CHAPTER 9

ADAPTIVE CONTROL WITH APPLICATION TO MINIATURE AERIAL VEHICLES

9.1 INTRODUCTION

In this chapter, we build on nonlinear control methods introduced in Chapter 5 (Modern Control Techniques) by including ideas from adaptive control. We are motivated by the fact that many methods for control rely on an accurate model of the aircraft dynamics. The dynamic inversion methods introduced in Chapter 5 are an example, where an accurate model is required to achieve a desired reference model response. A highly accurate model is very difficult to achieve in practice. Allowing that model to be corrected and improved while the controller is in operation could address this issue.

A controller that automatically adjusts the feedforward or feedback gains is an *adaptive* controller. The basic idea that an adaptive controller could be adjusted automatically to maximize performance is nearly as old as automatic control itself. One of the early important examples of adaptive controllers being applied to flight control was an experimental controller tested as part of the NASA X-15 program in 1967 (Dydek et al., 2010). Test pilot Michael Adams was killed during one such test. Although the adaptive controller worked as intended during many flights, for this fatal flight the adaptive controller was partially implicated in the accident. So, although adaptation has the potential to increase reliability and performance, it clearly also has potential pitfalls. In the years that have followed, a large number of new potential adaptive control methods have been proposed and utilized for flight control. However, none has achieved anywhere near the level of trust in the aerospace community as nonadaptive flight control.

One method to increase trust in an adaptive control approach is to utilize methods that have an associated mathematical proof of stability or at least boundedness of system signals. The existing examples tend to be highly conservative, to the point where the limits implied are of little practicality. However, the existence of a stability proof is clearly better than the absence of one. The methods discussed in this chapter are of this class. Because the approach focused on in this chapter has been applied to more than 1000 flights of many different types of miniature aircraft over twelve years of diverse flight operations, this method represents one of the most heavily verified adaptive flight control methods.

Although any control system could potentially benefit from an adaptive control perspective, unmanned aerial vehicle flight control can obtain several specific advantages that are uniquely important. The lack of a human pilot means that the ability of the human controller to provide adaptation is often not available. It may be possible to regain some of the reliability benefit by allowing adaptation within the control system itself. This need for adaptation may arise from either uncertainty in the vehicle behavior (itself perhaps typically larger for aircraft that are unmanned) or damage/faults.

In this chapter, deviations from a reference model are utilized to enable a dynamic inversion approach to adjust the controller response over time. Due to the fact that the controller gains are changing, we can recognize this is an adaptive control method. The basis of this gain adjustment depends upon a comparison between expected (reference) and actual responses. Due to this last property, this is also an example of model reference adaptive control (MRAC).

Within the realm of adding adaptation to a dynamic inversion flight control law, we will illustrate that we have effectively formulated a real-time nonlinear curve-fitting problem. We will then explore the specific case of utilizing artificial neural networks (NNs) as a proven method to perform this fitting.

The sections that follow will then cover a number of important realistic implementation issues that arise when attempting to implement an adaptive flight controller. This will start with the issue of limited authority (including input saturation) as well as accounting for a multiloop architecture that is common for automated guidance and flight control systems. For example, it is not unusual to see an attitude controller used by a position tracking guidance system. If one or both of these interconnected feedback control systems are adaptive, then this must be accounted for in the design.

A detailed example of adaptive guidance and flight control of a quadrotor and a small unmanned helicopter follows. This will include a discussion of several important implementation considerations, simulation results (using a model from Chapter 8), and flight test results. These implementation considerations will include observers for unmeasured states (Chapter 6) and digital control considerations from Chapter 7.

9.2 MODEL REFERENCE ADAPTIVE CONTROL BASED ON DYNAMIC INVERSION

In Chapter 5.8, the method of dynamic inversion was described for control of nonlinear systems. For this method, the true dynamics of the aircraft were replaced

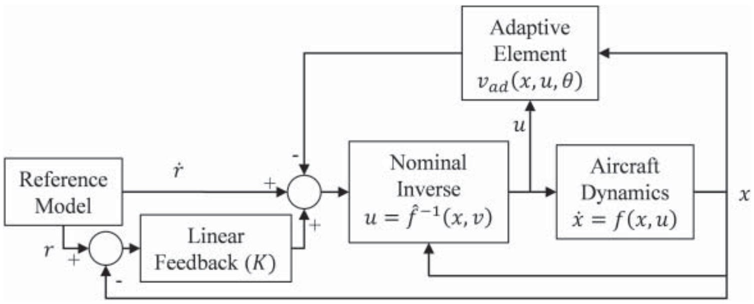

Figure 9.2-1 Use of adaptation to correct model error in a dynamic inversion controller.

by desired dynamics through feedback. In this section, we look at addressing one of the most significant issues with this approach: the availability of an accurate model of the aircraft dynamics. Here, we attack the problem by refining the dynamics model used in the dynamic inversion controller in real time. This means that gains will be changing within the controller in response to changes in the aircraft (or differences between the true aircraft and the assumed model).

An illustration of such a controller is shown in Figure 9.2-1. Here, the architecture similar to that shown in Figure 5.8-5 is simplified to the special case of full state feedback and an adaptive component is added. This architecture includes a "nominal" controller based on the best available information about the aircraft dynamics embodied in the design. The adaptive element then works on the error in this nominal model and provides a correction. This correction can be an arbitrarily accurate correction given sufficient training information, sufficient inputs to the correction block in the form of states and inputs, and sufficient power in the adaptive element to curve fit this model error. Let the plant be described by

$$\dot{x} = f(x, u), \qquad (9.2\text{-}1)$$

with state $x \in R^n$ and $u \in R^m$. The nominal dynamic inverse needs to be of a similar form, but we distinguish here between the true plant dynamics in (9.2-1) and a known approximation to it of the form

$$v = \hat{f}(x, u), \qquad (9.2\text{-}2)$$

with *pseudocontrol* signal $v \in R^n$ as the "desired" $\dot{x}$. These dynamics are then inverted to find the plant input

$$u = \hat{f}^{-1}(x, v), \qquad (9.2\text{-}3)$$

where the existence and uniqueness of this inverse are important points for further consideration below. For the moment, we will assume that there exists a set of inputs that will achieve our desired $\dot{x}$ at the current state x and limit ourselves to the issue that $f \neq \hat{f}$ in general.

When one introduces the adaptive element in the controller, it will ideally be a function of both the states and inputs of the original system [Equation (9.2-1)]. The total pseudocontrol input used in (9.2-3) will now have three components:

$$v = \dot{r} + Ke - v_{ad}(x, u, \theta), \tag{9.2-4}$$

where the $\dot{r}$ term represents the desired response, the Ke term a linear feedback on the tracking error, and v_{ad} the new adaptive signal. In addition to all original system states and inputs, the latter includes internal parameters θ.

By including v_{ad} in this way, a clear definition of v_{ad} emerges as the model error in the nominal controller design. To see this, we consider the reference model tracking error dynamics (e). This can be written as

$$\begin{aligned} e &= r - x \\ \dot{e} &= \dot{r} - \dot{x} \end{aligned} \tag{9.2-5}$$

Substituting for $\dot{x}$ using (9.2-1), (9.2-3), and (9.2-4) we find

$$\dot{e} = \dot{r} - f(x, u) = \dot{r} - \hat{f}(x, u) - f(x, u) + \hat{f}(x, u) \tag{9.2-6}$$

$$\dot{e} = -Ke + v_{ad}(x, u, \theta) - f(x, u) + \hat{f}(x, u) \tag{9.2-7}$$

We now have a useful definition for what we would like our adaptive controller to converge to. We seek to achieve the chosen linear tracking error dynamics (K) by exact cancellation of the model error through adjustment of parameters θ:

$$\dot{e} = -Ke \tag{9.2-8}$$

$$v_{ad}(x, u, \theta) = f(x, u) - \hat{f}(x, u) \tag{9.2-9}$$

We see explicitly in (9.2-9) that this is an exercise in nonlinear curve fitting. Specifically, we need to achieve a curve fit of the model error as a function of state and input.

This is a particularly powerful adaptive controller architecture in the sense that it can, in principle, adjust internal parameters in order to achieve desired dynamics for an unknown nonlinear system. As a practical matter, it is sufficient that the controller is able to fully correct for errors in the subset of the input and state space that the aircraft has experienced. We might also be satisfied in many cases with being able to adjust only to the recently experienced subset of the input and state space. That is, it may be acceptable for the controller to "forget" information about aspects of the aircraft no longer prevailing. One criterion for this being acceptable would be the ability of the controller to respond sufficiently fast when encountering new information. In this way, we distinguish between *long-term learning*, where we achieve improved performance if we return to a part of the state space that the vehicle has experienced before, and *short-term learning*, where the controller must readapt upon returning to previous portions of the state space.

In the next section, a specific method to achieve short-term learning is presented based on the concept of the artificial neural network. This construct has been used successfully in many curve-fitting applications. Within this adaptive control architecture, it could be referred to as neural network adaptive control.

9.3 NEURAL NETWORK ADAPTIVE CONTROL

A method that can be used to perform the curve fitting motivated in the previous section (9.2-9) is the artificial NN, of which there are many varieties. This section includes a description of several of the most common NN formulations utilized in adaptive control. For starters, a useful distinction for classifying these NN formulations is *parametric* NN vs. *nonparametric* NN (Lewis et al., 1999), as shown in Figure 9.3-1.

In the parametric NN, model error has a known basis function with one or more unknown parameters. For example, the functional form of the effectiveness of an airplane rudder may be known to sufficient accuracy, but the designer may want to account for an unknown scale and/or bias error. In this case, the adaptive signal can be written as

$$v_{ad}(x, u, \theta) = \sum_{i=1}^{Np} [\theta_i \Delta f_i(x, u)], \tag{9.3-1}$$

where Np is the number of basis functions and $\Delta f_i(x, u)$ are the basis functions. Inclusion of a constant basis function would allow for the correction of a bias error. The inclusion of a basis function proportional to the rudder input would allow for the correction of a scale factor error.

In the nonparametric NN the designer does not explicitly include knowledge about the functional form of the model error. Here, sufficient parameterization is included to allow the NN to perform the curve fit to desired accuracy, even for a nonlinear system. One way to do this is inspired by how biological NNs work in the brain. In practice, the complexity level needed is small compared to that found in biology.

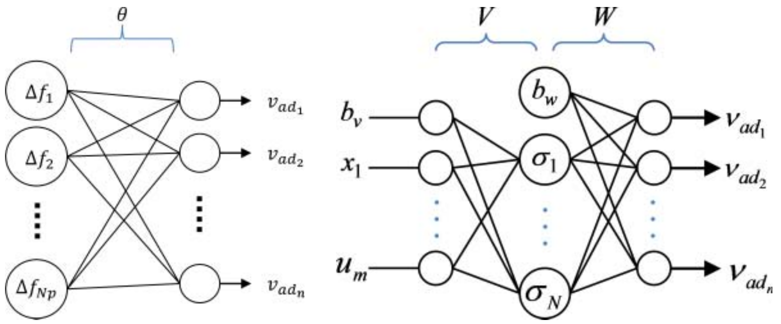

Figure 9.3-1 Parametric (left) and nonparametric (right) NN structures.

Specifically, biology inspires the idea of utilizing interconnections between *neurons* (Figure 9.3-1). In this case, the single hidden layer (SHL) NN is illustrated (Lewis et al., 1996, 1997). That is, there is one/single layer of neurons between the input layer and the output layer. Another important approach utilizes radial basis functions (RBFs) (Lavretsky and Wise, 2013). For the SHL, a *squashing function* is utilized at the hidden layer neurons, such as

$$\sigma_j(z) = \frac{1}{1 + e^{-a_j z}}, \tag{9.3-2}$$

for $j = 1, \ldots, N$, with the value a_j chosen to be distinct for each j, where N is the number of hidden layer neurons. Using this, the full SHL NN can be written as

$$v_{ad}(x, u, \theta) = W^T \sigma(V^T \overline{x}), \tag{9.3-3}$$

where $W \in R^{N \times n}$ are the output weights, $V \in R^{n+m+1 \times N}$ are the input weights, and $\overline{x} \in R^{n+m+1}$ is the input to the NN, normally the states x, plant inputs u, and a bias. The complete set of the elements of W and V are the NN adjustment parameters.

Universal Approximation Theorem

One question that comes up is the choice of the number of hidden layer neurons (N). With a very small number, it is clear by inspection that there may be insufficient capability to curve fit a reasonable model error function. Adding additional hidden layer neurons will certainly improve this curve-fitting capability. However, there will clearly be diminishing returns if the number increases beyond some level. In practice, it has been found that the incremental benefit is not particularly great beyond $N = 5$ for the types of flight control problems addressed here, which is the number used in all work included in this chapter.

The *universal approximation theorem* for the SHL NN tells us that we can bound the fitting error within a compact set of states and plant inputs. Significantly, it can be achieved for any chosen error bound by adding additional middle layer neurons (N) to the NN (Cybenko, 1989).

There are many examples in the literature of proofs of boundedness of all system signals for these types of controllers. These proofs typically involve a Lyapunov function candidate and invoking the universal approximation theorem to show that the Lyapunov candidate decreases outside of a compact set. This ensures convergence to a set containing zero tracking error (Yesildirek and Lewis, 1995; Kim and Lewis, 1998; Johnson and Calise, 2003; Lavretsky and Wise, 2013).

Very often, considerations of the proof of boundedness will inspire the adaptive laws themselves. A typical approach to training the NN that comes directly from a proof of boundedness is (Johnson and Calise, 2003)

$$\dot{W} = -[(\sigma - \sigma' V^T \overline{x}) e^T + \lambda \|e\| W] \Gamma_W \tag{9.3-4}$$

$$\dot{V} = -\Gamma_V [\overline{x} e^T W^T \sigma' + \lambda \|e\| V], \tag{9.3-5}$$

where Γ_W and Γ_V are appropriately dimensioned diagonal matrices of learning rates. The matrix σ' is the gradient of σ. The e-modification scalar $\lambda > 0$ is necessary for the associated boundedness theorem proof. Note the important role of tracking error (e) here. When tracking error is zero, these parameters do not change.

Example 9.3-1: Neural Network Adaptive Controller In this example, a first-order system with significant model error is controlled using a SHL NN to provide adaptation. This corresponds to pitch-rate control of an aircraft with a significant unknown nonlinearity in pitch damping using the elevator. For the purposes of this example, the true dynamics are taken as

$$\dot{q} = M_q q + M_{\delta_e} \delta_e + \sin q, \tag{9.3-6}$$

where the last term, $\sin q$, represents an unanticipated fault as is entirely unknown during controller design. As a result, the approximate dynamic inverse is missing this term,

$$\delta_e = \frac{1}{M_{\delta_e}} (v - M_q q), \tag{9.3-7}$$

with pseudocontrol input corresponding to the derivative of pitch rate.

The following MATLAB code is able to simulate the results of such a controller:

```
% parameter choices
dt = 0.05;  tfinal  = 50;  tswitch = 5;
Mdelta = −10;  Mq = −1;
K = −1;
gammaw = 1;  gammav = 10;
lambda = 0.01;
nmid = 5;  nin  = 3;
amin = 0.01;  amax = 10;

% precompute activation potentials
a = zeros(nmid,1);
for i=1:nmid−1,
   a(i) = tan( atan(amin) + ( atan(amax) − atan(amin) )*(i+1)
          /nmid );
end;

% preallocate arrays
points = tfinal/dt + 1;
t = zeros(points,1);  r = zeros(points,1);
x = zeros(points,1);  u = zeros(points,1);
w = zeros(points,nmid);      wdot = zeros(1,nmid);
v = zeros(points,nmid*nin);  vdot = zeros(1,nmid*nin);
rdot = zeros(1,1);  xdot  = zeros(1,1);
Wdot = zeros(1,nmid);    W = zeros(1,nmid);
Vdot = zeros(nmid,nin);  V = zeros(nmid,nin);
xbar = zeros(nin,1);
sig = zeros(nmid,1);  sigp  = zeros(nmid,nmid);
```

```
for i=1:points,
    t(i)=(i−1)*dt;

    % external input
    if mod(t(i),2*tswitch)<tswitch, externalInput = 1;
    else                            externalInput = −1;
    end;

    % reference model
    rdot = −K*( externalInput − r(i,1) );

    % error signals
    e = r(i,1) − x(i,1);

    % NN inputs
    if i>1, oldu = u(i−1);
    else oldu = 0; end;
    xbar(1) = 1;
    xbar(2) = x(i,1);
    xbar(3) = oldu;

    % get weights from state vector
    W = w(i,:);
    for j=1:nmid,
    V(j,:) = v(i,(j−1)*nin+1:j*nin);
    end;

    % adaptive controller
    vx = V*xbar;
    for j=1:nmid−1,
       ez = exp( −a(j)*vx(j) );
       sig(j) = 1/( 1 + ez );
       sigp(j,j) = a(j)*ez*sig(j)*sig(j);
    end;
    sig(nmid) = 1;
    vad = W*sig;
    u(i) = ( rdot(1) − K*e − vad − Mq*x(i,1) )/Mdelta;

    % plant model
    xdot = sin( x(i,1) ) + Mdelta*u(i) + Mq*x(i,1);

    % learning law
    Wdot = −gammaw*( e*( sig' − xbar'*V'*sigp ) + lambda*norm
            ( e )*W );
    Vdot = −gammav*( sigp*W'*e*xbar'            + lambda*norm
            ( e )*V );

    % put NN update in a vector
    wdot = Wdot;
    for j=1:nmid,
    vdot((j−1)*nin+1:j*nin) = Vdot(j,:);
    end;
    % numerically integrate
    if i==1,
    x(i+1,:) = x(i,:) + xdot*dt;
    r(i+1,:) = r(i,:) + rdot*dt;
    w(i+1,:) = w(i,:) + wdot*dt;
    v(i+1,:) = v(i,:) + vdot*dt;
    elseif i<points,
    x(i+1,:) = x(i,:) + ( 1.5*xdot - 0.5*oldxdot )*dt;
    r(i+1,:) = r(i,:) + ( 1.5*rdot - 0.5*oldrdot )*dt;
    w(i+1,:) = w(i,:) + ( 1.5*wdot - 0.5*oldwdot )*dt;
    v(i+1,:) = v(i,:) + ( 1.5*vdot - 0.5*oldvdot )*dt;
    end;
    oldxdot = xdot;  oldrdot = rdot;
    oldwdot = wdot;  oldvdot = vdot;
end;
```

The state history when the desired response is an aggressive square wave is shown in Figure 9.3-2. Here, we see that the controller performs better on each successive attempt at this maneuver, even though there was extreme error in the original model. In fact, the pitch-rate error is actually vanishing at these desired rates. The control effort (usage of elevator) is reasonable, as also seen in the figure.

