Aerodynamic Decelerator Systems Technology Conferences
30 Mar - 2 Apr 2015, Daytona Beach, FL
23rd AIAA Aerodynamic Decelerator Systems Technology Conference

10.2514/6.2015-2107

Adaptive Control of a 10K Parafoil System

Eve Law*
C.S. Draper Laboratory, 555 Technology Square, Cambridge, MA 02139

Guided parafoil systems lacking propulsion pose significant c ontrol challenges. Landing accurately is challenging, and guidance strategies demand control signals be kept stable in the face of noise and uncertain dynamics. Larger systems present additional hurdles leading to an interest adaptive control. Conventional pole-placement based non-adaptive and adaptive control are compared to $\mathcal{L}_1$ adaptive control on a dynamic parafoil model. The results illustrate the advantages and disadvantages of each approach, and provide design and implementation guidelines for future work.

I. Introduction

Large parafoil systems (on the order of 10,000 lbs) pose control challenges, including a potentially dangerous aerodynamic mode (dutch-roll) at low frequency, large motor lag, varying dynamics from system to system as a result of motor wear, frequent hardware changes, motor saturation, and more prominent non-minimum phase behavior than smaller systems. Operational constraints dictate fewer flight tests for system identification–each system takes on the order of days to rig. As a result, adaptive control is an appealing method to address some these challenges.

Given the constraints outlined above, achieving "perfect" tracking is less critical than maintaining system stability. In this context, ensuring control signals do not contain frequency content too near or above the naturally unstable dutch-roll is crucial. If this mode is excited, the system responds drastically differently from the nominal model. In an adaptive control context, this is especially problematic, given the large transients and high frequency commands generally associated with adaptive schemes.

An approach that filters the control signal to separate the high frequency adaptive estimators from directly impacting the system is desirable. A relatively recent method, referred to as $\mathcal{L}_1$ adaptive, is designed with just such considerations, and presents an interesting alternative to more conventional adaptive schemes.

*Graduate Research Assistant, Boston University, Department of Mechanical Engineering, evelaw427@gmail.com

This study will consider $\mathcal{L}_1$ adaptive control, as compared with both conventional adaptive control, as well as non-adaptive control, in tracking representative signals for a large parafoil system. The contribution of this paper is the complete discretization of the $\mathcal{L}_1$ adaptive control algorithm suitable for real-time implementation on embedded parafoil hardware. In addition, the $\mathcal{L}_1$ adaptive control algorithm is directly compared with adaptive pole placement on representative parafoil dynamics in a number of challenging scenarios, providing insight into the advantages and disadvantages of both algorithms.

The rest of this paper is organized as follows. Section II describes the model for the 10K parafoil. Section III introduces and describes $\mathcal{L}_1$ Adaptive control in detail. Section IV briefly describes adaptive pole placement control. Section V presents comparison results which are discussed in section VI.

II. System Description

The toggle-command to heading rate of a large parafoil system (henceforth 10K) can be modeled as a linear time-invariant (LTI) system of the form:

$$A(s) = \frac{(s+a)(s+b)}{(s+c)(s^2+2\zeta_A\omega_A+\omega_A^2)} \tag{1}$$

where the constants are known to be approximately:

$$\begin{aligned} a &\approx 0.35 \\ b &\approx -1.44 \\ c &\approx 0.23 \\ \omega_A &\approx \frac{2\pi}{10} \\ \zeta_A &\approx 0.19 \end{aligned} \tag{2}$$

The dutch-roll mode is captured in the underdamped system of frequency ω_A. While the value of ω_A is known to be reasonably accurate, system identification for the other values varies significantly from one flight to another.

III. $\mathcal{L}_1$ Adaptive

A method referred to as $\mathcal{L}_1$ adaptive control is a relatively recent development in the literature, and has developed as a result of considerations similar to those posed above: "protecting" systems from the transient behavior characteristic of most adaptive strategies by using filtering. A full treatment of $\mathcal{L}_1$ is beyond the scope of the present discussion, but the essential features will be reviewed. We will focus on the output-feedback formulation of $\mathcal{L}_1$ control. This discussion closely follows the development in §4 of [1].

III.A. Problem Formulation

Consider the following single-input single-output (SISO) plant:

$$y(s) = A(S)(u(s) + d(s)) \tag{3}$$

where $u(s)$, $y(s)$, and $d(s)$ are scalars corresponding to system input, output, and disturbance. $A(s)$ is a strictly-proper unknown system, and the time domain version of the disturbance $d(t, y)$ is Lipschitz continuous:

$$|d(t, y_1) - d(t, y_2)| \leq L|y_1 - y_2|,\ |d(t, y)| \leq L|y| + L_0 \tag{4}$$

for arbitrarily large $L > 0$ and $L_0 > 0$.
The objective is to track a bounded reference input $r(t)$ following an ideal reference plant model $M(s)$, which is a strictly proper minimum phase stable transfer function.
The system (3) can be rewritten as:

$$y(s) = M(s)(u(s) + \sigma(s)) \tag{5}$$
$$\sigma(s) = \frac{(A(s) - M(s))u(s) + A(s)d(s)}{M(s)} \tag{6}$$

Where the $\sigma(s)$ term lumps together both the unknown plant dynamics and disturbance. In the time domain this can be represented as a minimal, controllable, observable, stable system:

$$\begin{aligned} \dot{x}(t) &= A_m x(t) + B_m(u(t) + \sigma(t)),\ x(t) = 0 \\ y(t) &= C_m x(t) \end{aligned} \tag{7}$$

where $x \in \mathbb{R}^n$.
The role of the control $u(s)$ will be to cancel out all these uncertainties, and to do so with bounded transient behavior. A natural choice of control form is:

$$u(s) = C(s)(r(s) - \sigma(s)) \tag{8}$$

where $C(s)$ is a stable filter with unity gain to be chosen. Some manipulation enables writing (3) in yet another way:

$$y(s) = H(s)\left(C(s)r(s) + (1 - C(s))d(s)\right) \tag{9}$$

where

$$H(s) = \frac{A(s)M(s)}{C(s)A(s) + (1 - C(s))M(s)} \tag{10}$$

is stable and strictly proper, and an $\mathcal{L}_1$-norm condition is satisfied (hence the name):

$$\|H(s)(1 - C(s))\|_{\mathcal{L}_1} L = \|G(s)\|_{\mathcal{L}_1} L < 1 \tag{11}$$

From the structure of (9), it is clear the choice of $u(s)$ in the form of (8) means the closed loop system will not match the "ideal" reference model $M(s)$ perfectly, but will rather represent a filtered version. We will refer to this system, where the uncertainty is entirely cancelled out within filter $C(s)$ bandwidth, corresponding to the *achievable control objective* for $\mathcal{L}_1$ control, as y_{ref}. If $C(s) = 1$ in expressions (9) and (10) shows $H(s)$ reduces to $M(s)$ exactly.

III.B. Controller Structure

The $\mathcal{L}_1$ controller is composed of an output-predictor, controller, and an adaptation law.

Output Predictor Since we are lumping all plant dynamics and disturbance into the σ term, the most general form this can appear is an "unmatched" form:

$$\begin{aligned} \dot{\hat{x}}(t) &= A_m \hat{x}(t) + B_m u(t) + \hat{\sigma}(t),\ \hat{x}(0) = 0 \\ \hat{y}(t) &= C_m \hat{x}(t) \end{aligned} \tag{12}$$

This choice for the form of $\hat{\sigma}(t)$ will complicate the form of the control law.

III.B.1. Control

The control law is:

$$u(s) = C(s)r(s) - C(s)\frac{c_m(sI - A)^{-1}}{c_m(sI - A)^{-1}b_m}\hat{\sigma} \tag{13}$$

This peculiar form results from the choice of $\hat{\sigma}$ as being "unmatched", to transform it from a vector into a scalar ($\mathbb{R}^n \to \mathbb{R}$). Effectively, the second term is attempting to calculate something like B_m^{-1}.

For practical implementation, this second term can be created in the following manner. By making a state-space model using A_m, C_m, and letting the B matrix be I_n, the n-dimensional identity matrix, the numerator term can be created. Both this model and the regular state-space model for $M(s)$ can be transformed into discrete transfer functions, and n new transfer functions can be formed out of the numerators of both systems (since both have the same poles). This cascade of discrete transfer functions can be be put back into a discrete state-space form. Discrete systems are used, since the structure of this term would result in improper transfer functions (or descriptor continuous state-space models).

