JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 15, No. 2, March-April 1992

Adaptive Simulator Motion Software with Supervisory Control

M. A. Nahon*
Aercol Ltd., Toronto, Ontario, M3H 5T6 Canada
and
L. D. Reid† and J. Kirdeikis‡
University of Toronto, Toronto, Ontario, M3H 5T6 Canada

Concepts for flight simulator motion-drive algorithms range from the most basic to the relatively complex, with little to guide a choice between these, short of implementing them all and choosing the best. To avoid this lengthy process and to put into practice the experience gained in a previous large-scale evaluation exercise, a flexible motion algorithm has been implemented. It can be run as a simple classical algorithm with few free parameters and can be quickly adjusted to yield good motion performance. The more sophisticated adaptive features of the algorithm can then be brought in gradually to improve performance. Various forms of cost functions and adaptive features were formulated and subsequently evaluated on a synergistic 6 degrees-of-freedom motion-base simulator. It was found that a fourth-order cost function coupled with adaptive gain filters yielded the most favorable results. Finally, a supervisory code to ease motion adjustment and to provide a safe interactive interface with the designer is described. This facility permits reduced turnaround time during motion tuning and allows the pilot to experience different motions back-to-back for easier comparison. The supervisory control also allows automatic motion adjustment to different flight conditions.

Nomenclature§

a_{AA}	= body-axis components of the aircraft acceleration at the cockpit reference point
a_{SI}, S_I	= inertial components of the simulator reference point acceleration and position
a_c	= intermediate acceleration variable in all washout filters
f_{AA}	= body-axis components of the aircraft specific force at the cockpit reference point, $f_{AA} = a_{AA} - g_A$
f_p	= body-axis components of the specific force at the pilot's head location
f_1, f_2	= intermediate specific force variables in all washout filters
$G_{y1}, \ldots, G_{y4}$	= steepest descent slopes in the hybrid algorithm cost function
g, g_A, g_I	= gravitational acceleration expressed as a scalar and as components in the body and inertial frames, $g_I = [0\ 0\ g]^T$
J_y	= cost function for the roll/sway degrees of freedom
K_A^y, K_A^ϕ	= sway and roll input scaling factors
$K_{y1}, \ldots, K_{y3}$	= fixed coefficients in the hybrid algorithm
L_{IS}	= rotation matrix that transforms vector components from the simulator reference frame to the inertial frame
N	= order of the cost function
$P_{y1}, \ldots, P_{y4}$	= adaptive coefficients in the hybrid algorithm
$P_{y10}, \ldots, P_{y40}$	= reference values of the adaptive coefficients in the hybrid algorithm
p	= roll rate
p'	= adaptive coefficients in the coordinated adaptive algorithm
s	= Laplace operator
T_S	= transformation matrix from angular velocity to Euler angle rates
$W_{y0}, \ldots, W_{y9}$	= weights in the hybrid algorithm cost function
α	= generic variable
β_A	= aircraft Euler angles, $\beta_A = [\phi_A\ \theta_A\ \psi_A]^T$
β_S	= simulator Euler angles, $\beta_S = \beta_{SL} + \beta_{SH} = [\phi_S\ \theta_S\ \psi_S]^T$
β_{SL}	= component of β_S produced by the low-pass filter channel, $\beta_{SL} = [\phi_{SL}\ \theta_{SL}\ 0]^T$
β_{SH}	= component of β_S produced by the high-pass filter channel, $\beta_{SH} = [\phi_{SH}\ \theta_{SH}\ \psi_{SH}]^T$
ω_{AA}	= body-axis components of the aircraft angular velocity, $\omega_{AA} = [p_{AA}\ q_{AA}\ r_{AA}]^T$
ω_1	= scaled and limited aircraft angular velocity
$(*)^{x,y,z}$	= x, y, or z component of $(*)$

Introduction

A TYPICAL flight simulator installation is made up of the subsystems shown in Fig. 1. Of these, the ones that affect the quality of the flight simulator motion have been highlighted; the flight/turbulence model, the "washout" algorithm, and the soft-limiting system. The washout filters (whose name originates from the fact that one of their functions is to "wash out" the position of the simulator back to its neutral point) remain a primary source of poor fidelity in the motion cues, and even a small improvement in this area can yield a significant improvement in motion realism. Briefly stated, the purpose of the washout subsystem is to take the motions generated by the aircraft equations of motion—which include very large displacements—and filter them to provide simulator motion-base commands. These commands must provide the pilot with realistic motion cues, while remaining within the simulator's motion limits. Various techniques have been proposed to achieve this effect.

One of the earliest and simplest types is the classical washout in its many forms,[1-3] generally characterized by mostly linear elements assembled by trial and error. Its principal advantages are its simplicity and ease of adjustment.

Received July 31, 1990; presented as Paper 90-3133 at AIAA Flight Simulation Technologies Conference and Exhibit, Dayton, OH, Sept. 17-19, 1990; revision received Oct. 17, 1990; accepted for publication Dec. 17, 1990. Copyright © 1990 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Project Engineer, 4925 Dufferin St., Downsview. Student Member AIAA.

†Professor and Associate Director, Institute for Aerospace Studies, 4925 Dufferin St., Downsview.

‡Research Assistant, Institute for Aerospace Studies, 4925 Dufferin St., Downsview.

§Vectors are denoted by lower-case boldface symbols, and matrices are denoted by upper-case boldface symbols.

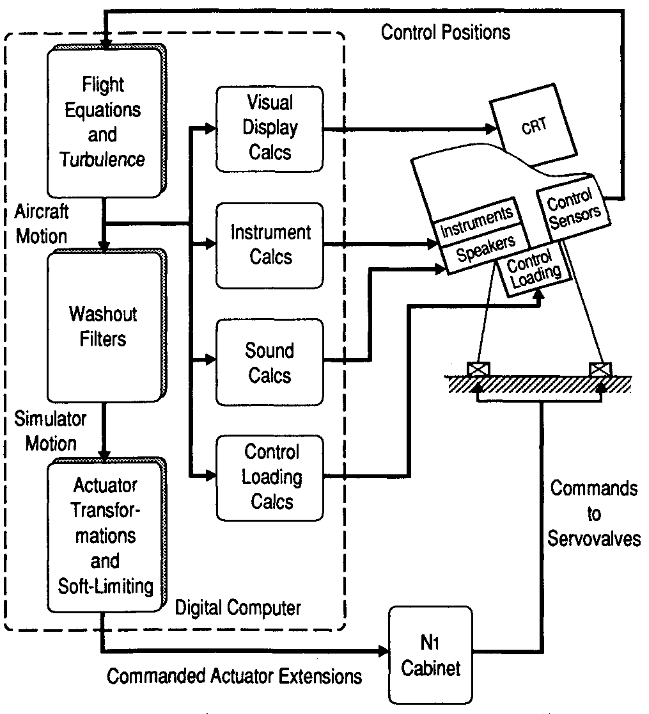

Fig. 1 Typical flight simulator installation.

