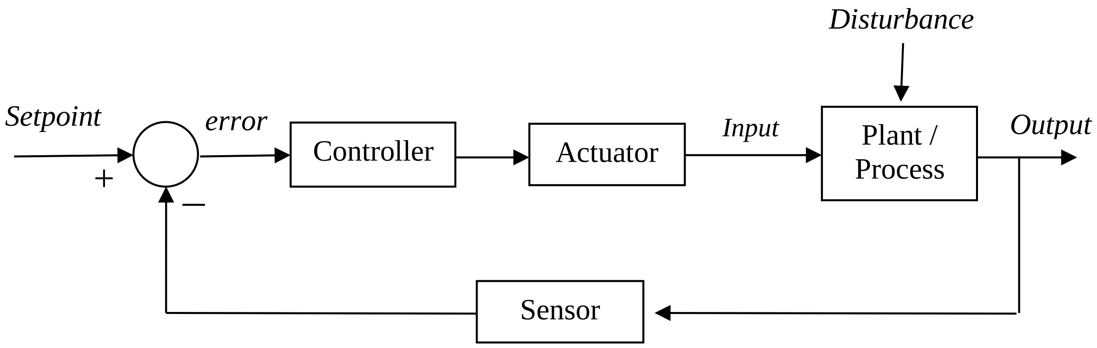
# 9. Controller Design

## 9.1 Introduction

- Recall the basic control system loop:




- At this stage of the module we have learned to model and analyze a selection of reasonably simple first and second-order continuous-time dynamical processes or systems.

- We are now at a stage where we can start to design suitable controllers for such systems.

- In general, there is a multitude of controllers that can be used, each one with their own strengths and weaknesses.

- Here, we are going to consider three of the simpler (and commonly used) controllers – an on/off controller, a proportional controller and a PID controller.

- *You will study more advanced controllers and control techniques in later modules in the BE programme.*

## 9.2 On/off Control


- **On/off control** is also known as **bang-bang control** or **hysteresis control** and refers to the most basic type of control technique whereby the controller output allows for only one of two possible states.

- There is no middle state in this type of control. The controller output (and hence the controlled system input) is either **off** or **on**.

- Such controllers are typically used in heating systems where, for example, a furnace is switched on when the room temperature drops below a specified setpoint temperature and is switched off the temperature exceeds the setpoint temperature.

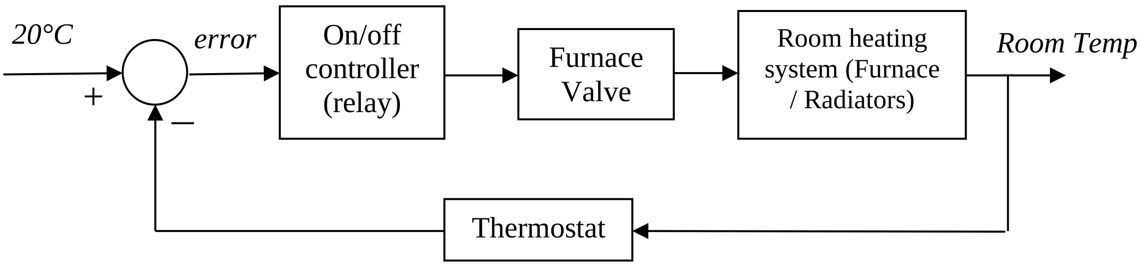
- Recall that the actuator (or the final control element) is the device that changes the input to the process.

- Actuators could be motors that vary position and speed or valves that control the flow of liquids (such as fuel to the furnace in the heating system).

- In on-off control, it is the actuator that is switched on or off as needed.

