much lower than those of previously described facet-searching methods, which increase in proportion to the square of the number of controls.

I. Introduction

TRADITIONAL aircraft design included three aerodynamic controls for each of the rotational degrees of freedom: ailerons for roll, elevator for pitch, and rudder for yaw. Modern tactical aircraft have more than the classical 3 moment generators in numbers nearing 20.

The redundancy of these control effectors admits an infinite number of combinations that satisfy a particular objective, as long as the physical limits of the effectors are not considered. Consideration of these physical limits leads to unique solutions at the maximum collective capabilities of the effectors. The distribution of these several controls to achieve specific objectives is the general control-allocation problem. The determination of the unique combinations of control effectors that yield maximum collective capabilities is the optimal control-allocation problem.

The geometry of the control-allocation problem was developed in Refs. 1–3. The geometry of the attainable moments in the three-objective problem is, in general, the projection of an m-dimensional rectangular box (where m is the number of control effectors) into three dimensions.

Calculation of the geometry of the subset of attainable moments (more generally, objectives) is simple[2] but requires $m(m-1)$ sets of calculations, which, therefore, increase in proportion to the square of the number of controls. The calculation of the complete geometry of the polytope is by itself of sufficient computational complexity as to render it impractical for real-time implementation in current flight-control computers. It is not desirable to precalculate the geometry for at least two reasons:

1) The attainable moments continually change with the state of the aircraft, and precalculation would require a large amount of storage to represent adequately the flight envelope.

2) Control redundancy makes feasible real-time reconfiguration following control failure identification.[4] The number of permutations of control failures required to account for all eventualities is quite large, and each permutation creates a new polytope.

Therefore, there is no current interest in methods of solving the optimal allocation problem that require calculation of the complete geometry of the attainable moments.

The method of solving the optimal control-allocation problem as previously practiced by researchers at Virginia Polytechnic Institute and State University does not generally require calculation of the complete geometry of the polytope. Instead, facets are generated in pairs and tested until the facet containing the intersection is found.[5,6] Extensive efforts have been made to find ways to ensure that the solution was found quickly, that is, with few facets being generated. Although these methods can reduce the average search time, none of these methods are able to obviate the worst-case possibility, in which case all of the facets have to be generated.

Only two other control-allocation methods have demonstrated the capability to generate optimal, or near-optimal solutions: the method of cascaded generalized inverses[5] and linear programming methods.[5,7,8] Cascaded generalized inverse algorithms, as originally described by Bordignon,[5] have computational requirements that vary linearly with the number of controls, but frequently return solutions to optimal problems that have extremely large errors. This form of the cascaded generalized inverse algorithm was employed for comparison in Ref. 9. Linear programming methods typically do return admissible solutions to optimal problems, but have computational requirements as bad as or worse than the facet-searching method.[5]

The primary focus of this paper is to describe an alternative method to solving the linear, three-objective optimal control-allocation problem. There are many control-allocation issues motivated by the flight-control problem that are not addressed in this paper, for example, rate limiting, antiintegral windup, chatter, impact of moment error, how to address unattainable objectives and nonzero error, and the effect of singular partitions of the control effectiveness matrix. Each of these issues is worthy of study, but none are peculiar to the algorithm presented in this paper.

II. Problem Statement

We consider the linear map B_3 from $\Re^m \rightarrow \Re^3$, $\boldsymbol{y}_3 = B_3\boldsymbol{u}$, where $\boldsymbol{u} \in \Re^m$ are the controls with effectiveness B_3 in generating the objectives $\boldsymbol{y}_3 \in \Re^3$. The subset of admissible controls is Ω, $\boldsymbol{u} \in \Omega \Leftrightarrow u_{\min,i} \leq u_i \leq u_{\max,i}$, where $i = 1, \ldots, m$. The subset of attainable objectives Φ_3 is the image of Ω, $\boldsymbol{y}_3 = B_3\boldsymbol{u} \in \Phi_3 \Leftrightarrow \boldsymbol{u} \in \Omega$. The three-objective optimal control-allocation problem is, given B_3, Ω, and a half-line $\mathcal{L}_3$ in the direction of some arbitrary desired objective $\boldsymbol{y}_{3_d}$, find the intersection p_3 of $\mathcal{L}_3$ with $\partial(\Phi_3)$, the convex hull of Φ_3.

III. Method

The method described in this paper consists of identifying vertices and edges of $\partial(\Phi_3)$. Certain geometrical properties of these vertices and edges are used to identify those that are in the vicinity of the solution p_3. Those vertices and edges are then systematically collected to form a set of vertices and edges that are, hopefully, part of the facet that contains p_3. If the solution facet is thus determined, then the problem is solved. Otherwise, the set of vertices and edges are combined in a manner that permits calculation of an approximate solution.

We assume, without loss of generality, that the desired objective $\boldsymbol{y}_{3_d}$ is in the direction of the y_1 axis, or $\hat{\boldsymbol{y}}_{3_d}^T = \hat{\boldsymbol{e}}_{3,1} = \{1, 0, 0\}$. For any arbitrary $\boldsymbol{y}_{3_d} \neq \boldsymbol{0}$ there is a (nonunique) nonsingular transformation of the problem that satisfies this condition. Then $\mathcal{L}_3 = \{\alpha_3, 0, 0\}$ where $\alpha_3 > 0$.

Received 4 January 1999; revision received 28 May 1999; accepted for publication 7 August 2000.

*Associate Professor, Department of Aerospace and Ocean Engineering.

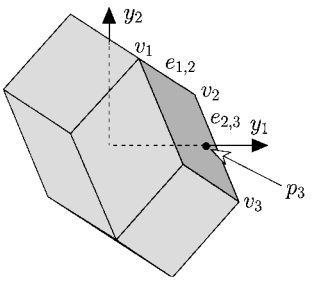

Fig. 1a Φ_3.

