Experiment 5

Aim: To analyze the effect of proportional, integral and derivative controllers on the step response of 2nd order system.

Consider a process controlled by PID controller $G_p = \frac{400}{s(s+50)}$

1. Obtain step response of overall transfer function of the system.
2. Observe effect of proportional controller.
3. Observe effect of proportional + integral control.
4. Let K_p be not equal to zero, then add derivative control and investigate its effect on step response

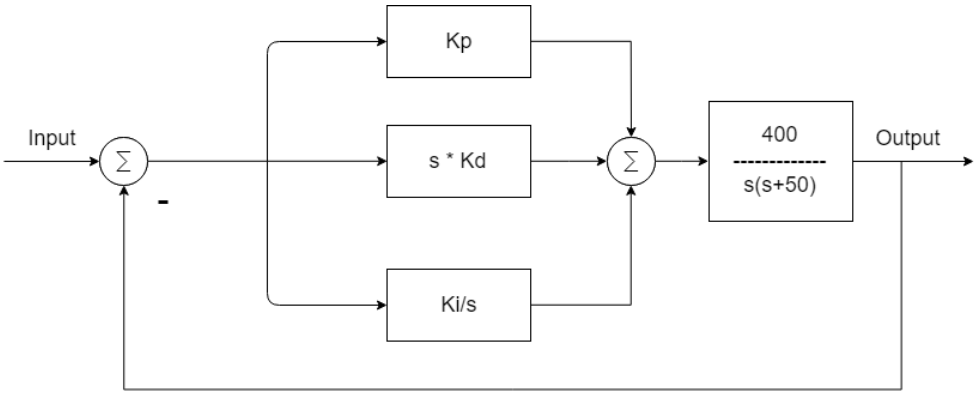


Fig 1: Given System

Software: Matlab 2018a

Theory: A proportional–integral–derivative controller (PID controller. or three-term controller) is a control loop mechanism employing feedback that is widely used in industrial control systems and a variety of other applications requiring continuously modulated control. A PID controller continuously calculates an error value e(t) as the difference between a desired setpoint (SP) and a measured process variable (PV) and applies a correction based on proportional, integral, and derivative terms (denoted P, I, and D respectively), hence the name.

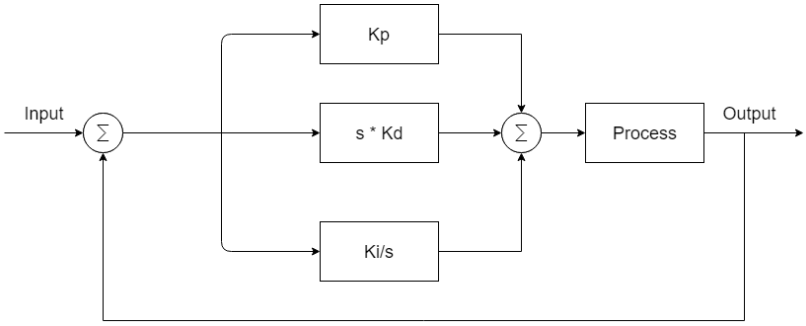


Fig. 2: PID controller

For PID action:

$$G_{PID} = Kp + s * Kd + Ki/s$$

Where $K_p$is Proportional gain, K_d is derivative gain and Ki is integral gain.

1. **Proportional Action:** Proportional action provides an instantaneous response to the control error. This is useful for improving the response of a stable system but cannot control an unstable system by itself. Additionally, the gain is the same for all frequencies leaving the system with a nonzero steady-state error. We set $K_d = K_i = 0$ for for proportional action.

2. **Integral Action:** Integral action drives the steady-state error towards 0 but slows the response since the error must accumulate before a significant response is output from the controller. Since an integrator introduces a system pole at the origin, an integrator can be detrimental to loop stability.

3. **Derivative Action:** Derivative action acts on the derivative or rate of change of the control error. This provides a fast response, as opposed to the integral action, but cannot accommodate constant errors (i.e. the derivative of a constant, nonzero error is 0). Derivatives have a phase of +90° leading to an anticipatory or predictive response. However, derivative control will produce large control signals in response to high frequency control errors such as set point changes (step command) and measurement noise. In order to use derivative control the transfer functions must be proper. This often requires a pole to added to the controller.

