Software Tools for Nonlinear Missile Autopilot Design

P.K. Menon[*], V.R. Iragavarapu[†]
and G. Sweriduk[‡]
Optimal Synthesis Inc.
470 San Antonio Road, Suite 200
Palo Alto, CA 94306

E. J. Ohlmeyer[§]
Naval Surface Warfare Center
Dahlgren, VA

Abstract

A computer-aided design software package for nonlinear control synthesis is discussed. The software incorporates five different modern nonlinear control methods. The versatility of the software lies in its ability to develop nonlinear controllers using a simulation model. As a result, models of arbitrary complexity can be used in the control law development. The use of the design software is illustrated through the design of nonlinear regulators for a longitudinal missile model.

1. Introduction

Methods for nonlinear control system design has been of interest in the recent literature[1-7]. This paper discusses the development of a software package that can be used for nonlinear controller synthesis. Design techniques presented here can handle a large class of system nonlinearities found in applications, including saturation limits, friction and backlash. They can also provide nonlinear controllers for systems containing look-up tables such as the aerodynamic data in flight vehicle dynamic models.

Nonlinear control system design methods discussed in the literature assume that the mathematical models of the dynamic system are available in symbolic form. Nonlinear controllers are synthesized by manipulating the symbolic model. While useful as a teaching tool, methods based on symbolic manipulations are not amenable for use in more complex applications due to presence of lookup tables and complex nonlinearities not directly describable in terms of symbolic expressions.

The objective of the present paper is to present a set of nonlinear control techniques that allow the use of a numerical simulation model of the system for nonlinear controller synthesis. The application of these design techniques will be demonstrated using a longitudinal missile model obtained from Reference 8. In the interests of completeness, the missile model from Reference 8 is described in section 2. Nonlinear controller design methods are detailed in section 3. Section 4 presents the simulation results using the design software.

2. Missile Longitudinal Dynamics

The longitudinal dynamic model of the missile employed in the present research is from Reference 8. This model consists of four nonlinear differential equations describing the pitch-plane rigid-body dynamics of a missile. This model assumes constant mass, with all the lateral state variables being zero. The equations of motion are of the form:

$$\begin{aligned}\dot{M} = {} & 0.4008\,M^2\alpha^3\sin(\alpha) - 0.6419\,M^2|\alpha|\alpha\sin(\alpha) \\ & - 0.2010\,M^2\left(2-\frac{M}{3}\right)\alpha\sin(\alpha) - 0.0062\,M^2 \\ & - 0.0403\,M^2\sin(\alpha)\delta - 0.0311\sin(\gamma)\end{aligned} \quad (1)$$

$$\begin{aligned}\dot{\alpha} = {} & 0.4008\,M\alpha^3\cos(\alpha) - 0.6419\,M|\alpha|\alpha\cos(\alpha) \\ & - 0.2010\,M\left(2-\frac{M}{3}\right)\alpha\cos(\alpha) - 0.0403\,M\cos(\alpha) \\ & + 0.0311\frac{\cos(\gamma)}{M} + Q\end{aligned} \quad (2)$$

$$\begin{aligned}\dot{\gamma} = {} & -0.4008\,M\alpha^3\cos(\alpha) + 0.6419\,M|\alpha|\alpha\cos(\alpha) \\ & + 0.2010\,M\left(2-\frac{M}{3}\right)\alpha\cos(\alpha) + 0.0403\,M\cos(\alpha)\delta \\ & - 0.0311\frac{\cos(\gamma)}{M}\end{aligned} \quad (3)$$

$$\begin{aligned}\dot{Q} = {} & 49.82\,M^2\alpha^3 - 78.86\,M^2|\alpha|\alpha \\ & + 3.60\,M^2\left(-7-\frac{8M}{3}\right)\alpha - 14.54\,M^2\delta - 2.12\,M^2 Q\end{aligned} \quad (4)$$

[*] President, Associate Fellow AIAA

[†] Research Scientist

[‡] Research Scientist, Senior Member AIAA

[§] Associate Fellow AIAA

In these equations, the aerodynamic coefficients have been replaced with their functional relationships given in Reference 8. The equations of motion incorporate Mach number M, angle of attack α, flight path angle γ and pitch rate Q as the state variables and the pitch fin deflection δ as the control input. Note that the dynamic system is strongly nonlinear.

3. Nonlinear Control Design Methods

The nonlinear controller design software is a collection of functions that numerically implement five nonlinear control synthesis techniques. As stated in the introduction, these methods employ a simulation of the dynamic system. The well-known MATLAB®/SIMULINK® numerical environment[9, 10] is used to standardize the formulation of the simulation models to be used in the design process. The user can provide the model as a SIMULINK® block diagram, a MATLAB® function or a *dynamic link library* for use with MATLAB®. The design model of the system is a simplified version of the original system, with no actuator and sensor dynamics. Moreover, disturbances and uncertainties that may be present in the truth model are eliminated in the design model.