III.B.2. Adaptation Law

Define the error dynamics between the output predictor (12) and the plant in state space form (7), or $\hat{x} - x$, as:

$$\begin{aligned} \dot{\tilde{x}}(t) &= A_m \tilde{x}(t) + \hat{\sigma}(t) - B_m \sigma(t),\ \tilde{x}(t) = 0 \\ \tilde{y}(t) &= C_m \tilde{x}(t) \end{aligned} \tag{14}$$

The adaptation law is given in terms of the CPU sampling time T_s of the system running the adaptation at time steps $i = 0, 1, 2, ..$, where the estimate is held at each time step.
With the error signal $\tilde{y}$ as in (14), the adaptation law for $\hat{\sigma}$ is:

$$\begin{aligned} \hat{\sigma}(t) &= \hat{\sigma}(iT_s),\ t \in [iT_s, (i+1)T_s) \\ \hat{\sigma}(iT_s) &= \Gamma(T_s)\tilde{y}(iT_s) \end{aligned} \tag{15}$$

where $\Gamma(T_s)$ is a precomputed constant defined as:

$$\Gamma(T_s) = -\Phi^{-1}(T_s) e^{\Lambda A_m \Lambda^{-1} T_s} \mathbb{1}_1 \tag{16}$$

where $\mathbb{1}_1 = [1, 0, ..., 0]^T \in \mathbb{R}^n$, and

$$\Phi(T_s) = \int_0^{T_s} e^{\Lambda A_m \Lambda^{-1}(T_s - \tau)} \Lambda \, d\tau \tag{17}$$

and

$$\Lambda = \begin{bmatrix} C_m \\ D\sqrt{P} \end{bmatrix} \tag{18}$$

where $P = P^T > 0$ satisfies the algebraic Lyapunov equation:

$$A_m^T P + P A_m = -Q,\ Q = Q^T > 0 \tag{19}$$

and D is an $(n-1) \times n$ matrix containing the null space of $c_m(\sqrt{P})^{-1}$. The structure of P ensures $\sqrt{P}$ can always be found (using Cholesky decomposition, for example). This construction ensures Λ^{-1} exists.

For implementation the following simplification is helpful:

$$\Phi = \Lambda A_m \Lambda^{-1}(e^{\Lambda A_m \Lambda^{-1} T_s} - I)\Lambda \tag{20}$$

This whole structure seems quite complicated, but, as expected, is helpful in proving stability, as will be described in §III.D. This form does not require parameter projection, like many other adaptive laws, including other formulations for $\mathcal{L}_1$ control.

III.C. Discrete Time Implementation of Output-Predictor

The formulations presented thus far, following [1], use continuous time transfer functions and state-space models. Guided airdrop systems run control loops relatively slowly, on the order of 1-10 Hz. As a result, continuous time represents a poor approximation to real system behavior. To examine the applicability of $\mathcal{L}_1$ in this context, all expressions are discretized, including the output-predictor.

The non-matched formulation in §III.B requires careful transformation from continuous to discrete time. We proceed using a procedure from [2] for converting continuous systems to discrete time using zero-order-hold sampling.

Recall the expression for the output-predictor:

$$\begin{aligned} \dot{\hat{x}}(t) &= A_m \hat{x}(t) + B_m u(t) + \hat{\sigma}(t) \\ \hat{y}(t) &= C_m \hat{x}(t) \end{aligned} \tag{21}$$

Using the variation of constants formula, expressing the effects of sampling the control signal $u(t_k)$ and adaptation term $\hat{\sigma}(t_k)$ for discrete steps k over $t_k \leq t \leq t_k$ is straightforward:

$$\hat{x}(t) = e^{A_m(t-t_k)}\hat{x}(t_k) + \int_{t_k}^{t} e^{A_m(t-s')}(B_m u(s') + \hat{\sigma}(s'))ds' \tag{22}$$

$$= e^{A_m(t-t_k)}\hat{x}(t_k) + \int_{t_k}^{t} e^{A_m(t-s')}ds'(B_m u(t_k) + \hat{\sigma}(t_k)) \tag{23}$$

$$= e^{A(t-t_k)}\hat{x}(t_k) + \int_{0}^{t-t_k} e^{A_m(s)}ds(B_m u(t_k) + \hat{\sigma}(t_k)) \tag{24}$$

$$= \Phi(t, t_k)\hat{x}(t_k) + \Gamma^*(t, t_k)(B_m u(t_k) + \hat{\sigma}(t_k)) \tag{25}$$

where the second expression reflects u and $\hat{\sigma}$ taking constant values over the sampling time. We use the symbol Γ^* to distinguish from the traditional Γ term, which incorporates B. We treat the system as having input u and output y sampled at the same instants. Assuming sampling with constant period T_s, giving $t_k = kT_s$, allows us to write the difference equation for this system as:

$$\begin{aligned}\hat{x}((k+1)T_s) &= \Phi\hat{x}(kT_s) + \Gamma^*(B_m u(kT_s) + \hat{\sigma}(kT_s)) \\ \hat{y}(kT_s) &= C_m\hat{x}(kT_s)\end{aligned} \tag{26}$$

where

$$\begin{aligned}\Phi &= e^{AT_s} \\ \Gamma^* &= \int_0^{T_s} e^{As} ds\end{aligned} \tag{27}$$

From (27) we have:

$$\frac{d\Phi(t)}{dt} = A\Phi(t) = \Phi(t)A \tag{28}$$

$$\frac{d\Gamma^*(t)}{dt} = \Phi(t) \tag{29}$$

Matrices Φ and Γ satisfy the expression:

$$\frac{d}{dt}\begin{bmatrix}\Phi(t) & \Gamma^*(t) \\ 0 & I\end{bmatrix} = \begin{bmatrix}\Phi(t) & \Gamma^*(t) \\ 0 & I\end{bmatrix}\begin{bmatrix}A & I \\ 0 & 0\end{bmatrix} \tag{30}$$

Where the identity matrices I are appropriately sized. To explicitly solve for $\Phi(T_s)$ and $\Gamma^*(T_s)$ the following block matrix can be evaluated:

$$\begin{bmatrix}\Phi(T_s) & \Gamma^*(T_s) \\ 0 & I\end{bmatrix} = \exp\left(\begin{bmatrix}A & I \\ 0 & 0\end{bmatrix}T_s\right) \tag{31}$$

Using (31) the entire $\mathcal{L}_1$ algorithm, including the output-predictor in (21), can be implemented in discrete time, as shown in Figures 1 and 2.

III.D. Main Boundedness/Stability Result and Proof Outline

MAIN RESULT Restating the entire proof from [1] is beyond the scope of the current discussion. Rather, we state the main result, and briefly outline the key steps involved in the proof. Once again, this section draws heavily from [1].

Theorem 1. *Consider the system in (1) and the $\mathcal{L}_1$-controller defined in the previous section. If*