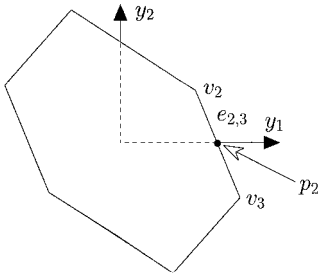

Fig. 1b Φ_2.

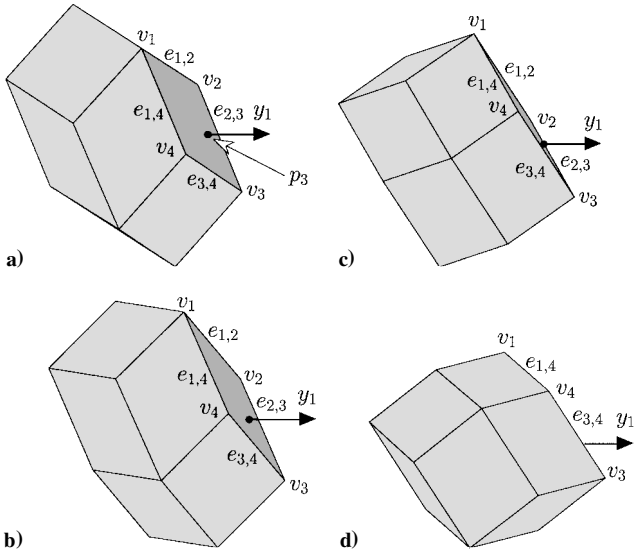

Fig. 2 Rotation of Φ_3.

The first part of the method, identifying vertices and edges of $\partial(\Phi_3)$, is done by examining a two-dimensional projection of Φ_3, that projection denoted Φ_2. The motivation for examining Φ_2 is twofold: 1) The convex hull $\partial(\Phi_2)$ is the projection of $\partial(\Phi_3)$, so that vertices and edges of $\partial(\Phi_2)$ are in a one-to-one correspondence with some of the vertices and edges of $\partial(\Phi_3)$. 2) The two-dimensional problem is easy to solve. Figure 1 depicts this relationship. Figure 1a shows some Φ_3, with y_3 oriented directly out of the page, and the optimal solution p_3. Figure 1b shows the corresponding Φ_2 and its optimal solution p_2. Clearly, the edge that contains p_2 is one of the four edges that define the facet that contains p_3. In the particular case shown, solution of the two-dimensional problem yields information about the solution to the three-dimensional problem.

The identification of a single edge of the solution facet of Φ_3 is insufficient to uniquely determine that facet. Figure 2 shows how other edges may be determined. Figure 2a is the same as Fig. 1a, with several vertices and edges identified. Axis y_2 is understood to be vertically upward, and y_3 is directly out of the page, in each figure. As before, $\mathcal{L}_3$ is coincident with y_1. The solution point p_3 is identified in Fig. 2a, along with the four vertices v_i and edges $e_{i,j}$ of the solution facet.

In Fig. 2a, solving the two-dimensional problem will identify edge $e_{2,3}$. To identify other edges, Φ_3 is rotated about y_1, and y_2–y_3 redefined such that y_3 is always normal to the projection. In Fig. 2 the rotation about y_1 is negative as conventionally defined, so that the top of Fig. 2a is receding. In Figs. 2b and 2c, the two-dimensional solution remains $e_{2,3}$, but the solution facet becomes more nearly edge-on to the viewpoint along y_3. From Figs. 2c to 2d, the solution facet proceeds to the back of the figure, and edge $e_{3,4}$ becomes the edge containing the solution to the two-dimensional problem. The two edges $e_{2,3}$ and $e_{3,4}$ uniquely determine the solution facet, and the three-dimensional problem is solved.

Various geometrical features of the vertices and edges of Φ_3 and Φ_2 will be used to help figure out which edges are part of the solution. In Fig. 2a, viewed as Φ_2, when traversing the edges around Fig. 2a v_2 is above $\mathcal{L}_2$ (positive y_2 component), and v_3 is below. The above–below feature may be used to identify the edge without actually solving for p_2. In Figs. 2a–2c, viewed as Φ_3, the identified edge $e_{2,3}$ is behind $\mathcal{L}_3$ (has a negative y_3 component when its y_2 component is zero), and in Fig. 2d $e_{3,4}$ is in front of $\mathcal{L}_3$. These relationships are simplified by the transformation of the problem to align $\boldsymbol{y}_{3_d}$ with the y_1 axis, and this is the primary motivation for doing so. The qualities of in-frontness and behindness will play a role in determining which edges are candidate members of the solution facet.

It should be clear from Fig. 2 that, when the solution facet is exactly edge-on to y_1–y_2, infinitesimal rotations in either direction about y_1 will alternatively present $e_{2,3}$ and $e_{3,4}$ as the solution edge to the two-dimensional problem. Thus, two edges, one in front of and one behind $\mathcal{L}_3$, are more likely part of the solution facet if the angle of rotation about y_1 that separates them is small.

It is natural at this point to ask if there is always a rotation that places the solution facet edge-on. The answer is yes. Denote the normal vector to the solution facet in $\Re^3$ as $\boldsymbol{n}$. The facet will be edge-on when $\boldsymbol{n}$ lies in the plane y_1–y_2, or when its y_3 component is zero. It is easy to verify that for a transformation $T(\theta)$

$$T(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix} \tag{1}$$

there is always some θ_n such that the y_3 component of $T(\theta_n)\boldsymbol{n}$ is zero.

In summary, the method to be used to solve the three-dimensional optimal allocation problem is, therefore, as follows:

1) Transform the problem to align $\boldsymbol{y}_{3_d}$ with y_1.

2) Find the edge of Φ_2 that crosses $\mathcal{L}_2$.