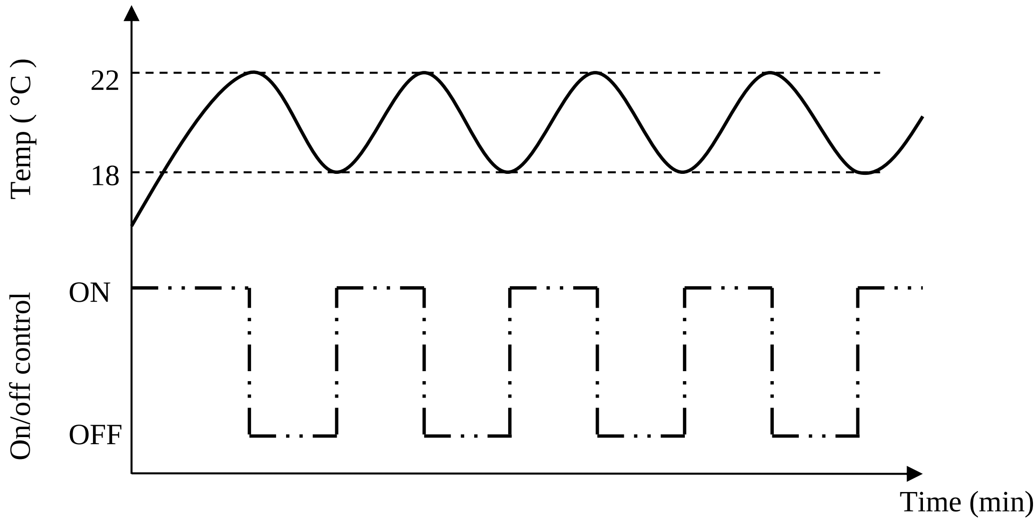

- Consider a room temperature system that uses on-off control as follows – the fuel valve to the furnace is switched on whenever the thermostat in the room records a temperature below 18°C and switches off whenever the temperature is recorded at above 22°C.

- The simplified block diagram for this control system is as follows:



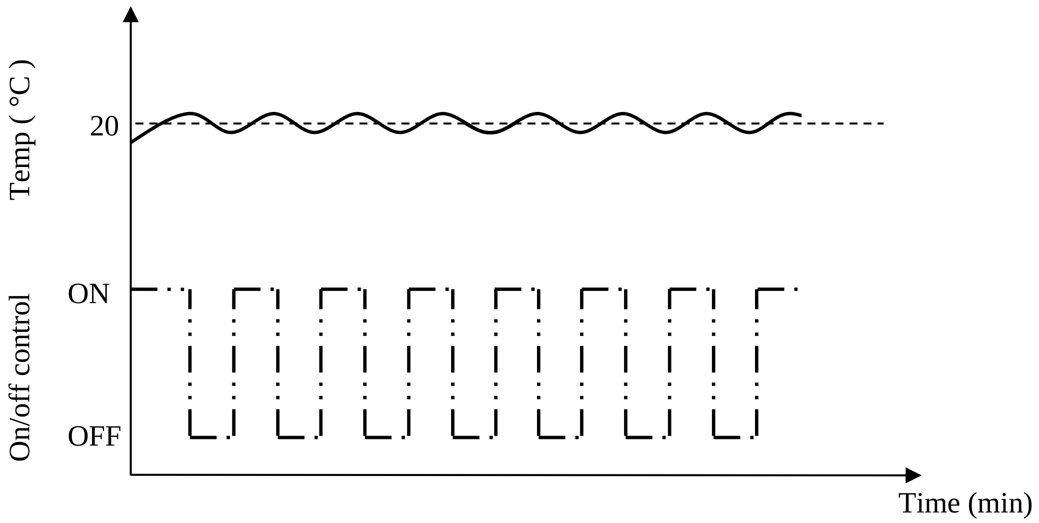

- Note, in the above diagram, a **relay** is an electrically operated switch that provides the on/off control output as required.

- Below are the expected controller and system outputs:




- Note that it is common practice to have different setpoints for switching the controller on and off.

- If the same setpoint was used for both, then the controller output would continually switch at a high frequency as follows:

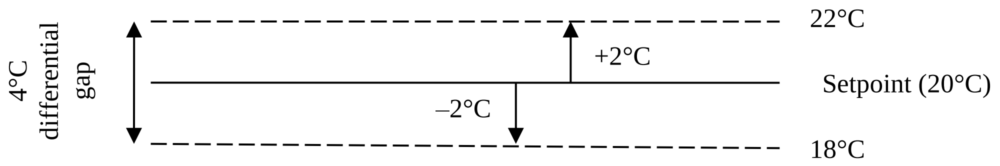



- This rapid on/off oscillation would accelerate wear of the actuator (in this case the flow valve) and hence shorten its life.

