$(n'-1)\times 1$ vectors; $\boldsymbol{a}_{22}$ is a $(n'-1)\times(n'-1)$ matrix; and the subscript $y$ represents the component along $y$ axis. The fundamental matrix of the autopilot, if obtainable, is expressed as

$$\boldsymbol{\Phi}^A(t,t_0)=\begin{bmatrix}\Phi^A_{11}(t,t_0) & \boldsymbol{\Phi}^A_{12}(t,t_0)\\ \boldsymbol{\Phi}^A_{21}(t,t_0) & \boldsymbol{\Phi}^A_{22}(t,t_0)\end{bmatrix} \tag{A4}$$

where $\Phi^A_{11}$ is scalar, $\boldsymbol{\Phi}^A_{21}$ and $\boldsymbol{\Phi}^{A^T}_{12}$ are $(n'-1)\times 1$ vectors, and $\boldsymbol{\Phi}^A_{22}$ is a $(n'-1)\times(n'-1)$ matrix. The ZEM for the interceptor is obtained as

$$\mathrm{ZEM}_y = y^*(t_f) - y - \nu t_g - \int_t^{t_f}(t_f-\xi)c_y(\xi)\,\mathrm{d}\xi - \begin{bmatrix}\int_t^{t_k}(t_f-\xi)\Phi^A_{11}(\xi,t)\,\mathrm{d}\xi & \int_t^{t_k}(t_f-\xi)\boldsymbol{\Phi}^A_{12}(\xi,t)\,\mathrm{d}\xi\end{bmatrix}\begin{bmatrix}a_{yc}\\ \boldsymbol{p}\end{bmatrix} \tag{A5}$$

where $\nu=\dot{y}$. Differentiating the preceding relation yields the relation $\dot{\mathrm{ZEM}}_y=-\alpha(t)u_y$ where

$$\alpha(t)=b_1(t)\int_t^{t_k}(t_f-\xi)\Phi^A_{11}(\xi,t)\,\mathrm{d}\xi + \left[\int_t^{t_k}(t_f-\xi)\boldsymbol{\Phi}^A_{12}(\xi,t)\,\mathrm{d}\xi\right]\boldsymbol{b}_2(t) \tag{A6}$$

Thus, we can write $\dot{\mathbf{ZEM}}(t)=-\alpha(t)\boldsymbol{u}$, which integrates for the first class into

$$\boldsymbol{m}=\mathbf{ZEM}(t)-\int_t^{t_c}\alpha(\xi)\boldsymbol{u}(\xi)\,\mathrm{d}\xi \tag{A7}$$

Therefore, Eq. (9) holds for the first class. The preceding relation is valid for the second class when we substitute $t_b$ instead of $t_c$ in this relation. Therefore, the results of Sec. III are applicable for the two classes of systems.

In the case that $A$ and $B$ are time invariant and $t_0=0$, Eq. (5A) simplifies to

$$\mathrm{ZEM}_y = y^*(t_f) - y - \nu t_g - \int_t^{t_f}(t_f-\xi)c_y(\xi)\,\mathrm{d}\xi - \left\{\mathcal{L}^{-1}\left[\tilde{R}(s)\frac{\tilde{a}_{yc}(s)}{a_{yc}(0)}\right]\Bigg|_{t_k-t}\ \mathcal{L}^{-1}\left[\tilde{R}(s)\frac{\tilde{a}_{yc}(s)}{\boldsymbol{p}(0)}\right]\Bigg|_{t_k-t}\right\}\begin{bmatrix}a_{yc}\\ \boldsymbol{p}\end{bmatrix} \tag{A8}$$

where $s$ is the Laplace domain variable, $\mathcal{L}^{-1}$ is the inverse Laplace transform operator, $\tilde{a}_{yc}(s)/a_{yc}(0)=\tilde{\Phi}^A_{11}(s)$ and $\tilde{a}_{yc}(s)/\boldsymbol{p}(0)=\tilde{\boldsymbol{\Phi}}^A_{12}(s)$, $\tilde{R}(s)=s^{-2}+(t_f-t_k)s^{-1}$, and the superscript tilde indicates the Laplace transform. We also have

$$\alpha(t)=\mathcal{L}^{-1}\left[\frac{1+(t_f-t_k)s}{s^2}\frac{\tilde{a}_{yc}(s)}{\tilde{u}_y(s)}\right]\Bigg|_{t_k-t} \tag{A9}$$

where $\tilde{a}_{yc}(s)/\tilde{u}_y(s)$ is the autopilot transfer function for the $y$ channel. Note that the two classes of systems are identical for a perfect autopilot.

## References


[1]Zarchan, P., *Tactical and Strategic Missile Guidance*, 3rd ed., Progress in Astronautics and Aeronautics, Vol. 176, AIAA, Reston, VA, 1997, Chap. 8.

[2]Hough, M. E., "Optimal Guidance and Nonlinear Estimation for Interception of Accelerating Targets," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 5, 1995, pp. 959–968.

[3]Cottrell, R. G., "Optimal Intercept Guidance for Short-Range Tactical Missiles," *AIAA Journal*, Vol. 9, No. 7, 1971, pp. 1414, 1415.

[4]Rusnak, I., and Meir, L., "Optimal Guidance Law for Acceleration Constrained Missile and Maneuvering Target," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 26, No. 4, 1990, pp. 618–624.

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

[6]Blackburn, T. R., "Method for Improving Autopilot Lag Compensation in Intercept Guidance," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 3, 1996, pp. 724–726.

