High Performance Motion Tracking Control for Electronic Manufacturing

Benjamin Potsaid
e-mail: potsab@rpi.edu

John Ting-Yung Wen[1]
Director
e-mail: wenj@rpi.edu

Center for Automation Technologies and Systems (CATS),
Rensselaer Polytechnic Institute,
Troy, NY 12180

Mark Unrath
e-mail: unrathm@esi.com

David Watt[2]
e-mail: david.watt@sri.com

Mehmet Alpay
e-mail: alpaym@esi.com

Electro Scientific Industries, Inc.,
Portland, OR 97229

Motion control requirements in electronic manufacturing demand both higher speeds and greater precision to accommodate continuously shrinking part/feature sizes and higher densities. However, improving both performance criteria simultaneously is difficult because of resonances that are inherent to the underlying positioning systems. This paper presents an experimental study of a feedforward controller that was designed for a point-to-point motion control system on a modern and state of the art laser processing system for electronics manufacturing. We systematically apply model identification, inverse dynamics control, iterative refinement (to address modeling inaccuracies), and adaptive least mean square to achieve high speed trajectory tracking. The key innovations lie in using the identified model to generate the gradient descent used in the iterative learning control, encoding the result from the learning control in a finite impulse response filter and adapting the finite impulse response coefficients during operation using the least-mean-square update based on position, velocity, and acceleration feedforward signals. Experimental results are provided to show the efficacy of the proposed approach, a variation of which has been implemented on the production machine.
[DOI: 10.1115/1.2789467]

Keywords: precision motion, iterative learning, inverse dynamics, motion control, adaptive FIR filter

1 Introduction

A key challenge facing electronic manufacturers today is that as feature sizes continuously decrease and component counts increase, maintaining or boosting production rates requires improvement in both the speed and precision of the manufacturing equipment. However, improving both of these performance criteria simultaneously is difficult because higher speeds and accelerations tend to excite structural modes in the systems, which are typically lightly damped. For electronic memory repair [1] and microvia drilling applications [2], these unwanted vibrations reduce accuracy and detrimentally affect settling times. Consequently, feedforward compensation schemes to augment the feedback controller are becoming an increasingly critical component of the overall control architecture to achieve today's demanding performance requirements.

Various feedforward control schemes have been proposed to suppress undesired vibration. The simplest is to generate a smooth trajectory with bounded velocity, acceleration, and possibly even higher motion derivatives (e.g., trapezoidal and S-curve velocity profiles are available in most industrial motion controllers) [3]. This approach simply limits the bandwidth of the input trajectory and does not take any model information into account. When an analytic model (linear or nonlinear) is available, various model inversion techniques have been proposed [4–8]. When the model is nonminimum-phase (i.e., the zero dynamics are unstable), the causal inverse is unstable. A noncausal stable inverse may be used, but the output would be phase shifted with respect to the desired output, resulting in added motion time or tracking error. The inverse dynamics approach requires a high fidelity nonlinear model, which is usually unavailable in practice. Furthermore, if nonlinearity is present, the computation could also be quite demanding. A similar model inversion approach for nonminimum-phase discrete time linear time invariant (LTI) systems has also been proposed to achieve the overall zero phase shift [9–11]. However, it is not possible to simultaneously achieve zero phase as well as unity gain (since that would require causal inversion of a nonminimum-phase transfer function). To avoid explicit model inversion, an H_∞ model matching approach [12,13] may be used to find the feedforward controller. As the bandwidth of the weighting functions in the objective function increases, the solution converges to the causal inversion. If the system has a few dominant vibrational modes at known frequencies, then a finite impulse response (FIR) filter may be designed to cancel vibration after a finite period of time [14–18]. However, the approach does not generalize to more complex systems or nonlinear systems.

In this paper, we develop and demonstrate an adaptive feedforward control approach to significantly improve the tracking performance of a modern and state of the art electronics processing machine used for microvia laser drilling operations. The plant is a single-input/single-output (SISO) positioning system and we assume that a suitable feedback controller has already been implemented, as shown in Fig. 1. This controller has been carefully designed to ensure stability even in the presence of slight plant variations with the remaining design freedom used to achieve good tracking and settling performance. However, the performance of the production feedback controller alone was not acceptable, which motivated this study of adaptive feedforward compensation.

The laser scanning configuration used in the production machine and the experimental testbed in our laboratory are shown in Fig. 2. In the machine, a beam steering mirror redirects a collimated laser beam toward a scan lens, which collects the laser light and focuses the beam onto the working area. This scan lens obeys an f-θ relationship between the angle of the incoming light beam and the position of the focused spot on the surface to be processed

[1]Also at Department of Electrical, Computer, Systems Engineering.
[2]Also at SRI International.

Contributed by the Dynamic Systems, Measurement, and Control Division of ASME for publication in the JOURNAL OF DYNAMIC SYSTEMS, MEASUREMENT, AND CONTROL. Manuscript received April 6, 2006; final manuscript received March 2, 2007. Review conducted by Santosh Devasia. Paper presented at the IEEE Conference on Control Applications, Taipei, Taiwan, September 2004.

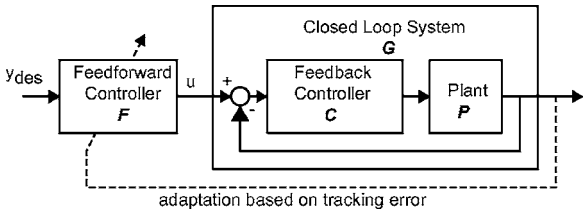

Fig. 1 Feedback and feedforward control architecture

(i.e., the position of the spot in x or y coordinates is directly proportional to the angle of the associated beam steering mirror, θ, not the tangent of θ as is usually the case) [19]. For reference, the transformation to convert between rotary joint angle and linear spot position is given by

$$x = 0.1667 \frac{\mu\text{m}}{\mu\text{rad}} \theta_x$$

$$y = 0.1667 \frac{\mu\text{m}}{\mu\text{rad}} \theta_y \tag{1}$$

The angular position of the beam steering mirror is measured through a high resolution rotary joint encoder and, in actuality, the controller acts to direct the angle of the steering mirror. However, we have chosen to perform all calculations with respect to the projected laser spot coordinate system as this is the true critical dimension of interest during the laser processing operation and to be consistent with the existing production closed loop controller specifications.

The existing closed loop control system and the data acquisition system operate at a sampling frequency of 200 kHz, which was chosen to effectively address the high frequency oscillatory modes of the system. To avoid internal saturation within the feedback controller and electronics of the servo amplifier, the trajectory must obey certain velocity and acceleration constraints. A maximum velocity $v_{\max}=5$ m/s and maximum acceleration $a_{\max}=1000\text{g}=9800$ m/s^2 were chosen based on the thermal/electronic limitations of the hardware and the requirements of the electronic manufacturing process. A high fidelity model that includes these effects has been considered, but the approach was rejected due to the model complexity. Note that these values are associated with the trajectory of the projected laser spot on the working surface and the transformation back to the rotary coordinate system is given by Eq. (1). The motion range varies from 200 μm to 5000 μm and the motion objective is to minimize the settling time in point-to-point motion subject to the velocity and acceleration constraints. We transform the problem into a trajectory tracking problem by generating a desired output trajectory that satisfies the velocity and acceleration constraints, and then designing the feedforward controller to track this desired trajectory as closely as possible. If only velocity and acceleration bounds are imposed, the minimum time trajectory would be of a trapezoidal velocity (rectangular acceleration) profile. If additional jerk bounds are also required, the acceleration profile is trapezoidal (this is the so-called S-curve velocity profile). In our experimental study, we have found that a half-sine acceleration profile gives the best performance. All the experimental data presented in the paper are based on this trajectory profile.

