Understanding a Universal Testing Machine Motor Control System

An Experimental PID Gain Tuning Procedure

By: Richard Gedney, ADMET, Inc.

1.0 PID Controller Basics

The mechanical properties of a material are determined by using a testing machine to push, pull or twist a sample of the material. Many materials are strain rate sensitive which means their properties vary with test speed. A valid comparison of mechanical properties between suppliers can only be achieved if the same test speed is used by all.

The American Society of Testing and Materials (ASTM) and the International Standard Organization (ISO) are two organizations that govern mechanical test specifications. Each specification requires that the force be applied at a specific strain, crosshead position or stress rate. One function of the testing machine controller is to ensure that the specified test rate is accurately maintained throughout the test. Universal testing machines are electromechanically or hydraulically actuated. Here, we will consider an electromechanical testing machine that uses a motor control system for force actuation. ADMET's MTESTQuattro(R) Materials Testing System controller is part of the motor control system and is responsible for regulating the speed of the motor. A schematic diagram of the motor control system is shown in Figure 1.

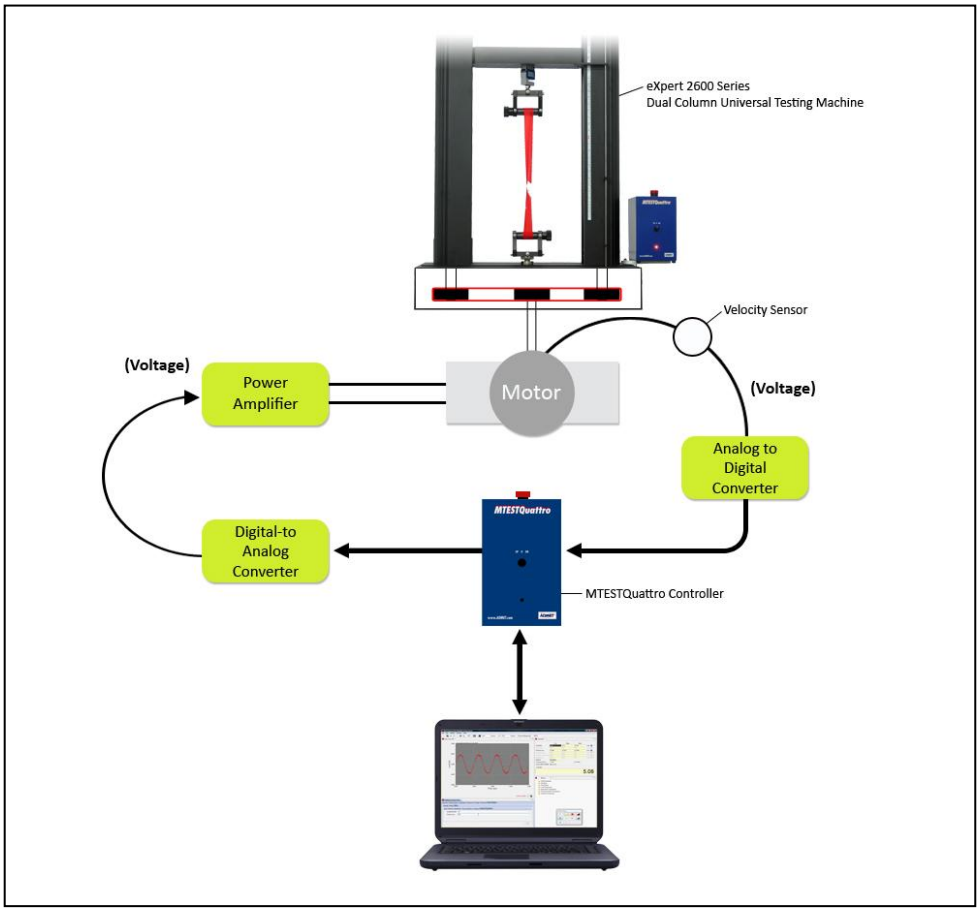


Figure 1 – Electromechanical universal testing machine motor control system.

In Figure 1, the test rate is proportional to the motor speed. Control of motor speed is accomplished by increasing the voltage to the power amplifier if the test speed is too low or decreasing it if the test speed is too high. A simple rule for regulating motor speed is to make a change in the power amplifier voltage proportional to the test speed error (difference between the actual and desired test speed).

Test Speed Error = Desired Test Speed – Actual Test Speed Eq. 1

Amplifier Voltage = Kp x Test Speed Error Eq. 2

Equation 2 is a proportional control algorithm. Kp is the proportional gain and is adjusted to minimize the test speed error.

From the point of view of the motor control system, how well the test speed is controlled will depend on how much demand is placed on the MTESTQuattro controller. If the analog to digital (A/D) and digital to analog (D/A) converters as part of the controller require little intervention; if the MTESTQuattro control algorithm used for computing the power amplifier voltage as a function of measured test speed can be updated quickly; then it is reasonable to expect minimal error between the actual and desired test speeds. Since the calculation of the power amplifier output voltage does not depend on time (see Eq. 2), the strategy for the MTESTQuattro controller is to update the power amplifier voltage as frequently as possible. We define the servo update rate as the time interval between each amplifier voltage computation. The servo update rates for our examples are fixed at 1 millisecond or 1,000 times per second (1,000 Hz).

To relate the servo update rate to actual testing applications, several examples follow.

Example 1: ASTM D638 Standard Test Method for Tensile Properties of Plastics specifies a constant crosshead testing speed of 2 in/min or 0.033 in/sec. With a servo update rate of 1,000 times per second, the desired crosshead movement per servo update = 0.033/1000 = 0.000033 inches/servo update.

Example 2: Apply a 10 lbf peak to peak sinewave force amplitude at 5 Hz (cycles per second) to a test sample. Based on the 5 Hz specification, each cycle is to be completed every 0.2 seconds (1/5 Hz). The number of servo updates per cycle = 1000 x 0.2 = 200 servo updates per cycle. During each cycle the actuator will apply 10 lbf then remove 10 lbf from the sample for a total force traversal of 20 lbf. The average change in force per servo update is 20 lbf / 200 = 0.1 lbf/servo update.

Example 3: Apply a 10 lbf peak to peak sinewave force amplitude at 50 Hz to a test sample. Based on the 50 Hz specification, each cycle is to be completed every 0.02 seconds (1/50 Hz). The number of servo updates per cycle = 1000 x 0.02 = 20 servo updates per cycle. The average change in force per servo update is 20 lbf / 20 = 1 lbf/servo update.

In Example 1, the test is performed at a constant displacement rate of 0.000033 inches/servo update. Because the desired rate does not vary during the entire test and the control algorithm updates the amplifier voltage 1,000 times per second, the motor control system is capable of precisely following the desired test speed. On average there is a 0.1 lbf and 1 lbf change in force per servo update in Examples 2 and 3, respectively. However, the motor during the sinewave profile is continuously accelerating and decelerating producing a varying test speed error. As we increase the cycling frequency, accelerations get larger and there are fewer servo updates each cycle. Therefore, the servo update errors will grow with increasing frequency which will demand more from the controller to achieve accurate control.

1.1 A Simple Explanation of a Universal Testing Machine Feedback Control System

Figure 2 is a block diagram of our Example 1 testing application under closed loop feedback control. In Example 1, the test rate is based on crosshead position and is specified as 0.033 in/sec or 0.000033 in/servo update. What is happening during the test? At each servo update, the MTESTQuattro controller subtracts the actual crosshead position from the desired crosshead position to obtain the crosshead position error. If the error for that servo update is zero, the power amplifier voltage will be zero. If there is an error, corrective action will occur. The motor will be told to speed up if the actual position lags the desired position (positive error) or told to slow down if the error is negative. As the test is progressing, the force is increasing but then the material begins to yield (a process disturbance). Suddenly there is less resistance to stretching the test specimen and the actual position gets ahead of the desired position. The controller will decrease the power amplifier voltage so that the motor slows down. After a while, the material may begin to strain harden creating more resistance to movement. The actual position falls behind and the controller then increases the power amplifier voltage to speed the motor up. Without a feedback loop, the testing machine would have no knowledge of it's actual crosshead position. Once a disturbance is encountered such as increasing load, yield, or rupture, the error between the actual and desired crosshead position would vary along with the test speed.

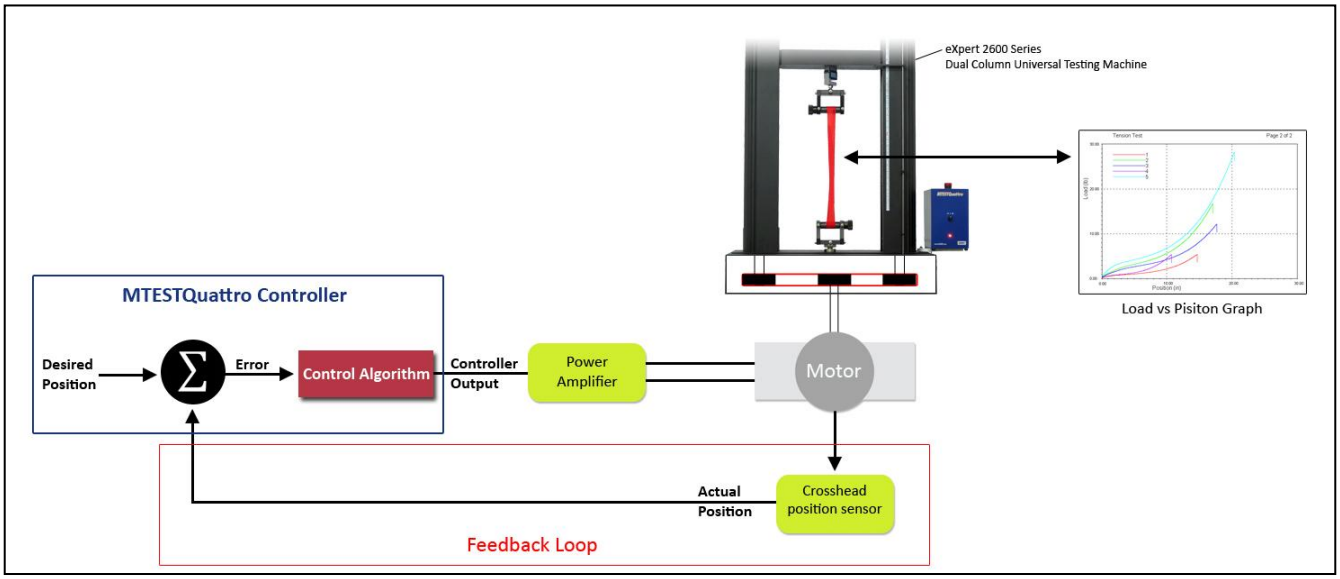


Figure 2 - Block diagram of test being performed under crosshead position rate control.