- Hence, we introduce a gap between the upper and lower setpoints (as shown previously) to prevent this rapid cycling between on and off.

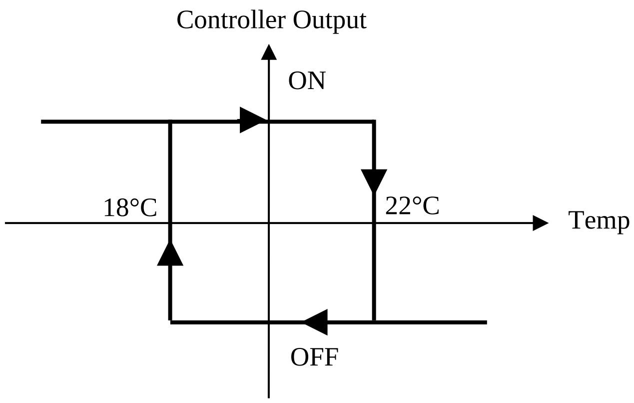
- This gap is known as the **differential gap** or a **deadband** and allows the temperature to change for a certain amount before the valve condition is altered.

- In our first system, we used a 4°C differential gap in our controller, as follows:



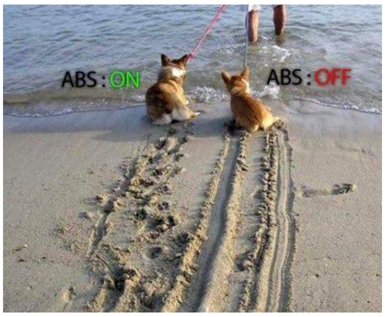

- Hence, for a setpoint of 20°C the valve will only turn on when the temperature has dropped 2°C below the setpoint and will only turn off when the temperature rises 2°C above the setpoint.

- This effect is also commonly referred to as **hysteresis** effect**,** where the output depends not only on the current input but also on knowledge of its past inputs.



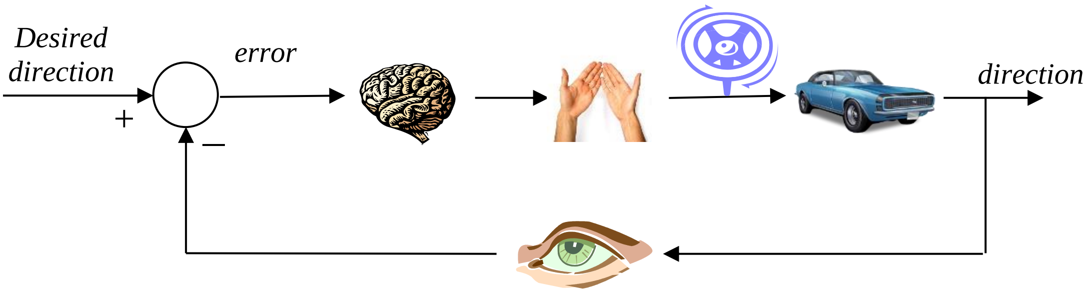
- Here, when the temperature is 20°C, is the valve (or the controller output) on or off? The answer will depend on the previous temperature.

- If the temperature is dropping, then the valve is off. If however, the temperature is increasing, then the value is actually on.




## 9.3 Proportional Control

- Recall our car driving control system from the first lecture and consider the case where we are driving down a road, making sure to stay within the correct driving lane.




- As we notice the car drifting a bit to the left, we steer slightly to the right. Likewise, if we notice the car drifting a bit to the right, we steer slightly to the left.

- Furthermore, if the car drifts a lot to the left (or right), we steer a lot to the right (or left) to compensate.

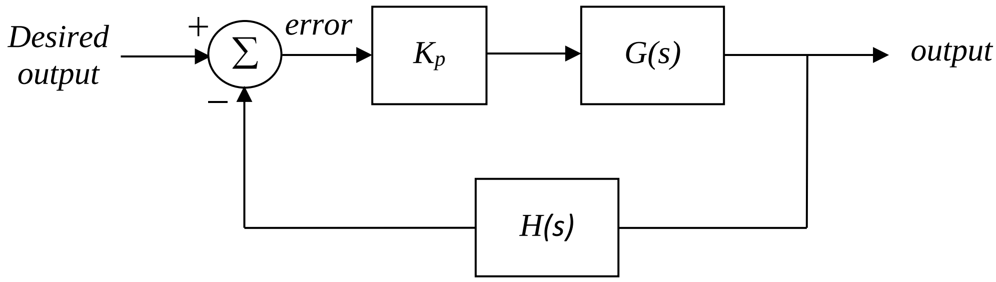
- This type of control is known as **proportional control**, where the *control output is proportional to the error measurement*.