Our design approach starts by identifying a LTI model by choosing the input amplitude in the linear regime. The LTI model is then used in two ways: first generate a nominal feedforward based on its approximate inverse, and then use it as the gradient in the iterative learning control (ILC) algorithm to obtain the small additional corrections for a set of motion trajectories covering the range of move lengths expected during operation. The corrective input generation is encoded in a FIR filter (for computational efficiency) with the nominal values of the coefficient obtained based

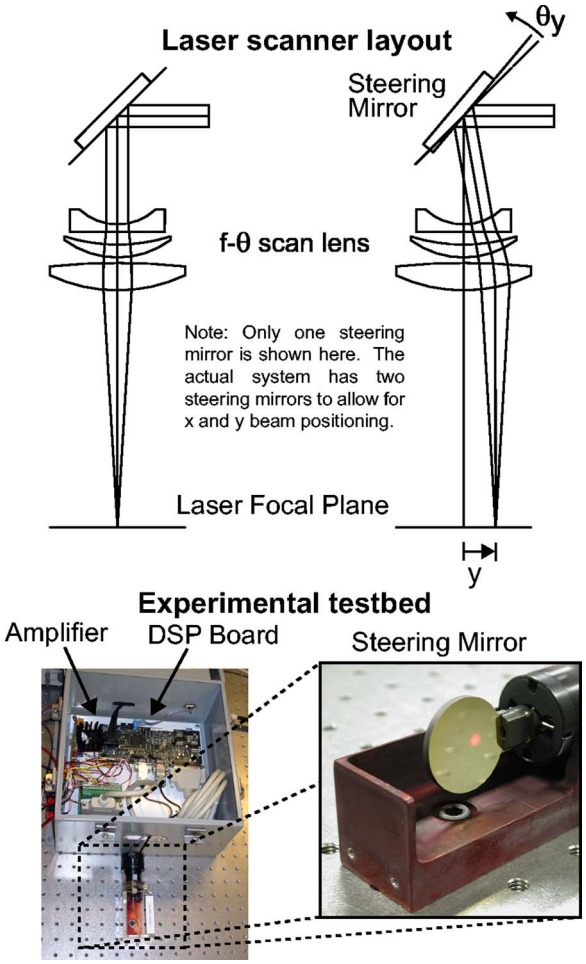

Fig. 2 Laser scanner configuration and experimental testbed. The *f-θ* lens provides a flat image field and a linear relationship between *x*, *y* coordinates and θ_x, θ_y, respectively.

on the learning results. The FIR coefficients are adaptively updated in run time using the least-mean-square (LMS) algorithm based on the position, velocity, and acceleration tracking errors. The use of inverse dynamics for the nominal feedforward allows ILC and adaptive LMS to provide only a small correction. This is important for the convergence of ILC, since only the approximate gradient (the LTI model instead of the true linear time varying (LTV) , gradient) is used, and adaptive for LMS, since the corrective signal is assumed to be linear in the FIR coefficients.

Iterative learning was first proposed for robot tracking control in Ref. [20], and has been extensively applied to motion control problems (see Ref. [21] for a survey and references therein). We have chosen the gradient algorithm [22–25], which is similar to the Newton algorithm based ILC in Refs. [8,26,27]. We use the identified LTI model in the gradient update rather than the true LTV gradient. The LTI model is a reasonable approximation for two reasons: The desired trajectory is chosen (with velocity and acceleration bounds) to avoid thermal/electronic saturation of the system, and the required correction by ILC is of very small amplitude. Using the LTI model to approximate the true LTV gradient has also been used in Ref. [27]. Adaptive feedforward control has also been extensively applied to motion control and vibration suppression [28–30]. We apply adaptation to the FIR coefficients for the desired position, velocity, and acceleration separately, and show that the position adaptation loop reduces steady state errors, velocity adaptation reduces tracking error in the high velocity zero acceleration regions, and acceleration adaptation reduces tracking error in high acceleration/deceleration regions.

The contribution of this work is to systematically integrate existing techniques in system identification, inverse dynamics control, gradient based ILC, and adaptive LMS to achieve rapid and precise motion in an industry control system. The novelty in our approach includes the use of the identified LTI model for both inverse dynamics and gradient based learning control, the use of ILC result to initialize the adaptive LMS algorithm, and the use of position, velocity, and acceleration feedforward filter adaptation to specifically target steady state, high velocity, and high acceleration/deceleration regions. Experimental results based on an industrial strength controller are also included to show the results and effectiveness of this approach.

The rest of this paper is organized as follows. Section 2 presents our methods for system identification and our identified model. Section 3 shows the result of applying an inverse filter based on the identified system as the first component of the feedforward controller. Section 4 presents an iterative refinement (i.e., ILC) based approach for producing a corrective input to the feedforward controller using a model-based gradient descent algorithm. Section 5 generates a FIR approximation to the iterative refinement corrective input that facilitates real-time corrections for arbitrary move lengths. Section 6 shows how we adapt the FIR coefficients during run time using position, velocity, and acceleration tracking errors and a gradient update algorithm. The final experimental results for tracking a random move length trajectory are shown and summarized in Sec. 7. Conclusions and future work are presented in Sec. 8.

2 Identification

LTI model identification based on input/output responses may be performed in either the time or frequency domains. An approximately 50 KHz noise signal was present in the position measurement that adversely affected the effectiveness of the time domain approach. Therefore, we decided to use the frequency domain subspace identification method [31,32]. By weighting the frequencies appropriately, we were able to emphasize the frequency band of interest while rejecting the noise in our identification process.

We first obtained the experimental frequency response of the system by stimulating the hardware with a known input signal then collecting and processing the experimental response. Input excitation is an important consideration in system identification. Common choices include impulse (could be approximately generated through an impact hammer), pulse train, sine sweep, pseudo-random binary sequence (PRBS), and Schroeder-phase signal [33]. Important attributes of a good excitation signal include excitation of the frequencies of interest, small enough amplitude to avoid saturation and other nonlinear effects, and large enough amplitude for good signal-to-noise ratio (SNR).

Working with production hardware that had limited communications ability, we were forced to implement within the memory and storage constraints of the digital signal processor (DSP) based closed loop controller board. The operating frequency of 200 kHz with only 20,000 data points available for the arbitrary input wave form and response wave form limited the complete excitation and response signals to 0.1 s in duration. Note, however, that the index into the wave form could be reset to the start of the signal upon reaching the end, which allowed for the signal to be repeated indefinitely. This capability suggested that signals that are continuous across the wraparound would be particularly desirable in order to allow for the initial transients to decay and the system to reach a steady state. A sinusoidal signal with a period equal to 0.1 s (frequency of 10 Hz) satisfies these criteria, as well as all higher order harmonics. These limitations discouraged the use of a sine sweep as the rate of frequency change was too rapid for the system to reach steady state and the signal is not continuous across the wraparound.

Both applying a series of single frequency sinusoidal signals at a time and applying a Schroeder-phase excitation gave particularly useful frequency response results when using multiples of the fundamental 10 Hz frequency and allowing the system to reach a steady state through several repetitions of the signal. The single frequency sinusoidal signal approach provided a much better SNR but was time consuming as a separate experiment was performed for every frequency of interest. The Schroeder-phase signal was attractive since it could excite multiple frequencies while satisfying a specified time domain amplitude bound. We found that the single frequency sinusoidal approach provided an excellent experimental frequency response, while the Schroeder-phase approach proved most useful for obtaining a very quick and sometimes noisy snapshot of the frequency response. For this reason, we chose the single frequency sinusoidal approach to generate the experimental frequency response used for system identification and we gathered data at 20 Hz intervals from 0 Hz to 20 kHz.

In system identification, it is important to consider possible input/output delays. If the experimental system contains a pure delay, direct application of finite dimensional LTI identification will often approximate the delay with nonminimum-phase zeros and additional poles (as in the Padé approximation of the delay). The approximation can be avoided by including a pure time delay in the identification procedure by time shifting the output signal relative to the input signal before performing the identification. Including the delay in this manner can result in identified models of lower order and better agreement between simulated and experimental responses (no artificial undershoot). Ultimately, when we implement the inverse dynamics controller, the nonminimum-phase zeros can compromise the tracking performance, while the pure delay simply leads to a time shift in the response.