Adaptive schemes were later proposed in which the filter gains are altered *in real time* to minimize a cost function using steepest descent techniques.[4-6] Since the cost function contains simulator excursions from the neutral point as well as motion errors, smaller inputs should be attenuated less than larger ones since they are less likely to result in large simulator displacements. Thus, false cueing should be reduced and better use made of the motion-base capabilities. More recently, optimal control theory has been applied to the problem[7,8] to generate an optimal linear filter that would minimize a specified quadratic cost function *off line.*

The work of Refs. 9-11 summarized in Refs. 12 and 13 presented a comparison of representative classical, adaptive, and optimal algorithms undertaken with the aim of choosing one of these for the University of Toronto Institute for Aerospace Studies (UTIAS) Flight Research Simulator. Results of that work showed that, on balance, the classical and adaptive algorithms were better suited to our application.[12] Further work has since been directed at combining the best features of the classical and adaptive schemes into a more comprehensive *hybrid* scheme. This approach was originally proposed in Ref. 14, although few details were given there of the implementation.

In the present work, nonlinear adaptive filter blocks have been incorporated in the classical layout. The choice of layout, as well as the location of the adaptive filter blocks, is justified step-by-step to show that this hybrid washout filter should combine the best features of the classical and adaptive algorithms with few of their disadvantages. Alternative cost functions were investigated in an attempt to enhance the adaptive features of the algorithm. These included second, fourth, and sixth order, as well as piecewise linear formulations. The adaptive filter equations were also extended to add adaptive frequency and damping features to the filters to see whether these would be more effective than the more traditional adaptive gain. The performance of these features was first evaluated off line to determine their resistance to instabilities, their effect on ease of adjustment, and their improvement of the motion cues. The most promising algorithms were then implemented and evaluated on the UTIAS Flight Research Simulator in an informal pilot evaluation.

All washout algorithms contain a number of free parameters that must be adjusted by trial and error to produce the desired motion. In the process of implementing this and previous algorithms, it was noted that the parameter values critically affect an algorithm's performance. Because so much time was spent adjusting these parameters, a facility to ease this *tuning* process was implemented. This supervisory software enabled the designer to modify the motion of the simulator in consultation with the evaluation pilot in a smooth interactive manner. This has significantly reduced turnaround time and enabled the pilot to experience different motions during simulator "flight" in quick back-to-back succession for easier comparison. A number of safety features were incorporated to ensure that changes in the motion parameters did not cause bumps or instabilities in the motion. Finally, since the washout filter parameters necessary for realistic motion differ according to the flight phase (e.g., they are different for ground handling and for flight), the supervisory software was extended to allow automatic changes of the filter parameters according to certain control flags. This feature could also be used to cater to differing pilot preferences in simulator motion.

Layout

In this section the similarities between the classical and adaptive algorithms will be highlighted in support of including these same features in the new hybrid algorithm. The principal differences between the two algorithms will then be discussed and justification will be given for choosing one approach or the other for the new algorithm. Figures 2 and 3 show the flowcharts of the classical and adaptive algorithms as they were implemented in Refs. 9-13. Their similarities are as follows:

1) The inputs to the washout filters are the aircraft specific forces f_{AA} and angular rates ω_{AA}. Specific forces are composed of accelerations and gravitational effects and are the variables normally sensed by an accelerometer. Although other motion variables could be used as inputs, the objective of the washout filter is inherently to reproduce f_{AA} and ω_{AA}, a task more easily accomplished by using these as inputs from the outset. Although some authors have suggested using sensed f_{AA} and ω_{AA} as the variables that should be reproduced,[6] our evaluation of this suggestion has shown that the benefits do not warrant the added CPU time.[9]

2) The motion inputs f_{AA} and ω_{AA} are scaled and limited, producing f_1 and ω_1, to reduce the amplitude of the motions to

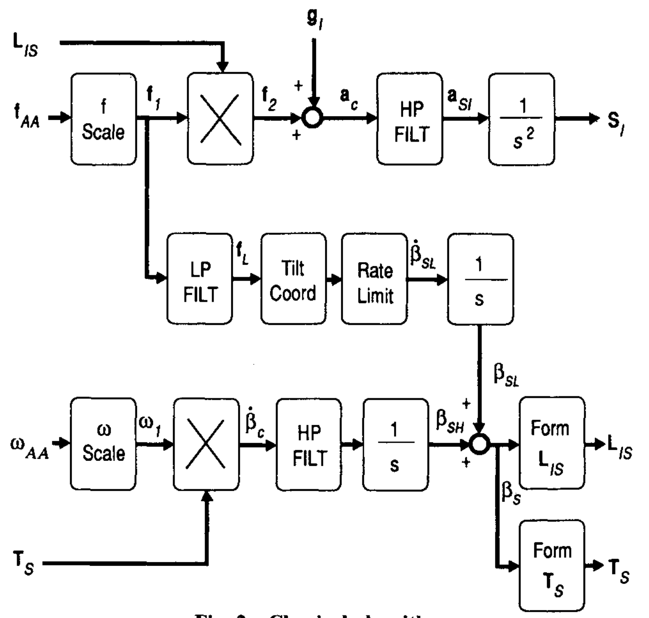

Fig. 2 Classical algorithm.

be simulated. Input scaling renders the task of filtering easier by subjecting all input motions to attenuation at all frequencies. Input limiting serves as a safety feature by ensuring that excessive commands do not enter the washout filter. In the present implementation, these were set at experimentally determined motion-base limits in acceleration and velocity. The application of these limits, which represent the simulator's capabilities in an inertial frame, to body-axis components of the input variables is an approximation. The error incurred due to this is not expected to be large since the simulator's angular motion is moderate for common synergistic motion bases.

