Proceedings of the 2007 American Control Conference
Marriott Marquis Hotel at Times Square
New York City, USA, July 11-13, 2007

Adaptive Tuning and Control of a Hard Disk Drive

Néstor O. Pérez Arancibia, Tsu-Chin Tsao and Steve Gibson

Abstract— **This paper deals with the implementation of adaptive tuning and control on a computer hard disk drive (HDD). A model-based adaptive loop is added to a LTI stabilizing controller to minimize the position control error of the HDD head. This adaptive loop may be used in either of two modes: control tuning (CT) or real-time fully adaptive control (RTFAC). The former (CT) consists of adapting a FIR filter to compensate for unknown disturbance dynamics using the inverse QR-RLS algorithm. Once the filter has converged it is wrapped into the LTI controller and a new tuning process is performed, iteratively improving the control performance. The second method (RTFAC) consists of adapting indefinitely a FIR filter to compensate for unknown disturbances. This method is implemented using an unwindowed lattice RLS filter, which makes the scheme converge substantially faster than when using other adaptive algorithms. This pertains to the HDD data seek time. Another innovation in this paper is the introduction of a frequency-weighting filter in the RTFAC scheme, which accounts for robust stability. Experimental results are presented for both adaptive tuning and control methods, programmed on a commercial hard disk drive.**

I. INTRODUCTION

The control of hard disk drives involves track-seeking and track-following. The former deals with the motion control of the head between tracks, the latter with maintaining the head on the center of the track. In this paper we focus on track-following.

It has been reported that track-following can be thought of as an output disturbance rejection problem [1], [2]. There exist two main sources of disturbances in the hard disk drive dynamics. The first is *repeatable runout* (RRO), which is produced by imperfections and eccentricities on the tracks. The second is *nonrepeatable runout* (NRRO), produced by the aggregated effects of disk drive vibrations, electrical noise in the circuits and the measurement channels.

Model-based noise rejection adaptive schemes are well suited to deal with the problems previously described here. There exist a series of slightly different adaptive structures. The two most prominent are the *adaptive inverse control* (AIC) [3] and *adaptive-Q* (A-Q) [4] schemes. Despite the different denominations, they are essentially the same.

The implementation of an add-on adaptive controller for disk drives, based on the AIC scheme is described in [2]. There, the adaptive algorithm employs the classical RLS recursions in [5]. The experimental implementation of an A-Q scheme is demonstrated in [1]. There, the disk drive open-loop plant is stabilized using a *linear time-invariant* (LTI) LQG compensator and then an adaptive filter Q is added using the typical configuration in [4]. The adaptation is done using low order LMS and RLS algorithms and implemented in real time using a TMS320C67 floating point *digital signal processor* (DSP).

In this paper, we adopt the frequency weighting adaptive structure introduced in [6], using two versions of the RLS algorithm. The first is the inverse QR-RLS algorithm described in [7]. This algorithm belongs to the family of array methods, that possess better numerical properties than the classical RLS recursions in [5]. The second algorithm employed is the lattice filter in [8], which converges significantly faster than any other RLS method employed on this experiment.

The experimental implementation of the control system was done using a *Mathworks*® xPC Target system over the same commercial hard drive employed in the experiments reported in [1]. This paper reports a significant improvement in the control performance, which is attributable to the use of better algorithms for system identification and control, and higher order filters, made possible by using the significatively faster xPC Target system for signal processing.

The paper is organized as follows. Section II describes the experimental setup and explains some relevant practical issues relating to the real time implementation of the controller. Section III describes the system identification of the hard disk drive dynamics required for control system design. Section IV explains the design of the control system, which consists of a LTI feedback control loop augmented by an adaptive control loop. Experimental results are presented in Section V, and finally, some conclusions are drawn in Section VI.

II. DESCRIPTION OF THE EXPERIMENT

A *hard disk drive* (HDD) is a mechatronic device that uses rotating *platters* to store data. Information is recorded on, and read from concentric cylinders or tracks by read-write magnetic transducers called *heads*, that fly over the magnetic surfaces of the HDD platters. The position of the heads over the platters is changed by an *actuator* that consists of a coil attached to a link, which pivots about a ball bearing. This actuator connects to the head by a steel leaf called *suspension* [9], [10]. This description of the HDD is shown in Fig. 1.

