JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 23, No. 5, September–October 2000

Adaptive Autopilot Design for Guided Munitions

Anthony J. Calise* and Manu Sharma †
Georgia Institute of Technology, Atlanta, Georgia 30332
and
J. Eric Corban‡
Guided Systems Technologies, Inc., McDonough, Georgia 30253

Traditional autopilot design for guided munitions requires an accurate aerodynamic model and relies on a gain schedule to account for system nonlinearities. This paper presents an approach that simplifies the autopilot design process by combining an inverting controller designed at a single flight condition with an on-line neural network to account for errors that arise because of the approximate inversion. This eliminates the need for an extensive design process and also the requirement for accurate aerodynamic data, which can be especially critical at high angles of attack or in other regimes at which the aerodynamics become highly nonlinear. The choice of the inversion process itself has been found to be critical in the implementation and is therefore discussed at length. Finally, results from an application of this approach to a full nonlinear six-degree-of-freedom guided munition simulation are presented.

Introduction

CONTINUALLY decreasing defense budgets have forced the armed forces to seek more performance at lower costs. This has especially been true for guided munitions. An added complexity for these weapons is the need for modularity so that variants can be created by combining various components around the same body. Thus the ideal controller for these guided munitions would be able to handle several different configurations without compromising performance. Meeting these requirements can be extremely challenging because designing either classical or optimal controllers for an entire family of vehicles requires both wind-tunnel testing and gain scheduling, both of which in turn increase cost dramatically. Finally, these munitions often maneuver through flight regimes characterized by highly nonlinear aerodynamics (such as high angle of attack), which make the dynamics and the control effectiveness highly nonlinear.

One approach to autopilot design that achieves the effect of gain scheduling is the use of dynamic model inversion.[1−4] This approach transforms the nonlinear system to a linear time invariant form. The transformation is then inverted to obtain a nonlinear control law. However, this approach does not eliminate the need for an accurate aerodynamic database. In this paper we consider the addition of an adaptive element to the linearizing controller. Research has shown that on-line neural networks are well suited to cancel inversion errors in such controllers.[5,6] The network weight update law ensures boundedness of both tracking error and network weights.

Another drawback to dynamic model inversion is that it cannot be applied directly to nonminimum phase plants. The transfer function from control surface to acceleration (at the center of gravity) is always nonminimum phase for tail-mounted surfaces. Reference 1 presents an approach to directly invert nonminimum phase systems by simply ignoring the nonminimum phase characteristic of the plant. However, this approach is restricted to plants that are slightly nonminimum phase, a condition that is difficult to quantify. An alternate approach that is applicable to strongly nonminimum phase systems is presented in Ref. 2, when the system possesses a flat output that can be used to control the tracking output indirectly. Neither of these methods allows for explicit inclusion of a rate-damping loop, which is common in traditional missile-autopilot design.

Reference 3 describes an indirect approach that avoids the nonminimum phase problem by first designing an inverting controller for a transfer function that is minimum phase and then adding a classically designed outer loop around it. The nonminimum phase zero will still appear in the outer loop part of the design, but will not interfere with the inverting part of the design. For the longitudinal and directional dynamics this could be done by using pitch and yaw rates respectively for the inner loop and implementing an acceleration commanded outer loop to map from an acceleration error to rate command to the inner loop. A problem with this approach is that the transfer function from control surface deflection to body rate has a zero very close to the origin, the effect of which is to produce a very slow mode when the inversion is inexact. This slow mode can make it difficult to meet design criteria such as rise time and settling time. Reference 3 also noted that this approach can be sensitive to parameter error.

A second possibility is to control the aerodynamic angles (α and β) in the inner loop. This approach, discussed extensively in Ref. 3, avoids the problem of a slow mode in rate design, but results in a zero far in the left-half plane instead. As a result, there are rapid transients in the control response. Moreover, this approach was later found to be sensitive to time delays.[4] This characteristic can be especially problematic in systems that have inherently slow dynamics such as munitions. Because both of the preceding schemes are hampered by zero locations in their respective transfer functions, Ref. 4 uses an alternative approach called *output redefinition* for plant inversion of a tail-controlled missile. In this approach, originally proposed in Ref. 7, the inner-loop variable is defined as a linear combination of the state variables. This allows the designer to place the zero of the associated transfer function at a desirable location. Thus, for example, a combination of both angle of attack and pitch rate could be used to define the commanded inner-loop variable.

In this paper we first make use of output redefinition to design an autopilot for a guided munition based on feedback inversion. The inverting controller is then augmented with a neural network so that the final design can adapt to parametric uncertainty in the inversion process. The paper begins with a summary of the neural-network-based adaptive control approach and follows with the details of the output redefinition approach to inversion. These are then combined to address the guided munition application. Finally, numerical simulation results are presented.

Neural-Network Augmented Dynamic Inversion

This section briefly reviews the use of neural networks to augment feedback control design via dynamic inversion.[5] Reference 5 treats

Received 22 March 1999; revision received 20 November 1999; accepted for publication 4 December 1999. Copyright © 2000 by the American Institute of Aeronautics and Astronautics, Inc. All rights reserved.

*Professor, School of Aerospace Engineering; anthony.calise@ae.gatech.edu. Fellow AIAA.

†Graduate Research Assistant, School of Aerospace Engineering; gt7547d@cad.gatech.edu.