The nonlinear dynamic model used for controller synthesis is assumed to be of the form:

$$\dot{x} = f(x) + g(x)u$$

If the system is specified in a more general form:

$$\dot{x} = h(x, u),$$

it can be augmented with integrators at the input to convert the model into the standard form. Thus, the augmented model

$$\dot{x} = h(x, u),\ \dot{u} = u_c$$

is in the standard form with u_c as the control vector.

The nonlinear design techniques discussed in this paper may be broadly classified into Transformation Based Methods and Direct Methods as illustrated in Figure 1. This classification is based on the way the nonlinear design techniques utilize the system dynamic model. In the transformation-based approaches, the given dynamic model is transformed either to the Brunovsky's canonical form[2, 11] or to the state-dependent coefficient form[1]. Transformed models are then used to design a state-dependent Riccati equation controller[1] or a feedback linearized controller[2 - 4].

Direct Methods, as the name implies, do not require any transformation to be applied on the original nonlinear system. These methods employ the user supplied models in the standard form to synthesize the controller. Three design methods are available in the design software. These are, a) Quickest Descent method[5], b) Back-Stepping technique[6] and c) Predictive Control[7] approach.

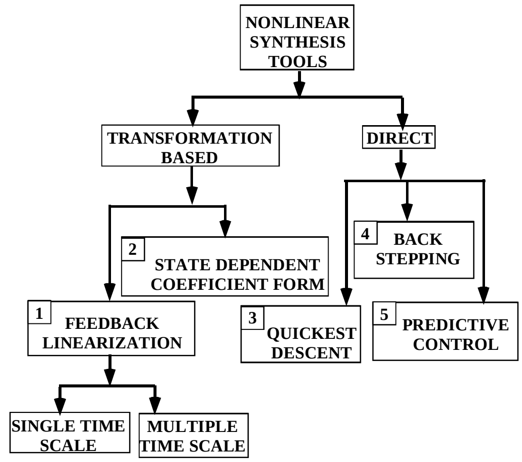

Figure 1 Methods implemented in the Nonlinear Control Design Software

The following subsections will describe the five different design approaches in further detail.

3.1. Feedback Linearization Method

In this approach, the system dynamic model is transformed into the Brunovsky's canonical form. Pseudo-control variables are then defined to place the system in linear, time-invariant form.

For a dynamic system with one control variable, the transformed model is the form: $\dot{z} = Az + Bv$, where z is the transformed state. A and B are constant matrices described as in equation (5):

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \text{ and } B = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \qquad (5)$$

Pseudo control variable, v will be of the form: $v = F(x) + G(x)u$. $F(x)$ and $G(x)$ are the nonlinear functions of the states. Note that the transformation pushes all the system nonlinearities to the input.

After the system is transformed into this form, any linear control design method can be applied to derive the pseudo control, v. Actual control, u can then be obtained from above relationship. Three pseudo control design methods are incorporated in the software. These are: a) Robust Pole Placement method[12], b) Linear Quadratic design technique[13] and c) the Sliding Mode technique[3].

It has previously been observed[14] that in higher-order dynamic systems, the nonlinear controller robustness can be significantly enhanced by designing multiple time-scale controllers. Robustness in time-scale separated controllers result from the fact that higher-order partial derivatives of the nonlinearties on the right hand sides of the nonlinear model are not used in control law derivation. Additionally, time-scale separated nonlinear controllers exploit the hierarchical structure of the system states to simplify control law implementation. The nonlinear control system design software allows the user to impose any desired time-scale structure on the controllers. Time-scale separated dynamics can then be employed for feedback linearized controller design. Transformed model after time-scale separation can be used in conjunction with any of the linear control design methods.

The design software does not permit the use of sliding mode controllers in multiple time-scale designs. Since multiple time scale designs are based on the assumption that fast time-scale designs do not interact with the slow time-scale designs, the sliding mode design methodology is inconsistent with the notion of multiple time-scale controllers. This is due to the fact that sliding mode controllers employ high-frequency switching control laws. With sliding mode designs , any assumed time-scale structure is no longer valid since all time scale designs will essentially chatter at the same frequency.

3.2. State-Dependent Riccati Equation Method

State Dependent Riccati Equation (SDRE) method[1] is the second technique that uses transformed dynamic model for nonlinear controller design. This approach first transforms the user specified dynamic model into the State Dependent Coefficient (SDC) form. SDC form is an instantaneous parameterization of the original nonlinear system with state dependent coefficients A(x) and g(x).