3) The scaled and limited body-axis aircraft motions f_1 and ω_1 are premultiplied by L_{IS} and T_S to obtain f_2 and $\dot{\beta}_c$, respectively. This allows the subsequent filtering to be performed in an inertial frame. As discussed in Ref. 9, filtering can be performed in the body frame, the inertial frame, or some combination of the two. Inertial-frame filtering is advantageous because it ensures that all simulator motions will be washed out to zero (assuming filters of a sufficiently high order), although it introduces some small second-order coupling effects. Body-frame filtering does not introduce this coupling but does not ensure that the motion-base motions will be washed out. Test cases were run in Ref. 9 that showed that the cross-coupling effect was negligible compared to the problem of large offsets when the motion was not properly washed out. As a result, all filtering in the hybrid algorithm is performed in the inertial frame.

4) The gravitational acceleration g is added to the vertical component of f_2 to obtain a_c, i.e.,

$$a_c = [a_c^x \; a_c^y \; a_c^z]^T = f_2 + g_I = [f_2^x f_2^y f_2^z + g]^T$$

The remainder of the translational channel deals with acceleration rather than specific force, since the rotational contribution to the specific force is assumed to be handled by the crossfeed channel described in item 6 below. An alternative would be to rely on the vertical translational high-pass filter to remove this constant component. Additional care would then have to be taken to ensure that the filter was properly initialized to avoid startup transients and steady-state position offsets.

5) The transformed aircraft translational acceleration, represented by a_c, is passed through high-pass filters to obtain the simulator translational acceleration a_{SI}. Low-frequency inputs to the washout filters will tend to generate excessive simulator displacement, whereas high-frequency signals, such as those resulting from turbulence, will, in general, be producible. To separate the high- and low-frequency components, the translational accelerations are passed through high-pass filters. The filter output is then integrated twice to produce the simulator translational position S_I.

6) The previous filtering will only yield the high-frequency content of the aircraft translational motion. Low-frequency aircraft translational motion is simulated using "tilt coordination," or crossfeed, from aircraft translational motion to simulator rotational motion. This mechanism accounts for a great deal of the realism possible in a simulator and makes the motion simulation of large transport aircraft substantially easier than that of fighter aircraft, since the frequency content is lower for the former than the latter. Both the classical and adaptive algorithms include a form of crossfeed, although the detailed implementation differs (this will be discussed later in more detail). Furthermore, to prevent the pilot from noticing this "trick," tilt-rate limiting is used on the crossfeed channels. The tilt-rate limits used in all of the algorithms considered here were 3 deg/s in pitch and 2 deg/s in roll, consistent with the findings of a previous study.[11]

7) The outputs from the washout filter are S_I and β_S, the translational and angular positions of the motion base. These are processed through actuator transformation[9] and soft-limiting[10] algorithms to obtain commanded actuator lengths. Finally, the simulator Euler angles β_S are used to form the transformation matrices L_{IS} and T_S to be used in the next pass through the washout algorithm.

There are also some substantial differences between the classical and adaptive washout algorithms. These are now enumerated, and an argument is made as to which of the two approaches should be used in the hybrid algorithm.

1) The primary difference between the adaptive and classical algorithms is their filter gains. Whereas the classical filters have fixed gains (of 1.0 in the present case), the gain of the adaptive filters varies during "flight" in an attempt to reduce a cost function. The cost function is composed of motion "errors," which are defined as the difference between simulator and aircraft motion components and simulator dispacements and velocities. This feature gives the adaptive algorithm a certain amount of "intelligence" that allows it to reduce "false cueing" and to attenuate motion commands that would tend to drive the motion base into its limits. Adaptiveness was therefore included in the hybrid algorithm.

2) The location of the crossfeed relative to the L_{IS} transformation is different in the two algorithms. In the classical washout algorithm, the crossfeed signal is taken before the transformation, whereas in the adaptive algorithm, it is taken after. This has an important effect on *when* the crossfeed channel is active. In the classical algorithm, the crossfeed will be active only when a body-axis longitudinal or lateral specific force is present in the aircraft, whereas in the adaptive algorithm, the crossfeed is active over a much broader range of situations. For example, in a coordinated turn—a maneuver during which the classical crossfeed remains inactive—the aircraft body-axis vertical specific force will be fed to the lateral crossfeed channel via the L_{IS} transformation in the adaptive algorithm. In fact, this mechanism is responsible for canceling the low-frequency angular rate, since the adaptive algorithm does not have explicit filtering of roll and pitch. Because the adaptive crossfeed channel is more active than the classical crossfeed, its adjustment is more complex since it entails finding a compromise in its behavior over a wider range of maneuvers. The classical crossfeed is much easier to adjust since it performs a simpler task. For these reasons, the hybrid algorithm used the crossfeed layout found in the classical algorithm.

3) Roll and pitch motions are explicitly high-pass filtered in the classical algorithm, whereas the adaptive algorithm relies on the crossfeed channel to cancel low-frequency angular motion. The final behavior of both systems is to let uncoordinated aircraft rotational motions through unattenuated at all frequencies, while acting as a high-pass filter for coordinated aircraft rotational motions.[10] The question of whether or not to include explicit high-pass filtering on these channels is

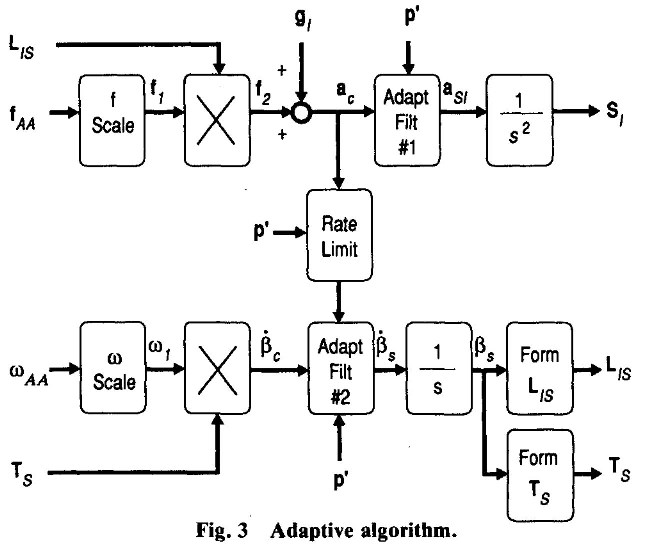

Fig. 3 Adaptive algorithm.

intrinsically tied to the location of the crossfeed. Once it was decided to take the crossfeed from upstream of the L_{IS} transformation, we were constrained to include explicit rotational filters in the hybrid algorithm.