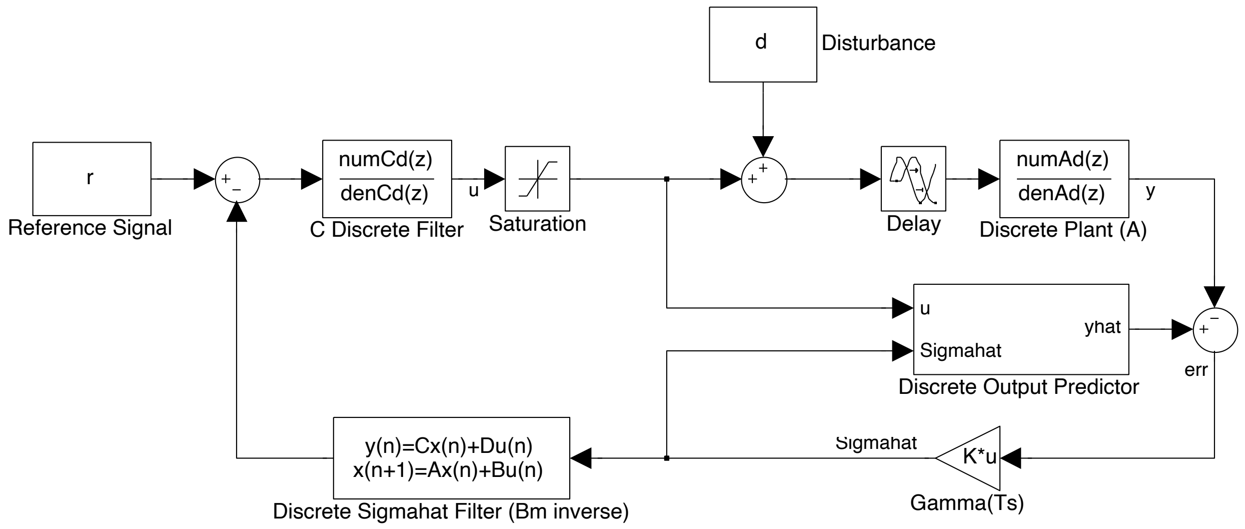

Figure 1. $\mathcal{L}_1$ **Adaptive Control Structure, discrete time.**

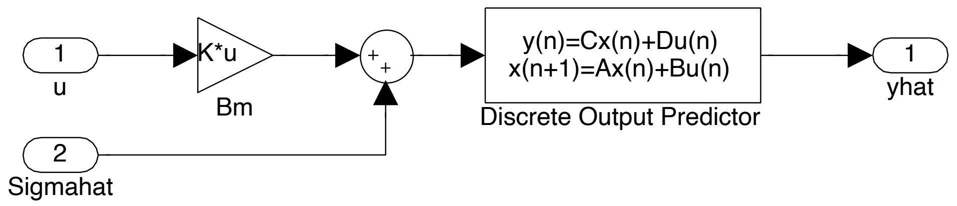

Figure 2. $\mathcal{L}_1$ **Adaptive Output Predictor Structure, discrete time. Note the way the** $\hat{\sigma}$ **term is augmented to the reference model dynamics, representing "unmatched" uncertainty (see §III.C).**

the CPU sampling time T_s *is chosen such that*

$$\gamma_0(T_s) < \bar{\gamma}_0 \tag{32}$$

for some positive constant $\bar{\gamma}_0$*, then:*

$$\|\tilde{y}\|_{\mathcal{L}_\infty} < \bar{\gamma}_0 \tag{33}$$

$$\|y_{ref} - y\|_{\mathcal{L}_\infty} < \gamma_1 \tag{34}$$

$$\|u_{ref} - u\|_{\mathcal{L}_\infty} \leq \gamma_2 \tag{35}$$

where

$$\gamma_1 \equiv \frac{\|H(s)C(s)/M(s)\|}{1-\|G(s)\|_{\mathcal{L}_1}L}\bar{\gamma}_0 \tag{36}$$

$$\gamma_2 \equiv \left\|\frac{H(s)C(s)}{M(s)}\right\|_{\mathcal{L}_1} L\gamma_1 + \left\|\frac{H(s)C(s)}{A(s)M(s)}\right\|_{\mathcal{L}_1}\bar{\gamma}_0 \tag{37}$$

and $H(s)$ is given in (10).

In essence, this theorem states the tracking error between the real system and achievable $\mathcal{L}_1$ reference system (y_{ref}) is bounded by a constant determined by T_s, as are the magnitude of control signals. Since the reference achievable $\mathcal{L}_1$ system is stable and designed with desired performance, the adaptive controller will perform similarly, with uniform bounds, even during the transient phase. In addition, these bounds can be reduced by decreasing the CPU sampling time.
Also noteworthy is the lack of requirements on the reference signal, with the exception of being bounded–no requirements of sufficient richness are needed to ensure the results.
As stated earlier, however, the reference achievable $\mathcal{L}_1$, as defined in (9) and (10), does not track with the same behavior as the "ideal" reference system $M(s)$, thus the adaptive controller will not either. The filter $C(s)$ has the effect of canceling only uncertainties within its bandwidth, and, as expected, introducing a filter will cause some phase delay. The issue of designing the filter is discussed in §III.E.

III.D.1. Outline of Proof

At this point we will outline the essential steps used in proving Theorem 1.
The first step is to demonstrate $\|\tilde{y}\|_{\mathcal{L}_\infty} < \bar{\gamma}_0$.
Introduce a state transformation:

$$\tilde{\xi} = \Lambda\tilde{x} \tag{38}$$

which gives the following alternative expression for the error dynamics (14):

$$\dot{\tilde{\xi}} = \Lambda A_m \Lambda^{-1}\tilde{\xi} + \Lambda\hat{\sigma}(t) - \Lambda B_m \sigma(t),\ \tilde{\xi}(0) = 0 \tag{39}$$

$$\tilde{y} = \tilde{\xi}_1(t) \tag{40}$$

where $\tilde{\xi}_1$ is the first element of $\tilde{\xi}(t)$. Demonstrating $\tilde{\xi}$ remains bounded will thus demonstrate $\|\tilde{y}\|_{\mathcal{L}_\infty} < \bar{\gamma}_0$.
Noting (39) corresponds to a linear system with exogenous input forcing functions $\Lambda\hat{\sigma}(t)$ and $-\Lambda B_m\sigma(t)$, we can see the standard variation of constants solution will involve an integral for each term, and from the structure of $\tilde{\xi}$, with its first element being $\tilde{y}$, we can express the solution

to (39) as the sum of two components:

$$\tilde{\xi}(iT_s + t) = \chi(iT_s + t) + \zeta(iT_s + t) \tag{41}$$

Re-expressing the argument of $\tilde{\xi}$, we have:

$$\chi((j+1)T_s) = e^{\Lambda A_m \Lambda^{-1} T_s} \begin{bmatrix} \tilde{y}(jT_s) \\ 0 \end{bmatrix} + \int_0^{T_s} e^{\Lambda A_m \Lambda^{-1}(T_s - \tau)} \Lambda \hat{\sigma}(jT_s)\, d\tau \tag{42}$$

$$\zeta((j+1)T_s) = e^{\Lambda A_m \Lambda^{-1} T_s} \begin{bmatrix} 0 \\ \tilde{z}(jT_s) \end{bmatrix} + \int_0^{T_s} e^{\Lambda A_m \Lambda^{-1}(T_s - \tau)} b_m \sigma(jT_s)\, d\tau \tag{43}$$

Substituting the adaptive law for $\hat{\sigma}$ from (15) into (42) gives:

$$\chi((j+1)T_s) = 0 \tag{44}$$

What remains to show is the stability and boundedness of $\zeta((j+1)T_s)$ in (43). The Lyapunov equation

$$V = \zeta^T(t)\Lambda^{-T} P \Lambda^{-1} \zeta(t) \tag{45}$$

can be used to do this, and the resulting bound is a function of T_s, as expected. Thus $\|\tilde{y}\|_{\mathcal{L}_\infty} < \bar{\gamma}_0$ is established.
Combining this result with

$$\|y_{ref} - y\|_{\mathcal{L}_\infty} < \frac{\|H(s)C(s)/M(s)\|}{1 - \|G(s)\|_{\mathcal{L}_1} L} \|\tilde{y}\|_{\mathcal{L}_\infty} \tag{46}$$

which is proven in [1], proves (34), which in turn combines with the Lipschitz assumption for the disturbance to give (35).
For the exact bound forms see [1].

III.E. Filter Design

The results above omit the most crucial design aspect of $\mathcal{L}_1$ control: choosing appropriate reference models M and filters C. Other than appearing in the forms for the bounds of γ_i above, which shows a filter is required for bounded controls for non-minimum phase systems, for instance, the result is not constructive. There are numerous results in the literature for this process for the output-feedback case, including those based on satisfying another $\mathcal{L}_1$ condition, using optimization [3], [1], as well as robust control [4]. For the present study, a method based on pole-placement [5] is pursued.
The idea is to match the poles of the $\mathcal{L}_1$ closed loop system, H given in (10), with another pole-

placement controlled system.
In general, the method of pole placement relies on solving the Diophantine equation:

$$Z(s)P(S) + R(s)Q_m(s)L(s) = A^*_{CL}(s) \tag{47}$$

where the plant is defined as:

$$A(s) = \frac{Z(s)}{R(s)}u \tag{48}$$

and the feedback control to place the poles at $A^*_{CL}(s)$ is:

$$K(s) = \frac{P(s)}{Q_m(s)L(s)} \tag{49}$$

and $Q_m(s)$ is defined as in [6], corresponding to the internal model of the reference signal to be tracked. After some simple manipulation the method of choosing C and M is given by matching the two expressions:

$$\frac{C(s)}{M(s)(1-C(s))} = \frac{C_n(s)M_d(s)}{M_n(s)(C_d(s)-C_n(s))} = \frac{P(s)}{Q_m(s)L(s)} \tag{50}$$

requiring both C and M are stable, and the closed loop system H given in (10) satisfies the $\mathcal{L}_1$ stability condition.
Thus there are three design parameters to play with ($A^*_{CL}(s)$, $C(s)$, $M(s)$), subject to these conditions.
In an ideal setting, more exotic forms for both C and M could be investigated, including Chebyshev and elliptic filter schemes. For the sake of simplicity, however, the structure of $C(s)$ is restricted to a conventional second order low-pass filter:

$$C(s) = \frac{\omega_C^2}{s^2 + 2\zeta_C\omega_C s + \omega_C^2} \tag{51}$$

where the values of ω_C and ζ_C are chosen to correspond to desired control signal filter behavior. With this choice, (50) becomes:

$$\frac{\omega_C^2 M_d(s)}{M_n(s)(s^2 + 2\zeta_C\omega_C s)} = \frac{\omega_C^2 M_d(s)}{M_n(s)s(s + 2\zeta_C\omega_C)} = \frac{P(s)}{Q_m(s)L(s)} \tag{52}$$