[7]Massoumnia, M.-A., "Optimal Midcourse Guidance Law for Fixed-Interval Propulsive Maneuvers," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 3, 1995, pp. 465–470.

[8]Jalali-Naini, S. H., and Ebrahimi, B., "Modified Guidance with N-Fixed-Interval Propulsive Maneuvers," AIAA Paper 2002-4951, Aug. 2002.

[9]Jalali-Naini, S. H., "Generalized Explicit Guidance Law for Time-Variant High-Order Dynamics," *Proceedings of the 4th Iranian Aerospace Society Conference*, Iranian Aerospace Society, Tehran, Iran, 2003, pp. 249–261.

[10]Jalali-Naini, S. H., "Modern Midcourse Guidance Law with High-Order Dynamics," AIAA Paper 2003-5356, Aug. 2003.

[11]Jalali-Naini, S. H., and Pourtakdoust, S. H., "Modern Guidance Laws Based on Minimum Fuel Consumption," *Proceedings of the 5th Iranian Aerospace Society Conference*, Iranian Aerospace Society, Tehran, Iran, 2004, pp. 221–232.


# Methods for Compensating for Control Allocator and Actuator Interactions


Michael W. Oppenheimer* and David B. Doman†

*U.S. Air Force Research Laboratory, Wright–Patterson Air Force Base, Ohio 45433-7531*


## Introduction

NUMEROUS control allocation algorithms have been developed for aircraft for the purpose of providing commands to suites of control effectors to produce desired moments or accelerations. A number of approaches have been developed that ensure that the commands provided to the effectors are physically realizable. These actuator command signals are feasible in the sense that they do not exceed hardware rate and position limits. Buffington[1] developed a linear programming-based approach that separately considered cases where sufficient control power was available to meet a moment demand and a control deficiency case where the moment deficiency was minimized. Bodson[2] developed a linear programming approach where the sufficiency and deficiency branches were considered simultaneously, which resulted in computational savings. Quadratic programming approaches have also been considered.[3] In the early 1990s, Durham[4,5] developed a constrained control allocation approach called direct allocation that was based on geometric concepts of attainable moment sets. Page and Steinberg[6] as well as Bodson[2] have presented excellent survey papers that compare and contrast many of the control allocation approaches developed over the last two decades. Recent work in the area has resulted in the development of control allocation algorithms that can accommodate cases where the moments or accelerations produced by the control effectors are nonlinear functions of the effector position.[7,8]

A review of the constrained control allocation literature shows that the coupling effects that result from combining constrained

Received 11 December 2003; revision received 9 April 2004; accepted for publication 14 April 2004. 

*Electronics Engineer, Control Theory and Optimization Branch, 2210 Eighth Street, Building 146, Room 305; Michael.Oppenheimer@wpafb.af.mil. Member AIAA.

†Senior Aerospace Engineer, Control Theory and Optimization Branch, 2210 Eighth Street, Building 146, Room 305; David.Doman@wpafb.af.mil. Senior Member AIAA.

control allocators and actuator dynamics have largely been ignored, although some recent research has been performed on directly including actuator dynamics in the control allocation problem.[9,10] The underlying assumption of most previous work is that actuators respond instantaneously to commands. This assumption may at first seem justified because, in practice, actuator dynamics are typically much faster than the rigid-body modes that are to be controlled. However, interactions between a constrained control allocator and an actuator with linear dynamics can result in a system that falls well short of its potential. These interactions can significantly reduce the effective rate limits of the system while at the same time yield control effector positions that are different than the commanded positions.[11]

The objective of this work is to develop a method, which postprocesses the output of a control allocation algorithm, to compensate for actuator dynamics. In reaching this objective, it is of great importance to consider the real-world applicability of the approach. To obtain an algorithm that can operate in real time on a typical flight computer, it is pertinent to not add to the complexity of the constrained control allocation algorithm. Although the results by Härkegård[9] and Venkataraman et al.[10] overcome some of the issues raised here, it is not certain that either method can guarantee convergence in one flight control system update for both the saturated and unsaturated cases. One method that can accomplish the goal is to postprocess the output of the control allocation algorithm to overdrive the actuators so that at the end of a sampling interval the actual actuator positions are equivalent to the desired actuator positions. This is the approach taken in this work, and it yields a simple, but effective, means of compensating for actuator dynamics.

Bolling[11] has shown that the interaction between first-order actuator dynamics and constrained control allocation algorithms can be eliminated by overdriving the actuators. In this work, the details of the interaction between constrained control allocators and first-order actuator dynamics are provided. This scheme is then extended to a second-order actuator dynamics case. Simulation results are presented for a four control surface vehicle using a linear programming-based control allocation algorithm that takes into account control effector position and rate limits. The results show that the desired control effector positions can be obtained by postprocessing the control allocation commands.