4) The crossfeed channel in the adaptive algorithm feeds directly into the angular rate channel, whereas in the classical algorithm, it feeds through a low-pass filter into angular position. The difference in these approaches is again a result of the location of the source of the crossfeed signal relative to the L_{IS} transformation. Therefore, the form of crossfeed chosen was the same as that in the classical algorithm.

Although adaptive filters are to be included in the new hybrid algorithm, not *all* of the filter blocks need to be adaptive. The hybrid algorithm includes the following filters: 1) high-pass filters on all three translational channels (second-order in the longitudinal and lateral directions, and third-order in the vertical direction), 2) high-pass filters on all three rotational channels (first-order in pitch and roll and second-order in yaw), and 3) low-pass filters on the pitch and roll crossfeed channels (second-order). Of these, the first two sets were made adaptive, whereas the last was left fixed. The justification for this was that the adaptive nature of the crossfeed in the original adaptive algorithm led to a great deal of trouble in tuning while yielding few improvements in motion realism. Furthermore, since the output of the crossfeed channel is meant to "trick" the pilot, it is important to know at all times when and why this channel is active and to keep strict limits on this activity. This requires crossfeed of the simple fixed form found in the original classical washout.

From the preceding discussion, the layout shown in Fig. 4 was obtained for the new algorithm. It is identical to that of the classical filter except that certain filter blocks can be made adaptive. One of the advantages of this layout is that the degree of adaptiveness can be limited as desired, and it is possible to reduce it to a classical algorithm that can be fairly easily tuned to obtain good performance. From this point, adaptiveness can be introduced gradually, one filter at a time, to improve the performance.

Form of the Adaptive Filters

The original adaptive washout algorithm developed in Refs. 4 and 5 used a quadratic cost function to vary the gain of the filters in real time. The present work investigates the use of higher order cost functions as well as the use of adaptive break

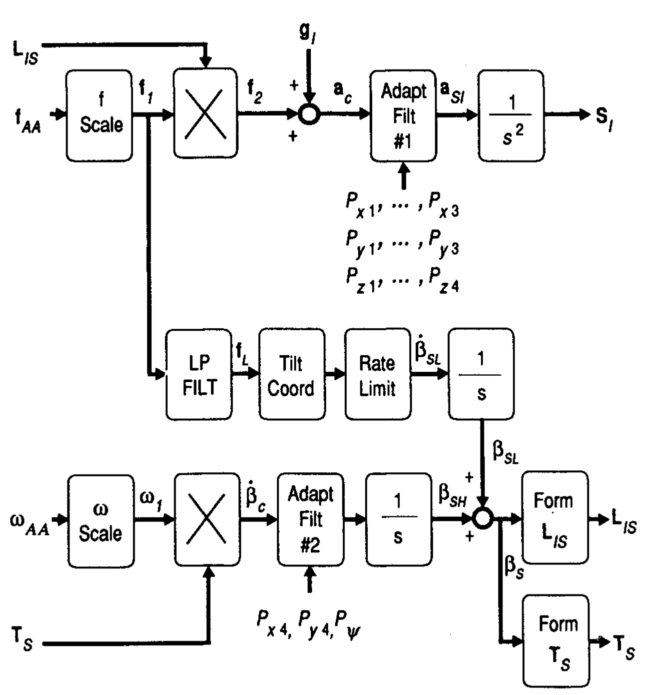

Fig. 4 Hybrid algorithm.

frequency and damping to provide a more versatile adaptive filter. The basic adaptive filter equations implemented in the present work will now be reviewed in more detail. The roll/sway equations are given as an example:

$$a_c^y = f_1^x \cos\theta_S \sin\psi_S + f_1^y(\sin\phi_S \sin\theta_S \sin\psi_S + \cos\phi_S \cos\psi_S)$$
$$+ f_1^z(\cos\phi_S \sin\theta_S \sin\psi_S - \sin\phi_S \cos\psi_S) \tag{1a}$$

$$\dot\phi_c = p_1 + (q_1 \sin\phi_S + r_1 \cos\phi_S) \tan\theta_S \tag{1b}$$

$$\ddot S_I^y = P_{y1}a_c^y - P_{y2}S_I^y - P_{y3}\dot S_I^y \tag{2a}$$

$$\ddot\phi_{SL} = -K_{y1}\frac{f_1^y}{g} - K_{y1}\phi_{SL} - K_{y2}\dot\phi_{SL} \tag{2b}$$

$$\dot\phi_{SH} = P_{y4}\dot\phi_c - K_{y3}\phi_{SH} \tag{2c}$$

$$\dot\phi_S = LIM(\dot\phi_{SL}) + \dot\phi_{SH} \tag{2d}$$

The cost function is given by

$$J_y = \frac{1}{2^{N-1}}[W_{y0}(a_c^y - \ddot S_I^y)^N + W_{y1}(\dot\phi_c - \dot\phi_S)^N + W_{y2}(\dot S_I^y)^N$$
$$+ W_{y3}(S_I^y)^N + W_{y4}(\dot\phi_S)^N + W_{y5}(\phi_S)^N + W_{y6}(P_{y1} - P_{y10})^N$$
$$+ W_{y7}(P_{y2} - P_{y20})^N + W_{y8}(P_{y3} - P_{y30})^N$$
$$+ W_{y9}(P_{y4} - P_{y40})^N] \tag{3}$$

The P_{yi} coefficients in Eq. (2) are varied in real time according to

$$\dot P_{yi} = -G_{yi}\frac{\partial J_y}{\partial P_{yi}} \qquad i = 1, \dots, 4 \tag{4}$$

Equation (2a) represents the second-order adaptive high-pass lateral filter, Eq. (2b) represents the fixed second-order low-pass crossfeed filter, whereas Eq. (2c) represents the first-order adaptive high-pass roll filter. The P_{yi} parameters are adaptive: P_{y1} is the gain, P_{y2} is the square of the break frequency, and P_{y3} is twice the product of the filter damping ratio and break frequency, all for the translational filter. The P_{y4} is the adaptive gain of the roll high-pass filter. The K_{yi} parameters are fixed: K_{y1} and K_{y2} are the square of the break frequency and twice the product of break frequency and damping ratio for the crossfeed filter, respectively, whereas K_{y3} represents the break frequency of the rotational high-pass filter. Adaptive break frequencies were not included in the rotational filters as it was felt sufficient to evaluate this feature on the translational channels. Had the performance of this feature been satisfactory, it could have been implemented in the rotational channels as well.

