A Nonlinear, Human-Centered Approach to Motion Cueing with a Neurocomputing Solver

Robert J. Telban* and Frank M. Cardullo†
State University of New York
Binghamton, New York

Jacob A. Houck‡
NASA Langley Research Center
Hampton, Virginia

Abstract

This paper discusses the continuation of research into the development of new motion cueing algorithms first reported in 1999. In this earlier work, two viable approaches to motion cueing were identified: the coordinated adaptive washout algorithm or "adaptive algorithm", and the "optimal algorithm". In this study, a novel approach to motion cueing is discussed that would combine features of both algorithms.

The new algorithm is formulated as a linear optimal control problem, incorporating improved vestibular models and an integrated visual-vestibular motion perception model previously reported. A control law is generated from the motion platform states, resulting in a set of nonlinear cueing filters. The time-varying control law requires the matrix Riccati equation to be solved in real time. Therefore, in order to meet the real time requirement, a neurocomputing approach is used to solve this computationally challenging problem.

Single degree-of-freedom responses for the nonlinear algorithm were generated and compared to the adaptive and optimal algorithms. Results for the heave mode show the nonlinear algorithm producing a motion cue with a time-varying washout, sustaining small cues for a longer duration and washing out larger cues more quickly. The addition of the optokinetic influence from the integrated perception model was shown to improve the response to a surge input, producing a specific force response with no steady-state washout. Improved cues are also observed for responses to a sway input. Yaw mode responses reveal that the nonlinear algorithm improves the motion cues by reducing the magnitude of negative cues.

The effectiveness of the nonlinear algorithm as compared to the adaptive and linear optimal algorithms will be evaluated on a motion platform, the NASA Langley Research Center Visual Motion Simulator (VMS), and ultimately the Cockpit Motion Facility (CMF) with a series of pilot controlled maneuvers. A proposed experimental procedure is discussed. The results of this evaluation will be used to assess motion cueing performance.

Introduction

It was reported in 1999[1] that two viable approaches to motion cueing were identified. The first technique is a modification of the coordinated washout algorithm developed at NASA by Parrish, et al.[2], hereafter referred to as the "adaptive algorithm". This algorithm uses both first-and second-order linear washout filters in conjunction with an optimization method that adjusts the filter gains in real time by minimizing the error between the simulated aircraft and the motion platform responses. This methodology effectively produces a set of nonlinear washout filters.

The second method is the "optimal algorithm" based on that first developed by Sivan, et al.[3], and later implemented by Reid and Nahon.[4] This algorithm uses higher-order filters that are determined, prior to real time application, using optimal control methods. This method incorporates a model of the human vestibular system, constraining the sensation error between the simulated aircraft and motion platform dynamics. The authors[1] initially formulated a version of this approach that resulted in improved motion cues, but was less capable of tracking changes to aircraft inputs as compared to the former approach. Therefore, a new algorithm is desired that combines features of both the optimal and adaptive algorithms.

The proposed algorithm is formulated as a linear optimal control problem similar to the approach previously reported[1], but can also be solved in real time. Furthermore, it incorporates models of the human vestibular sensation system, with improved semicircular canals and otoliths models, along with an integrated visual-vestibular perception model. A nonlinear control

* PhD. Candidate, Department of Mechanical Engineering, Student Member AIAA
† Associate Professor, Department of Mechanical Engineering, Associate Fellow AIAA
‡ Systems Development Branch, Associate Fellow AIAA

law is implemented to generate a scalar coefficient α that is a function of the motion platform states. For large platform motions, the coefficient α is large, resulting in faster control action. The matrix Riccati equation is then solved in real time as a function of α, resulting in the feedback matrix needed to compute the desired motion cues. In order to meet the real time requirement, a neurocomputing approach has been chosen to solve this computationally challenging problem. The structure of the problem is illustrated in Figure 1.

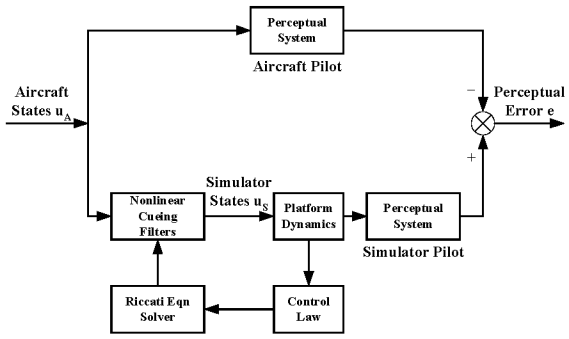

Figure 1. Problem Structure of Nonlinear Cueing Algorithm.

Algorithm Development

The algorithm is implemented in four modes: two single-degree-of-freedom modes (yaw and heave), and two two-degree-of-freedom modes (pitch/surge and roll/sway). The algorithm development for the longitudinal mode is given below. The control input $\mathbf{u}$ is formulated as

$$\mathbf{u} = \begin{bmatrix} \dot{\theta} \\ a_x \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}, \tag{1}$$

where $\dot{\theta}$ is angular velocity, and a_x is the translational acceleration, with each term respectively set equal to u_1 and u_2.