A natural choice for $Q_m(s)$ in this context is s, corresponding to tracking of step inputs, giving

(50):

$$\frac{\omega_C^2 M_d(s)}{M_n(s)(s+2\zeta_C\omega_C)} = \frac{P(s)}{L(s)} \tag{53}$$

We can then write:

$$\begin{aligned} M_d(s) &= \frac{P(s)}{\omega_C^2} \\ M_n(s) &= \frac{L(s)}{(s+2\zeta_C\omega_C)} \end{aligned} \tag{54}$$

In order to match the poles exactly, $(s+2\zeta_C\omega_C)$ must divide $L(s)$ with no remainder.
At this point, we use the system description for 10K to design the filter coefficients for C. As described above, the dutch-roll mode appears at roughly $2\pi/10 = 0.6283$ rad/s, and ω_C is chosen to be well below this to ensure attenuation of control signals near the dutch-roll. Then ζ_C can be chosen to ensure $(s+2\zeta_C\omega_C)$ divides $L(s)$ evenly. As described above, this choice will result in a phase shift, which is determined to be acceptable.
The challenge of this method now comes with picking $A^*_{CL}(s)$, the desired closed loops poles. Since the system is non-minimum phase, there are performance limitations using any control scheme, and given the control signal is a real-life system with saturation, the poles cannot be moved too far without requiring unattainably large control signals.
As described in [7], an ideal closed loop parafoil system should exhibit characteristics of a low-pass filter, with near unity gain at low frequencies, and attenuate signals at frequencies near the dutch-roll mode. Complicating matters, both M and C must themselves be stable and minimum phase.
To address all these issues an optimization problem could be formulated, but for the sake of simplicity a method of trial and error was chosen. Since the LQR algorithm can be thought of as optimal pole-placement given state and cost weightings, poles were generated using the MATLAB `lqr()` method until satisfactory performance was achieved satisfying the stability conditions.
The final values for $A^*_{CL}(s)$, C and M are chosen as:

$$A^*_{CL}(s) = \begin{bmatrix} 1.0000 & 2.6190 & 3.7092 & 2.9763 & 1.4718 & 0.3636 & 0.0332 \end{bmatrix} \tag{55}$$

corresponding to two sets of roots at both $-0.5305 \pm 0.6724i$ and -0.2485, and

$$\begin{aligned} C &= \frac{(0.33)^2}{s^2 + 2(4.3314)(0.33)s + (0.33)^2} \\ M &= \frac{-0.097983(s+0.403)}{(s+0.232)(s^2+0.3374s+0.2557)} \end{aligned} \tag{56}$$

These correspond to an overdamped filter for C, a system M with poles somewhat more negative and less oscillatory than the original 10K system specified in (2). The non-unity gain must be divided out of reference signals for tracking.

IV. Adaptive Pole Placement Conrol (APPC)

Given the potentially non-minimum phase nature of the 10K system, conventional model-reference adaptive control (MRAC), which relies on dynamics inversion, is not applicable. Given the pole placement method used in §III.E to design the filter and reference system for $\mathcal{L}_1$, Adaptive Pole Placement Control (APPC), as described in [6], for example, is a natural conventional scheme for comparison. Block diagrams for both non-adaptive PPC and APPC are shown in Figures [3,4].

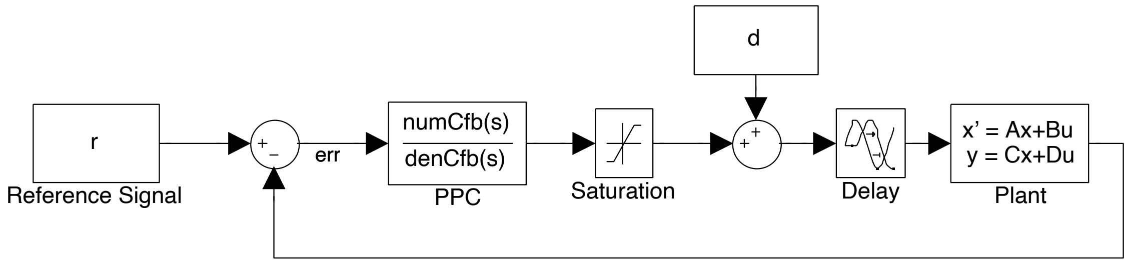

Figure 3. Conventional (non-adaptive) PPC Structure.

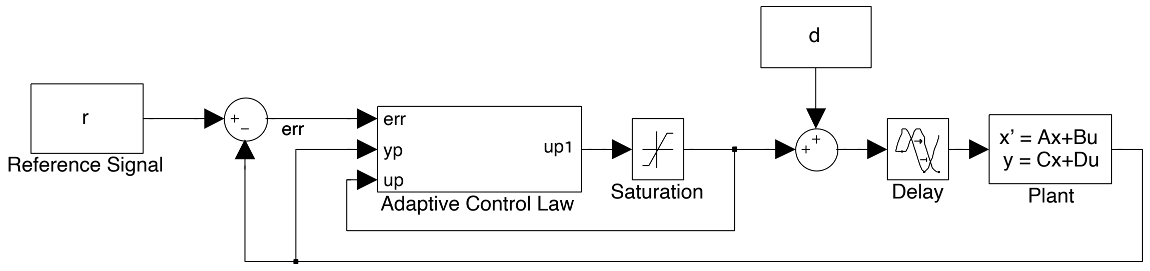

Figure 4. General Adaptive Control Structure. The adaptive law output is fed directly into the system, transient behavior and all.

In addition to the description given with respect to how the $\mathcal{L}_1$ filters were chosen, the following considerations were necessary for implementing a working APPC scheme:

- PPC Issues:
  - Singularity checking, implemented by explicitly checking for pole-zero cancellation at each time step.
  - Ensuring Sylvester Matrix is well conditioned, and contains only finite numbers.

- Adaptation Issues:
  - Scaling of ϕ using standard scaling by $m_s^2 = 1 + \phi^T \phi$.
  - Robust Adaptation, implemented using dead-zone at (1E-8 sec)m_s^2

The final block diagram structure is shown in Figure 5.

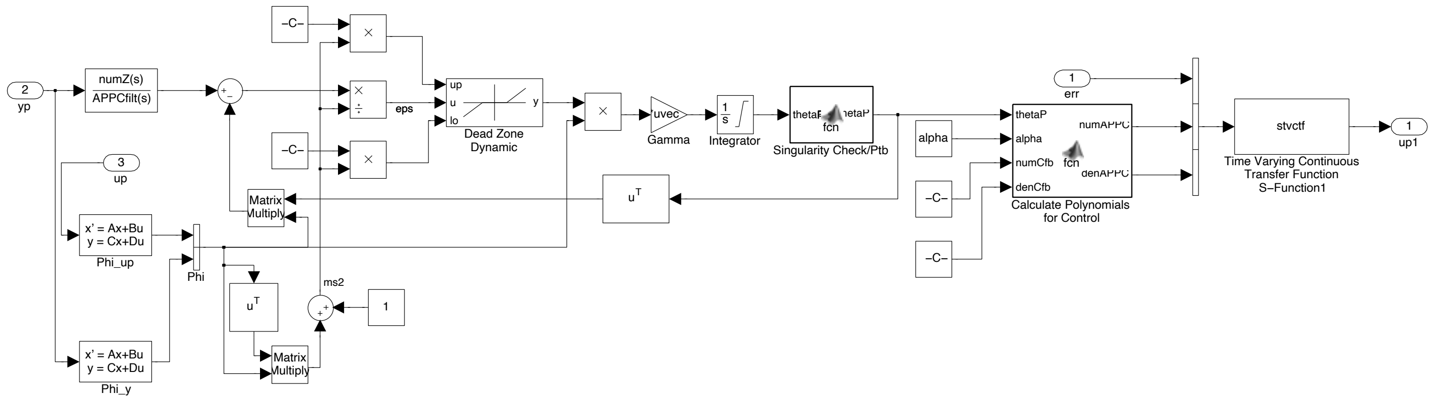

Figure 5. APPC Structure. Incorporates singularity checking, Sylvester matrix conditioning checking, as well as standard scaling and robust adaptation with dynamic deadzone.