The production controller did include a pure digital delay for internal signal synchronization and explicitly including the delay in our identification procedure was critical for obtaining the desired performance results. Notice how the identified plant model exhibits significant nonminimum phase-zero behavior when identified with no delay, while including a delay of 16 samples (at 5 μs sampling period) shows minimal nonminimum-phase zero contribution (see Fig. 3 for comparison). For the final identified model used in the controller design, we included a 16-sample delay pure delay and identified a 16th order model, $\hat{G}$. The identified model is compared to the experimental response with respect to the frequency response in Fig. 4. We verified the model by returning to the time domain and comparing simulated and experimental step responses, as shown in Fig. 5.

3 Inverse Dynamics Control

The next step is to use the identified model $\hat{G}$ to construct an approximate inverse dynamics filter $\hat{G}^{\dagger}$. Two aspects of this procedure need to be treated with care. First, if $\hat{G}$ is strictly proper, its inverse is improper. Therefore, higher order derivatives of y_{des} would be needed to implement $\hat{G}^{\dagger}y_{\text{des}}$ in practice. In our case, this is not a problem because the identified model has the same number of poles and zeros. However, $\hat{G}$ has a few nonminimum-phase zeros with small real parts and therefore a stable causal inverse does not exist. There are several alternative approaches. One could apply the noncausal (but stable) inverse of $\hat{G}$ to y_{des} [5]. This would effectively delay the output response by shifting the time origin. Since the real part of the zero is small, this delay could be significant. The zero phase error tracking control method was proposed in Ref. [9], which multiplies the nonminimum-phase zeros by its mirror image to achieve the zero overall phase shift. The gain, however, would deviate from 1, which would then require further compensation. We have chosen to replace the unstable zeros by their stable mirror images, and then invert the

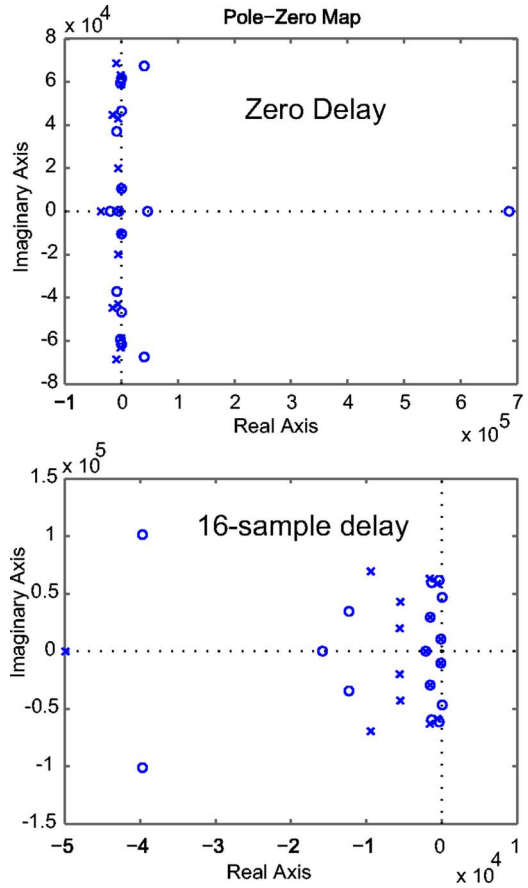

Fig. 3 Pole/zero comparison between zero-delay and 16-sample delay identified models

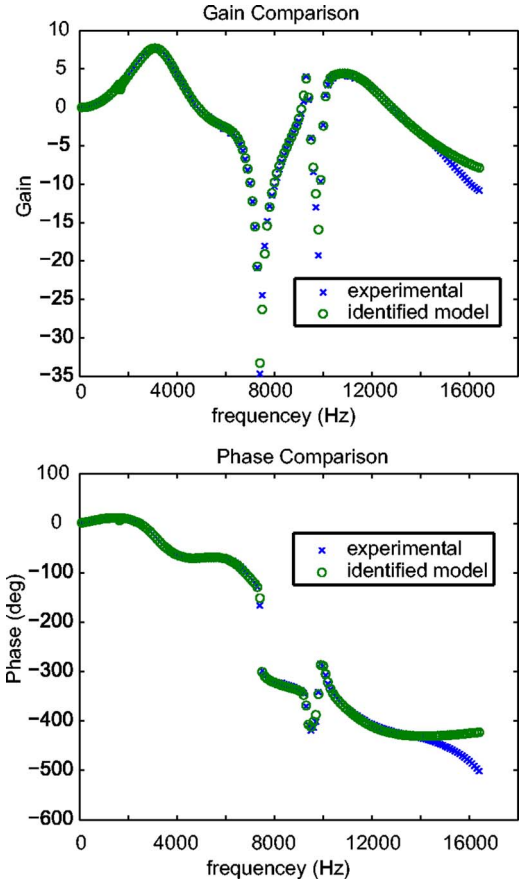

Fig. 4 Gain/phase comparison between experimental data and identified model with 16-sample delay

resulting minimum-phase transfer function (a similar method has also been suggested in Ref. [34]). The approximate inverse, $\hat{G}^{\dagger}$, is then applied to the desired output y_{des} to generate the command input u, as shown in Fig. 6. The transfer function from y_{des} to y is an all pass function, which consists of a pair of high frequency (about 7 KHz) lightly damped poles and their mirror image zeros. As a result, the phase delay of the output with respect to the desired trajectory (based on the half-sine acceleration profile) is almost negligible. In simulation, the output tracks the desired trajectory as expected. However, when this controller was implemented on the physical experiment, large tracking errors near the entry into and inside the settling zone (low velocity and high acceleration) are observed. Since this occurs in the low velocity region, the cause is likely the nonlinear friction in the positioner. The experimental result of the 500 μm move is shown in Fig. 7. A ± 1 μm band is shown in this figure and represents the maximum acceptable settling specification. We concluded that though the model matches the linear (small amplitude) behavior well, it is not good enough to meet the desired performance specification in practice.

4 Iterative Refinement

The discrepancy between the experimental and simulated responses is due to the mismatch between the model and the physical system. To correct for this mismatch, we modify the input to be $u=\hat{G}^{\dagger}y_{\text{des}}+\Delta u$, where Δu is obtained iteratively based on the output tracking error. We use the gradient ILC algorithm [22–25] to generate Δu with the identified LTI system $\hat{G}$ to approximate the gradient operator (as in Ref. [27]). The derivation of the algorithm and its convergence property are given in Appendix A.1. The overall control architecture is shown in Fig. 8. Note that y_{des} is delayed to match with the pure delay in the actual plant. The basic algorithm that we have implemented is summarized below:

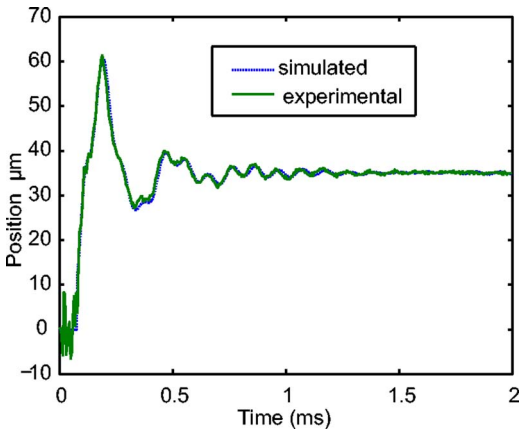

Fig. 5 Step response comparison between experimental data and identified model with 16-sample delay

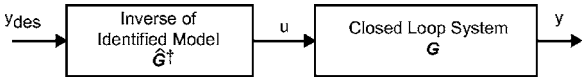

Fig. 6 Inverse dynamics feedforward control architecture using the identified model

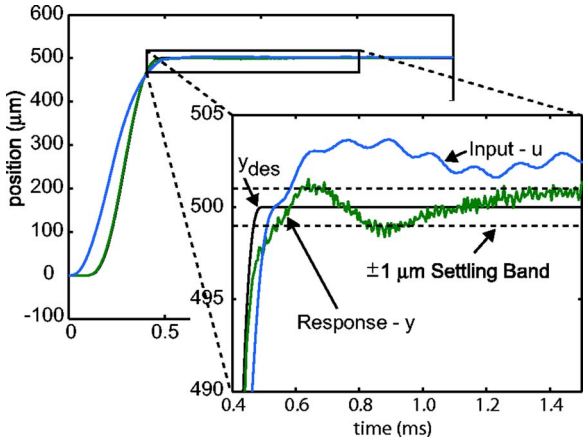

Fig. 7 Experimental results using the inverse dynamics filter for a 500 μm move length