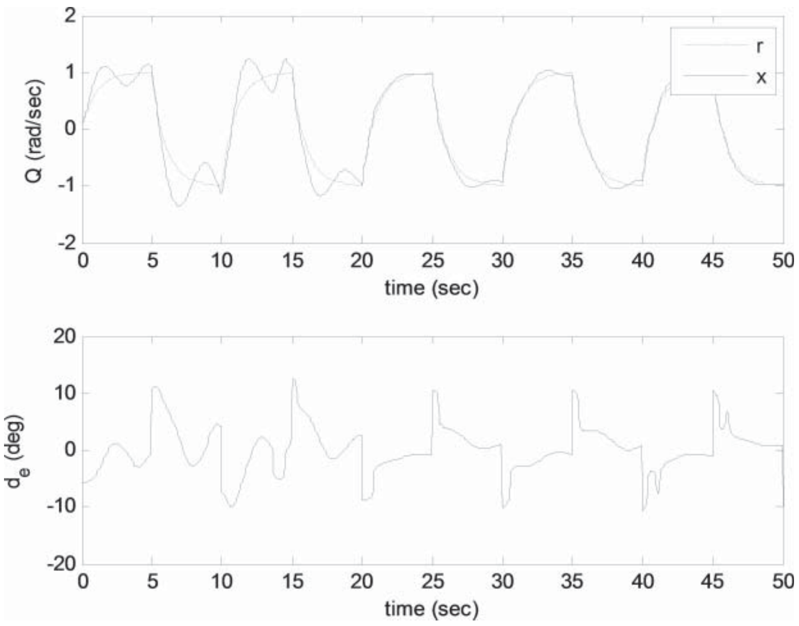

Figure 9.3-2 State history (top) and input history (bottom) for Example 9.3-1.

Figure 9.3-3 shows the adaptive parameters. That is, all of the elements of V and W are plotted. After an initial transient, we see that the weights are settling in to nearly constant values, even though the vehicle is maneuvering rapidly. This, along with the low tracking error achieved, indicates successful adaptation.

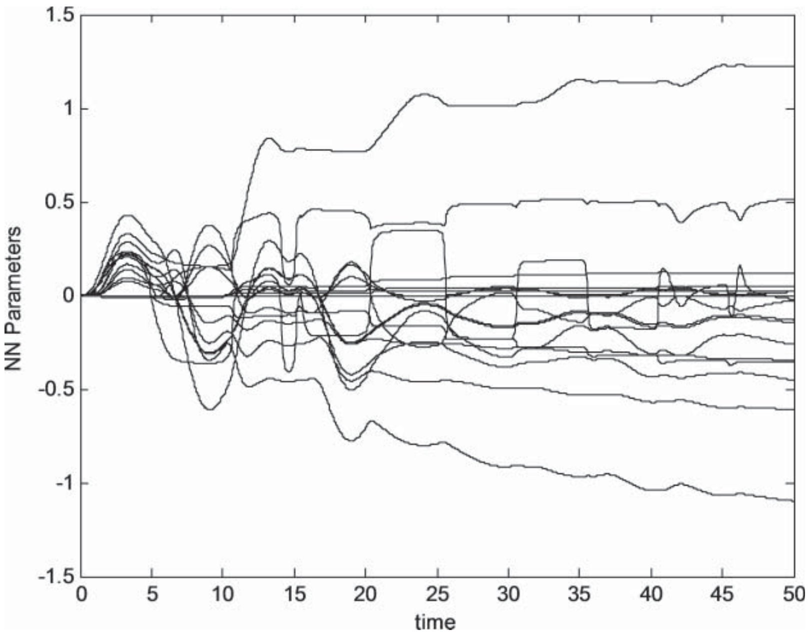

Figure 9.3-3 Neural network parameters for Example 9.3-1. ■

9.4 LIMITED AUTHORITY ADAPTIVE CONTROL

Input saturation and input rate saturation present a significant problem for adaptive control, even more so than for nonadaptive control (see Section 7.5 for a discussion of saturation and linear controllers). Saturation violates any *affine in control* assumption, which is common in the literature. It also violates the assumption that the sign of the effect of the control is locally known and nonzero, since the effect of any attempted additional control input is zero once saturation is encountered. The behavior is analogous to integrator windup in a linear controller but worse. Adaptation may attempt to correct for the unexpected effects of saturation, driving the system further into saturation. Like integrator windup, these effects can dramatically reduce the domain of attraction of a commanded equilibrium, in the worst case precluding any command or disturbance that would involve an input saturating for more than a brief period. However, it is possible for an adaptive law to function properly (as designed) even during input saturation, depending on the adaptation method used.

The notion that adaptation can continue even when the system is encountering input saturation is significant. An approach that halts adaptation when saturation is encountered, in order to prevent issues with input saturation, would be removing the benefit of adaptation during what is arguably the most important period to ensure reliable adaptation. That is because saturation is indicative of a transient response and/or a return from a higher gain condition. In either case, the flight control system designer would clearly prefer adaption to proceed at full effectiveness.

Another issue for adaptive control relates to the control of systems in *cascade*. This is where the state of one subsystem becomes the input to a second subsystem. An important example in flight control is the use of attitude to control velocity. To allow adaptation to take place for both of these interconnected subsystems, it is necessary to account for the response of the attitude subsystem when enabling adaptation for the velocity control (and vice versa). If the attitude subsystem is not able to respond fully for any reason, then the control system designer wants to ensure that the velocity subsystem adaptation can still proceed without issue.

Pseudocontrol Hedging

Pseudocontrol hedging (PCH) is a method that can be applied to a dynamic inversion of a *limited authority* adaptive controller to achieve the goals outlined above. It involves feeding back expected system response to the reference model in order to allow adaptation to effectively proceed regardless of system response. This method requires finding the difference between commanded and predicted pseudocontrol and using this to modify the reference model response in such a way that (9.2-7) is satisfied. If (9.2-7) is still satisfied, then adaptive laws derived using it, such as (9.3-4 and 9.3-5), remain valid.

In developing a PCH design, the control system designer can select which system characteristics the system will attempt to correct for with adaptation. In this context, we expect the designer does not want the controller to attempt to correct for absolute input saturation or the response of subsystems that have their own adaptive process.

To formalize this in the context of input saturation only, let us introduce variable u_{cmd} to be the unsaturated input and u to be the saturated (limited) input to the system. The PCH controller is illustrated in Figure 9.4-1. Here, the PCH block is shown that computes the hedge signal

$$v_h = v - \hat{f}(x, u) \tag{9.4-1}$$

The reference model motion is modified by PCH in a specific way as described here. For the case where the original reference model is of the general form

$$\dot{r} = f_{rm}(r) \tag{9.4-2}$$

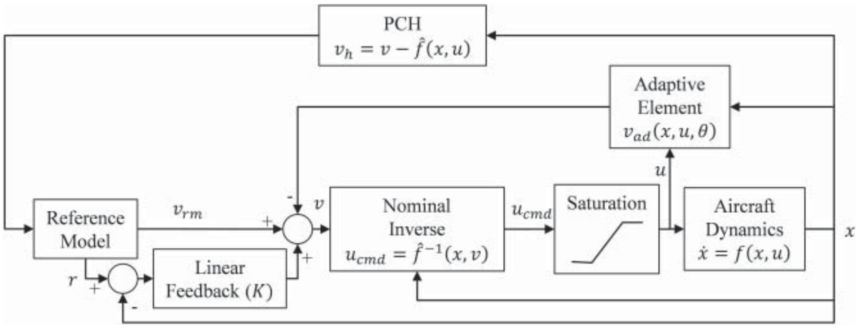

Figure 9.4-1 Use of PCH to address input saturation.

the modified form with PCH becomes

$$\dot{r} = f_{rm}(r) - v_h \tag{9.4-3a}$$

$$v_{rm} = f_{rm}(r), \tag{9.4-3b}$$

where v_{rm} is the feedforward signal from the reference model. Note that the PCH signal (v_h) is integrated before it affects reference model output.

With PCH, if we rederive the reference model tracking error dynamics (e), we find

$$\dot{e} = -Ke + v_{ad}(x, u, \theta) - f(x, u) + \hat{f}(x, u), \tag{9.4-4}$$

which is the same as (9.2-7) and one sees that u_{cmd} does not appear. As a result, adaptation is able to function properly and attempt to correct for the desired form of model error: $f(x, u) - \hat{f}(x, u)$ as before.

Example 9.4-1: Neural Network Adaptive Controller with PCH In this example, a first-order system with significant model error from Example 9.3-1 is controlled using a SHL NN to provide adaptation. PCH is utilized to account for input saturation in the design. The true dynamics are once again taken as

$$\dot{q} = M_q q + M_{\delta_e}\delta_e + \sin q, \tag{9.4-5}$$

where the last term is considered entirely unknown. The input δ_e is limited in magnitude. The approximate dynamic inverse is missing the unknown term and was designed ignoring the possibility of a saturated input,

$$\delta_{ecmd} = \frac{1}{M_{\delta_e}}(v - M_q q) \tag{9.4-6}$$

The elevator command from (9.4-6) is limited to the known limit before being sent to the elevator actuator. PCH is used to allow adaptation to proceed correctly even if the elevator command encounters saturation.

The following MATLAB code is able to simulate the results of such a controller:

```
% parameter choices
dt = 0.05;  tfinal  = 50;  tswitch = 5;
Mdelta = -10;  Mq = -1;
u_max = 0.1;
K = -1;
gammaw = 1;  gammav = 10;
lambda = 0.01;
nmid = 5;  nin  = 3;
amin = 0.01;  amax = 10;

% precompute activation potentials
a = zeros(nmid,1);
for i=1:nmid-1,
   a(i) = tan( atan(amin) + ( atan(amax) - atan(amin) )*(i+1)
         /nmid );
end;
```

```
% preallocate arrays
points = tfinal/dt + 1;
t = zeros(points,1);  r = zeros(points,1);
x = zeros(points,1);  u = zeros(points,1);
w = zeros(points,nmid);      wdot = zeros(1,nmid);
v = zeros(points,nmid*nin);  vdot = zeros(1,nmid*nin);
rdot = zeros(1,1);  xdot  = zeros(1,1);
Wdot = zeros(1,nmid);    W = zeros(1,nmid);
Vdot = zeros(nmid,nin);  V = zeros(nmid,nin);
xbar = zeros(nin,1);
sig = zeros(nmid,1);  sigp  = zeros(nmid,nmid);

for i=1:points,
    t(i)=(i−1)*dt;

    % external input
    if mod(t(i),2*tswitch)<tswitch, externalInput = 1;
    else                            externalInput = −1;
    end;

    % reference model
    vrm = −K*( externalInput − r(i,1) );

    % error signals
    e = r(i,1) − x(i,1);

    % NN inputs
    if i>1, oldu = u(i−1);
    else oldu = 0; end;
    xbar(1) = 1;
    xbar(2) = x(i,1);
    xbar(3) = oldu;

    % get weights from state vector
    W = w(i,:);
    for j=1:nmid,
    V(j,:) = v(i,(j−1)*nin+1:j*nin);
    end;

    % adaptive controller
    vx = V*xbar;
    for j=1:nmid−1,
       ez = exp( −a(j)*vx(j) );
       sig(j) = 1/( 1 + ez );
       sigp(j,j) = a(j)*ez*sig(j)*sig(j);
    end;
    sig(nmid) = 1;
    vad = W*sig;
    vv = vrm − K*e − vad;
    u_cmd = ( vv − Mq*x(i,1) )/Mdelta;
    if abs(u_cmd) > u_max,  u(i) = u_max*sign(u_cmd);
    else                    u(i) = u_cmd;
    end
    % plant model
    xdot = sin( x(i,1) ) + Mdelta*u(i) + Mq*x(i,1);

    % hedge signal
    vh = vv − ( Mdelta*u(i) + Mq*x(i,1) );

    % reference model modified
    rdot = vrm − vh;

    % learning law
    Wdot = −gammaw*( e*( sig' − xbar'*V'*sigp ) + lambda*norm
            ( e )*W );
    Vdot = −gammav*( sigp*W'*e*xbar'             + lambda*norm
            ( e )*V );

    % put NN update in a vector
    wdot = Wdot;
    for j=1:nmid,
    vdot((j−1)*nin+1:j*nin) = Vdot(j,:);
    end;

    % numerically integrate
    if i==1,
    x(i+1,:) = x(i,:) + xdot*dt;
    r(i+1,:) = r(i,:) + rdot*dt;
    w(i+1,:) = w(i,:) + wdot*dt;
    v(i+1,:) = v(i,:) + vdot*dt;
    elseif i<points,
    x(i+1,:) = x(i,:) + ( 1.5*xdot − 0.5*oldxdot )*dt;
    r(i+1,:) = r(i,:) + ( 1.5*rdot − 0.5*oldrdot )*dt;
    w(i+1,:) = w(i,:) + ( 1.5*wdot − 0.5*oldwdot )*dt;
    v(i+1,:) = v(i,:) + ( 1.5*vdot − 0.5*oldvdot )*dt;
    end;
    oldxdot  = xdot;  oldrdot = rdot;
    oldwdot  = wdot;  oldvdot  = vdot;
end;
```

The state history when the desired response is an aggressive square wave is shown in Figure 9.4-2. Here, we see that the controller still performs better on each successive attempt at this maneuver. This is true even though there is extreme error in the model and input saturation. As seen in the figure, the input is in fact at saturation for much of the response. Adaptation proceeds effectively despite significant input saturation (Figure 9.4-3). ■

Adaptive Control for Cascaded Systems

The idea behind PCH can be expanded to allow systems in cascade to be controlled effectively. Here, adaptation in one subsystem is protected from interference by another subsystem which exists "downstream." Here, this architecture is specifically applied to the case of controlling the position/velocity of an aircraft (one subsystem) utilizing attitude control (second subsystem). Clearly, what happens in terms of

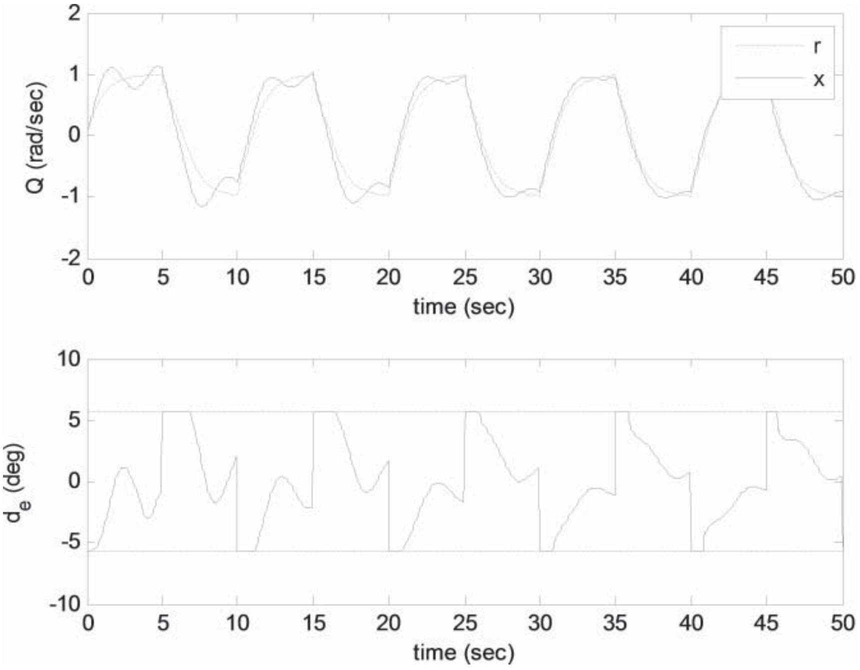

Figure 9.4-2 State history (top) and input history (bottom) for Example 9.4-1.