- Had we employed a more rigid on/off control approach, we would not adapt the steering until we had crossed a lane marker, at which point we'd make a large and sudden steering adjustment to return the car back into the lane – *not something other drives want to see!*

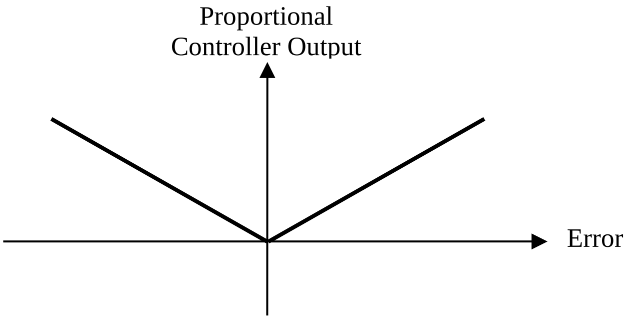
- Proportional control offers better flexibility and a smoother output then on/off control and works by changing the input to the process (or system) in proportion to the error between the desired output and the measured output.

- While on/off control offers one of two possible outputs, proportional control offers a range of outputs which are simply scaled quantities of the error signal (in this case, the difference between the desired driving position and the actual driving position).

- In general, the proportional controller is normally denoted by a $K_p$ gain block and hence the block diagram of the control system looks like:



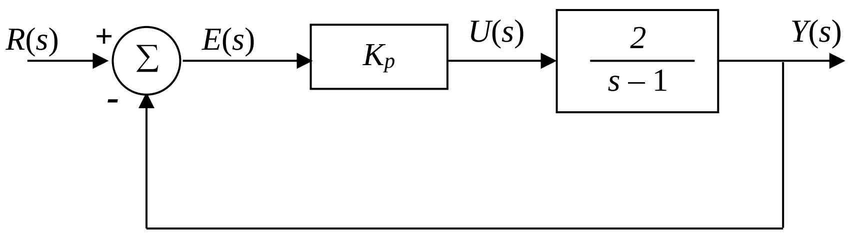

- The value of gain $K_p$ can vary from 0 to $\infty$ (in theory) and thus the output of the controller takes the form:




where $K_p$ determines the slope of the line.

- Let us now consider the transient, steady-state and stability of a control system using a proportional controller by considering a serious of examples.

- ***Ex 9.1: Determine the range of Kp for which the following closed-loop feedback control system is stable:***




**Solution:**

Firstly, using block algebra we obtain the closed-loop transfer function (CTLF) as follows:

Combine the blocks in the forward path to give: $G(s) = \dfrac{2K_p}{s-1}$

Since $H(s) = 1$, i.e. a unity feedback system, the CLTF is given by:

$$\frac{G}{1+GH}=\frac{\dfrac{2K_p}{s-1}}{1+\dfrac{2K_p}{s-1}}=\frac{2K_p}{s-1+2K_p}$$

Poles of the CLTF are given by: $s-1+2K_p=0 \quad \Rightarrow s=1-2K_p$

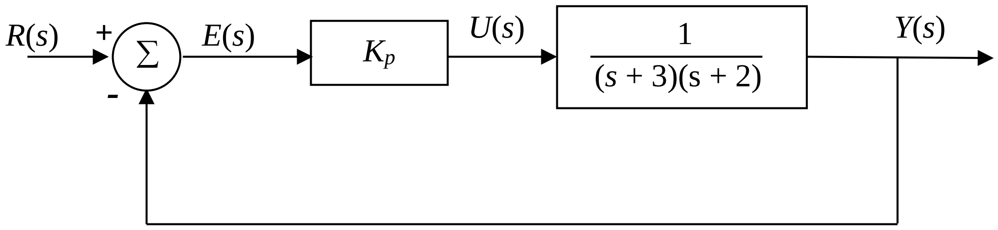
For stability: $\mathrm{Re}(s)<0$

Hence: $1-2K_p<0 \quad \Rightarrow 2K_p>1 \quad \Rightarrow K_p>\dfrac{1}{2}$

- In this example, a proportional controller gain *Kp* greater than 0.5 is needed to ensure that the closed loop system is stable.