## Effective Rate Limits and Attenuation of Zero-Order-Hold Inputs for First-Order Actuator Dynamics

Figure 1 shows the system to be analyzed in this work. Inputs to the control allocation algorithm consist of a vector of desired moment or acceleration commands, $\boldsymbol{d}_{\text{des}} \in \mathbb{R}^n$, and a vector containing the current control surface deflections, $\boldsymbol{\delta} \in \mathbb{R}^m$. The output of the control allocator is the commanded control surface deflection vector $\boldsymbol{\delta}_{\text{cmd}} \in \mathbb{R}^m$. The actuator dynamics respond to $\boldsymbol{\delta}_{\text{cmd}}$ to produce the actual control deflections $\boldsymbol{\delta}$. The individual actuators are assumed to have uncoupled dynamics, hardware rate limits $\pm\dot{\boldsymbol{\delta}}_{\max}$, and position limits $\boldsymbol{\delta}_{\min}$ and $\boldsymbol{\delta}_{\max}$. In most control allocator implementations, rate limits are taken into account by converting them into effective position limits at the end of the next sampling period and constraining the effector commands to respect the most restrictive of the rate or position limits, that is,

$$\bar{\boldsymbol{\delta}} = \min\left(\boldsymbol{\delta}_{\max}, \boldsymbol{\delta} + \dot{\boldsymbol{\delta}}_{\max_{\text{CA}}} \Delta t\right) : \underline{\boldsymbol{\delta}} = \max\left(\boldsymbol{\delta}_{\min}, \boldsymbol{\delta} - \dot{\boldsymbol{\delta}}_{\max_{\text{CA}}} \Delta t\right) \tag{1}$$

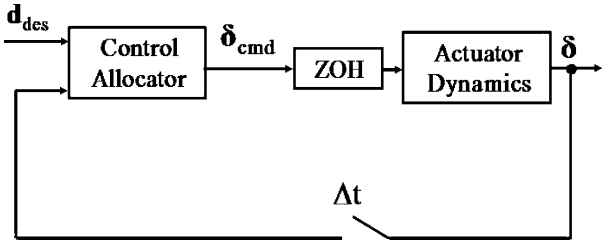


**Fig. 1 Control allocator and actuator interconnection.**

where $\boldsymbol{\delta}$ is the current location of the control effectors, $\bar{\boldsymbol{\delta}}$ and $\underline{\boldsymbol{\delta}}$ are the most restrictive upper and lower bounds on the effectors, respectively, and $\Delta t$ is the sampling period of the digital flight control system.

In typical implementations, each element of the vector of rate limits provided to the control allocation algorithm, $\dot{\boldsymbol{\delta}}_{\max_{\text{CA}}}$, is taken to be the true hardware rate limit of the corresponding actuator. Hence, the software rate limit is equal to the hardware rate limit or $\dot{\boldsymbol{\delta}}_{\max_{\text{CA}}} = \dot{\boldsymbol{\delta}}_{\max}$; however, as was shown by Bolling,[11] the effective rate limit of a scalar system composed of a constrained control allocation algorithm and first-order actuator dynamics of the form

$$\delta(s)/\delta_{\text{cmd}}(s) = a/(s+a) \tag{2}$$

becomes

$$\dot{\delta}_{\max_{\text{EFF}}} = \Gamma \dot{\delta}_{\max_{\text{CA}}} \tag{3}$$

where $\Gamma \triangleq 1 - e^{-a\Delta t}$, $\dot{\delta}_{\max_{\text{EFF}}}$ is the effective rate limit, and $\dot{\delta}_{\max_{\text{CA}}}$ is the rate limit provided to the control allocation algorithm by the user. For example, when $a = 20$ rad/s, $\dot{\delta}_{\max_{\text{CA}}} = 60$ deg/s, and the flight control system operates at 50 Hz ($\Delta t = 0.02$ s), the effective rate limit is 19.78 deg/s. This result is well short of the ideal 60 deg/s value.

A few comments are in order regarding Fig. 1 and the analysis described. First, note that the instantaneous position limit given by Eq. (1) makes use of a sampled vector of actuator position measurements to compute the maximum distance that the actuator can move in the next time instant. This is in contrast to using the previous value of actuator command vector $\boldsymbol{\delta}_{\text{cmd}}$, as is often done in simulation. The motivation for using actuator measurements is that when actuator dynamics, disturbances, and uncertainties are taken into account, the actuator command vector and the true actuator positions will differ. This difference can cause a control allocator to generate inappropriate actuator commands that do not deliver the desired moments or accelerations. Thus, feeding the measured actuator position vector to the control allocator has the advantage of reducing uncertainty in the actuator position. It is important, however, to be aware of the effects of using the measured actuator position vector.

In the same manner in which rate limits are effectively attenuated by actuator dynamics, so are the magnitudes of the commanded changes to control effector positions. Refer to Fig. 1: The desired situation would be for $\boldsymbol{\delta} = \boldsymbol{\delta}_{\text{cmd}}$. However, actuator dynamics alter the command signals so that, in general, $\boldsymbol{\delta} \neq \boldsymbol{\delta}_{\text{cmd}}$. For actuators with high bandwidths relative to the rigid-body modes, this is not a serious concern. However, situations exist where the actuator dynamics are not sufficiently fast and need to be taken into account. In this section, the effects of first-order actuator dynamics on the system shown in Fig. 1 will be discussed. To illustrate the idea for postprocessing control allocator outputs, the method proposed by Bolling[11] that compensates for both magnitude and rate limit attenuation will be described.