‡President, Guided Systems Technologies Inc.; corban@mindspring.com. Member AIAA.

the case of plant dynamics in second-order form. In this application it is more convenient to consider the dynamics in the first-order form

$$\dot{x} = f(x, \delta) \tag{1}$$

where $x \in R^n$ is the state vector, $\delta \in R^n$ is the control, and $f(x, \delta)$: $R^n \times R^n \longrightarrow R^n$. This system may be recast in the form

$$\dot{x} = u \tag{2}$$

$$u = f(x, \delta) \tag{3}$$

where u is termed the *pseudocontrol*. Equation (3) represents a linearizing transformation of the control. To implement the control, the inverse $f(x, \delta)$ must be known and computed in real time. In general, the mapping is only known approximately. Thus the system is actually transformed into the following form:

$$\dot{x} = u + \Delta(x, u) \tag{4}$$

where

$$\Delta(x, u) = f(x, \hat{\delta}) - \hat{f}(x, \hat{\delta}) \tag{5}$$

In Eqs. (4) and (5) Δ: $R^n \times R^n \longrightarrow R^n$ represents the inversion error, and $\hat{\delta} = \hat{f}^{-1}(x, u)$ is the approximate inverse. Figure 1 illustrates the approximate dynamic inversion process.

In the ideal case the inversion is exact, and a linear time-invariant control theory can be used to design the pseudocontrol to satisfy stability and performance objectives. If, however, the inversion is based on an approximate model (e.g., a linear model at a single flight condition), the inverting controller must be augmented by an adaptive element to compensate for inversion error. In Ref. 5 this is accomplished by augmenting the pseudocontrol with an adaptive component in the following form:

$$u_i(t) = u_{p_i}(t) + \dot{x}_{c_i}(t) - \hat{u}_{\mathrm{ad}_i}(t), \qquad i = 1, 2, \ldots, n \tag{6}$$

where i indicates each individual pseudocontrol input channel, and $u_{p_i}(t)$ and $\hat{u}_{\mathrm{ad}_i}(t)$ are the linear (proportional) and adaptive control contributions, respectively. The term $\dot{x}_{c_i}$ represents the first time derivative of the command input and is introduced by the proof of stability.[5] This approach allows for the implementation of a separate adaptive element in each state equation. Figure 2 gives the adaptive inverting control architecture for a single channel. In general, all of the states and all of the pseudocontrols are potential inputs to each neural network. The cross-fed pseudocontrol signals are depicted as u_{-i}.

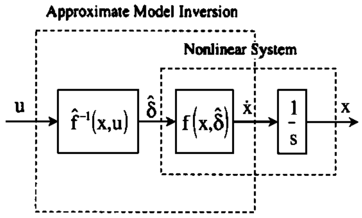

Fig. 1 Approximate model inversion.

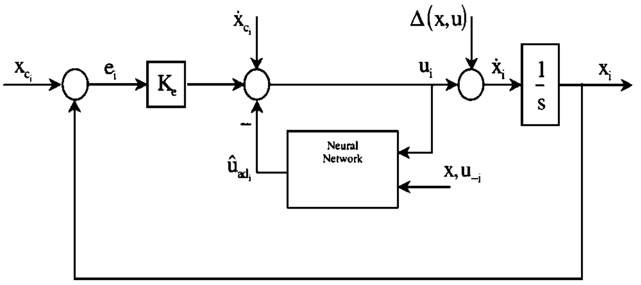

Fig. 2 Adaptive control architecture.

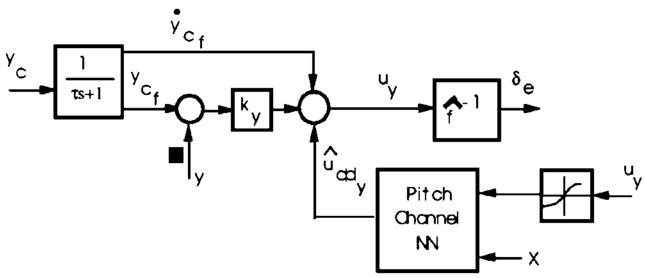

Fig. 3 Pitch channel inner loop.

The network architecture used in Ref. 5 is linear in the weights and can be represented in the following form:

$$\hat{u}_{\mathrm{ad}_i} = \sum_{j=1}^{N} \hat{w}_{i,j}\xi_{i,j}(x, u) = \hat{w}_i^T \xi_i \tag{7}$$

where $\hat{w}_{ij}$ are scalar weights and the ξ_i are an N-dimensional set of basis functions. For this form, the update law for the weights in first-order form reduces to

$$\dot{\hat{w}}_i = -\gamma_i \xi_i e_i \tag{8}$$

where $\gamma_i > 0$ is the learning rate and

$$e_i = x_{c_i} - x_i \tag{9}$$

To ensure boundedness of the weighting vector, a standard e-modification term was also incorporated as follows:

$$\dot{\hat{w}}_i = -\gamma_i(\xi_i e_i + \eta |e_i| \hat{w}_i) \tag{10}$$

where $\eta > 0$ is the e-modification factor. An extension of this approach to networks with a hidden layer (networks that are nonlinearly parameterized by their weights) may be found in Ref. 8.

