Lab # 3 - Appendix C

PID Control

Analytical Model for the PID Controller in the Servo-Motor Module

As you know from the description of the Servomotor module setup in Controls Lab (Appendix B), several different controller schemes can be implemented on it, including several different variations of the PID Controller, as well as the Lead-Lag Controller. In this experiment, we will limit ourselves to implementing a form of a PID Controller – it is up to you to investigate which form of the PID Controller will work best for your servo.

Recall from Lab 2 (Appendix) that the standard configuration is the so-called **parallel form** of the PID Controller:

$$G_{PID}(s) = K_p + \frac{K_i}{s} + K_d s$$

In the servo module, a variation of that configuration is implemented, as follows:

$$G_{PID}(s) = K_p\left(1 + \frac{1}{\tau_i s} + \tau_d s\right)$$

Recall that often, for practical implementation reasons, it is often helpful to limit the Derivative action by replacing the pure Derivative term with a Derivative and a pole term, to limit the resulting bandwidth of the controller:

$$G_{PID}(s) = K_p\left(1 + \frac{1}{\tau_i s} + \frac{\tau_d s}{1 + N\tau_d s}\right)$$

N = 20 is one of the menu options when one sets the controller parameters for the Controller option selected.

Also recall that it is often helpful to limit the Integral action when the PI Controller saturates (the so called controller “windup” effect), by “switching off” the Integral term and replacing it with a combination of an Integral and a pole term. This is called an “anti-windup” configuration, as described below. The servo module options include variations on the three mode PID, with and without the anti-windup, as well as the limited (practical) form of the Derivative.

Anti-Windup Scheme for Integral Control

In real life systems, the actuator input can saturate due to physical limitations on its dynamic range. When this happens and the integral action is not switched off, the controller output keeps growing because the error signal is still present. Once the system comes out of saturation, the response will reflect a large overshoot, as a result of the energy stored in the integrator. This is known as Integrator Windup effect. To remedy the problem, an anti-windup scheme is implemented, illustrated in Figure 1, which turns off the integral action as soon as actuator saturation occurs. Figure 2 shows a Simulink implementation of this scheme.