For the given system, it can be reduced to:

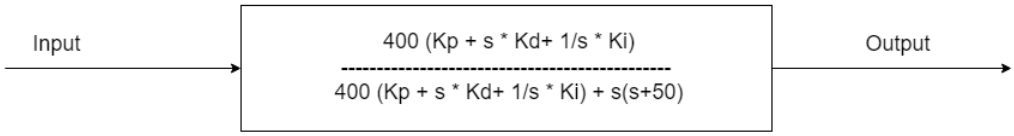


Fig. 3: Reduced System

Matlab Code:

```
clc;
clear all;
close all;
Gs = tf([0 0 400],[1 50 0 ]);
Y = feedback(Gs,1);
% step(Y);

Kp = [2 10 40 50];
for i = 1:length(Kp)
    t = tf([Kp(i)],[1]);
    s = series(t,Gs);
    C(i) = feedback(s,1);
    str = sprintf('kp = %d ',Kp(i));
    subplot(2,2,i),step(C(i));
    title(str);
end

k=1;
Kpp = [2 6 10];
Ki = [0.1 10 60];
for i = 1:length(Kpp)
    tp(i) = tf([Kpp(i)],[1]);
    for j = 1:length(Ki)
```

```
        ti(j) = tf([Ki(j)],[1 0]);
        a = parallel(tp(i),ti(j));
        se(k) = series(Gs,a);
        R(k) = feedback(se(k),1);
        str = sprintf('kp = %d and ki = %d',Kpp(i),Ki(j));
        subplot(3,3,k),step(R(k));
        title(str);
        k = k+1;
    end
end
```

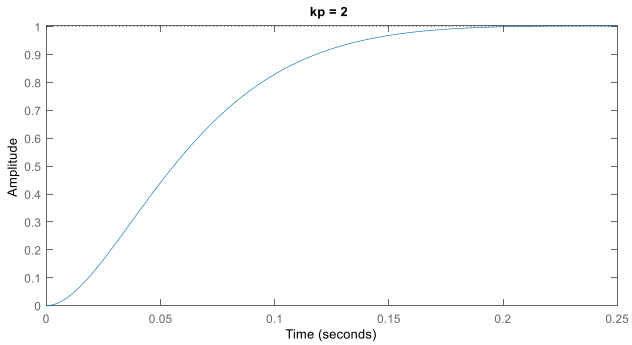
```
k=1;
kii = [0.1 1 5];
kppp = 10;
kd = [0.1 1 10];
tpp = tf([kppp],[1]);
for i = 1:length(kii)
    tii(i) = tf([kii(i)],[1 0]);
    for j = 1:length(kd)
        td(j) = tf([kd(j) 0],[0 1]);
        a = parallel(tii(i),parallel(td(j),tpp));
        see(k) = series(Gs,a);
        Re(k) = feedback(see(k),1);
        str = sprintf('kp = %d, ki = %0.2f and kd =
%0.2f',kppp,kii(i),kd(j));
        subplot(3,3,k),step(Re(k));
        title(str);
        k = k+1;
    end
end
```

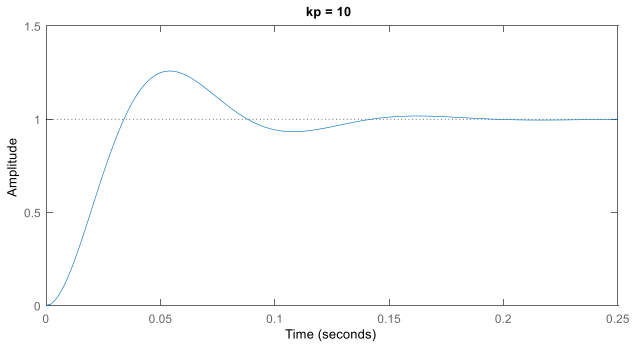
Calculation:

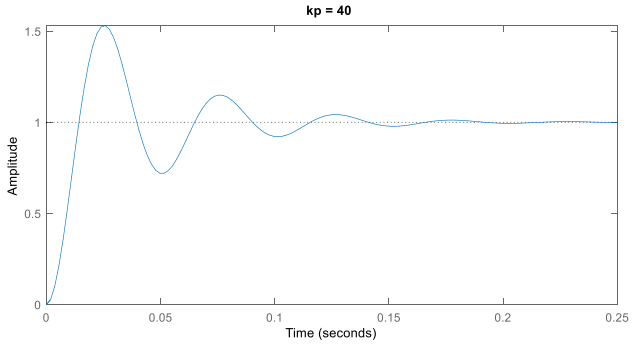
Control	P controller			PI controller($K_p = 10$)			PID Controller(K_p = 10 & $K_p = 10$)		
	K_p = 2	K_p =10	K_p = 20	K_i=0.6	K_i=10	K_i=60	K_d=0.1	K_d=1	K_d=10
T_r	0.0994	0.023	0.0095	0.0339	0.0327	0.0288	0.0266	0.0093	0.0005
T_s	0.1601	0.134	0.1557	0.1619	0.3629	0.2609	0.0897	0.1567	0.0012
%M	0.26	25.85	52.997	15.53	20.04	40.35	-	0.0976	-

Result:

Various step responses were observed with different values of K_p, K_i and K_d







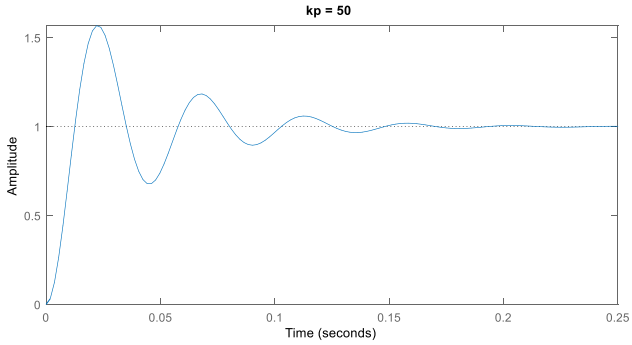
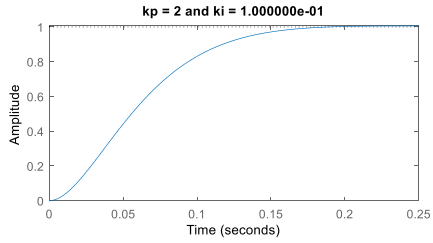
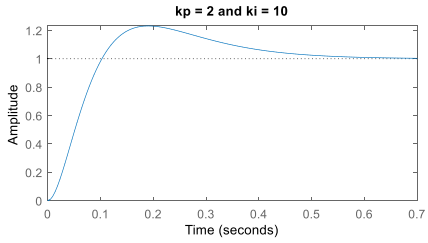
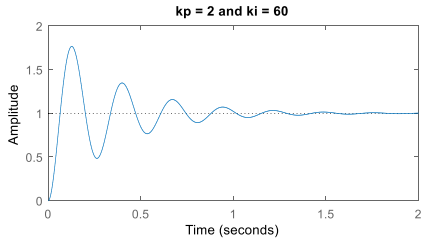
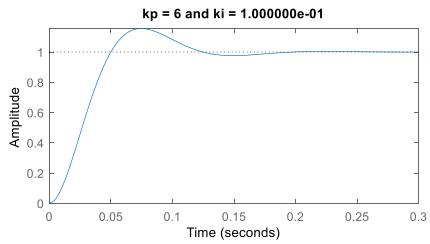