Figure 2 shows that the network input u_i depends on the current network output. This induces a critical assumption on the existence of a fixed-point solution for the output $\hat{u}_{\mathrm{ad}_i}$ in the stability proof. A simple iterative scheme can be employed to compute the network output when a stable fixed point exists. To ensure the existence of a fixed point, the input u_i is passed through a squashing function at the input side of the network (see Fig. 3).

Output Redefinition

Reference 3 shows that dynamic inversion amounts to inverting the open-loop transfer function from the control to the output being controlled. Thus, nonminimum phase plants cannot be inverted. Difficulties are also encountered when the plant has zeros located either near the imaginary axis or far in the left-hand plane. In Ref. 4 a method called *output redefinition* is recommended for non-minimum phase plants, which is briefly outlined next within the setting of pitch autopilot design.

Consider the linearized longitudinal dynamics for which the transfer functions from effective elevator deflection to pitch rate q and angle of attack α are given by

$$G_{q,\delta}(s) = \frac{K_q(s + a_q)}{D(s)} \tag{11}$$

and

$$G_{\alpha,\delta}(s) = \frac{K_\alpha(s + a_\alpha)}{D(s)} \tag{12}$$

In Eqs. (11) and (12) $-a_q$ and $-a_\alpha$ are the zeros of their respective transfer functions. A short-period approximation yields the following expressions for K_q, a_q, K_α, and a_α in terms of conventionally defined stability and control derivatives in dimensional form.[9]

$$K_q = M_\delta \qquad K_\alpha = Z_\delta / U_0$$

$$a_q = (1/U_0)(M_\alpha Z_\delta / M_\delta - Z_\alpha) \qquad a_\alpha = U_0 M_\delta / Z_\delta \tag{13}$$

The difficulty that arises in applying dynamic inversion to guided munition is that $a_q \approx 0$ and $a_\alpha \gg 1$, largely because $U_0 \gg 1$.

Now redefine the output as

$$y \equiv \alpha + C_q q \tag{14}$$

This yields the following transfer function:

$$G_{y,\delta}(s) = \frac{K_y(s + a_y)}{D(s)} \tag{15}$$

The zero of this transfer function is located at $-a_y$, where

$$a_y = (K_\alpha a_\alpha + C_q K_q a_q)/K_y \tag{16}$$

and

$$K_y = K_\alpha + C_q K_q \tag{17}$$

Solving for C_q yields

$$C_q = \frac{K_\alpha(a_\alpha - a_y)}{K_q(a_y - a_q)} \tag{18}$$

The objective in output redefinition is to select C_q so that a_y is $\mathcal{O}(1)$. Because $a_q \approx 0$ and $a_\alpha \gg 1$, it follows that $a_\alpha - a_y \approx a_\alpha$ and $a_y - a_q \approx a_y$. Thus C_q can be reasonably approximated as

$$C_q \approx K_\alpha a_\alpha / K_q a_y \approx 1/a_y \tag{19}$$

A similar analysis of the yaw channel yields $C_r \approx -1/a_z$ for $z = \beta + C_r r$, where a_z is the zero of the transfer function from rudder deflection to the new output z. The difference in sign between the longitudinal and directional control arises solely from a difference in sign convention between longitudinal and directional axes. Output redefinition was not used in the roll channel because the transfer function from effective aileron deflection to roll rate does not have any finite zeros.

Application to Guided Munitions

The objective in this section is to design a bank-to-turn autopilot that tracks guidance commands consisting of z-body axis normal acceleration ($A_{\rm nc} = -A_{\rm zc}$), zero y-body axis acceleration ($A_{\rm yc} = 0$) and roll rate (p_c). The dynamic plant inversion is designed at a single flight condition, and the neural network is used to correct for error as the vehicle moves away from this design point.

Longitudinal and Directional Control

The design in the longitudinal and directional axes is based on a classical two-loop design. The outer loop is used to regulate acceleration and generates a command signal to the inner loop. Feedback inversion with output redefinition is then applied in the inner loop. The outer-loop portion is illustrated in Fig. 4.

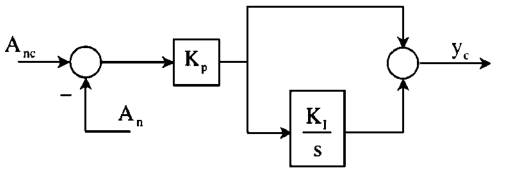

Fig. 4 Outer-loop schematic for the longitudinal and directional channels.

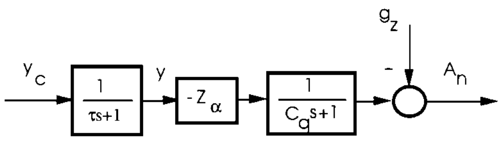

Fig. 5 Idealized transfer function representation of the inner loop.

Only the design for the longitudinal axis is presented because the design for the directional axis is essentially identical in form. The linearized longitudinal equations are given by

$$U_0(\dot\alpha - q) = Z_\alpha \alpha + Z_\delta \delta + g_z = A_z \tag{20}$$

$$\dot q = M_\alpha \alpha + M_q q + M_\delta \delta \tag{21}$$

where g_z is the z-body axis component of the gravity vector. Dynamic inversion is based on the redefined output variable y. Differentiating Eq. (14) and using Eqs. (20) and (21) results in the following approximate expression for the inverse:

$$\delta_e = \frac{\{U_0[u_y - q - C_q(M_\alpha \alpha + M_q q)] - Z_\alpha \alpha - g_z\}}{(Z_\delta + U_0 C_q M_\delta)} \tag{22}$$

where u_y is the corresponding pseudocontrol (desired rate of change of y).

