2008 American Control Conference
Westin Seattle Hotel, Seattle, Washington, USA
June 11-13, 2008

Semi-Automatic Tuning of PID Gains for Atomic Force Microscopes

Daniel Y. Abramovitch, Storrs Hoen, and Richard Workman

***Abstract*— The control of a typical commercial Atomic Force Microscope (AFM) is through some variant on a Proportional, Integral, Derivative (PID) controller. Typically, the gains are hand tuned so as to keep the bandwidth of the system far below the first resonant frequency of the actuator. This paper shows a straightforward method of selecting PID gains from the actuator model so as to allow considerably higher bandwidths.**

I. Introduction

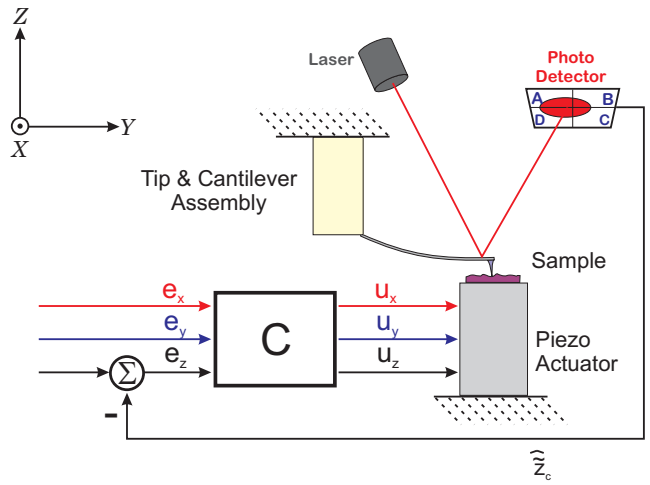

Fig. 1. A block diagram of a typical Atomic Force Microscope (AFM). In this design, the sample is scanned in X, Y, and Z. Typically, feedback is always done on the Z axis, but may also be done on the X and Y axes if sensors are added to these directions of motion.

A block diagram of a typical Atomic Force Microscope (AFM) control loop is shown in Figure 1. As mentioned in [1] typical AFM the control loops are implemented using PI, PII, or PID controllers. The PI and PII controllers are often chosen specifically because the limited knowledge of the piezo tube actuator models and high levels of noise in the optical detection of the cantilever deflection make it necessary to limit the bandwidth of the loop to far below the main resonant mode of the actuator.

In this paper, a new approach is shown which allows for substantially higher bandwidth in the loop. The AFM actuator is modeled as a simple resonance and a PID controller is then generated that affects an inverse dynamics model of this resonance. The resulting open loop response allows for a crossover frequency substantially above the actuator resonance. The fact that this procedure requires relatively few operator interactions makes it substantially easier for AFM users to generate high performance control.

Daniel Abramovitch, Storrs Hoen, and Richard Workman, Nanotechnology Group at Agilent Laboratories, 5301 Stevens Creek Blvd., M/S: 4U-SB, Santa Clara, CA 95051 USA, danny@agilent.com

II. Algorithm Outline

This algorithm can be summarized as follows:

- Typically a simple model of an AFM actuator is a simple second order resonance.
- The inverse dynamics for a resonance resembles a notch filter.
- A second order filter model of a PID controller has high gain at low frequency, high gain at high frequency, and a dip in the middle. In other words, it resembles a notch filter.
- The PID gains can be related to the notch parameters to cancel the selected dynamics of the plant model.
- This allows for near automatic gain generation of the PID parameters. These gains, which generate an inverse dynamics of the resonance allow an open loop bandwidth well beyond the main resonant frequency.
- Provided that the measurement noise is minimized and the cantilever dynamics are kept out of the main loop, this allows for substantial increases in closed-loop bandwidth.

The rest of this paper is organized as follows. Section III gives a very brief sketch of the AFM Control problem. Section IV does a quick review of PID control. Section V discusses fitting frequency response functions of an AFM actuator to a second order model. Section VI describes using a PID controller as a dynamic inverse feedback controller. Section VII describes the backyards rectangular rule implementation of a discrete PID controller. We close with a design example in Section VIII.