The rotational perception q_{PE} is then related to the input u by

$$\begin{aligned} \dot{\mathbf{x}}_{1\sim3} &= \mathbf{A}_{SCC}\mathbf{x}_{1\sim3} + \mathbf{B}_{SCC}\mathbf{u} \\ q_{PE} &= \mathbf{C}_{SCC}\mathbf{x}_{1\sim3} + \mathbf{D}_{SCC}\mathbf{u}, \end{aligned} \tag{2}$$

where $\mathbf{x}_{1\sim3}$ are the rotational perception states, and $\mathbf{A}_{SCC}$, $\mathbf{B}_{SCC}$, $\mathbf{C}_{SCC}$, and $\mathbf{D}_{SCC}$ represent in state space form the semicircular canals model along with an additional state that represents the optokinetic influence from the interaction of visual and vestibular stimuli as presented in the integrated motion perception model.[5]

The translational perceived velocity v_{xPE} is then related to the input $\mathbf{u}$ by

$$\begin{aligned} \dot{\mathbf{x}}_{4\sim9} &= \mathbf{A}_{OTO}\mathbf{x}_{4\sim9} + \mathbf{B}_{OTO}\mathbf{u} \\ v_{xPE} &= \mathbf{C}_{OTO}\mathbf{x}_{4\sim9} + \mathbf{D}_{OTO}\mathbf{u}, \end{aligned} \tag{3}$$

where $\mathbf{x}_{4\sim9}$ are the translational perception states, and $\mathbf{A}_{OTO}$, $\mathbf{B}_{OTO}$, $\mathbf{C}_{OTO}$, and $\mathbf{D}_{OTO}$ includes a proposed otolith model,[6] along with an additional state representing the optokinetic influence.

The representations in Eqs. (2) and (3) can be combined to form a single representation for the human perception model:

$$\begin{aligned} \dot{\mathbf{x}}_{1\sim9} &= \mathbf{A}_V\mathbf{x}_{1\sim9} + \mathbf{B}_V\mathbf{u} \\ \mathbf{y}_{PE} &= \mathbf{C}_V\mathbf{x}_{1\sim9} + \mathbf{D}_V\mathbf{u}, \end{aligned} \tag{4}$$

where $\mathbf{x}_{1\sim9}$ and $\mathbf{y}_{PE}$ are, respectively, the combined states and perceived responses, and $\mathbf{A}_V$, $\mathbf{B}_V$, $\mathbf{C}_V$, and $\mathbf{D}_V$ represent the perceptual models as one set of state equations:

$$\mathbf{A}_V = \begin{bmatrix} \mathbf{A}_{SCC} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_{OTO} \end{bmatrix}, \quad \mathbf{B}_V = \begin{bmatrix} \mathbf{B}_{SCC} \\ \mathbf{B}_{OTO} \end{bmatrix},$$
$$\mathbf{C}_V = \begin{bmatrix} \mathbf{C}_{SCC} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_{OTO} \end{bmatrix}, \quad \mathbf{D}_V = \begin{bmatrix} \mathbf{D}_{SCC} \\ \mathbf{D}_{OTO} \end{bmatrix}.$$

It is assumed that the same perceptual model can be applied to both the pilot in the aircraft and the pilot in the simulator as shown in Figure 1. We define the pilot perceptual error $\mathbf{e}$, resulting in

$$\begin{aligned} \dot{\mathbf{x}}_e &= \mathbf{A}_V\mathbf{x}_e + \mathbf{B}_V\mathbf{u}_S - \mathbf{B}_V\mathbf{u}_A \\ \mathbf{e} &= \mathbf{C}_V\mathbf{x}_e + \mathbf{D}_V\mathbf{u}_S - \mathbf{D}_V\mathbf{u}_A, \end{aligned} \tag{5}$$

where $\mathbf{u}_S$ and $\mathbf{u}_A$ represent the simulator and aircraft inputs as given in Eqn. (1).

It is also necessary for the control algorithm to explicitly access motion states such as linear velocity and displacement of the motion platform that appear in the cost function. For this purpose, additional terms are included in the state equations:

$$\dot{\mathbf{x}}_d = \mathbf{A}_d\mathbf{x}_d + \mathbf{B}_d\mathbf{u}_S, \tag{6}$$

where $\mathbf{x}_d$ represents the additional motion platform states:

$$\mathbf{x}_d = \left[\iiint a_x dt^3 \quad \iint a_x dt^2 \quad \int a_x dt \quad \theta\right]^T,$$

and are related to the simulator input $\mathbf{u}_S$ by

$$\mathbf{A}_d = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B}_d = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

The aircraft input $\mathbf{u}_A$ consists of filtered white noise, and can be expressed as

$$\begin{aligned} \dot{\mathbf{x}}_n &= \mathbf{A}_n\mathbf{x}_n + \mathbf{B}_n\mathbf{w} \\ \mathbf{u}_A &= \mathbf{x}_n, \end{aligned} \tag{7}$$

where $\mathbf{x}_n$ are the filtered white noise states, $\mathbf{w}$ represents white noise, with $\mathbf{A}_n$ and $\mathbf{B}_n$ given as

$$\mathbf{A}_n = \begin{bmatrix} -\omega_1 & 0 \\ 0 & -\omega_2 \end{bmatrix}, \quad \mathbf{B}_n = \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix},$$

where ω_1 and ω_2 are the first-order filter break frequencies for each degree-of-freedom.

The state equations given in Eqns. (5), (6), and (7) can be combined to form the desired system equation

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}_S + \mathbf{H}\mathbf{w} \\ \mathbf{y} &= \begin{bmatrix} \mathbf{e} & \mathbf{x}_d \end{bmatrix}^T = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}_S, \end{aligned} \tag{8}$$

where y is the desired output, and $\mathbf{x} = \begin{bmatrix} \mathbf{x}_e & \mathbf{x}_d & \mathbf{x}_n \end{bmatrix}^T$ represents the combined states. The combined system matrices **A**, **B**, **C**, **D**, and **H** are then given by

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_V & \mathbf{0} & -\mathbf{B}_V \\ \mathbf{0} & \mathbf{A}_d & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{A}_n \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_V \\ \mathbf{B}_d \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{H} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{B}_n \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_V & \mathbf{0} & -\mathbf{D}_V \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} \mathbf{D}_V \\ \mathbf{0} \end{bmatrix}.$$