The control objective is to position the center of the head over the center of a data *track*. Thus, the typical measure of HDD tracking performance is the deviation of the center of the head from the center of a given track, which is often called *track misregistration* (TMR) [10]. There exist many indices to quantify TMR. Here we adopt

$$TMR = 3\sigma. \tag{1}$$

The authors are with the Mechanical and Aerospace Engineering Department, University of California, Los Angeles, CA, 90095–1597, USA. nestor@seas.ucla.edu, ttsao@seas.ucla.edu, gibson@ucla.edu.

1-4244-0989-6/07/$25.00 ©2007 IEEE.

Where σ is the empirical *standard deviation* (STD) of the control error signal. It is common to express 3σ as a percentage of the track pitch [1], [10], which must be less than 10% in order to be considered acceptable. TMR values larger than this figure will produce excessive errors during the reading and recording processes.

In this experiment we use a 2-platter (10 GB/platter), 4-head, 7200 rpm, commercial HDD, and a *Mathworks*® xPC Target system for control. The sample-hold rate of 9.36 KHz, used for communication, control and filtering, is internally determined by the HDD and transmitted trough a *clock* signal to the target PC used for control. Both systems must operate in a synchronized manner, as shown in the diagram of the experiment (Fig. 2).

The position of a given HDD head is digitally transmitted by the use of two signals. The first conveys the *track number* (TN) over where the head is positioned. The second is the so called *position error signal* (PES), which conveys the position of the head on the track pitch. Thus, the measured position y is function of both the TN and PES signals.

The loop is closed when the digital controller outputs the sequence x which is converted into an analog signal to command the HDD actuator. At this stage, we pose the control problem in the discrete-time domain, defining the mapping from x to y as our open-loop plant P.

Any controller designed for this system should robustly stabilize the slightly damped system dynamics of the HDD and reject the disturbances to which the system is subjected. Typical sources of disturbances are: disk vibrations, actuator arm vibrations, air turbulence (windage) and external shocks.

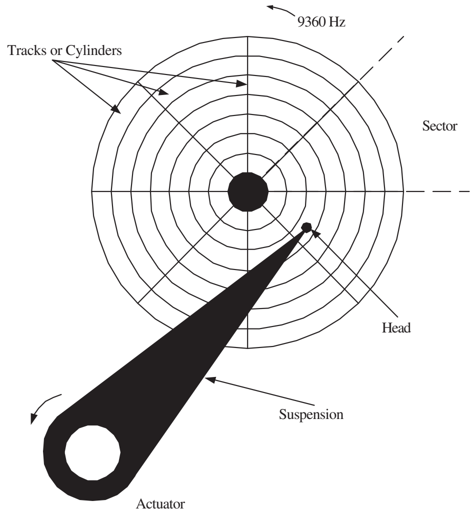

Fig. 1. Schematic idealization of the hard disk drive (HDD) system.

III. SYSTEM IDENTIFICATION

A. Identification of the Open-Loop Plant

The first step, in order to design a controller, is to find a LTI model for the open-loop plant P of the HDD. This model, denoted by $\hat{P}$, is obtained by system identification (SID). Since the system is slightly damped, the SID of P must be performed under feedback control. In general, there exist two methods for performing SID in closed-loop [11]. The first method is called *direct*, because P is directly estimated using signals x and y in Fig. 3. The second one is called *indirect*, because it first finds an estimate $\hat{M}$ of the closed-loop plant M using signals r and y in Fig. 3, and then computes

$$\hat{P} = \frac{\hat{M}}{K(1-\hat{M})}. \tag{2}$$

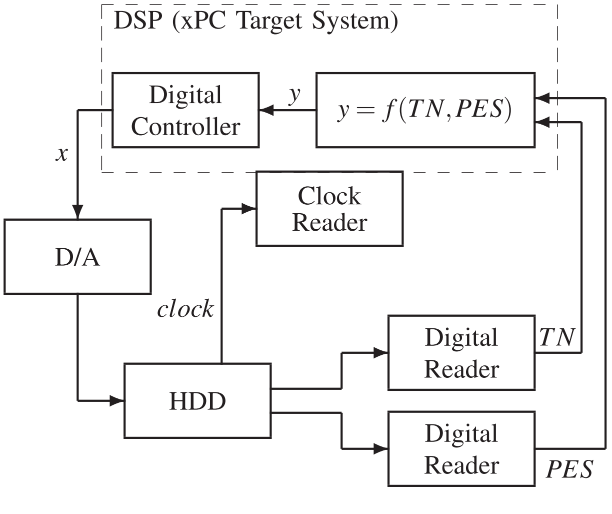

Fig. 2. Diagram of Experiment.