Given $\boldsymbol{y}^* := \{\boldsymbol{y}^*(t_i) : i \in 0,1,\ldots,N\}$ and $\boldsymbol{u}_0 := \{u_0(t_i) : i \in 0,1,\ldots,N\}$ (generated from inverse dynamics). Set $u = u_0$

1. Apply $\boldsymbol{u}$ to the physical system and obtain the output sequence $\boldsymbol{y} := \{y(t_i) : i \in 0,1,\ldots,N\}$.
2. Update $\boldsymbol{u}$ by adding a corrective term

$$\Delta \underline{\mathbf{u}} = -\alpha \hat{G}^*(\underline{\mathbf{y}} - \underline{\mathbf{y}}^*) \tag{2}$$

where $\hat{G}^*$ is the adjoint of $\hat{G}$, and α may be set as a sufficiently small constant or found by using a line search (which would require additional runs).

3. Iterate until $\|\boldsymbol{y} - \boldsymbol{y}^*\|$ or $\|\Delta \boldsymbol{u}\|$ becomes sufficiently small.

The key step in the above algorithm is the updated equation (2). Let the state space parameters of $\hat{G}$ be (A,B,C,D). The adjoint $\hat{G}^*$ is given by $(-A^T,-C^T,B^T,D^T)$, but it must propagate backward in time from the zero state (see Appendix A.2). To implement $\hat{G}^* \Delta y$, we use the time reversal filtering approach as in Ref. [35]: First reverse Δy to be backward in time, filter it forward in time through the filter (A^T,C^T,B^T,D^T) (for SISO, it is the same as $\hat{G}$), and then reverse the result in time again. The procedure is illustrated in Fig. 9.

As suggested by our industrial collaborators and also independently published by other researchers [36], the forward filtering part may be performed using the actual system (by feeding the time reversed error signal into the plant as the input), thus avoiding using the analytic gradient altogether.

The result of iterative refinement for the move lengths of 500 μm and 1000 μm is shown in Fig. 10. In both cases, the output tracking error essentially converged after eight iterations. The very high frequency oscillation (around 50 kHz) in the output signal does not correspond to mirror movement but to a repeatable electrical coupling effect between the sensor and actuator electronics and other noises.

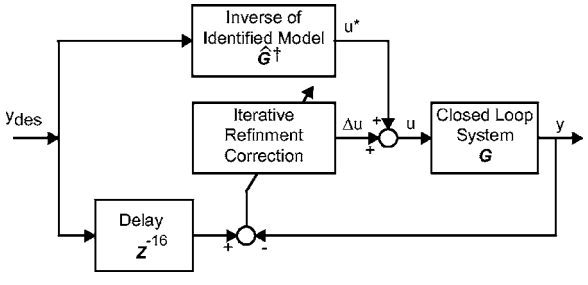

Fig. 8 Inverse dynamics combined with iterative refinement feedforward control architecture

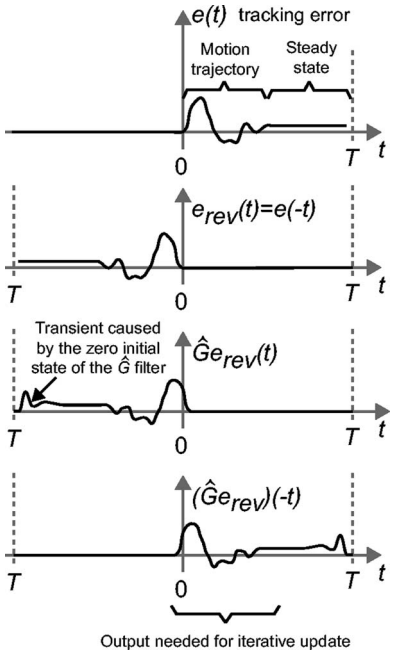

Fig. 9 Procedure of calculating G^*e

5 Finite Impulse Response Approximation

Excellent tracking performance is obtained by using iterative refinement, but the update process is nonreal time, i.e., the output error of a complete run is needed to update the input at any given time. To allow for the real-time trajectory tracking, we use the results from iterative refinement to train a filter that maps y_{des} to the corrective input, Δu. Many filter parametrizations are possible; we decided on the FIR filter structure due to its efficient real-time implementation in the DSP real-time controller, its guaranteed stability (adaptation of IIR filters can be dangerous because adaptation of the coefficients can result in an unstable filter), and the ease for finding the filter coefficients. The overall control architecture is shown in Fig. 11. In principle, it is possible to approximate the inverse dynamics portion by a FIR filter as well. However, since this approximation needs to hold for a much larger input range, a high order FIR filter and higher real-time computational load would be required. In the adaptive case, as in the next section, there may be additional convergence issues associated with the update of a large number of coefficients and the possible nonlinear effect due to the large input range.

We parametrize the corrective input by the following FIR filter:

$$\Delta u_k = w_1 y^*_{k+n_1} + w_2 y^*_{k+n_1-1} + \ \ldots \ + w_n y^*_{k-n_2+1} \tag{3}$$

where $n = n_1 + n_2$ is the order of the filter and n_1 is the look-ahead horizon. The filter coefficients, $\{w_i, i = 1,\ldots,n\}$, may be obtained through a least-squares fit to the $(y_{\text{des}}, \Delta u)$ data obtained from iterative refinement. The idea is to choose a sufficiently rich set of y_{des} so that the FIR filter is applicable to the set of desire trajectories and move lengths of interest. To find the filter coefficients,

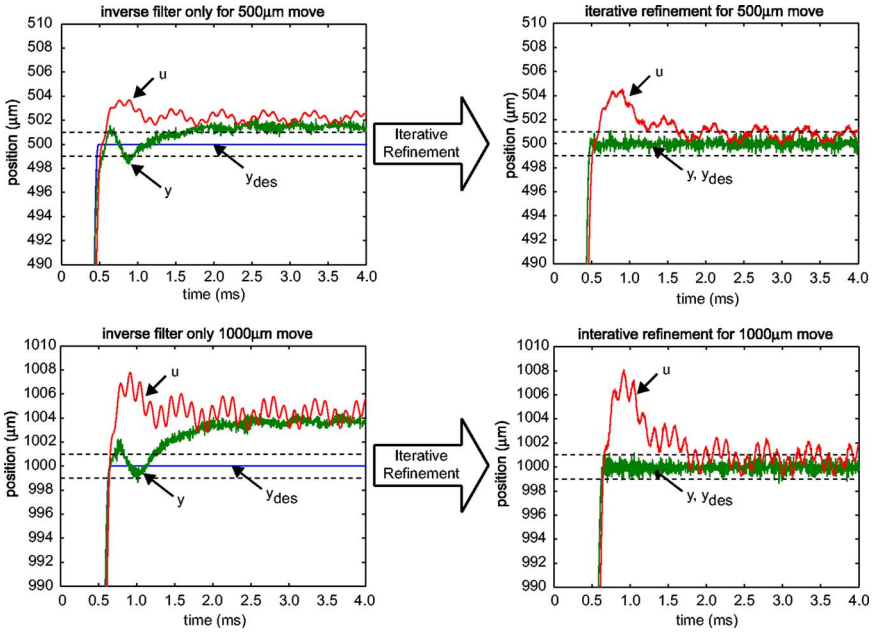

Fig. 10 Experimental results of applying iterative refinement to 500 μm and 1000 μm moves

first write the FIR equation as

$$\underbrace{\begin{bmatrix}\Delta u_{k_1}\\ \vdots \\ \Delta u_{k_2}\end{bmatrix}}_{U} = \underbrace{\begin{bmatrix} y^*_{k_1+n_1} & \cdots & y^*_{k_1-n_2+1} \\ \vdots & \ddots & \vdots \\ y^*_{k_2+n_1} & \cdots & y^*_{k_2-n_2+1}\end{bmatrix}}_{Y} \underbrace{\begin{bmatrix} w_1 \\ \vdots \\ w_n\end{bmatrix}}_{w}$$

where (k_1, k_2) (range of data used for the fit), n (order of FIR), and n_1 (look-ahead horizon) are parameters to be chosen. The least-squares solution of the filter coefficients, w, are given by

$$w = Y^{\dagger} U \tag{4}$$

where $Y^{\dagger}$ is the Moore–Penrose pseudoinverse of Y. Note that U and Y contain the iteratively refined results of a wide range of move lengths that are concatenated together. Thus, a single FIR filter is generated to work well in a least-squares sense over the complete range of expected move lengths.