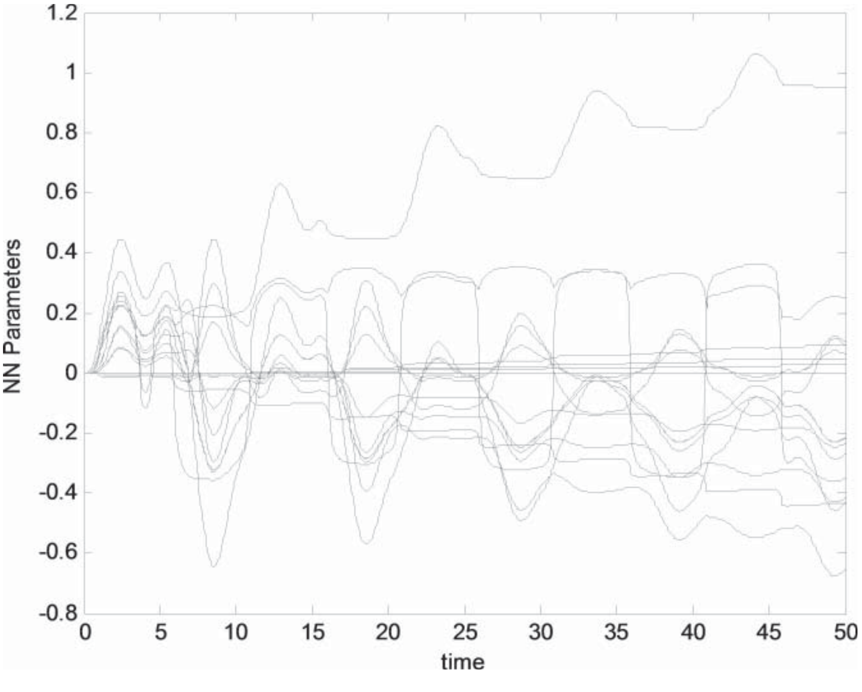

Figure 9.4-3 Neural network parameter for Example 9.4-1.

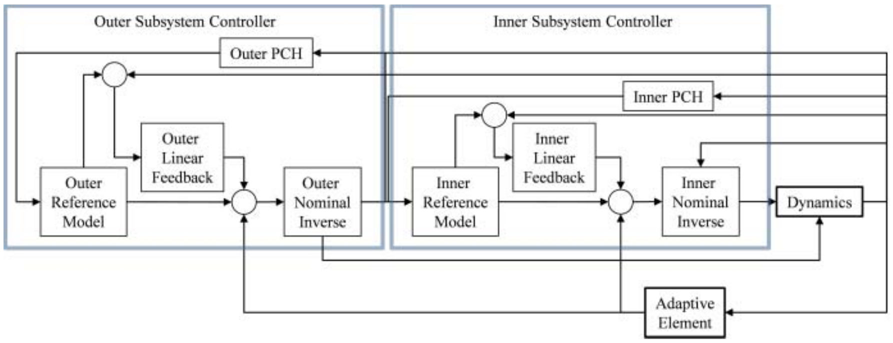

Figure 9.4-4 Use of PCH to address systems in cascade (for clarity, full state/input feedback is not shown to all elements).

attitude on an aircraft will have a dramatic effect on velocity. Conceptually, PCH will be utilized to protect the outer loop (velocity) loop adaptation from the response of the inner loop (attitude). This architecture is shown in Figure 9.4-4.

Note that a single NN can provide the adaptation necessary to correct for all six degrees of freedom. That is, it can correct for model error in all three linear velocity components and all three angular velocity components. This allows the controller to utilize both direct force/moment controls as well as attitude changes to achieve desired accelerations. The design choices here would be specified by the two dynamic inversion blocks. In the following section, this architecture will be utilized as the basis for a small helicopter adaptive guidance and flight control system suitable for helicopters and multirotors/quadrotors. With minor modification, it can be utilized for airplanes.

9.5 NEURAL NETWORK ADAPTIVE CONTROL EXAMPLE

One early use of NN adaptive flight control was on the NASA X-36 unmanned aircraft (Brinker and Wise, 2001; Calise et al., 2001). Here, the approach was successfully applied to the attitude control of an unmanned airplane. There have been a variety of tests of similar adaptive controllers in a variety of programs in the years since. A notable example is the use on the Boeing Joint Direct Attack Munition (JDAM), for which operational benefits have been claimed (Sharma et al., 2000).

This section contains a detailed design of an adaptive guidance and flight control system applied to rotorcraft flight control, including helicopters (Johnson and Kannan, 2005) and multirotors/quadrotors. The aircraft included here are the AscTech Pelican quadrotor and the Yamaha RMAX small helicopter described in Chapter 8, with instrumentation and processing added to enable automatic flight. It combines concepts from this chapter as well as Chapters 5, 6, and 8. It has been utilized successfully to operate several other VTOL aircraft. With minor modification, it has also been utilized to fly airplanes (Johnson et al., 2008) and other configurations (Johnson and Turbe, 2006).

Description of an Adaptive Guidance, Navigation, and Control System for Miniature Aircraft

The position tracking of an aircraft is accomplished by utilizing the cascaded systems approach described in Section 9.4. This system utilizes attitude changes and direct force controls (such as rotor/propeller thrust) to achieve the desired position tracking. A state estimation system is utilized to generate estimates of vehicle state (angular velocity, attitude, linear velocity, and position) based on available sensors. The section below outlines the state estimation system that utilizes the Global Positioning Satellite System (GPS), an inertial measurement unit (IMU), and a magnetometer (heading). This configuration is illustrated in Figure 9.5-1. Using the common convention, the symbol $\hat{x}$ is used to indicate an estimated actual state x. Section 6.4 covers state estimation systems, and the section below outlines the specific system estimation system utilized in this chapter.

State Estimation From Section 6.4 we find that the separation principle justifies designing the guidance/control system and the state estimation system separately. However, this principle only applies for the case of linear systems, which is clearly not the case here. In fact, the controller alone is nonlinear due to the adaptive process. So, validation of the closed-loop behavior of the entire system over the intended flight regimes is important.

The estimator is based on utilizing inertial measurements (accelerometers and gyros) in an IMU. The IMU data is integrated forward in time to propagate attitude, velocity, and position. An extended Kalman filter (EKF), based on the Kalman filter from Section 6.4, is utilized to incorporate other sensor inputs to reduce error and bound drift in this propagation. Estimates of accelerometer and gyro biases are also updated in this manner.

The state estimation system estimates the vehicle attitude quaternion, $q_{frd/ned}$ seen in Chapter 1, Equation (1.8-16),

$$q_{frd/\text{ned}} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T, \tag{9.5-1}$$

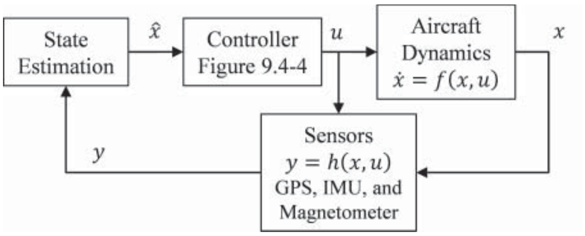
State Estimation
$\hat{x}$
Controller Figure 9.4-4
u
Aircraft Dynamics $\dot{x} = f(x,u)$
x
y
Sensors $y = h(x,u)$ GPS, IMU, and Magnetometer

Figure 9.5-1 Integrated adaptive guidance, navigation, and control systems.

the North/East/down (NED) position,

$$\begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^T, \tag{9.5-2}$$

the NED velocity,

$$\begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T, \tag{9.5-3}$$

accelerometer biases,

$$\begin{bmatrix} b_{ax} & b_{ay} & b_{az} \end{bmatrix}^T, \tag{9.5-4}$$

and gyro biases,

$$\begin{bmatrix} b_{\omega x} & b_{\omega y} & b_{\omega z} \end{bmatrix}^T, \tag{9.5-5}$$

These are propagated utilizing inertial data for the gyros,

$$\begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T, \tag{9.5-6}$$

and accelerometers corrected to the cg (measuring nongravitational specific force),

$$\begin{bmatrix} s_x & s_y & s_z \end{bmatrix}^T, \tag{9.5-7}$$

with

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \begin{bmatrix} 0 & -\omega_x + b_{\omega x} & -\omega_y + b_{\omega y} & -\omega_z + b_{\omega z} \\ \omega_x - b_{\omega x} & 0 & \omega_z - b_{\omega z} & -\omega_y + b_{\omega y} \\ \omega_y - b_{\omega y} & -\omega_z + b_{\omega z} & 0 & \omega_x - b_{\omega x} \\ \omega_z - b_{\omega z} & \omega_y - b_{\omega y} & -\omega_x + b_{\omega x} & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{9.5-8}$$

$$\begin{bmatrix} \dot{p}_x \\ \dot{p}_y \\ \dot{p}_z \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{9.5-9}$$

$$\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} = C_{ned/frd} \begin{bmatrix} s_x - b_{ax} \\ s_y - b_{ay} \\ s_z - b_{az} \end{bmatrix} + G, \tag{9.5-10}$$

where $C_{ned/frd}$ is the transpose of $C_{frd/ned}$, itself dependent on the quaternion (9.5-1) via Equation (1.8-20), and

$$\begin{bmatrix} \dot{b}_{ax} \\ \dot{b}_{ay} \\ \dot{b}_{az} \end{bmatrix} = \begin{bmatrix} \dot{b}_{\omega x} \\ \dot{b}_{\omega y} \\ \dot{b}_{\omega z} \end{bmatrix} = 0 \tag{9.5-11}$$

Equations (9.5-8) through (9.5-11) represent the process model and can be written as a single function,

$$\frac{d}{dt}\hat{x} = f(\hat{x}, \omega_x, \omega_y, \omega_z, s_x, s_y, s_z) \tag{9.5-12}$$

Note that in (9.5-10) G is an estimate of the acceleration due to gravitation not measured by the accelerometers expressed in the NED frame, described in Section 1.6. The accelerometer data is corrected for mounting location by addressing the expression of acceleration in a rotating frame described in Section 1.5. Specifically, Equation (1.5-4) can be utilized to estimate what a theoretical set of accelerometers located at the center of mass of the aircraft would measure based on measures from a known location relative to the center of mass ($\boldsymbol{p}_{P/Q}$):

$$\boldsymbol{s}_{CM/ned} = \boldsymbol{s}_{P/ned} - \boldsymbol{\alpha}_{frd/ned} \times \boldsymbol{p}_{P/Q} - \boldsymbol{\omega}_{frd/ned} \times (\boldsymbol{\omega}_{frd/ned} \times \boldsymbol{p}_{P/Q}) \tag{9.5-13}$$

To implement the EKF discrete update, it is also necessary to propagate a state estimate error covariance matrix P, appropriately initialized, with

$$\dot{P} = A^T P + PA + Q, \tag{9.5-14}$$

where

$$A = \frac{\partial f}{\partial \hat{x}} \tag{9.5-15}$$

is a matrix of partial derivatives of (9.5-12) and Q represents appropriate measurement noise covariance for the inertial measurements.

A discrete update of the state estimate and covariance matrix is made whenever any other sensor information is received using standard EKF methods (Gelb, 1974). For measurement y of the form

$$y = h(x) + v, \tag{9.5-16}$$

where v represents a zero mean measurement error with Gaussian distribution and variance R, the update is

$$C = \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}} \tag{9.5-17}$$

$$K = PC^T[CPC^T + R]^{-1} \tag{9.5-18}$$

$$\hat{x}^+ = \hat{x} + K[y - h(\hat{x})] \tag{9.5-19}$$

$$P^+ = [I - KC]P \tag{9.5-20}$$

For the results presented in this chapter, sensors are utilized and incorporated utilizing (9.5-17) to (9.5-20): a GPS receiver and a magnetometer. Both have a number of important issues described below.

For a tightly coupled filter, GPS data would require a clock bias estimate state, specifically a clock bias between the GPS constellation and the receiver clock. Then, each satellite measurement would be treated as the measurement of a single so-called *pseudorange* to each tracked satellite. In the loosely coupled design described here, the GPS receiver obtains estimates for clock bias, position, and velocity utilizing tracked satellite signals. As a result, enough satellite pseudoranges must be available for the GPS receiver to independently estimate 3D position and clock bias. These four unknowns require four pseudoranges. For this loosely coupled design, Equation (9.5-17) for the GPS measurement contains six values, which correspond to the velocity and position of the GPS receiver antenna on the aircraft. It is necessary to account for the fact that the antenna is not necessarily located at the center of mass of the vehicle. It is also necessary to account for the time delay of the estimate coming from the GPS receiver. Here, this was done by keeping a buffer of expected GPS measurements $[h(\hat{x})]$ from the past and using the stored values that correspond to the newly arrived GPS information (Christophersen et al., 2006). When performing the state estimate update for any sensor, it is critical that this buffer of expected measurements is also updated.

In this case a differential receiver is utilized that achieves position errors that are less than 1 ft more than 90% of the time. For a conventional GPS receiver, the accuracy is normally not sufficient for altitude control of a helicopter operating close to the ground. This altitude information might be augmented with a barometric altimeter or some other source of altitude measurement.

For the magnetometer, the sensor provides the direction and magnitude of the measured magnetic field resolved in a body-fixed coordinate system. In practice, the data only provide significant value in terms of aircraft heading. As a result, the measured data is utilized first to come up with a single measurement of aircraft heading utilizing a model of Earth's magnetic field as well as current aircraft roll/pitch angle estimates. So, (9.5-16) is simply the equation for aircraft heading. It is necessary when computing the difference between measured heading and estimated heading in (9.5-19) to take the angle between $-180°$ and $180°$ by adding or subtracting $360°$ to the difference as necessary.

Integrated Adaptive Guidance and Control The update rate of the IMU for the Yamaha RMAX example described below is 100 Hz. The update rate of the control system is limited to 50 Hz by the actuator interface utilized. These update rates are quick enough to make any aliasing effects discussed in Chapter 7 unlikely to occur. The important exception would be higher-frequency structural vibrations, presumably induced by the rotor, becoming aliased going into the IMU. To prevent this issue, the IMU was carefully mounted within a vibration isolated structure. This is intended to suppress those vibrations that could produce aliased signals (low-frequency noise).

At several places in the overall system, there is time delay occurring. This includes sensor dynamics, sensor processing, communication delays, sensor fusion processing time, actuation dynamics, and more. To the extent that the state estimation system has gotten all signals to approximately the same time synchronization (e.g., to that of the IMU), it may be a reasonable assumption that each element of the system adds a nearly consistent time delay to all feedback signals that pass through it. Because pure time delay is a linear phenomenon, a system with the net total time delay around the entire feedback loop can be approximated as a single time delay at one point in the system. That is, it may be less important to know how much time delay occurs at every step than to have a good estimate of total loop time delay. The design of the feedback control system should then account for this time delay and be robust to changes in it.

The total time delay was estimated by having the aircraft in an automatic hover and initiating a step response in the commanded horizontal position. A comparison was made between actuator command leaving the controller and sensed/estimated state arriving at the controller, with the most critical channel being cyclic pitch control and estimated angular velocity. Correlation between actuator command and resultant motion indicates a total of 160 ms of time delay around the feedback loop. It is important to realize that not all of this is likely pure time delay. Some of it is very likely other forms of higher-order dynamics, such as actuator motors and structural strain. Whatever the true source, it is important to ensure that the feedback controller can handle the full phase loss regardless of the source.

Processing requirements of this design are dominated by the state estimation system. The adaptive control scheme described here uses far less processing. The state estimation processing is itself dominated by propagating Equation (9.5-14). This involves multiplying two square matrices with a size equal to the number of states at every update rate of the filter, 100 Hz in this case. The numerical stability and accuracy of these calculations are critical to the performance of the filter. The results shown here actually propagate a factored version of P to improve accuracy, leading to a further increase in processing requirements (approximately double). Specifically, Bierman upper diagonal (UD) factorization is used (Bierman, 1977; Grewal and Andrews, 1993). A modern smartphone, personal computer, or laptop processor can easily handle these computations in real time. However, there may be an issue with some less capable and cheaper embedded processors utilized for miniature aircraft.

The architecture shown in Figure 9.4-4 requires a choice for reference models for both inner and outer subsystems. That is, we need to specify how we want the aircraft to try to move. An obvious choice is to look for a second-order response on each axis for attitude control (with selected frequency and damping ratios) as well as for each axis of position control. That is the choice made here. In the simplest version, we may want to achieve a natural frequency ω and a damping ratio ζ. In this case, the reference model dynamics would be

$$\ddot{r} = -K_D\dot{r} - K_P r, \tag{9.5-21}$$

where $K_D = 2\zeta\omega$ and $K_P = \omega^2$.

However, there are two important issues. First, there may be coupling between the inner and outer loop responses. Second, the designer will likely desire internal limits on the states.