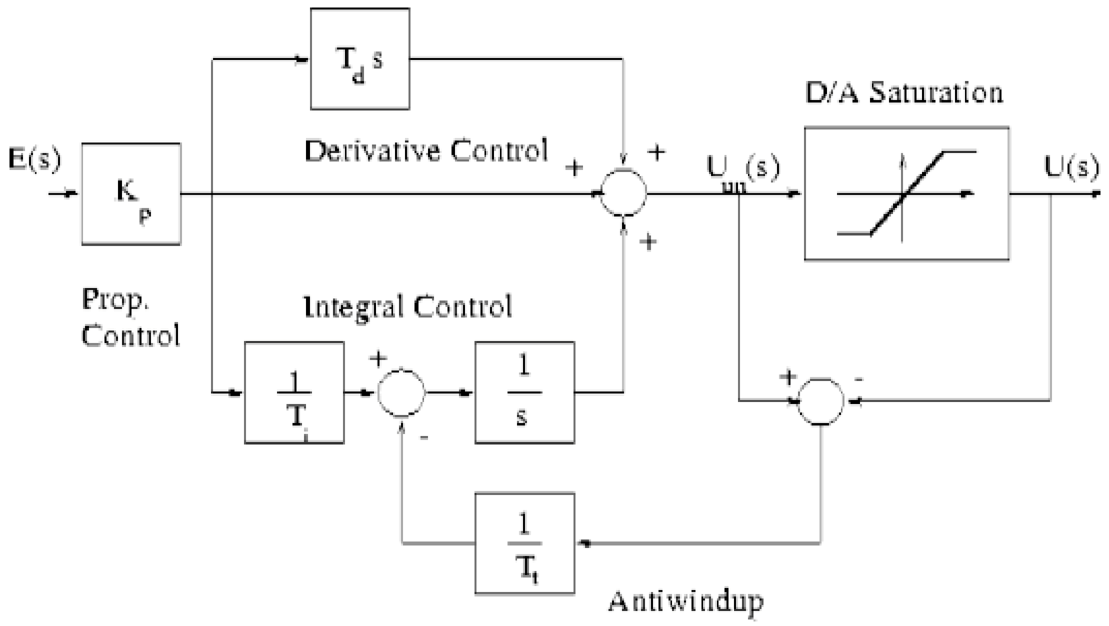

Figure 1: Parallel Structure of the PID Controller with Anti-Windup

You will notice that the menu of the Servo software gives you an option of using the Anti-Windup in your implementation of the PID Controller. Since employing the Anti-Windup will affect your response specifications, you should use it only if there is a visible problem with the controller saturation.

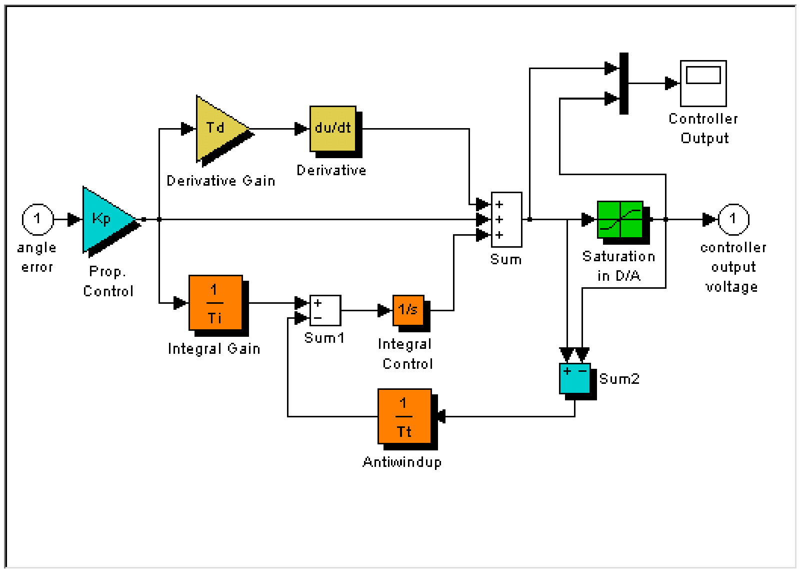

Figure 2: Simulink Implementation of the Integrator Anti-windup Scheme

In case of our Servo, the saturation occurs in the D/A converter. The unsaturated signal represents the controller digital algorithm output, and the saturated signal, U(s), is the actual analog controller output, which stays between ± 1.4 volts. When the system is unsaturated, the anti-windup signal is zero, and the integral action works as intended. When the controller saturates, an additional loop around the integrator is created, effectively replacing it with a first-order term. The smaller the anti-windup time constant T_t is, the less effective the integral action is (i.e. the stronger anti-windup action):

$$G(s) = \frac{T_t}{T_t s + 1}$$

Introducing the Anti-windup scheme into the controller structure increases the number of adjustable PID controller parameters to four:

- Proportional Gain K_p
- Integral Time Constant T_i (or Integral Gain K_i)
- Derivative Time Constant T_d (or Derivative Gain K_d)
- Anti-windup Time Constant T_t

The following simulations show what happens to the system response when Integrator Wind-up occurs. Figure 3 shows the Controller Output and the System Output under the PID Control when there is no saturation.