III. AFM Control

An example set of idealized frequency response curves for a piezo tube is shown in Figure 2. The piezo tube resonances shown here are around 1 kHz, which is in the typical range of 500 Hz to 20 kHz. Some experimental systems have resonances above 40 kHz [3], [4]. In Figure 2 a series of five models of the piezo-cantilever system are plotted with the resonant frequency varying between 900 Hz and 1.1 kHz, and the quality (Q) factors varying between 10 and 30. These idealized models were discussed in the tutorial paper [1].

Because the piezo actuator is modeled as a second-order resonance, the lack of integrators in the forward path necessitates the use of integral action for zero steady-state error to any steps in the surface height. The addition of a second

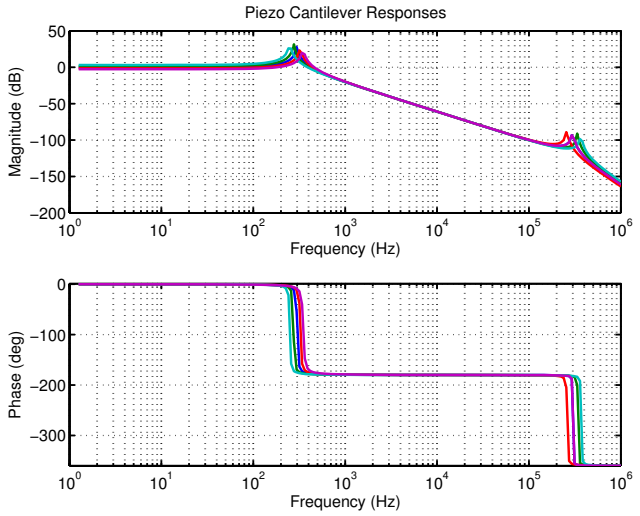

Fig. 2. A set of "generic" AFM plants. This shows the combination of the Z-piezo actuator and a 300 kHz cantilever. Note that hysteresis, creep, and nonlinearity in the piezo [2] makes the exact modeling of a given actuator difficult, and thereby hampers the control. The cantilever properties also vary considerably within a batch.

integrator via PII control can provide zero steady-state error to surface slopes, which are common in many samples. Such controllers are necessarily low bandwidth, since the lack of phase lead means that the gain must be rolled off below the resonance of the actuator.

The typical industrial AFM control loop, whether done in contact or dynamic mode [1], is a low frequency PI or PII loop. A general form of an analog controller that admits PI, PD, PID, PII, and even PIID is:

$$C(s) = \frac{U(s)}{E(s)} = K_P + \frac{K_I}{T_I s} + \frac{K_{II}}{(T_I s)^2} + K_D T_D s, \quad (1)$$

where $E(s)$ is the Laplace transform of the error signal $e(t)$. The constants T_I and T_D are the time intervals over which integration and differentiation are done. Putting these terms explicitly in PID equations such as (1) is optional, but has some advantages mentioned in Section IV and in more detail in [5] For a P, PI, PII, or PID controller, one or more of the K_D, K_I, or K_{II} gains are set to zero. Note that as written the derivative term, $K_D s$, is not practically implementable, but this is often rectified by having some low pass filter added to it. For digital implementation, the backward rectangular integration rule is most often used for PID controllers since this allows for direct translation from (1) [6], [7].

It is tempting to try to increase the bandwidth of the system by adding phase lead, such as with a PID controller. However, the use of this is limited by the uncertainty in the modeling of the piezo actuator. Furthermore, boosting the bandwidth with a PID requires lower noise in the optical measurement of deflection, otherwise this noise will be amplified by the effects of the derivative term.

However, if the actuator can be accurately modeled and the optical noise can be limited, then the loop bandwidth can be raised significantly through the use of PID control. In particular, if the control parameters are chosen to give an inverse dynamic model of the actuator, then the resulting open-loop system will have high bandwidth with reasonable margins.

IV. PID Control

There are multiple forms for PID controllers which can be related to second order sections. These relationships receive a more complete treatment in [5]. For this paper, the time and frequency domain forms are given by

$$u(t) = K_P e(t) + \frac{K_I}{T_I} \int_0^t e(\tau) d\tau + K_D T_D \dot{e}(t) \quad (2)$$

and

$$C(s) = \frac{U(s)}{E(s)} = K_P + \frac{K_I}{T_I s} + K_D T_D s, \quad (3)$$

respectively. These forms have two distinguishing features:

- explicit time specification and
- no differentiator filtering.