3) Perform systematic rotations $T(\theta)$ about y_1, repeating step 2 until step 4 is satisfied.

4) When two edges are found, one in front of and the other behind $\mathcal{L}_3$, separated by a suitably small rotation angle, form the union of the two edges (see Sec. IV for union operations).

a) If the union is a facet, and p_3 is on that facet, solve the problem.

b) If the union is a facet, and p_3 is not on that facet, either approximate the solution or return to step 3.

c) If the two edges do not form a facet, either approximate the solution or return to step 3.

IV. Algebra for Control Allocation

It is well known[10] that the subset of attainable moments Φ_n, a convex polytope, is bounded by faces that are images of $(n-1)$-dimensional objects in Ω_n. Each face may be thought of as the structure that results from varying $(n-1)$ controls over their ranges of admissible values while all other controls are fixed at some combination of minimum and maximum values. Each of the $(n-1)$-dimensional faces are themselves convex polytopes, bounded by polytopes of dimension one less than the face. This geometry continues down to the minimal nonempty face, the zero-dimensional vertices. For the three-dimensional problem being considered, $n = 3$ and Φ_3 is bounded by two-dimensional facets.

A. Object Notation

Vertices, edges, facets, and higher-dimensional objects in $\Re^m$ are conveniently denoted in object notation by an m vector $\boldsymbol{o}^o$ (with the superscript o signifying that object notation is employed), defined as follows: If on the object a control is at its maximum, the integer 1

is assigned; if it is at its minimum, -1 is assigned; and if it is free to vary on the object, 0 is assigned. Thus, a vertex would be represented by a vector consisting only of -1 and $+1$, an edge would have one 0, a facet two 0s, and so on. The dimension of an object $\boldsymbol{o}^o$, dim $(\boldsymbol{o}^o)$, is the number of 0s in its representation in object notation.

B. Union

Given any two objects thus represented, it is easy to find their union, which is the smallest object that contains the two. The two objects are compared on an element-by-element basis. If the corresponding elements of the first and second objects are different, a zero is assigned to the result, otherwise their common value is assigned. Given $\boldsymbol{o}_1^o$ and $\boldsymbol{o}_2^o$, for $i = 1, \ldots, m$,

$\boldsymbol{o}_1^o(i) \rightarrow$ $\boldsymbol{o}_2^o(i) \downarrow$	-1	0	1
-1	-1	0	0
0	0	0	0
1	0	0	1

For an example of the union operation, consider five controls and the three objects $\mathbf{a}^o$, $\mathbf{b}^o$, and $\mathbf{c}^o$, with

$$\mathbf{a}^o = \{1 \quad 0 \quad -1 \quad 1 \quad 1\}^T$$

$$\mathbf{b}^o = \{-1 \quad 1 \quad -1 \quad 1 \quad -1\}^T$$

$$\mathbf{c}^o = \{1 \quad 1 \quad -1 \quad 1 \quad 0\}^T$$

Objects $\boldsymbol{a}^o$ and $\boldsymbol{c}^o$ are edges (one 0), and object $\boldsymbol{b}^o$ is a vertex (no zeros). Moreover, we have

$$\mathbf{a}^o \cup \mathbf{b}^o = \{0 \quad 0 \quad -1 \quad 1 \quad 0\}^T$$

$$\boldsymbol{a}^o \cup \boldsymbol{c}^o = \{1 \quad 0 \quad -1 \quad 1 \quad 0\}^T$$

$$\boldsymbol{b}^o \cup \boldsymbol{c}^o = \{0 \quad 1 \quad -1 \quad 1 \quad 0\}^T$$

From this process we may easily establish that the smallest object that contains both $\boldsymbol{a}^o$ and $\boldsymbol{b}^o$ is three dimensional, that of $\boldsymbol{a}^o$ and $\boldsymbol{c}^o$ is two dimensional, and that of $\boldsymbol{b}^o$ and $\boldsymbol{c}^o$ is also two dimensional.

C. Application to Φ_3

Although this object notation is defined in terms of the m-dimensional set of admissible control deflections Ω, it may be used to describe objects in the lower-dimensional sets of attainable objectives. These lower-dimensional objects are simply the images, through the linear transformations B_3 (for Φ_3) or B_2 (for Φ_2), of all points on the object in m space. Thus, when we speak of an edge in Φ_3 using this object notation, we mean it is the image of the corresponding object in Ω.

If we desire the coordinates of an object in Φ_n ($n = 2$ or 3), then we convert the object notation to a control vector: From the object notation, actual minimum/maximum limits are assigned to the nonzero elements. As for the zero elements, the vertices of the object in Φ_n are determined by setting all combinations of minima and maxima of the corresponding controls, then multiplying the resulting control vectors by B_3. Alternatively, some specific values of the controls associated with the zero elements, within their admissible ranges, may be assigned. Multiplying this control vector by B_n yields a general point on the object in Φ_n.

With respect to notation, we use the superscript u to denote an object in $\Re^m$ with actual control deflections in place of the integers. This is notional, of course, for those controls that are given by zero, and the interpretation is that control ranges from its minimum to its maximum deflection. Thus, $\boldsymbol{o}^u$ is $\boldsymbol{o}^o$ with actual deflections. The image of $\boldsymbol{o}^u$ in $\Re^n$ is loosely $B_n \cdot \boldsymbol{o}^u$, which is understood to map whole ranges of control deflections if required. We denote these images with the superscript y.

Note that all objects defined by object notation are on the convex hull of Ω with one exception. The exception is the object consisting of all zeros, which represents the entire set of admissible controls. A single nonzero entry denotes one of the $(m-1)$-dimensional hyperfacets of Ω. Smaller-dimensional objects are analogous to vertices or edges on a three-dimensional cube: They do not constitute the entire boundary (convex hull) of the cube, but they are all on that boundary.