In this case, the indirect method is preferable, because r can be chosen to be independent of disturbances w_o and w_i. On the contrary, x always depends on w_o and w_i.

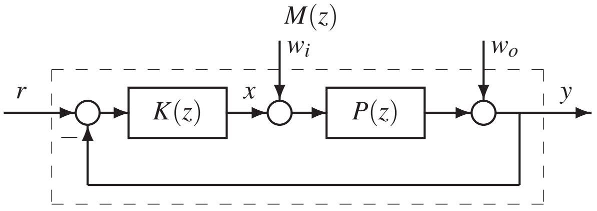

Fig. 3. Block diagram of PD feedback control system used for identification. $P(z)$ = open-loop plant; $K(z)$ = classical PD feedback controller; $y = Mr$; y = position of the head; r = excitation for SID; w_o = output disturbance; w_i = input disturbance.

At this stage, a plant model is not available, therefore, K is chosen to be a PD controller that is tuned on-line. Thus, after selecting suitable gains for K, a 20^{th} order closed-loop plant model $\hat{M}$ is computed using the *n4sid* algorithm [12] from $30,000$ input-output data points. This yields an open-loop plant model which is reduced to a 2^{nd} order model after performing a balanced truncation. The resulting $\hat{P}$ is shown in *blue* in Fig. 4.

B. Identification of the Closed-Loop Plant

Section IV will describe the model-based design of the LTI controller C that will be used from this point onwards.

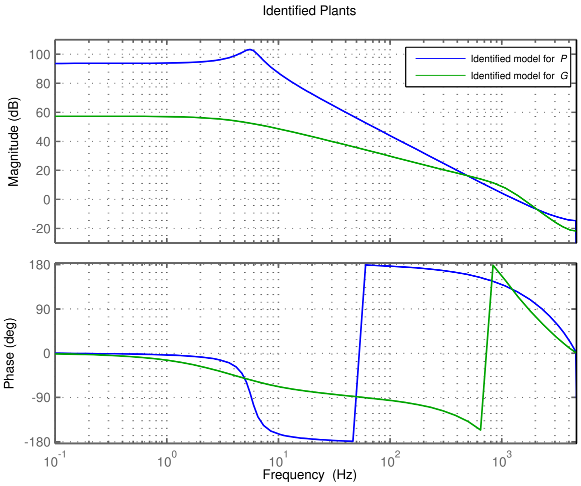

Fig. 4. *In blue:* Bode plot of the identified open-loop plant model $\hat{P}(z)$. *In green:* Bode plot of the identified closed-loop plant model $\hat{G}(z)$

This controller is connected to P as shown in Fig. 5, defining the closed-loop plant G. Analogously to the open-loop case, the identified transfer function of the closed-loop plant will be denoted by $\hat{G}$. This function $\hat{G}$ is estimated from $30,000$ data points using the *n4sid* algorithm and truncated to a 4^{th} order model using standard balanced truncation techniques. The resulting $\hat{G}$ is shown in *green* in Fig. 4.

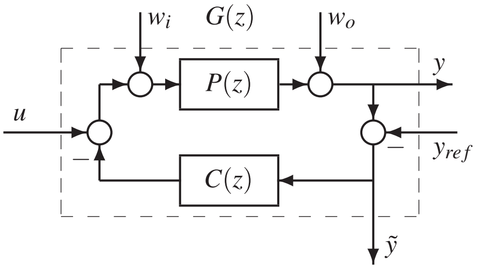

Fig. 5. Block diagram of LTI feedback control system. $P(z) =$ open-loop plant; $C(z) =$ classical LTI feedback controller; $y = Gu$; $y =$ position of the head; $u =$ adaptive filter output; $w_o =$ output disturbance; $w_i =$ input disturbance; $y_{ref} =$ position reference; $\tilde{y} =$ position error.

IV. CONTROL DESIGN

The control system proposed in this paper consists of two feedback loops. First, the LTI controller C robustly stabilizes the slightly damped open-loop dynamics and rejects low-frequency components of the disturbance w_0 in Fig. 5. Then, an adaptive control loop is added, as shown in Fig. 6, to augment the LTI controller and produce a more significant disturbance-rejection effect over the signal w, which represents the combined influences of w_o and w_i.

The adaptive loop may be used in either of two modes: *control tuning* (CT) or *real-time fully adaptive control* (RTFAC). CT denotes the process of adaptively finding optimal gains for rejecting the disturbance w in Fig. 6 until the system reaches steady state performance. Then, the adaptive filter is turned off and these gains are considered to be indefinitely optimal. This method is appropriate when w is statistically stationary, and has the advantage of allowing fast tracking once the adaptive gains have converged. This idea is implemented using the inverse QR-RLS algorithm [7].