The given nonlinear dynamic system $\dot{x} = f(x) + g(x)u$ is transformed into the SDC form, $\dot{x} = A(x)x + g(x)u.$ Infinite number of such realizations can be shown to exist[1]. However, only those parameterizations for which the pair $[A(x), g(x)]$ is controllable at the given x should be considered for the design.

Note that the SDC parameterization is distinct from the conventional Taylor series linearization, or optimization-based linear approximations[15].

The SDC form of the system state equations are used to cast the control problem as an infinite horizon nonlinear regulator minimizing the cost:

$$J = \frac{1}{2}\int_{t_0}^{\infty} x^T Q(x) x + u^T R(x) u \, dt \tag{6}$$

subject to the nonlinear differential constraint: $\dot{x} = f(x) + g(x)u$.

The matrices Q(x) and R(x) are the design parameters representing the state and control weights as functions of x.

Reference 1 shows that the solution to this problem can be obtained by solving a state dependent Riccati equation:

$$\begin{aligned} &A^T(x)P(x) + P(x)A(x) \\ &- P(x)g(x)R^{-1}(x)g^T(x)P(x) + Q = 0 \end{aligned} \tag{7}$$

The state dependent feedback gain can then be computed as:

$$K(x) = R^{-1}(x)g^T(x)P(x) \tag{8}$$

The SDRE nonlinear control law is of the form:

$$u_c = -K(x)x \tag{9}$$

Reference 1 has shown that under rather mild restrictions of Q(x) and R(x), the SDRE control law will globally stabilize the nonlinear dynamic system.

3.3. Quickest Descent Method

The Quickest Descent method discussed in Reference 5 is a Lyapunov function optimizing approach to feedback controller design. In this approach, the control problem is viewed as a function minimization problem in the state space. A descent function, W(x) satisfying certain specified properties is first selected. Control is then chosen to minimize this descent function.

The descent function W(x) is required to be bounded, continuous and continuously differentiable in the region of interest. In addition, the target in the state is required to be contained within the region of interest. Note that these requirements are more restrictive than the choice of a Lyapunov function.

Once the descent function W(x) is selected, the feedback control u(x) is chosen so that W(x) decreases at each state x. If the minimization process is cast as a steepest decent optimization problem, the resulting technique can be termed as the steepest decent control methodology. Reference 5 shows that a more direct approach is to choose the control variables to minimize the time-rate of change of the descent function approach. In this case, the control methodology can be termed as the *Quickest Decent* method.

In the quickest descent method, control is obtained from the optimization problem:

$$\min_{u} \frac{dW(x)}{dt} \text{ or } \min_{u}\left[\frac{\partial W}{\partial x}\{f(x)+g(x)u\}\right] \quad (10)$$

Since the control variable appears linearly in the system dynamics, the optimization problem is meaningful only if the control variables are constrained. The control constraints implemented in the design software are of the form: $|u| \le u_{max}$.

With this, the quickest descent control is given as:

$$\text{If } \left\{\frac{\partial W}{\partial x} g(x)\right\} > 0, u = u_{min} \quad (11)$$

$$\text{If } \left\{\frac{\partial W}{\partial x} g(x)\right\} < 0, u = u_{max} \quad (12)$$

Note that the control is bang-bang. Under the quickest descent method, the control variables will chatter between their limits as the system approaches the minimum of the descent function W(x).

In order to simplify the implementation, the present implementation of the nonlinear control design software assumes that the descent function is always of the form:

$$W(x) = x^T P x \quad (13)$$

with P being a positive definite matrix. In addition to being positive definite, the elements of the matrix P has to be carefully to ensure that the control objectives are satisfied. Thus, selecting P to be a diagonal matrix with positive entries may not result in a desired control system response. The user has to specify a properly populated P matrix to preserve the coupling between the system state variables to achieve the desired response.

3.4. Recursive Back-stepping Method

As observed in Subsection 3.1, the feedback linearization method cancels the nonlinear system dynamics and replaces it with a fixed dynamic system. The main premise behind the recursive back stepping technique[6] is that certain portions of the system nonlinearities are worth preserving. This objective is satisfied by formulating the control problem using the second method of Lyapunov. Since there is no direct way to accomplish this in a direct manner, a recursive procedure is defined for synthesizing control laws. Just as the system nonlinearities were pushed-back from the system states to the inputs, recursive back stepping technique constructs Lyapunov function for the nonlinear dynamic system by stepping-back from the output state variables to the controls. In some respects, this technique bears a strong resemblance to the multiple time-scale feedback linearization design technique.