1.2 Exploring the MTESTQuattro Controller in More Detail

The MTESTQuattro controller shown in Figure 2 employs a Proportional, Integral, Derivative (PID) control algorithm. What does the PID controller actually do? It applies 1, 2 or 3 calculations to the position error each servo update. These calculations, termed modes of control are:

1. Proportional, P
2. Integral, I
3. Derivative, D

Figure 3 is a block diagram of the PID controller. There are several forms of a PID controller but the one shown in Figure 3 is a PID controller in parallel form.

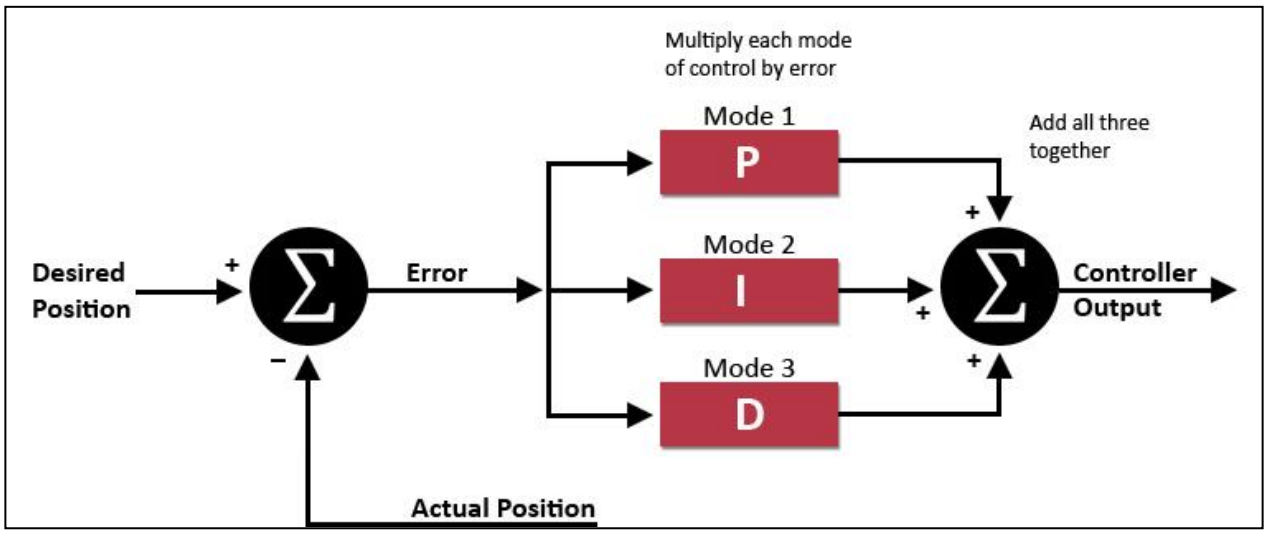


Figure 3 - Block diagram of PID controller.

The PID controller works the following way. Each servo update, the actual crosshead position is subtracted from the desired crosshead position to obtain a position error. The error is passed into one, two or all three of the P, I and D modes depending on which modes are turned on. Then the outputs from each mode are added together. The resulting sum is the controller output or power amplifier voltage which sets the speed of the motor for that servo update.

One, two or all three of the modes can be turned on. The possible combinations are listed below with the most common being PI control.

- Proportional Control Only, P
- Proportional plus Integral Control, PI
- Proportional plus Integral plus Derivative, PID
- Proportional plus Derivative, PD

Following is a description of how each one of the modes works.

1.2.1 Proportional Control (Mode 1)

Figure 4 is a block diagram of proportional only control.

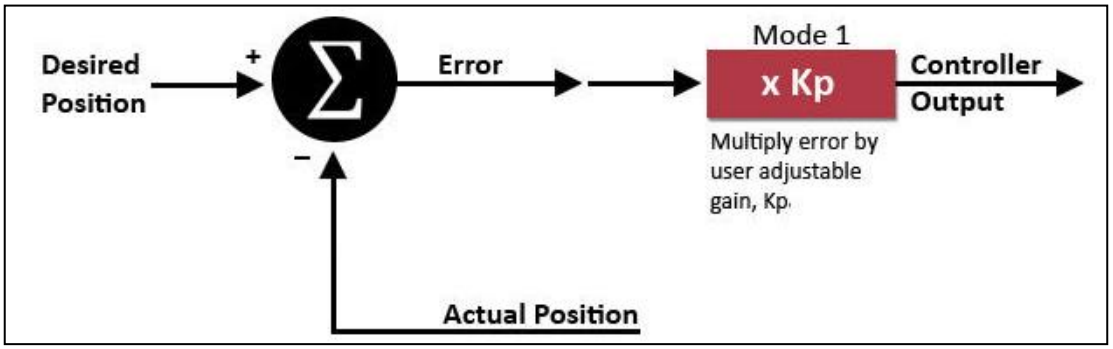


Figure 4 - Proportional Controller.

For a proportional only controller, the actual crosshead position is subtracted from the desired crosshead position to determine the error for each servo update. The error is then multiplied by the proportional gain, Kp, to produce the mode 1 controller output (see Eq. 3).

Mode 1 Proportional Controller Output = Kp x Error Eq. 3

The proportional gain, Kp, is adjustable. The optimum value for Kp is dependent on the universal testing machine motor control system and the test type. By definition, a positive error means that the motor is not moving fast (lag) enough. Increasing Kp will increase the amplifier voltage and decrease the error. Notice also that as the error gets smaller the voltage decreases and the motor slows down. If Kp is made too large, the error can go negative (actual position > desired position) which may start a process where the motor alternates between speeding up, slowing down, speeding up, slowing down... This oscillation can result is the motor control system going unstable and getting out of control. A simple way to eliminate the instability is to decrease Kp until the oscillation stops. The safest Kp will cause the actual crosshead position to constantly lag behind the desired crosshead position. During single mode P only control, the actual test speed may match the desired test speed with the actual crosshead position lagging behind the desired crosshead position creating a constant error. However, the force during the Example 1 tensile test will continue to increase until it reaches the ultimate strength of the material. The motor seeing greater resistance as the load is increasing will cause the error to increase in proportion to the load. Therefore, the test speed may vary with changing loads. The inability to drive the servo update error to zero and overcome disturbances introduced by varying loads is a deficiency in single mode proportional only control. To overcome the constant error, we add integral control.

1.2.1 Integral Control (Mode 2)

Integral control accumulates the error from each servo update then multiplies the sum by the integral control gain, Ki. Like Kp, Ki is adjustable. Eq. 4 is the algorithm for the accumulated error sum and Eq. 5 is the formula for mode 2 integral control action.

$$\text{Accumulated Error Sum} = \text{Error}_i + \text{Error}_{i+1} + \text{Error}_{i+2} + \text{Error}_{i+3} + \ldots\ldots \quad \text{Eq. 4}$$

Mode 2 Integral Control Output = Accumulated Error Sum x Ki Eq. 5

Where, i, i+1, i+2, i+3, etc. are successive servo updates in time.

By summing the errors at each servo update, there is a component of the power amplifier voltage that is always increasing as long as the error is positive. As a result, the ramp like integral control action will overcome the increasing motor torques caused by increasing loads and will work to drive the error to zero. Figure 5 is a block diagram of a PI controller and shows how the controller output/power amplifier voltage is calculated.

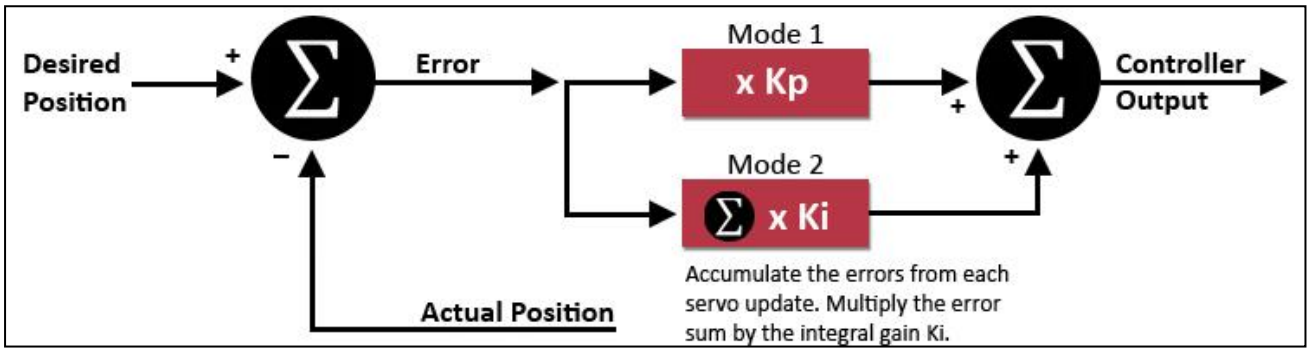


Figure 5 - Block diagram of Proportional-Integral control.

Like the proportional gain Kp, Ki is adjusted to obtain optimum control. Ki equal to zero turns off integral control (NOTE: The MTESTQuattro controller requires integral gain, Ki>0.). Increasing Ki will cause the motor control system to respond faster to errors. When Ki is made too large, it will cause the motor control system to go into oscillation. Two mode PI control is the most common form of control and in many cases is all that will be required. However, there may be times where derivative control action may improve performance.

1.2.3 Derivative Control (Mode 3)

Derivative action measures the rate of change of servo update errors and multiplies the value by the derivative gain, Kd. Eq. 6 is the formula for mode 3 derivative control action.

$$\text{Mode 3 Derivative Control Output} = Kd \times (Error_i - Error_{i-1})/\text{Servo Update Time} \qquad \text{Eq. 6}$$

Where, $Error_i$ and $Error_{i-1}$ are the errors at the current and previous servo update times, respectively. Figure 6 is a block diagram of a three mode PID controller.