On the other hand, RTFAC denotes an indefinite process of adaptively finding optimal gains for rejecting the disturbance w in Fig. 6. Implementing RTFAC is considerably more challenging than CT, because it requires not only an algorithm able to find the optimal gains, but also to converge fast in order to perform tracking. This requirement is satisfied using the lattice algorithm in [8], which is also computationally very efficient.

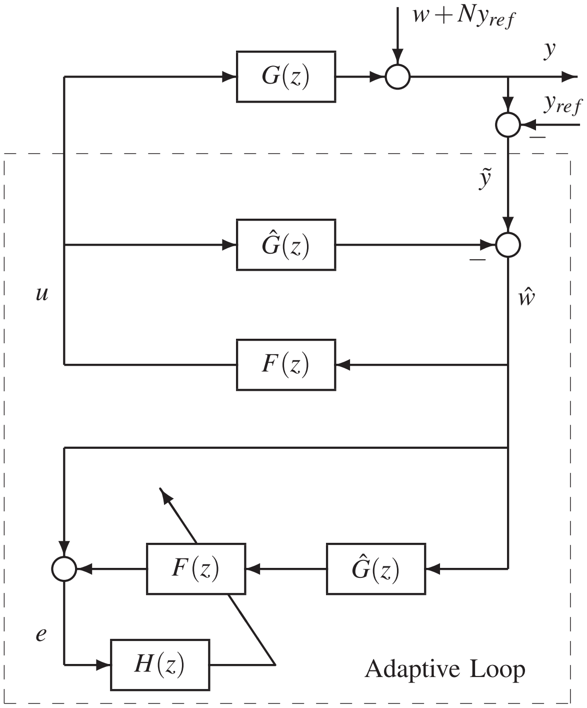

Fig. 6. Block diagram of the adaptive control system.

A. LTI Feedback Loop

The LTI feedback controller structure is shown in Fig. 5. This controller was designed using discrete-time domain classical techniques. It consists of a digital integrator and a digital notch filter. The integrator gain and notch parameters were tuned to maximize the output-disturbance rejection bandwidth. The input u in Fig. 5 will become the adaptive control command, and the output y is the position signal. The signal w_o represents the effect of all output disturbances, and similarly, w_i represents the effect of all input disturbances. The signal y_{ref} is constant and corresponds to the desired position of the HDD head.

Fig. 7 shows, in *blue*, an estimate of the output sensitivity transfer function, computed as $\hat{S}_o = (1+\hat{P}C)^{-1}$. Notice that in this case the input sensitivity function, S_i coincides with G, therefore, a good estimate of S_i is the identified function $\hat{G}$. Fig. 7 shows, in *green*, the computed transfer function

$\hat{N} = \hat{P}C(1+\hat{P}C)^{-1}$ which is the mapping from y_{ref} to y. Notice that $\hat{N}y_{ref} = y_{ref}$ when y_{ref} is constant.

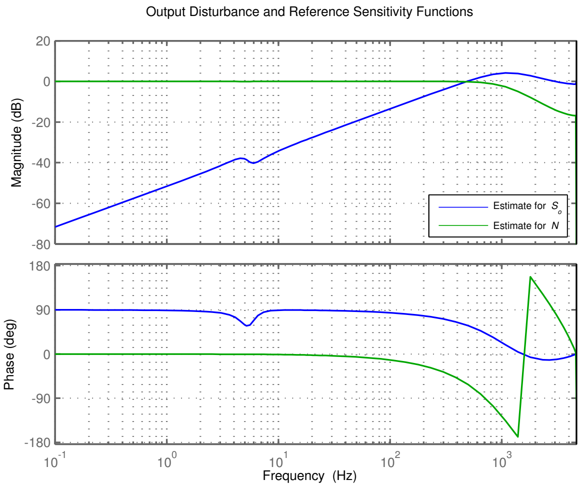

Fig. 7. Bode plots of the computed output disturbance and reference sensitivity transfer functions $\hat{S}_o(z) = (1+\hat{P}(z)C(z))^{-1}$ (*blue*) and $\hat{N} = \hat{P}(z)C(z)(1+\hat{P}(z)C(z))^{-1}$ (*green*).