V. Results

$\mathcal{L}_1$, APPC, and PPC schemes were implemented in Simulink. APPC and PPC were simulated using a fixed time step (T_s) of 1E-4 seconds (10 kHz). $\mathcal{L}_1$ was tested at 10 Hz. To demonstrate some key differences between them, three different scenarios are presented:

1. Disturbance added
2. Time-delay to input added
3. System dynamics changed

All trajectory plots are angular velocity in rad/s versus time in seconds, corresponding to turn rate.

V.A. Disturbance Added

The following disturbance is added:

$$d = 0.05 \max(r) \sin(20\omega_A t) \tag{57}$$

where r is the representative trajectory. The results are shown in Figures 6, 7, and 8.

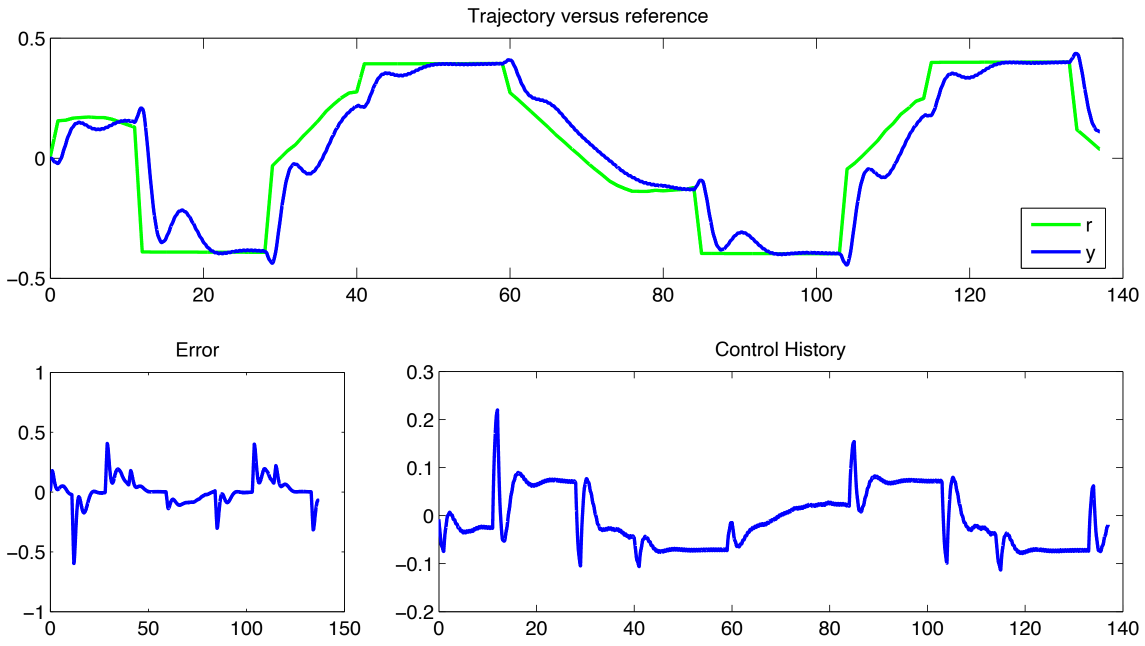

Figure 6. PPC tracking with disturbance $d = 0.05\max(r)\sin(20\omega_A t)$. All plots are versus time in seconds. While tracking the reference trajectory is quite good, the control signal contains large spikes with high frequency content.

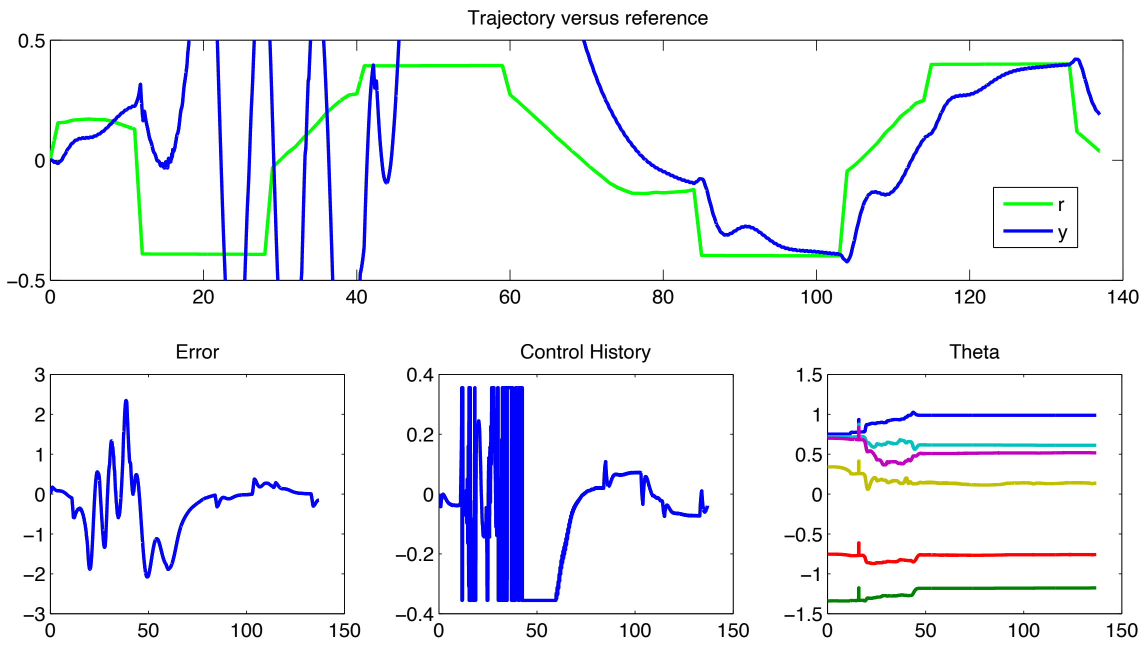

Figure 7. APPC tracking with disturbance $d = 0.05\max(r)\sin(20\omega_A t)$. Note the large transient behavior for over a minute of the trajectory, with high frequency and full saturation of the control channel. The maximum and minimum trajectory values reach are 2.4761 and -2.0820 rad/s, respectively.

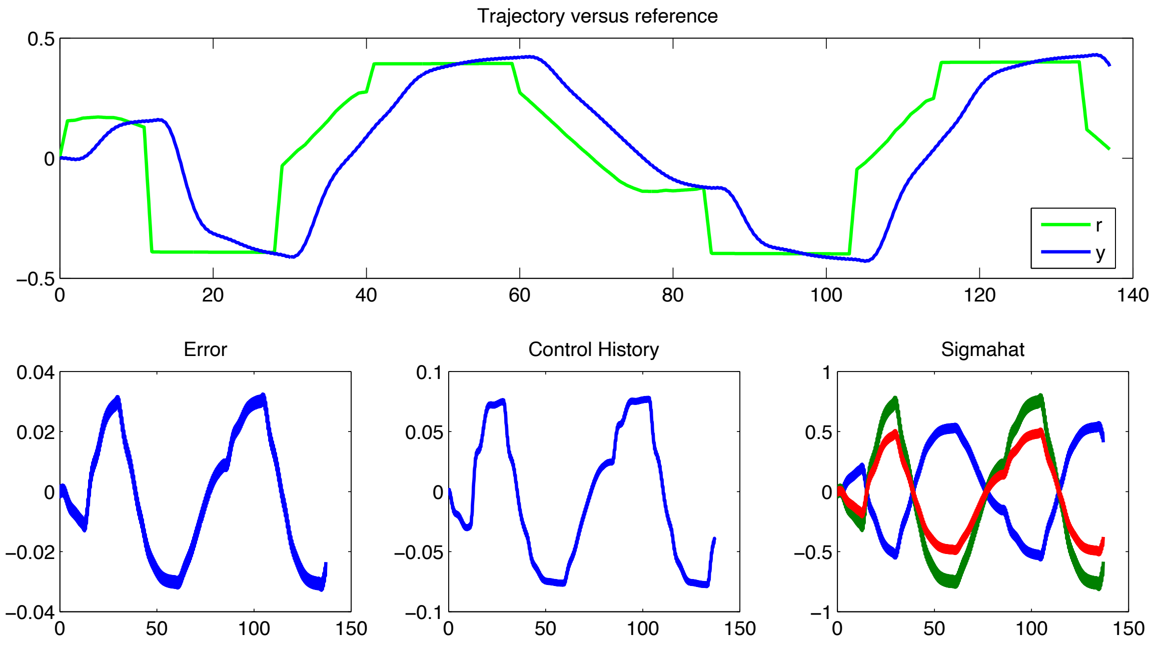

Figure 8. $\mathcal{L}_1$ tracking with disturbance $d = 0.05\max(r)\sin(20\omega_A t)$, discrete implementation (10Hz). All plots are versus time in seconds. Note the bounded transient behavior, with a smooth, bounded control signal throughout the trajectory.