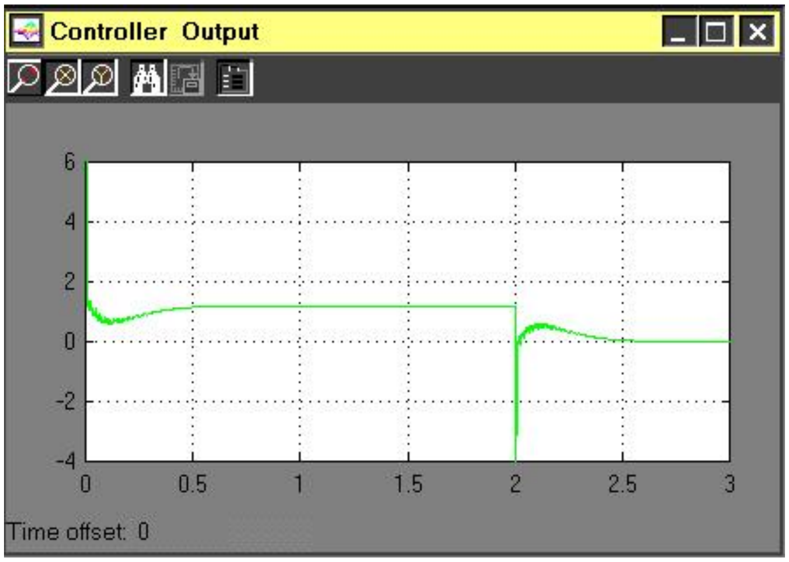

Figure 3 a)

PID Controller output when no saturation is present

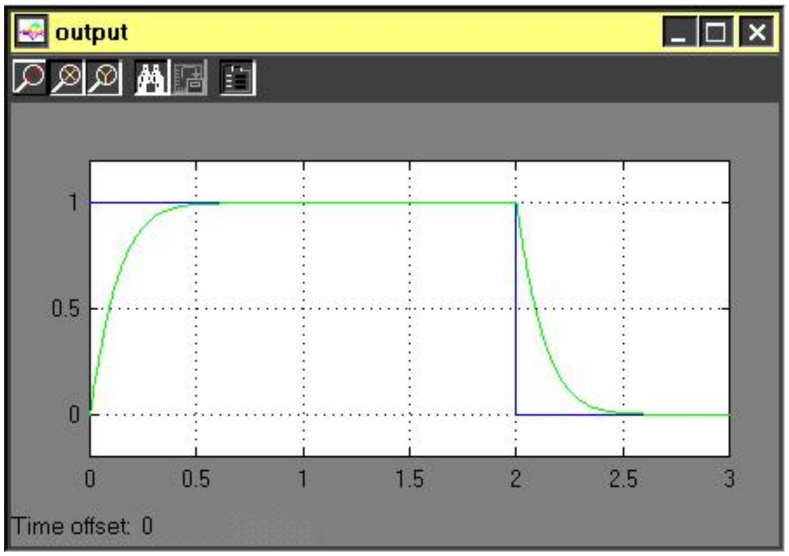

Figure 3 b)

Process output under PID Control when no saturation is present

Figure 4 shows the responses under PID Control when saturation occurs, and its effect on the Controller and Output and the System Output. Note the visible "clipping" of the controller output, as it saturates. The effect on the system response is its "winding-up", i.e. increasing, while the controller is saturated.

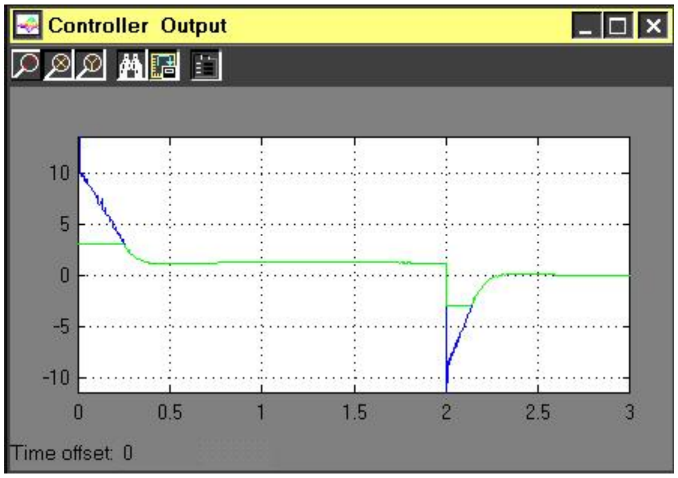

Figure 4 a)