The recursive back stepping design technique assumes that the model is specified in a triangular form as shown in equations (14) – (16):

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \quad (14)$$

$$\dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 \quad (15)$$

$$\vdots$$

$$\dot{x}_n = f_n(x_1, x_2, \cdots, x_n) + g_n(x_1, x_2, \cdots, x_n)u \quad (16)$$

Each scalar system is stabilized with the following state as the control variable. For example, x_2 is the control variable for x_1- dynamics, x_3 for x_2 - dynamics and so on.

Controllers are synthesized for each scalar dynamic system using second method of Lyapunov. Lyapunov functions are selected based on the nonlinearities that need to be preserved. Quadratic Lyapunov functions are used more often than other types. As mentioned elsewhere, the advantage with this design method is its ability to retain useful nonlinearities in the system. However, the method is useful only if the system has the desired triangular structure.

The user is required to specify the back-stepping structure of the system through a matrix. At each stage of back-stepping, the user also has the freedom to specify the Lyapunov function and the controller rate of convergence. For the sake of simplicity, only quadratic Lyapunov functions are implemented in the current version of the design software.

3.5. Predictive Control

The predictive control methodology[7] has been popular in the process control industry for the past several years. In this technique, a control history that will drive the system states to the desired values at the end of a prediction interval is computed at every sample interval using an optimization algorithm. Most of the techniques described in the literature employ multi-step predictors to implement the controllers. In nonlinear systems, the predictive control technique requires the use of an on-line iterative optimization technique. Since the convergence of nonlinear optimization techniques are not assured, the performance of the predictive control technique cannot be guaranteed.

However, if the nonlinear dynamic system is of the standard form employed in this paper, and if the control problem is cast as a one-step ahead predictive control, the system performance becomes more predictable. This approach is adopted in the present formulation of the nonlinear predictive control technique in the design software.

The control problem is cast as the minimization of the objective function

$$J \underset{=}{\Delta} x_{i+1}^T Q(x,t) x_{i+1} + u_i^T R(x,t) u_i \quad (17)$$

with respect to the control variable u, subject to the differential constraint:

$$\dot{x} = f(x) + g(x)u \quad (18)$$

The differential constraint can be used to eliminate x_{i+1} from the objective function by defining a numerical integration algorithm. If the nonlinear dynamic system is specified in the standard form, the linearity of the differential constraints with respect to the control variables can be preserved if techniques such as Euler's method or the Adams-Bashforth numerical integration scheme[16] are employed. In this case, the necessary conditions for minimizing the objective function will turn-out to be linear with respect to the control variables, and can be solved in closed-form.

The use of the design methods described the preceding subsections will now be applied to the missile autopilot design problem in the following section.

4. Nonlinear Missile Autopilot Design

This section will illustrate the application of the nonlinear control design software for designing autopilots for the missile model given in Section 2.

Autopilot deign will include only the short-period dynamics of the missile, with α and Q as the states. The other two variables, γ and M, are necessary for calculating the acceleration due to gravity and the aerodynamic force and moment. These are treated as auxiliary inputs. Pitch fin deflection, δ is the control variable.

Simulation results are presented for each design method. Each design function requires a separate set of input data. The inputs common to all functions are, time, state vector, auxiliary inputs and the design model. The auxiliary inputs consist of all variables other than the state and control variables that would be necessary to compute the state derivatives.

4.1. Feedback Linearized Autopilot Designs

The first step in this design process is the specification of the manner in which the control variable affects the state variables. In the missile autopilot design example, the fin deflection generates a pitch rate, which in tern generates a change in angle of attack. Although fin deflection also causes a change in the angle of attack, this mechanism is not useful for controlling the missile dynamics. Thus, denoting α as the first state variable, and Q as the second state variable, the *control flow matrix* is:

$$cfmatrix = [1 \;\; 2] \quad (19)$$

The variables γ and M are treated as auxiliary inputs as mentioned earlier.

The feedback linearization function in the design software generates a new dynamic system

$$\ddot{\alpha} = F(\alpha, \dot{\alpha}) + G(\alpha, \dot{\alpha})\delta \quad (20)$$

The differentiation process is carried out numerically in the software, using a state perturbation of magnitude: 10^{-4}. The right hand side of this equation is next replaced with a pseudo control variable. The control laws are then designed using the transformed dynamics. As discussed earlier, this approach enables the use of linear control system design methods. Three distinct designs will be presented in the following. Control law design without time-scale separation will only be demonstrated in this paper.

4.1.1. <u>Robust Pole Placement design</u>

The pole locations for the feedback linearized dynamic system are chosen as:

$$poles = \begin{bmatrix} -14.142 + 14.142i \\ -14.142 - 14.142i \end{bmatrix}$$

A 10 ms integration step size is used for all the simulations unless otherwise specified. State and control responses from the pole placement design are shown in Figure 2 and Figure 3 respectively.