Explicit time specification simply refers to whether the T_I and T_D terms are explicitly specified in the PID equation, or whether they are absorbed into K_I and K_D, respectively. It's perfectly legitimate to have

$$K_{I,i} = \frac{K_I}{T_I} \quad \text{and} \quad K_{D,i} = K_D T_D, \quad (4)$$

where $K_{I,i}$ and $K_{D,i}$ can be considered "implicit time" versions of the integral and differential gains. However, leaving the T_I and T_D terms in the equation give the designer some flexibility *and* also allow these terms to drop out when the discrete-time PID is generated. It is also easy to go from explicit to implicit time simply by setting $T_D = T_I = 1$. However, doing that makes the common backward rule discrete equivalent more complicated.

The second option is differentiator filtering, in which the differential term is made practical with a low pass filter. Differentiator filtering is necessary because an ideal differentiator will have huge gain at high frequency. More practically, we know that an analog differentiator will eventually roll off. However, it makes sense to include this in the controller design. It likely escapes much mention in the literature because most discrete implementations of PID control use a backward rule equivalent, which provides a realizable filter even when the differentiator is not filtered. This will be seen in Section VII.

It should be obvious that we can put these separate terms into one transfer function. What may not be obvious is how this will look once it is combined. This section explains all of that.

Equation (3) can be related to a second order section (in the numerator) and this can be interpreted as a notch filter. Thus, by setting the parameters of the PID, we can set the parameters of the notch. More importantly, from the parameters of a notch, we can set the PID parameters. Section IV-A will relate the analog PID with the ideal differentiator from (3) to a second order section.

A. Unfiltered PID and Second Order Sections

Starting with (3), let's set $T_D = T_I = T$ and put everything over a common denominator:

$$C(s) = \frac{K_D T}{s}\left[s^2 + \frac{K_P}{K_D}\frac{s}{T} + \frac{K_I}{K_D T^2}\right], \quad (5)$$

$$= \frac{K_D T s^2 + K_P s + \frac{K_I}{T}}{s}. \quad (6)$$

Note that while (5) allows us to solve for the numerator parameters as a second order section, (6) is a standard numerator/denominator form that we might use in Matlab. Note, however, that this implementation of the PID is not proper, i.e., the numerator is of higher order than the denominator. Practically speaking, nothing gets realized this way for one of two reasons:

1) Any real differentiation circuit eventually flattens out, so there is some low pass filter, even if it's not acknowledged in the design.
2) Discretizing the PID using a backward rectangular rule fixes this and makes the discrete-time transfer system proper. Discretizing a PID without filtering of the derivative using the trapezoidal rule results in a compensator pole a $z = -1$, which is a ringing pole inside the compensator. So, while the closed-loop system might be stable, we wouldn't have internal stability. This is not a good thing for an intuitive controller such as a PID.

The numerator also has the form of a second order section i.e.,

$$N(s) = \frac{K}{\omega_n^2}\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right), \quad (7)$$

where we have normalized the DC gain of this section to K. Now we should be able to set

$$\frac{K_D T}{s}\left[s^2 + \frac{K_P}{K_D}\frac{s}{T} + \frac{K_I}{K_D T^2}\right] = \frac{K}{\omega_n^2 s}\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right). \quad (8)$$

If we were simply to try to match $N(s)$ portion of (7) then we might choose to match the DC gain to some prespecified value, K. However, the form that we want our PID to match in (8) has an integrator in it, making the DC gain infinite. So, we need to pick a frequency and gain that we wish to match and then evaluate the right side of (8) to match at that frequency. More specifically, if

$$\frac{N(s)}{s} = \frac{K}{\omega_n^2 s}\left(s^2 + 2\zeta\omega_n s + \omega_n^2\right), \quad (9)$$

then

$$\frac{N(j\omega_0)}{j\omega_0} = \frac{K}{j\omega_0\omega_n^2}\left(\omega_n^2 - \omega_0^2 + j2\zeta\omega_n\omega_0\right) \quad (10)$$

$$= \frac{K}{j\omega_0\omega_n^2}\left(\omega_n^2 - \omega_0^2 + j\frac{\omega_n\omega_0}{Q}\right), \quad (11)$$

where $Q = \frac{1}{2\zeta}$. If we pick our desired gain, K_0, at a certain frequency, $\omega_0 = 2\pi f_0$, then we get

$$K_0 = \left|\frac{N(j\omega_0)}{j\omega_0}\right| \quad (12)$$

$$= \frac{K}{\omega_0\omega_n^2}\sqrt{(\omega_n^2 - \omega_0^2)^2 + 4\zeta^2\omega_n^2\omega_0^2} \quad (13)$$

$$= \frac{K}{\omega_0\omega_n^2}\sqrt{(\omega_n^2 - \omega_0^2)^2 + \left(\frac{\omega_n\omega_0}{Q}\right)^2}. \quad (14)$$