Both PPC and and APPC, once its estimates converge, achieve relatively good tracking of the reference signal. However, both in the adaptive portion of APPC, as well as in the steady-state behavior of both controllers, this tracking is achieved at the price of high frequency control signals. $\mathcal{L}_1$ tracking shows larger phase delay throughout the trajectory, but otherwise tracks the reference signal reasonably well. This is achieved with much smoother, lower frequency control signals, which do not have large transients.

V.B. Time Delay Added

An input time delay of 0.5 sec is added to each system. The results are shown in Figures 9, 10, and 11.

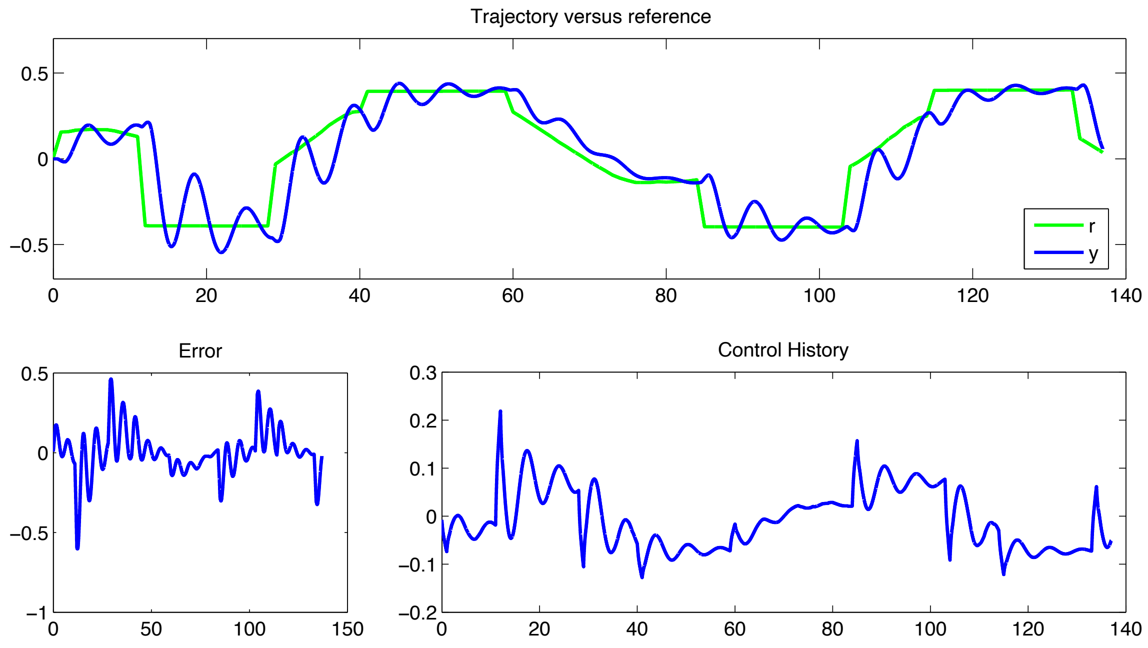

Figure 9. PPC tracking with input delay of 0.5 sec. All plots are versus time in seconds. Tracking the reference trajectory has deteriorated somewhat, becoming oscillatory. The control signal again contains large spikes with high frequency content.

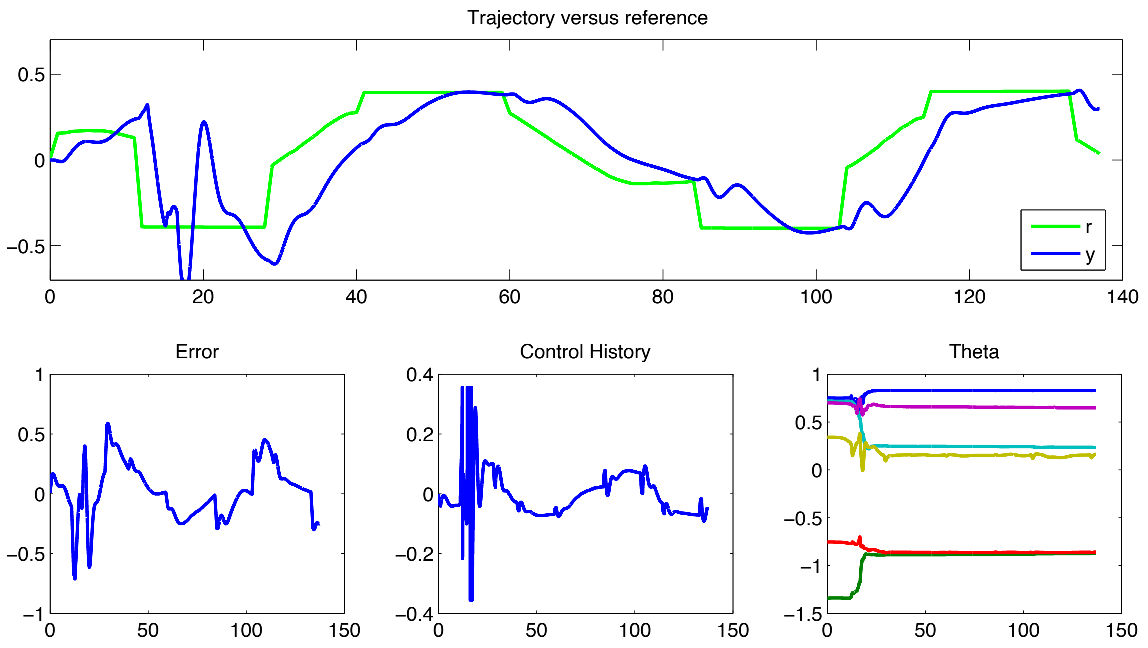

Figure 10. APPC tracking with input delay of 0.5 sec. All plots are versus time in seconds. Note the large transient behavior at the beginning of the trajectory, with high frequency and full saturation of the control channel, in addition to the expected phase lag.

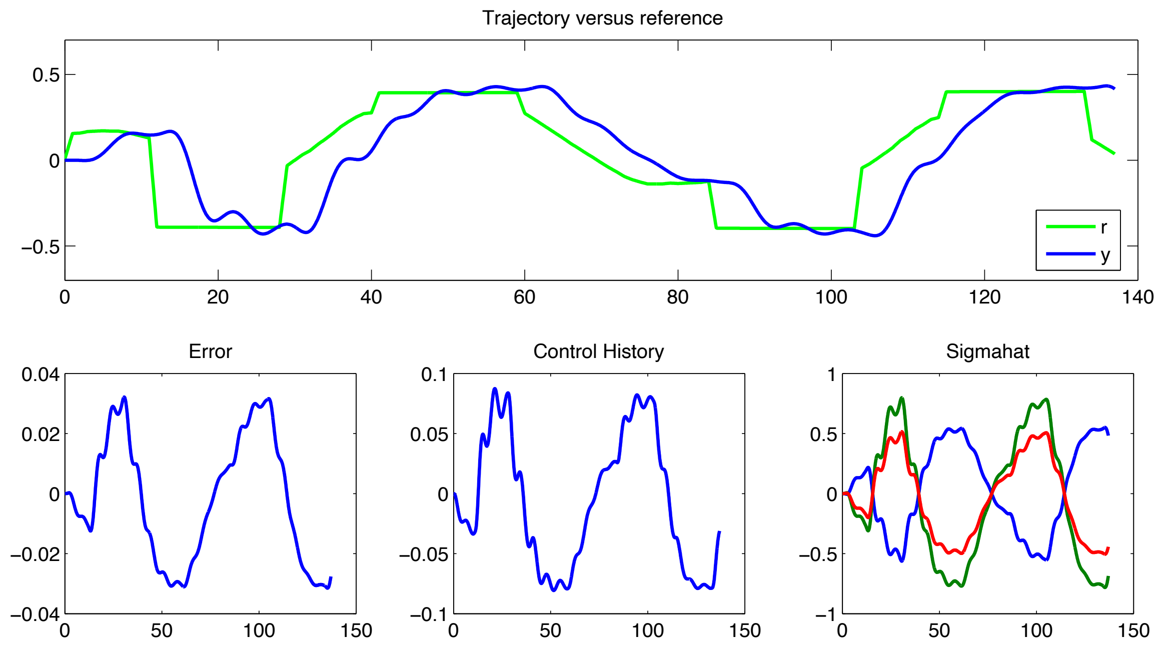

Figure 11. $\mathcal{L}_1$ tracking with input delay of 0.5 sec, discrete implementation (10 Hz). All plots are versus time in seconds. Note the minimal transient behavior throughout, with a the time delay introducing more muted oscillations, and still maintains a relatively smooth control signal.