PID Controller output when saturation is present

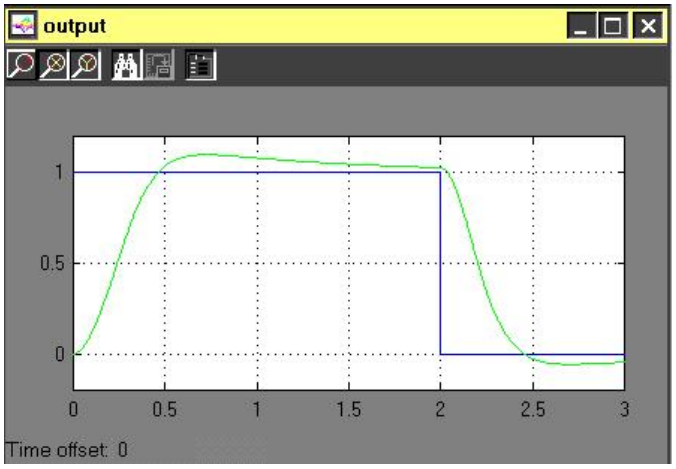

Figure 4 b)

Process output under PID Control when saturation is present and there is no Anti-windup protection – Integrator Windup visible.

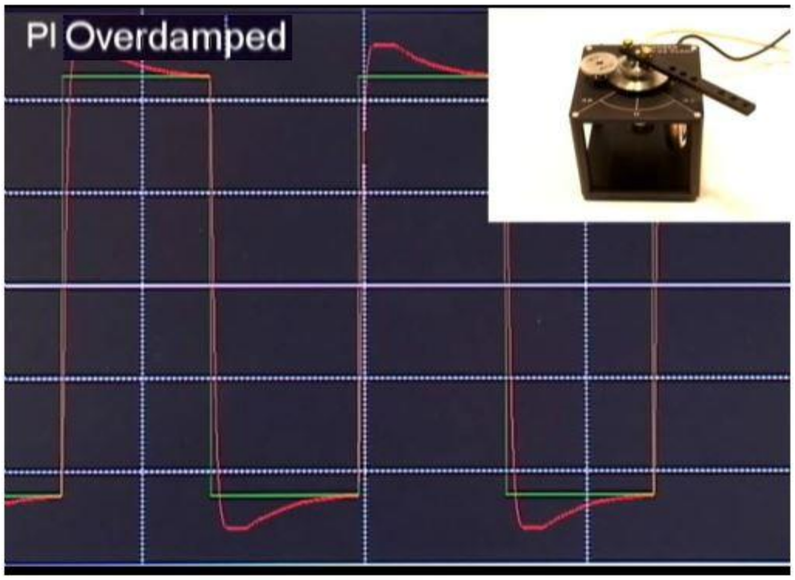

Figure 4 c)

Servomotor output under PI Control with Integrator Wiindup effect visible in the servo output.

Note the system is overdamped, and the Overshoot visible is the effect of the saturation in the controller output, and not of the low system damping.

Figure 5 shows the Controller Output and the System Output under PID Control when saturation occurs, but the PID Controller is augmented with the Anti-Windup block.