B. Adaptive Loop

The adaptive control algorithm presented in this paper assumes known LTI plant dynamics but unknown disturbance dynamics. The adaptive controller requires an estimate $\hat{G}$ of the closed-loop transfer function G in Fig. 6. The signal w, represents the aggregated effects of w_o and w_i over the system. It is clear, after some algebraic manipulations, that

$$w = \frac{P}{(1+PC)}w_i + \frac{1}{(1+PC)}w_o. \tag{3}$$

The control objective is to minimize the RMS value of the position error $\tilde{y} = y - y_{ref}$. In order to achieve this objective, we pose the control problem as a least squares problem, solvable recursively by the use of any of the versions of the RLS algorithm.

To begin with, let τ be an integer and x be a generic sequence in ℓ_2. Then define

$$x_\tau = \begin{cases} x_\tau(k) = x(k) & 0 \le k \le \tau \\ x_\tau(k) = 0 & otherwise \end{cases}. \tag{4}$$

Thus, the control objective becomes finding an operator F that solves

$$\min_F \|H\tilde{y}_\tau\|_{\ell_2}, \tag{5}$$

where τ must be interpreted as the value of the digital counter at *present time*.

The filter H provides frequency weighting in the adaptive controller, which allows us to give different relative importance to different bands of the frequency spectrum. In many cases it is convenient to use weighting filters, for example, the results in [6] provide empirical evidence that the use of a high pass filter H reduces the H_∞ norm of the steady-state filter F, increasing the stability robustness of the adaptive scheme.

Now, choosing F to be a FIR filter, noticing that $y_{ref} = Ny_{ref}$ when y_{ref} is constant, assuming perfect matching between G and $\hat{G}$, and taking into consideration that SISO systems commute, (5) is equivalent to

$$\min_F \|H\hat{w}_\tau + F\hat{G}H\hat{w}_\tau\|_{\ell_2} \quad s.\ t. \quad F(z) = z^{-1}\sum_{i=0}^{L} a_i z^{-i}. \tag{6}$$

Once the problem has been posed in the form of (6), by simple algebraic manipulations it can be transformed into a typical RLS problem. Here, we conduct experiments using two versions of the RLS algorithm: The inverse QR-RLS algorithm in [7], and the lattice filter in [8].

C. Stability of the Adaptive Scheme

The adaptive scheme in Fig. 6 can be put into the typical feedback configuration shown in Fig. 8, where $\Delta = G - \hat{G}$, F is the filter adaptively computed according to the laws described in the previous subsection, and $r = F(N-1)y_{ref}$. The configuration in Fig. 8 is well posed according to the definition in [13], the open-loop dynamics is robustly stabilized by a LTI controller, therefore, Δ is ℓ_2-stable, and F is always stable because of having a FIR form. Thus, it follows immediately by the use of the *small gain theorem* that a sufficient condition for stability of the adaptive scheme in Fig. 6 is given by

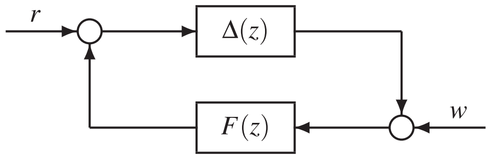

Fig. 8. Typical feedback connection.

$$\|\Delta\|_\infty \|F\|_{\ell_2 \to \ell_2} < 1. \tag{7}$$

It is worth noticing that for the special case when $\hat{G} = G$, the configuration in Fig. 8 is stable for any stable F. The relation in (7) is not applicable directly, however, it gives powerful insights and guidelines about the way the weighting filter H should be chosen. It is clear that (7) does not ignore the underlying nonlinear dynamics used to generate the parameters of F, simply states that a desirable adaptive algorithm should produce filters with small enough induced norms.

V. EXPERIMENTAL RESULTS

In the experiments described here, the sample-and-hold rate for control and filtering was 9.36 KHz, externally determined by the HDD clock as shown in Fig. 2. First, we discuss the experiments performed on *control tuning* (CT). These consist of adaptively finding gains for the filter F in Fig. 6 employing the inverse QR-RLS algorithm. Once convergence has been achieved these gains are fixed and consequently the whole scheme becomes LTI, in other words, we tune F. The filter F is tuned in a given location of the HDD using the corresponding closed-loop model $\hat{G}$ at that location. A location is defined by a head and by a cylinder, which in this case is head 0 – cylinder 10,000.