The discrete-time representation of the first-order actuator dynamics equation is given by

$$\delta(t_{k+1}) = \Phi\delta(t_k) + \Gamma\delta_{\text{cmd}}(t_k) \tag{4}$$

where $\Phi \triangleq e^{-a\Delta t}$, $\Gamma \triangleq 1 - e^{-a\Delta t}$, and it has been assumed that the input to the actuator dynamics, $\delta_{\text{cmd}}(t_k)$, is held constant over each sampling period. The command to the actuator can be written as

$$\delta_{\text{cmd}}(t_k) = \Delta\delta_{\text{cmd}_{\text{CA}}}(t_k) + \delta(t_k) \tag{5}$$

where the commanded incremental change in actuator position over one time step is defined by $\Delta\delta_{\text{cmd}_{\text{CA}}}(t_k) \triangleq \delta_{\text{cmd}_{\text{CA}}}(t_k) - \delta(t_k)$ and where $\delta_{\text{cmd}_{\text{CA}}}(t_k)$ is the actuator position command from the control allocator. Because the effector commands are held constant over each sampling period, $\Delta\delta_{\text{cmd}_{\text{CA}}}(t_k)$ will appear to the actuators to be a step command from the measured position. Recall that $\delta_{\text{cmd}_{\text{CA}}}(t_k)$ is calculated by the control allocator based on the assumption that

the effector will respond instantaneously to commands. Substituting Eq. (5) into Eq. (4) yields

$$\delta(t_{k+1}) = \Phi\delta(t_k) + \Gamma\left[\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k)\right] \tag{6}$$

Because $\Gamma < 1$, the incremental command signal from the control allocation algorithm, $\Delta\delta_{\mathrm{cmd_{CA}}}(t_k)$, is attenuated by the actuator dynamics; thus, $\delta(t_{k+1}) \neq \delta_{\mathrm{cmd_{CA}}}(t_k)$. The objective is to find a gain $M$ that modifies the output of the control allocation algorithm such that $\delta(t_{k+1}) = \delta_{\mathrm{cmd_{CA}}}(t_k) = \Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k)$. Hence,

$$\delta(t_{k+1}) = \Phi\delta(t_k) + \Gamma\left[M\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k)\right] \tag{7}$$

and solving for $M$ yields

$$M = 1/\Gamma \tag{8}$$

Thus, the actuator command signal must be modified such that

$$\tilde{\delta}_{\mathrm{cmd}}(t_k) = M\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k) = (1/\Gamma_s)\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k) \tag{9}$$

where $\Gamma_s = 1 - e^{-a_{\mathrm{nom}}\Delta t}$ is the software scaling factor and $a_{\mathrm{nom}}$ is the nominal bandwidth of the first-order actuators. Replacing $\delta_{\mathrm{cmd}}(t_k)$ in Eq. (4) with $\tilde{\delta}_{\mathrm{cmd}}(t_k)$ from Eq. (9) yields

$$\begin{aligned}\delta(t_{k+1}) &= \Phi\delta(t_k) + \Gamma\tilde{\delta}_{\mathrm{cmd}}(t_k)\\ &= \Phi\delta(t_k) + \Gamma\left[(1/\Gamma_s)\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k)\right]\end{aligned} \tag{10}$$

which, when $a = a_{\mathrm{nom}}$ and thus $\Gamma = \Gamma_s$, results in $\delta(t_{k+1}) = \delta_{\mathrm{cmd_{CA}}}(t_k)$. The distinction between $\Gamma$ and $\Gamma_s$ is made because it is possible for the actuators to at some point have a bandwidth that is less than the nominal value, due to power loss, partial failure, etc. In other words, the potential exists for $a < a_{\mathrm{nom}}$. In this analysis, it has been assumed that $a_{\mathrm{nom}}$ is the nominal bandwidth of the actuators and is an upper bound on bandwidth. That is, if $a \neq a_{\mathrm{nom}}$, then $a < a_{\mathrm{nom}}$. In other words, the bandwidth of the actuator $a$ cannot be larger than the nominal bandwidth, $a_{\mathrm{nom}}$. Under this assumption, $\Gamma \leq \Gamma_s$.

Because $\Gamma_s$ can be computed from the known quantities $a_{\mathrm{nom}}$ and $\Delta t$, one can compensate for command increment attenuation using Eq. (9). For a bank of decoupled first-order actuators with nominal bandwidths of $a_{\mathrm{nom}_i}$, corresponding values of $\Gamma_{s_i}$ can be computed using $\Gamma_{s_i} = (1 - e^{-a_{\mathrm{nom}_i}\Delta t})$. The command increment compensation can then be implemented in discrete time as shown in the block diagram of Fig. 2. In Fig. 2, the dashed lines are for compensation for second-order actuator dynamics, as will be discussed in the next section, and, therefore, are not part of the first-order actuator dynamics compensation scheme. Note that, for multiple actuators, $\boldsymbol{M}$ in Fig. 2 is a diagonal matrix with the entries on the main diagonal being $\Gamma_{s_1}, \Gamma_{s_2}, \ldots, \Gamma_{s_m}$, where the subscript $m$ is defined as the number of control effectors. Hence, the magnitude of the control allocation command increment is modified to counteract the attenuation that results from the interaction between first-order actuator dynamics and the control allocator. Note that the allocator continues to operate under the assumption that the actuator will respond instantaneously and ensures that the commands obey rate and position limits. It is after the allocator computes a new set of commands that the command increment is scaled and added to the measured actuator position. A beneficial consequence of this is that the effective rate limit will be equal to both the hardware and software rate limit when the magnitude increment is scaled in this fashion. Thus, when this technique is used, there is no need to adjust the software rate limit.