4.1.2. <u>Linear Quadratic design</u>

The state and control weighting matrices for the feedback linearized missile dynamics are chosen as:

$$Q = \begin{bmatrix} 1000 & 0 \\ 0 & 1 \end{bmatrix}, \; R = [0.01]$$

State and control responses from the linear quadratic design are shown in Figure 4 and Figure 5 respectively. Note that the responses from the quadratic design are comparable to the ones from the pole placement design.

4.1.3. <u>Sliding Mode design</u>

The first step in this control system design methodology is that of the specification of the sliding manifold. This is achieved by specifying a set of poles, one less than the order of the system. A best estimate of the lower and upper bounds of the uncertainties on the right hand sides of the system equations are then specified. Finally, the designer has to specify the convergence rate of the system to the sliding manifold, and the boundary layer width. The reader is referred to Reference 3 for a detailed discussion on the sliding-mode design methodology. For the present missile autopilot design example, the pole *P*, the uncertainty bounds δ, the convergence rate η and the boundary layer thickness ε are chosen as:

$P = [-20]$, $\delta = 0$, $\delta_l = 0$, $\delta_u = 0$, $\eta = 20$, $\varepsilon = 0.1$

The state and control responses for the feedback linearized sliding mode autopilot are shown in Figure 6 and Figure 7 respectively.

4.2. SDRE Autopilot

State Dependent Riccati Equation method requires the designer to specify the state weighting matrix Q and the control weighting matrix R. If the designer desires it, the weighting matrices can also be functions of the state. Constant weights are employed in this paper. These are:

$$Q = \begin{bmatrix} 100 & 0 \\ 0 & 10 \end{bmatrix}, R = 10$$

The resulting State and control responses are shown in Figure 8 and Figure 9 respectively.