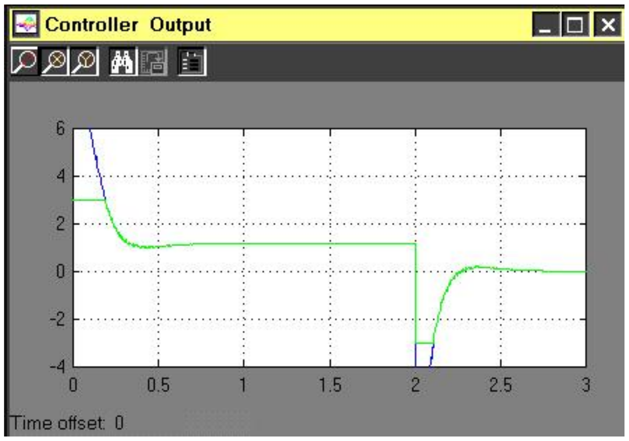

Figure 5 a)
PID+A Controller output when saturation is present – it is reduced.

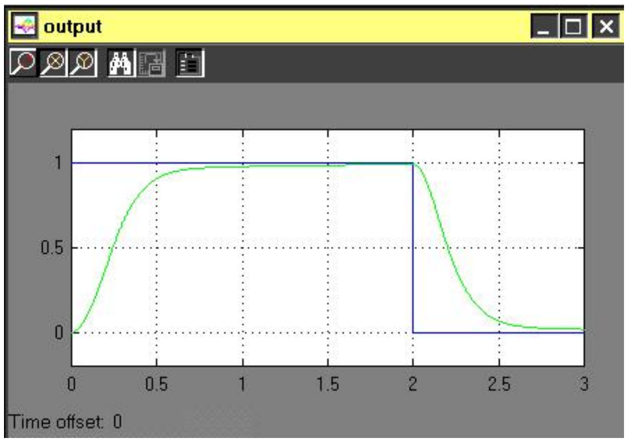

Figure 5 c)
Process output under PID Control with Integrator Anti-windup when saturation is present- the Integrator windup is not visible.

More on PID Controller Tuning

Recall that the PID Controller can be designed analytically, using either a Root Locus or a frequency response approach, or tuned directly (empirical approach), based on its responses in time domain, and following certain sets of tuning rules. Recall that strong PI action reduces steady state errors, but oscillations are the downside of using PI. PD action increases the system damping, and can therefore be used to reduce the oscillatory transients.

The tuning rules were listed as:

- Intuitive Tuning (also known as "The Seat of the Pants" tuning approach)
- Ziegler-Nichols Ultimate Cycle Method
- Ziegler-Nichols Process Reaction Method

The general rule is to use Proportional Control as a “muscle” doing work, use Integral Control to improve tracking, and use Derivative Control to damp out oscillations.

HINT: Think of the Proportional Mode as the "main meal", and of the Integrator Mode and the Derivative Mode as "salt" and "pepper".

Method 1: Intuitive Tuning, a.k.a. “Seat of the Pants” Approach

Figure 6 shows screen shots of a simulated LTI system response where PID Controller settings were intuitively tuned.

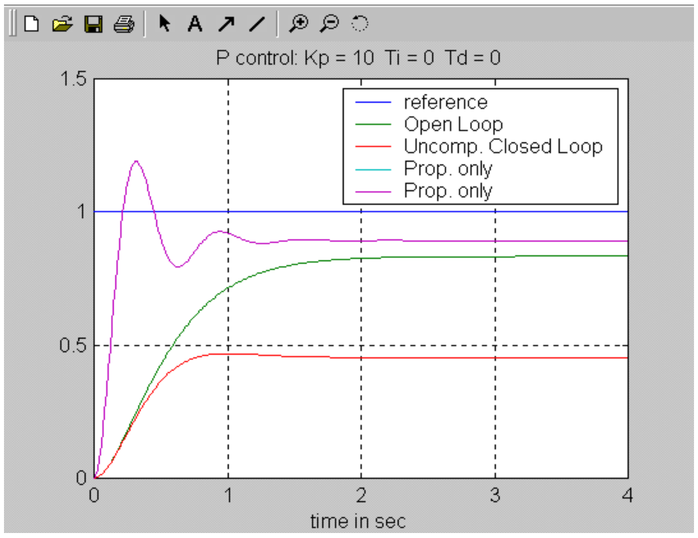

Figure 6 a)