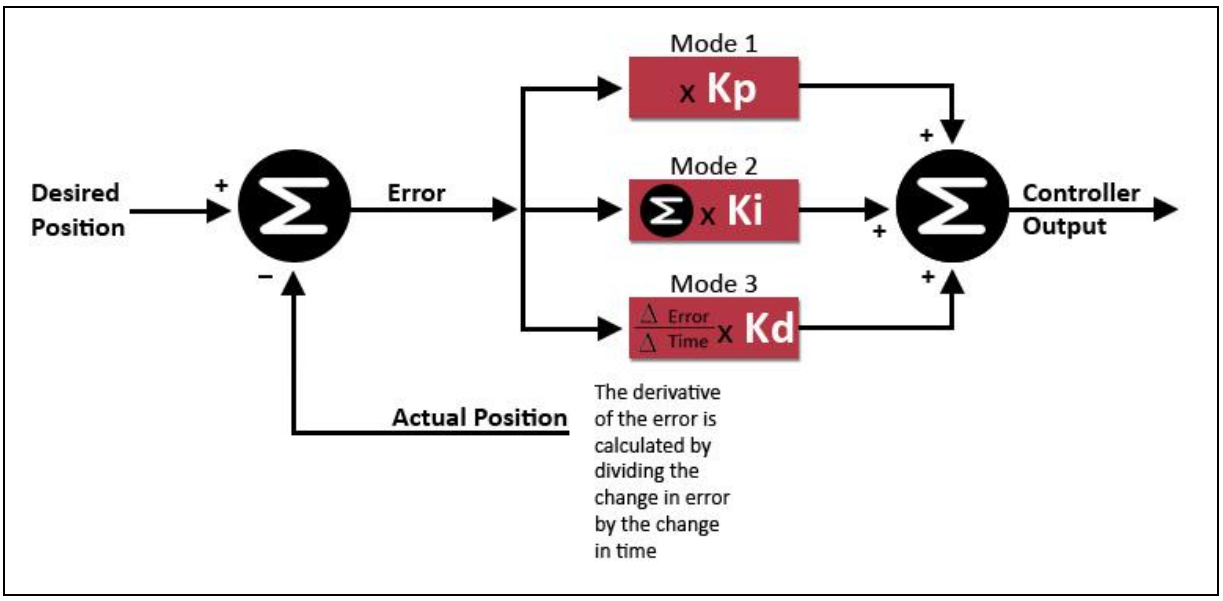


Figure 6 - Block diagram of Proportional-Integral-Derivative control.

To illustrate how derivative control works, we apply a step position movement to the motor control system. Before the step movement is executed, the desired and actual crosshead positions are zero. At the very first servo update, the desired crosshead position is set to 0.1 inches. Figure 7 is a graph of the desired and actual crosshead position over time. Table 1 contains the numerical values of both curves at particular servo updates.

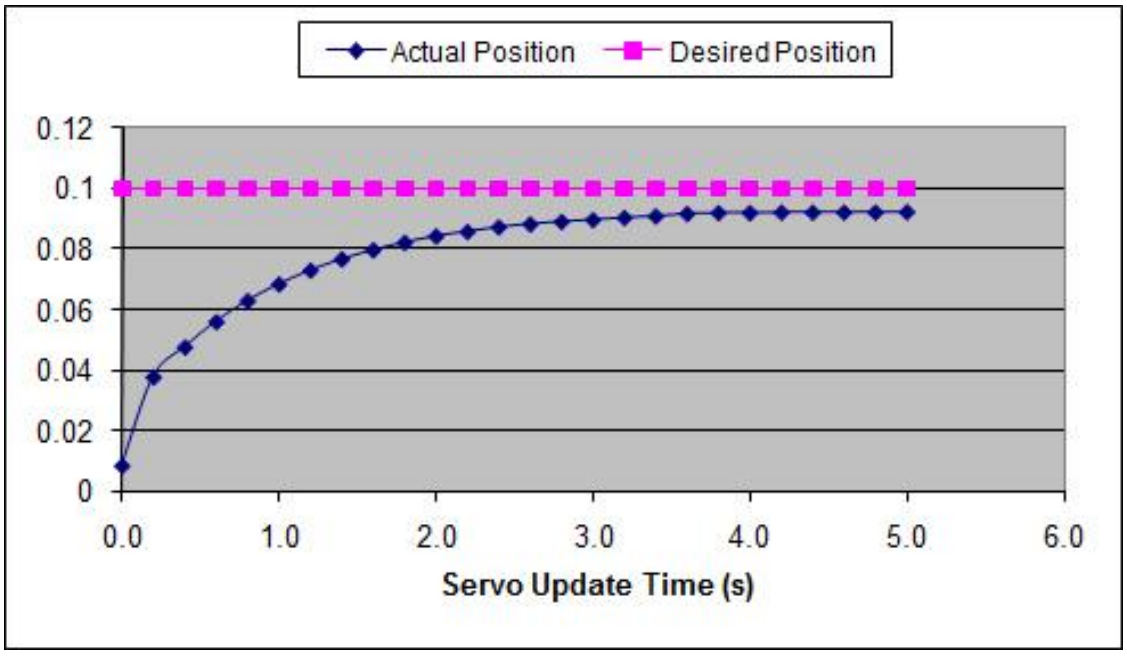


Figure 7 - Motor control system step response.

Table 1 – Step response values at particular servo update times.

Servo Update Time (s)	Actual Position	Desired Position	Error	Error Change
0.0	0.0085	0.1	0.0915	
0.2	0.0378	0.1	0.0622	-0.0293
0.4	0.0477	0.1	0.0523	-0.0099
0.6	0.0561	0.1	0.0439	-0.0084
0.8	0.0630	0.1	0.0370	-0.0069
1.0	0.0685	0.1	0.0315	-0.0055
1.2	0.0731	0.1	0.0269	-0.0046
1.4	0.0768	0.1	0.0232	-0.0037
1.6	0.0798	0.1	0.0202	-0.0030
1.8	0.0823	0.1	0.0177	-0.0025
2.0	0.0844	0.1	0.0156	-0.0021
2.2	0.0859	0.1	0.0141	-0.0015
2.4	0.0874	0.1	0.0126	-0.0015
2.6	0.0884	0.1	0.0116	-0.0010
2.8	0.0892	0.1	0.0108	-0.0008
3.0	0.0898	0.1	0.0102	-0.0006
3.2	0.0905	0.1	0.0095	-0.0007
3.4	0.0910	0.1	0.0090	-0.0005
3.6	0.0918	0.1	0.0082	-0.0008
3.8	0.0921	0.1	0.0079	-0.0003
4.0	0.0921	0.1	0.0079	0.0000
4.2	0.0923	0.1	0.0077	-0.0002
4.4	0.0923	0.1	0.0077	0.0000
4.6	0.0923	0.1	0.0077	0.0000
4.8	0.0923	0.1	0.0077	0.0000
5.0	0.0924	0.1	0.0076	-0.0001

How does mode 3 derivative control action affect the response of the motor control system? Look at the last column in Table 1, the change in error between the current and previous servo update. If the error change values are applied to Eq. 6, the negative values highlighted in red will produce a negative mode 3 control output. The resultant control output from the PID controller will then be reduced by the

mode 3 contribution; effectively dampening the overall response of the motor control system. Thus, derivative action will help maintain loop stability and minimize overshoot while larger proportional and integral gains will quicken the response of the motor control system.

1.3 A Review of PID Control

The proportional action of a PID controller produces a power amplifier voltage proportional to the current servo update error. Increasing Kp will speed up the response of the motor control system and reduce but never eliminate servo update errors. This means that in Example 1, the actual crosshead position will never equal the desired crosshead position. Furthermore, the servo update errors may also grow with increasing force.

Integral control action continuously adds the errors at subsequent servo updates together producing a ramp like change in the power amplifier voltage. This action will drive the servo update error to zero over time and also overcome the changing motor resistance caused by varying loads during a test.

Increasing Kp and Ki will speed up the motor's response to servo update errors. If Kp and Ki are too large, the system will overshoot its desired position and could eventually go unstable. Derivative control allows the control loop to have larger Kp and Ki values which produce a faster response and smaller errors while still keep the motor control system stable. This is accomplished because the negative error rate changes as shown in Table 1 have a dampening effect by reducing the resultant PID control output. A faster better performing control loop results. However, there is one caveat. Noisy feedback sensor measurements will produce incorrect error rate readings which may result in faulty mode 3 derivative control action.

2.0 An Experimental Procedure for Tuning a Testing Machine Position Control Loop

The basic operation of a testing machine motor control system has been explained above. The three modes of PID control and how they affect motor response were also introduced. Next we apply the information to tune the crosshead position control loop for optimal response. Optimal response can be defined in many ways. For a static testing application, it could mean precise speed regulation over a wide range of velocities and variations in load. For a fatigue testing application, it could mean maintaining a peak to peak amplitude at a given frequency over a period of time. In general, one set of crosshead position PID control gains on a static testing machine will produce acceptable crosshead position control over almost the entire speed range of the machine. For very slow speeds, a second set of crosshead position PID control gains with larger Kp and Ki values may be required. For fatigue testing applications, a different set of PID control gains may be required for cyclic waveforms differing in amplitude and frequency. Thus, requiring more frequent gain tuning.

A common experimental method of obtaining optimal PID gains is to perform a step movement and adjust the gains in the following set order.

Step 1: P only control. Disable integral and derivative action by setting Ki = 1 and Kd = 0, respectively. Adjust the proportional gain, Kp to achieve the best overall response for rise time, overshoot and settling time. Rise time, overshoot and settling time are defined in Figure 8.

Step 2: PI control. Disable derivative action by setting Kd = 0. Set the proportional gain, Kp to the “best” value as determined in step 1. Adjust the integral gain, Ki to achieve the “best” overall response for rise time, overshoot and settling time. For most applications, implementing PI control provides acceptable response and the tuning process could stop at the end of step 2. If derivative action is desired, proceed to step 3.

Step 3: PID Control. Set the proportional gain, Kp to the "best" value as determined in step 1. Set the integral gain, Ki to the "best" value as determined in step 2. Adjust the derivative gain, Kd to achieve the "best" overall response for rise time, overshoot and settling time.