This can be solved for K via

$$K = \frac{K_0\omega_0\omega_n^2}{\sqrt{(\omega_n^2 - \omega_0^2)^2 + 4\zeta^2\omega_n^2\omega_0^2}} \quad (15)$$

$$= \frac{K_0\omega_0\omega_n^2}{\sqrt{(\omega_n^2 - \omega_0^2)^2 + \left(\frac{\omega_n\omega_0}{Q}\right)^2}}. \quad (16)$$

Another option is to pick the crossover frequency of the idealized open loop model. Using a method described in [5] setting

$$K = \frac{\omega_C}{K_P}. \quad (17)$$

will result in an open loop crossover frequency that is close to ω_C, especially if ω_C is chosen to be far away from the the resonant/notch frequency. Using (15), (16), or (17) to pick K allows us to equate terms in (8) to give

$$K_D T = \frac{K}{\omega_n^2}, \quad (18)$$

$$\omega_n^2 = \frac{K_I}{K_D T^2} \quad \text{and} \quad (19)$$

$$2\zeta\omega_n = \frac{\omega_n}{Q} = \frac{K_P}{K_D T}. \quad (20)$$

We can solve for ω_n, ζ, and Q:

$$\omega_n = \frac{1}{T}\sqrt{\frac{K_I}{K_D}}, \quad (21)$$

$$\zeta = \frac{K_P}{2\sqrt{K_I K_D}}, \quad \text{and} \quad (22)$$

$$Q = \frac{1}{2\zeta} = \frac{\sqrt{K_I K_D}}{K_P}. \quad (23)$$

However, for design, we might want to specify ω_n and ζ or Q and then re-derive the PID gains as a function of those parameters. This should give us a better way of picking K_P, K_I, and K_D, if we know which center frequency and damping we want the controller numerator to achieve.

Let's set ω_n as a known quantity. We also know T, which is our integration and differentiation time, but will – when we discretize things – also be our sample time. Finally, we set Q (which is equivalent to setting ζ).

From (19) we have

$$\frac{K_I}{K_D} = (\omega_n T)^2, \quad (24)$$

and from (20)

$$\frac{K_P}{K_D} = \frac{\omega_n T}{Q} = 2\zeta\omega_n T. \quad (25)$$

If we now let K_D be our overall controller gain, scaling K_D means scaling K_P and K_I in the same proportions to maintain the desired shape of the compensator.

In summary let:

$$K = \frac{K_0\omega_0\omega_n^2}{\sqrt{(\omega_n^2-\omega_0^2)^2+4\zeta^2\omega_n^2\omega_0^2}} \quad (26)$$

$$= \frac{K_0\omega_0\omega_n^2}{\sqrt{(\omega_n^2-\omega_0^2)^2+\left(\frac{\omega_n\omega_0}{Q}\right)^2}}, \quad (27)$$

$$K_D = \frac{K}{T\omega_n^2}. \text{ Then} \quad (28)$$

$$K_I = KT \text{ and} \quad (29)$$

$$K_P = \frac{K}{Q\omega_n}, \quad (30)$$

where $\omega_n = 2\pi f_n$ is the frequency of the notch and $\omega_0 = 2\pi f_0$ is the frequency at which the desired gain, K_0 is matched.

It is worth noting that this particular analog PID controller is not a real physically realizable device since it is impossible to design a true analog differentiator that works over all frequencies. More likely is the case that the analog differentiator is implemented as a high pass filter, in which the response at low frequencies looks like a differentiator, but the high frequency response is either flat or rolls off. Generally, it is assumed that the frequency at which the response rolls off is high enough to be ignored from the perspective of the control design. The other likely scenario is that this PID controller will actually be implemented using some sort of discrete-time system (microprocessor, DSP, or FPGA). In this case, a discrete equivalent of this controller is practical. However, the differentiator once again limits what we can do. We will see that an unfiltered differentiator can be implemented using a backward rectangular rule but not the trapezoidal rule.