Proportional Control Only
$K_p = 10$

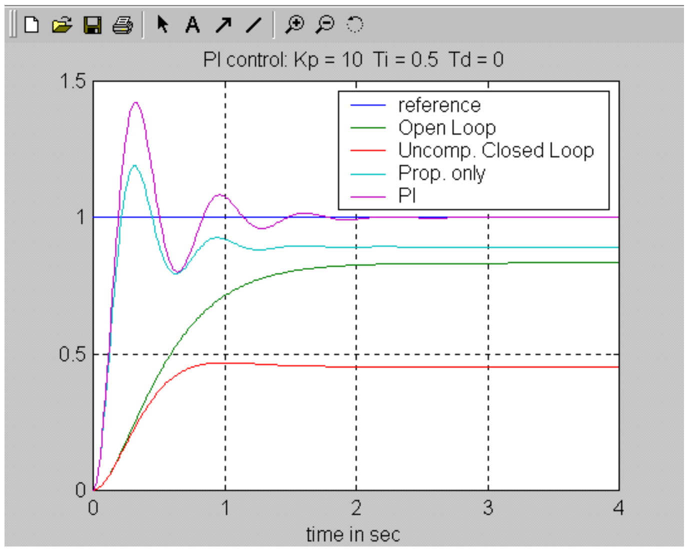

Figure 6 b)

Proportional Control
$K_p = 10$

Integral Control
$T_i = 0.5$

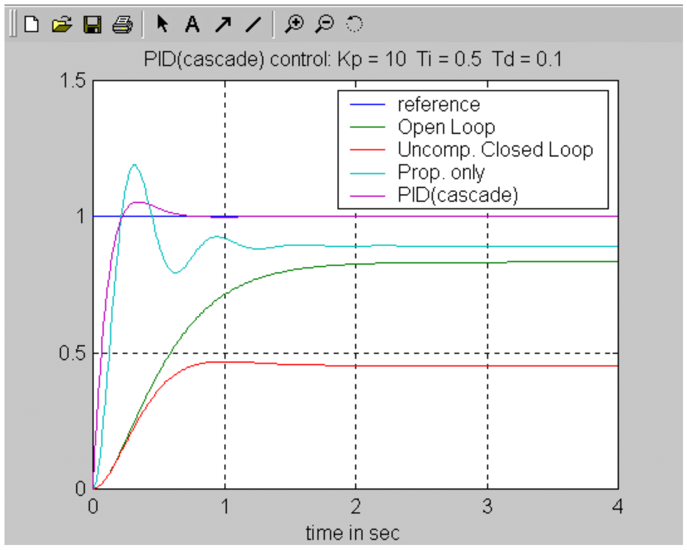

Figure 6 c)

Proportional Control
$K_p = 10$

Integral Control
$T_i = 0.5$

Derivative Control

$T_d = 0.1$

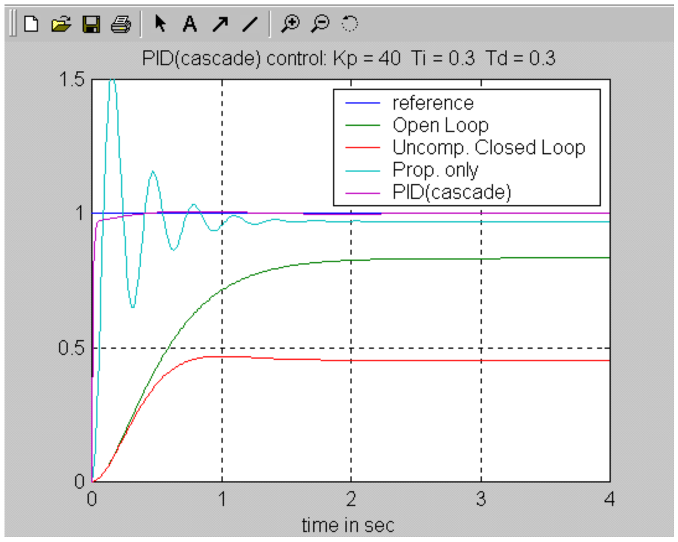

Figure 6 d)