- Note that the open loop system as defined by $G(s)$ is unstable as its pole is located at $s = 1$.

- ***Ex 9.2: Determine the relationship between Kp and the damping and natural frequency for the following closed-loop feedback control system:***




**Solution:**

CLTF: $$\frac{G}{1+GH}=\frac{\dfrac{K_p}{(s+3)(s+2)}}{1+\dfrac{K_p}{(s+3)(s+2)}}=\frac{K_p}{(s+3)(s+2)+K_p}=\frac{K_p}{s^2+5s+(K_p+6)}$$

Compare this with the standard second order transfer function:

$$\frac{K\omega_n{}^2}{s^2+2\zeta\omega_n s+\omega_n{}^2}$$, where $K$ is the gain of the system itself in this case.

Comparing denominators:

$$\omega_n{}^2=K_p+6 \quad \Rightarrow \omega_n=\sqrt{K_p+6}$$

$$2\zeta\omega_n=5 \quad \Rightarrow \zeta=\frac{5}{2\sqrt{K_p+6}}$$

- Thus, in the above system, both the damping and the natural frequency are directly affected by changes in the gain value of the proportional controller (akin to changing the gain of the system itself).

- For this system, increasing $K_p$ increases the natural frequency $\omega_n$ and decreases the damping value $\zeta$.

- In other words, increasing $K_p$ makes the system response more oscillatory in nature.

- ***Ex 9.3: Determine the relationship between Kp and the steady-state output for the closed-loop feedback control system in Ex. 9.2 given that the desired setpoint is a constant A.***

  **Solution:**

  We know that the CLTF is: $$\frac{K_p}{s^2 + 5s + (K_p + 6)}$$

  The steady-state output is given by:

$$CLTF_{ss} = \lim_{s \to 0} CLTF(s) \quad = \lim_{s \to 0} \frac{K_p}{s^2 + 5s + (K_p + 6)} \quad = \frac{K_p}{K_p + 6}$$

  Thus, for a constant input A, the steady-state output will simply be:

$$A \ \text{x} \ \frac{K_p}{K_p + 6} \quad = \quad \left( \frac{K_p}{K_p + 6} \right) A$$

- It is worth noting that in the closed loop system $A$ represents the desired output. This is the value that we want our control system to achieve.

- When $K_p = 1$, the output is: $\frac{A}{7} = 0.143A$

- When $K_p = 10$, the output is: $\frac{10}{16} A = 0.625A$

- When $K_p = 100$, the output is: $\frac{100}{106} A = 0.943A$

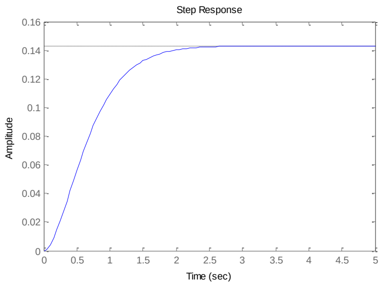
- When $K_p = 1000$, the output is: $\frac{1000}{1006} A = 0.994A$

- Thus, as we increase the value of $K_p$, we increase the steady-state output. However, we never actually achieve the desired output.

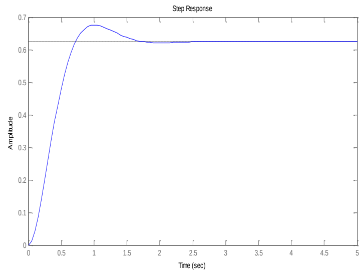
- In theory, this is only possible if $K_p = \infty$. This is clearly not very practical!

- Furthermore, as we have seen in the previous example, as $K_p$ increases, the response will become increasingly oscillatory, which is also not a practically desirable outcome.