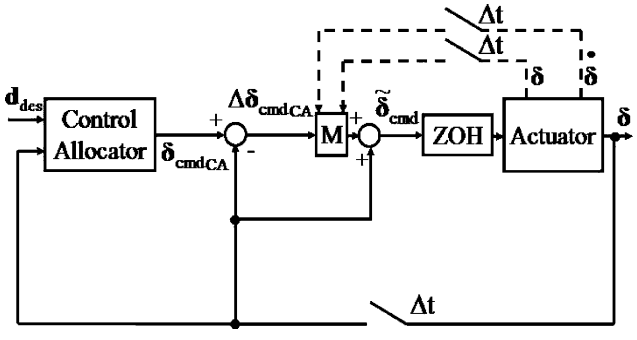


**Fig. 2 Block diagram of command increment compensation.**

Note that this technique effectively modifies the gain of the inner loop. In the typical situation, where actuator dynamics are assumed to be much faster than the rigid-body modes to control and are, hence, ignored, a stability analysis would use $\delta_{\mathrm{cmd}}$ as the input to the plant. In this case, it can be shown that for $\Gamma \leq \Gamma_s$, even after this compensation scheme is applied, $\delta \leq \delta_{\mathrm{cmd}}$. Therefore, something less than or equal to $\delta_{\mathrm{cmd}}$ is applied to the plant, and the loop gain is reduced. The performance of the entire control system will be burdened with the task of mitigating the effects of $\delta \neq \delta_{\mathrm{cmd}}$. Care must be taken in applying this approach because it is possible for the loop gain to be reduced.

## Attenuation of Zero-Order-Hold Inputs for Second-Order Actuator Dynamics

With the knowledge gained from the first-order actuator dynamics case, it is now possible to extend the earlier results to second-order actuator dynamics. Let the second-order actuator dynamics be represented by the general equation

$$\delta(s)/\delta_{\mathrm{cmd}}(s) = k\Big/\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right) \tag{11}$$

which, in state-space form, becomes

$$\begin{aligned}\begin{bmatrix}\dot{\delta}(t)\\ \ddot{\delta}(t)\end{bmatrix} &= \begin{bmatrix}0 & 1\\ -\omega_n^2 & -2\zeta\omega_n\end{bmatrix}\begin{bmatrix}\delta(t)\\ \dot{\delta}(t)\end{bmatrix} + \begin{bmatrix}0\\ k\end{bmatrix}\delta_{\mathrm{cmd}}(t)\\ &= \boldsymbol{A}\begin{bmatrix}\delta(t)\\ \dot{\delta}(t)\end{bmatrix} + \boldsymbol{B}\delta_{\mathrm{cmd}}(t)\end{aligned} \tag{12}$$

$$\delta(t) = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\delta(t)\\ \dot{\delta}(t)\end{bmatrix} = \boldsymbol{C}\begin{bmatrix}\delta(t)\\ \dot{\delta}(t)\end{bmatrix} \tag{13}$$

When Eqs. (12) and (13) are used, the discrete-time solution to the second-order actuator dynamics differential equation becomes

$$\delta(t_{k+1}) = \begin{bmatrix}\Phi_{1,1} & \Phi_{1,2}\end{bmatrix}\begin{bmatrix}\delta(t_k)\\ \dot{\delta}(t_k)\end{bmatrix} + \delta_{\mathrm{cmd}}(t_k)\int_{t_k}^{t_{k+1}} k\Phi_{1,2}(t_{k+1} - \tau)\,\mathrm{d}\tau \tag{14}$$

where

$$\boldsymbol{\Phi} = e^{A(t_{k+1} - t_k)} = \begin{bmatrix}\Phi_{1,1} & \Phi_{1,2}\\ \Phi_{2,1} & \Phi_{2,2}\end{bmatrix} \tag{15}$$

is the state transition matrix. Because $\boldsymbol{A}$ is constant, the state transition matrix depends on the time difference $\Delta t = t_{k+1} - t_k$, and the explicit dependence on time has been eliminated. Performing the operations required in Eqs. (14) and (15) produces

$$\delta(t_{k+1}) = C_1\delta(t_k) + C_2\dot{\delta}(t_k) + C_3\delta_{\mathrm{cmd}}(t_k) \tag{16}$$

where

$$\begin{aligned}C_1 &= (\omega_n/\omega_d)e^{\sigma\Delta t}\sin[\omega_d\Delta t + \arctan(\omega_d/-\sigma)]\\ C_2 &= \left(e^{\sigma\Delta t}\Big/\omega_d\right)\sin[\omega_d\Delta t]\\ C_3 &= (k/\omega_d)\left\{\omega_d + e^{\sigma\Delta t}[\sigma\sin(\omega_d\Delta t) - \omega_d\cos(\omega_d\Delta t)]\right\}\Big/\omega_n^2\end{aligned} \tag{17}$$

where $\omega_d = \omega_n\sqrt{(1 - \zeta^2)}$ and $\sigma = -\zeta\omega_n$. Substituting Eq. (5) into Eq. (16) produces

$$\delta(t_{k+1}) = C_1\delta(t_k) + C_2\dot{\delta}(t_k) + C_3\left[\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k)\right] \tag{18}$$

The objective is to find a gain $M$ that will modify $\Delta\delta_{\mathrm{cmd_{CA}}}(t_k)$ in such a way that $\delta(t_{k+1}) = \delta_{\mathrm{cmd_{CA}}}(t_k)$. Hence, it is desired to find $M$ such that

$$\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k) = C_1\delta(t_k) + C_2\dot{\delta}(t_k) + C_3\left[M\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + \delta(t_k)\right] \tag{19}$$

Solving for $M$ gives

$$M = \frac{\Delta\delta_{\mathrm{cmd_{CA}}}(t_k) + (1 - C_3 - C_1)\delta(t_k) - C_2\dot{\delta}(t_k)}{C_3\Delta\delta_{\mathrm{cmd_{CA}}}(t_k)} \tag{20}$$