Proportional Control
$K_p = 40$

Integral Control
$T_i = 0.3$

Derivative Control

$T_d = 0.3$

In reality, because increasing the gain of the controller will typically saturate it, as for example in the Servo Module, the actual, nonlinear, simulation results will look slightly different, as shown in Figure 7. The controller saturation can actually work to our advantage, increasing the system damping. The Anti-windup Scheme can be used as well.

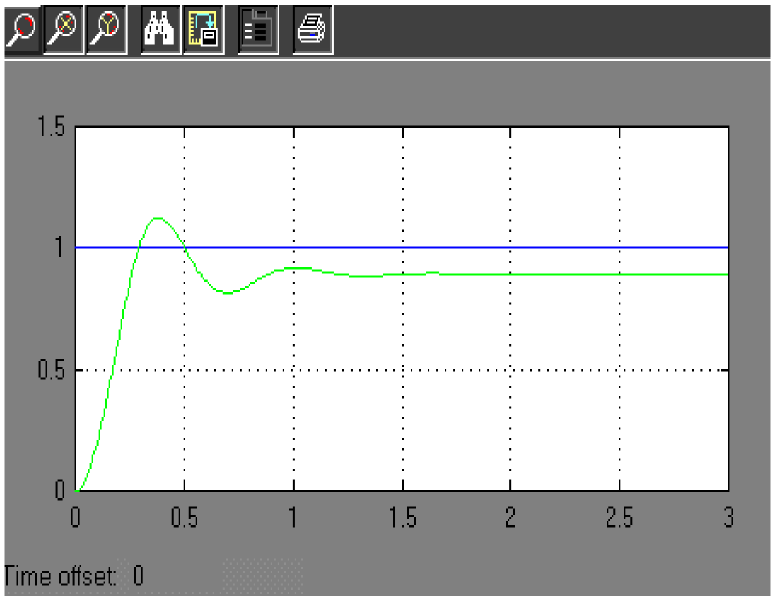

Figure 7 a)

Proportional Control Only
$K_p = 10$

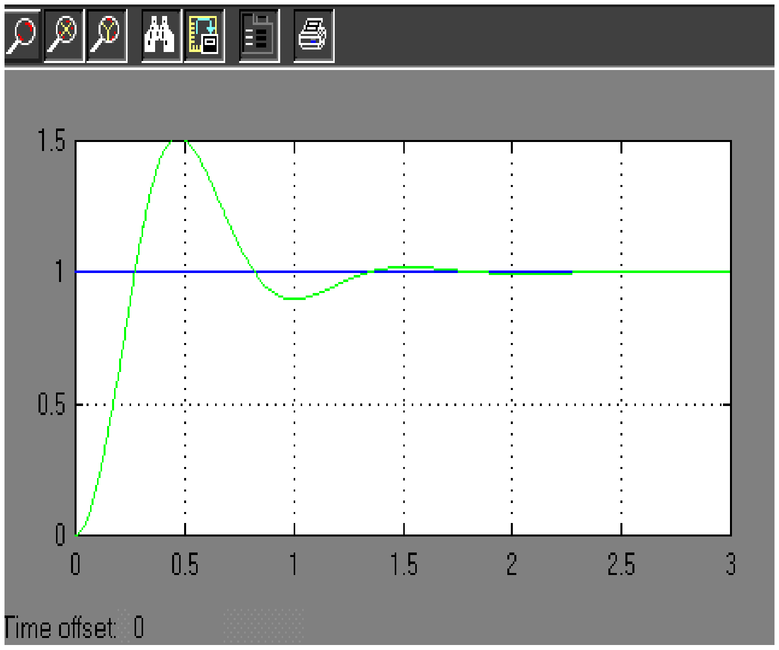

Figure 7 b)