Equations (1), (2), and (3) can be substituted into Eq. (4) to obtain $\dot P_{yi}$ ($i = 1, \dots, 4$), as explicit functions of the input variables f_1 and ω_1. These relations can then be integrated in real time to obtain the values of the adaptive parameters P_{yi} ($i = 1, \dots, 4$). All integrations in the digital implementation of the washout algorithm were performed using the (first order) "ABC" method—a variation on the Euler method that uses updated values of variables as they become available.[9] This was done to avoid introducing lags in the filter response that are characteristic of the explicit Euler method. If computing power were a lesser constraint, a higher order implicit method would be preferred.

Varying the adaptive parameters in real time according to Eq. (4) will tend to minimize the cost function J_y given by Eq. (3). The first two terms in the cost function penalize motion error in y acceleration and in roll rate, the next four terms penalize the simulator motion, whereas the last four terms are included to return the adaptive parameters to their original reference state. As a safety measure to avoid instabilities in the adaptive parameters that develop when they vary too much, the values P_{yi} ($i = 1, \ldots, 4$) were passed through limiting blocks to ensure they remained within reasonable bounds. These bounds were established for each parameter as a function of its reference value (e.g., $0 \le P_{y1} \le P_{y10}$).

An important feature of the hybrid algorithm is that it becomes identical to the original classical algorithm when the steepest descent slopes G_{yi}($i = 1, \ldots, 4$) are set to zero. In this case, the parameters W_{yi}($i = 0, \ldots, 9$) no longer affect the simulator motion and the relation $P_{yi} = P_{yi0}$($i = 1, \ldots, 4$) holds identically. The designer then need only adjust K_{yi} ($i = 1, \ldots, 3$) and P_{yi0}($i = 1, \ldots, 4$) as best possible—a task known to be relatively easy.[13] If better performance is required, the steepest descent slopes can be gradually increased while choosing less restrictive values for K_{yi}($i = 1, \ldots, 3$) and adjusting the cost function weights as needed. It should also be noted that selectively setting certain steepest descent slopes to zero will alter the resulting type of adaptive filter. Referring, for example, to Eqs. (2-4), if G_{y2} and G_{y3} are set to zero while G_{y1} and G_{y4} are nonzero, the resulting filters will have adaptive gain and fixed break frequencies and damping. This feature allowed the evaluation of various combinations of adaptive and fixed parameters.

Different values of the order of the cost function, N in Eq. (3), were also evaluated: 2, 4, 6, and piecewise linear with varying slopes as shown in Fig. 5. In this figure, $J_y(\alpha)$ represents the variation in J_y due to a change in α, where α is any one of the terms in the summation in Eq. (3), with the corresponding weight set to unity [e.g., $J_y(\phi_S) = \phi_S^N/2^{N-1}$]. The piecewise-linear cost function was composed of 100 linear segments over the range $-2 < \alpha < 2$ (with constant slope outside this range) to produce the shape shown in Fig. 5. It was conjectured that cost functions that were flatter near the origin and rose more quickly as the independent variable approached a "critical value" (i.e., the motion limits) would show greater tendency to attenuate large motions more than small ones.

In all, 12 different combinations of adaptive parameters and order of the cost function were evaluated. Table 1 lists the characteristics of and name assigned to each of these combinations.

Testing the Algorithm

The evaluation of the various forms of the adaptive filters was performed in three phases:

1) an off-line assessment of the motion response to idealized single degree-of-freedom motion inputs consisting of a) a double pulse in longitudinal acceleration, b) a double pulse in vertical acceleration, c) an uncoordinated sinusoidal roll motion, and d) a ramp up to a steady value of yaw rate.

2) an off-line evaluation of the motion response to prerecorded aircraft maneuvers. These motion inputs were recorded while a pilot flew a simulated Boeing 747 through the two following maneuvers: a) a series of three alternating turn entries and b) a pushover/pull-up maneuver.

3) an informal piloted assessment during which an evaluation pilot flew the simulated Boeing 747 on the UTIAS Flight Research Simulator described in Ref. 11. The pilot was rated

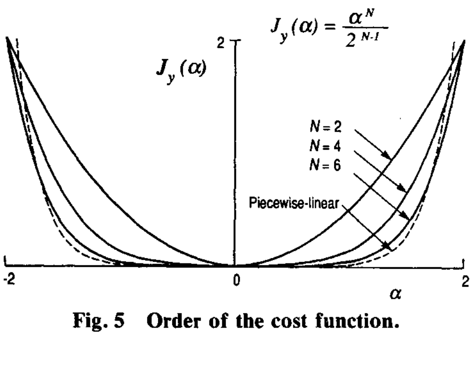

Fig. 5 Order of the cost function.

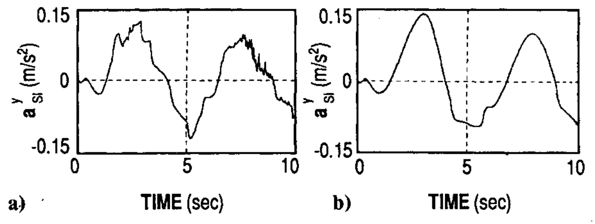

Fig. 6 Response to an uncoordinated roll input with a) the AW-PLGF filter and b) the AW4G filter.

on light aircraft and had considerable experience in the simulator. The maneuvers flown included a) coordinated turn entries, b) pushover/pull-up, c) sideslip, d) approach through turbulence, and e) touch and go landing.

The results of the first two phases of testing showed the following[15]:

1) The filters with adaptive gain only (suffix G) tended to adapt more rapidly than the filters that included adaptive break frequency and damping parameters (F and GF suffix).

2) The filters with adaptive break frequency and damping (suffix F) were not as effective at limiting maximum simulator displacements as the adaptive gain filters (suffix G). This dictated the use of more restrictive filter characteristics (larger values of P_{y20} and P_{y30}) to prevent the motion from exceeding its limits.

3) The filters with adaptive gain and break frequency and damping parameters (suffix GF) behaved midway between the filters with adaptive gain only (suffix G) and those with adaptive break frequencies and damping only (suffix F). However, the effect of adaptive gains seemed to predominate over that of the adaptive break frequencies and damping.

4) The second-order filters adapted more rapidly than the sixth-order filters.