V. Curve Fit to a Second Order Model

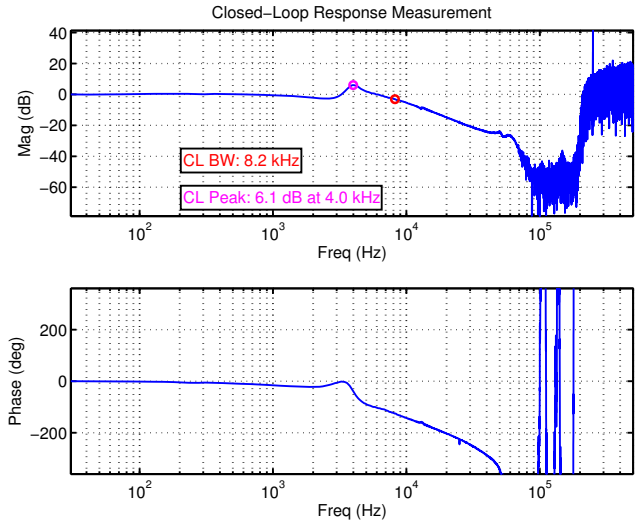

Fig. 3. Poorly tuned closed-loop response of Z actuator loop for an AFM.

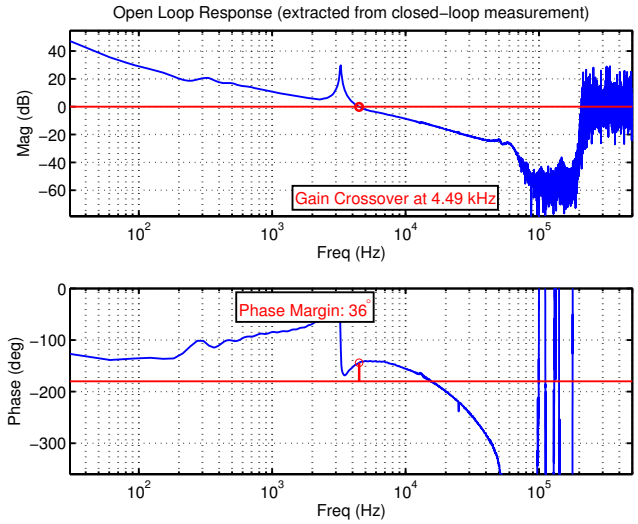

Fig. 4. Extracted open-loop response from measurement of Figure 3.

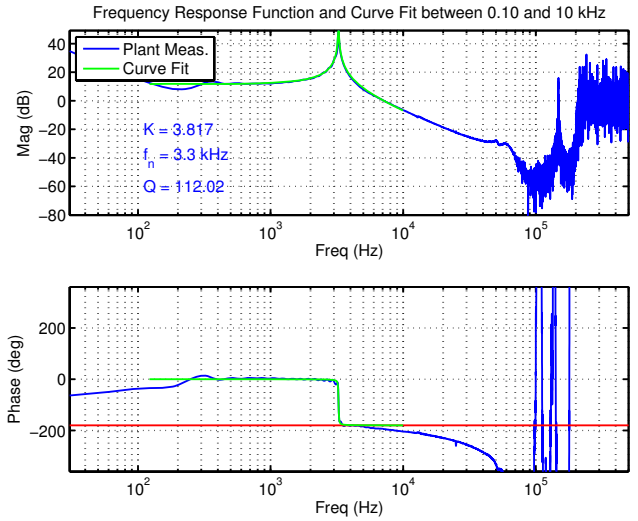

Fig. 5. Plant response and 2nd order curve fit model. From the curve fit, the parameters of a simple resonance can be immediately extracted: $K = 2.817$, $f_n = 3.3$ kHz, and $Q = \frac{1}{2\zeta} = 112.02$. Note that the resonance around 150 kHz is due to the cantilever. It is notched with a separate notch that is not part of the PID design in this paper.