Coupling between the inner and outer loops can be expected if the desired frequency of response of the outer loop is not much slower than that of the inner loop. When this is not the case, a codesign of both loops is necessary. In order to achieve the more precise position control, it is necessary to take this step. To perform the codesign, the internal dynamics of the inner and outer loops are based on the two sets of desired frequencies and damping ratios. Splitting up the inner and outer loop reference models into two sets is indicated here with subscripts:

$$\ddot{r}_i = -K_{D_i}\dot{r}_i - K_{P_i}r_i \tag{9.5-22}$$

$$\ddot{r}_o = -K_{D_o}\dot{r}_o - K_{P_o}r_o, \tag{9.5-23}$$

where the output of the outer loop controller (o) is the external input to the inner loop controller (i). One can then derive appropriate parameters for these reference models to achieve two chosen frequencies and damping ratios (now a fourth-order system). Here, we chose to make the damping ratios unity and the two frequencies equal. This is motivated by a desire to achieve the highest bandwidth position control within this architecture. The characteristic equation with four poles at $-\omega$ looks like

$$(s+\omega)^4 = s^4 + K_{D_i}s^3 + K_{P_i}s^2 + K_{P_i}K_{D_o}s + K_{P_i}K_{P_o} = 0 \tag{9.5-24}$$

This can then be utilized to solve for the reference model parameters:

$$K_{P_i} = 6\omega^2,\ K_{D_i} = 4\omega \tag{9.5-25}$$

$$K_{P_o} = \frac{\omega^2}{6},\ K_{D_o} = \frac{2\omega}{3} \tag{9.5-26}$$

The second issue is the potential need for internal limits. If the aircraft is asked to move a very large distance horizontally, the necessary velocities and attitudes to achieve a linear response will (with a sufficiently large distance) be inappropriate. For example, they may exceed the maximum possible speed of the aircraft. Internal limits on linear and angular velocity are included in the reference models in a similar manner to provide this functionality. First, a desired velocity component is found for each axis that seeks to eliminate position or attitude error:

$$v_{Unlimited} = -\frac{K_P}{K_D}r \tag{9.5-27}$$

This component is then limited as appropriate. For example, the norm of all three components might be limited while still maintaining the desired direction. The reference model motion is then computed by

$$\ddot{r} = K_D(v_{Limited} - \dot{r}) \tag{9.5-28}$$

In this way, we achieve the desired second-order response when the internal state is not limited. When it is limited, the system will track the limit. This reference model architecture is similar to nested saturation controllers (Teel, 1992), which can enable some important theoretical stability results to be stated (Johnson and Kannan, 2003).

In this design the linear feedback gains were chosen to be identical to the reference model feedback gains [Equations (9.5-21), (9.5-25), and (9.5-26)]. In some applications, it might be desirable to make them somewhat faster than the reference model so that the motion of the reference model is the dominant response. Here, the reference model is of no special significance.

The inner loop (attitude) nominal dynamics were simplified as much as possible here. This is justified by having the adaptive controller able to address the many factors not included in this model. For each rotational axis, we introduce a diagonal element into a control effectiveness matrix B and a diagonal element into the rate damping term A:

$$\dot{x} = Ax + B\delta_i, \tag{9.5-29}$$

where x is the angular velocity components (P, Q, and R) and δ_i are the attitude control inputs (roll, pitch, yaw). It is trivial to invert this model to find the δ_i to achieve a desired angular acceleration as required by the controller.

For the outer loop (position control) nominal dynamic inverse included in Figure 9.4-4, the simplified model takes the multirotor or helicopter as simply a thrust vector (the main rotor or direction of multiple propellers) that can be pointed in any direction, after accounting for the effect of gravity. This is equivalent to neglecting other side forces and relying on adaptation to account for them. At a similar level of approximation as applied for the attitude control case, the thrust per unit mass (specific force), directed along the body-fixed down axis, is modeled by

$$s_z = Z_w W + Z_\delta \delta_o, \tag{9.5-30}$$

where W is the down body-fixed axis velocity component.

In order to realize this design, it is necessary to relate outer loop pseudocontrol (linear acceleration commands) to this thrust specific force and attitude commands. First, the total necessary specific force is found by

$$s_z = |\mathbf{v}_o - \mathbf{G}|, \tag{9.5-31}$$

where $\mathbf{v}_o$ is outer loop pseudocontrol and $\mathbf{G}$ is the estimated effect of gravitation. For example, if the outer loop pseudocontrol command ($\mathbf{v}_o$) was zero (i.e., zero acceleration desired), then this would result in a commanded Z-axis specific force equal to the magnitude of gravitation. The angle that the body frame needs to be tilted is then found by first finding a unit vector in the direction of the desired specific force,

$$[-\mathbf{k}_{frd}]_{desired} = \hat{\mathbf{s}} = \frac{\boldsymbol{v}_o - \mathbf{G}}{|\boldsymbol{v}_o - \mathbf{G}|}, \tag{9.5-32}$$

which represents the desired negative body-fixed down axis direction. To use the same example again where $\mathbf{v}_o$ is zero, this would result in the desired negative body-fixed down axis pointing in the opposite direction as gravitation, which again corresponds to hover.

One can also independently specify a desired heading for the aircraft $\psi_{desired}$, which need not be aligned with the flight path of the aircraft for a helicopter or multirotor. The desired body-fixed forward axis needs to be perpendicular to both the desired down direction and the NED unit vector,

$$\mathbf{j}_{ned} = \begin{bmatrix} -\sin\psi_{desired} & \cos\psi_{desired} & 0 \end{bmatrix}^T \tag{9.5-33}$$

Because it is perpendicular to both of the unit vectors obtained, the forward direction is found by normalizing the cross product of them

$$[\mathbf{i}_{frd}]_{desired} = \frac{\mathbf{j}_{ned} \times [\mathbf{k}_{frd}]_{desired}}{|\mathbf{j}_{ned} \times [\mathbf{k}_{frd}]_{desired}|} \tag{9.5-34}$$

and the right direction is perpendicular to both, or

$$[\mathbf{j}_{frd}]_{desired} = [\mathbf{k}_{frd}]_{desired} \times [\mathbf{i}_{frd}]_{desired} \tag{9.5-35}$$

The combination of Equations (9.5-32) to (9.5-35) provides the full direction cosine matrix of the desired body-fixed attitude with respect to NED. This becomes the desired attitude for the inner-loop controller. If converted into a quaternion as described in Section 1.8, the symbol $q_{desired}$ would be appropriate.

Because attitude control is an inherently nonlinear problem, some additional development is required in order to relate desired and estimated attitude in order to produce an error angle that the inner-loop controller can track. There are many potential treatments of this problem found in the literature. Utilizing Euler angles to formulate this problem is one possibility (Das et al., 2009). Bach and Paielli (1993) present commonly used methods that utilize the direction cosine matrix as well as the quaternion to formulate this problem. Here, a commonly used quaternion error angle formulation is utilized to form a set of three error angles. This results in both the desired linear response for small perturbations and global stability under reasonable assumptions. The error angles are found by (Wie and Barba, 1985)

$$q_{desired} = \begin{bmatrix} q_{d0} & q_{d1} & q_{d2} & q_{d3} \end{bmatrix}^T \tag{9.5-36}$$

$$q_{frd/ned} = \begin{bmatrix} q_0 & q_1 & q_2 & q_3 \end{bmatrix}^T \tag{9.5-37}$$

$$q_{desired} * [q_{frd/ned}]^{-1} = \begin{bmatrix} \Delta q_0 \\ \Delta q_1 \\ \Delta q_2 \\ \Delta q_3 \end{bmatrix} = \begin{bmatrix} q_{d0} & -q_{d1} & -q_{d2} & -q_{d3} \\ q_{d1} & q_{d0} & -q_{d3} & q_{d2} \\ q_{d2} & q_{d3} & q_{d0} & -q_{d1} \\ q_{d3} & -q_{d2} & q_{d1} & q_{d0} \end{bmatrix} \begin{bmatrix} q_0 \\ -q_1 \\ -q_2 \\ -q_3 \end{bmatrix} \tag{9.5-38}$$

$$\begin{bmatrix} \Delta\theta_f \\ \Delta\theta_r \\ \Delta\theta_d \end{bmatrix} = \frac{2\Delta q_0}{|\Delta q_0|} \begin{bmatrix} \Delta q_1 \\ \Delta q_2 \\ \Delta q_3 \end{bmatrix}, \tag{9.5-39}$$

where $\Delta\theta_f$, $\Delta\theta_r$, and $\Delta\theta_d$ are now the roll, pitch, and yaw attitude error angles, respectively, in the body-fixed frame suitable for tracking by the inner-loop controller [Equation (9.5-22)]. These angles take the role of r_i in the expression, and the linear form of (9.5-22) will be exact for small tracking error.

It is important to handle the case where desired thrust cannot be achieved. That is not an issue encountered in the results described later in this chapter, but it is important to shape the desired trajectory to appropriately account for priorities in position control in general. There are times when it may be reasonable to sacrifice horizontal position tracking in order to maintain altitude. There may be times when it may be preferred to sacrifice altitude control in order to maintain rotor rpm, such as in the early phase of a helicopter autorotation. These trade-offs could be handled by a higher-level planner that is able to shape the desired trajectory under these more complex contingencies.

The outputs of the controller at this point are roll, pitch, yaw, and thrust commands. For any particular aircraft configuration, these are then mapped into appropriate actuator commands. For the conventional helicopter, these would be proportional to cyclic, tail rotor collective, and main rotor collective. For the multirotor, these would map to throttle commands for the individual motors: for example, by inverting a linear mapping of forces and moments as a function of the desired thrust from each propeller of a quadrotor (Das et al., 2009). For the quadrotor arranged as an example in Section 8.6, this would be

$$\begin{bmatrix} \delta_o \\ \delta_{i_1} \\ \delta_{i_2} \\ \delta_{i_3} \end{bmatrix} = \begin{bmatrix} 1/m & 1/m & 1/m & 1/m \\ y_P/J_x & y_P/J_x & -y_P/J_x & -y_P/J_x \\ x_P/J_y & -x_P/J_y & -x_P/J_y & x_P/J_y \\ -C/J_z & C/J_z & -C/J_z & C/J_z \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, \qquad (9.5\text{-}40)$$

where f_1, f_2, f_3, and f_4 are the desired thrust of each propeller and C is an assumed proportional relationship between the thrust and torque of the propeller.

There is one contingency similar to the above that can be addressed within the controller, at least in principle. In the development of this type of system, it is important that position control is still possible even when the heading control is not. That is, aircraft attitude changes utilized to hold position need not (and should not) assume that heading control is successful. Heading may not be maintained due to a saturated tail rotor, typically due to a wind or sideslip condition. For example, when attempting to fly sideways at an excessive speed the aircraft may not have enough tail rotor authority to maintain the sideslip. Fundamentally, this is achieved by utilizing current heading rather that desired heading when coming up with which way to roll/pitch to achieve desired position/velocity.

An important detail with this type of controller is what to do during takeoff and landing, specifically what to do with NN adaptation while the aircraft is on the ground. If this issue is ignored, then extreme control inputs may result on the ground as the controller attempts unsuccessfully to adapt to the fact that the aircraft is mechanically constrained by ground contact. Similar to integrators in nonadaptive controllers, a common approach is to halt adaptation when the system detects that the aircraft is in contact with the ground. That method has been utilized here.

Simulation Results

Simulation results for two different applications of the adaptive guidance and control methods described in this chapter are presented here. First, the trajectory-tracking flight of a quadrotor is described. Following this, more detailed results are provided for a small helicopter.

Simulation of Quadrotor Simulation tests were conducted for the guidance and control methods described above applied to the AscTec Pelican quadrotor described in Section 8.6. The relevant parameters are provided in Table 9.5-1. It is important to note that the "internal" limits are applied for capturing the desired state. The actual limits are determined when planning the desired trajectory itself. This controller was specifically optimized for indoor flight at relatively low speed but high precision. For this example, the simulation of the navigation (state estimation) system is dispensed with, and true values of aircraft state are utilized for feedback. Due to this fact, any transient or error is due entirely to the control system.

Figure 9.5-2 shows the position tracking for a rapidly flown small box pattern at constant altitude holding heading to zero. The desired trajectory consisted of

TABLE 9.5-1 AscTec Pelican Controller Parameter Choices for the Results Presented in this Section

Component	Parameter	Value
Reference model	Lateral pole locations (ω)	2.5
	Longitudinal pole locations	2.5
	Altitude pole location	3.0
	Yaw pole location	2.5
	Reference model internal velocity limit	1.5 ft/s
	Reference model internal rate limit	2 rad/s
Adaptation	Outer layer learning rate (Γ_W)	1.0 (all)
	Inner layer learning rate (Γ_V)	5.0 (all)
	e-Modification (λ)	0.1 (all)
Nominal inverse	Roll control effectiveness (B)	10
	Pitch control effectiveness	10
	Yaw control effectiveness	6
	Roll damping (A)	0
	Pitch damping	0
	Yaw damping	0
	Thrust control effectiveness (Z_δ)	−32
	Thrust damping (Z_w)	0

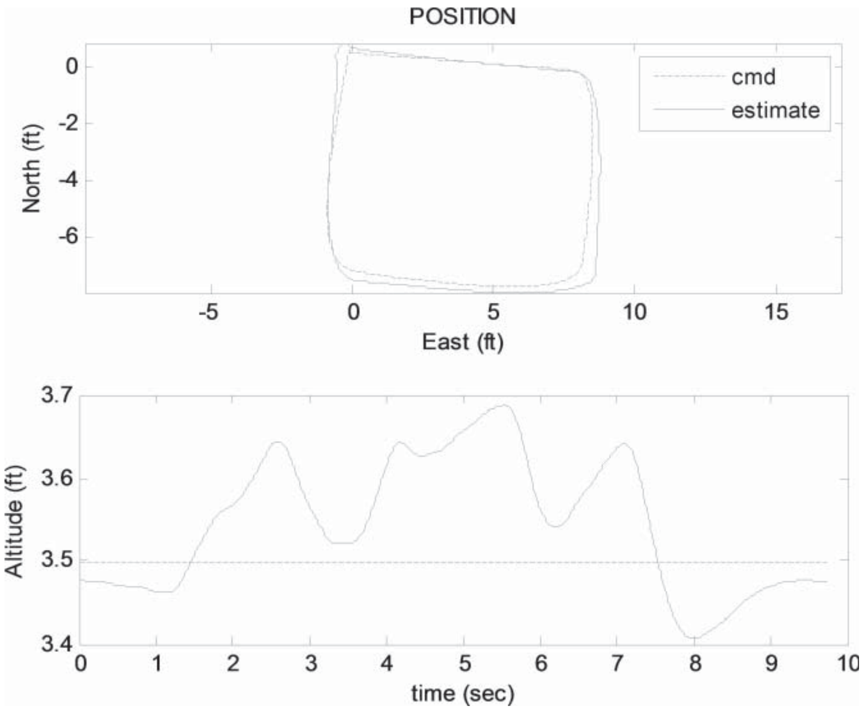

Figure 9.5-2 Adaptive GNC position tracking for rapidly flown small box pattern from simulation of quadrotor.

acceleration/deceleration segments of 10 ft/s^2 and a maximum speed of 10 ft/s. Figure 9.5-3 shows the corresponding attitude angles during the maneuvers. Roll and pitch rapidly change to around 20° to achieve the desired acceleration levels. Heading is held near zero. Figure 9.5-4 shows the revolution rate in RPM of each of the four propellers. Rapid changes are required to track this desired trajectory.

A video from this box pattern simulation is available at: http://uav.ae.gatech.edu /videos/gtqS150131_square.mp4

Simulation of Small Helicopter Simulation tests were conducted for the system described above as applied to a small helicopter. In addition to the dynamic model of the Yamaha RMAX helicopter described in Section 8.5, appropriate models were introduced for the sensors: IMU, GPS, and magnetometer. These sensor models included the effects of mounting location, time delay, update rate, quantization, and noise. A screenshot of the visualization capability of the simulator is shown in Figure 9.5-5. The relevant parameters are provided in Table 9.5-2. It is important to note that the "internal" limits are applied for capturing the desired state. The actual limits are determined when planning the desired trajectory.

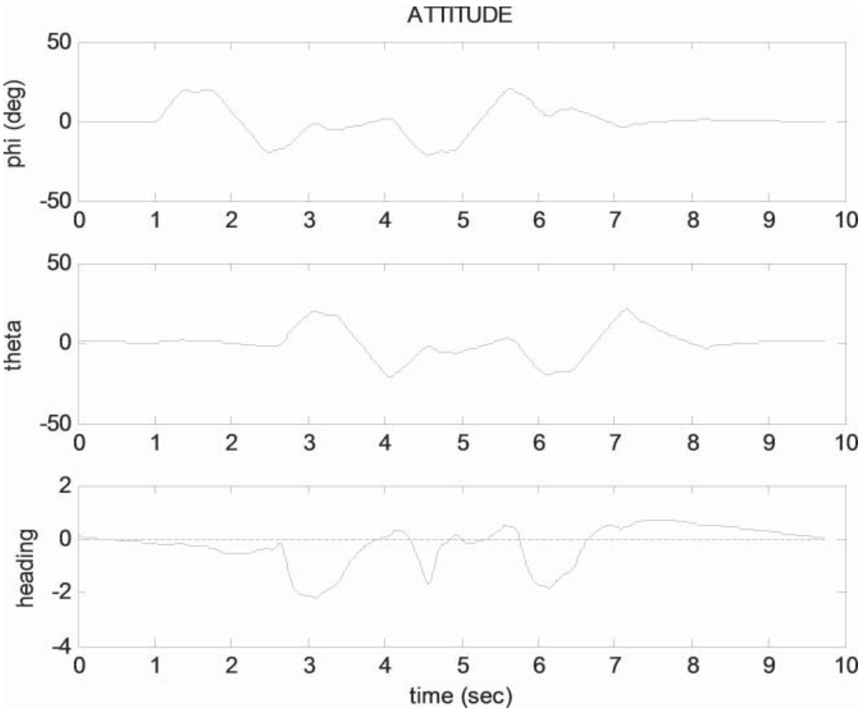

Figure 9.5-3 Adaptive GNC attitude angles for rapidly flown small box pattern from simulation of quadrotor.

The results for step responses are described first. Note that in normal operational flight one would utilize smooth desired-trajectory commands. Here, we intentionally provide nonsmooth trajectory commands to verify tracking performance, including achieved bandwidth. In other words, normal flight would not involve inputs this aggressive for such small movements. This normal operation is illustrated in results that follow the step responses, showing more conventional flight operations.