Not all of the objects defined by object notation in Φ_n are on $\partial(\Phi_n)$. For a three-dimensional Φ_3 only $(n-1)$, or two-dimensional and smaller objects, those with two or fewer zeros in object notation, can exist on the boundary $\partial(\Phi_3)$. Demonstrably, not all objects in Ω that are of acceptable dimension will map to $\partial(\Phi_3)$; the point of Ref. 2 was to determine which two-dimensional facets did and did not map to $\partial(\Phi_3)$.

We will use this algebra to determine whether two edges have as their union a facet that contains the intersection with $\mathcal{L}_3$.

V. Details of Method

A. Initial Transformation

The first step is to transform the problem to align $\boldsymbol{y}_{3_d}$ with y_1. There are many ways to derive such a transformation; the method shown here is reasonably efficient and easily generalizes to higher-dimensional problems.

Let

$$\boldsymbol{y}_{3_d} = \begin{Bmatrix} y_{1_d} \\ y_{2_d} \\ y_{3_d} \end{Bmatrix} \tag{2}$$

Assume that $\boldsymbol{y}_{3_d}$ has been normalized, $\|\boldsymbol{y}_{3_d}\|_2 = 1$. Consider a transformation G such that

$$G\boldsymbol{y}_{3_d} = \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix} \tag{3}$$

We assume that none of y_{i_d} where $i = 1, \ldots, 3$ are zero and relegate those problems to the special cases category. (Two zero elements just require possibly a shuffling of rows, and one zero element requires possibly a shuffling with subsequent solution of a trivial 2×2 problem.) With that assumption, we let

$$G = \begin{bmatrix} y_{1_d} & y_{2_d} & y_{3_d} \\ 0 & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix}$$

Take all square roots to be positive. Constructively, the elements g_{ij} may be determined according to

$$g_{31} = \sqrt{1 - y_{1_d}^2}, \qquad g_{32} = -y_{1_d} y_{2_d} \big/ g_{31}$$

$$g_{33} = -y_{1_d} y_{3_d} \big/ g_{31}, \qquad g_{22} = -\sqrt{1 - y_{2_d}^2 - g_{32}^2}$$

$$g_{23} = -\left(y_{2_d} y_{3_d} + g_{32} g_{33}\right) \big/ g_{22} \tag{4}$$

It is easy to verify that this algorithm yields a unitary G and $G\boldsymbol{y}_{3_d} = \{1, 0, 0\}^T$. To ensure $|G| = +1$, the second (or third) row is multiplied by $\operatorname{sgn}(y_{3_d})$, the sign of y_{3_d}.

In subsequent discussions, any reference to a B_3 matrix is taken to mean the original matrix, for example, $B_{3_{\text{orig}}}$, left-multiplied by G:

$$B_3 = GB_{3_{\text{orig}}} \tag{5}$$

Then, if $\boldsymbol{u}$ solves $B_3\boldsymbol{u} = \{1, 0, 0\}^T$, it also solves $B_{3_{\text{orig}}}\boldsymbol{u} = \boldsymbol{y}_{3_d}$, that is,

$$B_3\boldsymbol{u} = \{1, 0, 0\}^T \qquad \Rightarrow G^{-1}B_3\boldsymbol{u} = G^{-1}\{1, 0, 0\}^T$$

$$\Rightarrow B_{3_{\text{orig}}}\boldsymbol{u} = \boldsymbol{y}_{3_d} \tag{6}$$

Note that the transformation G has only to be nonsingular for Eq. (6) to hold, that is, rotation matrices are not required. Thus, the solution $\boldsymbol{u}$ to $B_{3_{\text{orig}}}\boldsymbol{u} = \boldsymbol{y}_{3_d}$ is also the solution to $GB_{3_{\text{orig}}}\boldsymbol{u} = G\boldsymbol{y}_{3_d}$ for any G where $|G| \neq 0$.

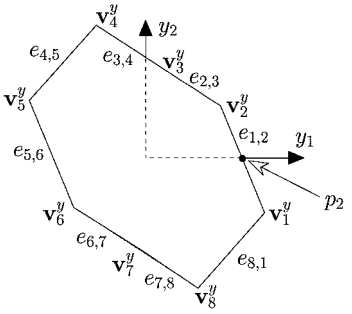

Fig. 3 Geometry of Φ_2.

B. Solution of the Two-Dimensional Problem

Any solution to the two-dimensional problem may be used. The method described in this section is robust and reasonably efficient. The method does require that a list of m elements be sorted, for which algorithms are readily available.

1. *Two-Dimensional Geometry*

The two-dimensional projection we are considering is that which results from retaining just the first two rows of B_3 and the first two components of $\boldsymbol{y}_{3_d}$. Denote the two-dimensional problem as B_2 and $\boldsymbol{y}_{2_d}$,

$$B_3 = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \boldsymbol{r}_3 \end{bmatrix}, \qquad \boldsymbol{y}_{3_d} = \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix}, \qquad B_2 = \begin{bmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \end{bmatrix}, \qquad \boldsymbol{y}_{2_d} = \begin{Bmatrix} 1 \\ 0 \end{Bmatrix} \tag{7}$$

The row vectors $\boldsymbol{r}_i$ are rows of B_n. Thus $B_2, \boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}$, and $\boldsymbol{y}_{2_d}$ collectively define a two-dimensional, m-control-allocation problem. The two-dimensional subset of attainable moments is denoted Φ_2 and is the projection of Φ_3 into the plane of y_1–y_2. Thus, $\mathcal{L}_2 = (\alpha_2, 0)$ where $\alpha_2 > 0$, and the optimal solution is at the point $p_2 = (\alpha_{2_{\max}}, 0)$, where $p_2 \in \partial(\Phi_2)$. We assume the existence of the point p_2.