Consider the frequency response function shown in Figure 3. This response is of a closed-loop measurement on an AFM Z-actuator loop. This closed-loop frequency response function (FRF) of the complimentary sensitivity function, T_{cl}, was converted to an open-loop response, PC, shown in Figure 4. That is, for every frequency measurement point, $j\omega_i$ if the closed-loop FRF is modeled as

$$T_{cl}(j\omega_i) = \frac{P(j\omega_i)C(j\omega_i)}{1+P(j\omega_i)C(j\omega_i)} \quad (31)$$

then

$$P(j\omega_i)C(j\omega_i) = \frac{T_{cl}(j\omega_i)}{1-T_{cl}(j\omega_i)}. \quad (32)$$

From here, a measurement of of the plant FRF, $P(j\omega_i)$, can be extracted by dividing $P(j\omega_i)C(j\omega_i)$ by the measured controller FRF, $C(j\omega_i)$. This is in Figure 5.

What is clear from the response in Figure 5, is that at least over a reasonable frequency range, the response is reasonably modeled by a second order simple resonance of the form

$$\hat{P}(s) = \frac{K_P\omega_n^2}{s^2 + 2\zeta_n\omega_n s + \omega_n^2} = \frac{K_P\omega_n^2}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2}. \quad (33)$$

To reduce this FRF to a transfer function of the form in (33), we need to use a curve fitter, as described in [8], [9]. However, while these algorithms typically are used to fit transfer functions of unknown order, we can use a much more simplified form to fit our second order model. If we have a frequency vector, $[j\omega_0 j\omega_1 \dots j\omega_N]^T$ and a complex response matrix, $H(j\omega) = diag(H(j\omega_0), H(j\omega_1), \dots, H(j\omega_N))$ then we can create the matrix equation

$$H(j\omega)XA = B \quad (34)$$

where $A = [A_0, A_1, A_2]^T$, $B = [1, 1, \dots 1]^T$, and

$$X = \begin{bmatrix} \omega_0^2 & j\omega_0 & 1 \\ \omega_1^2 & j\omega_1 & 1 \\ \vdots & \vdots & \vdots \\ \omega_N^2 & j\omega_N & 1 \end{bmatrix}. \quad (35)$$

Here the A_i and B_i elements represent unnormalized coefficients of the fit polynomials. By breaking up (34) into it's real and imaginary components, the curve fit can be done in Matlab or any other programming language. Once we have

$$\hat{P}_{fit}(s) = \frac{1}{A_0 s^2 + A_1 s + A_2} = \frac{b}{s^2 + a_1 s + a_2}, \quad (36)$$

where $b = \frac{1}{A_0}$, $a_1 = \frac{A_1}{A_0}$, and $a_2 = \frac{A_2}{A_0}$.

We can finally match coefficients between (33) and (36) to yield

$$\omega_n = \sqrt{a_2}, \quad (37)$$

$$Q = \frac{\omega_n}{a_2}, \text{ and} \quad (38)$$

$$K_P = \frac{b}{\omega_n^2}. \quad (39)$$

The resulting fit for our example is shown in Figure 5.

VI. Dynamic Inverse PID

The principle of dynamic inverse is to apply a controller which cancels substantial dynamics of the plant in question, replacing them with more desirable ones [10], [11]. It should be pretty clear from the previous sections that if one equates the fit ω_n, Q, and K of (37)–(39) to those of the notch filter in (21)–(23) one can then extract the PID parameters from (28–30).

Note that we do not have to match the notch filter gain, K, and quality factor, Q, exactly to the identified ones. It is entirely reasonable to scale down these values by some factor to add some robustness to the design.

Finally, if this design is chosen, the open loop model will largely be dominated by the integrator at any frequency away from the resonance/notch pair. Such a design should have excellent gain and phase margin. Higher frequency resonances, such as the one shown around 150 kHz in Figure 5 can be dealt with using a simple notch whose design is largely independent of the PID design.

Of course, the PID controller will still need to be implemented, probably using a discrete equivalent. As explained in [5], a trapezoidal rule equivalent without any low pass filter on the derivative section will lead to a PID controller with a pole at $z = -1$ and therefore an internal oscillation. In this paper, we have chosen an ideal PID without filtering, and so we will generate a discrete PID from a backward rule equivalent in Section VII.