The CT method was implemented with filters of orders 6, 12, 24 and 36. Fig. 11 shows time series, *power spectral densities* (PSDs) and the histogram for the case $L = 24$. Also, the achieved performances are shown in Table I. The first thing to notice is that CT significatively improves the performance achieved by the classical LTI controller alone. The second one is that there is not a significant improvement related to higher orders on the head 0. On the contrary, higher order feedforward filters make a clear improvement on head 2. This suggests that higher older filters compensate for inaccuracies on the identified model $\hat{G}$. Considering this evidence, a 2-stage CT scheme is proposed. The idea is to adaptively look for a new feedforward filter F_2, replacing $\hat{G}$ by $\hat{G}_2$ in the scheme shown in Fig. 6. This new F_2 will be specialized to some areas of the HDD. In this case, to every head. The transfer function $\hat{G}_2$ is the identified model of G_2 defined by the block diagram in Fig. 9, with $U = F(1+F\hat{G})^{-1}$.

The performances, quantified by the 3σ index, achieved by the 2-stage CT method, at different locations of the HDD are shown in Table I. It is important to remark that this method produces an improvement with respect to the simple CT method in every position where it was tested.

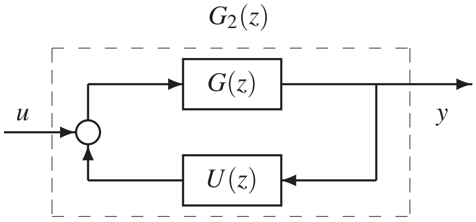

Fig. 9. Transfer function G_2.

Now we discuss the experiments performed on RTFAC. In order to implement a fully adaptive controller on the HDD it is crucial that the employed adaptive algorithm achieves both fast adaptation and empirical minimum variance steady state performance. These two requirements are satisfied by the unwindowed lattice RLS filter in [8]. Empirical evidence of this fact is shown in Fig. 10, where we compare the convergence speeds of the inverse QR-RLS and the lattice RLS algorithms.

Depending on the problem, fast adaptation may produce prediction filters with large ℓ_2-induced (H_∞ in steady state) norms. This is undesirable, because as stated in Section IV, stability is ensured when (7) is satisfied. In other words, a filter F with a large ℓ_2-induced norm makes the stability of the whole adaptive scheme less robust. The use of an appropriate weighting filter H is a way to deal with this issue. All prediction filters amplify high frequencies, therefore, if we use a high pass filter H, the undesirable effects are compensated. The price for using this technique is an amplification of low frequency errors [6].

Another phenomenon commonly observed when implementing adaptive controllers is the occurrence of large overshoots during the transient time. As shown in [14], this problem can be eliminated by varying the order of the adaptive filter F. The implementation of variable-order filters is possible because lattice filters are not only time recursive, but order recursive as well. Notice that it is not feasible to vary the order of the filter F when using classical RLS algorithms. Here, the lattice filter was implemented using the weighting filter H shown in Fig. 12 with orders 6 and 24, where the order of F was incremented by 2 every 100 steps. Fig. 13 shows time series, *power spectral densities* (PSDs) and the histogram for the case $L = 24$. The achieved performances are shown in Table I.

VI. CONCLUSIONS

This paper presented experimental results on the application of recursive techniques for implementing adaptive tuning and control on a commercial HDD. Two methods were demonstrated here: iterative adaptive tuning using the inverse QR-RLS algorithm and frequency-weighting RTFAC using an unwindowed lattice filter. Experimental results show the effectiveness of these two approaches. Especially important it is to remark that the use of the unwindowed lattice filter makes the RTFAC scheme converge considerably faster than when using other adaptive filters tested in this experiment. Future research will integrate track seeking with track following adaptive control.

REFERENCES

[1] K. Krishnamoorthy and T.-C. Tsao, “Adaptive-Q with LQG Stabilization Feedback and Real Time Computation for Disk Drive Servo Control,” in *Proc. of the American Control Conference*, Boston, MA, Jun. 2004, pp. 1171–1175.

[2] R. Horowitz and B. Li, “Adaptive Control for Disk File Actuators,” in *Proc. 34th IEEE Conf. on Decision and Control*, New Orleans, LA, Dec. 1995, pp. 655–660.

[3] B. Widrow and E. Walach, *Adaptive Inverse Control.* Upper Saddle River, NJ: Prentice-Hall, 1996.

[4] T.-T. Tay, I. Mareels, and J. B. Moore, *High Performance Control.* Boston, MA: Birkhauser, 1998.