Through experimentation, we decided on a 100-tap ($n=100$) FIR filter with 1 look-ahead step ($n_1=1$). For real-time implementation, the FIR runs at 20 kHz and is upsampled to 200 kHz through sample and hold (the inverse dynamics filter runs at the full 200 kHz). The iterative refinement data for move lengths from 200 μm to 700 μm at 100 μm increments were used to fit the FIR filter coefficients. The comparison between Δu for the iterative refinement case and FIR filter case for a 500 μm move is shown in Fig. 12, where the FIR reproduces the iterative refinement inputs very well. Also note that the high frequency noise generated by the iterative refinement has been eliminated by the least-squares fit operation. The output response for the 500 μm move is shown in Fig. 13. The use of an FIR approximation does an overall good job at reproducing the iterative refinement results, but does result in slightly larger errors, especially in terms of the steady state value.

6 Adaptive Finite Impulse Response Filter

The FIR approach as described requires fitting the filter coefficients to the ideal input/output responses pregenerated by the iterative refinement process applied to a range of move lengths. This presents two drawbacks: (1) As seen in the previous section, the FIR fit is not perfect, which results in a larger output tracking error; (2) when the system changes over time, the performance of the FIR filter will likely degrade. To address these issues, we include velocity and acceleration feedforward terms and apply the adaptive LMS controller [37] to adjust the FIR filter coefficients based on the output tracking error in real time. This adaptive

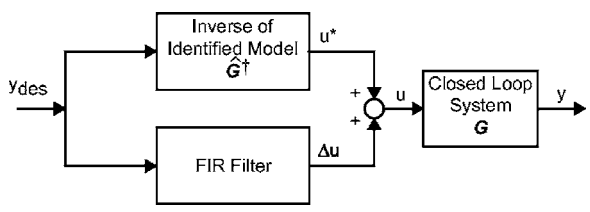

Fig. 11 Feedforward control architecture using a combination of inverse dynamics and the FIR filter

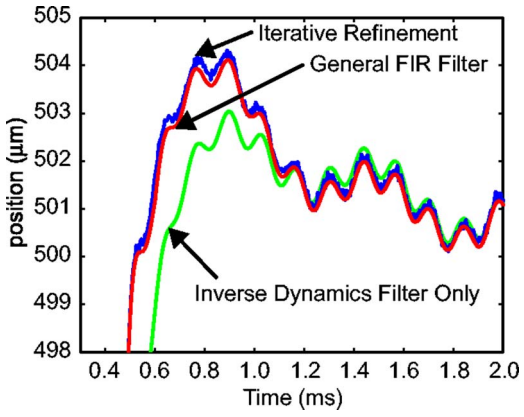

Fig. 12 Command input comparison between inverse dynamics, iterative refinement, and FIR filter: 500 μm move

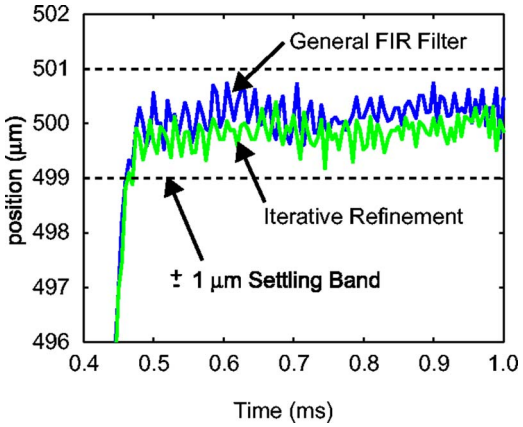

Fig. 13 Output comparison between iterative refinement and FIR filter: 500 μm move

approach is equivalent to the standard gradient parameter estimation algorithm, as shown in Appendix B. The resulting overall controller architecture is shown in Fig. 14.

We investigated an adaptive LMS architecture involving three FIR filters, with the corresponding inputs being the desired position, velocity, and acceleration:

$$\Delta u_k = w_x^T \underline{\mathbf{y}}_{\mathrm{des}k+n_{x_1}} + w_v^T \dot{\underline{\mathbf{y}}}_{\mathrm{des}k+n_{v_1}} + w_a^T \ddot{\underline{\mathbf{y}}}_{\mathrm{des}k+n_{a_1}} \tag{5}$$

where

$$\underline{\mathbf{y}}_{\mathrm{des}j} := \{y_{\mathrm{des}}(t_{i+j}) : i = 0, 1, \ldots, N_x\} \tag{6}$$

$$\dot{\underline{\mathbf{y}}}_{\mathrm{des}j} := \{\dot{y}_{\mathrm{des}}(t_{i+j}) : i = 0, 1, \ldots, N_v\} \tag{7}$$

$$\ddot{\underline{\mathbf{y}}}_{\mathrm{des}j} := \{\ddot{y}_{\mathrm{des}}(t_{i+j}) : i = 0, 1, \ldots, N_a\} \tag{8}$$

The coefficient update using adaptive LMS is given by

$$\delta w_x = -\epsilon_x (y - y^*) \hat{G}_d \underline{\mathbf{y}}^*$$

$$\delta w_v = -\epsilon_v (y - y^*) \hat{G}_d \dot{\underline{\mathbf{y}}}^*$$

$$\delta w_a = -\epsilon_a (y - y^*) \hat{G}_d \ddot{\underline{\mathbf{y}}}^* \tag{9}$$

where $\hat{G}_d$ is the identified LTI system represented in discrete time, y^* is the delayed y_{des} (by 16 samples), and $(\underline{\mathbf{y}}^*, \dot{\underline{\mathbf{y}}}^*, \ddot{\underline{\mathbf{y}}}^*)$ are the delayed version of the output position, velocity, and acceleration time sequences. Through some initial experimentation, we decide to use 100 taps for the position portion of FIR, and 4 taps each for the velocity and acceleration portion. The FIR filter and adaptive LMS are both implemented at 20 kHz.

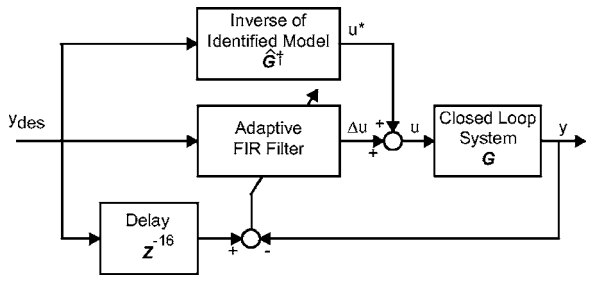

Fig. 14 Feedforward control architecture using a combination of inverse dynamics and adaptive FIR filtering