Clearly, unlike the simple first-order case, this compensation is much more complex. In fact, this requires not only the position of the control effector, but also the rate of change of the control effector. As with the first-order case, this method postprocesses the control effector commands, without modifying the control allocation algorithm. Refer to Fig. 2: For a system with more than one actuator, $\boldsymbol{M}$ would be a diagonal matrix with entries along the main diagonal being $M_1$, $M_2, \ldots, M_m$. Here, $M_i$ would be computed using Eq. (20) and $\omega_{n_i}$ and $\zeta_i$ corresponding to the $i$th actuator. Also note that a combination of first- and second-order actuators could be treated by using Eq. (8) for the first-order actuators and Eq. (20) for the second-order actuators. For second-order actuators, the dashed lines in Fig. 2 would be used because the scaling factor, Eq. (20), depends on the actuator's past position and rate.

Recall, in the first-order actuator dynamics case, that due to a power loss, partial failure, etc., it is possible for $\Gamma \leq \Gamma_s$, that is, for the bandwidth of the physical actuator to be less than the bandwidth used to determine the scaling factors $M_i$. In the second-order actuator dynamics case, two parameters can vary, namely, the damping ratio $\zeta$ and the natural frequency $\omega_n$. If the control system time step is large enough, this could have an adverse impact on the performance of the system. However, for a 50-Hz update rate, the time step is 0.02 s and the impact of differences between the true $\zeta$ and $\omega_n$ and those used in computing $M_i$ are minimal. Basically, if the sampling is fast enough, then the differences between two actuator positions at the end of an update cycle are small even when the actuators have slightly different damping factors and natural frequencies.

## Simulation Results

In this section, results from a simulation of the system displayed in Fig. 2 will be shown. A rate and position constrained linear programming-based control allocator will be utilized in this work. In this case, the control allocation algorithm's objective, referring to Fig. 1, is to find $\delta_{\mathrm{cmd}}$ such that

$$\boldsymbol{d}_{\mathrm{des}} = \boldsymbol{B}\boldsymbol{\delta}_{\mathrm{cmd}} \tag{21}$$

where $\boldsymbol{B}$ is the control effectiveness matrix and $\boldsymbol{d}_{\mathrm{des}}$ is typically a set of moment or acceleration commands for the roll, pitch, and yaw axes. Although, if feasible, the control allocator will be able to find a $\boldsymbol{\delta}_{\mathrm{cmd}}$ such that Eq. (21) holds, the real test is to determine what happens after the actuator dynamics operate on $\boldsymbol{\delta}_{\mathrm{cmd}}$. Hence, the overall system goal is to achieve $\boldsymbol{\delta}$ such that

$$\boldsymbol{d}_{\mathrm{des}} = \boldsymbol{B}\boldsymbol{\delta} \tag{22}$$

Equation (22) is the metric on which the quality of results will be judged. In this example, four control effectors are present, and the control effectiveness matrix is fixed at

$$\boldsymbol{B} = \begin{bmatrix} -0.4 & 0.4 & -0.1 & 0.1 \\ -0.1 & -0.1 & -0.6 & -0.6 \\ -0.1 & 0.1 & -0.1 & 0.1 \end{bmatrix} \tag{23}$$

where the elements of $\boldsymbol{B}$ have units of radians per second squared per degree. Because there are more control effectors (four) than axes to control (three), a control mixer or allocator must be used. In this work, a flight-tested linear programming-based control allocation algorithm with rate and position limiting will be used.[2,7] Note, however, that any control allocation method could be used.

In the following simulations, only second-order actuator dynamics will be used. Each of the four actuators have the following model:

$$\delta(s)/\delta_{\mathrm{cmd}}(s) = 1/(s^2 + 7.071s + 25) \tag{24}$$

which gives $\zeta = \sqrt{(2)}/2$, $\omega_n = 5$, and $k = 1$. Each actuator is rate and position limited by the following values:

$$\begin{aligned} \boldsymbol{\delta}_{\min} &= [-1.5 \quad -1.5 \quad -1.5 \quad -1.5] \text{ (deg)} \\ \boldsymbol{\delta}_{\max} &= [0.3 \quad 1.5 \quad 1.5 \quad 1.5] \text{ (deg)} \\ \dot{\boldsymbol{\delta}}_{\max_{\mathrm{CA}}} &= [10 \quad 10 \quad 10 \quad 1.5] \text{ (deg/s)} \end{aligned} \tag{25}$$

These limits were selected so that at least one position and one rate limit were in effect at some time during the simulation. This was done to show that the method developed in this work is applicable when control effectors are saturated. As will be shown, when compensation is used, actuator 1 becomes upper position limited and actuator 4 becomes rate limited as the frequency of the commands increases. The command signals $\boldsymbol{d}_{\mathrm{des}}$ consist of chirps of magnitude 0.025, 0.075, and 0.05 rad/s$^2$ in the roll, pitch, and yaw channels and where the frequency ranged from 0.5 to 2 Hz as a linear function of time over a 10-s time interval.