The results are similar to the case of disturbance added, with Both PPC and and APPC, once its estimates converge, achieve relatively good tracking of the reference signal, with additional oscillations resulting from the input delay. However, both in the adaptive portion of APPC, as well as in the steady-state behavior of both controllers, this tracking is achieved at the price of high frequency control signals.

In contrast, $\mathcal{L}_1$ tracking shows very similar behavior to the disturbance case, with more muted oscillations from the time delay, maintaining similar performance to the un-delayed case. Once again, this performance is achieved with much smoother control signals than either A/PPC.

V.C. System Dynamics Changed

The closed loop poles chosen and PPC controller were synthesized using the nominal system model (1). Recalling the form from before:

$$A(s) = \frac{(s+a)(s+b)}{(s+c)(s^2+2\zeta_A\omega_A+\omega_A^2)} \tag{58}$$

Now, the values are changed from (2) to:

$$\begin{aligned} a &= 0.25\ (0.35) \\ b &= -2\ (-1.44) \\ c &= -0.31\ (0.23) \\ \omega_A &= \frac{2\pi}{11}\ \left(\frac{2\pi}{10}\right) \\ \zeta_A &= 0.12\ (0.19) \end{aligned} \tag{59}$$

which includes a slower, more lightly-damped dutch roll mode. The results are shown in Figures 12, 13, 14, and 15.

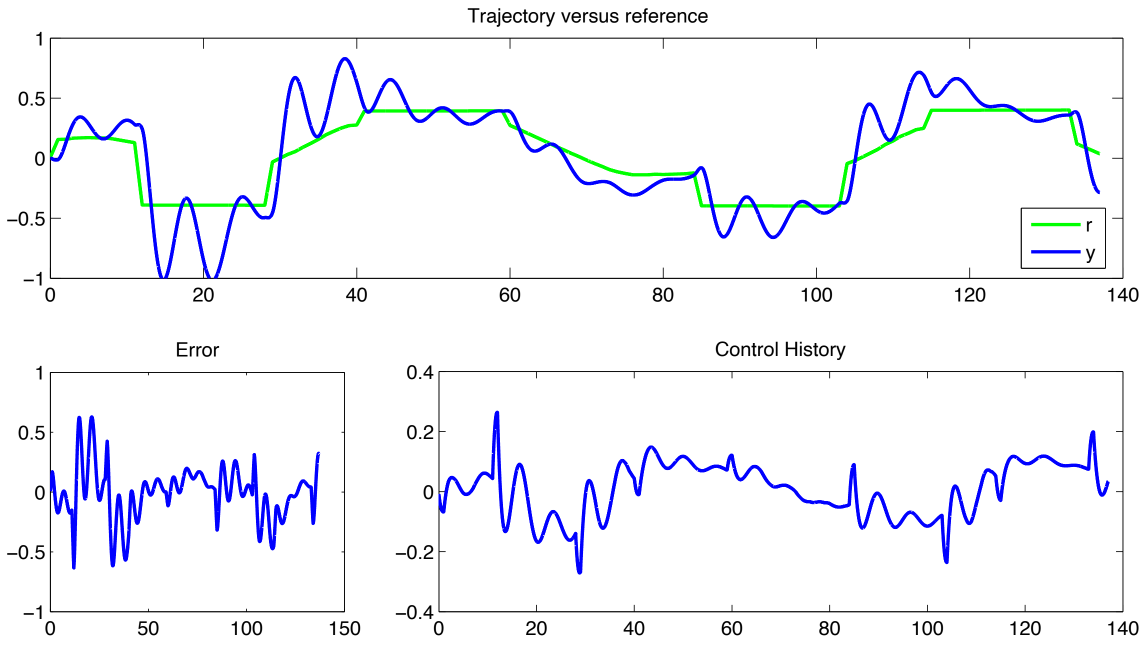

Figure 12. PPC tracking with system dynamics changed from (2) to (59). All plots are versus time in seconds. Tracking the reference trajectory has deteriorated somewhat, becoming more oscillatory. The control signal again contains large spikes with high frequency content.

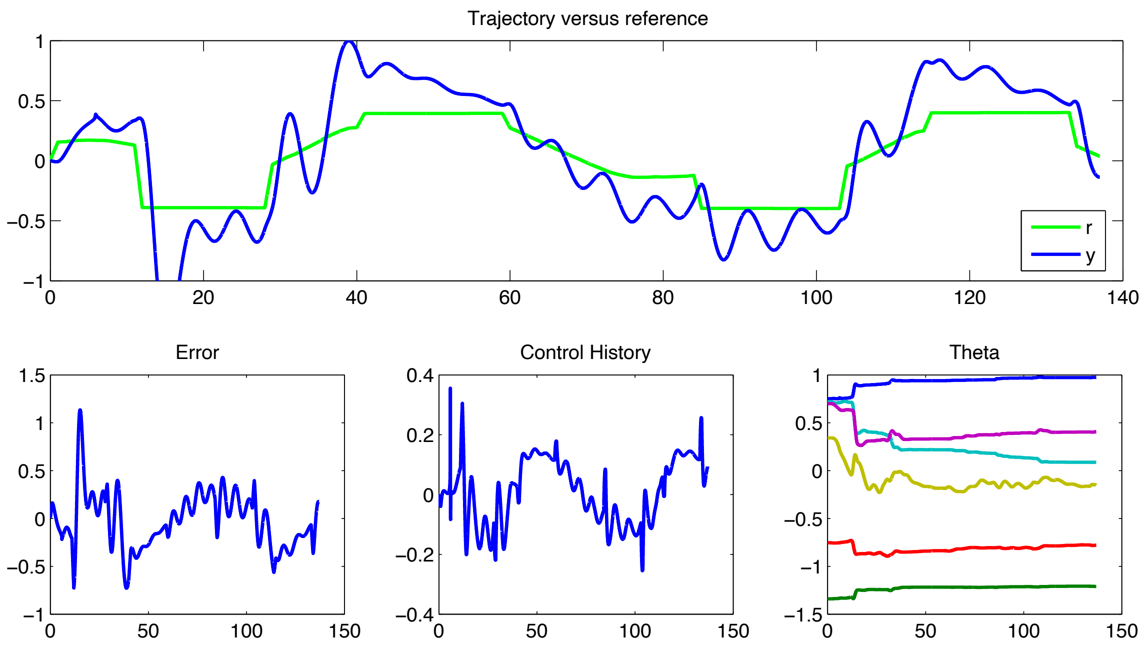

Figure 13. APPC tracking with system dynamics changed from (2) to (59). All plots are versus time in seconds. Note the high frequency control signal and large oscillatory tracking.