VII. Backward Rectangular Rule Discrete PID

We can generate a backward rule discrete PID using explicit or implicit integration and differentiation times. However, leaving the T_I and T_D terms in the equation give the designer some flexibility *and* also allow these terms to drop out when the discrete-time PID is generated. In particular, for the backward rule equivalent of an ideal PID controller with the sample period, $T_S = T_I = T_D$ the time terms drop out of the equation, making it appear much simpler. This equating of sample, integration, and differentiation time is very common in the literature, although it is not often stated as such.

Applying the backward rectangular rule [6] to (3) for $C(z) = \frac{U(z)}{E(z)}$ yields

$$C(z) = K_P + \frac{K_I T_S z}{T_I(z-1)} + K_D T_D \frac{z-1}{T_S z}. \quad (40)$$

Setting $T_S = T_I = T_D$ we get

$$C(z) = K_P + K_I \frac{z}{z-1} + K_D \frac{z-1}{z}. \quad (41)$$

In terms of z^{-1} this is

$$C(z^{-1}) = K_P + K_I \frac{1}{1-z^{-1}} + K_D(1 - z^{-1}). \quad (42)$$

Equation (42) is useful for generating the time domain difference equation in 3 separate units, proportional, integral, and derivative. It is the equation from which the PID can be programmed, since it would make it easy to add anti-windup to just the integral portion. However, it is somewhat hard to work with in the z domain. We can rewrite (41), though, as:

$$C(z) = \frac{K_P(z-1)(z) + K_I z^2 + K_D(z-1)^2}{z(z-1)} \quad (43)$$

or

$$C(z) = \frac{(K_P + K_I + K_D)z^2 - (2K_D + K_P)z + K_D}{z^2 - z} \quad (44)$$

Using (44), we can examine the discrete-time properties of the linear model of this PID in Matlab.

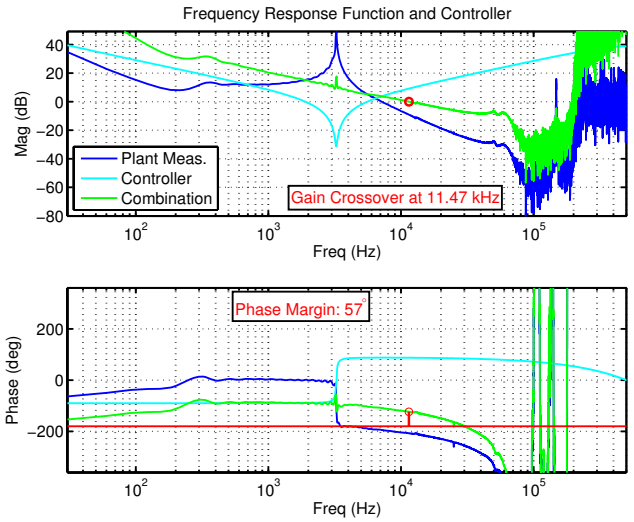

Fig. 6. PID controller generated from resonance/notch parameters and new open-loop response. Notice the improved crossover frequency and higher phase margin, compared with the response in Figure 4.

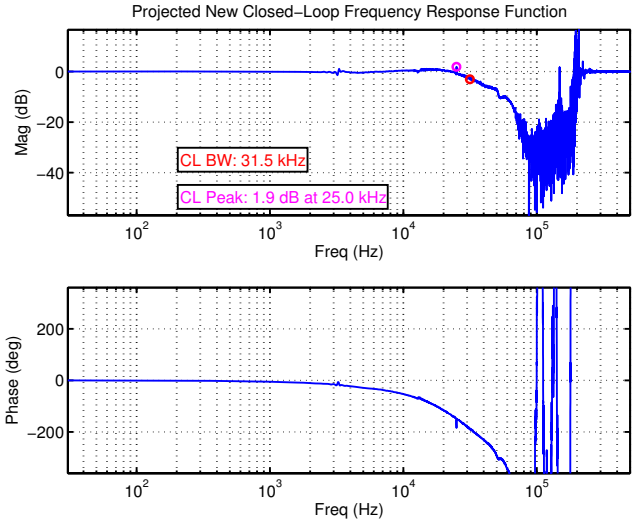

Fig. 7. Projected closed-loop response of system with new controller. Notice the improved bandwidth and lower peaking, compared with the response in Figure 3.