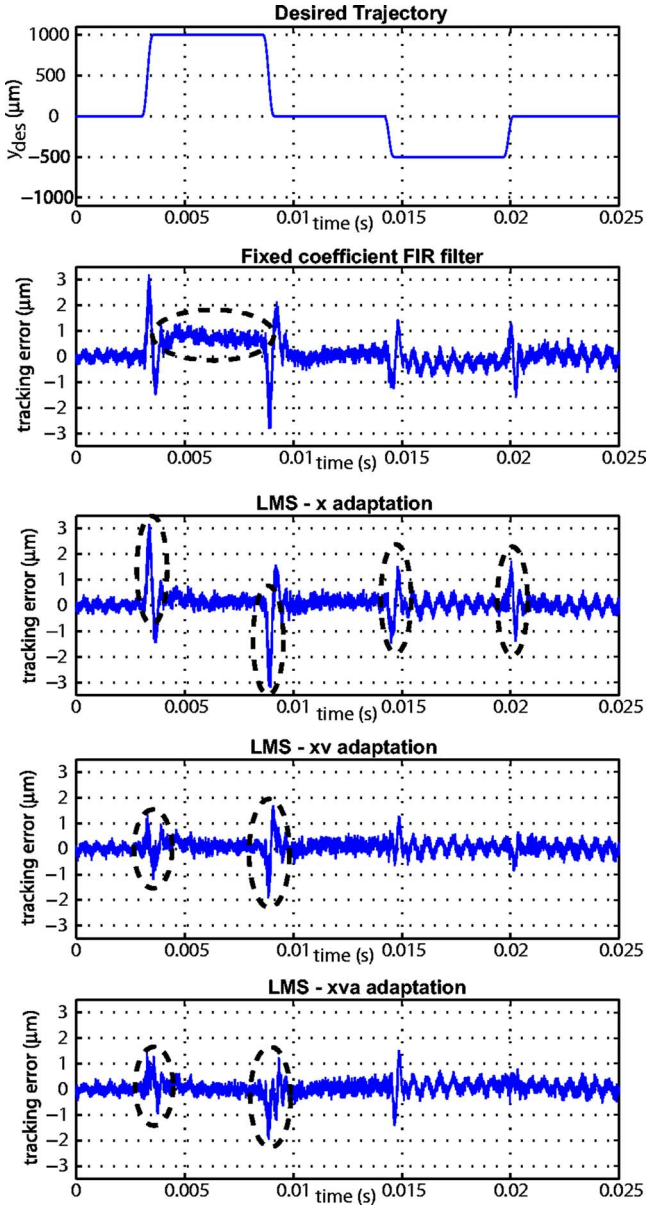

Fig. 15 Comparison between fixed coefficient FIR, LMS updating position only, LMS updating position and velocity, and LMS updating position, velocity, and acceleration

Figure 15 in the top panel shows a desired motion trajectory with a 1000 μm up and a 500 μm down movement. The performance of the fixed coefficient FIR described in Sec. 5 is shown in the next panel. Note the presence of steady state error as well as large tracking error in the high velocity and acceleration regions (highlighted in the figure). Using this filter as a starting point, we first apply adaptive LMS to only the position part of the filter in Eq. (5) (i.e., w_x is updated by the first equation in Eq. (9), w_v and w_a remain zero). The result after 330 repeated runs (to allow the coefficients to reach steady state) is shown in the third panel. The steady state error is reduced since the position error strongly affects the update of w_x. However, the transient error in the high velocity region remains.

We next add the velocity feedforward filter and apply adaptive LMS to the position and velocity portions of the FIR filter (5) (i.e., w_x and w_v are updated by the first two equations in Eq. (9) with w_v initially set at zero, w_a remains zero). The tracking error is shown in the fourth panel. Compare the fourth panel to the third

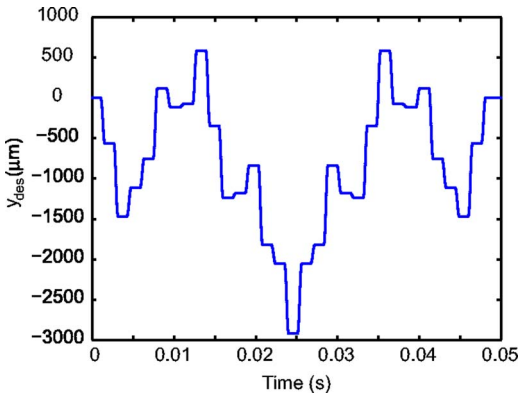

Fig. 16 Desired output trajectory

panel. The result is quite striking: the peak tracking error in the high velocity range is reduced by about 2/3. This is a direct result of using w_v to reduce the velocity tracking error. The steady state error also remains small due to w_x.

Lastly, we use all three portions (position, velocity, acceleration) of the FIR filter in Eq. (5) and update them using the adaptive LMS rule (9). The result is shown in the last panel of 15. There is some improvement in the 1000–0 μm transition region, but the improvement is inconclusive in other high acceleration regions. This may be due to the fact that the tracking error is already quite small (less than 2 μm throughout the move). Thus the coefficient update is very slow. Another factor is the presence of noise, which limits the amount of achievable tracking performance.

7 Random Move Result

To compare the performance of the various approaches described so far, we used a random motion profile as a test case. The desired trajectory is shown in Fig. 16. A FIR filter was trained using the results of move lengths 200–700 μm at 100 μm intervals as was described in Sec. 5. The comparison between the tracking errors for the inverse dynamics filter and fixed-coefficient FIR filter are shown in Fig. 17. As expected, the tracking error is significantly and uniformly reduced in the fixed FIR case. The addition of the adaptive LMS filters further improves the response, as shown in Fig. 18. The adaptive filter clearly shows an improvement over the FIR filter with the *xv* and *xva* filters demonstrating the best performance. A quantitative comparison between these different controllers is summarized in Table 1. The adaptive *xv*-filter gives the best performance in terms of rms, mean, and standard deviation of the trajectory tracking error. The adaptive *xva*-filter is slightly better in terms of the peak error. Based on this result, the adaptive *xv*-filter appears to be the best overall choice in terms of performance and algorithm complexity.

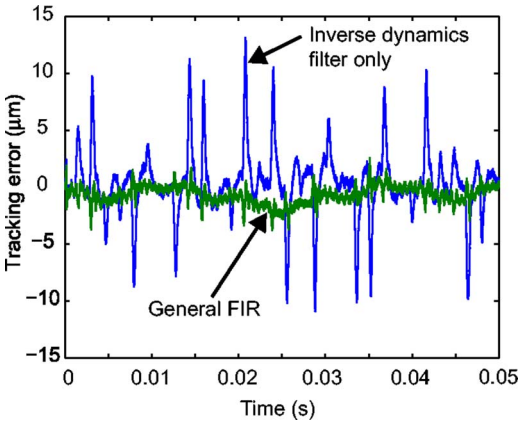

Fig. 17 Tracking error for random moves: inverse dynamics versus FIR filter

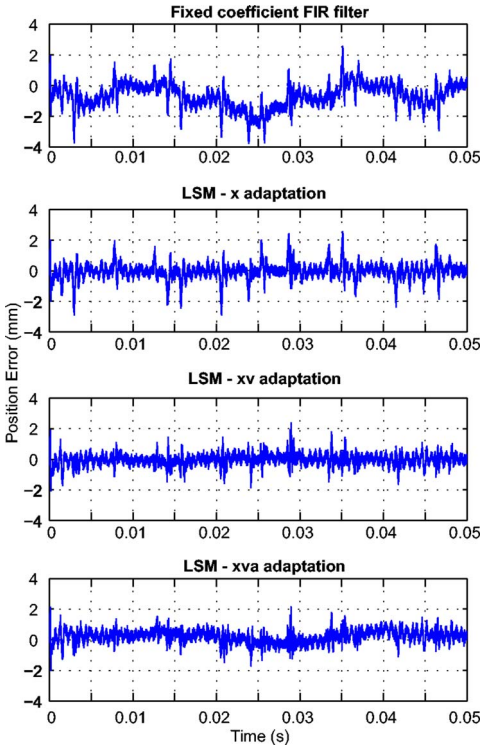

Fig. 18 Tracking error for random moves: comparison between fixed coefficient FIR, LMS updating position only, LMS updating position and velocity, and LMS updating position, velocity, and acceleration

8 Conclusion

This paper presents the experimental results of a high performance feedforward controller for a positioning system used in modern electronics processing equipment. A LTI model is first identified by using the frequency domain subspace identification method. The identified model is used in an inverse dynamics approach to generate the nominal feedforward control, but significant tracking error still remains due to model nonlinearity. By applying the gradient based ILC using the identified model as an approximate gradient, a corrective input is generated to reduce the tracking error to about 1 μm. The ILC results are next used to train an FIR filter to allow for real-time trajectory tracking. Finally, the FIR filter coefficients are updated by using an adaptive LMS scheme. It is critical to combine the inverse dynamics with ILC and adaptive LMS: the corrective input is sufficiently small so that ILC and adaptive LMS may be effectively applied. Application to the tracking of a random motion profile shows excellent trajectory tracking capability by using the proposed method. A variation of this work has been implemented in the production machine and our current focus is on extending this approach to motion system containing a significant amount of Coulomb friction and other nonsmooth nonlinearities.