To demonstrate the tuning process, a testing machine operated by ADMET's MTESTQuattro controller will perform a crosshead position step move of 1mm. Rise time, overshoot and settling time will be used to characterize each step response (See Figure 8). They will also be used to measure the relative performance of one set of control gains against another set. The goal of the tuning process is to adjust the PID control gains to minimize rise time, overshoot and settling time. However, decreasing the rise time may increase the overshoot and settling time. In other words, adjusting the gains to improve one parameter may make the other ones worse. In general, the tuning process produces a series of tradeoffs, requiring the engineer to choose the "best" gain values for their testing application(s).

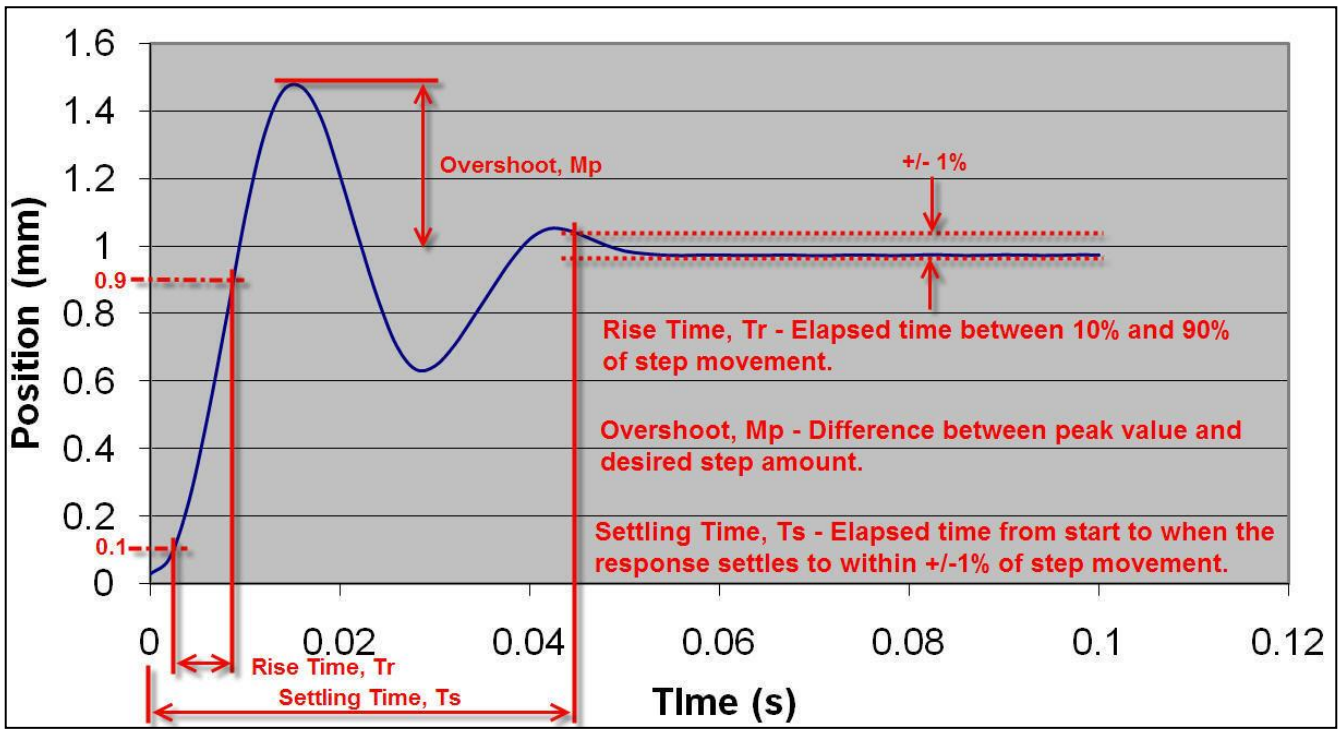


Figure 8 – A step response is characterized by rise time, overshoot and settling time.

Figure 9 depicts the MTESTQuattro Servo Control Profile Menu with a 1mm squarewave profile defined. While the motor control system is stepping back and forth, we will follow the instructions outlined in steps 1-3 above to adjust the PID gains for optimal response. In order for the gain changes to take immediate effect in MTESTQuattro, Segment Cycles must equal 1 and Options Cycle should be a number large enough to allow sufficient time to tune the PID gains. The red arrows in Figure 9 point to the Segment and Option Cycles boxes in the Servo Control Profile Menu.

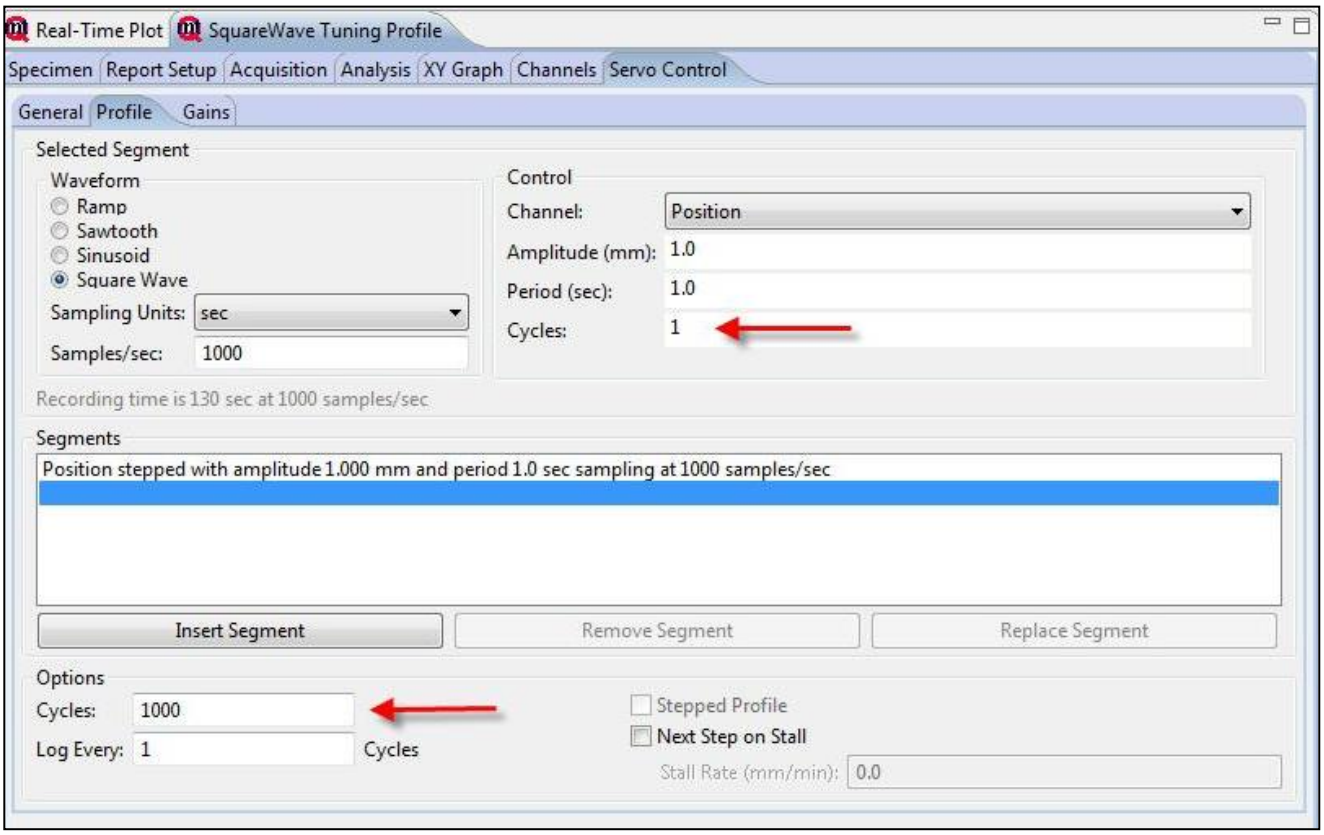


Figure 9 – MTESTQuattro Servo Control Profile Menu depicting squarewave profile.

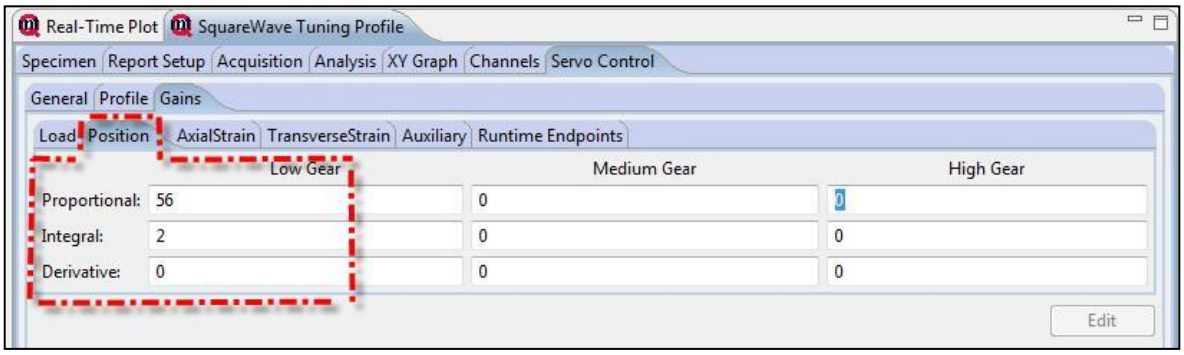


Figure 10 – Crosshead position PID gain values shown in MTESTQuattro Servo Gains Menu.

We begin step 1 of the tuning process by accessing the Servo Control Gains Menu (Figure 10) and setting the low gear position derivative gain, Kd = 0 and the integral gain, Ki = 1 (The MTESTQuattro PID control algorithm requires Ki > 0.). Press the green START icon (See Figure 12) on the Control Panel to begin the 1mm squarewave profile. When the squarewave commences, right click on the realtime position vs. time graph to pop the Context Menu as shown in Figure 11. Place the graph in SCROLL MODE then SET SCALE to adjust the X and Y axis limits for appropriate viewing of the step response. Next, click the MODIFY TUNING PARAMETERS icon from the Control Panel as shown in Figure 12 to split the live screen. The Servo Control Gains Menu should appear below the live XY graph. In the Servo Control Gains Menu, enter increasingly larger proportional gains, Kp, for the position channel (low gear) and view the step response on the live graph.