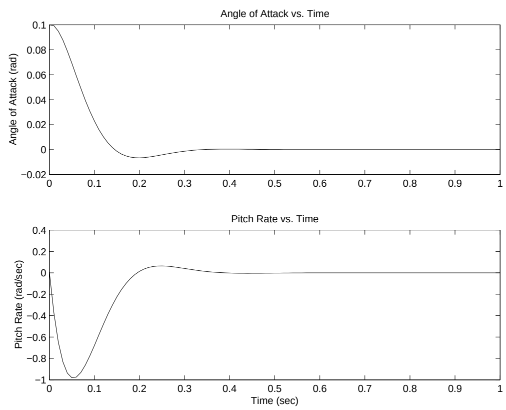

Figure 2 State Responses for Feedback Linearization with Pole Placement Design

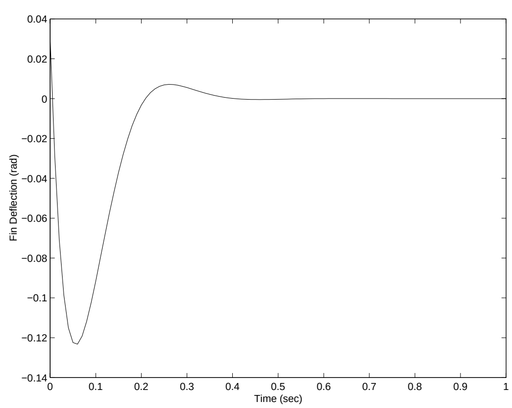

Figure 3 Fin Deflection for Feedback Linearization with Pole Placement Design

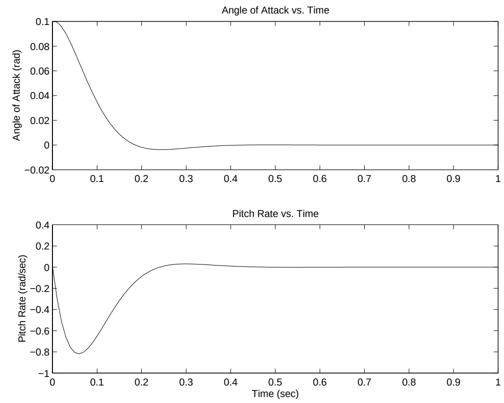

Figure 4 State Responses for Feedback Linearization with Linear Quadratic Design

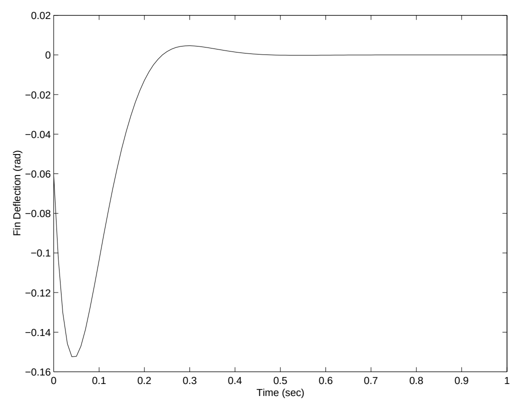

Figure 5 Fin Deflection for Feedback Linearization with Linear Quadratic Design

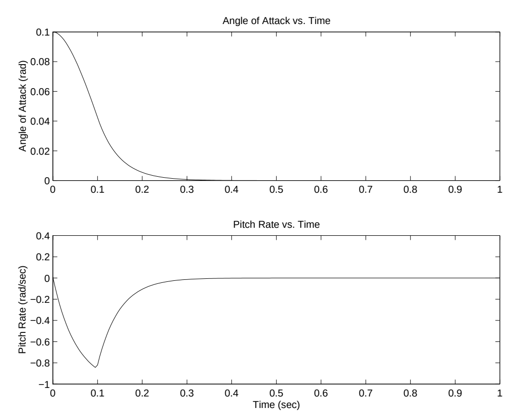

Figure 6 State Responses for Feedback Linearization with Sliding Mode Design

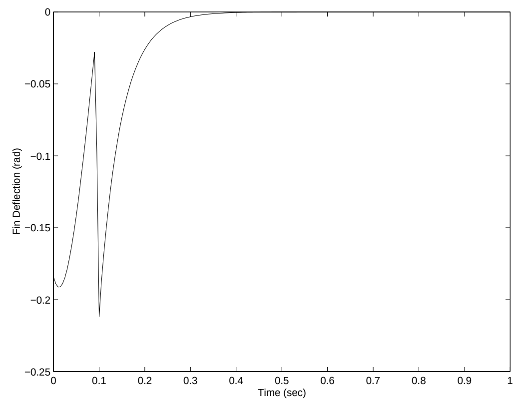

Figure 7 Fin Deflection for Feedback Linearization with Sliding Mode Design

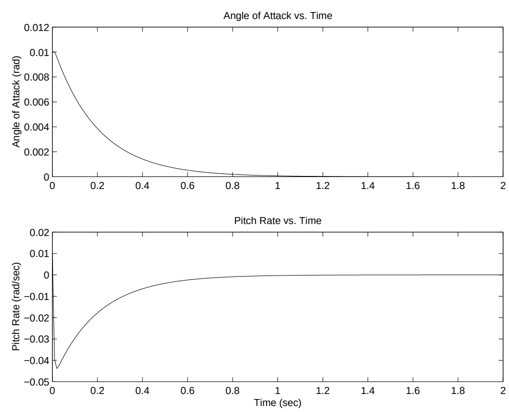

Figure 8 State Responses with SDRE Design

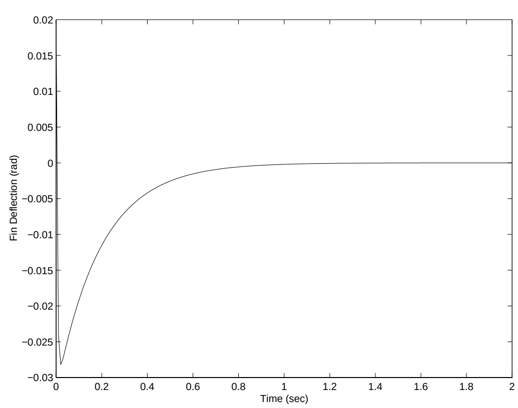

Figure 9 Fin Deflection with SDRE Design

4.3. Quickest Descent Autopilot Design

As formulated in the design software, the only design parameter in the quickest descent method is the weighting matrix in the quadratic descent function. Additionally, the limits on the control variable also needs to be specified. As indicated elsewhere in this paper, the descent function is assumed to be of the form:

$$W(x) = x^T P x$$

For the present research, the design parameters are chosen as:

$$P = \begin{bmatrix} 5000 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

with the fin deflection bounded as: $|u| \le 0.01$. The state and control responses for the quickest descent autopilot are shown in Figure 10 and Figure 11 respectively.