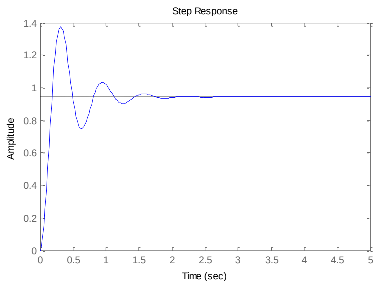
- By way of illustration, the unit step response for the closed-loop system in ex 9.2 is shown below for a range of $K_p$ values.



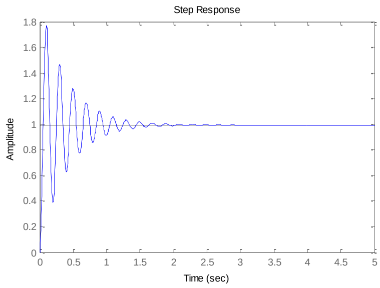

$K_p = 1$



$K_p = 10$




$K_p = 100$




$K_p = 1000$

- We can clearly see that as $K_p$ increases, the response tends towards instability.

- We can also see that the steady-state error (i.e. the desired output – the actual output in steady-state) gets smaller, but remains nevertheless. The desired output in this case is 1.

- Herein lies the problem with proportional control.

- While it is a relatively straightforward simple controller that can affect system stability and its transient characteristics, it fails to guarantee the most fundamental demand of achieving the desired final outcome.

- Hence, the need for a more advanced controller – here we will consider the PID controller.

- *Recall using the P controller (in lab 4) on both the simulated mass-spring-damper system and the actual balance control system (hardware) – a steady-state error always remained!*

## 9.4 PID Control

- The PID controller consists of the sum of three parts, namely **p**roportional, **i**ntegral and **d**erivative as follows:

$$u(t) = K_p e(t) + K_i \int e(t) + K_d \frac{de(t)}{dt}$$

- Here, $K_p$, $K_i$ and $K_d$ represent the proportional, integral and derivative gain values respectively.

- This can be readily expressed in transfer function form as:

$$\frac{U(s)}{E(s)} = G_c(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

- The integral action is captured in the $\frac{1}{s}$ term, while the derivative action is captured by the $s$ term (recall section on Laplace transforms).

- The PID controller is also often expressed as follows: $G_c(s) = K_p\left(1 + \frac{1}{t_i s} + t_d s\right)$

- Here, $t_i$ is known as the *integral time* and satisfies: $K_i = \frac{K_p}{t_i}$

  and $t_d$ is known as the *derivative time* and satisfies: $K_d = K_p t_d$


- So why do we need three components? Good question!

- We have already seen the use of a proportional controller, its benefits and its weaknesses.

- **Why integral**? Without getting into detail mathematics, it suffices to say that integral action is used to improve the long term precision of the control loop.

- Note that the integral of a function is the same as the area under that function. In the control loop, we are integrating the error function and hence obtaining a value for the accumulation of error over time.

- By using this parameter, we can modify the system input to eliminate this error.

- **Thus, in summary, integral action allows us to eliminate the steady-state error in a given process or system**, i.e. we can get the output of the system to match the desired setpoint.

- **Then why derivative**? The disadvantage of adding integral action is that it tends to increase the oscillations and destabilize the system (similar to the effect of increasing the gain in a proportional controller, but even more pronounced).

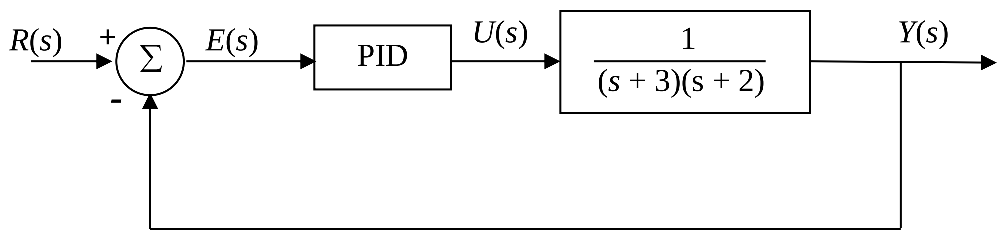
- Derivative action is this used to negate this effect by slowing the response down. The derivative relates to the rate of change of the error.

- As such, we now have a parameter that can slow the system down when the rate of change of error increases. The bigger the value of gain $K_d$ the slower our response becomes.