It is important to note that plots of estimated vehicle state below are from the state estimate system driven by simulated measurements. It would be possible to also plot truth values for states to see the effect of both the controller and the state estimate components separately. However, due to the performance of the GPS utilized, there is very little difference between true state and estimated state in the simulator, typically within 0.2 ft. Given this fact and our interest in looking at controller performance primarily in this section, state estimates are plotted. This has the added benefit of allowing a more direct comparison of controller performance to the flight test results that follow, where true states are not known.

Simulation of Small Helicopter Step Responses Figure 9.5-6 shows the response for an instantaneous 180° change in the commanded heading. A bit more

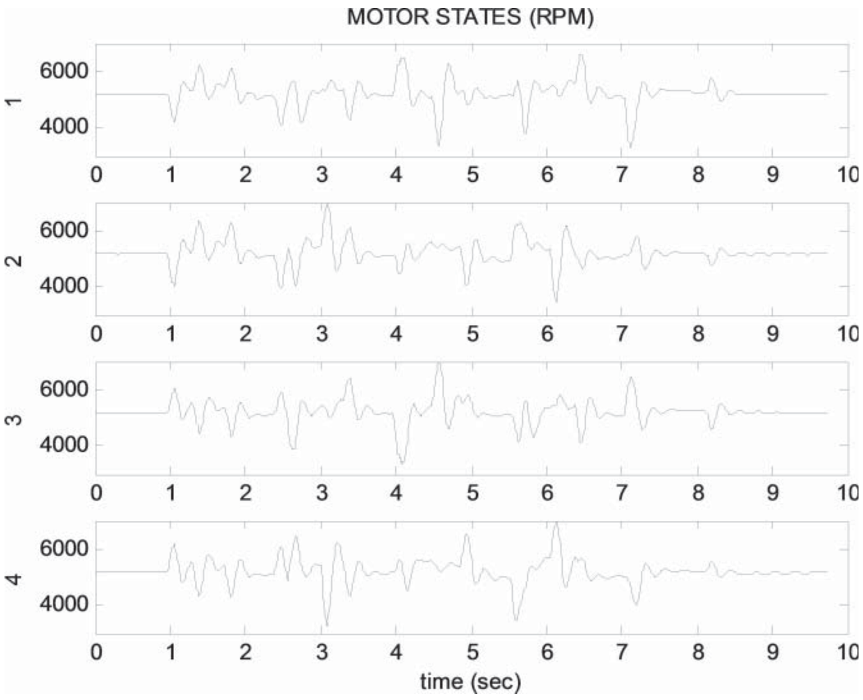

Figure 9.5-4 Adaptive GNC motor states for rapidly flown small box pattern from simulation of quadrotor.

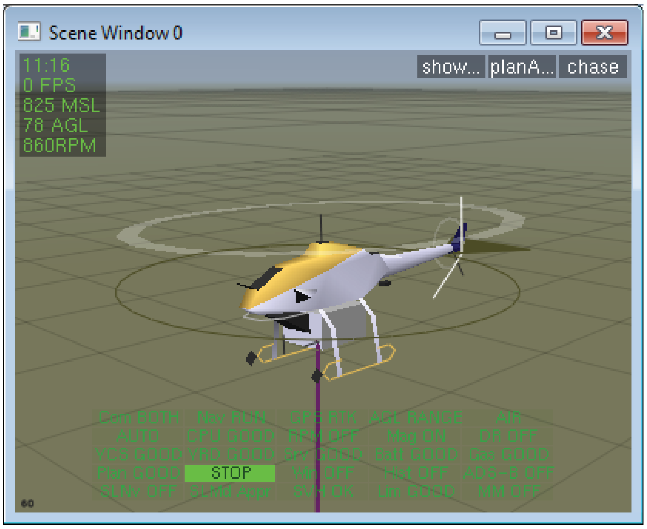

Figure 9.5-5 Visualization of Yamaha RMAX in simulation environment.

TABLE 9.5-2 Yamaha RMAX Controller Parameter Choices for the Results Presented in this Section

Component	Parameter	Value
Reference model	Lateral pole locations (ω)	2.5
	Longitudinal pole locations	2.0
	Altitude pole location	3.0
	Yaw pole location	3.0
	Reference model internal velocity limit	10 ft/s
	Reference model internal rate limit	2 rad/s
Adaptation	Outer layer learning rate (Γ_W)	1.0 (all)
	Inner layer learning rate (Γ_V)	5.0 (all)
	e-Modification (λ)	0.1 (all)
Nominal inverse	Roll control effectiveness (B)	15
	Pitch control effectiveness	10
	Yaw control effectiveness	15
	Roll damping (A)	-10
	Pitch damping	-8
	Yaw damping	-6
	Thrust control effectiveness (Z_δ)	-25
	Thrust damping (Z_w)	0

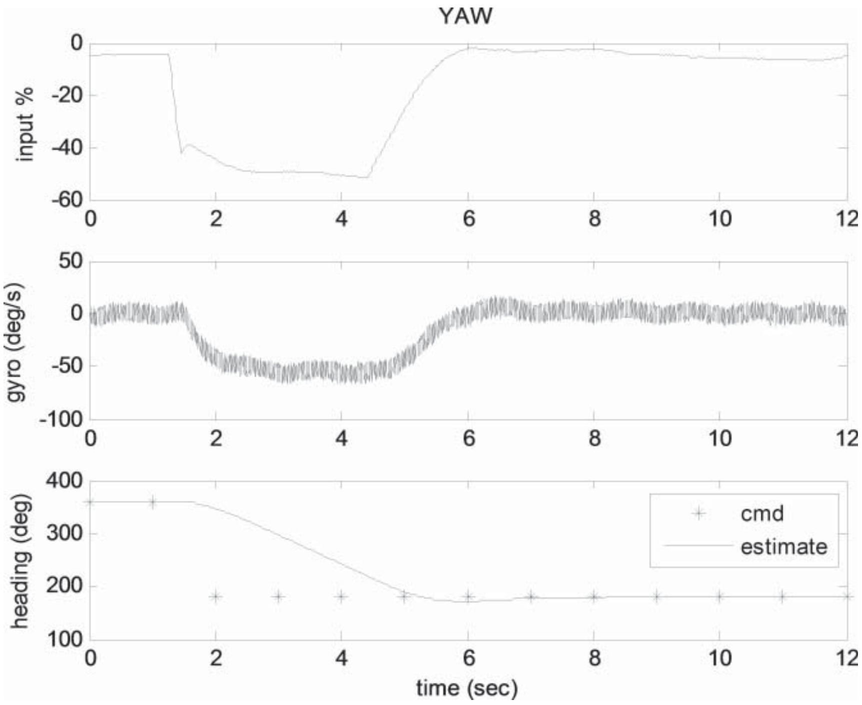

Figure 9.5-6 Adaptive GNC 180° heading command step response (left) from simulation of RMAX.

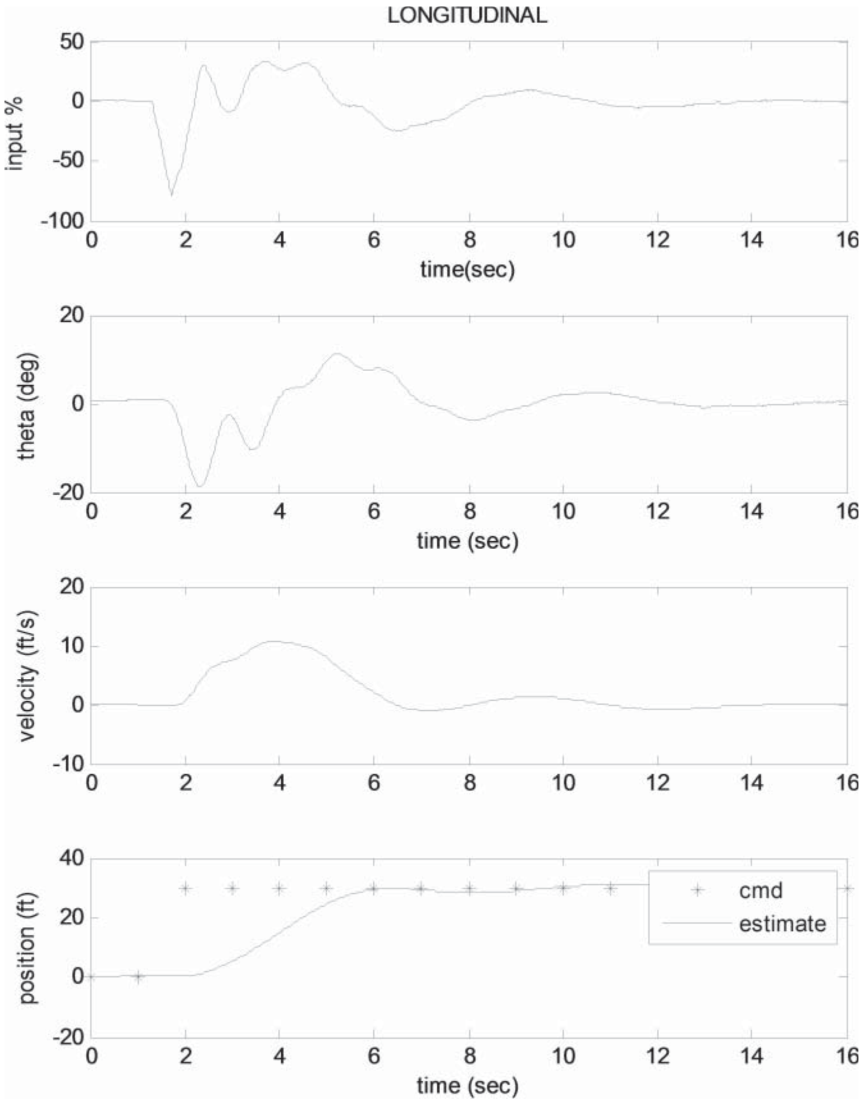

Figure 9.5-7 Adaptive GNC 30 ft longitudinal position command step response (forward) from simulation of RMAX.

than 50% of maximum tail rotor authority is utilized to generate a maximum of approximately 50 deg/s of rotation, bringing the aircraft around for a total settling time of approximately 4 s. The raw yaw gyro is shown in the middle trace, and includes a significant once-per-rotor-revolution oscillation in this simulated raw measurement (about 14 Hz).

Figure 9.5-7 shows the longitudinal position response to a step input of 30 ft forward. This is accomplished primarily with longitudinal cyclic input. Approximately

75% of the full throw of the cyclic input is utilized to initiate the maneuver, quickly getting the aircraft up to the reference model internal rate limit of 10 ft/s. Note that this internal rate limit is only applied to intercepting the desired trajectory. The trajectory command has a much higher limit corresponding to the maximum speed of the aircraft. The position settling time of 5 s corresponds very well to the theoretical settling time of a fourth-order system with the desired frequencies specified here ($\omega = 2$).

Figure 9.5-8 is the corresponding plot for lateral position response to a step input (to the left) of 30 ft. This is accomplished primarily with lateral cyclic input. Again, about 75% of full throw is utilized to start the maneuver. The internal rate limit is just reached (10 ft/s), and we see a settling time of approximately 4 s. This is close to the theoretical value based on the desired frequency ($\omega = 2.5$).

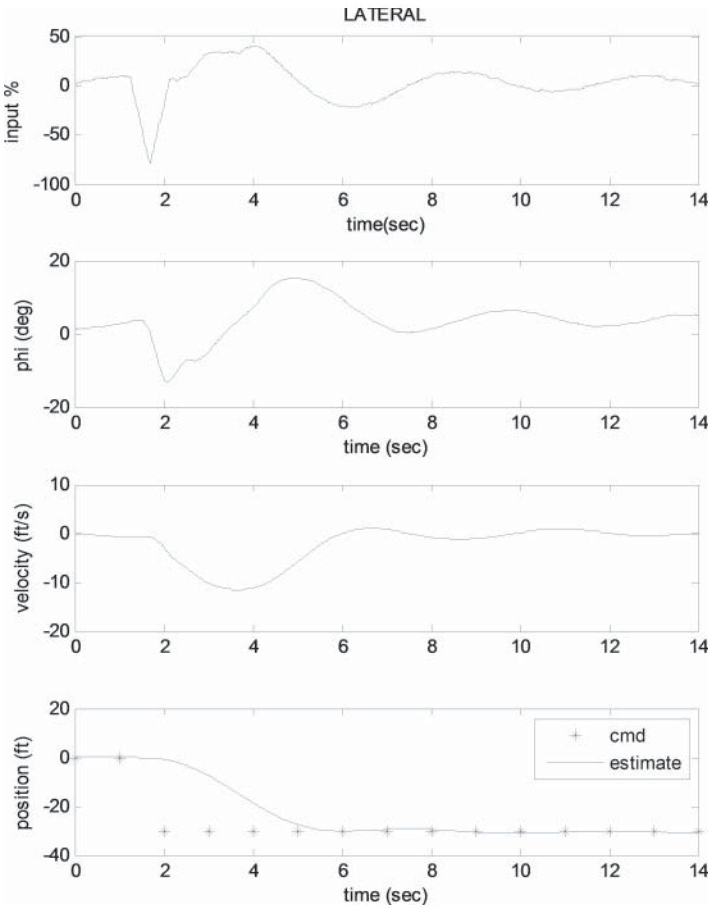

Figure 9.5-8 Adaptive GNC 30 ft lateral position command step response (left) from simulation of RMAX.

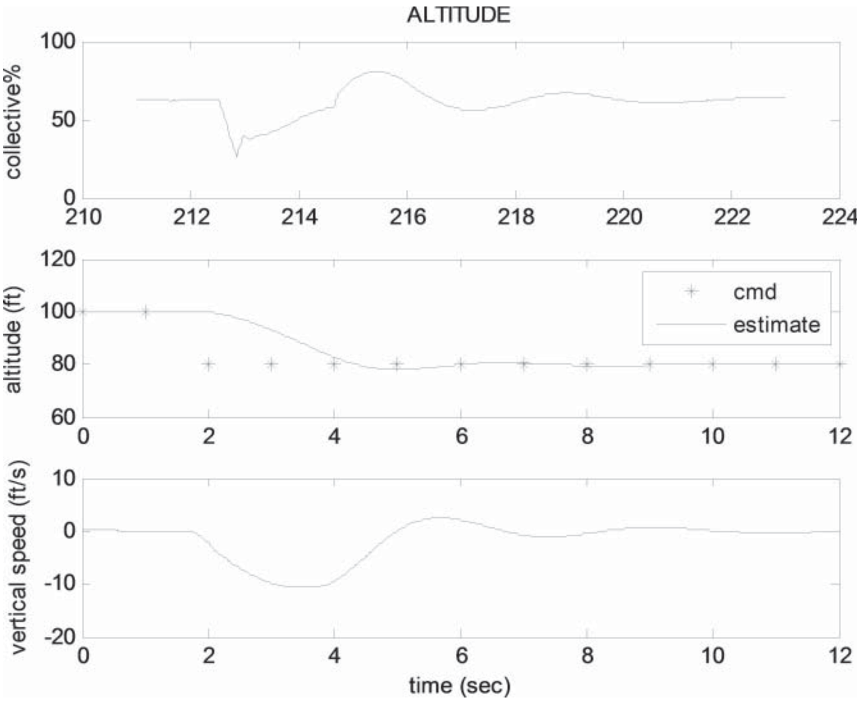

Figure 9.5-9 Adaptive GNC 20 ft altitude command step response (down) from simulation of RMAX.

Figure 9.5-9 shows the altitude control response for a step input of 20 ft down. Again, about 50% of full throw of the collective pitch is utilized to start the descent and somewhat less in the reverse direction to stop it. The settling time is about 3 s, just reaching the internal rate limit of 10 ft/s (600 ft/min).

Figure 9.5-10 shows the position for the response to a "superstep" test of the system, where simultaneous position steps in all three directions (NED) and a 180° change in heading take place simultaneously. This maneuver is used to verify the performance of the system when there is movement on more than one axis. Essentially, this is a worst case transient response. Figure 9.5-11 shows velocity just approaching the internal limit of 10 ft/s in magnitude. Figure 9.5-12 shows the attitude during the maneuver. Notice that the initial reaction (nose down) is correct even though the aircraft is, at that moment, still 180° away from the desired heading. Figure 9.5-13 shows the inputs, utilizing between one-fourth and one-half of full throw in each axis.