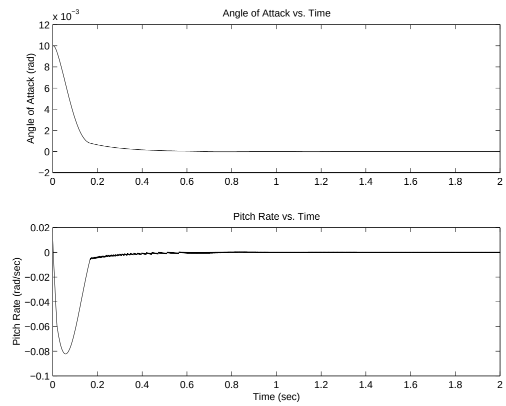

Figure 10 State Responses with Quickest Descent Design

4.4. Recursive Back-Stepping Autopilot Design

Just as in the feedback linearization technique, the recursive back-stepping design technique requires the designer to specify the control distribution matrix. As indicated in Subsection 4.1, this matrix is used to convey the flow of the control variables through the system states. The recursive back-stepping technique uses this matrix to carry out the back stepping procedure.

In addition to the control flow matrix, the designer is required to specify the convergence rate of the Lyapunov function at each step of the back-stepping procedure. The designer iterates on the convergence rates to obtain the desired time-response. For the present example, the convergence rates employed were:

$$D = [20 \quad 20]^T$$

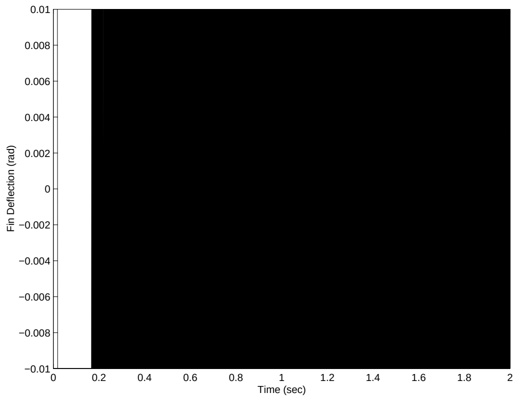

Figure 11 Fin Deflection with Quickest Descent Design

The state and control responses for the recursive back-stepping autopilot are shown in Figure 12 and Figure 13 respectively.

4.5. Predictive Autopilot Design

The design parameters in this technique are the state and control weighting matrices, order of the numerical integration technique and the prediction step size. The following parameters were used in the present case:

$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, R = 2$$

Numerical Integrator order = 2

Prediction Step = 0.001

The design software allows up to 12th order integration. The state and control responses using the predictive autopilot are shown in Figure 14 and Figure 15 respectively.