A cost function J is then defined as

$$J = E\left\{\int_{t_0}^{t_1} \left(\mathbf{e}^T\mathbf{Q}\mathbf{e} + \mathbf{x}_d^T\mathbf{R}_d\mathbf{x}_d + \mathbf{u}_S^T\mathbf{R}\mathbf{u}_S\right)dt\right\}, \tag{9}$$

where $E\{\ \}$ is the mathematical mean of statistical variable, $\mathbf{Q}$ and $\mathbf{R}_d$ are positive semi-definite matrices, and $\mathbf{R}$ is a positive definite matrix. Eqn. (9) implies that three variables are to be constrained in the cost function: the sensation error $\mathbf{e}$ along with the additional terms $\mathbf{x}_d$ and $\mathbf{u}_S$ which together define the linear and angular motion of the platform. The cost function constrains both the sensation error and the platform motion.

The system equation and cost function can be transformed to the standard optimal control form as shown in Kwakernaak and Sivan[7] and noted in Reid and Nahon[4] by the following equations:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}'\mathbf{x} + \mathbf{B}\mathbf{u}' + \mathbf{H}\mathbf{w} \\ J' &= E\left\{\int_{t_0}^{t_1} \left(\mathbf{x}^T\mathbf{R}_1'\mathbf{x} + \mathbf{u}'^T\mathbf{R}_2\mathbf{u}'\right)dt\right\}, \end{aligned} \tag{10}$$

where

$$\begin{aligned} &\mathbf{A}' = \mathbf{A} - \mathbf{B}\mathbf{R}_2^{-1}\mathbf{R}_{12}^T, \ \mathbf{u}' = \mathbf{u}_S + \mathbf{R}_2^{-1}\mathbf{R}_{12}^T\mathbf{x}, \\ &\mathbf{R}_1' = \mathbf{R}_1 - \mathbf{R}_{12}\mathbf{R}_2^{-1}\mathbf{R}_{12}^T, \ \mathbf{R}_1 = \mathbf{C}^T\mathbf{G}\mathbf{C}, \\ &\mathbf{R}_{12} = \mathbf{C}^T\mathbf{G}\mathbf{D}, \ \mathbf{R}_2 = \mathbf{R} + \mathbf{D}^T\mathbf{G}\mathbf{D}, \ \mathbf{G} = diag\left[\mathbf{Q}, \mathbf{R}_d\right]. \end{aligned}$$

The cost function in Eq. (10) is augmented with an additional term $e^{2\alpha t}$ proposed by Anderson and Moore:[8]

$$J' = E\left\{\int_{t_0}^{t_1} e^{2\alpha t}\left(\mathbf{x}^T\mathbf{R}_1'\mathbf{x} + \mathbf{u}'^T\mathbf{R}_2\mathbf{u}'\right)dt\right\}, \tag{11}$$

where $\mathbf{R}_1'$ is positive definite, $\mathbf{R}_2$ is positive semi-definite, and the scalar coefficient α represents a minimum degree of stability in the closed-loop system where $\alpha > 0$.

Anderson and Moore[8] show that the system equation and cost function can be transformed to eliminate the exponential term, resulting in

$$\begin{aligned}\dot{\tilde{\mathbf{x}}} &= \left(\mathbf{A}' + \alpha\mathbf{I}\right)\tilde{\mathbf{x}} + \mathbf{B}\tilde{\mathbf{u}} + \mathbf{H}'\mathbf{w} \\ \tilde{J}' &= E\left\{\int_{t_0}^{t_f}\left(\tilde{\mathbf{x}}^{\mathrm{T}}\mathbf{R}_1'\tilde{\mathbf{x}} + \tilde{\mathbf{u}}^{\mathrm{T}}\mathbf{R}_2\tilde{\mathbf{u}}\right)dt\right\},\end{aligned} \tag{12}$$

where $\tilde{\mathbf{x}} = e^{\alpha t}\mathbf{x}$ and $\tilde{\mathbf{u}} = e^{\alpha t}\mathbf{u}'$.

We now wish to compute the simulator control input $\mathbf{u}_S$ that minimizes the cost function given in Eqn. (12). Anderson and Moore[8] note that for this problem, $\mathbf{A}' + \alpha\mathbf{I}$ is positive definite, $\left(\mathbf{A}' + \alpha\mathbf{I}, \mathbf{B}\right)$ is controllable and $\left(\mathbf{A}' + \alpha\mathbf{I}, \mathbf{R}_1'\right)$ is observable. Under these conditions, the cost function is minimized when

$$\mathbf{u}_S = -\mathbf{K}(\alpha)\mathbf{x}, \tag{13}$$

where $\mathbf{K}(\alpha) = \mathbf{R}_2^{-1}\left(\mathbf{B}^{\mathrm{T}}\mathbf{P}(\alpha) + \mathbf{R}_{12}\right)$, and $\mathbf{P}(\alpha)$ is the solution of the algebraic Riccati equation

$$\begin{aligned}&\left(\mathbf{A}' + \alpha\mathbf{I}\right)\mathbf{P}(\alpha) + \mathbf{P}(\alpha)\left(\mathbf{A}' + \alpha\mathbf{I}\right) \\ &\quad - \mathbf{P}(\alpha)\mathbf{B}\mathbf{R}_2^{-1}\mathbf{B}^{\mathrm{T}}\mathbf{P}(\alpha) + \mathbf{R}_1' = \mathbf{0}\end{aligned} \tag{14}$$

The feedback matrix $\mathbf{K}(\alpha)$ is partitioned corresponding to the partition of $\mathbf{x}$ in Eq. (8):

$$\mathbf{u}_S = -\left[\mathbf{K}_1(\alpha) \quad \mathbf{K}_2(\alpha)\right]\begin{bmatrix}\mathbf{x}_e \\ \mathbf{x}_d\end{bmatrix} - \mathbf{K}_3(\alpha)\mathbf{u}_A. \tag{15}$$