A video from this superstep simulation is available at: http://uav.ae.gatech.edu /videos/fgS141026_superstep.mp4

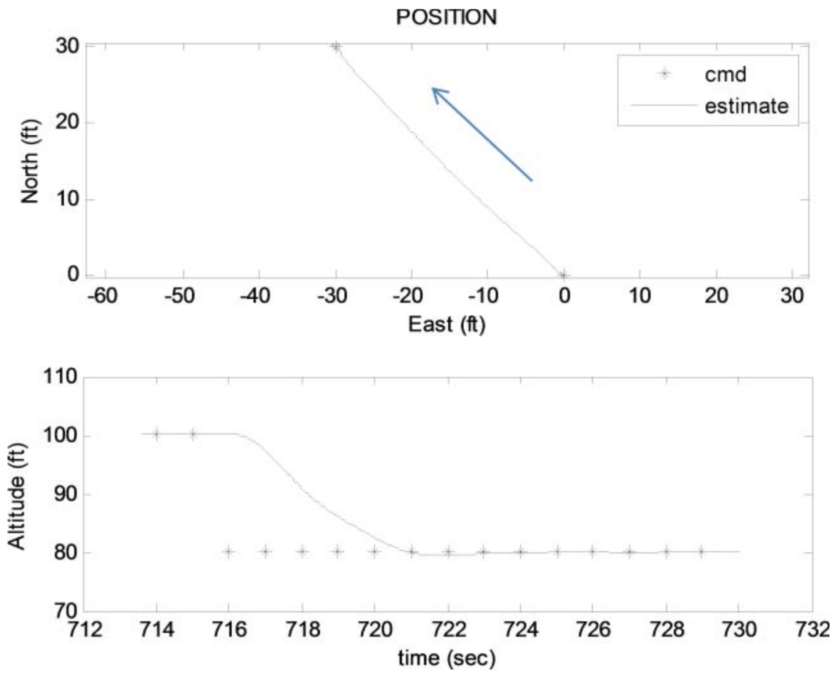

Figure 9.5-10 Adaptive GNC position and altitude tracking step response on all axes simultaneously from simulation of RMAX.

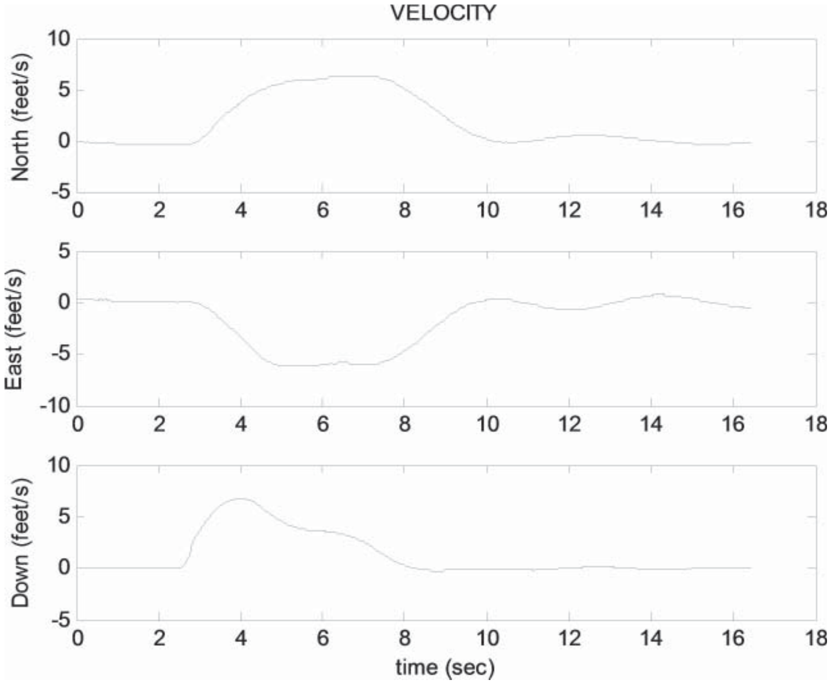

Figure 9.5-11 Adaptive GNC velocity during step response on all axes simultaneously from simulation of RMAX.

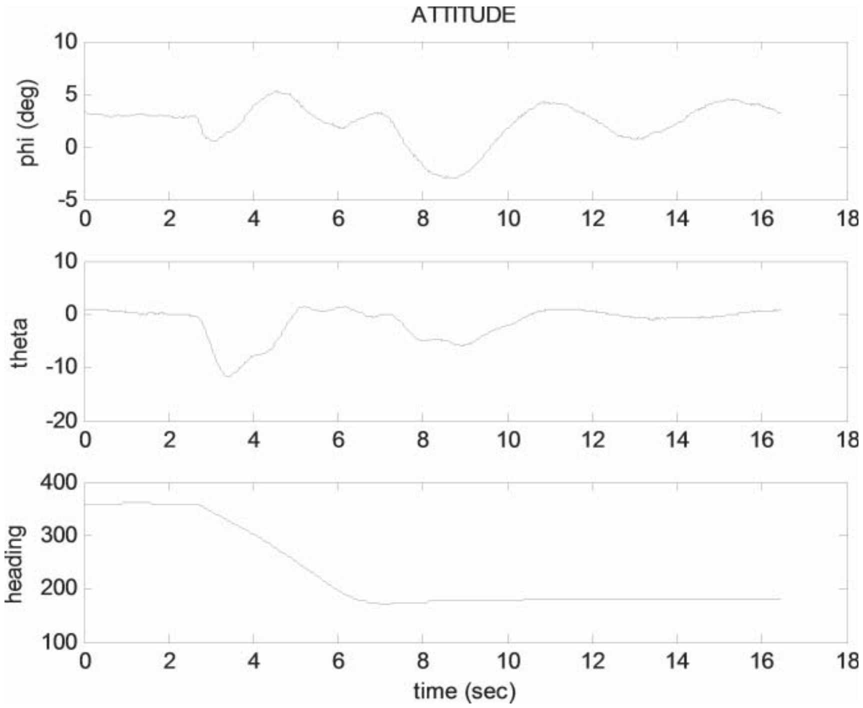

Figure 9.5-12 Adaptive GNC altitude during step response on all axes simultaneously from simulation of RMAX.

Simulation of Normal Small Helicopter Maneuvers A complex flight pattern is utilized to exercise the system executing a series of maneuvers that might be used in normal flight. These maneuvers include acceleration from a hover, deceleration to a hover, sideways flight, rearward flight, and forward flight. Climbs and descents are also included. The test pattern lasts approximately 120 s, with the first half at low speed and the second half at a typical cruise speed.

Figure 9.5-14 shows the position tracking for a simulated test of these maneuvers. The right side of the figure shows a close-up of the low-speed portion of the flight. Tracking is good throughout. Note that the low-speed flight segment includes sideways flight (both ways) and rearward flight.

Figure 9.5-15 shows altitude and speed tracking. One can see that the low-speed portion includes speeds between zero and 15 ft/s. The high-speed segment is at 50 ft/s. Note that the deceleration chosen for the stop-to-hover at the end of the profile was a relatively high 10 ft/s^2, which does result in a few feet of altitude tracking error during the tail end of this relatively aggressive maneuver. We see that the maneuver resulted in 20° of nose-up pitch attitude in Figure 9.5-16. The high-speed turns were done at around 20° of bank in this case.

Figure 9.5-17 shows the inputs utilized during these same maneuvers. A significant fraction of the full collective pitch capability was utilized for the high-speed flight

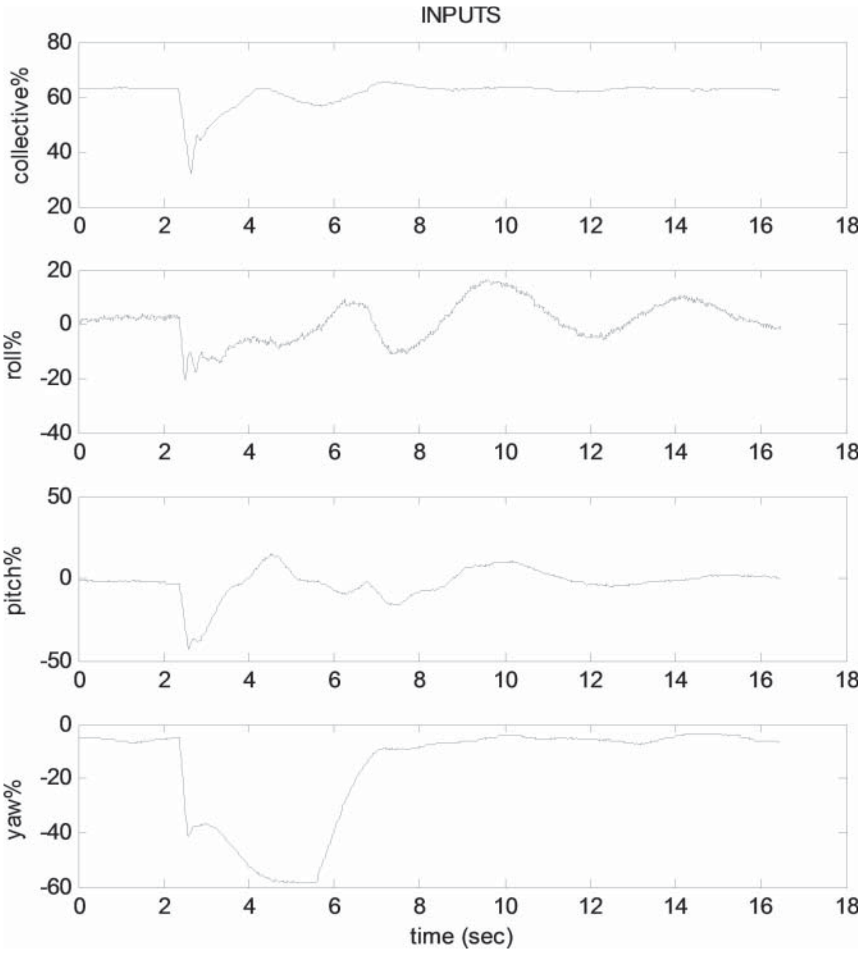

Figure 9.5-13 Adaptive GNC inputs during step response on all axes simultaneously from simulation of RMAX.

segment. Approximately half of full throw was utilized for cyclic input (both in roll and pitch axes), and a slightly lesser amount of tail rotor authority was required.

A video from this simulation is available at: http://uav.ae.gatech.edu/videos /fgS141026_exercises.mp4

Flight Test Results

A flight test of the above RMAX small helicopter system was conducted and is described here. The results for step responses are described first. Following this, normal operation is illustrated in the results that follow the step responses, showing a conventional oval racetrack flight path.

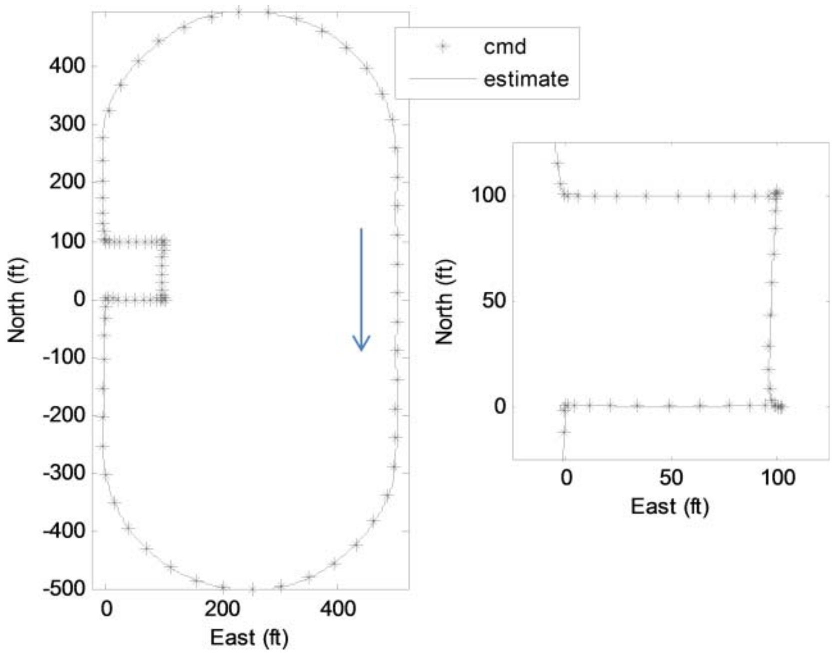

Figure 9.5-14 Adaptive GNC position tracking during complex flight pattern from simulation of RMAX.

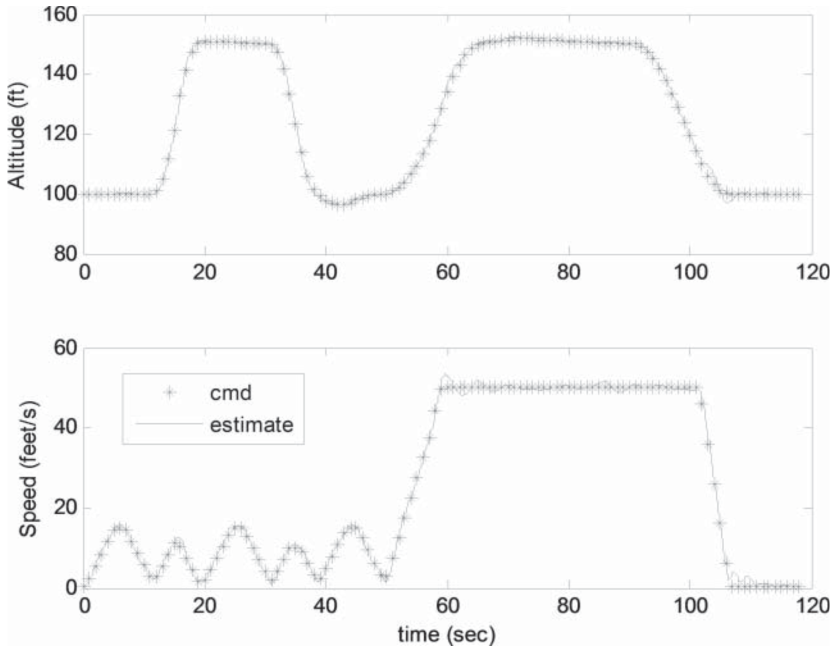

Figure 9.5-15 Adaptive GNC altitude and speed tracking during complex flight pattern from simulation of RMAX.

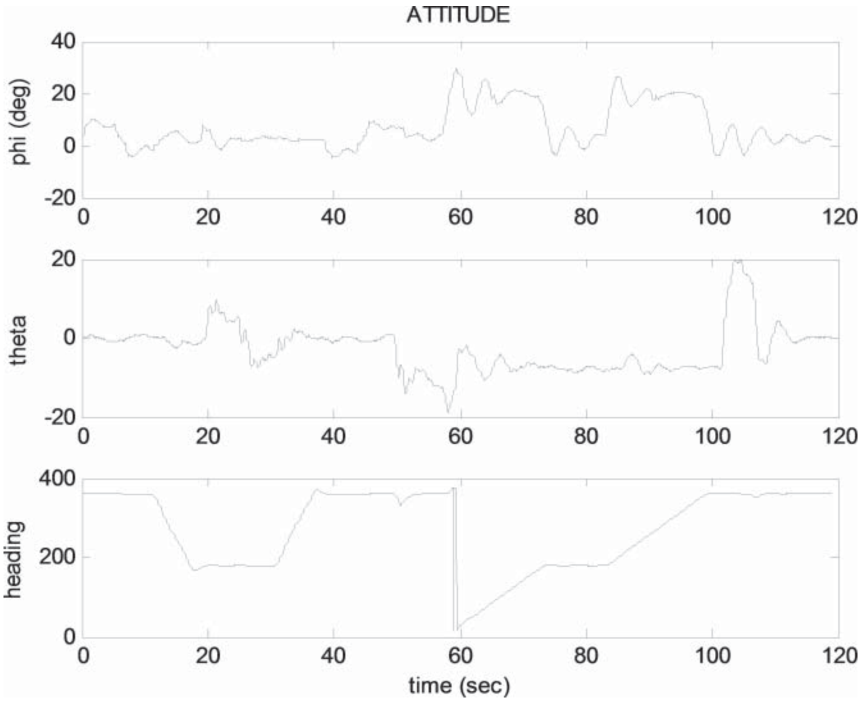

Figure 9.5-16 Adaptive GNC altitude during complex flight pattern from simulation of RMAX.

Flight Test of Step Responses Figure 9.5-18 shows the response for an instantaneous 180° change in the commanded heading. A bit more than 50% of maximum tail rotor authority is utilized to generate a maximum of approximately 50 deg/s of rotation, bringing the aircraft around for a total settling time of approximately 5 s.

Figure 9.5-19 shows the longitudinal position response to a step input of 30 ft forward. This is accomplished primarily with longitudinal cyclic input. Approximately 50% of full throw of the cyclic input is utilized to initiate the maneuver, quickly getting the aircraft up to the reference model internal rate limit of 10 ft/s. The position settling time of 5 s corresponds very well to the theoretical settling time of a fourth-order system with the desired frequencies specified here ($\omega = 2$).

Figure 9.5-20 is the corresponding plot for lateral position response to a step input (to the left) of 30 ft. This is accomplished primarily with lateral cyclic input. Again, about 50% of full throw is utilized to start the maneuver. The internal rate limit is just reached (10 ft/s), and we see a settling time of approximately 4 s. This is close to the theoretical value based on the desired frequency ($\omega = 2.5$).