Denote the vertices of Φ_2 as $\boldsymbol{v}_i^y = \{v_{i,1}^y, v_{i,2}^y\}^T \in \Re^2$, where $i = 1, \ldots, n_v$. In general, there exists one vertex, denoted by $\boldsymbol{v}_1^y$, whose y_1 component $v_{1,1}^y$ is greater than or equal to that of all other vertices:

$$\boldsymbol{v}_1^y = \left\{ \left\{ v_{1,1}^y, v_{1,2}^y \right\}^T \in \Phi_2 \;\middle|\; v_{1,1}^y \geq v_{i,1}^y, i = 2, \ldots, n_v \right\} \tag{8}$$

In Fig. 1b, $\boldsymbol{v}_1^y$ is the vertex denoted v_3. (In subsequent rotations of Φ_3 about y_1, the same vertex is always $\boldsymbol{v}_1^y$ because the values of y_1 components will not change.) The vertex is properly labeled in Fig. 3. In the event that there is more than one vertex that shares the maximum y_1 component, any one may be chosen for purposes of this discussion.

Denote by $\mathcal{V} = \{\boldsymbol{v}_1^y, \ldots, \boldsymbol{v}_{n_v}^y\}$ the ordered set of all vertices of $\partial(\Phi_2)$. The vertices are placed in clockwise order about Φ_2 if the y_2 component of $\boldsymbol{v}_1^y$ is positive or zero, $v_{1,2}^y \geq 0$, and ordered counterclockwise otherwise. Denote by $\mathcal{E} = \{\boldsymbol{e}_{i,i+1}^y, \boldsymbol{e}_{n_v,1}^y\}$, where $i = 1, \ldots, n_v - 1$, the set of edges of Φ_2 generated by connecting two consecutive vertices in $\mathcal{V}$. Thus, $\mathcal{E}$ is the convex hull of Φ_2. Figure 3 depicts these definitions for the figure introduced in Fig. 1b.

2. *Determination of the Edge*

One of the edges $\boldsymbol{e}_{i_p,i_p+1}^y \equiv \boldsymbol{e}_p^y$ contains the point p_2. This edge is characterized by its crossing the y_1 axis, or

$$\operatorname{sgn}\left(v_{i_p,2}^y\right) \neq \operatorname{sgn}\left(v_{i_p+1,2}^y\right)$$

There will be two such edges, but by ordering the vertices in the clockwise or counterclockwise manner described, the solution edge is always the first such one encountered in traversing the edges in $\mathcal{E}$ starting at $\boldsymbol{e}_{1,2}^y$. Thus, if we have $\boldsymbol{v}_1^y$, and the sets $\mathcal{V}$ and $\mathcal{E}$, the two-dimensional optimal control allocation problem is easily solved.

We begin by finding $\boldsymbol{v}_1^u$ and $\boldsymbol{v}_1^y$ as follows:

1) Determine the controls $\boldsymbol{v}_1^u$ that generate $\boldsymbol{v}_1^y$ by inspection of the first row of B_2. If $B_2(1, i) < 0$, then the ith control in $\boldsymbol{v}_1^u, \boldsymbol{v}_{1,i}^u$, is $u_{\min_i}$, otherwise it is $u_{\max_i}$. This maximizes the product of the first row of B_2 with $\boldsymbol{v}_1^u$, which is the y_1 component of $B_2\boldsymbol{v}_1^u$.

2) Calculate $\boldsymbol{v}_1^y = B_2\boldsymbol{v}_1^u$.

Now consider a continuous rotation of the y_1–y_2 axes through an angle ϕ to new coordinates y_1'–y_2'. This rotation is not actually performed, but it is described as such to make a point. The rotation is represented by

$$B_2' = TB_2 = \begin{bmatrix} \boldsymbol{r}_1' \\ \boldsymbol{r}_2' \end{bmatrix} \tag{9}$$

We take the rotation matrix to be clockwise if $v_{1,2}^y \geq 0$ and counterclockwise otherwise:

$$T = \begin{bmatrix} \cos\phi & s\sin\phi \\ -s\sin\phi & \cos\phi \end{bmatrix}, \qquad s \equiv \operatorname{sgn}\left(v_{1,2}^y\right) \tag{10}$$

As the axes are rotated, the vertex with the maximum y_1' component changes; through a complete rotation each vertex on $\partial(\Phi_2)$ will be identified, thus generating $\mathcal{V}$. New vertices will be identified as signs of the entries in $\boldsymbol{r}_1'$ change. The angle at which a sign change takes place is the angle at which an entry in $\boldsymbol{r}_1'$ becomes zero. The angle ϕ_j at which the jth entry $\boldsymbol{r}_{1,j}'$ of $\boldsymbol{r}_1'$ is zero is given by

$$\boldsymbol{r}_{1,j}' = \boldsymbol{r}_{1,j}\cos\phi_j + \boldsymbol{r}_{2,j}\sin\phi_j = 0$$

$$\phi_j = \pm\tan^{-1}(\boldsymbol{r}_{1,j}/\boldsymbol{r}_{2,j}), \qquad 0 \leq \phi_j < 2\pi \tag{11}$$

The $\pm$ sign here means to take both angles, which are then placed in the range of principal values $0 \leq \phi_j < 2\pi$. By calculating ϕ_j, where $j = 1, \ldots, m$, we create a list of $2m$ angles. Consider the least of these angles, for example, ϕ_i, where $\phi_i = \min(\phi_j)$ and $j = 1, \ldots, 2m$. In our hypothetical continuous rotation (here counterclockwise), ϕ_i would be the angle at which $\boldsymbol{v}_1^y$ just ceases being the vertex with the maximum y_1' component. The new vertex with maximum y_1' component will differ from $\boldsymbol{v}_1^o$ only in the ith entry. The same would obviously be true for the greatest of the angles generated, except that a clockwise path is described. Hence, the list, originally ordered by control numbering, is sorted from greatest to least (clockwise) or from least to greatest (counterclockwise), along with their original indices. The resulting list of indices is the sequence in which controls change from vertex to vertex, beginning at $\boldsymbol{v}_1^o$ and progressing around $\partial(\Phi)$. Hence, the set of vertices $\mathcal{V}$ and the set of edges $\mathcal{E}$ are easily constructed. In summary, perform the following:

1) Calculate the list $\mathcal{L}_\phi = \{\phi_j\}$; associated with this list is a list of control numbers $\mathcal{L}_u$:

$$\phi_j = \pm\tan^{-1}(\boldsymbol{r}_{1,j}/\boldsymbol{r}_{2,j}), \qquad 0 \leq \phi_j < 2\pi, \qquad j = 1, \ldots, m$$

$$\mathcal{L}_u = \{1,\ 1,\ 2,\ 2, \ldots,\ m,\ m\}$$

2) Sort $\mathcal{L}_\phi$ from least to greatest if $v_{1,2}^y < 0$, or greatest to least if $v_{1,2}^y \geq 0$. Sort $\mathcal{L}_u$ along with $\mathcal{L}_\phi$, and denote the result $\mathcal{L}_u'$.

3) Beginning with $\boldsymbol{v}_1^o$, generate as many elements of $\mathcal{V}$ and $\mathcal{E}$ as needed. Given that the ith element of $\mathcal{L}_u'$ is the integer j, $\boldsymbol{v}_{i+1}^o$ is obtained from $\boldsymbol{v}_i^o$ by changing the sign of the jth component of $\boldsymbol{v}_i^o$. Then any edge $\boldsymbol{e}_{i,i+1}^o = \boldsymbol{v}_i^o \cup \boldsymbol{v}_{i+1}^o$, and $\boldsymbol{e}_{n_v,1}^y = \boldsymbol{v}_{2m}^o \cup \boldsymbol{v}_1^o$.

4) As each new vertex $\boldsymbol{v}_i^o$ is generated, find $\boldsymbol{v}_i^u$ and $\boldsymbol{v}_i^y$. When $\operatorname{sgn}(v_{i,2}^y) \neq \operatorname{sgn}(v_{i+1,2}^y)$, then $\boldsymbol{e}_p^o = \boldsymbol{v}_i^o \cup \boldsymbol{v}_{i+1}^o \equiv \boldsymbol{v}_{i_p}^o \cup \boldsymbol{v}_{i_p+1}^o$ contains the point p_2.

5) The y_3 component of $\boldsymbol{e}_p^y$, at the point its y_2 component is zero, is calculated, and the y_3 component's sign is used to determine if $\boldsymbol{e}_p^y$ is in front of or behind $\mathcal{L}_3$.

Two implementation notes are as follows:

1) The actual angles ϕ are not needed, just their ordering. Actual calculation of the arctangent is not required.

2) Although we describe the list as $0 \leq \phi_j < 2\pi$, the whole range is needed only if the entire convex hull of Φ_2 is to be generated. For purposes of identifying $\boldsymbol{e}_p^o$, only those angles in the first and fourth quadrants are needed.

C. Systematic Rotations

The objective of the rotations $T(\theta)$ is to find two edges, one in front of and the other behind $\mathcal{L}_3$, separated by a suitably small rotation angle. In the current real-time implementations, a bisection method is used. Beginning with the original orientation, the two-dimensional problem is solved, and the edge is tested to determine if it is in front of or behind $\mathcal{L}_3$. Rotations of $\theta = \pi/4$ are performed, and each edge thus generated is tested. When two consecutive edges are found, one in front of and one behind $\mathcal{L}_3$, the angle θ is halved and the direction of rotation reversed. A set number of bisections is performed, and the last two edges satisfying the criteria are retained. Thus, for eight bisections the last rotation is through an angle of about $\frac{1}{6}$ deg, which is sufficiently small for most problems.

D. Solutions

Given a half-line $\mathcal{L}_3$ in the direction of some arbitrary desired objective $\boldsymbol{y}_{3d}$, to find the intersection of $\mathcal{L}_3$ with $\partial(\Phi_3)$ means to find the facet $\boldsymbol{f}^y$, $\dim(\boldsymbol{f}^o) = 2$, on $\partial(\Phi_3)$ that contains the intersection. We will generate candidate facets by finding two edges $\boldsymbol{e}_1^y \subset \partial(\Phi_3)$ and $\boldsymbol{e}_2^y \subset \partial(\Phi_3)$ and their union $\boldsymbol{o}^o = \boldsymbol{e}_1^o \cup \boldsymbol{e}_1^o$. If $\dim(\boldsymbol{o}^o) = 2$, then $\boldsymbol{o}^y$ is a candidate $\boldsymbol{f}^y$. We then test $\boldsymbol{f}^y$ to determine if it contains the intersection of $\mathcal{L}_3$.

1. Testing the Facet

The facet $\boldsymbol{f}^o$ is two-dimensional and has two zeros in (for example) the ith and jth positions. The four vertices of $\boldsymbol{f}^o$ are constructed by assigning all combinations of -1 and $+1$ to those positions. Construct three vertices as the following combinations in the ith and jth positions: (1, 1) (object $\boldsymbol{v}_1^o$), (1, -1) (object $\boldsymbol{v}_2^o$), and (-1, 1) (object $\boldsymbol{v}_3^o$). Hence, $\boldsymbol{e}_{1,2}^o = \boldsymbol{v}_1^o \cup \boldsymbol{v}_2^o$ and $\boldsymbol{e}_{1,3}^o = \boldsymbol{v}_1^o \cup \boldsymbol{v}_3^o$ are both edges of $\boldsymbol{f}^o$. That is, we have a vertex and two edges emanating from that vertex. Thus, all points p_f on $\boldsymbol{f}^y$ are given by the vector equation,