5) The response of the piecewise-linear filters was very similar to that of the continuous quadratic formulation, but the motion produced was not as smooth. The occasional high-frequency oscillations in the motion output of the piecewise-linear filters was attributable to a toggling between two discrete values of the slope of the cost function. For example, Fig. 6a shows a high-frequency ringing in the lateral acceleration response of the AW-PLGF filter to the idealized roll input. By contrast, Fig. 6b shows the smooth response obtained with the AW4G filter.

As an example of the variations in the motion response obtained with the different filters, Fig. 7 shows the y component of the body-axis specific force at the pilot's head and the body-axis roll rate for some test configurations. In this figure, "AC" denotes the values existing in the simulated aircraft,

Table 1 Test configurations evaluated

	Order of the cost function			
Adaptive parameters	2	4	6	Piecewise linear
Adaptive gains on all filters $G_{y1} \neq 0$, $G_{y2} = G_{y3} = 0$, $G_{y4} \neq 0$	AW2G	AW4G	AW6G	AWPLG
Adaptive break frequency and damping on transl. channels; adaptive gain on rotational channels $G_{y1} = 0$, $G_{y2} \neq 0$, $G_{y3} \neq 0$, $G_{y4} \neq 0$	AW2F	AW4F	AW6F	AWPLF
Adaptive gain, break frequency, and damping on transl. channels; adaptive gain on rot. channels $G_{y1} \neq 0$, $G_{y2} \neq 0$, $G_{y3} \neq 0$, $G_{y4} \neq 0$	AW2GF	AW4GF	AW6GF	AWPLGF

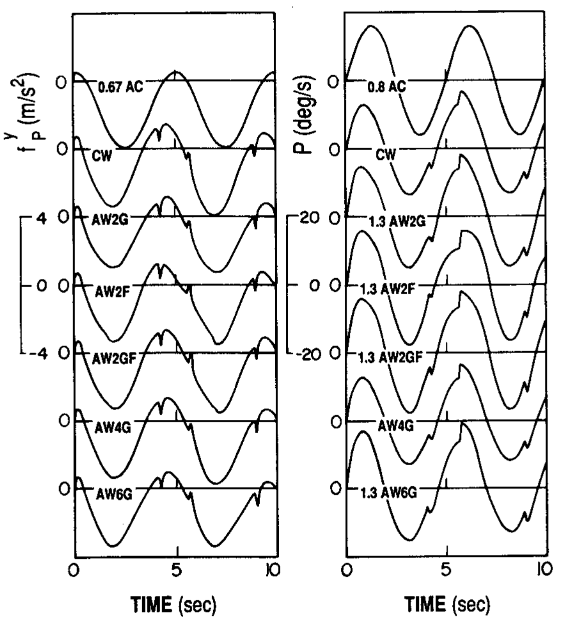

Fig. 7 Response of selected filters to an uncoordinated sinusoidal aircraft roll input.

"CW" denotes the simulator motion produced by the classical washout filter, and the remaining configurations—AW2G, AW2F, AW2GF, AW4G, and AW6G—are those listed in Table 1. It should be noted that some of the plots have been scaled to ease presentation; thus, "1.3 AW2G" represents the motion produced by the AW2G filter scaled by 1.3. It is emphasized that, within reasonable bounds, the *shape* of the motion response tends to correlate much better to pilot comments than does the amplitude of the response. As is often the case when attempting to compare transient responses, the differences in motion are not apparent at first glance. On closer examination, the plots of lateral specific force in Fig. 7 show that the response of the AW2F and AW2GF filters tends to lag that of the other filters. This correlated well with pilot complaints of phase lags whenever adaptive break frequencies and damping were used. The parameters of the roll/sway filter that produced these responses are given in Table 2. The adaptive filters used are initially the same as those in the classical washout. However, their adaptiveness allows the use of higher input scaling gains (1.0 vs 0.5). It is also noted that W_{y0}, the weight on the lateral acceleration error, was set to zero in all the adaptive filters. This was due to the fact that even small values of this parameter resulted in motion instabilities.

Following the first two phases of evaluation, seven of the candidate formulations were eliminated from further testing since they showed no benefits over the other formulations. These included all three formulations with piecewise-linear cost functions as they produced frequent oscillations and discontinuities due to changes in the slope of the adaptive parameters. The AW4F, AW4GF, AW6F, and AW6GF formulations were discarded since they showed little or no adaptive response. Thus, in the third phase, the AW2G, AW2F, AW2GF, AW4G, and AW6G formulations were compared to each other and to the standard classical algorithm shown in Fig. 2. The pilot comments were as follows:

1) CW: The classical washout provided good smooth motion with no noticeable phase lags. However, most motion cues were attenuated, and the actuator displacement limits were hit during the pull-up maneuver.

2) AW2G: This formulation provided strong motion cues that were well coordinated with control inputs and the visual display. The touchdown bump was more realistic as were the motion cues during approach. The washout (motion of the moving platform returning to its neutral position) was more noticeable than in the classical washout but was not objectionable. The actuator limits were not reached.

3) AW2F: The pilot strongly disliked the motion produced by this algorithm. It produced noticeable phase lags and frequent false cues that were disorienting. The washout motion was perceptible and jerky, and the actuator limits were reached during the pull-up maneuver.

4) AW2GF: The motion produced by this formulation was midway between AW2G and AW2F. Some phase lags were apparent during the turn entries, and some false cues were perceptible. The pilot compared the touchdown bump to landing on a sponge. The actuator limits were reached during the pull-up maneuver.

5) AW4G: This formulation elicited the most favorable comments. The turn entries produced a strong rolling sensation that was in phase with the visuals, the sideslip cues were realistic and well coordinated, and the touchdown bump was convincing. The actuator limits were not hit. The washout motion was perceptible but not jerky.

6) AW6G: This formulation produced strong onset cues, but the pilot found the washout motion too strong, particularly during turn entries. The actuator limits were reached during the pull-up maneuver, and some pitch oscillations were experienced.

To summarize the preceding comments, it was found that the formulations were rated from best to worst as AW4G, AW2G, CW, AW6G, AW2GF, and AW6F. The adaptive frequency and damping parameters did not do a good job of avoiding the actuator limits and introduced significant phase lags that the pilot found disorienting. The AW2G and AW4G formulations provided stronger cues than the classical algorithm while also doing a better job of avoiding the actuator limits. A further advantage of dropping the adaptive gain and break frequency would be a reduction in the number of parameters to be specified; referring back to Eqs. (2–4), P_{y2} and P_{y3} could be replaced by fixed parameters, G_{y2}, G_{y3}, P_{y20}, and P_{y30} could be dropped, and the number of differential equations to be solved would be reduced.