VIII. Design Example

An example of this method, applied to the plant response of Figure 5 is shown in Figures 5, 6, and 7. Figure 5 shows a curve fit as described in Section V. Note that the curve fit is limited to measurements in frequencies of interest, in this case between 100 Hz and 10 kHz. Below 100 Hz, the open-loop plant FRF is just not accurate enough. At frequencies significantly above 10 kHz, we can see the effects of other dynamics which will not be addressed by the PID. Finally, at frequencies significantly higher than 150 kHz, the FRF completely breaks down. So, by avoiding these regions, we can extract a very reasonable second order model from the curve fit.

Applying the dynamic inverse theme of using the identified model to drive the notch design and therefore the PID design, we arrive at the PID controller shown in Figure 6. Note that the FRF for this controller is that of the backward rule discrete equivalent (as can be seen from the phase roll off at higher frequency). The combined open-loop response exhibits very clean behavior. The open-loop crossover frequency and phase margin are considerably improved: 11.47 kHz and 57° in Figure 6 as compared with 4.49 kHz and 36° in Figure 4. Likewise, the closed-loop bandwidth and peaking are considerably better: 31.5 kHz and 1.9 dB peaking in Figure 6 as compared with 8.2 kHz and 6.1 dB peaking in Figure in Figure 3.

IX. Conclusions

This paper has shown a new PID tuning method that can be applied to AFM actuators. The method produces a PID which can invert the dynamics of a main resonance of an AFM actuator, producing a new loop shape which can have a crossover frequency above the resonant frequency. The use of this methodology depends upon an understanding of certain useful relationships between linear PID controllers and second order sections [5]. While this paper has shown this method being applied a Z-actuator, it is also applicable to X and Y actuators, especially when their responses are significantly determined by a single main resonance [12].

References

[1] D. Y. Abramovitch, S. B. Andersson, L. Y. Pao, and G. Schitter, "A tutorial on the mechanisms, dynamics, and control of atomic force microscopes," in *Proceedings of the 2007 American Control Conference*, (New York, NY), pp. 3488–3502, AACC, IEEE, July 11–13 2007.

[2] D. Croft, G. Shed, and S. Devasia, "Creep, hysteresis, and vibration compensation for piezoactuators: Atomic force microscopy application," *ASME J. Dyn., Sys., Meas., & Ctrl.*, vol. 128, no. 35, pp. 35–43, 2001.

[3] T. Ando, T. Kodera, E. Takai, D. Maruyama, K. Saito, and A. Toda, "A high-speed atomic force microscope for studying biological macromolecules," *PNAS*, vol. 98, pp. 12468–12472, 2001.

[4] G. Schitter, K. J. Åström, B. DeMartini, G. E. Fantner, K. Turner, P. J. Thurner, and P. K. Hansma, "Design and modeling of a high-speed scanner for atomic force microscopy," in *Proc. Amer. Ctrl. Conf.*, (Minneapolis, MN), pp. 502–507, June 2006.

[5] D. Y. Abramovitch, "Some useful relationships between PID controllers and second order sections," in *preparation*, 2008.

[6] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*. Menlo Park, California: Addison Wesley Longman, third ed., 1998.

[7] K. J. Åström and T. Hägglund, *Advanced PID Control*. Oxford Series on Optical and Imaging Sciences, ISA Press, August 15 2005.

[8] J. L. Adcock, "Curve fitter for pole-zero analysis," *Hewlett-Packard Journal*, vol. 38, pp. 33–37, January 1987.

[9] R. L. Dailey and M. S. Lukich, "MIMO transfer function curve fitting using chebyshev polynomials." Presented at the SIAM 35th Anniversary Meeting, Denver, CO, October 1987.

[10] M. Tomizuka, "Zero phase error tracking algorithm for digital control," *ASME Journal of Dynamic Systems, Measurement, and Control*, March 1987.

[11] Q. Zou and S. Devasia, "Preview-based stable-inversion for output tracking: Application to scanning tunneling microscopy," *IEEE Transactions on Control Systems Technology*, vol. 12, pp. 375–386, May 2004.

[12] L. Y. Pao, J. A. Butterworth, and D. Y. Abramovitch, "Combined feedforward/feedback control of atomic force microscopes," in *Proceedings of the 2007 American Control Conference*, (New York, NY), pp. 3509–3515, AACC, IEEE, July 11–13 2007.