$$p_f = \boldsymbol{v}_1^y + C_{1,2}\left(\boldsymbol{v}_2^y - \boldsymbol{v}_1^y\right) + C_{1,3}\left(\boldsymbol{v}_3^y - \boldsymbol{v}_1^y\right)$$

$$0 \le C_{1,2} \le 1, \qquad 0 \le C_{1,3} \le 1 \tag{12}$$

Note that the vertices $\boldsymbol{v}_1^y$, $\boldsymbol{v}_2^y$, and $\boldsymbol{v}_3^y$ are generally not the same as those defined for Φ_2 [as in Eq. (8)]. The half-line $\mathcal{L}_3$ consists of all points $(\alpha_3, 0, 0) = \alpha_3 \hat{\boldsymbol{e}}_{3,1}$ where $\alpha_3 > 0$. For $\boldsymbol{f}^y$ to contain the intersection p_3 of $\mathcal{L}_3$, it is necessary and sufficient that there be some point on $\mathcal{L}_3$ that satisfies Eq. (12). In other words, we solve the following equation for α_3, $C_{1,2}$, and $C_{1,3}$:

$$\begin{Bmatrix} \alpha_3 \\ C_{1,2} \\ C_{1,3} \end{Bmatrix} = \left[\hat{\boldsymbol{e}}_{3,1}\left(\boldsymbol{v}_1^y - \boldsymbol{v}_2^y\right)\left(\boldsymbol{v}_1^y - \boldsymbol{v}_3^y\right)\right]^{-1} \boldsymbol{v}_1^y \tag{13}$$

Then, if $\alpha_3 > 0$, $0 \le C_{1,2} \le 1$, and $0 \le C_{1,3} \le 1$, $\boldsymbol{f}^y$ is the solution facet, and the optimal solution $\boldsymbol{u}^u$ is

$$\boldsymbol{u}^u = \boldsymbol{v}_1^u + C_{1,2}\left(\boldsymbol{v}_2^u - \boldsymbol{v}_1^u\right) + C_{1,3}\left(\boldsymbol{v}_3^u - \boldsymbol{v}_1^u\right) \tag{14}$$

2. Approximations

In combining the two edges $\boldsymbol{e}_1^y \subset \partial(\Phi_3)$ and $\boldsymbol{e}_2^y \subset \partial(\Phi_3)$ and their union $\boldsymbol{o}^o = \boldsymbol{e}_1^o \cup \boldsymbol{e}_1^o$, two problems can arise:

1) The union $\boldsymbol{o}^o$ may be a facet but not contain p_3.

2) The union $\boldsymbol{o}^o$ may not be a facet, that is, $\dim(\boldsymbol{o}^o) \neq 2$.

Case 1 is rare, and usually occurs when p_3 is very near a vertex. For case 2, It is extremely rare for $\dim(\boldsymbol{o}^o) < 2$. For any of the described cases, however, the following approximation returns very good results.

With reference to Fig. 4, the approximation is performed entirely in the y_2–y_3 plane. The two edges $\boldsymbol{e}_1^y$ and $\boldsymbol{e}_2^y$ are shown, with $\boldsymbol{e}_1^y$ in front of $\mathcal{L}_3$ and $\boldsymbol{e}_2^y$ behind. Also, the vertices defining the two edges are depicted with $\boldsymbol{v}_1^y$ and $\boldsymbol{v}_3^y$ above and $\boldsymbol{v}_2^y$ and $\boldsymbol{v}_4^y$ below $\mathcal{L}_3$.

The approximation is performed by interpolating along each edge to find points $\boldsymbol{w}_1^y$ and $\boldsymbol{w}_2^y$, then interpolating between $\boldsymbol{w}_1^y$ and $\boldsymbol{w}_2^y$ to find the point of intersection with $\mathcal{L}_3$, denoted $\hat{\boldsymbol{p}}_3^y$. That is, the scalar interpolation factors k_i, $i = 1, \ldots, 3$, are determined such that

$$\boldsymbol{w}_1^y = k_1 \boldsymbol{v}_1^y + (1 - k_1)\boldsymbol{v}_2^y, \qquad \boldsymbol{w}_2^y = k_2 \boldsymbol{v}_3^y + (1 - k_2)\boldsymbol{v}_4^y$$

$$\hat{\boldsymbol{p}}_3^y = k_3 \boldsymbol{w}_1^y + (1 - k_3)\boldsymbol{w}_2^y \tag{15}$$

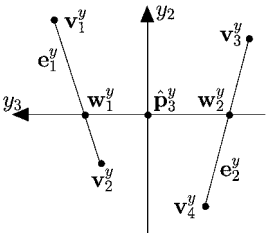

Fig. 4 Approximating the solution.

The solution $\boldsymbol{u}$ is then approximated by

$$\boldsymbol{u} = \hat{\boldsymbol{p}}_3^u = k_3 \boldsymbol{w}_1^u + (1 - k_3)\boldsymbol{w}_2^u \tag{16}$$

In using this approximation, one must check that the resulting control deflections are within their limits and truncate them if necessary.

Note that if $\boldsymbol{e}_1^y$ and $\boldsymbol{e}_2^y$ actually form the solution facet, then the approximation, Eq. (16), will return the same result as Eqs. (13) and (14). Testing the facet is useful for determining if the optimal solution has been achieved, but is not necessary to calculate it.

VI. Implementation

A. MATLAB® Implementation

Reference [9] contains details of the MATLAB implementation. Those results are summarized in the following.

The algorithm described earlier was tested in version 5 of MATLAB. Uniformly random B_3 matrices and desired objectives were generated for numbers of controls ranging from 4 to 20. The values in the B_3 matrices were within the range ± 1, and control limits were set at ± 1 for each case.