Simulation runs were performed with and without compensation for magnitude attenuation due to second-order actuator dynamics. Ideal conditions are when the actuator dynamics can be represented by $\delta(s)/\delta_{\mathrm{cmd}}(s) = 1$. If sufficient control authority and ideal conditions exist, then the control system would achieve $\boldsymbol{d}_{\mathrm{des}} = \boldsymbol{B}\boldsymbol{\delta}$, which is the best possible performance. Figures 3 and 4 show $\boldsymbol{d}_{\mathrm{des}}$ and $\boldsymbol{B}\boldsymbol{\delta}$ without and with the magnitude compensation described in Eq. (20) applied. Clearly, when the magnitude compensation is not used (Fig. 3), $\boldsymbol{d}_{\mathrm{des}} \neq \boldsymbol{B}\boldsymbol{\delta}$, and a large error exists between these two quantities. When magnitude compensation is used (Fig. 4), however, $\boldsymbol{d}_{\mathrm{des}} \cong \boldsymbol{B}\boldsymbol{\delta}$ and near ideal performance is achieved.

Figures 5 and 6 show the control effector commands $\boldsymbol{\delta}_{\mathrm{cmd_{CA}}}$ and actual deflections $\boldsymbol{\delta}$ when magnitude compensation is not utilized. As expected, the actual deflections are considerably different than the commanded deflections, and this directly translates to the deviations between $\boldsymbol{d}_{\mathrm{des}}$ and $\boldsymbol{B}\boldsymbol{\delta}$ as shown in Fig. 3. On the other hand, Figs. 7 and 8 show the control effector commands $\boldsymbol{\delta}_{\mathrm{cmd_{CA}}}$ and actual deflections $\boldsymbol{\delta}$ when magnitude compensation is used. In this case, the actual deflections are nearly equal to the commands, and as a result,

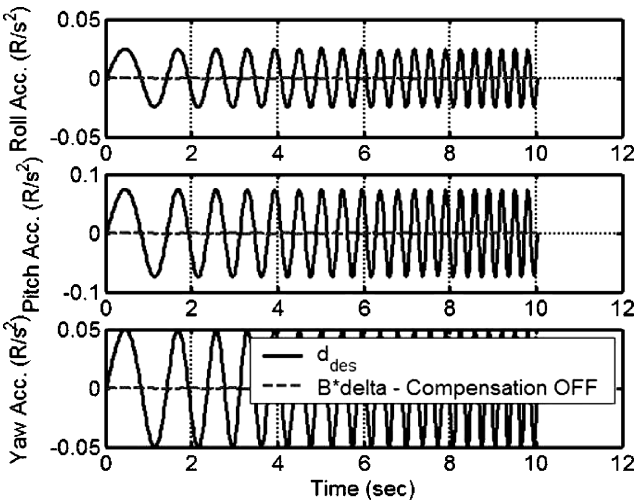


**Fig. 3 Commanded, $d_{\mathrm{des}}$, and simulated, $B\delta$, angular accelerations (radians per second squared) compensation off.**

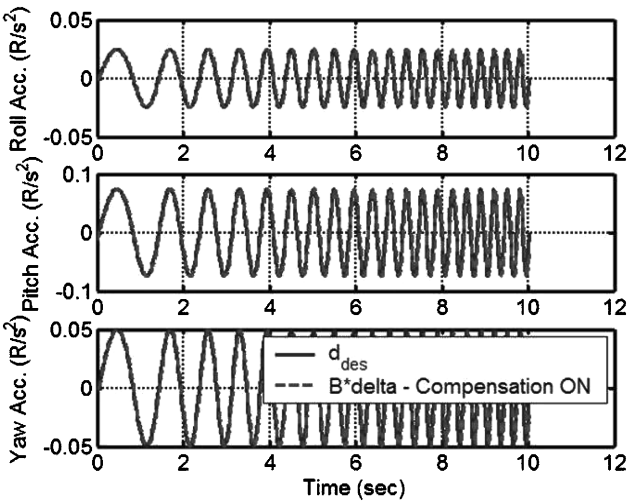


**Fig. 4 Commanded, $d_{\text{des}}$, and simulated, $B\delta$, angular accelerations (radians per second squared) compensation on.**

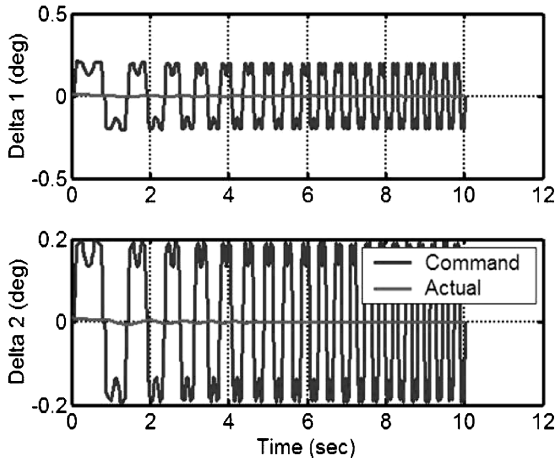


**Fig. 5 Control effector 1 and 2 positions, compensation off.**

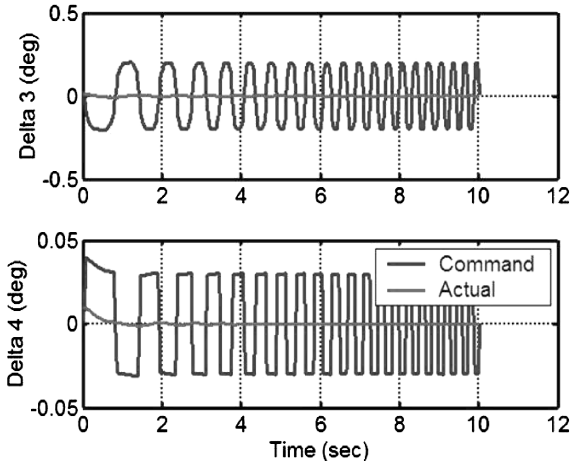


**Fig. 6 Control effector 3 and 4 positions, compensation off.**

$\boldsymbol{d}_{\text{des}} \cong \boldsymbol{B\delta}$. Notice that control effector 1 is upper position limited and control effector 4 is rate limited for the last few seconds of the simulation run. Thus, using the simple gain adjustment described in Eq. (20) results in the actual control deflections being equal to the commanded control deflections. It is apparent that adjusting the control effector command increments can help to mitigate adverse interactions between discrete-time implementations of control allocation algorithms and actuator dynamics.