Proportional Control
$K_p = 10$
Integral Control
$T_i = 0.2$
Derivative Control
$T_d = 0.1$

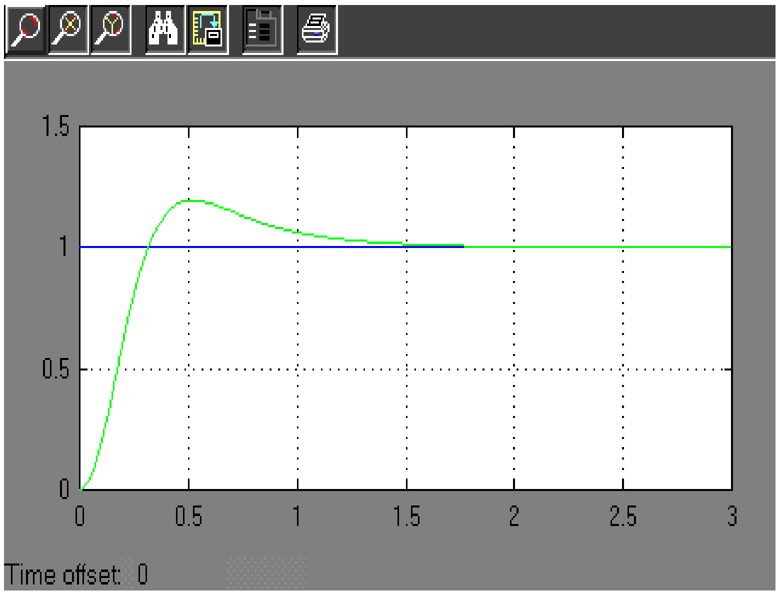

Figure 7 c)

Proportional Control
$K_p = 40$
Integral Control
$T_i = 0.2$
Derivative Control
$T_d = 0.1$
Antiwindup Constant
$T_t = 0.2$

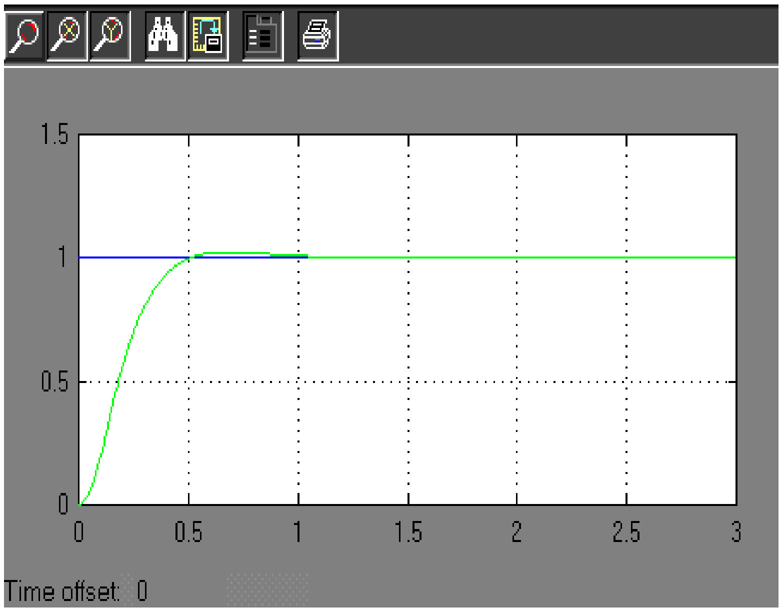

Figure 7 d)

Proportional Control
$K_p = 40$
Integral Control
$T_i = 0.3$
Derivative Control
$T_d = 0.1$
Antiwindup Constant
$T_t = 0.2$