Noting that $\mathbf{x}_n = \mathbf{u}_A$, remove the states corresponding to the $\mathbf{x}_n$ partition from Eqn. (13):

$$\begin{bmatrix}\dot{\mathbf{x}}_e \\ \dot{\mathbf{x}}_d\end{bmatrix} = \begin{bmatrix}\mathbf{A}_V & \mathbf{0} & -\mathbf{B}_V \\ \mathbf{0} & \mathbf{A}_d & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{x}_e \\ \mathbf{x}_d \\ \mathbf{u}_A\end{bmatrix} + \begin{bmatrix}\mathbf{B}_V \\ \mathbf{B}_d\end{bmatrix}\mathbf{u}_S, \tag{16}$$

and substituting Eq. (15) into Eq. (16) results in

$$\begin{aligned}\begin{bmatrix}\dot{\mathbf{x}}_e \\ \dot{\mathbf{x}}_d\end{bmatrix} &= \begin{bmatrix}\mathbf{A}_V - \mathbf{B}_V\mathbf{K}_1(\alpha) & -\mathbf{B}_V\mathbf{K}_2(\alpha) \\ -\mathbf{B}_d\mathbf{K}_1(\alpha) & \mathbf{A}_d - \mathbf{B}_d\mathbf{K}_2(\alpha)\end{bmatrix}\begin{bmatrix}\mathbf{x}_e \\ \mathbf{x}_d\end{bmatrix} \\ &\quad + \begin{bmatrix}-\mathbf{B}_V\left(\mathbf{I} + \mathbf{K}_3(\alpha)\right) \\ -\mathbf{B}_d\mathbf{K}_3(\alpha)\end{bmatrix}\mathbf{u}_A.\end{aligned} \tag{17}$$

A nonlinear control law is chosen to make α dependent upon the system states:

$$\alpha = \mathbf{x}_d^{\mathrm{T}}\mathbf{Q}_2\mathbf{x}_d, \tag{18}$$

where $\mathbf{Q}_2$ is a weighting matrix that is at least positive semi-definite. As the system states increase in magnitude, i.e. with large platform motions, then α increases, resulting in faster control action and increased system stability. For small responses there will be limited control action. The feedback matrix $\mathbf{K}(\alpha)$ is then determined by solving the Riccati equation of Eq. (14) in real time as a function of α.

Real Time Solution of the Riccati Equation

Solving the nonlinear Riccati equation in Eqn. (14) is a computational challenge in real time as a new solution is required at each time step. Since the solution to the preceding time step is available, it is advantageous to use this as an initial solution when computing the solution for the current time step, thus reducing the computational burden. The initial Riccati equation solution to the linear optimal algorithm that is computed off-line in MATLAB is available and can be used as the initial solution for the first time step. To this end we desire a technique that assumes the initial solution is "close" to the computational solution at a given time step.

Two types of techniques were investigated for implementation with the nonlinear algorithm. Blackburn[9] developed a method of solution by using a Newton-Raphson iteration. With this technique, computation of the Jacobian matrix as a Kronecker product is required along with matrix inversion, which can result in a singular solution for an ill-conditioned system. Neurocomputing approaches suggested by Wang and Wu[10] and Ham and Collins[11] eliminate these operations, taking advantage of the matrix structure associated with the algorithm, and thus reducing the computational burden. For these reasons, the neurocomputing approaches were further evaluated.

The neurocomputing approach proposed by Ham and Collins[11] uses a structured neural network for obtaining the Riccati equation computational solution $\mathbf{P}$ as shown in Figure 2.

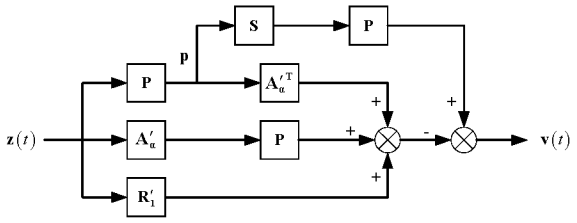

Figure 2. Structured Neural Network for Solving the Riccati Equation.

The error signal **v**(t) in Figure 2 is given as

$$\mathbf{v}(t) = \left[\mathbf{P}(t)\mathbf{S}\mathbf{P}(t) - \mathbf{A}_{\alpha}'^{\mathrm{T}}\mathbf{P}(t) - \mathbf{P}(t)\mathbf{A}_{\alpha}' - \mathbf{R}_1'\right]\mathbf{z}(t), \tag{19}$$

where $\mathbf{S} = \mathbf{B}\mathbf{R}_2^{-1}\mathbf{B}$, $\mathbf{A}_{\alpha}' = \mathbf{A}' + \alpha\mathbf{I}$, and $\mathbf{z}(t)$ is an excitatory input signal. An energy function is then formulated as $\mathbf{E}(\mathbf{P}) = \frac{1}{2}\|\mathbf{v}\|_2^2$ (where $\|\bullet\|_2$ is the Euclidean norm), which is minimized using the method of steepest descent, resulting in a system of first-order matrix differential equations

$$\dot{\mathbf{P}}(t) = \mu\left[\mathbf{A}_{\alpha}'\mathbf{v}(t)\mathbf{z}(t) + \mathbf{v}(t)\mathbf{z}^{\mathrm{T}}(t)\mathbf{A}_{\alpha}' - \mathbf{v}(t)\mathbf{p}^{\mathrm{T}}(t)\mathbf{S}\right], \tag{20}$$

where $\mu > 0$ is the learning rate parameter, and $\mathbf{p}(t) = \mathbf{P}(t)\mathbf{z}(t)$ as shown in Figure 2. In discrete-time form (the time step Δt is absorbed into the learning rate μ), the learning rule for each training step k becomes

$$\mathbf{P}(k+1) = \mathbf{P}(k) + \mu\Delta\mathbf{P}(k), \tag{21}$$

with the update term $\Delta\mathbf{P}(k)$ given as

$$\Delta\mathbf{P}(k) = \left[\mathbf{A}_{\alpha}'\mathbf{v}(t)\mathbf{z}(t) + \mathbf{v}(t)\mathbf{z}^{\mathrm{T}}(t)\mathbf{A}_{\alpha}' - \mathbf{v}(t)\mathbf{p}^{\mathrm{T}}(t)\mathbf{S}\right].$$