Acknowledgment

This work is supported in part by the Electro Scientific Industries, Inc., the National Science Foundation under Grant No. CMS-0301827, and the Center for Automation Technologies and Systems (CATS) under a block grant from the New York State Office of Science, Technology, and Academic Research

Table 1 Performance comparison for a random move profile

All units in μm	Min	Max	rms	Mean	Stdev
Inv dyn	−10.88	13.11	2.96	0.29	2.94
FIR	−3.77	2.56	1.04	−0.68	0.79
Adaptive x	−2.91	2.57	0.52	−0.05	0.52
Adaptive xv	−2.07	2.38	0.37	−0.03	0.37
Adaptive xva	−1.99	2.15	0.46	0.21	0.41

(NYSTAR). J.T.Y.W. is supported in part by the Outstanding Overseas Chinese Scholars Fund of Chinese Academy of Sciences (No. 2005-1-11).

Appendix A: Gradient Based Iterative Refinement Algorithm

A.1 Convergence Property

We first show that an iterative gradient update scheme of the input will lead the output tracking error to converge to zero, provided that the gradient is full rank.

Consider a nonlinear dynamical system $H: L_2^m[0,T] \rightarrow L_2^p[0,T]$ that is initially at rest:

$$y = H(u) \tag{A1}$$

Define

$$V = \frac{1}{2}\|y - y_{\text{des}}\|_{L_2^p}^2 \tag{A2}$$

Note that a positive time dependent weighting may be used in the above norm with obvious modifications in the subsequent update law. The variation of V due to a variation in y is

$$\delta V = \langle y - y_{\text{des}}, \delta y \rangle_{L_2^p} \tag{A3}$$

where $\langle \cdot, \cdot \rangle_{L_2}$ denotes the L_2 inner product. From Eq. (A1), δy is

$$\delta y = \nabla_u H \delta u \tag{A4}$$

where $\nabla_u H$ is the Fréchet derivative of H. Suppose H is given by the following state space representation:

$$\dot{x} = f(x,u) \quad x(0) = x_0 \quad y = h(x,u) \tag{A5}$$

Then $\nabla_u H$ is the time varying linearized system about x and u (x is the state trajectory corresponding to u):

$$\delta \dot{x} = \frac{\partial f(x,u)}{\partial x}\delta x + \frac{\partial f(x,u)}{\partial u}\delta u \quad \delta x(0) = 0$$

$$\delta y = \frac{\partial h(x,u)}{\partial x}\delta x + \frac{\partial h(x,u)}{\partial u}\delta u \tag{A6}$$

where $\nabla_u H$ is the operator that maps δu to δy.

The gradient descent algorithm chooses δu based on $\nabla_u H^*$:

$$\delta u = -\alpha \nabla_u H^*(y - y_{\text{des}}) \tag{A7}$$

If this can be implemented exactly, then from Eq. (A4)

$$\delta y = -\alpha(\nabla_u H)(\nabla_u H^*(y - y_{\text{des}})) \tag{A8}$$

which implies

$$\delta V = -\alpha\|\nabla_u H^*(y - y_{\text{des}})\|_{L_2^p}^2 \tag{A9}$$

The actual implementation of Eq. (A7) would be the following iteration:

$$u_{k+1} = u_k - \alpha_k \nabla_u H^*(u_k)(y_k - y_{\text{des}}) \tag{A10}$$

where α_k is found through a line search to ensure V_{k+1} (V with u_{k+1}) is strictly smaller than V_k. Such α_k can always be found provided that $\nabla_u H(u_k)$ is onto, i.e., $\nabla_u H(u_k)\nabla_u H^*(u_k)$ is positive definite. In this case, V_k converges to zero as $k \rightarrow \infty$.

A.2 Computation of the Gradient Operator

We now consider the computation of $\nabla_u H^*$. Write $\nabla_u H$ as a linear time varying system (same as Eq. (A6)):

$$\delta \dot{x} = A(t)\delta x + B(t)\delta u, \quad \delta y = C(t)\delta x + D(t)\delta u \tag{A11}$$

Let $\Phi(t,\tau)$ be the state transition matrix from τ to t. The output can now be written as

$$\delta y(t) = C(t)\int_0^t \Phi(t,\tau)B(\tau)\delta u(\tau)d\tau + D(t)\delta u(t)$$

To find $\nabla_u H^*$, we use the defining relationship $\langle \delta v, \nabla_u H \delta u \rangle_{L_2^p} = \langle \nabla_u H^* \delta v, \delta u \rangle_{L_2^m}$ for all δv in $L_2^p[0,T]$ and all δu in $L_2^m[0,T]$. We then obtain

$$\nabla_u H^* \delta v(\tau) = \int_\tau^T B^T(\tau)\Phi^T(t,\tau)C^T(t)\delta v(t)dt + D^T(\tau)\delta v(\tau) \tag{A12}$$

Define

$$z(\tau) = \int_\tau^T \Phi^T(t,\tau)C^T(t)\delta v(t)dt \tag{A13}$$

The state space representation of $\nabla_u H^*$ is then given by the mapping from δv to w as follows:

$$\dot{z}(t) = -A^T(t)z(t) - C^T(t)\delta v(t) \quad z(T) = 0$$

$$w(t) = B^T(t)z(t) + D^T(t)\delta v(t) \tag{A14}$$

Equation (A14) needs to be integrated backward in time. Through a change of variable, $z_1(t) = z(T-t)$, we can instead use the forward solution. The adjoint system in this case is then (with $z_1(0) = 0$ and $\delta v_1(t) := \delta v(T-t)$):

$$\dot{z}_1(t) = A^T(T-t)z_1(t) + C^T(T-t)\delta v_1(t)$$

$$w(T-t) = B^T(T-t)z_1(t) + D^T(T-t)\delta v_1(t) \tag{A15}$$

This can be implemented by reversing δv backward in time, filtering it through Eq. (A15), and then reversing the result in time again.

Appendix B: Adaptive Finite Impulse Response Algorithm

Consider a SISO discrete time LTI system, $y = Hu$. Choose

$$u = \underline{\mathbf{y}}_{\text{des}}^T \mathbf{w} + u^* \tag{B1}$$

where $\underline{\mathbf{y}}_{\text{des}}$ is a vector of the delayed desired output, $\mathbf{w}$ is the weighting vector, and u^* is a feedforward control that depends on y_{des} only. Then the output is

$$y = (H\underline{\mathbf{y}}_{\text{des}})^T \mathbf{w} + Hu^* \tag{B2}$$

Assume that there exists a constant weighting vector w_o such that the desired output could be tracked exactly:

$$y_{\text{des}} = (H\underline{\mathbf{y}}_{\text{des}})^T w_o + Hu^* \tag{B3}$$

Our goal is to choose an update law for w so that $y \to y_{\text{des}}$ even though w_o is unknown.