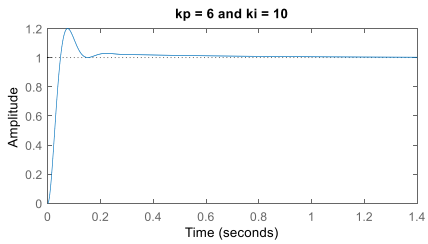
Fig 4: After applying P controller

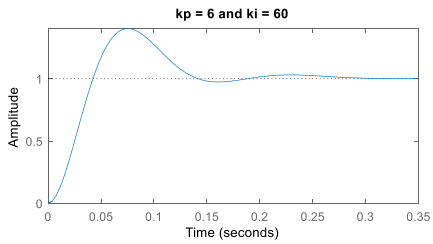


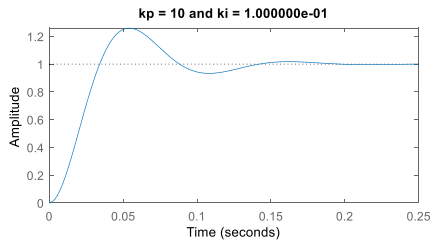


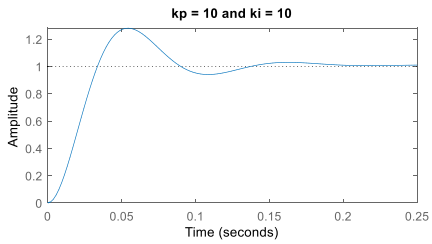












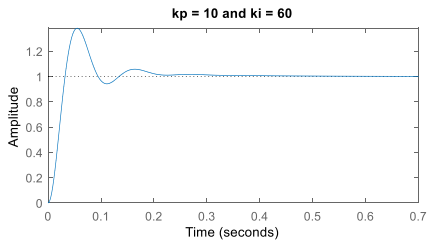


Fig 5: After applying PI to the system

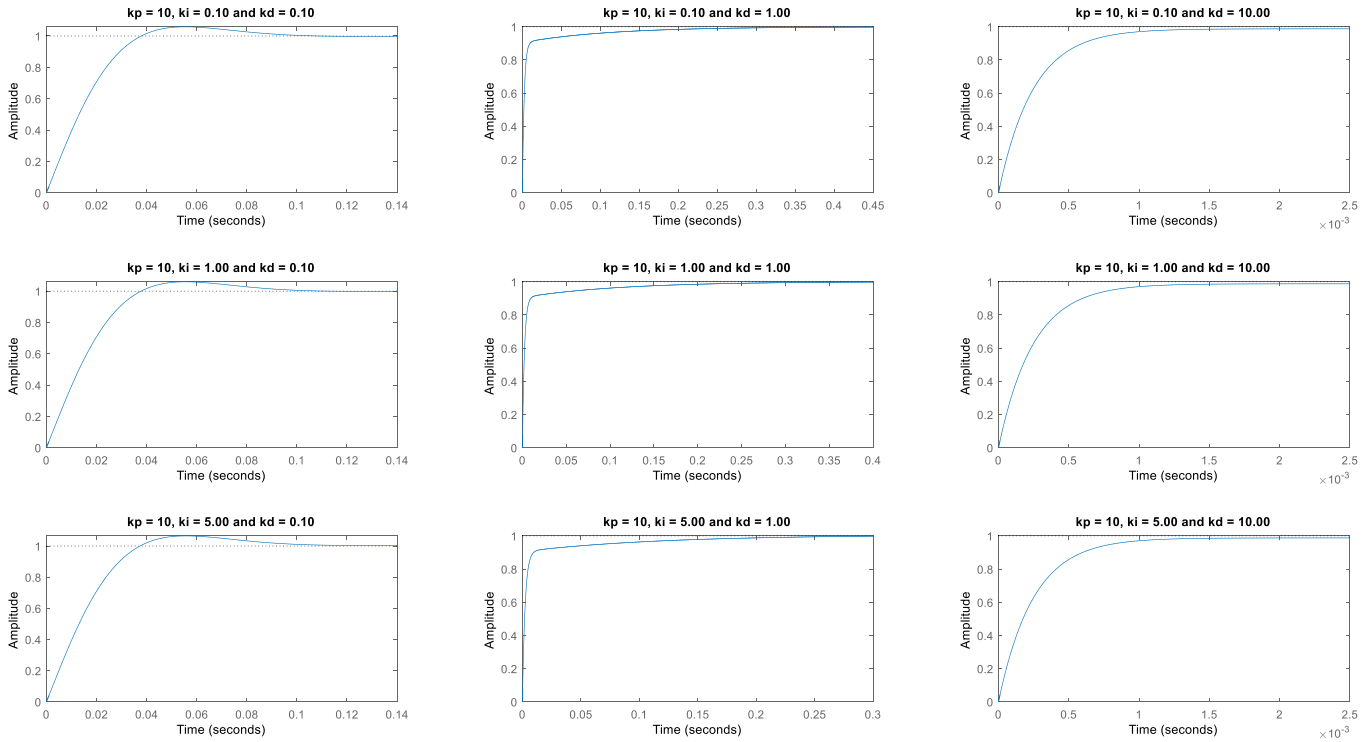


Fig 6: After applying PID to the system

Conclusion:

We observed various effect of K_p, K_i and K_d on the system are as follow:

- When K_p increase then rise time and steady-state decrease, overshoot increase and there is a little change in settling time
- When K_i increase then rise time decrease, overshoot and settling time increase and steady-state error tends to zero.
- When K_i increase then there is a small change in rise time and steady-state error, overshoot and setting time decrease.