Ham and Collins[11] report that even though the update term is not symmetric, the learning rule will still converge to the positive definite, symmetric solution. They note, however, that performing an additional computation resulting in a symmetric update term will improve convergence:

$$\mathbf{P}(k+1) = \mathbf{P}(k) + \frac{\mu}{2}\left[\Delta\mathbf{P}(k) + \Delta\mathbf{P}^{\mathrm{T}}(k)\right]. \tag{22}$$

Ham and Collins[11] note that the external excitatory vector input signals $\mathbf{z}(t)$ are a set of linearly independent bi-polar vectors given as

$$\begin{aligned} z^{(1)} &= [1 \quad -1 \quad \cdots \quad -1], \\ z^{(2)} &= [-1 \quad 1 \quad \cdots \quad -1], \\ &\vdots \\ z^{(n)} &= [-1 \quad -1 \quad \cdots \quad 1], \end{aligned} \tag{23}$$

where each vector $z^{(k)}$ is presented once to the neural network in an iteration, i.e. for one iteration there is a total of n presentations of the training step given in Eqn. (22), with the solution **P**(k) updated with each training step.

Algorithm Evaluation

Comparisons of single degree-of-freedom responses for the nonlinear algorithm are made with the linear optimal algorithm and the adaptive algorithm. Comparisons are made of both specific force cues at the pilot's head and angular velocity cues, as well as the linear and angular displacement of the simulator.

Wu[12] developed an algorithm that scales the aircraft inputs by a third-order polynomial, maximizing the available motion cues while remaining within the operational limits of the motion system. In order to determine the polynomial gain coefficients for each degree-of-freedom that result in the most desired pilot performance, a series of pilot controlled maneuvers were executed with the adaptive and linear optimal algorithms on the NASA Langley Visual Motion Simulator (VMS). A series of maneuvers were first executed for each algorithm with the nonlinear gain coefficients determined prior to testing. Individual gain coefficients for each degree-of-freedom were then adjusted until the desired pilot perception and performance were reached, while ensuring that the simulator motion platform limits were not exceeded. Using these results, coefficients for the nonlinear algorithm were then tuned separately to produce the desired performance within the motion platform limits.

For the heave mode, the off-line solution of the Riccati equation initially produced one closed-loop eigenvalue of zero, which resulted in the linear optimal control weights being very difficult to tune. This

eigenvalue was a result of the inclusion of the optokinetic channel in the algorithm formulation; the formulation based on the vestibular model alone did not produce a zero eigenvalue. A state reduction using the MATLAB function **"minreal"** was performed on the perceptual model, removing one state and in turn eliminating the closed-loop eigenvalue of zero. The linear optimal control weights could then be tuned to produce the desired specific force cue; matrices **Q** and $\mathbf{R_d}$ were increased to produce the desired onset ramp and magnitude while the filtered white noise break frequency ω was increased to 20π rad/s to eliminate false cues. The heave mode responses for a pulse input of 1 m/s^2 magnitude and 10-second duration are shown in Figure 3.

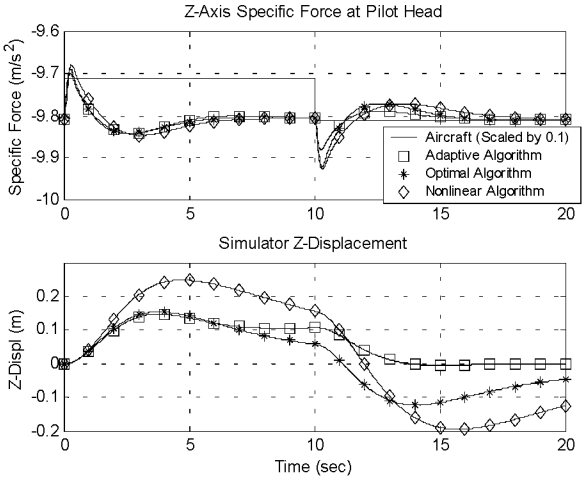

Figure 3. Algorithm Responses to Heave Pulse Input of 1 m/s^2, 10-Second Duration.

A learning rate parameter $\mu = 2 \times 10^{-6}$ is used in computing the real-time solution of the Riccati equation. The onset ramp is very close to that of the adaptive and optimal algorithms, with a slightly larger peak magnitude. The cue is sustained for a longer duration, resulting in 62 percent more z-axis displacement as compared to the linear optimal algorithm. The negative cue at the end of the pulse is twice the magnitude as the adaptive algorithm response.

Figure 4 compares responses with the pulse magnitude increased to 3 m/s^2. The nonlinear algorithm response washes out faster due to the nonlinear effects generated from the Riccati equation solution. A larger z-axis displacement still results, but reduced to 25 percent greater than the linear optimal algorithm. The negative cue at the end of the pulse is slightly smaller than the optimal algorithm response, but is much larger than the negative cue that results from the adaptive algorithm.