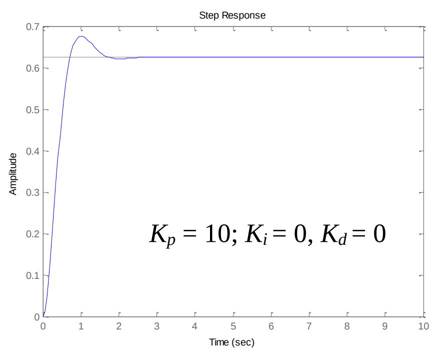
- **Thus, in summary, derivative action helps in stabilizing a system.**

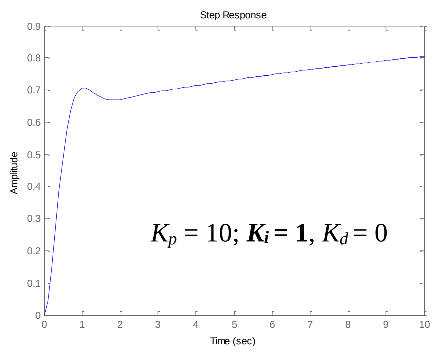
- We will now illustrate these important traits of the PID controller through a series of simulation examples, using the same system as for the proportional controller earlier.

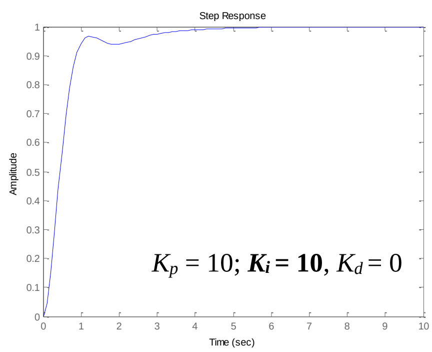
- The block diagram of the system is:

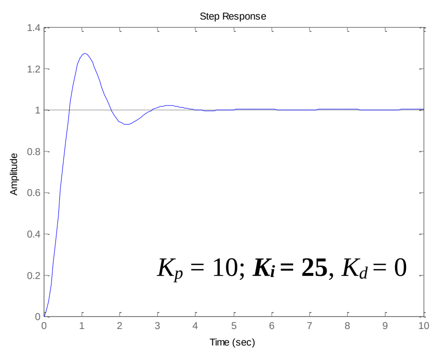


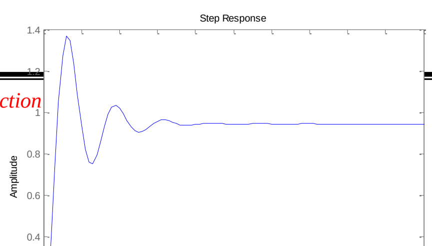

- The unit step response for this system is shown below for a range of $K_p$, $K_i$, and $K_d$, values, as specified for each graph:

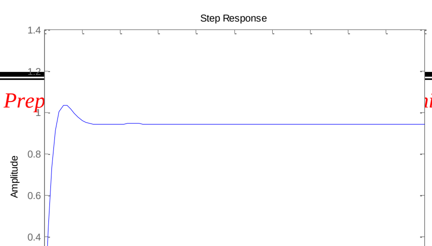


- From these graphs we can observe the following:

  - When $K_i = 0$ (i.e. no integral action), the output of the system does not meet the desired setpoint of 1. There is a steady-state error in the final output.

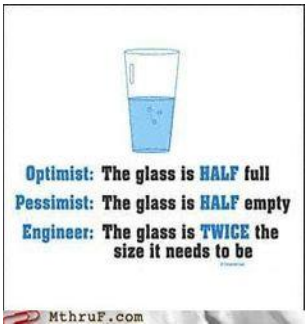
  - Once integral action is included, this error is eliminated. Note, in the case of $K_i = 1$, the response will eventually settle to 1 but we would need to view the response over a longer time period.

  - As we increase the value of $K_i$ we observe that the system output meets its desired setpoint quicker. However, this also results in increased oscillations (tending the system towards instability).

  - Finally, comparing the last two cases in terms of derivative action we see that the overshoot and oscillations of the system response is significantly reduced when $K_i$ = 10, i.e. when derivative action is introduced into the controller.

  - Note also that when derivative action is introduced alongside proportional control, the steady-state error still exists. In order words, we still need integral action to eliminate this error.