Supervisory Control

The adaptive filter equations include a number of parameters whose values must be chosen by the designer. For exam-

Table 2 Roll/sway parameter values for selected filters configuration

	Configuration					
Parameter	CW	AW2G	AW2F	AW2GF	AW4G	AW6G
K_A^y	0.5	1	1	1	1	1
K_A^Φ	0.5	1	1	1	1	1
P_{y1}	1	1	1	1	1	1
P_{y2}	16	16	16	16	16	16
P_{y3}	8	8	8	8	8	8
P_{y4}	1	1	1	1	1	1
K_{y1}	25	25	25	25	25	25
K_{y2}	10	10	10	10	10	10
K_{y3}	1	1	1	1	1	1
G_{y1}	—	5	0	1	5	750
G_{y2}	—	0	75	75	0	0
G_{y3}	—	0	50	60	0	0
G_{y4}	—	3	1	1	3	50
W_{y0}	—	0	0	0	0	0
W_{y1}	—	0.5	0.5	0.5	0.5	0.5
W_{y2}	—	3	25	30	20	150
W_{y3}	—	10	75	75	50	450
W_{y4}	—	4	4	4	4	4
W_{y5}	—	3	3	3	3	40
W_{y6}	—	0.5	0	4	3	2
W_{y7}	—	0	0.05	0.01	0	0
W_{y8}	—	0	0.1	0.01	0	0
W_{y9}	—	1	1	1	2	5

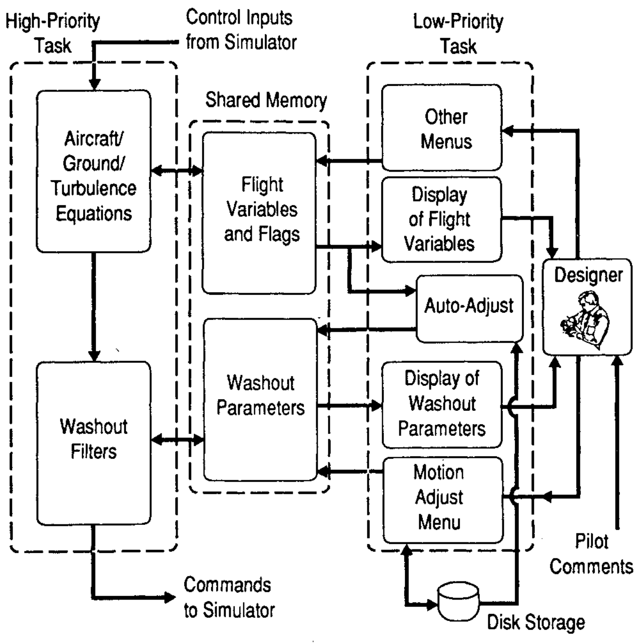

Fig. 8 Complete simulation environment.

ple, the roll/sway filter given by Eqs. (2–4) is adjusted by choosing $W_{yi}(i = 0, \ldots, 9)$ $G_{yi}(i = 1, \ldots, 4)$ $P_{yi0}(i = 1, \ldots, 4)$, and $K_{yi}(i = 1, \ldots, 3)$. The values chosen for these parameters will strongly affect the motion produced by the washout algorithm for a given aircraft motion. The adjustment of these parameters is commonly referred to as motion tuning or adjustment; it is an interactive exercise where a test pilot flies the simulator and comments on the motion that he experiences, and the designer modifies the motion parameters accordingly. Since the parameters in any washout algorithm greatly affect the simulator's motion, the tuning process becomes critical. In the present exercise, it was therefore important to facilitate this operation as much as possible. This section describes the software developed to implement and supervise the tuning process, thereby making motion adjustment an integral part of the overall simulation.

Structure of the Simulation Environment

The subsystems shown in Fig. 1 are only part of the total simulation environment, those that make up the real-time simulation. In fact, a more complete representation of the total environment is shown in Fig. 8 and includes the designer who acts as a high-level controller. In the present implementation, the designer (who may be located inside or outside the simulator cab) is seated at a console that allows him to control a low-priority interactive task in the host computer. A high-priority real-time task coexists in the computer that schedules and executes all of the software subsystems shown in Fig. 1. Both tasks have access to certain variables in shared memory. These are either information variables transmitted from the real-time task to the designer to inform him of the progress of the simulation (e.g., airspeed, altitude) or control variables that can be altered by the designer or instructor to control the evolution of the training session (e.g., engine failure flag).

Since the present application required fast modification of the washout filter parameters without interrupting the flight, these parameters were placed in shared memory and an interactive menu was provided to alter them. The menu has options to display or change any parameter, load a complete parameter set from file, or store a complete parameter set to file. Thus the pilot could voice a complaint, and the designer would identify which parameters needed alteration, perform the change, and query the pilot for any perceived changes. This efficient setup allowed complete motion tuning within three or four 2-h sessions (the sessions were kept short to avoid tiring the pilot).

Motion Adjustment

During the flight, the designer interacts with the low-priority task via a menu that allows him to select an action from a list that includes the following motion-related options: 1) change the complete set of washout filter parameters, 2) display/adjust individual washout filter parameters, and 3) store the complete set of washout filter parameters.

A complete change of washout filter entailed reading a new set of filter parameters from a file and replacing the old filter parameters with these. In the filter adjustment option, the designer chooses a specific parameter, its present value is displayed, and the designer is prompted for the new desired value. Once any of these changes have been made, the value of the appropriate parameter in shared memory is changed, and the change takes effect in the real-time simulation. Once the motion adjustment session is completed, all of the parameter values can be read from shared memory and stored in a file.

Since large instantaneous changes in certain washout filter parameters could result in bumps or instabilities in the motion, a rate-limiting algorithm was implemented to filter out these discontinuities as follows:

$$\dot{X} = (X^i_{\text{in}} - X^{i-1}_{\text{out}})/\delta t \tag{5a}$$

$$\dot{X}_{\text{out}} = LIM(\dot{X}) \tag{5b}$$

$$X^i_{\text{out}} = X^{i-1}_{\text{out}} + \delta t \cdot \dot{X}_{\text{out}} \tag{5c}$$

where X_{in} represents the input variable, X_{out} represents the rate-limited variable, and the i and $i - 1$ superscripts denote values at the present and previous time steps, respectively. This algorithm ensures that the output is identical to the input as long as the rate limit is not reached, and that even after rate limit *has* occurred, the output will eventually resume tracking the input. Preset limits were set for each variable, and the entry console emitted an auditory cue as long as any parameter was being rate limited to notify the designer that the change just made was not yet fully in effect.