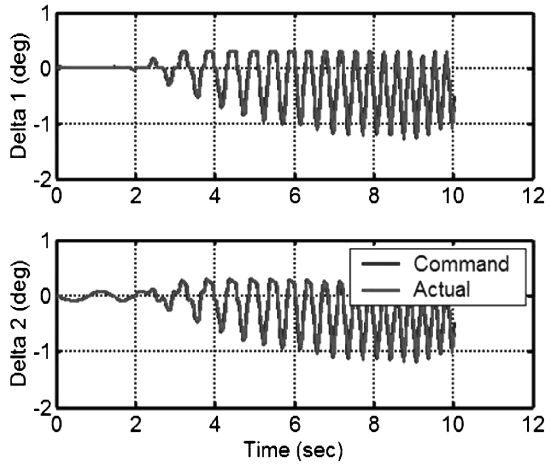


**Fig. 7 Control effector 1 and 2 positions, compensation on.**

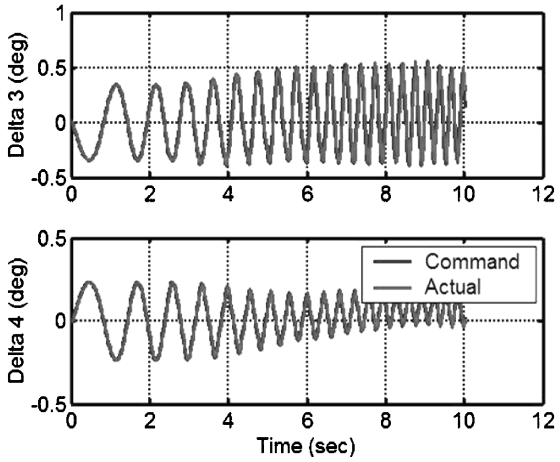


**Fig. 8 Control effector 3 and 4 positions, compensation on.**

## Conclusions

Interactions between constrained control allocation algorithms and the dynamics of actuators can result in degraded performance if not carefully implemented. The effective rate limit of a constrained control allocator operating in conjunction with a rate and position limited first-order actuator will always be less than the software rate limit enforced by the control allocator. Likewise, it was shown how the control effector commands, from a control allocation algorithm, are attenuated by actuator dynamics. One method, which can be used to extract maximum performance from such a system, is to modify the control effector commands. The gains used to modify the commands were computed for both first-order actuators and for second-order actuators. Simulation results show that significant improvements can be achieved by using this method to postprocess the commands from the control allocation algorithm. Benefits of this method are that the control allocation algorithm need not be modified, it requires minimal additional computations, and it is valid for saturated and unsaturated control effectors. Because the additional required computations are minimal, this method would be suitable for implementation in a real-time environment.

## References

[1]Buffington, J. M., "Modular Control Law Design for the Innovative Control Effectors (ICE) Tailless Fighter Aircraft Configuration 101-3," Technical Rept. AFRL-VA-WP-TR-1999, U.S. Air Force Research Lab., Wright–Patterson AFB, OH, 1999, pp. 93–94.

[2]Bodson, M., "Evaluation of Optimization Methods for Control Allocation," *Journal of Guidance, Control, and Dynamics*, Vol. 25, No. 4, 2002, pp. 703–711.

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

[4]Durham, W., "Constrained Control Allocation: Three Moment Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 4, 1993, pp. 717–725.
[5]Durham, W., "Attainable Moments for the Constrained Control Allocation Problem," *Journal of Guidance, Control, and Dynamics*, Vol. 17, No. 6, 1994, pp. 1371–1373.
[6]Page, A. B., and Steinberg, M. L., "A Closed-Loop Comparison of Control Allocation Methods," AIAA Paper 2000-4538, Aug. 2000.
[7]Doman, D. B., and Oppenheimer, M. W., "Improving Control Allocation Accuracy for Nonlinear Aircraft Dynamics," AIAA Paper 2002-4667, Aug. 2002.
[8]Bolender, M., and Doman, D. B., "Non-Linear Control Allocation Using Piecewise Linear Functions," AIAA Paper 2003-5357, Aug. 2003.
[9]Härkegård, O., "Dynamic Control Allocation Using Constrained Quadratic Programming," AIAA Paper 2002-4761, Aug. 2002.
[10]Venkataraman, R., Oppenheimer, M., and Doman, D., "A New Control Allocation Method That Accounts for Effector Dynamics," Inst. of Electrical and Electronics Engineers, Paper IEEEAC 1221, March 2004.
[11]Bolling, J. G., "Implementation of Constrained Control Allocation Techniques Using an Aerodynamic Model of an F-15 Aircraft," M.S. Thesis, Dept. of Aerospace Engineering, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, 1997.