[5] G. C. Goodwin and K. S. Sin, *Adaptive Filtering, Prediction and Control.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[6] N. O. Pérez Arancibia, N. Chen, S. Gibson, and T.-C. Tsao, “Adaptive Control of Jitter in Laser Beam Pointing and Tracking,” in *Proc. SPIE 6304, Free-Space Laser Communications VI*, San Diego, CA, Sep. 2006, 63041G.

[7] A. H. Sayed, *Fundamentals of Adaptive Filtering.* New York, NY: Wiley, 2003.

[8] S.-B. Jiang and J. S. Gibson, “An unwindowed multichannel lattice filter with orthogonal channels,” *IEEE Transactions on Signal Processing*, vol. 43, no. 12, pp. 2831–2842, Dec. 1995.

[9] D. Abramovitch and G. Franklin, “A Brief History of Disk Drive Control,” *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 28–42, June 2002.

[10] W. Messner and R. Ehrlich, “A Tutorial on Controls for Disk Drives,” in *Proc. of the American Control Conference*, Arlington, VA, Jun. 2001, pp. 408–420.

[11] U. Forssell and L. Ljung, “Closed-loop identification revisited,” *Automatica*, vol. 35, no. 7, pp. 1215–1241, July 1999.

[12] P. Van Overschee and B. De Moor, *Subspace Identification for Linear Systems.* Norwell, MA: Kluwer Academic Publishers, 1996.

[13] M. A. Dahle and I. J. Diaz-Bobillo, *Control of Uncertain Systems.* Englewood Cliffs, NJ: Prentice-Hall, 1995.

[14] N. O. Pérez Arancibia, N. Chen, T.-C. Tsao, and S. Gibson, “Variable-order adaptive control of a microelectromechanical steering mirror for suppression of laser beam jitter,” *Optical Engineering*, vol. 45, no. 10, pp. 104 206–1–12, Oct. 2006.

TABLE I

3σ VALUE OF THE POSITION ERROR SIGNAL (PES) AS A PERCENTAGE OF THE TRACK WIDTH

	Head 0			Head 1			Head 2		
	$y_{ref} = 10^3$	$y_{ref} = 10^4$	$y_{ref} = 2 \cdot 10^4$	$y_{ref} = 10^3$	$y_{ref} = 10^4$	$y_{ref} = 2 \cdot 10^4$	$y_{ref} = 10^3$	$y_{ref} = 10^4$	$y_{ref} = 2 \cdot 10^4$
LTI Feedback Control	18.0487	21.8415	23.2253	21.4312	19.1459	23.0018	23.5694	22.0469	24.1790
CT using the *inverse* QR algorithm (L=6)	5.3372	5.2591	5.3670	6.2368	5.8984	5.8023	6.2368	5.8984	7.3703
CT using the *inverse* QR algorithm (L=12)	5.2713	5.2087	5.3659	5.9567	5.6555	5.7407	6.1090	5.4013	7.0173
CT using the *inverse* QR algorithm (L=24)	5.2856	5.1800	5.3217	5.8038	5.4684	5.6170	5.8377	5.3387	5.9302
CT using the *inverse* QR algorithm (L=36)	5.2588	5.1791	4.8073	5.6199	5.4865	5.5195	5.6709	5.3149	5.5072
2-stage CT using the *inverse* QR algorithm (L=36)	5.0341	5.0497	4.7815	5.3806	5.2741	5.2297	5.5260	5.2111	5.4395
RTFAC using the *lattice* algorithm (L=6)	5.5935	5.6775	5.2485	6.4157	5.7239	5.6059	6.7697	7.5971	6.4161
RTFAC using the *lattice* algorithm (L=24)	5.7151	5.4988	5.0875	6.4810	5.8122	5.6111	6.9194	6.7757	6.4046

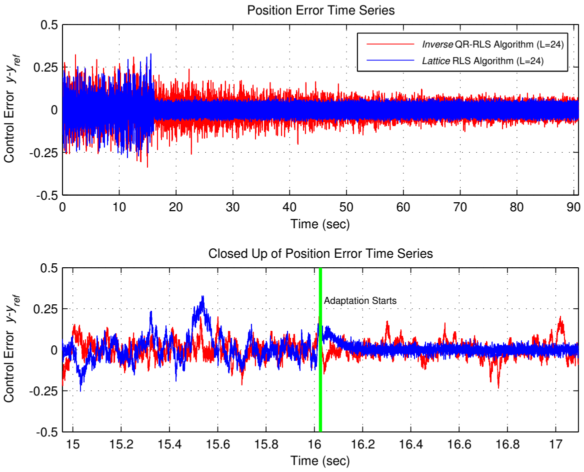

Fig. 10. *Top plot:* Time series compare convergence speeds of the inverse QR-RLS, and lattice RLS algorithms. *Bottom plot:* Closed up of top plot.