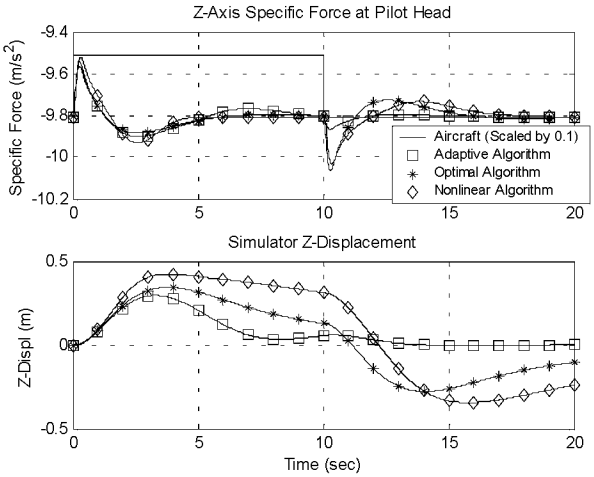

Figure 4. Algorithm Responses to Heave Pulse Input of 3 m/s^2, 10-Second Duration.

For the two-degree-of-freedom longitudinal mode, the initial formulation with the integrated perception model resulted in a higher-order system (15^{th}-order) that is much larger than either heave or yaw (5^{th}-order). Two closed-loop eigenvalues of zero resulted from the off-line solution of the Riccati equation. The first originated from the additional simulator state θ. The second resulted from the optokinetic channel for the translational degree-of-freedom. Removal of the additional platform state combined with a state reduction of the perceptual model eliminates the two closed-loop eigenvalues of zero, reducing the system to 11^{th}-order.

Figure 5 compares the algorithm responses to an aircraft surge ramp to step input. A learning rate parameter $\mu = 2 \times 10^{-6}$ is used in computing the real-time solution of the Riccati equation. Note that the specific force response for the nonlinear algorithm increases to a larger magnitude after onset and does not wash out as a function of time, resulting from the steady-state tilt angle sustaining a constant magnitude. A small increase in the angular velocity (tilt) rate is also observed.

Figure 6 compares the responses for this surge cue from the integrated perception model. The sensed specific force responses show the nonlinear algorithm closely tracks the shape of the sensed response from the aircraft.

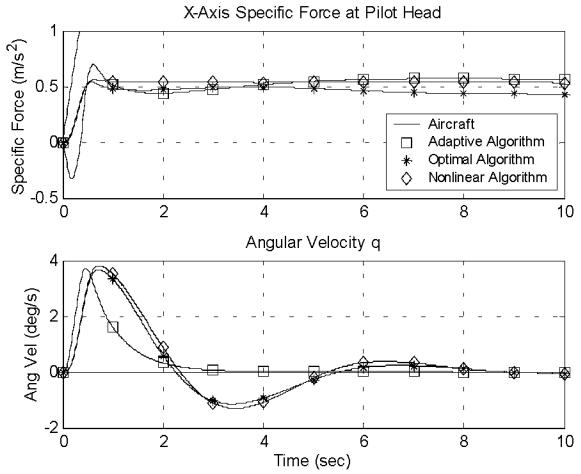

Figure 5. Algorithm Responses to Surge Ramp to Step Input of 1 m/s^2, 3 m/s^2/s Slope.

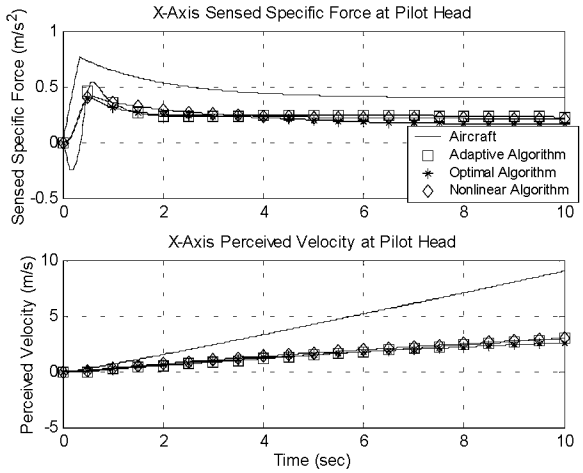

Figure 6. Integrated Perception Model Responses to Surge Cues of Figure 5.

The optimal algorithm produces about the same onset as the nonlinear algorithm, but results in noticeably less sensed response, especially for the first few seconds after the peak magnitude is reached. The perceived velocity responses show larger magnitudes for the nonlinear algorithm, increasing to 15 percent greater magnitude after 10 seconds. The adaptive algorithm shows a negative, or false specific force cue sensed at the onset that results in a subsequent lag and a reduction in the perceived velocity for several seconds.

Figure 7 compares the algorithm responses to an aircraft sway half sine input. As with the longitudinal mode, a state reduction was performed on the integrated perceptual model to eliminate one zero eigenvalue, and the additional simulator state ϕ was removed from the algorithm formulation. A learning rate parameter $\mu = 2 \times 10^{-6}$ was again used.

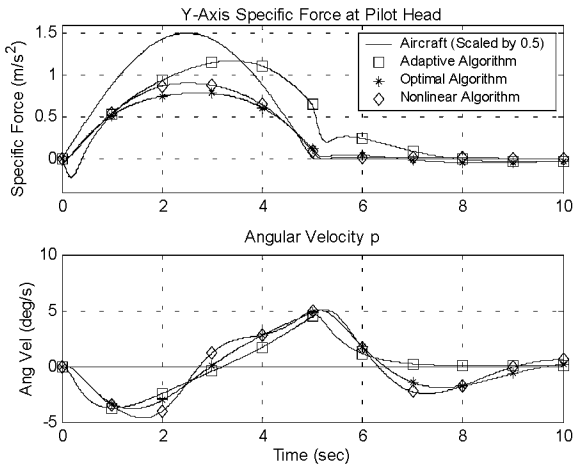

Figure 7. Algorithm Responses to Sway Half Sine Input of 3 m/s^2, 5-Second Duration.