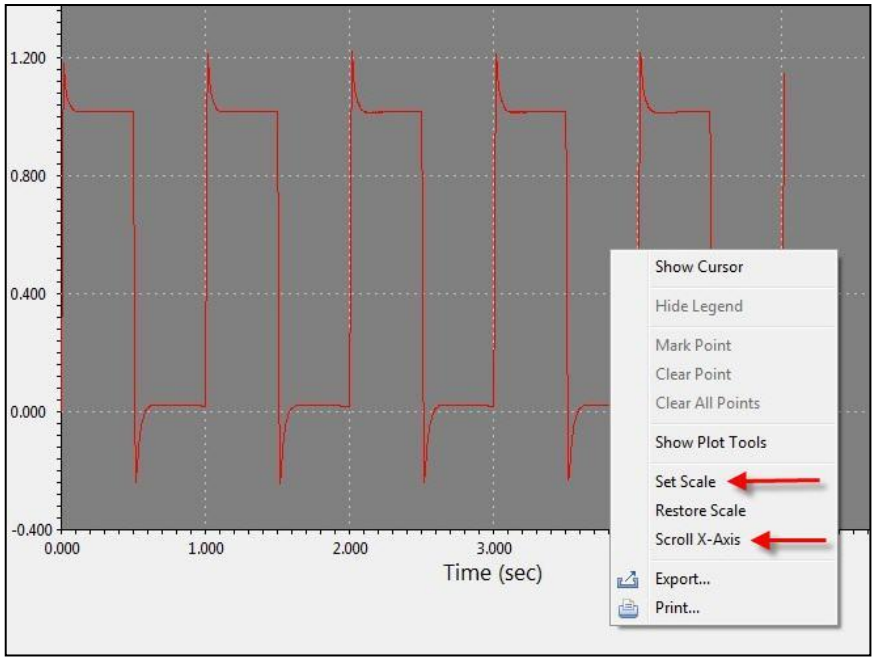


Figure 11 – Real-time graph Context Menu.

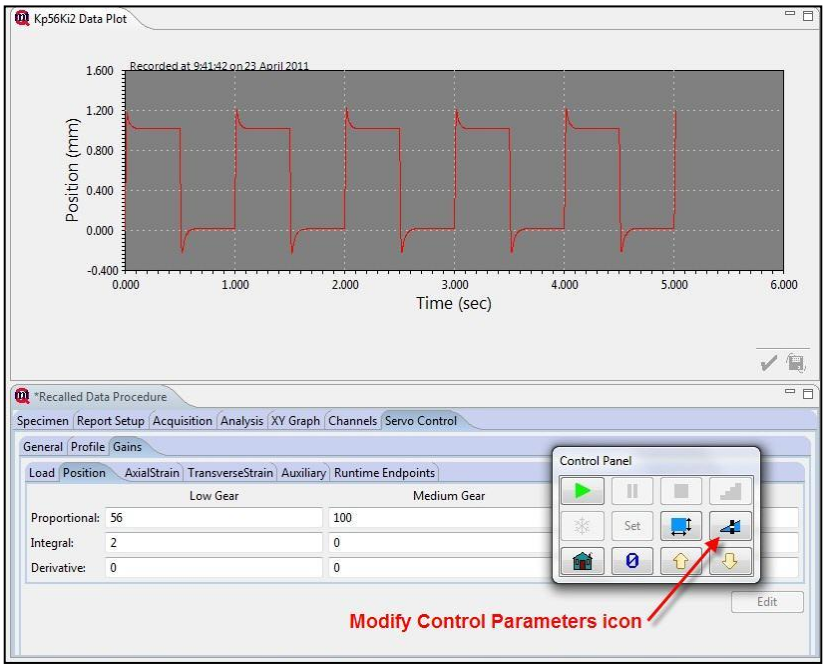


Figure 12 – MTESTQuattro realtime XY plot, Servo Profile Gain Menu and Control Panel.

As Kp increases, the rise time will decrease and the overshoot increase. As Kp is further increased, the response will begin to oscillate around the 1 mm setpoint which will extend the settling time.

Figure 13 depicts step response curves for five proportional gain values. A review of Figure 13 shows that Kp = 5 produces a sluggish response, oscillations begin at Kp = 120 and the system goes unstable at Kp = 150 (not shown). Our criteria for the "best" proportional gain, Kp is to minimize rise time, overshoot and settling time. By comparing the responses in Figure 13, Kp = 60 has a reasonably fast rise

time and minimal overshoot which appears to be a good compromise. For Kp = 60, the rise time and overshoot are 0.0089s and 0.173mm, respectively. However, the response never settles to within +/- 1% of the 1mm step value (See definition of settling time in Figure 8.). Integral control action is, therefore, required to achieve a finite settling time.

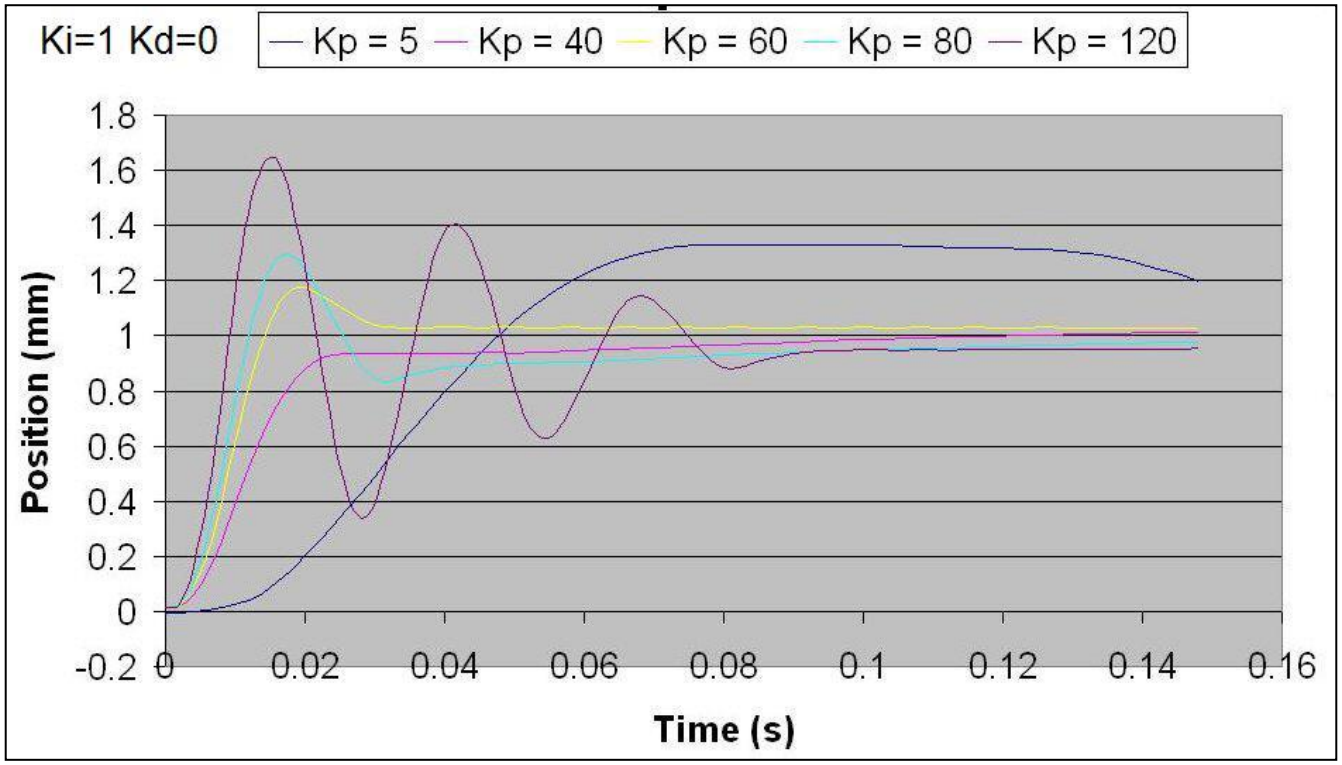


Figure 13 – Step 1 P only control step responses for five proportional gains with constant integral and derivative gains of 1 and 0, respectively.

Step 2 of the tuning process starts by fixing the proportional gain to the "best" value obtained is step 1 (Kp = 60, Kd = 0). Then the integral gain, Ki is adjusted upward. As Ki is increased, the rise time and settling time will decrease and the overshoot increase. As Ki is made larger still, the settling time will decrease then begin to increase as oscillations creep into the response (Ki = 5). Figure 14 displays step responses for five integral gains. Our criteria for adjusting Ki, is to minimize rise time, overshoot and settling time. With proportional only control, it was not possible to achieve a finite settling time. Integral control action works to drive the error to zero; making it possible to achieve a finite settling time. The rise time, overshoot and settling time for two of the "better" PI control step responses from Figure 14 are given in Table 2.

Table 2 – Rise times, overshoot and settling times for selected PI controller step responses.

Kp	**Ki**	**Rise Time (s)**	**Overshoot (mm)**	**Settling Time (s)**
60	2	0.0081	0.285	0.102
60	3	0.0074	0.454	0.078

A review of Table 2 shows that the PI controller with Ki = 2 produced a smaller overshoot and Ki = 3 produced a faster rise time and settling time. Depending on the desired response, either integral gain (Ki = 2 or 3) could be chosen as the "better" value. For most testing applications, we could stop the

tuning process and operate a two mode PI controller with Kp = 60 and Ki = 2 or 3. If further optimization is desired, derivative control action can be applied.