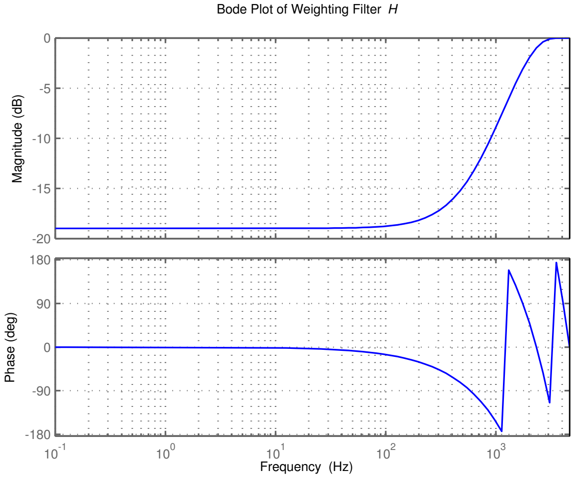

Fig. 12. Bode plot of weighting filter H used in the implementation of RTFAC.

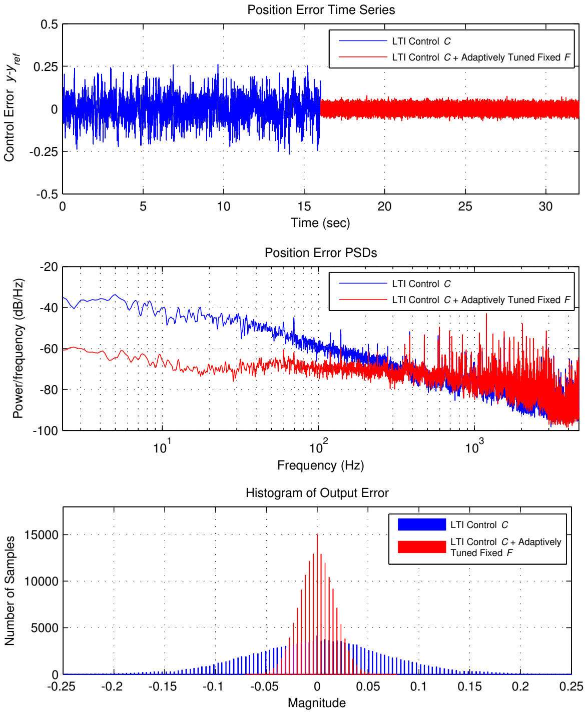

Fig. 11. Experiment performed on *control tuning* with $L = 24$ at head 0 and track $10,000$. *Top Plot:* Time series. *Middle Plot:* PSDs. *Bottom Plot:* Histogram.

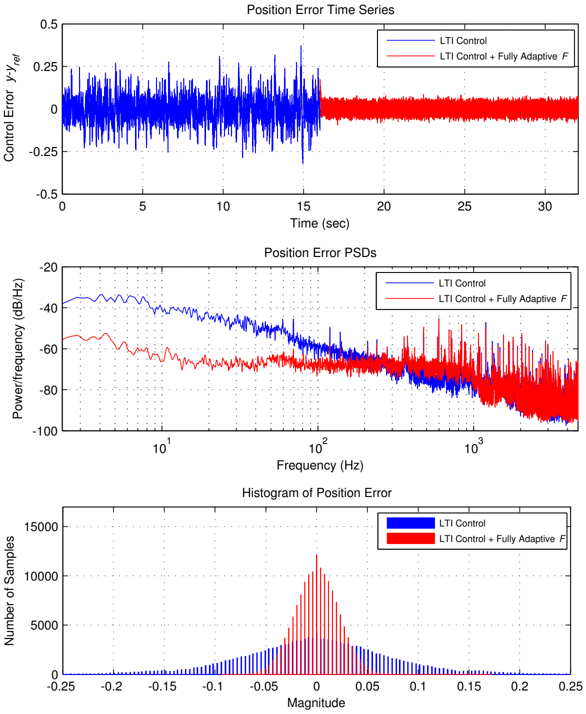

Fig. 13. Experiment performed on *real-time fully adaptive control* with $L = 24$ at head 0 and track $10,000$. *Top Plot:* Time series. *Middle Plot:* PSDs. *Bottom Plot:* Histogram.