Note that the specific force cue generated by the adaptive algorithm has some significant distortion. A false cue is generated at onset, resulting in a noticeable lag in the motion cue response. A large peak magnitude is reached, but nearly one second after the aircraft input reached its peak. A large residual specific force cue remains for about three seconds after the aircraft input ends. The response generated by the linear optimal algorithm shows no negative cue at the onset, a well-shaped half sine response with a less noticeable lag, and much less residual specific force cue. The nonlinear algorithm results in a peak specific force cue that is 15 percent larger than the linear optimal algorithm, with even less lag and almost no residual specific force cue after the half sine input ends.

Figure 8 compares the responses for these sway cues from the integrated perception model. As expected, the nonlinear algorithm peaks to a larger sensed specific force as compared to the optimal algorithm, resulting in a larger perceived velocity. After five seconds the conflict between the vestibular and visual stimuli is reduced, resulting in a gradual acceptance of the visual cues governed by the optokinetic influence in the model. The problems noted with the adaptive algorithm are evident; the false cue and delayed peak are noticeable along with excessive sensed and perceived responses observed in the last two seconds of the pulse input. In all three algorithms, the magnitude of the vestibular cues eliminates the latency to onset of linearvection that would occur with visual stimuli alone.

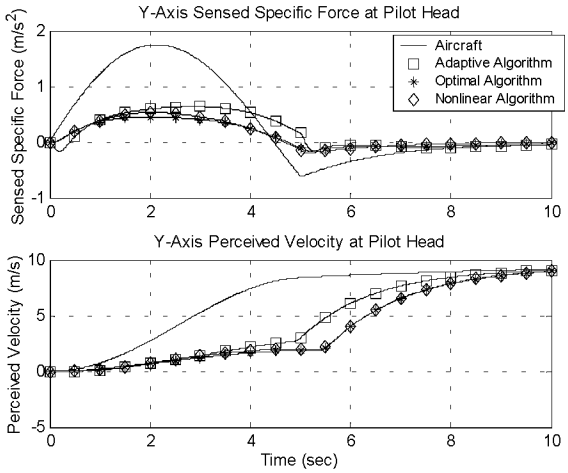

Figure 8. Integrated Perception Model Responses to Sway Cues of Figure 5.

The yaw mode responses for an angular acceleration doublet of 0.1 rad/s^2 magnitude and 5-second duration are shown in Figure 9. A learning rate $\mu = 2 \times 10^{-6}$ is used in computing the real-time solution of the Riccati equation. Note that the angular velocity cue near the end of the aircraft input is reduced for the nonlinear algorithm. The yaw angle displacement command returns to the neutral state (zero displacement) in less than twenty seconds, while the linear optimal algorithm requires more time to return to the neutral state.

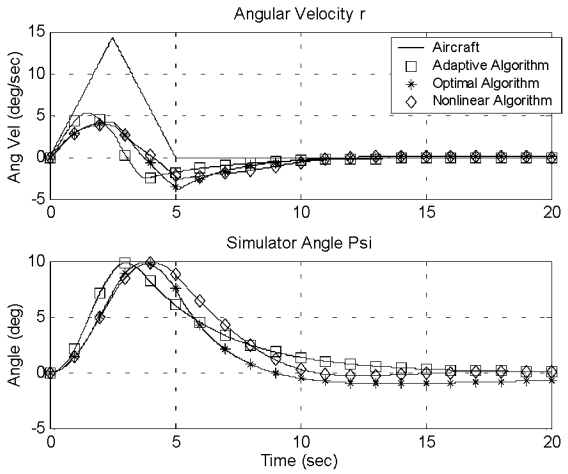

Figure 9. Algorithm Responses to Yaw Doublet of 0.1 rad/s^2 and 5-Second Duration.

Due to the tilt coordination limit that is needed for responses to surge and sway inputs, separate modes are needed respectively for both pitch and roll cues. For both pitch and roll inputs the linear filter frequency characteristics are very close to a unity-gain filter. No additional benefit resulted from solving the Riccati equation in real time. For these reasons, the formulations shown in Eqns. (1) through (23) are replaced by unity-gain filters for both pitch and roll modes.

Figure 10 shows the roll responses for an angular acceleration doublet of 0.1 rad/s^2 magnitude and 5-second duration. Note that the specific force response for the nonlinear algorithm is larger in magnitude (and closer to the aircraft response) as compared to the optimal and adaptive algorithm responses. Pitch responses for the nonlinear algorithm are similar to those previously reported in 1999.[1]

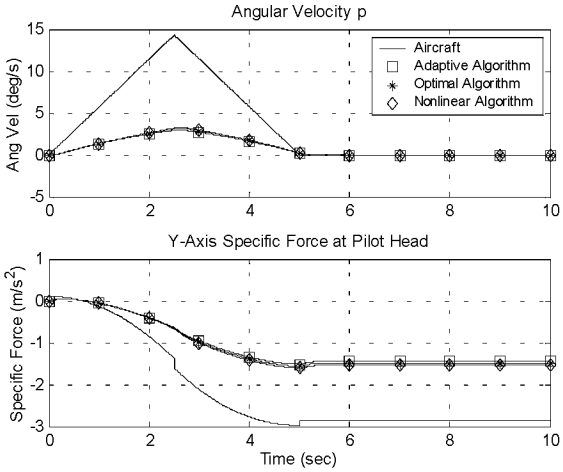

Figure 10. Algorithm Responses to Roll Doublet of 0.1 rad/s^2 and 5-Second Duration.