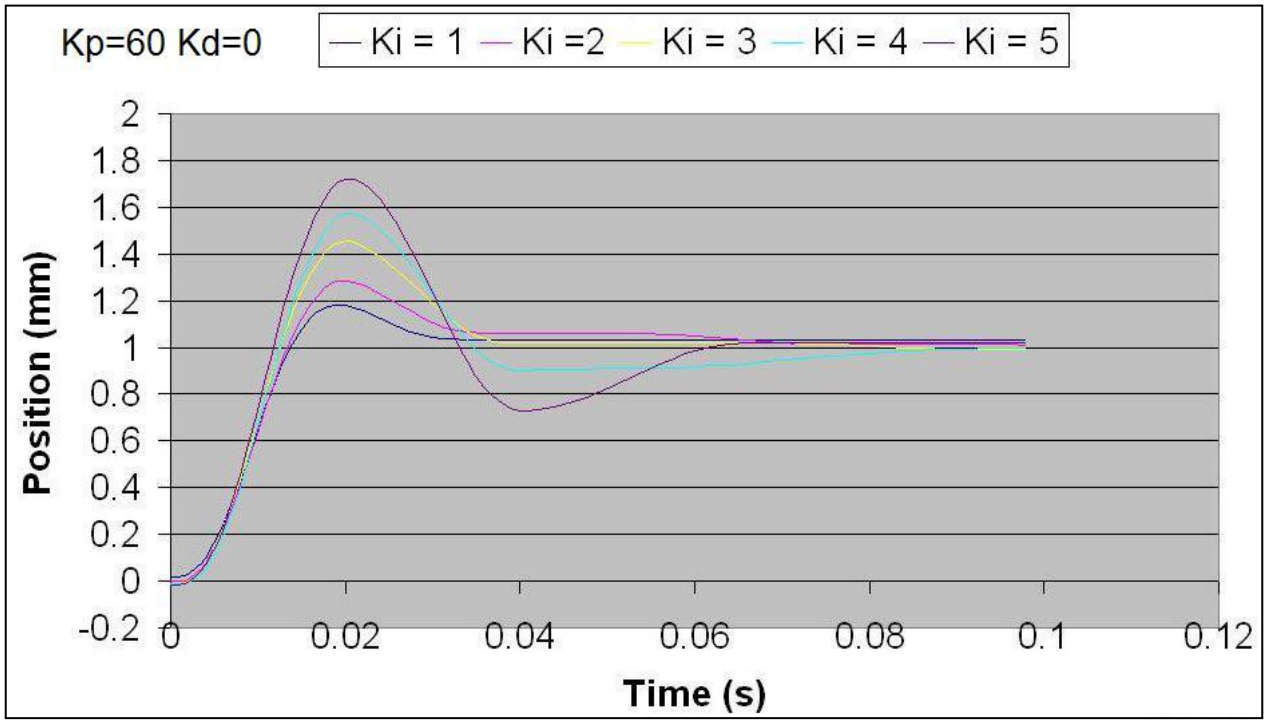


Figure 14 – PI control step responses for five integral gains with constant proportional and derivative gains of 60 and 0, respectively.

Earlier we learned that negative error rate changes occur during a step response (see Table 1), causing derivative control action to reduce the controller output resulting in a dampened response. In step 3, the dampening action of derivative control will be used to reduce the overshoot of the PI controller while at the same time trying to produce faster rise times and settling times.

Step 3 of the tuning process starts by fixing the proportional and integral gains to their "best" values as obtained in step 1 and 2 (Kp = 60, Ki = 3). Then the derivative gain, Kd is adjusted upward to reduce the overshoot. Figure 15 includes PID controller step responses for five derivative gains with fixed proportional and integral gains of 60 and 3, respectively. A review of Figure 15 and Table 3 confirms that increasing the derivative gain will reduce and if made large enough (Kd >= 200) eliminate overshoot.

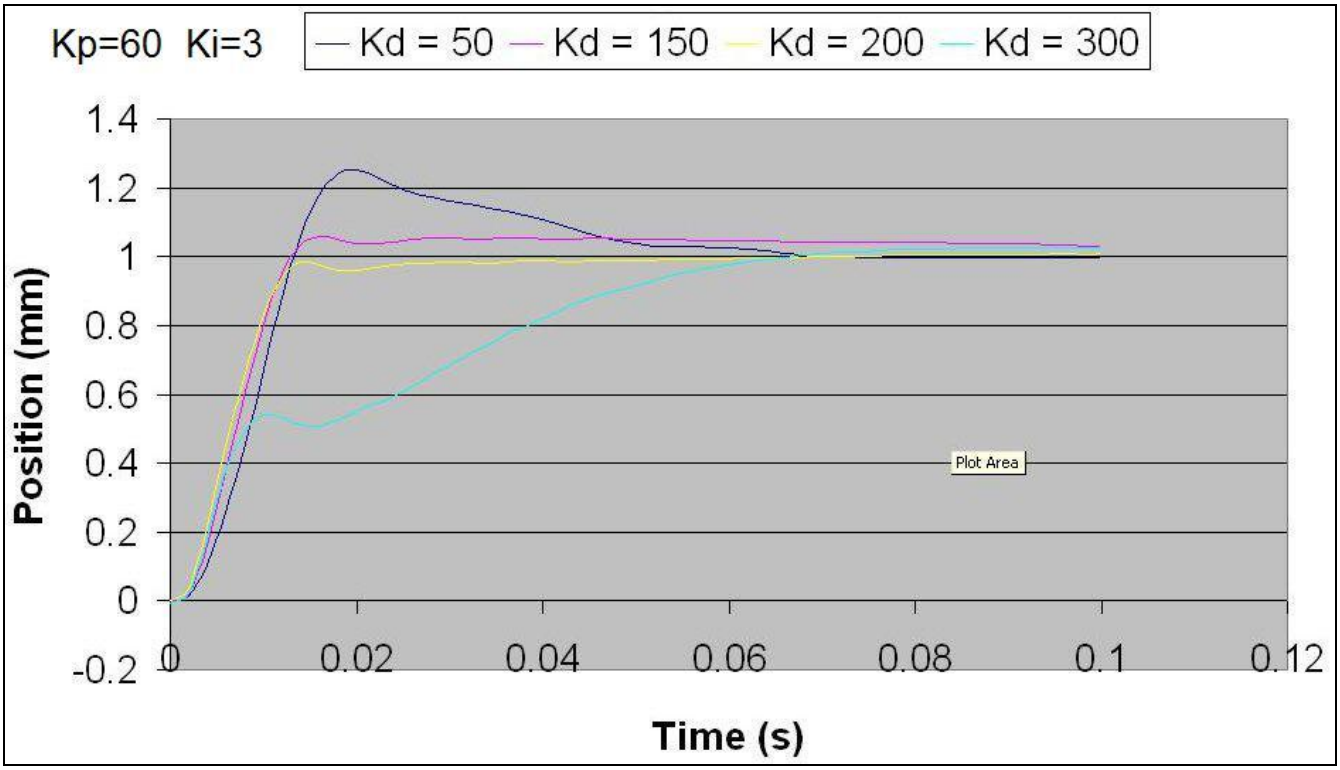


Figure 15 - PID control step responses for five derivative gains with constant proportional and integral gains of 60 and 3, respectively.

Table 3 – Rise times, overshoot and settling times from selected PID control step responses..

Kp	Ki	Kd	Rise Time (s)	Overshoot (mm)	Settling Time (s)
60	3	150	0.0082	0.053	0.106
60	3	200	0.0085	0	0.054

2.1 Summary of Experimental Gain Tuning Procedure

A three step procedure was followed to experimentally adjust the PID controller gain values to obtain the best response to a 1 mm step input. Integral control action was required to reduce the steady state error to less than +/- 1%. Integral action also decreased the rise time but increased the overshoot. Derivative control was then applied to eliminate the overshoot and decrease the settling time. Depicted in Figure 16 are step responses for the "best" P only, PI and PID controller gain values. Table 4 provides the step response rise times, overshoot and settling times for the three "best". As we can see from Figure 16, Figure 17 and Table 4, derivative control action improved the response.

Table 4 – Rise times, overshoot and settling times for “best” P only, PI and PID controller step responses.

Controller	Kp	Ki	Kd	Rise Time (s)	Overshoot (mm)	Settling Time (s)
P Control	60	1	0	0.0089	0.173	not achieved
PI Control	60	3	0	0.0074	0.454	0.078
PID Control	60	3	200	0.0085	0	0.054

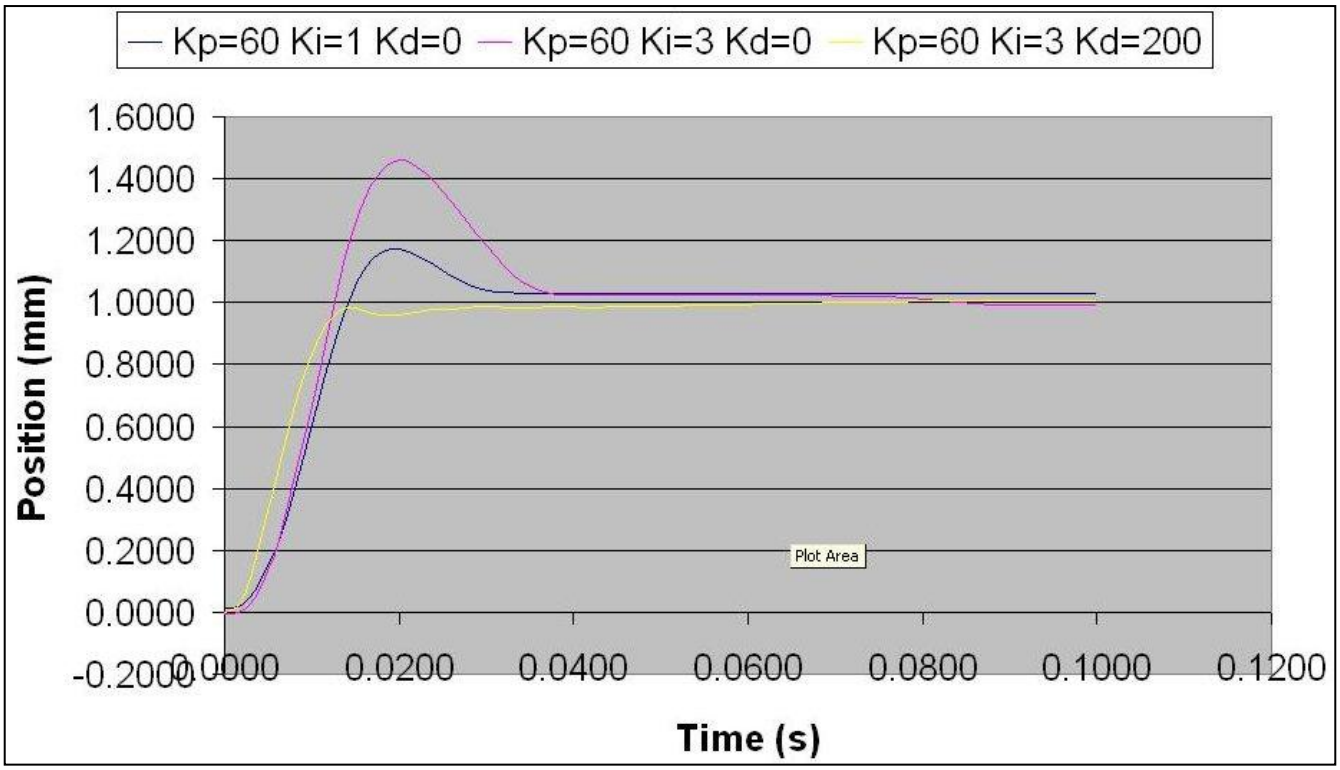
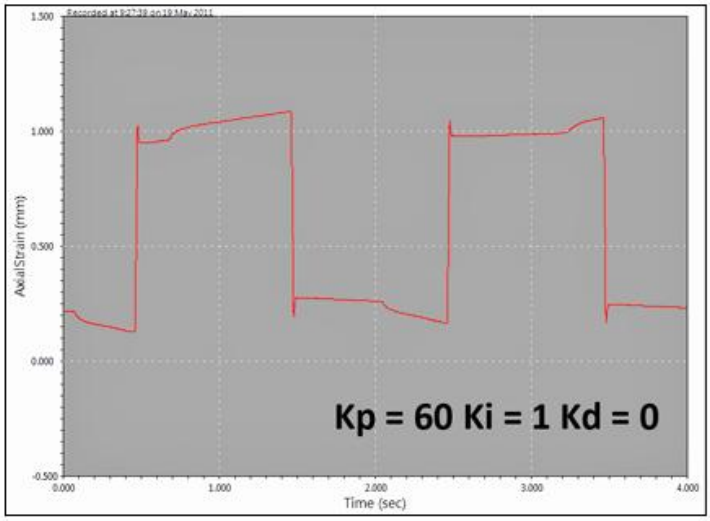
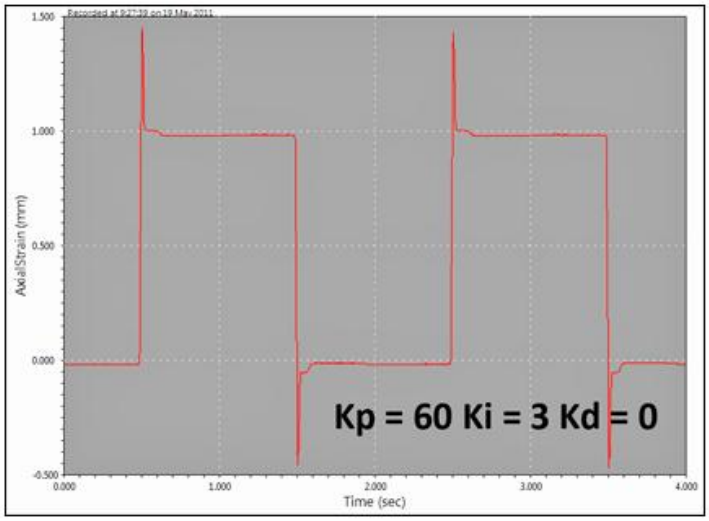