The architecture of the pitch-channel inner loop is in essence identical to that given in Fig. 2, except that the command is filtered. The filter time constant is used to regulate (in a model following sense) the transient response of the inner loop. The filter also generates the command time derivative required by the neural-network implementation. The resulting inner-loop architecture is given in Fig. 3, with the function $\hat f^{-1}$ given by Eq. (22). The gain k_y is used to regulate the error transient and is chosen sufficiently large to ensure that the error transient is fast compared to the time constant τ. A reasonable choice is $k_y = 3/\tau$.

With the outer and inner loops defined, only the controller parameters (K_p, K_I, C_q, and τ) remain to be determined. This requires examining the closed-loop dynamics under the assumption that perfect model following is achieved in the inner loop or that $y(s) = yc(s)/(\tau s + 1)$. Thus, to close the outer loop (Fig. 4), it is necessary to determine the transfer function from $y(s)$ to $A_n(s)$. A simple representation of this transfer function follows if the following approximations are introduced: 1) $Z_\delta = 0$, and 2) $-1/Z_\alpha \gg C_q/U_0$. Equation (20) is used to eliminate $\alpha(s)$ and $q(s)$ in Eq. (14). The block diagram in Fig. 5 summarizes this result.

From inspection of Figs. 4 and 5, the loop transfer function is given by

$$G(s) = \frac{-K_p Z_\alpha(s + K_I)}{s(\tau s + 1)(C_q s + 1)} \tag{23}$$

This expression shows that the system has a zero at $-K_I$ and three poles at 0, $-1/\tau$, and $-1/C_q$. There are several ways to approach the design. For example, the choice $K_I = 1/C_q$ results in a pole-zero cancellation in which both C_q and K_I are known exactly. The outer-loop proportional gain and the parameter τ can then be selected by specifying a natural frequency and damping ratio for the resulting closed-loop system using the following relationships:

$$\tau = 1/2\zeta\omega_n \tag{24}$$

$$K_p = -C_q \omega_n / 2\zeta Z_\alpha \tag{25}$$

An important consequence is that Z_α is the only aerodynamic coefficient that appears explicitly in the design equations.

The most significant approximation in the preceding analysis is the assumption that $Z_\alpha = 0$. It is straightforward to show that the

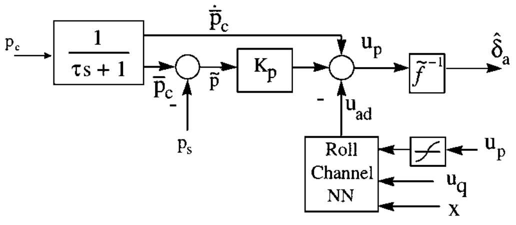

Fig. 6 Roll autopilot.

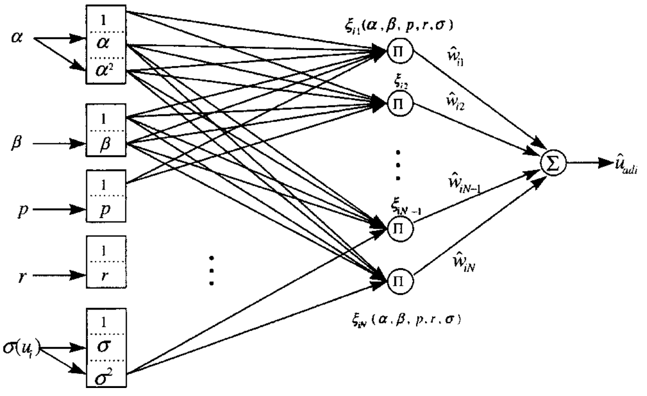

Fig. 7 Neural-network architecture.

transfer function in Eq. (23) contains two additional zeros, corresponding to the roots of

$$M_\delta Z_\alpha + Z_\delta D(s) = 0 \tag{26}$$

where

$$D(s) = s^2 - (Z_\alpha / U_0 + M_q)s - M_\alpha + Z_\alpha M_q / U_0 \tag{27}$$

For an accelerometer located at the c.g., Eq. (26) has two real zeros, one of which is nonminimum phase. However, Ref. 10 shows that by proper choice of the sensor location (or by appropriate modification of the sensor output so that it appears to be at a selected location along the x-body axis) it is possible to place these neglected outer-loop zeros arbitrarily distant from the origin. When the location is chosen so that the zeros of Eq. (26) bifurcate at ∞, the loop transfer function may be very closely approximated by Eq. (23), for values of Z_δ such that

$$|Z_\alpha M_\delta / Z_\delta M_\alpha| \gg 1 \tag{28}$$

which is of order 10 for most aerodynamic configurations.

Roll Control

The roll-channel implementation is much simpler because the transfer function from fin deflection to roll rate has no finite zeros. The block diagram for the roll channel is given in Fig. 6.