Figure 9.5-21 shows the altitude control response for a step input of 20 ft down. Again, about 50% of full throw of the collective pitch is utilized to start the descent

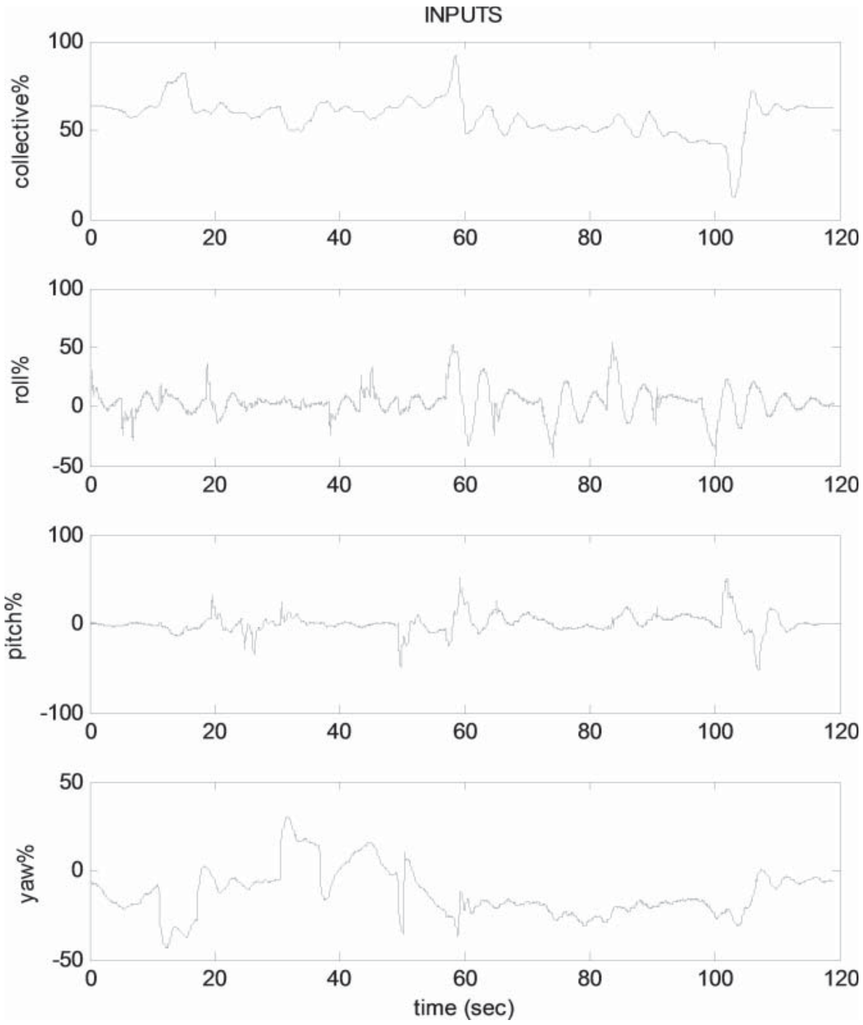

Figure 9.5-17 Adaptive GNC inputs during complex flight pattern from simulation of RMAX.

and somewhat less in the reverse direction to stop it. The settling time is about 3 s, just reaching the internal rate limit of 10 ft/s (600 ft/min).

A video of the pitch response is available at: http://uav.ae.gatech.edu/videos/g041222e2_30ftStep.mpg

A video of the yaw response is available at: http://uav.ae.gatech.edu/videos/g041222e3_180degStep.mpg

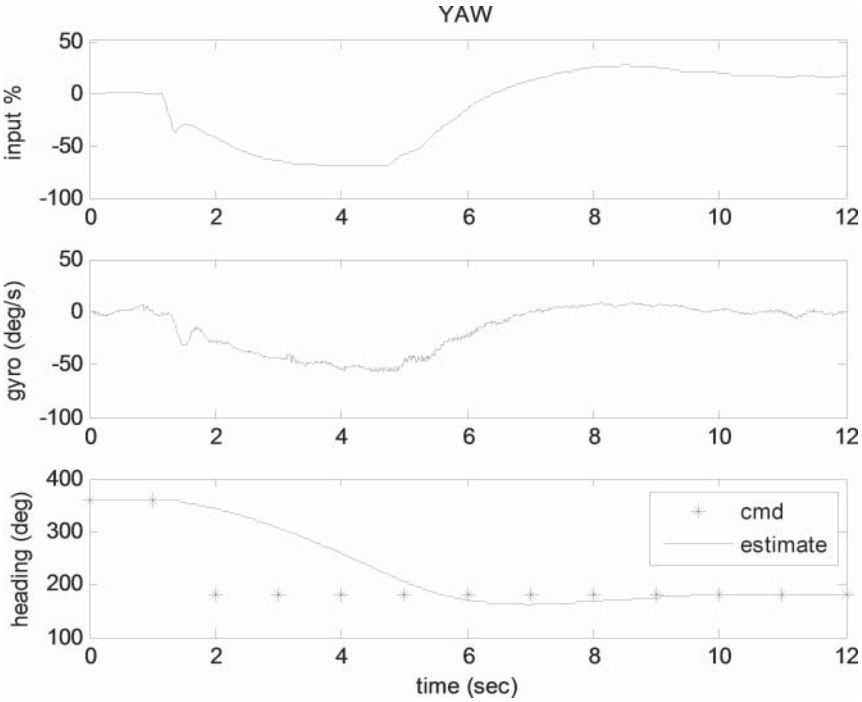

Figure 9.5-18 Adaptive GNC 180° heading command step response (left) from flight test of RMAX.

Flight Test of a Conventional Flight Pattern Figures 9.5-22 to 9.5-25 illustrate the response of the system when being utilized to fly a typical flight pattern consisting of accelerating, cruising, turning, and stopping. Figure 9.5-22 shows the position tracking. The aircraft starts near the origin and proceeds to the east, accelerating to 50 ft/s. It then turns right until it is heading west. It then turns right again so that it is heading toward where it started. It then proceeds to the starting location and stops. The commanded position shown in the figure is plotted once per second, providing an indication of the segment of flight that involved flight at a lower speed (near the origin).

Figure 9.5-23 shows the total speed of the aircraft during this maneuver. The desired acceleration was 5 ft/s^2, and so it requires 10 s to reach the intended speed. The reverse is true on the stop. The slight transient about halfway through the start occurred because the aircraft started the maneuver at the wrong heading and so was simultaneously making a large heading change and accelerating.

Figure 9.5-24 shows the estimated attitude during the maneuvers as standard Euler angles (Section 1.3). The roll angle is approximately 20° during the turns.

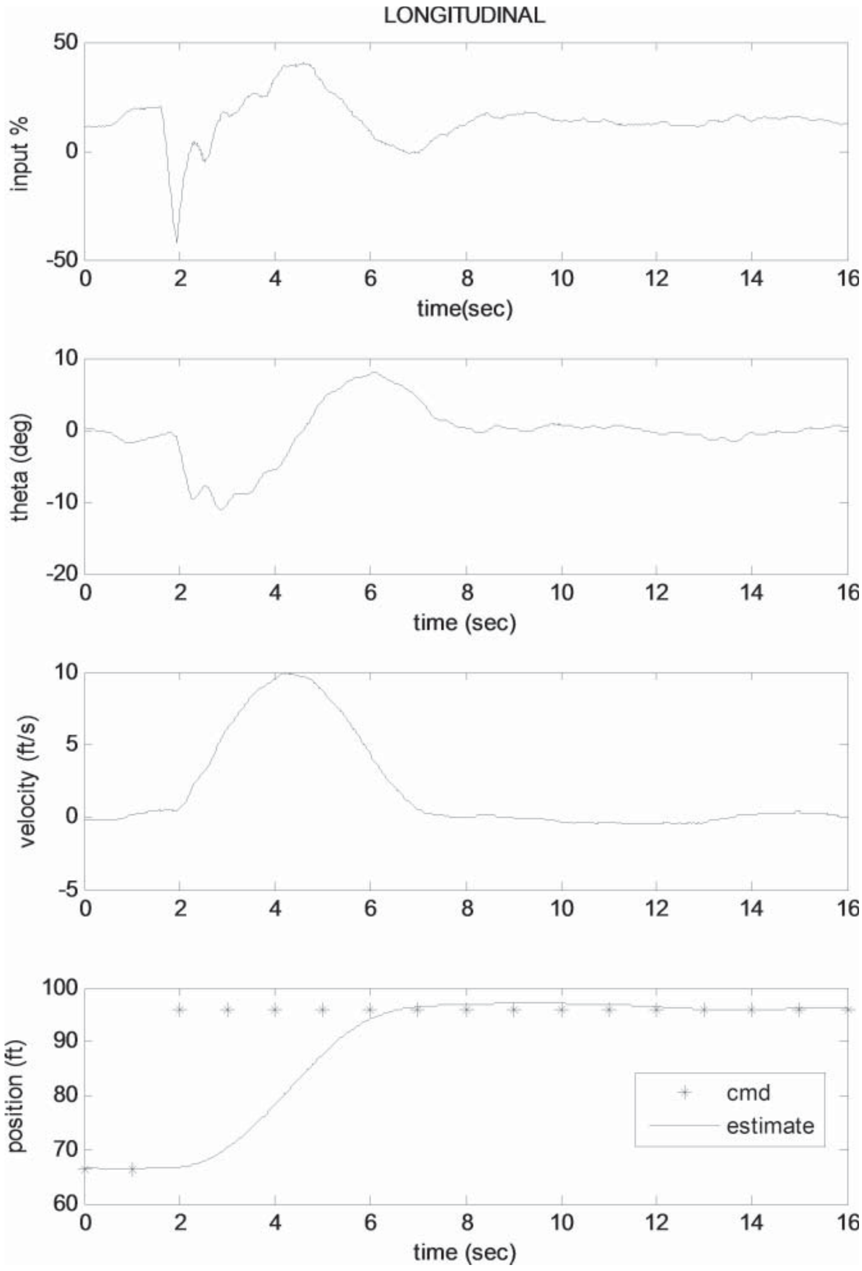

Figure 9.5-19 Adaptive GNC 30 ft longitudinal position command step response (forward) from flight test of RMAX.

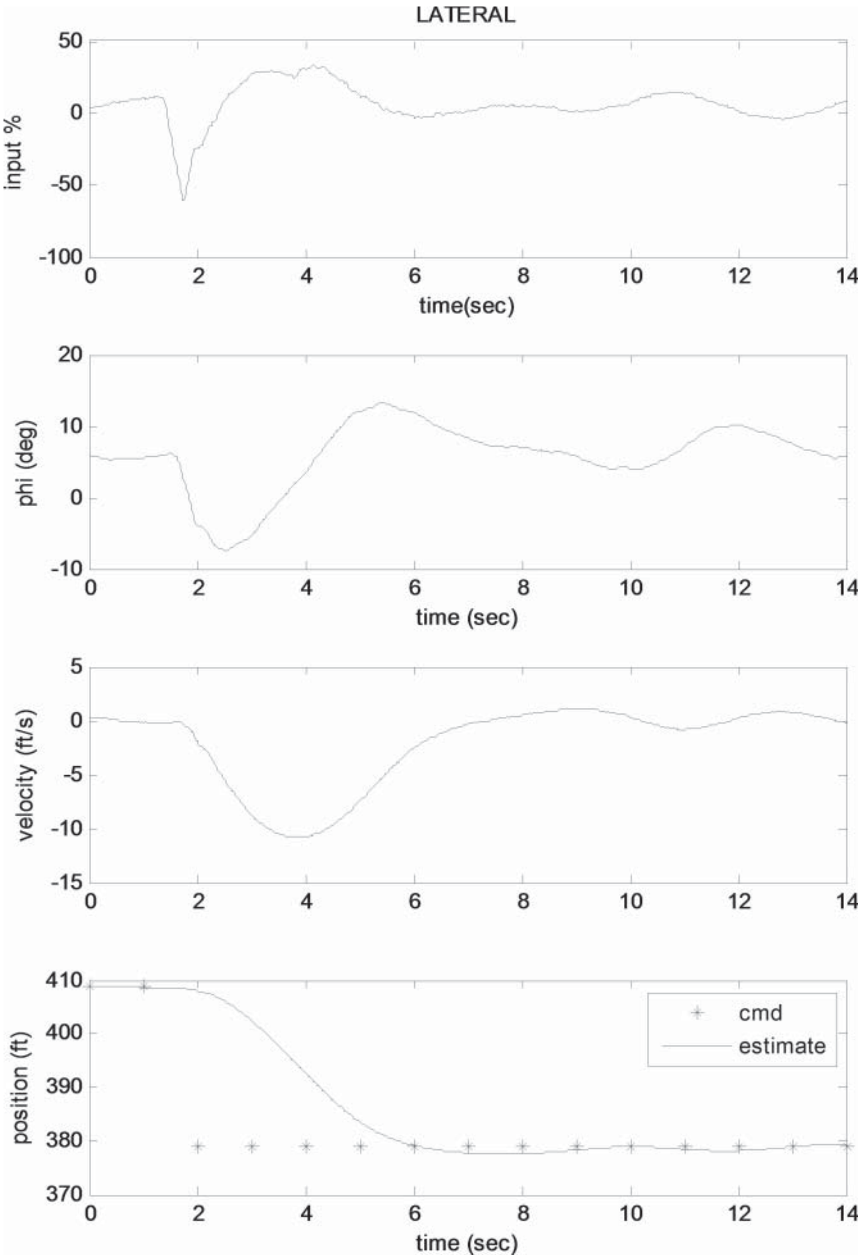

Figure 9.5-20 Adaptive GNC 30 ft lateral position command step response (left) from flight test of RMAX.

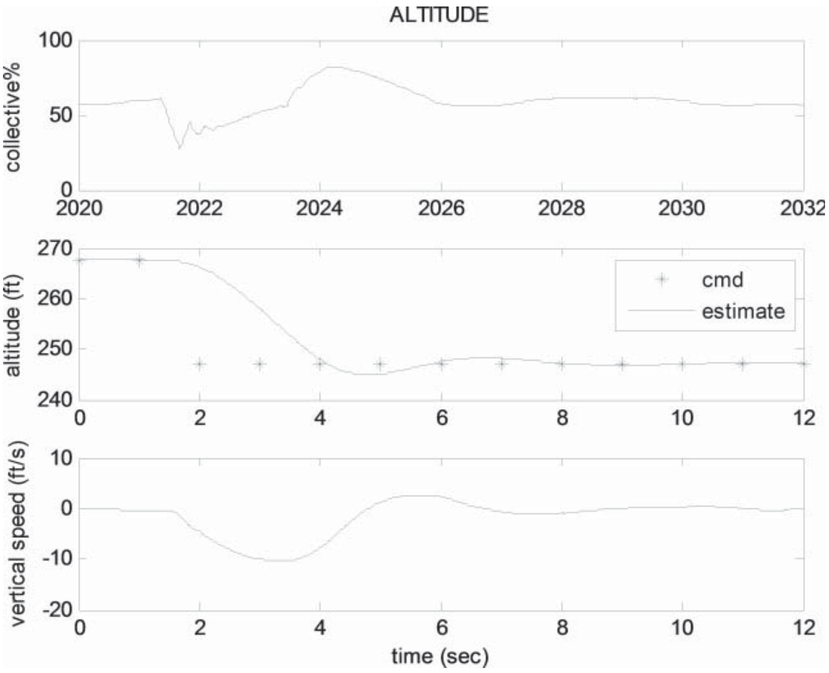

Figure 9.5-21 Adaptive GNC 20 ft altitude command step response (down) from flight test of RMAX.

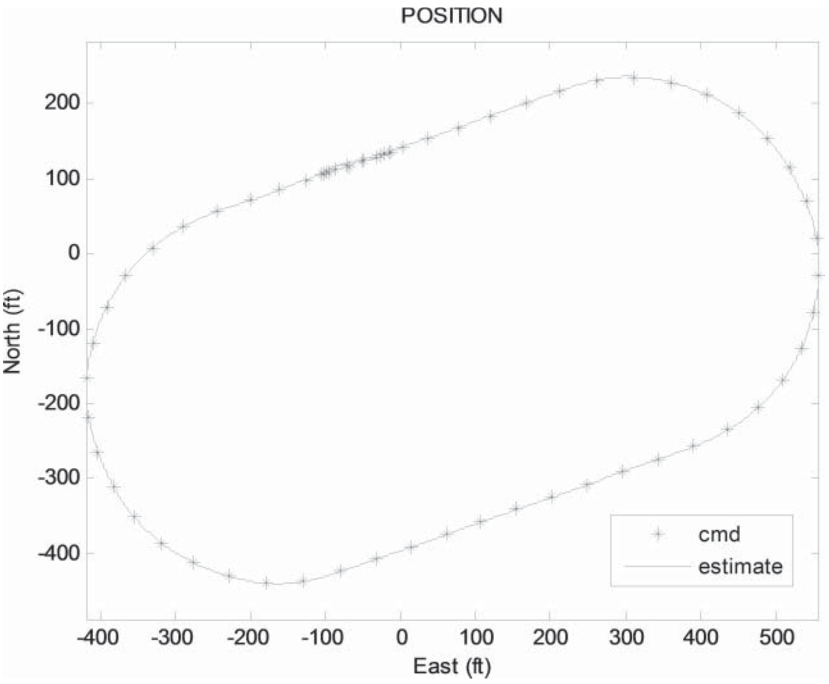

Figure 9.5-22 Adaptive GNC tracking during racetrack pattern at 50 ft/s from flight test of RMAX.

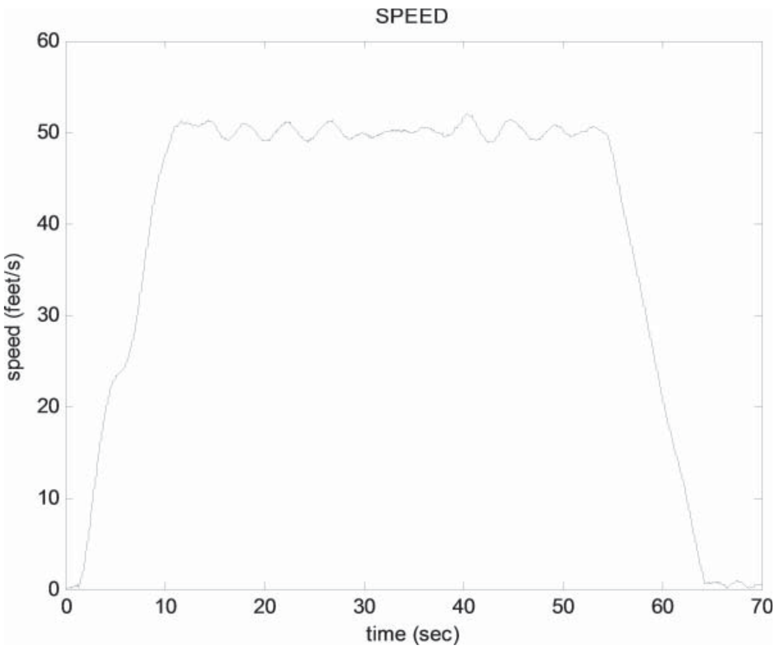

Figure 9.5-23 Adaptive GNC speed during racetrack pattern from flight test of RMAX.