It was verified that the algorithm displayed computational requirements that varied linearly with the number of controls. The maximum number of floating-point operations (n_{flops}) for the edge-searching algorithm varied as

$$n_{\text{flops}} = 1701 + 264(m - 4)(r^2 = 0.994) \tag{17}$$

The facet-searching algorithm employed earlier displayed quadratic dependency,

$$n_{\text{flops}} = 6968 - 1244(m - 4) + 901(m - 4)^2(r^2 = 0.999) \tag{18}$$

With respect to errors that resulted from approximations to the solution, 100 problems were generated for each case of 4–20 controls. Percentages of error were defined as follows:

$$\text{Err}(\%) = 100 \cdot \frac{\left\|\boldsymbol{p}_3^y - \hat{\boldsymbol{p}}_3^y\right\|_2}{\left\|\boldsymbol{p}_3^y\right\|_2} \tag{19}$$

where $\boldsymbol{p}_3^y$ is the optimal objective and $\hat{\boldsymbol{p}}_3^y$ is the objective generated by the approximate solution.

With 5 bisections maximum, the number of errors ranged from 3 per 100 cases with few (4–6) controls to 17 per 100 cases for 19 controls. Increasing the maximum permitted number of bisections generally decreased the number of errors. The greatest maximum error encountered was 4.4%, and the least maximum error was 0.034%. The greatest maximum error decreased rapidly as the number of controls was increased. For numbers of controls greater than seven, no errors greater than 1% were generated in any of the several thousand cases tested with numbers of bisections from five to eight.

B. F-15 Advanced Control Technology for Integrated Vehicles (ACTIVE) Simulation

Scalera[11] conducted extensive real-time evaluations of the edge-searching algorithm using an F-15 advanced control technology for integrated vehicles (ACTIVE) simulation. The F-15 ACTIVE control law and mixer were implemented in the modified 2F122A motion-based simulator at Virginia Polytechnic Institute and State University. All simulations were run at a 100-Hz frame rate.

The F-15 ACTIVE utilized 12 control effectors with individual position and rate limits. The effectors were left/right horizontal tails, left/right ailerons, left/right canards, left/right rudders, left/right

trailing-edge flaps, left/right pitch thrust vectoring and left/right yaw thrust vectoring.

A comparison of the original control mixer with the edge-searching control-allocation method was performed. Framewise, or moment rate allocation was implemented with various restoring methods, including minimum norm and minimum drag. The high-fidelity control law of the F-15 ACTIVE was retained. Offline batch as well as real-time piloted simulations were completed to evaluate the performance of the allocators.

The aircraft was flown real time through highly aggressive air-combat maneuvering and terrain-following tasks. Various tests intended to stress the control allocator were performed, including reduced control effectiveness and real-time reconfiguration following simulated control failures.

The results of all of these evaluations were unremarkable. No problems specific to the control-allocation method were encountered during the simulation.

VII. Conclusions

The method of bisecting edge searching is clearly superior, with respect to computational requirements, to that of facet searching because the former increases in computational complexity linearly with the number of controls, whereas the latter increases quadratically. With respect to optimality of the solutions afforded by these two methods, facet searching is superior because it always returns the optimal solution. However, the frequency of occurrence of suboptimal solutions associated with bisecting edge searching is not great (typically less than 10% of the number of cases examined), and the magnitude of the error, when encountered, is small. These errors are probably inconsequential compared to the errors in aerodynamic data used to generate control effectiveness matrices.

The relatively modest computational requirements of bisecting edge searching for typical aircraft applications (10–20 control effectors) mean that real-time flight-control applications are possible. The results of the high-fidelity piloted simulation of the F-15 ACTIVE with 12 control effectors indicated that the utilization of the edge-searching algorithm on a tactical aircraft with a highly redundant control suite is practical.

Acknowledgments

The research was performed under NASA Grants NAG-1-1449 and NAG-1-2111, supervised by John V. Foster of the NASA Langley Research Center. The inspiration for the method presented in this paper grew out of an informal collaboration of the author with Ranko Bojanic, retired mathematics professor of Ohio State University.

References

[1]Durham, W. C., "Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics,* Vol. 16, No. 4, 1993, pp. 717–725.

[2]Durham, W. C., "Attainable Moments for the Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics,* Vol. 17, No. 6, 1994, pp. 1371–1373.

[3]Durham, W. C., "Constrained Control Allocation: Three-Moment Problem," *Journal of Guidance, Control, and Dynamics,* Vol. 17, No. 2, 1994, pp. 330–336.

[4]Bolling, J. G., and Durham, W. C., "Control Allocation with Adaptive Failure Control," AIAA Paper 97-3775, Aug. 1997.

[5]Bordignon, K. A., "Constrained Control Allocation for Systems with Redundant Control Effectors," Ph.D. Thesis, Dept. of Aerospace and Ocean Engineering, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, Dec. 1996.

[6]Bolling, J. G., "Implementation of Constrained Control Allocation Techniques Using an Aerodynamic Model of an F-15 Aircraft," M.S. Thesis, Dept. of Aerospace and Ocean Engineering, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, May 1997.

[7]Buffington, J. M., "Tailless Aircraft Control Allocation," AIAA Paper 97-3605, Aug. 1997.

[8]Enns, D., "Control Allocation Approaches," AIAA Paper 98-4109, Aug. 1998.

[9]Durham, W. C., "Computationally Efficient Control Allocation," AIAA Paper 99-4214, Aug. 1999.

[10]Ziegler, G. M., "Lectures on Polytopes," *Graduate Texts in Mathematics*, edited by S. Axler, F. W. Gehring, and K. A. Ribet, 1st (rev.) ed., Vol. 152, Springer-Verlag, New York, 1995, pp. 51, 52.

[11]Scalera, K. R., "A Comparison of Control Allocation Methods for the F-15 ACTIVE Research Aircraft Utilizing Real-Time Piloted Simulations," M.S. Thesis, Dept. of Aerospace and Ocean Engineering, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, July 1999.