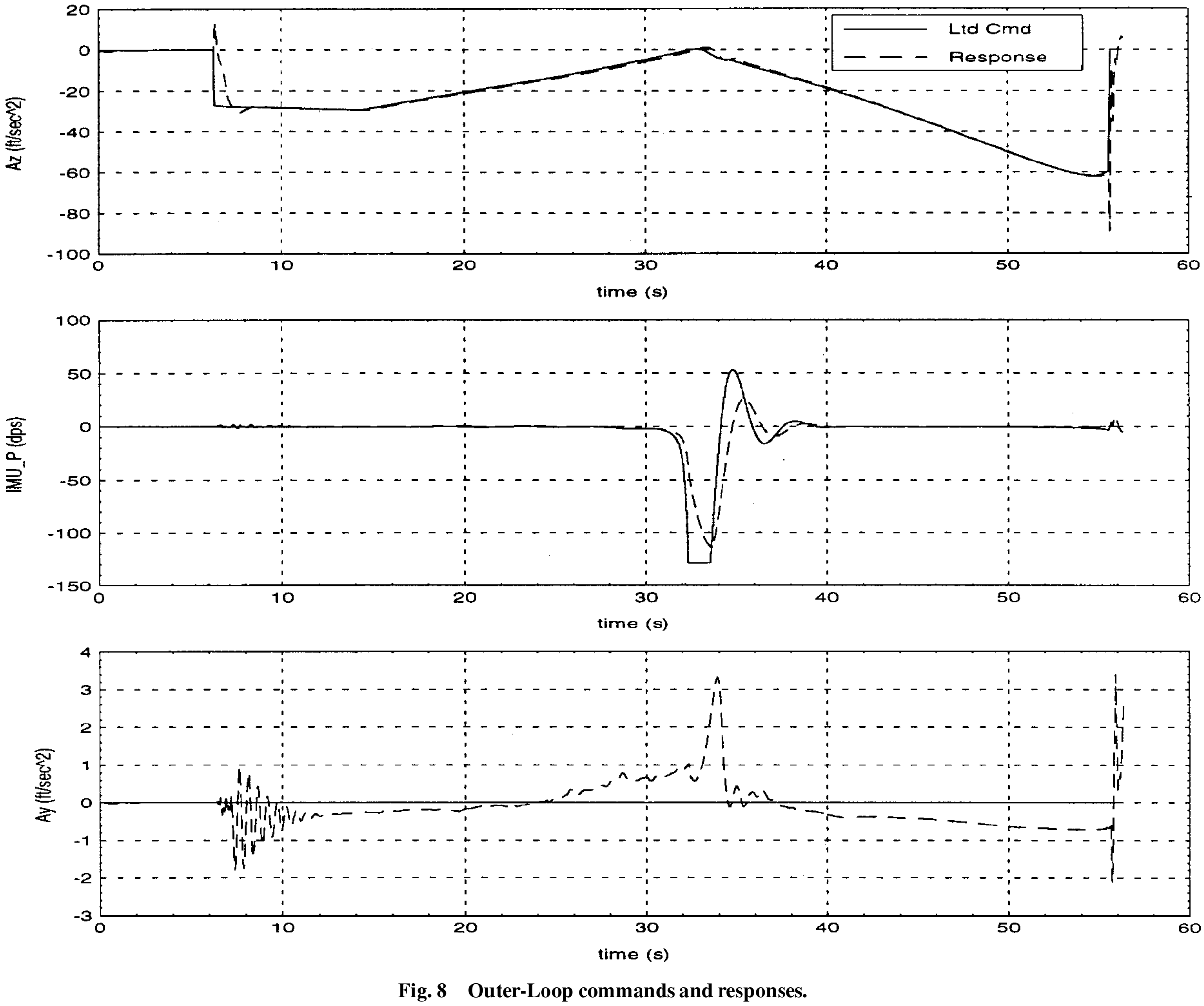

Fig. 8 Outer-Loop commands and responses.

Neural-Network Architecture

The final segment of this design is the neural-network architecture. The network for each channel is implemented separately. The lateral and directional channels both use the form diagrammed in Fig. 7, whereas pitch rate, Mach number, and altitude are substituted for sideslip, roll, and yaw rates in the longitudinal channel. The pitch-axis pseudocontrols are used as inputs to the lateral and directional neural networks to allow for cross coupling during hard pull-up maneuvers.

Simulation Results

A nonlinear, six-degree-of-freedom simulation of a tail-controlled guided munition was used to study the performance of the adaptive autopilot. Boeing Phantom Works, St. Louis, MO, provided the simulation model. A typical trajectory for this munition is a drop from 30 kft with a target 40 kft downrange. Figure 8 presents the limited commands from the guidance loop (Ltd Cmd) and system response in all three axes. The commands are normal acceleration (Az), roll rate (IMU_P), and lateral acceleration (Ay). The autopilot provides acceptable command following across the flight envelope without the benefit of gain scheduling and also compares well with autopilots that are gain scheduled. Performance was specified in terms of rise time, which was scheduled as a function of dynamic pressure.

The adaptive term allows the inverting design to use a fixed set of nondimensional aerodynamic coefficients. The case shown employs coefficients that are purposefully in error even at the starting condition. The simulation also includes phantom yaw effects (i.e., yaw disturbances that grow with angle of attack) and actuator dynamics. Phantom yaw effects produce the approximately 2.5-Hz oscillation in the lateral acceleration at 7 s. Figures 9–12 present the corresponding angle of attack (alpha), pitch rate (IMU_Q), angle of sideslip (beta), and yaw rate (IMU_R). Figures 13–15 present the inner-loop tracking performance and selected network weights for the pitch, roll, and yaw channels, respectively. Figure 16 shows the time history of each of the three tail fins.