A software emergency shutdown sequence was also provided as an additional safety feature. Hitting the return key in any submenu without entering a response would return control to the main menu. Repeating this action from the main menu would shut down the simulator motion without stopping the rest of the simulation. Thus, when a motion instability seemed incipient, the designer could halt all motion by simply hitting the return key twice. Of course, the designer also had the usual hardware panic button for more severe emergencies, although its use entailed a more complex restart procedure.

Extension to Automatic Motion Adjustment

One of the interesting results of the motion tuning was that the set of motion parameters that produced the best motion changed with the flight phase. A dramatic example of this is the substantial difference in the motion parameters needed for realistic motion in flight and during ground maneuvering. The software described earlier was therefore extended to provide automatic changes in parameters for different phases of flight according to (for example) the touchdown flag, the turbulence flag, etc. Thus, these changes were not made by the instructor through the console but rather by the appropriate flag changing in the real-time task, being communicated to the low-priority task, which would then summon automatic versions of the washout parameter change routine. The rate-limiting algorithm mentioned in the previous section ensured that all transitions occurred smoothly and went unnoticed by the pilot.

A further application of automatic motion adjustment pertained to the interpilot variability in motion assessment encountered during the piloted evaluations.[11] Different pilots have distinctly different perceptions of what type of motion is most realistic. These subjective preferences could not be satisfied with a single parameter set. Rather, each pilot could be assigned a file containing his preferred parameter set that would be invoked at the start of a simulator session. Of course, further investigation would be needed to determine whether the transfer of training is improved or degraded by catering to subjective pilot preferences.

Conclusions

A flexible motion system has been developed for use on moving-base simulators. It consists of a washout algorithm that combines the best features of the common existing algorithms, controlled by supervisory software. The new algorithm was derived as a hybrid of existing classical and adaptive algorithms. This allowed a filter that could be easily adjusted to obtain good performance, while adaptive features could be added as required. Various forms of the cost function were evaluated on a 6 degrees-of-freedom synergistic motion-base simulator of the type commonly used to train airline pilots. In a small-scale piloted evaluation, it was found that the fourth-order cost function with adaptive-gain filters received the most favorable response. Adaptive break frequency and damping were also investigated, but they were found to produce objectionable phase lags in the motion and did not do a good job of avoiding the actuator limits. A supervisory software system was constructed to allow fast interactive testing and adjustment of motion algorithms. It was also extended to include automatic changes according to flight phase and conditions or according to pilot preference.

Acknowledgments

The work reported here was funded jointly by Aercol Ltd. and the Natural Sciences and Engineering Research Council. The development of the simulator used in this study was funded by the Canadian Natural Sciences and Engineering Research Council and the Ontario Government.

References

[1]Schmidt, S. F., and Conrad, B., "Motion Drive Signals for Piloted Flight Simulators," NASA CR-1601, May 1970.

[2]Baarspul, M., "The Generation of Motion Cues on a Six-Degrees-of-Freedom Motion System," Delft Univ. of Technology, Dept. of Aerospace Engineering, Rept. LR-248, Delft, The Netherlands, June 1977.

[3]Parrish, R. V., Dieudonne, J. E., and Martin, D. J., Jr., "Motion Software for a Synergistic Six-Degree-of-Freedom Motion Base," NASA TN D-7350, Dec. 1973.

[4]Parrish, R. V., Dieudonne, J. E., Bowles, R. L., and Martin, D. J., Jr., "Coordinated Adaptive Washout for Motion Simulators," *Journal of Aircraft,* Vol. 12, No. 1, 1975, pp. 44–50.

[5]Parrish, R. V., and Martin, D. J., Jr., "Comparison of a Linear and Nonlinear Washout for Motion Simulators Utilizing Objective and Subjective Data from CTOL Transport Landing Approaches," NASA TN D-8157, June 1976.

[6]Ariel, D., and Sivan, R., "False Cue Reduction in Moving Flight Simulators," *IEEE Transactions on Systems, Man and Cybernetics,* Vol. SMC-14, No. 4, 1984, pp. 665–671.

[7]Sivan, R., Ish-Shalom, J., and Huang, J.-K., "An Optimal Control Approach to the Design of Moving Flight Simulators," *IEEE Transactions on Systems, Man and Cybernetics,* Vol. SMC-12, No. 6, 1982, pp. 818–827.

[8]Ish-Shalom, J., "The Design of Optimal Control Motion for Flight Simulators," Ph.D. Thesis, M.I.T. Centre for Space Research, Cambridge, MA, Dec. 1982.

[9]Reid, L. D., and Nahon, M. A., "Flight Simulator Motion-Base Drive Algorithms: Part 1—Developing and Testing the Equations," Univ. of Toronto, UTIAS Rept. 296, Toronto, Canada, Dec. 1985.

[10]Reid, L. D., and Nahon, M. A., "Flight Simulator Motion-Base Drive Algorithms: Part 2—Selecting the System Parameters," Univ. of Toronto, UTIAS Rept. 307, Toronto, Canada, May 1986.

[11]Reid, L. D., and Nahon, M. A., "Flight Simulator Motion-Base Drive Algorithms: Part 3—Pilot Evaluations," Univ. of Toronto, UTIAS Rept. 319, Toronto, Canada, Dec. 1986.

[12]Reid, L. D., and Nahon, M. A., "The Response of Airline Pilots to Variations in Flight Simulator Motion Algorithms," *Journal of Aircraft,* Vol. 25, No. 7, 1988, pp. 639–646.

[13]Nahon, M. A., and Reid, L. D., "Simulator Motion Drive Algorithms—A Designer's Perspective," *Journal of Guidance, Control, and Dynamics,* Vol. 13, No. 2, 1990, pp. 356–362.

[14]Dorbolo, G., and Van Sliedregt, J. M., "Improvements in Motion Drive Algorithms," *Proceedings of the 1987 Summer Computer Simulation Conference,* Society for Computer Simulation, San Diego, CA, July 1987, pp. 721–723.

[15]Kirdeikis, J., "Evaluation of Nonlinear Motion-Drive Algorithms for Flight Simulators," Univ. of Toronto, UTIAS TN 272, Toronto, Canada, June 1989.