- *Recall using the PID controller (in lab 4) on both the simulated mass-spring-damper system and the actual balance control system (hardware) – the steady-state error was eliminated when integral action was included.*




- In the previous system, we showed the output response for a range of control parameters. *But how do we decide on these parameters?*

- Numerous methods exist that allow us to select the appropriate PID parameters for a whole range of systems.

- Here, we are only going to consider one of the simpler, commonly used techniques, known as Ziegler-Nichols.

## 9.5 Ziegler-Nichols Tuning Rules for PID Controllers

- This is an ***experiment-based technique*** that is simple to use.

- It's based on the step response and does not require a model of the process.

- Ziegler-Nichols Tuning rules are based on one of two methods.

***Method 1***

- The first method is based on the *Ultimate Cycle* or *Stability Limit method* and works as follows:

  - **Step 1** – Set $K_d = K_i = 0$.

  - **Step 2** – Increase $K_p$ until the system is marginally stable, i.e. the system response oscillates. *(This makes the assumption that the system will oscillate at some point!)*

  - **Step 3** – Note the value of $K_p$ at this point and determine the period of the oscillations. We refer to these values as $K_c$ and $t_c$ respectively.

  - **Step 4** – Choose control parameters according to the Ziegler-Nichols rules, based on $K_c$ and the period of oscillation $t_c$ as follows:

| | $K_p$ | $t_i$ | $t_d$ |
|---|---|---|---|
| P | $0.5\ K_c$ | | |
| PI | $0.45\ K_c$ | $t_c / 1.2$ | |
| PID | $0.6\ K_c$ | $t_c / 2$ | $t_c / 8$ |

  - Note that we can obtain values for $K_d$ and $K_i$ as follows:

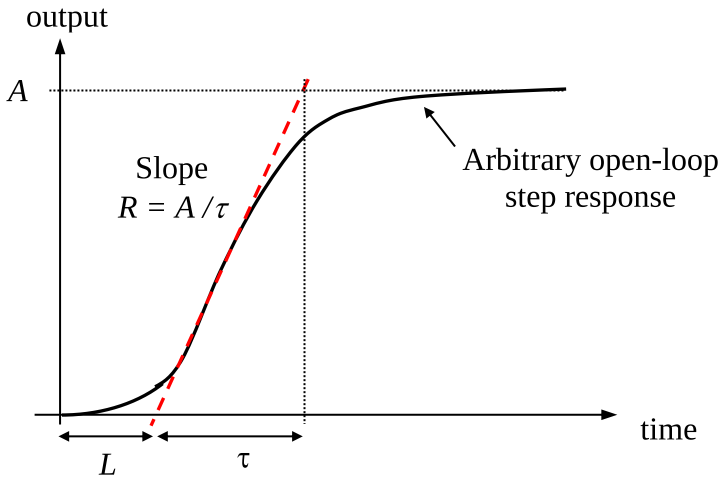
$$K_i = \frac{K_p}{t_i} \qquad \text{and} \qquad K_d = K_p t_d$$

- This heuristic method will give a reasonably good performance.

- *Note that you do not need to remember this table as it will be provided for you in an exam situation, if needed. However, you do need to understand the various parameters used.*

***Method 2***

- The second method operates upon the shape of the step response of the open-loop system, as follows:




- The controller parameters are calculated based on *R* (the steepest slope) and *L*, the dead-time of the system, according to the following table:

| | $K_p$ | $t_i$ | $t_d$ |
|---|---|---|---|
| P | 1 / *RL* | | |
| PI | 0.9 / *RL* | 3 *L* | |
| PID | 1.2 / *RL* | 2 *L* | 0.5 *L* |

- This method is only useful when the system / output signal has a large dead-time and a well-defined signature.

- An example of such a system is a control system for a water tank system, whereby the input flow valve is adjusted based on the height of water in the tank (with a delayed feedback).

- *Once again, you do not need to remember this table but you do need to understand the various parameters used and how to determine them.*

- *As the Ziegler-Nichols rules are primarily experimental-based, we will leave the design of PID controllers using these rules for our laboratories.*

- *Finally, you will study PID controllers in more detail in further modules in the BE programme, where you will also examine alternative and more advanced controllers.*


*... for EE114 only!*