The linear-in-the-parameters neural network has some limitations. During hard maneuvers, for instance, it is not able to completely parameterize the inversion error. Reference 8 describes an adaptive law for a single hidden layer network, which permits a nonlinear parameterization of the inversion error. Such networks are better suited for highly nonlinear applications and would be a better candidate in future applications of this approach.

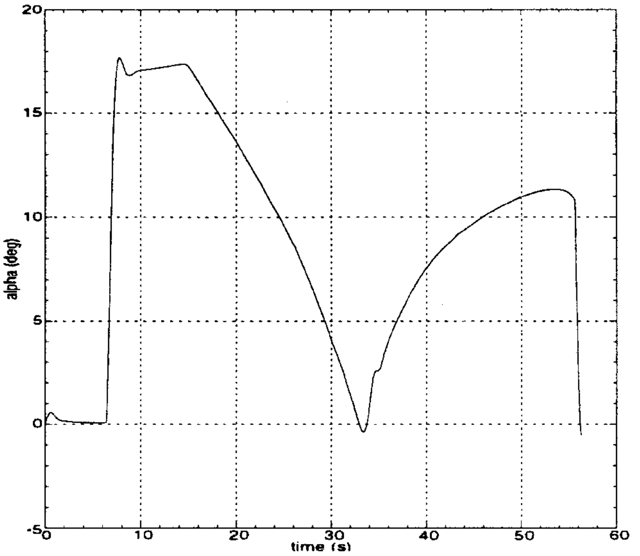

Fig. 9 Angle of attack (deg).

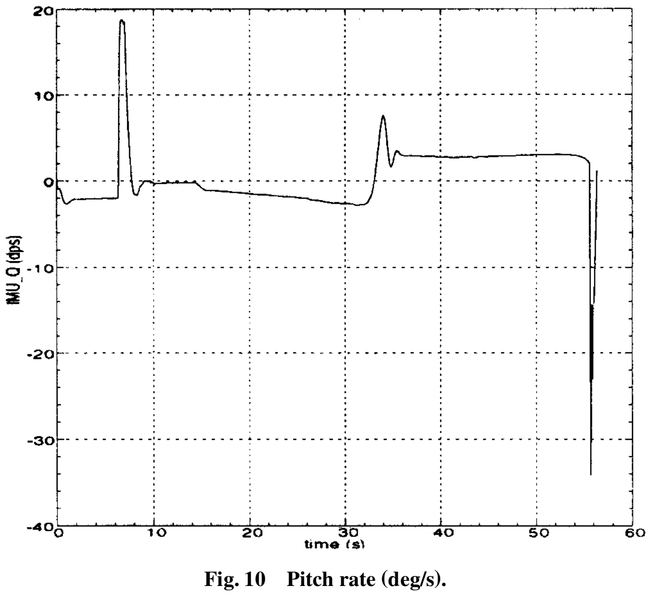

Fig. 10 Pitch rate (deg/s).

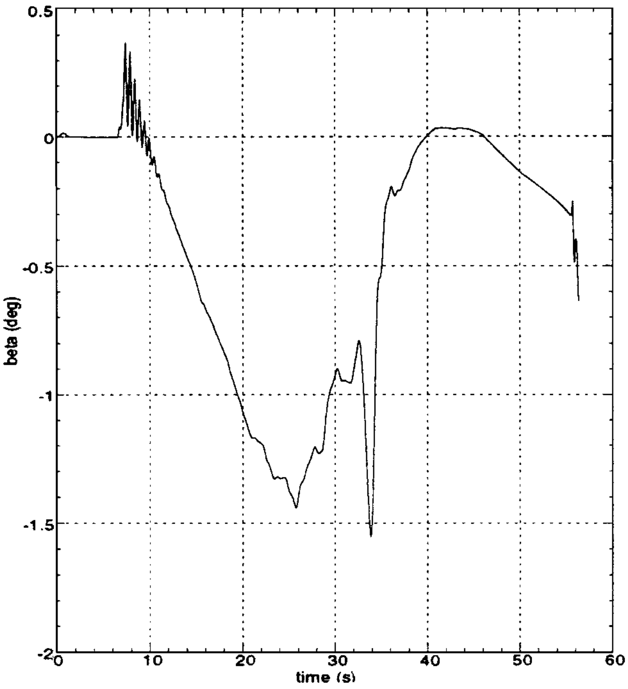

Fig. 11 Angle of sideslip (deg).

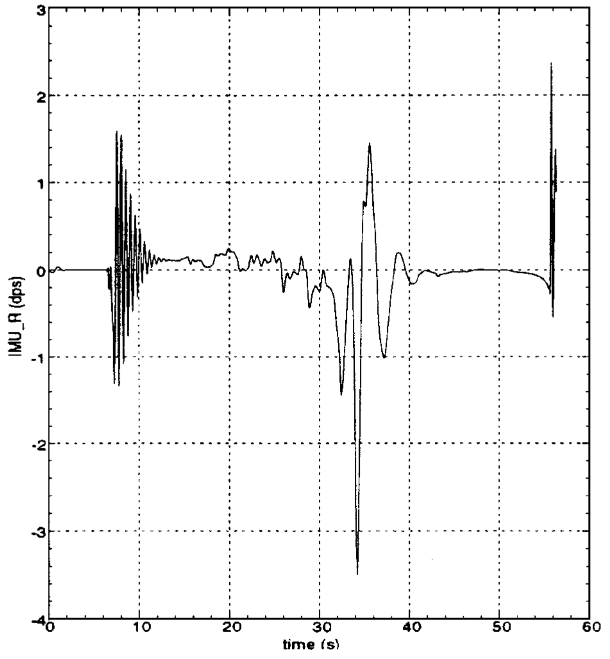

Fig. 12 Yaw rate (deg/s).