The systems of first-order differential equations given for the neurocomputing solver in Eqn. (22) require a numerical integration algorithm. A series of algorithms (Euler, 2nd-order Adams-Bashforth, 2nd- and 4th-order Runge-Kutta) were evaluated. No improvement was noticed with the higher-order methods as compared to the Euler method. However, for the system state equations in Eqn. (17), the Euler method was found unstable for small sampling frequencies; the 2nd-order Runge-Kutta method resulted in stable results for sample rates as small as 32 Hz.

The responses using a second neurocomputing solver proposed by Wang and Wu[10] are sensitive to the magnitude and stiffness of the closed-loop eigenvalues, with the responses dependent upon the choice and structure of the activation functions. The approach proposed by Ham and Collins[11] utilizes a structured network without activation functions; the responses are

more robust with respect to the closed-loop eigenvalues. This solver yields improved responses and convergence with less computational burden; only one solver iteration is required per time step.

Piloted Performance Testing

The effectiveness of the nonlinear algorithm as compared to the adaptive and optimal algorithms will be assessed in piloted simulations. Testing will be conducted on the NASA Langley Research Center Visual Motion Simulator (VMS), and ultimately on a new motion system, the Cockpit Motion Facility (CMF).[1] Preliminary testing has been conducted on the VMS with the adaptive and optimal algorithms. As a result of these preliminary tests the polynomial scaling coefficients were adjusted for each degree-of-freedom. Similar testing to adjust the polynomial scaling coefficients will be performed for the nonlinear algorithm.

A group of four pilots will execute a series of maneuvers on the simulator. For each maneuver the simulated aircraft dynamics is generated from manual pilot control. The pilot control inputs (throttle, elevator, aileron, and rudder) will be sampled for each maneuver. Accelerometer measurements for specific force and angular acceleration at the platform motion-base centroid in six degrees-of-freedom will be recorded for each maneuver.

Pilot perception, as computed from the vestibular and integrated perception models, will be recorded for each maneuver. From the pilot control inputs, power spectral density, crossover frequency, and phase margin will be analyzed to determine the effect of motion platform response upon pilot performance, comparing results for the various algorithms. The pilot will also evaluate each maneuver separately using the Cooper-Harper rating scale. Work is currently in progress to develop a performance metric that will utilize these data and benchmark the fidelity of each algorithm in replicating pilot performance and workload of actual aircraft maneuvers.

Conclusions

A nonlinear motion cueing algorithm was developed that combines features of the adaptive and optimal algorithms, and incorporates the vestibular and integrated perception models. A nonlinear control law was proposed that requires the solution of the Riccati equation in real time. The neurocomputing approach implemented for this task yields responses that are robust with respect to the closed-loop eigenvalues, with less computational burden as compared to a second neurocomputing solver.

Results for the heave mode show the nonlinear algorithm producing a motion cue with a time-varying washout, sustaining small cues for a longer duration and washing out larger cues more quickly. The addition of the optokinetic influence from the integrated perception model was shown to improve the response to a surge input, producing a specific force response with no steady-state washout. Improved cues are also observed for responses to a sway input. Yaw mode responses reveal that the nonlinear algorithm improves the motion cues by reducing the magnitude of negative cues.

The nonlinear algorithm will first be implemented on the Langley Visual Motion Simulator, and then on the Cockpit Motion Facility. As was recently done with the adaptive and optimal algorithms, the polynomial scaling coefficients will be tuned to produce the most desired pilot performance. From piloted simulations, a metric will be derived that will demonstrate the effectiveness of the new algorithm in simulating aircraft pilot performance.

References

[1] Telban, R.J., Cardullo, F. M., and Houck, J. A. *Developments in Human Centered Cueing Algorithms for Control of Flight Simulator Motion Systems*. in *AIAA Modeling and Simulation Technologies Conference*. 1999. Portland. OR.

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

[3] 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, 1982. **12**(6): p. 818-827.

[4] Reid, L.D., and Nahon, M. A, *Flight Simulation Motion-base Drive Algorithms: Part 1 - Developing and Testing the Equations*. 1985, UTIAS Report No. 296, CN ISSN 0082-5255.

[5] Telban, R.J., Cardullo, F. M. *An Integrated Model of Human Motion Perception with Visual-Vestibular Interaction*. in *AIAA Modeling and Simulation Technologies Conference*. 2001. Montreal, Canada.

[6] Telban, R.J., Cardullo, F. M., and Guo., L. *Investigation of Mathematical Models of Otolith Organs for Human Centered Motion Cueing Algorithms*. in *AIAA Modeling and Simulation Technologies Conference*. 2000. Denver, CO.

[7] Kwakernaak, H., and Sivan, R., *Linear Optimal Control Systems*. 1972: John Wiley and Sons, Inc., New York.

[8] Anderson, B., D., and Moore, J. B., *Linear Optimal Control*. 1971: Prentice-Hall, Inc., Englewood Cliffs, NJ.

[9] Blackburn, T.R., *Solution of the Algrebraic Riccati Equation via Newton-Raphsen Iteration.* AIAA Journal, 1969. **6**(5): p. 951-953.

[10] Wang, J., and Wu, G., *A Multilayer Recurrent Neural Network for Solving Continuous-Time Algebraic Riccati Equations*. Neural Networks, 1998. **11**: p. 939-950.

[11] Ham, F.M., and Collins, E. G. *A Neurocomputing Approach for Solving the Algebraic Matrix Riccati Equation*. in *IEEE International Conference on Neural Networks*. 1996.

[12] Wu, W., *Development of Cueing Algorithm for the Control of Simulator Motion Systems*. 1997, M.S. Thesis, State University of New York at Binghamton: Binghamton, NY.