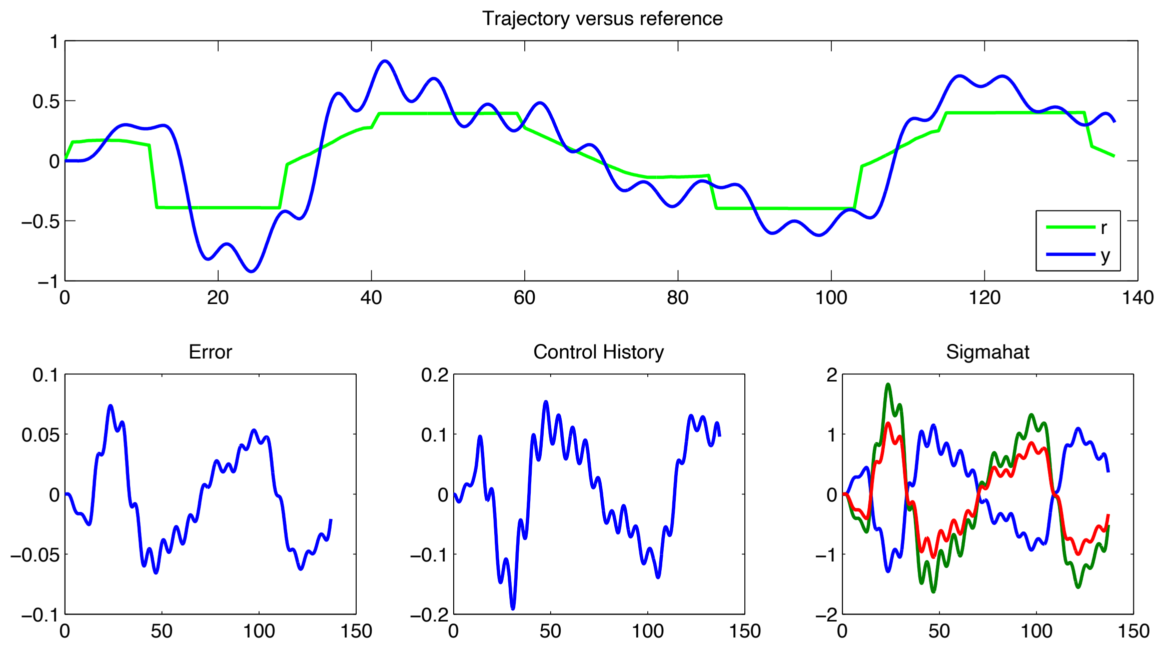

Figure 14. $\mathcal{L}_1$ tracking system dynamics changed from (2) to (59), discrete implementation (10 Hz). All plots are versus time in seconds. The increased oscillatory behavior is still slightly less than both A/PPC

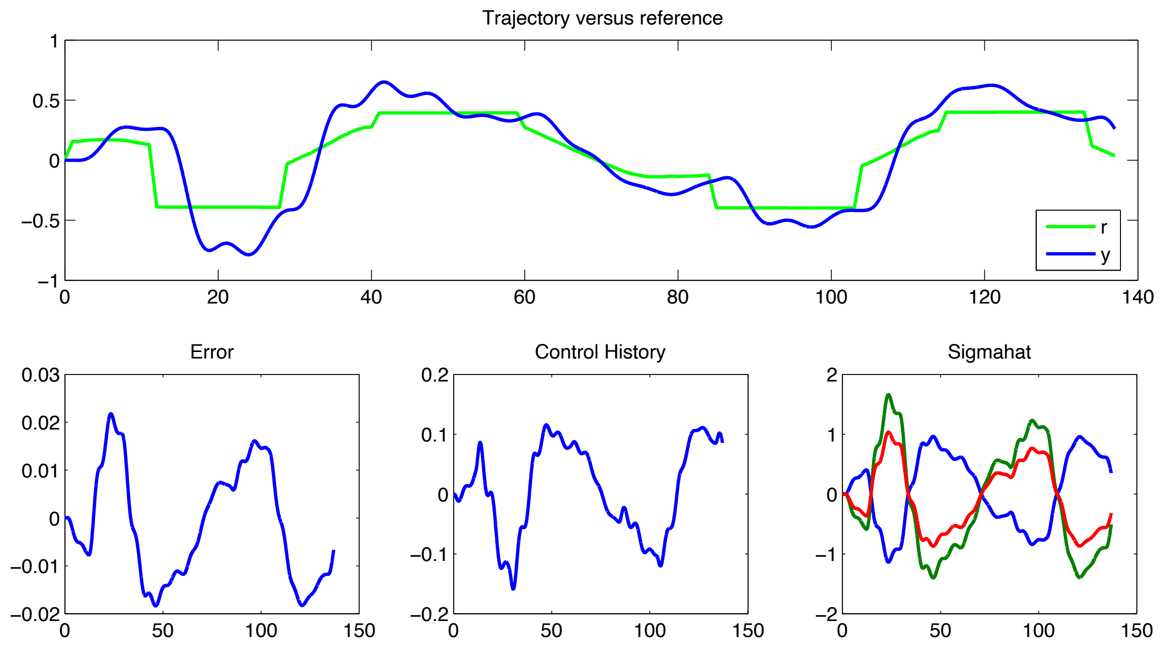

Figure 15. $\mathcal{L}_1$ tracking system dynamics changed from (2) to (59), discrete implementation (30 Hz). All plots are versus time in seconds. Note the minimal oscillatory behavior throughout, maintaining a relatively smooth control signal.

In this case, APPC never appears to converge to particularly good values, tracking with quite a lot of oscillatory behavior, as does the PPC. $\mathcal{L}_1$ tracking performance at 10 Hz likewise is also reduced somewhat, but it still overshoots and oscillates less than either A/PPC method. If the sampling rate in discrete $\mathcal{L}_1$ is increased 30 Hz, significantly improved performance is achieved.

VI. Discussion and Conclusion

The results for all three cases demonstrate different aspects of $\mathcal{L}_1$, as compared to both other adaptive control schemes, as well as conventional control.

The results shown for APPC, once the estimators converge, are on the whole relatively good. It is important to note this performance is extremely sensitive to initial guesses for $\hat{\theta}$, however. Despite including explicit singularity checking to ensure the calculation of control polynomials does not become ill-posed, as well as protections to ensure the Sylvester matrix is well formed, if the initial guess is too far from reasonable values, the adaptive scheme saturates the control and is unable to recover, and stops tracking entirely. The dead-zone robust estimator is likewise quite sensitive–setting the value too large has the same effects once the control saturates, and setting the value too small fails to prevent the bursting characteristic of constant adaptation. In addition, no attempt was made to ensure the reference signal was sufficiently rich, given the real-word application this controller is designed for.

$\mathcal{L}_1$ always appears to have phase shift, which is a fairly clear result of inserting a low pass filter in between the adaptation and the system. The advantages of having a smooth control signal, and not exciting large transient behavior likewise follow intuitively given this structure. Although this would not appear to be a justifiable practice in some contexts, in control of 10K, where smooth controls and avoiding transient behavior is critical, the advantages appear to outweigh the costs of increasing phase shift.

With respect to tracking, $\mathcal{L}_1$ appears comparable to PPC, by design. Since M and C were chosen using PPC design, this is unsurprising. As demonstrated in the third example, however, conventional PPC blindly places poles on a different system elsewhere, which can result in undesirable behavior. Even with adaption added, in APPC, this can still result in undesirable behavior. With $\mathcal{L}_1$, however, changes in the dynamics which do not violate the stability conditions detailed above can still maintain acceptable behavior, given the filtering design. Performance also improves greatly if the sampling frequency is increased from 10 Hz to 30 Hz, still far lower than the 10 kHz sampling rate for A/PPC.

A comparison of $\mathcal{L}_1$ adaptive control in discrete time with both APPC and conventional PPC has highlighted some tradeoffs of the different methods. In the context of large autonomous parafoil systems, such as 10K, the advantages of $\mathcal{L}_1$ adaptive control appear to outweigh the disadvantages, motivating further study into implementation and real-world testing.

References

[1] N. Hovakimyan and C. Cao. *L1 adaptive control theory: guaranteed robustness with fast adaptation*. SIAM, 2010.

[2] Karl Johan Åström and Björn Wittenmark. *Computer-controlled systems: theory and design*. Courier Dover Publications, 2011.

[3] Buddy Michini and Jonathan How. L1 adaptive control for indoor autonomous vehicles: design process and flight testing. AIAA-Paper-2009-5754.

[4] K.K.K. Kim, E. Kharisov, and N. Hovakimyan. Filter design for l1 adaptive output-feedback controller. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), 2011*, pages 5653–5658. IEEE, 2011.

[5] J. Wang, C. Cao, N. Hovakimyan, R. Hindman, and D.B. Ridgely. L1 adaptive controller for a missile longitudinal autopilot design. AIAA-Paper-2008-6282.

[6] P. Ioannou and B. Fidan. *Adaptive control tutorial*. SIAM, 2007.

[7] David Carter, Leena Singh, Leonard Wholey, Scott Rasmussen, Tim Barrows, Sean George, Marc McConley, Chris Gibson, Steve Tavan, and Brian Bagdonovich. Band-limited guidance and control of large parafoils. AIAA-Paper-2009-2981.