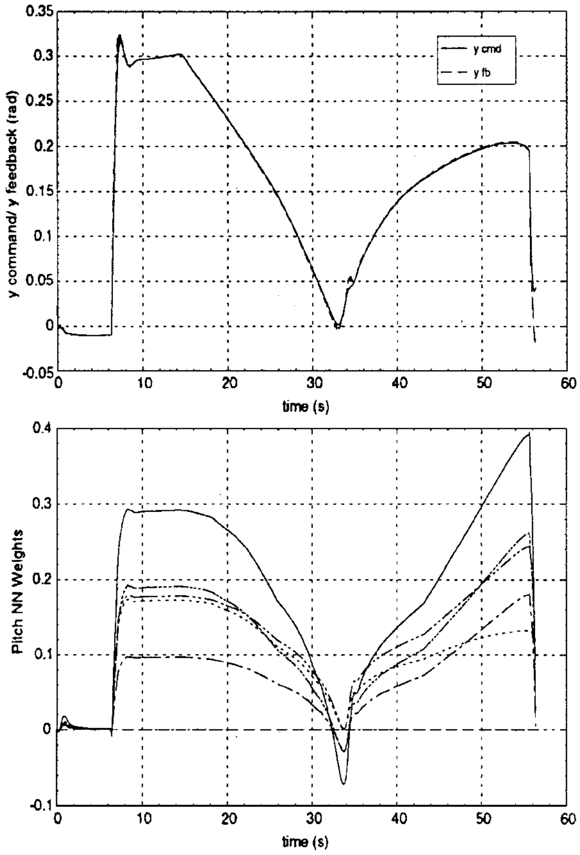

Fig. 13 Pitch inner-loop tracking and select network weights.

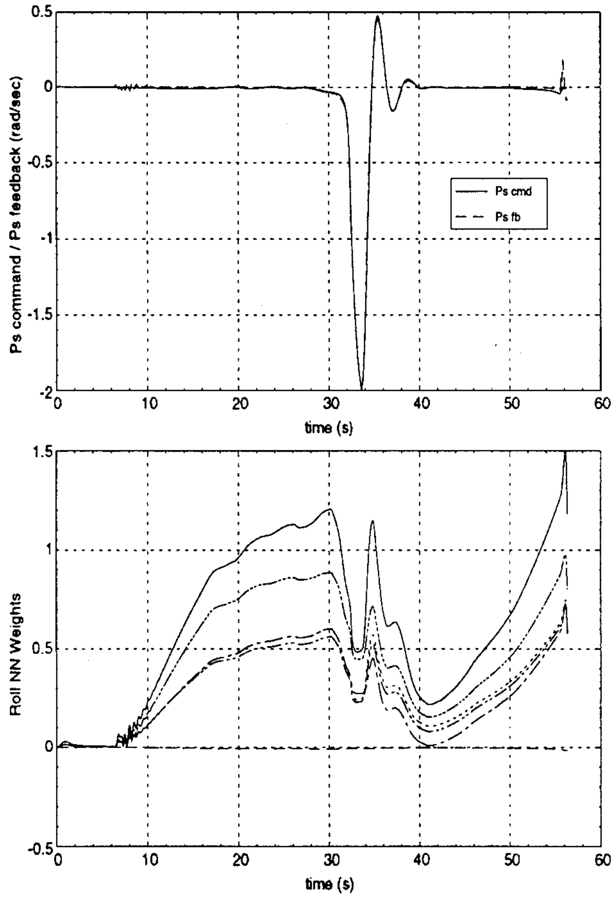

Fig. 14 Roll inner-loop tracking and select network weights.

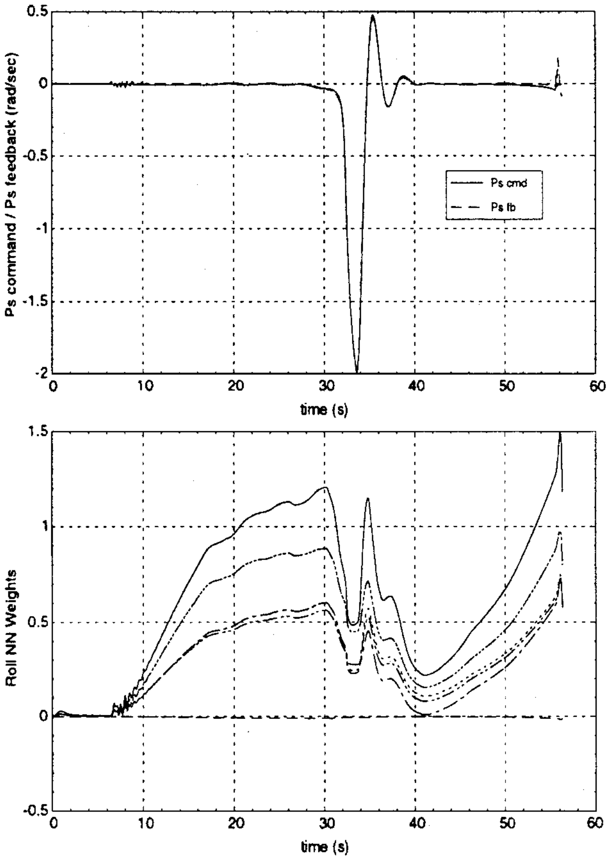

Fig. 15 Yaw inner-loop tracking and select network weights.