To simplify notation, let $a = H\underline{\mathbf{y}}_{\text{des}}$, and write Eq. (B3) in discrete time as

$$y_{\text{des}}(t) - (Hu^*)(t) = a^T(t-1)w_o \tag{B4}$$

where the left hand side and $a(t-1)$ are known. The problem then becomes a standard least-squares parameter estimation problem of estimating w_o based on the output error $y_{\text{des}}(t) - (Hu^*)(t) - a^T(t-1)\mathbf{w}(t-1)$, where $\mathbf{w}(t-1)$ is the estimate of w_o at time $t-1$ [38] (Sec. 3.3). Various algorithms, ranging from projection to least squares with covariance resetting may be used. To reduce the real-time computation load, we use the gradient update law:

$$\mathbf{w}(t) = \mathbf{w}(t-1) + \epsilon a(t-1)(y_{\text{des}}(t) - (Hu^*)(t) - a^T(t-1)\mathbf{w}(t-1)) \tag{B5}$$

Substituting Eq. (B2) into Eq. (B5), we obtain the following update law for $\mathbf{w}(t)$, which may be efficiently implemented in real time:

$$\mathbf{w}(t) = \mathbf{w}(t-1) - \epsilon a(t-1)(y(t) - y_{\text{des}}(t)). \tag{B6}$$

Other types of parameter estimation algorithms with more relaxed convergence conditions may be used (e.g., projection, least square with covariance reset, etc.). However, the real-time computation load would be more demanding.

References

[1] Gu, B., 2005, "Ultrafast Laser Processing for Next-Generation Memory Repair," Proc. SPIE, **5714** pp. 37–42.

[2] Lei, W., and Davignon, J., 2005, "Solid State UV Laser Technology for Electronic Packaging Applications," Proc. SPIE, **5629** pp. 314–326.

[3] ACS-Tech80 Inc., 2001, *Engineering Handbook*, ACS Tech80, Maple Grove, MN, Version 3.

[4] Silverman, L. M., 1969, "Inversion of Multivariable Linear Systems," IEEE Trans. Autom. Control, **AC-14**(3), pp. 270–276.

[5] Moylan, P. J., 1977, "Stable Inversion of Linear Systems," IEEE Trans. Autom. Control, **22**, pp. 74–78.

[6] Bayo, E., and Paden, B., 1987, "On Trajectory Generation for Flexible Robots," J. Rob. Syst., **4**(2), pp. 229–235.

[7] Bayo, E., and Moulin, H., 1989, "An Efficient Computation of the Inverse Dynamics of Flexible Manipulators in the Time Domain," *Proceedings of 1989 IEEE Robotics and Automation Conference*, Scottsdale, AZ, pp. 710–715.

[8] Devasia, S., Chen, D., and Paden, B., 1996, "Nonlinear Inversion-Based Output Tracking," IEEE Trans. Autom. Control, **41**(7), pp. 930–942.

[9] Tomizuka, M., 1987, "Zero Phase Error Tracking Algorithm for Digital Control," ASME J. Dyn. Syst., Meas., Control, **109**, pp. 65–68.

[10] Funahashi, Y., Yamada, M., and Fujiwara, S., 1997, "Zero Phase Error Tracking System With Arbitrarily Specified Gain Characteristics," ASME J. Dyn. Syst., Meas., Control, **119**, pp. 260–264.

[11] Funahashi, Y., Yamada, M., and Riadh, Z., 1999, "Generalized Optimal Zero Phase Error Tracking Controller Design," ASME J. Dyn. Syst., Meas., Control, **121**, pp. 165–169.

[12] Francis, B. A., 1987, *A Course in H_∞ Optimization Theory*, Lecture Notes in Control and Information Sciences, Springer-Verlag, New York, Vol. 88.

[13] Doyle, J. C., Francis, B. A., and Tannenbaum, A. R., 1992, *Feedback Control Theory*, Macmillan, London.

[14] Smith, O. J. M., 1958, *Feedback Control Systems*, McGraw-Hill Book Company, New York.

[15] Singer, N. C., and Seering, W. P., 1990, "Preshaping Command Inputs to Reduce System Vibration," ASME J. Dyn. Syst., Meas., Control, **112**, pp. 76–82.

[16] Tuttle, T. D., and Seering, W. P., 1994, "A Zero-Placement Technique for Designing Shaped Inputs to Suppress Multiple-Mode Vibration," *American Control Conference*, Baltimore, MD, pp. 2533–2537.

[17] Franklink, G. F., Powell, J. D., and Emami-Naeini, A., 1994, *Feedback Control of Dynamic Systems*, 3rd ed., Addison-Wesley, Reading, MA.

[18] Magee, D. P., and Book, W. J., 2000, "Optimal Arbitrary Time-Delay (OAT) Filter and Method to Minimize Unwanted System Dynamics," U.S. Patent No. 6078844.

[19] Weiner, D., 1997, "Design Considerations for Optical Scanning," Proc. SPIE, **3131**, pp. 59–64.

[20] Arimoto, S., 1985, "Mathematical Theory of Learning With Applications to Robot Control," *Adaptive and Learning Systems: Theory and Applications*, K. S. Narendra, ed., Plenum, New York.

[21] Moore, K., 1999, "Iterative Learning Control: An Expository Overview," *Applied and Computational Controls, Signal Processing, and Circuits*, 1(1), pp. 425–488.

[22] Yamakita, M., and Furuta, K., 1991, "Iterative Generation of Virtual Reference for a Manipulator," Robotica, **9**, pp. 71–80.

[23] Gorinevsky, D. M., 1992, "Direct Learning of Feedforward Control for Manipulator Path Tracking," *Proceedings of the 1992 IEEE International Symposium on Intelligent Control*, Glasgow, UK, pp. 42–47.

[24] Cheng, W., and Wen, J. T., 1993, "A Class of Learning Controllers With Application to Tracking Control of a Flexible Beam," *Proceedings, 1993 IEEE Conference on Robotics and Automation*, Atlanta, GA, pp. 411–416.

[25] Kinosita, K., Sogo, T., and Adachi, N., 2002, "Iterative Learning Control Using Adjoint Systems and Stable Inversion," Asian Journal of Control, **4**(1), pp. 60–67.

[26] Avrachenkov, K. E., 1998, "Iterative Learning Control Based on Quasi-Newton Methods," *Conference on Decision and Control*, Tampa, FL, pp. 170–174.

[27] Ghosh, J., and Paden, B., 2002, "A Pseudoinverse-Based Iterative Learning Control," IEEE Trans. Autom. Control, **47**(5), pp. 831–837.

[28] Davis, L., Hyland, D., Yen, G., and Das, A., 1999, "Adaptive Neural Control for Space Structure Vibration Suppression," Smart Mater. Struct., **8**, pp. 753–766.

[29] Ma, K., and Ghasemi-Nejhad, M. N., 2004, "Frequency-Weighted Adaptive Control for Simultaneous Precision Positioning and Vibration Suppression of Smart Structures," Smart Mater. Struct., **13**, pp. 1143–1154.

[30] Fei, J., and Song, G., 2004, "Adaptive Feedforward Model Reference Control Scheme for Vibration Suppression," *Proceedings of the 2004 IEEE International Symposium on Intelligent Control*, Taipei, Taiwan, pp. 357–362.

[31] McKelvey, T., Akcay, H., and Ljung, L., 1996, "Subspace-Based Multivariable System Identification From Frequency Response Data," IEEE Trans. Autom. Control, **41**(7), pp. 960–979.

[32] Van Overschee, P., and De Moor, B., 1996, "Continuous-Time Frequency Domain Subspace System Identification," Signal Process., Special Issue on Subspace Methods, Part: System Identification, **52**, pp. 179–194.

[33] Bayard, D., 1992, "Statistical Plant Set Estimation Using Schroeder-Phased Multisinusoidal Input Design," *Proceedings 1992 American Control Conference*, Chicago, IL, pp. 2988–2995.

[34] Markusson, O., Hjalmarsson, H., and Norrlof, M., 2001, "Iterative Learning Control of Nonlinear Nonminimum Phase Systems and Its Application to System and Model Inversion," *Proceedings of IEEE Conference on Decision and Control*, pp. 4481–4482.

[35] Oppenheim, A. V., Schafer, R. W., and Buck, J. R., 1999, *Discrete-Time Signal Processing*, 2nd ed., Prentice-Hall, Englewood Cliffs, NJ.

[36] Ye, Y., and Wang, D., 2005, "Zero Phase Learning Control Using Reversed Time Input Runs," ASME J. Dyn. Syst., Meas., Control, **127**, pp. 133–139.

[37] Haykin, S., 1996, *Adaptive Filter Theory*, 3rd ed., Prentice-Hall, Englewood Cliffs, NJ.

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