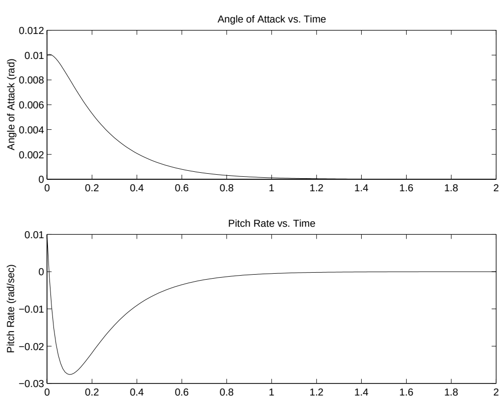

Figure 12 State Responses with Back-Stepping Design

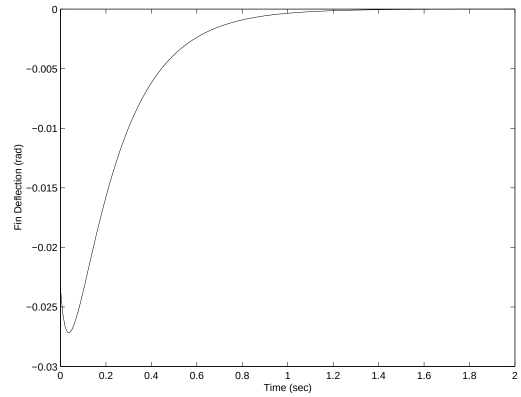

Figure 13 Fin Deflection with Back-Stepping Design

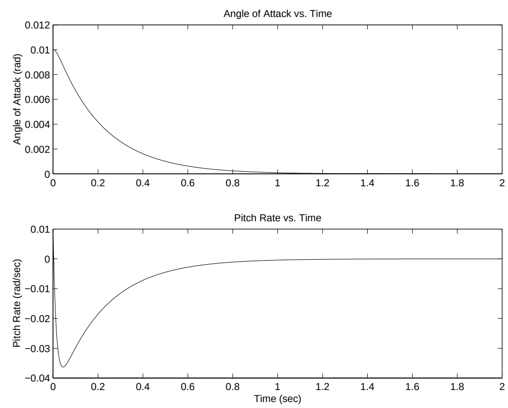

Figure 14 State Responses with Predictive Control Design

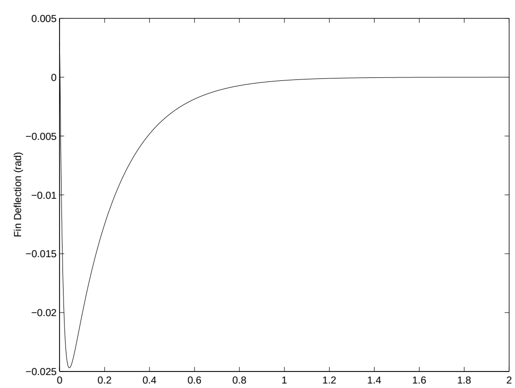

Figure 15 Fin Deflection with Predictive Control Design

The simulation results can be summarized as follows. Angle of attack (α) responses from the feedback linearized regulator designs in cases 4.1.1 and 4.1.2 resemble second order system responses as shown in Figure 2 and Figure 4 respectively. This is natural because feedback linearized dynamics of the system given by (20) is of second order. However, in case of sliding mode design given in Subsection 4.1.3, the response appears to be of first order nature as can be observed in Figure 6.

The responses from SDRE design technique are over-damped for the particular choice of the state and control weighting matrices, as shown in Figure 8. If desired, alternate weighting matrices can be selected to obtain any desired response. Quickest descent design technique yields a bang-bang control law with exponential characteristics as shown in Figure 10. Back-Stepping designs are cascaded first order designs and hence will possess exponential characteristics as shown in Figure 12. Responses from the predictive control design are exponential as shown in Figure 14 and resemble those from SDRE design, although the solution methods are entirely different between these two designs.

5. Summary

Application of a computer aided design software for nonlinear control synthesis is presented in this paper. Nonlinear longitudinal model of a missile was used to illustrate the use of the software. The versatility of the design software was then demonstrated through five different nonlinear designs.

All the discussions in this paper were focussed on the regulator problem. The nonlinear controller design methodologies discussed in this paper can be extended to handle the command tracking servo problem. This investigation is currently in progress

Acknowledgement

This work is performed under the Navy Contract No. N00024-97-C-4178.

References

[1] Cloutier, J. R., D’Souza, C. N. and Mracek, C. P., “Nonlinear regulation and nonlinear $H^{\bullet}$ control via the State Dependent Riccati Equation technique, Part 1: Theory, Part 2: Examples”, *Proceedings of the International Conference on Nonlinear Problems in Aviation and Aerospace*, Daytona Beach, FL, May 1996.

[2] Isidori, A., *Nonlinear Control Systems*, Springer-Verlag, New York, NY, 1985.

[3] Slotine, J.J.E. and Li, W., *Applied Nonlinear Control*, Prentice Hall, Englewood Cliffs, NJ, 1991.

[4] Marino, R., Tomei, P., *Nonlinear Control Design*, Prentice-Hall, New York, NY, 1995.

[5] Vincent, T. L. and Grantham, W. J., *Nonlinear and Optimal Control Systems*, John Wiley and Sons, Inc., NY, 1997.

[6] Krstic, M., Kanellakopoulos, I. and Kokotovic, P., *Nonlinear and Adaptive Control Design*, John Wiley and Sons, New York, NY, 1995.

[7] *The Control Handbook*, Levine, W. S. (Editor), CRC Press, Boca Raton, FL, 1996.

[8] Mracek, C.P. and Cloutier, J. R., “Missile longitudinal autopilot design using the State Dependent Riccati Equation method”, *Proceedings of the International Conference on Nonlinear Problems in Aviation and Aerospace*, Daytona Beach, FL, May 1996.

[9] Anon, *MATLAB User’s Manual*, The MathWorks, Inc., Natick, MA, 1998

[10] Anon, *SIMULINK User’s Manual*, The MathWorks, Inc., Natick, MA, 1998

[11] Kailath, T., *Linear Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[12] Kautsky, J., Nichols, N. K., and Van Dooren, P., “Robust Pole Assignment in Linear State Feedback”, *International Journal of Control*, Vol. 41, No. 5, 1985, pp. 1129-1155.

[13] Bryson, A. E., and Ho, Y. C., *Applied Optimal Control*, Hemisphere, New York, NY, 1975.

[14] Menon, P. K., Badgett, M. E., Walker, R. A., and Duke, E. L., "Nonlinear Flight Test Trajectory Controllers for Aircraft", *Journal of Guidance, Control and Dynamics*, Vol. 10, Jan. - Feb. 1987, pp. 67 - 72.

[15] Sharma, V. and Zhao, Y., “Dynamic Optimal Linearization of Nonlinear Systems”, *Proceedings of the American Control Conference*, June 1993, pp. 1196-1197, San Francisco, CA.

[16] Gerald, C. F., *Applied Numerical Analysis*, Addison-Wesley, Reading, MA, 1978.