Figure 16 – Step responses for the “best” P only, PI and PID control gains.





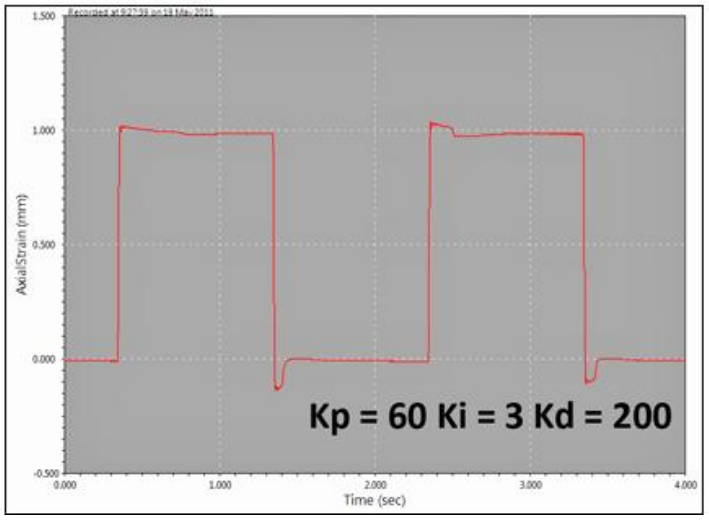


Figure 17 – Step responses for the "best" P only, PI and PID control gains from MTESTQuattro live screen.

3.0 Tuning the Position Control Loop to a Sawtooth Waveform Instead of a Step Input

In Section 2, we learned how to experimentally tune the PID control gains to a step input. Rise time, overshoot and settling time were used to characterize the dynamic response to the step input. Tuning PID controller gains to a step input is a more appropriate method for optimizing the response of a dynamic testing machine used for fatigue testing. However, most universal testing machines are designed for static testing applications where a sample is pulled at a constant rate until failure. For static testing applications, it is better to tune the PID gains to a sawtooth waveform instead of step input at relatively slow ramp rates. The methods learned in Section 2 for experimentally tuning the PID gains to a step input apply to the sawtooth waveform. Figure 18 depicts the Servo Profile Menu with the sawtooth waveform defined.

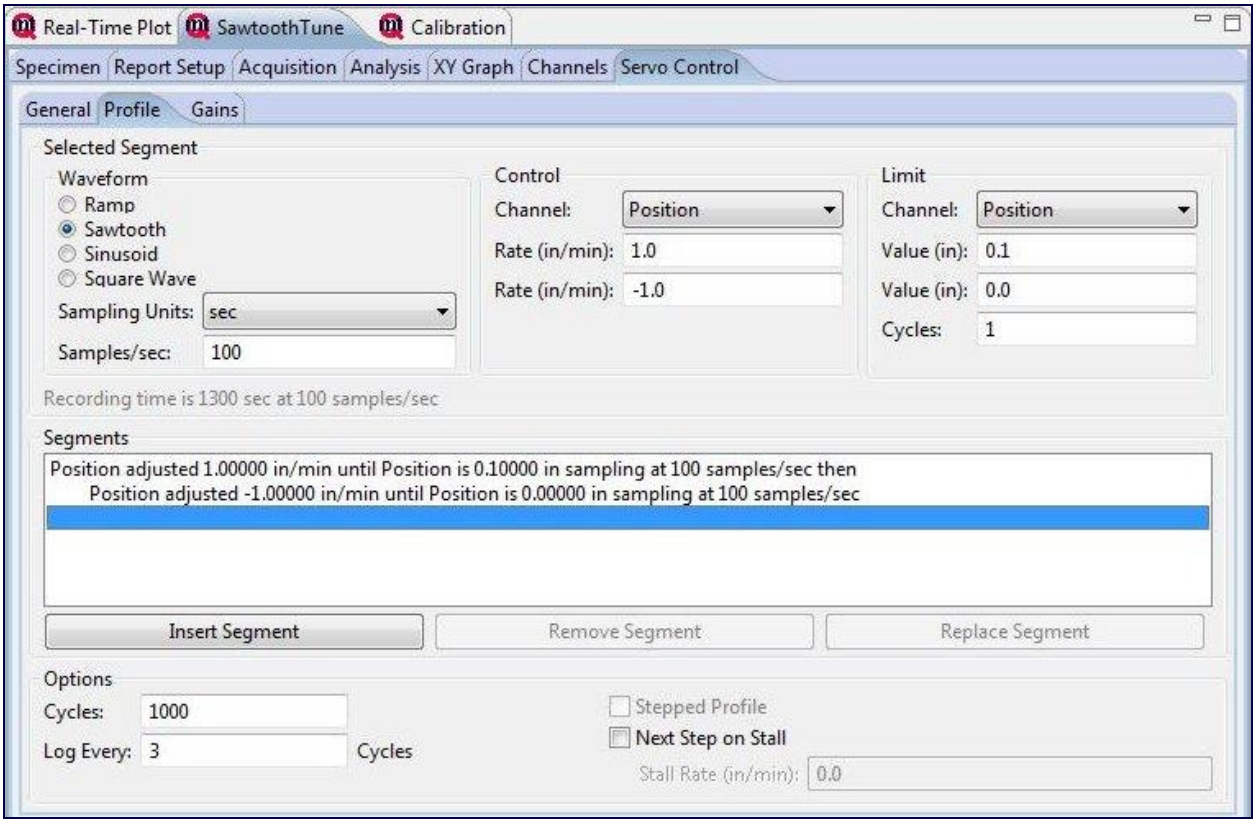


Figure 18 – Servo Profile Menu with sawtooth waveform specified.

Here, the crosshead position rate is set at 1 in/min to 0.1 in then reverse direction and travel a –1 in/min back to 0 in. Most static testing machines are capable of traveling at 20 to 40 in/min. The 1 in/min rate is chosen because it is between test rates required for many static pull tests (Metal test rates are slower, elastomer test rates are faster). The "best" response to a sawtooth waveform are straight lines leading up to sharp points. Straight lines correspond to good speed regulation. Sharp points correspond to minimal overshoot. Following is a PID gain tuning procedure for the position control loop of a static testing machine using a sawtooth waveform. The procedure is similar to the one outlined in Section 2 for the step input. Thus, some of the detail has been left out.

Step 1: P only control. Disable integral and derivative action by setting Ki = 1 and Kd = 0, respectively. Adjust the proportional gain, Kp to achieve the straightest lines between endpoints. Figure 19 depicts sawtooth responses for a P only controller with proportional gains Kp = 50 and 1000. A review of Figure 19 shows that a proportional gain Kp = 50 produces poor speed regulation and much overshoot; whereas, Kp = 1000 produces good speed regulation and minimal overshoot.