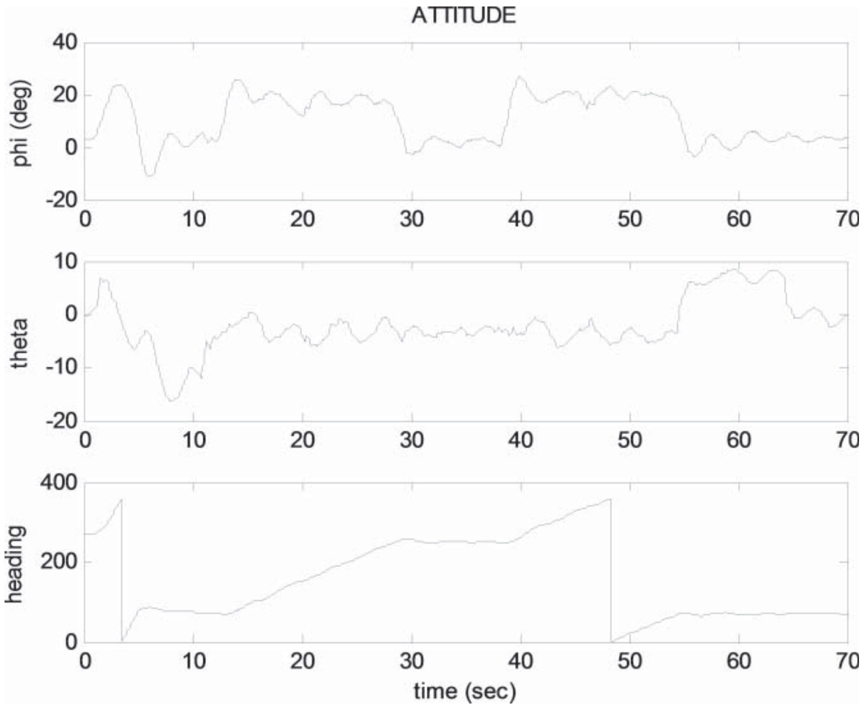

Figure 9.5-24 Adaptive GNC altitude during racetrack pattern from flight test of RMAX.

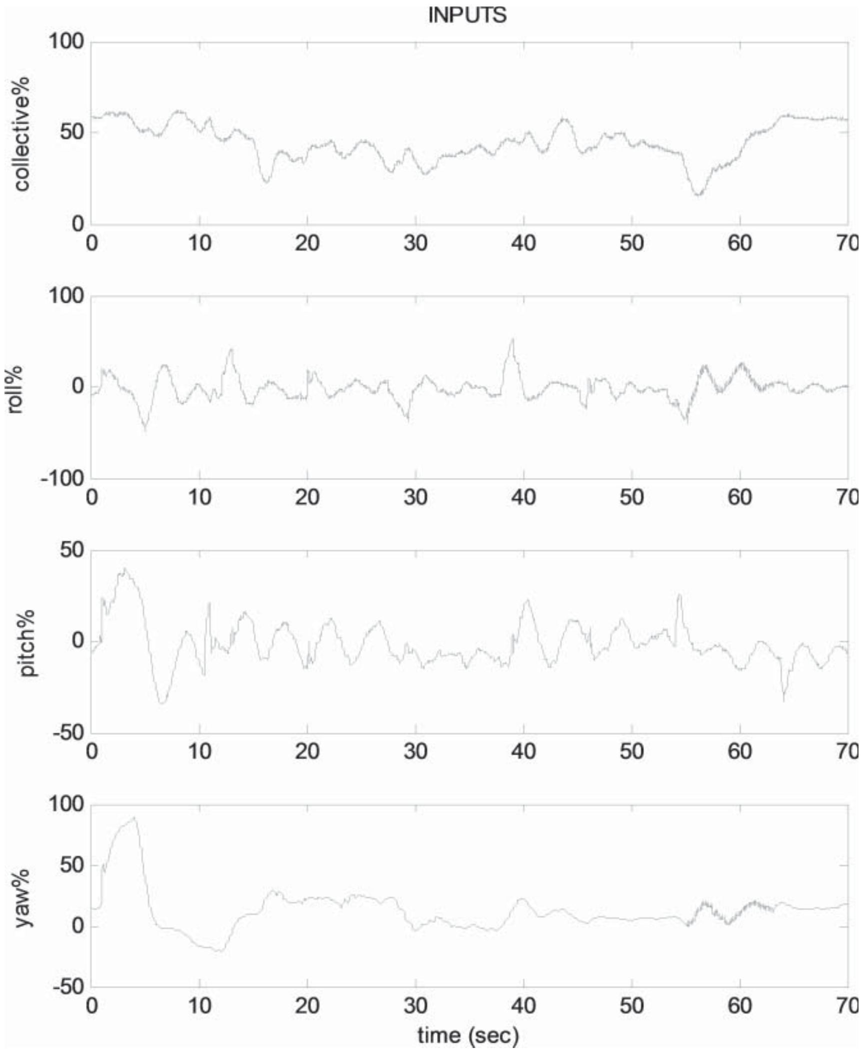

Figure 9.5-25 Adaptive GNC inputs during racetrack pattern from flight test of RMAX.

This corresponds approximately to what should theoretically be necessary for the selected acceleration for the turns of 10 ft/s^2. Figure 9.5-25 shows the vehicle inputs during this pattern. The large yaw input at the start is due to the need to get the aircraft headed in the right direction. That is, the flight starts with a rapid change in the heading command.

A video of a racetrack pattern similar to the one corresponding to the data is available at: http://uav.ae.gatech.edu/videos/f070911a1_racetrack50.wmv

9.6 SUMMARY

In this chapter, we developed the concept of including adaptation of the vehicle dynamics model in the design of a control system. This was done specifically in the context of dynamic inversion. The resulting controller is an example of model reference adaptive control (MRAC). We utilized an artificial NN as a framework for performing an online curve fit of the model error in our nominal dynamic inversion.

Following this, we included a description of methods to handle some of the most significant issues in adaptive control: input saturation and enabling our control systems to have a cascaded structure. Taken together, these ideas allowed the development of an adaptive guidance and flight control system that not only is able to fly an aircraft in order to follow a prescribed position/velocity trajectorybut also enables it to be adaptive in all six degrees of freedom.

This architecture was included in a full guidance, navigation, and control system design for a small helicopter, tying back to concepts from Chapters 5, 6, and 7. The discussion included issues of time delay, update rate, sensor mounting locations, sensor noise, and other related matters.

Simulation and the flight test results of this system were then covered, illustrating a system capable of high-bandwidth position tracking for normal aircraft maneuvers. The illustrated simulation and flight testing also included deliberate discontinuities in the desired trajectory command to verify the approach and implementation.

REFERENCES

Bierman, G. J. *Factorization Methods for Discrete Sequential Estimation*. Mineola, NY: Dover, 1977.

Brinker, J., and K. Wise. "Flight Testing of a Reconfigurable Flight Control Law on the X-36 Tailless Fighter Aircraft." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 24, no. 5 (2001): 903–909.

Calise, A., S. Lee, and M. Sharma. "Development of a Reconfigurable Flight Control Law for Tailless Aircraft." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 24, no. 5 (2001): 896–902.

Christophersen, H. B., et al. "A Compact Guidance, Navigation, and Control System for Unmanned Aerial Vehicles." *AIAA Journal of Aerospace Computing, Information, and Communication*, Washington, D.C.: AIAA, 3, no. 5 (May 2006): 187–213.

Cybenko, G. "Approximation by Superpositions of a Sigmoidal Function." *Mathematics of Control, Signals, and Systems* 2, no. 4 (1989): 303–314.

Das, A., F. L. Lewis, and K. Subbarao. "Backstepping Approach for Controlling a Quadrotor using Lagrange Form Dynamics." *Journal of Intelligent and Robotic Systems* 56, nos. 1–2 (September 2009): 127–151.

Dydek, Z. T., A. M. Annaswamy, and E. Lavretsky. "Adaptive Control and the NASA X-15-3 Flight Revisited." *IEEE Control Systems Magazine* (June 2010): 32–48.

Gelb, A. *Applied Optimal Estimation*. Cambridge, Mass.: M.I.T. Press, 1974.

Grewal, M. S., and A. P. Andrews. *Kalman Filtering Theory and Practice*. Englewood Cliffs, N.J.: Prentice Hall, 1993.

Johnson, E. N., and A. J. Calise. "Limited Authority Adaptive Flight Control for Reusable Launch Vehicles." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 26, no. 6 (2003): 906–913.

Johnson, E. N., and S. K. Kannan."Adaptive Control with a Nested Saturation Reference Model." *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Washington, D.C.: AIAA, (August 2003): 1–11.

Johnson, E. N., and S. K. Kannan. "Adaptive Trajectory Control for Autonomous Helicopters." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 28, no. 3 (May/June 2005): 524–538.

Johnson, E. N., and M. A. Turbe. "Modeling, Control, and Flight Testing of a Small Ducted Fan Aircraft." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 29, no. 4 (July/August 2006): 769–779.

Johnson, E. N., et al. "Flight Test Results of Autonomous Fixed-Wing Transition to and from Stationary Hover." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 31, no. 2 (March/April 2008): 358–370.

Kim, Y. H., and F. L. Lewis. *High-Level Feedback Control with Neural Networks*. Singapore: World Scientific, 1998.

Lavretsky, E., and K. A. Wise. *Robust and Adaptive Control*. Springer, 2013.

Lewis, F. L., A. Yesildirek, and K. Liu. "Multilayer Neural-Net Robot Controller with Guaranteed Tracking Performance." *IEEE Transactions on Neural Networks* 7, no. 2 (March 1996): 388–399.

Lewis, F. L., K. Liu, and A. Yesildirek. "Neural Net Robot Controller with Guaranteed Tracking Performance." *IEEE Transactions on Neural Networks* 6, no. 3 (May 1997).

Lewis, F. L, S. Jagannathan, and A. Yesildirek. *Neural Network Control of Robot Manipulators and Nonlinear Systems*. Taylor & Francis, 1999.

Sharma, M., A. Calise, and J. Corban. "Application of an Adaptive Autopilot Design to a Family of Guided Munitions." *Proceedings of the AIAA Guidance, Navigation, and Control Conference,* Washington, D.C.: AIAA, (August 2000): 1–9.

Teel, A. R. "Global Stabilization and Restricted Tracking for Multiple Integrators with Bounded Controls." *Systems & Control Letters* 18 (1992): 165–171.

Wie, B., and P. M. Barba. "Quaternion Feedback for Spacecraft Large Angle Maneuvers." *AIAA Journal of Guidance, Control, and Dynamics*, Washington, D.C.: AIAA, 8, no. 3 (1985): 360–365.

Yesildirek, A., and F. Lewis. "Feedback Linearization Using Neural Networks." *Automatica* 31, no. 11 (1995): 1659–1664.

PROBLEMS

Section 9.2

9.2-1 Develop a controller design within the architecture described by Figure 9.2-1 for the case where the aircraft dynamics are known exactly and the adaptive element is not necessary. The true dynamics are $\dot{x} = u + \sin x$ using the same nomenclature as the figure. Your answer should include:

(a) An appropriate reference model

(b) An appropriate linear feedback

(c) A nominal dynamic inverse

9.2-2 Using your answer from Problem 9.2-1, simulate the system for a step input of the set point for the reference model.

9.2-3 Using your answer from Problem 9.2-1, simulate the system for a step input of model error introduced as Δ in the dynamics $\dot{x} = u + \sin x + \Delta$.

9.2-4 Consider the case of a second order plant of the form $m\ddot{x} = -kx - c\dot{x} + u$, where m, k, and c represent spring and damper coefficients as indicated. Treat pseudocontrol (v) as desired $\ddot{x}$. Write the appropriate nominal inverse for this system with v and $\dot{x}$ as inputs.

9.2-5 In Chapter 4 (Equation 4.2-6), an approximation to the short-period mode was found to be

$$\begin{bmatrix} V_T & 0 \\ -M_{\dot{\alpha}} & 1 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} V_T & 0 \\ -M_{\alpha} & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ q \end{bmatrix} + \begin{bmatrix} Z_{\delta_e} \\ M_{\delta_e} \end{bmatrix} \delta_e$$

Write an appropriate dynamic inverse with as inputs. Note that the pseudocontrol in this cases consists of both $\dot{\alpha}$ and $\dot{q}$. The input to the system is elevator deflection, δ_e.

Section 9.3

9.3-1 One method to train an artificial NN is to iteratively adjust the interconnection weights based on a set of training data. Consider the true function $f(x) = \sin x$. Train a nonparametric NN to approximate this function as close as possible using a training set with 100 randomly selected values for x between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$. Use an SHL structure with five hidden layer neurons. Use an initial guess of zero for all interconnection weights (W and V). Then, go through each of the 100 points adjusting the weights. For each point, there is a model reconstruction error of $r = W^T\sigma(V^T\overline{x}) - \sin x$. Use this error as a basis to adjust the weights by the rule

$$\dot{W} = -r\sigma^T\gamma_W$$
$$\dot{V} = -\sigma' W^T r \overline{x}^T \gamma_V,$$

where γ_W and γ_V are scalar learning rates to be chosen to optimize convergence. Repeat this process (an iteration) until convergence occurs. Make a plot of the total RMS reconstruction error across the 100 points as a function of iteration number. Plot the output of your final NN compared to the function above using your final "best" interconnection weights.

9.3-2 Perform a trade study on Example 9.3-1 to confirm the choice of controller parameters. In each case, indicate any issues that arise with an extreme increase or decrease of that parameter. Make a final recommendation for each. Provide plots similar to Figures 9.3-2 and 9.3-3 for your final choices.

(a) Outer layer learning rate Γ_W

(b) Inner layer learning rate Γ_V

(c) Number of hidden layer neurons N

9.3-3 Modify Example 9.3-1 to create a new related example that represents pitch *angle* control rather than pitch-rate control. This will make the system second order rather than first order. That is, pitch angle is just the integral of pitch rate. There is still only one pseudocontrol however. The linear feedback will now have two gains, of the form $K_P e + K_D \dot{e}$. An appropriate adaptive law to use is (Johnson and Calise, 2003)

$$\dot{W} = -[(\sigma - \sigma' V^T \overline{x}) r^T + \lambda \|e\| W]\Gamma_W$$
$$\dot{V} = -\Gamma_V[\overline{x} r^T W^T \sigma' + \lambda \|\|e\| V]$$

with

$$r = \frac{K_P K_D e/2 + K_P \dot{e}}{0.25(N-1)+1}$$

Section 9.4

9.4-1 Explain why it is not possible to do a simple dynamic inverse of a system that has an input saturation nonlinearity.

9.4-2 Modify Example 9.4-1 to include an actuator rate limit of 0.1 rad/s. Include plots similar to Figures 9.4-2 and 9.4-3.

9.4-3 Modify Example 9.4-1 for the case of *quantized* control. That is, the actuator can only take on values of u equal to –0.1, –0.05, 0, 0.05, and 0.1. Implement this by leaving the nominal dynamic inverse alone and putting a filter after it to quantize the control (presumably by picking the value closest to the desired value). It may be necessary to modify the reference model and linear feedback to achieve good responses. Note that the resulting problem is similar to the use of a reaction control system for attitude control—typical for a spacecraft.

(a) Provide plots similar to Figures 9.4-2 and 9.4-3 for your solution.

(b) Add a plot that includes v_{ad} and $f(x,u) - \hat{f}(x,u)$ as a function of time.

9.4-4 PCH has some interesting properties when there is an error in the actuator limits. Repeat Example 9.4-1, but this time have the limit assumed in the controller be 0.12 and the true limit be 0.1 as before.

(a) Provide plots similar to Figures 9.4-2 and 9.4-3 for your solution.

(b) Add a plot that includes v_{ad} and $f(x,u) - \hat{f}(x,u)$ as a function of time.

Section 9.5

9.5-1 Explain why the separation principle cannot be invoked in order to design the state estimation scheme and the adaptive controller utilized in this section separately.

9.5-2 In one system design, it takes 20 ms to get the sensor inputs, 20 ms to compute the actuator commands, and 50 ms to move the actuators. In a second design, it takes 50 ms to get the sensor inputs, 20 ms to compute the actuator commands, and 20 ms to move the actuators. Make the argument that these systems are equivalent in terms of analyzing the true motion of the aircraft that results.

9.5-3 Why is it important to halt NN adaptation when the aircraft is not flying (sitting still on the ground)?

9.5-4 The simulation and flight test results indicated good performance at both hover and high-speed flight with the same controller design. The normal practice is to develop more than one controller designs for different flight speeds for a helicopter. Discuss some reasons why it may not have been necessary in this instance.