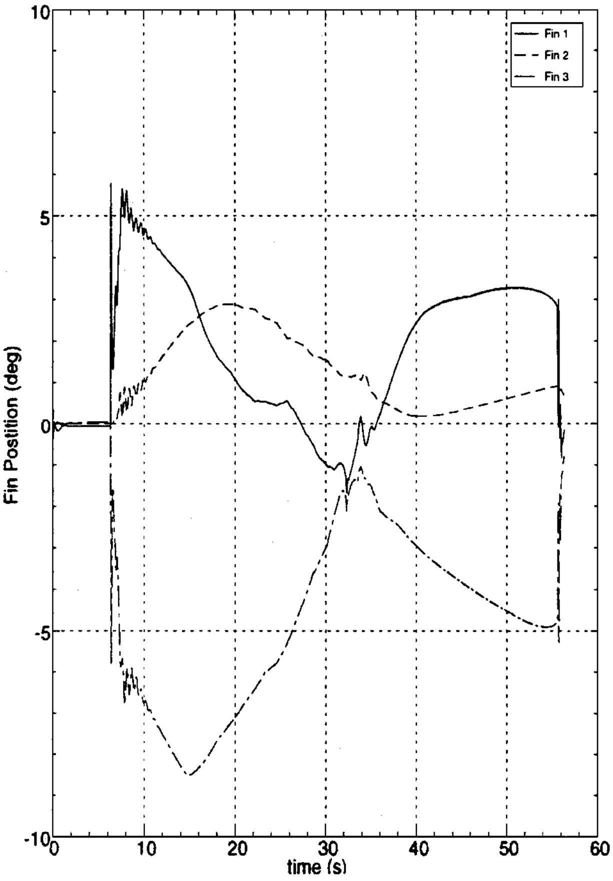

Fig. 16 Tail-fin deflections.

Conclusions

A neural-network-based adaptive inverting controller design has been developed and evaluated for a guided munition application. The design, which uses a single flight condition, resulted in acceptable performance across the flight envelope without the benefit of gain scheduling and illustrates the potential to reduce dependence on wind-tunnel data. Although adaptive control can eliminate the need for accurate aerodynamic data for new munition configurations, an accurate simulation is needed to validate the design. Therefore, it would be desirable to first demonstrate the approach for the existing inventory of munitions, for which aerodynamic data sets already exist. Such an effort is currently underway.

Acknowledgments

This research was sponsored by the U.S. Air Force under contract number F08630-97-C-0041. The Boeing Company provided the nonlinear munition simulation used to generate the numerical results and the gain-scheduled autopilot used for comparison.

References

[1] Sastry, S., *Nonlinear Systems: Analysis, Stability, and Control*, Springer–Verlag, New York, 1999, pp. 425–433, 468–498.

[2] Martin, P., Devasia, S., and Paden, B., "A Different Look at Output Tracking: Control of a VTOL Aircraft," *Automatica*, Vol. 32, No. 1, 1996, pp. 101–107.

[3] Tahk, M.-J., Briggs, M. M., and Menon, P. K. A., "Applications of Plant Inversion via State Feedback to Missile Autopilot Design," *Proceedings of the 27th IEEE Conference on Decision and Control*, IEEE, Piscataway, NJ, 1988, pp. 730–735.

[4] Ryu, J.-H., Park, C.-S., and Tahk, M.-J., "Plant Inversion Control of Tail Controlled Missiles," *Proceedings of the 1997 AIAA Guidance Navigation and Control Conference*, AIAA, Reston, VA, pp. 1691–1696.

[5] Kim, B. S., and Calise, A. J., "Nonlinear Flight Control Using Neural Networks," *Journal of Guidance,Control, and Dynamics*, Vol. 20, No. 1, 1997, pp. 26–33.

[6] McFarland, M. B., and Calise, A. J., "Neural-Adaptive Nonlinear Autopilot Design for an Agile Anti-Air Missile," AIAA Paper 96-3914, July 1996.

[7] Hedrick, J. K., and Gopalswamy, S., "Nonlinear Flight Control Design via Sliding Methods," *Journal of Guidance, Control, and Dynamics*, Vol. 13, No. 5, 1990, pp. 850–858.

[8] McFarland, M., and Calise, A. J., "Multilayer Neural Networks and Adaptive Nonlinear Control of Agile Anti-Air Missiles," AIAA Paper 97-3540, Aug. 1997.

[9] Etkin, B., *Dynamics of Atmospheric Flight*, Wiley, New York, 1972, pp. 144–150.

[10] Wise, K., and Broy, D., "Agile Missile Dynamics and Control," AIAA Paper 96-3912, July 1996.