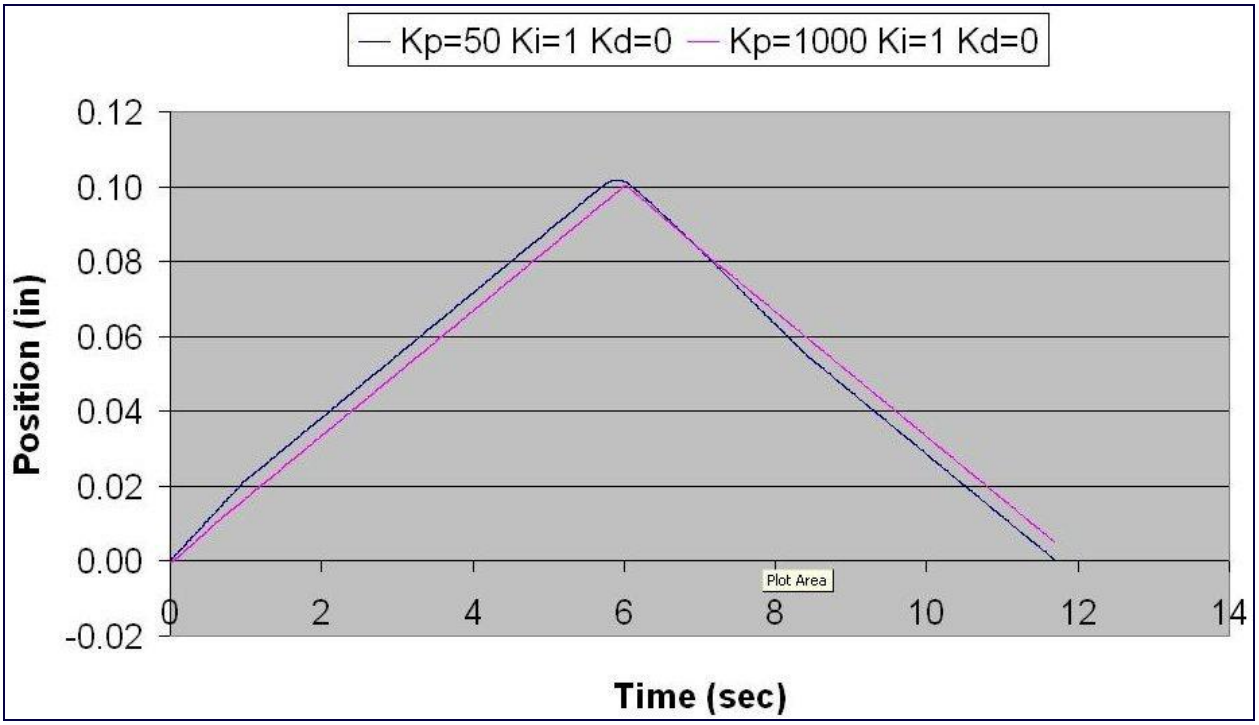


Figure 19 – Sawtooth waveform responses for a P only controller.

Step 2: PI control. Disable derivative action by setting Kd = 0. Set the proportional gain, Kp to the "best" value as determined in Step 1 (Kp = 1000). Adjust the integral gain, Ki to minimize overshoot and produce the sharpest point. Figure 20 is a magnified display of the sawtooth response about the end point. A review of Figure 20 shows that employing a PI controller with integral gain Ki = 25, reduces the overshoot and sharpens the point. For most applications, implementing PI control provides acceptable response and the tuning process could stop at the end of Step 2. If derivative action is desired, proceed to Step 3, else go to Step 4 to complete the sawtooth tuning process.

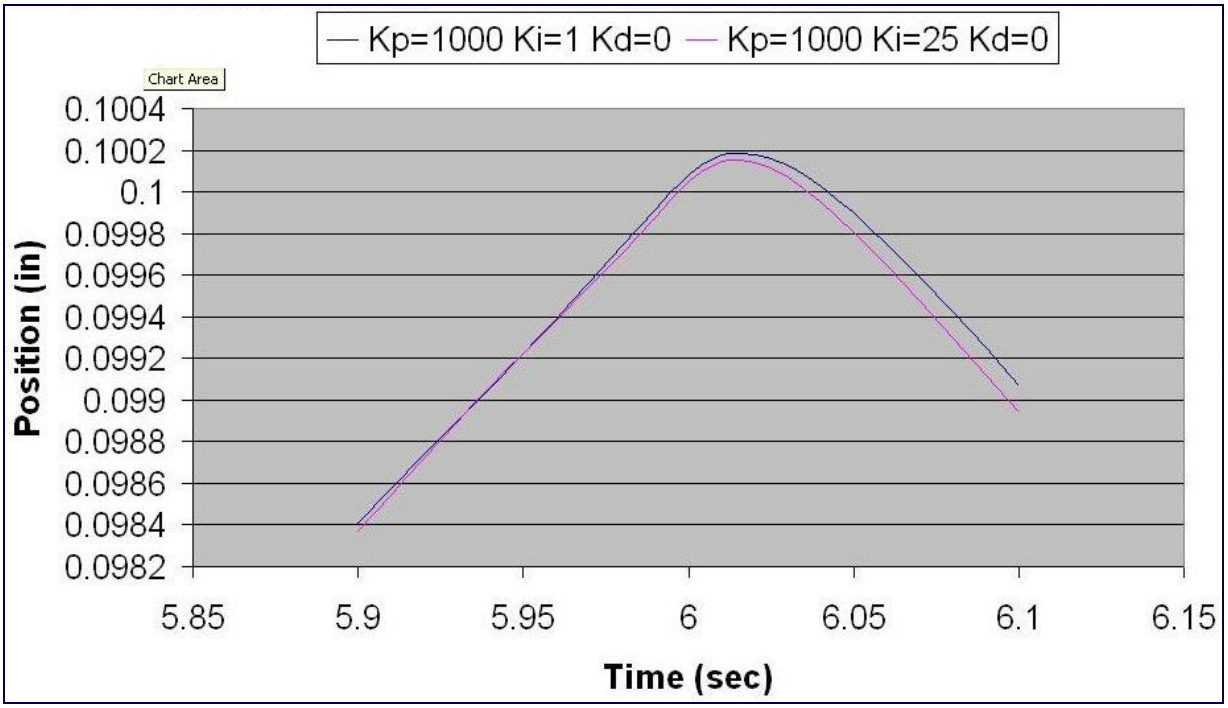


Figure 20 – Magnified sawtooth waveform responses for the "best" P only and PI controllers.

Step 3: PID Control. Set the proportional gain, Kp to the "best" value as determined in Step 1 (Kp = 1000). Set the integral gain, Ki to the "best" value as determined in Step 2 (Ki = 25). Adjust the derivative gain, Kd to minimize the endpoint overshoot and produce the sharpest point. Figure 21 is a magnified display of the sawtooth responses for the "best" PI and PID controllers. A review of Figure 21 shows that introducing derivative action further reduces endpoint overshoot and sharpens the point.

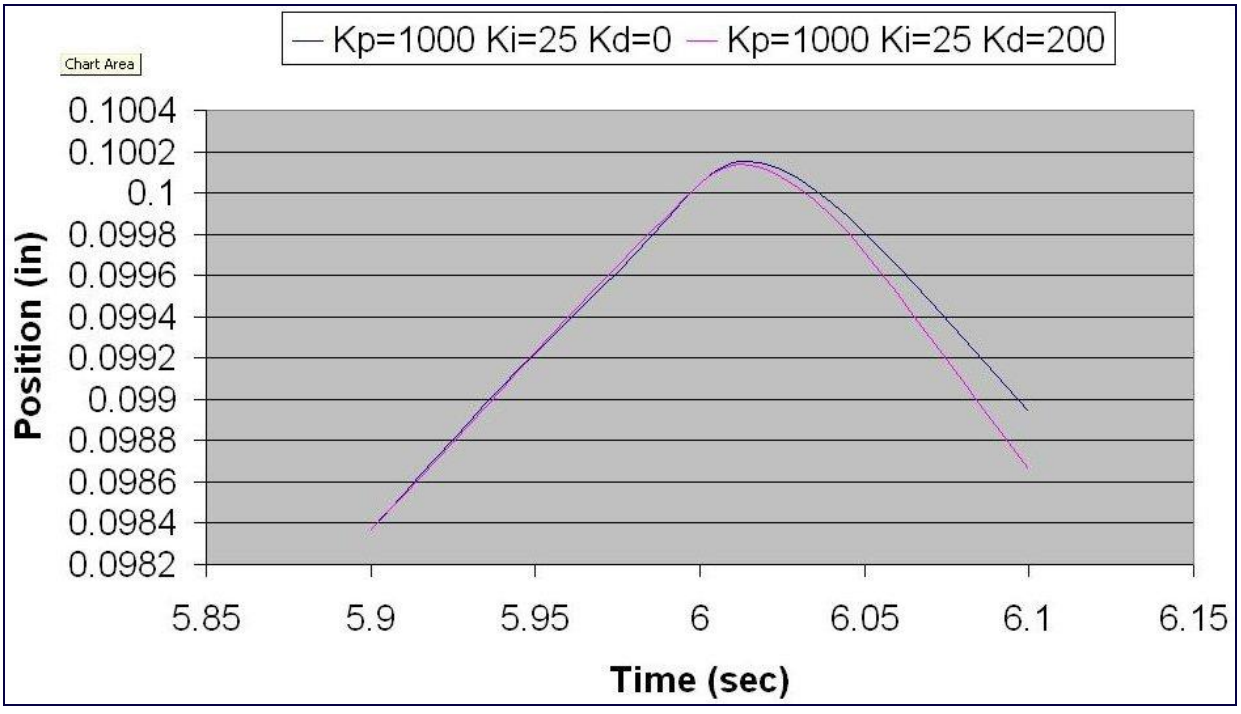


Figure 21 – Magnified sawtooth waveform responses for the "best" PI and PID controllers.

Step 4: Verify smooth operation at maximum crosshead speed. Set the manual Jog Rate to the testing machines maximum speed. Jog the crosshead up and down and verify smooth operation with little or no oscillation. Reduce the P and I gains if oscillations or a hammering effect is observed.

4.0 Gain Tuning Issues for Force and Strain Channels

Extensometers are used for axial and transverse strain measurements. Most extensometers are contacting type devices that clip on to the specimen. Like crosshead position transducers, extensometers are a displacement measuring sensor and are commonly used in static testing applications. As a result, the procedures outlined in Section 3.0 are well suited for tuning strain channel PID control gains.

Force feedback using a PID controller can be more difficult to tune. This is due to the change in the dynamic behavior of the test specimen and motor control system/testing machine with increasing force. For static testing applications, the procedures outlined in Section 3.0 are appropriate for tuning PID force control gains. However, there are several key points to keep in mind when tuning force control loops. First, the stiffness of the test specimen or how much it stretches under load relative to the stiffness of the testing machine load frame and motor control system matters. If the test specimen is very compliant (stretches much more) relative to the testing machine, good control over the entire force range is achievable. If the stiffness of the test specimen is equal or greater than the stiffness of the testing machine, then instability at higher forces may be experienced. To eliminate the instabilities at higher loads make the control loop more sluggish by reducing the proportional and integral gains. The end result of reducing the gains is larger control errors at lower loads but a stable control loop at higher loads.