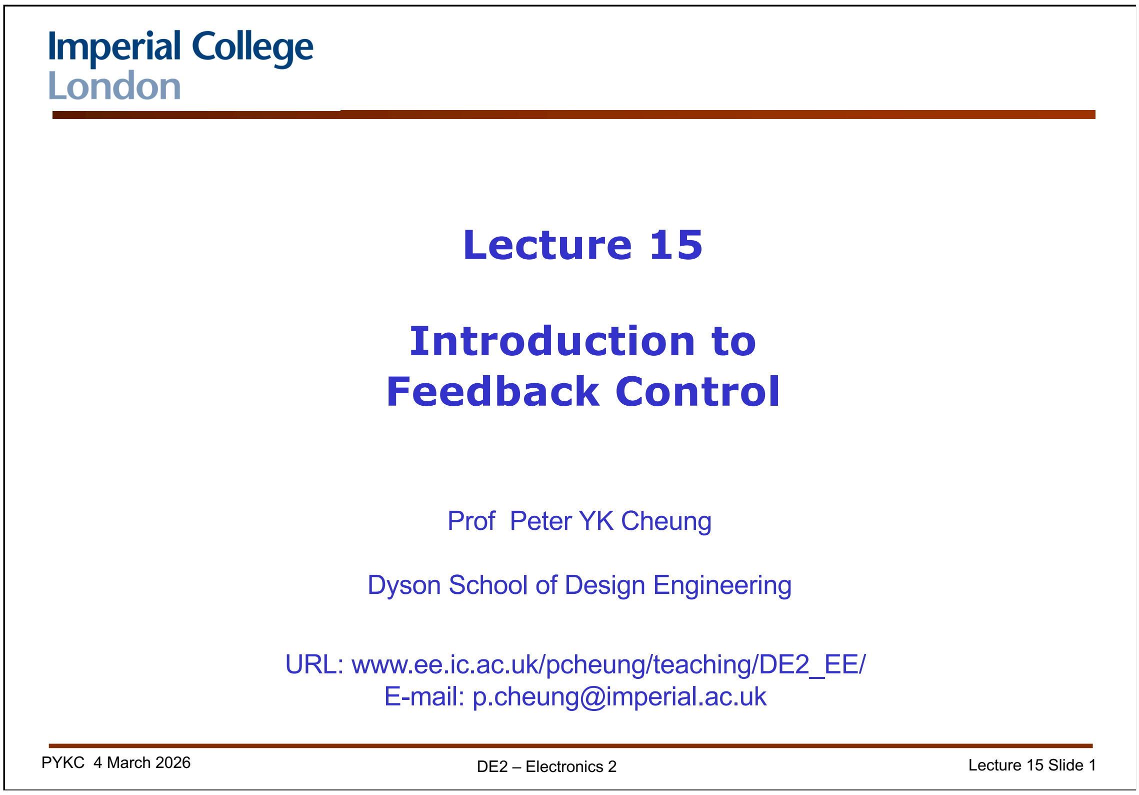


Imperial College London

Lecture 15

Introduction to Feedback Control

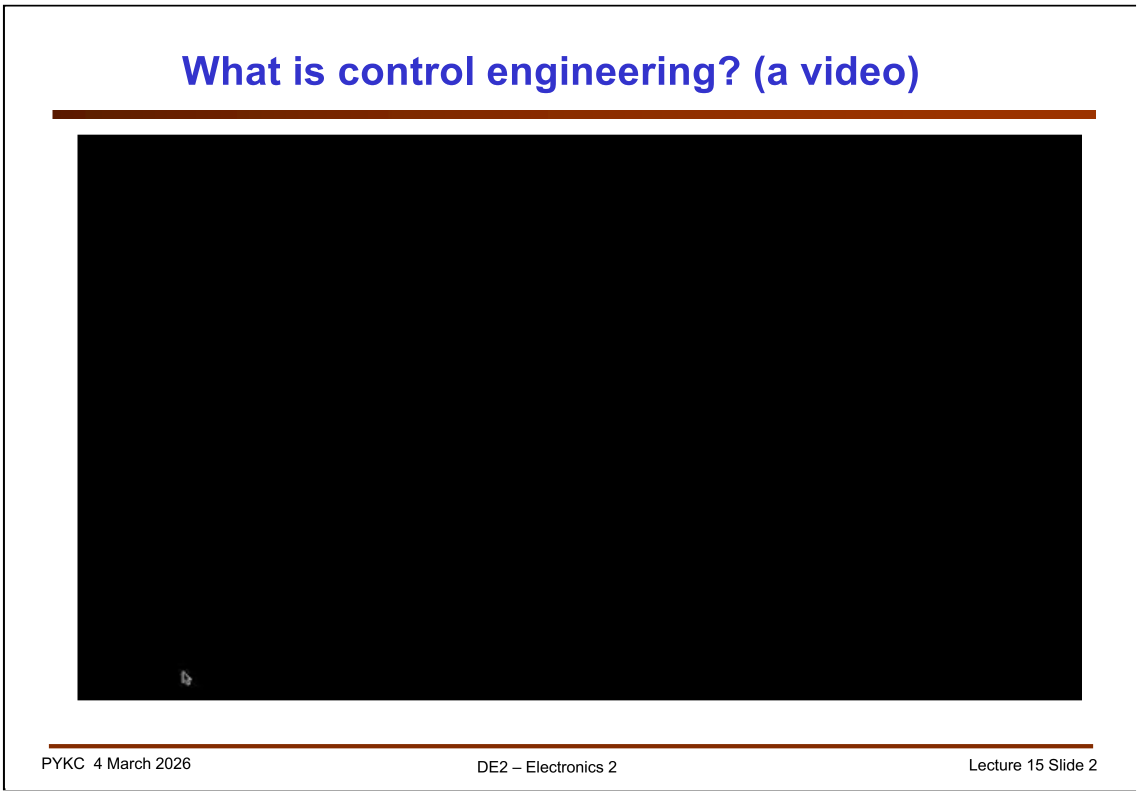
Prof Peter YK Cheung

Dyson School of Design Engineering

URL: www.ee.ic.ac.uk/pcheung/teaching/DE2_EE/
E-mail: p.cheung@imperial.ac.uk

In this lecture, we will start the final topic for this module on feedback control. This is required for you to design the self-balancing feature of the Segway.

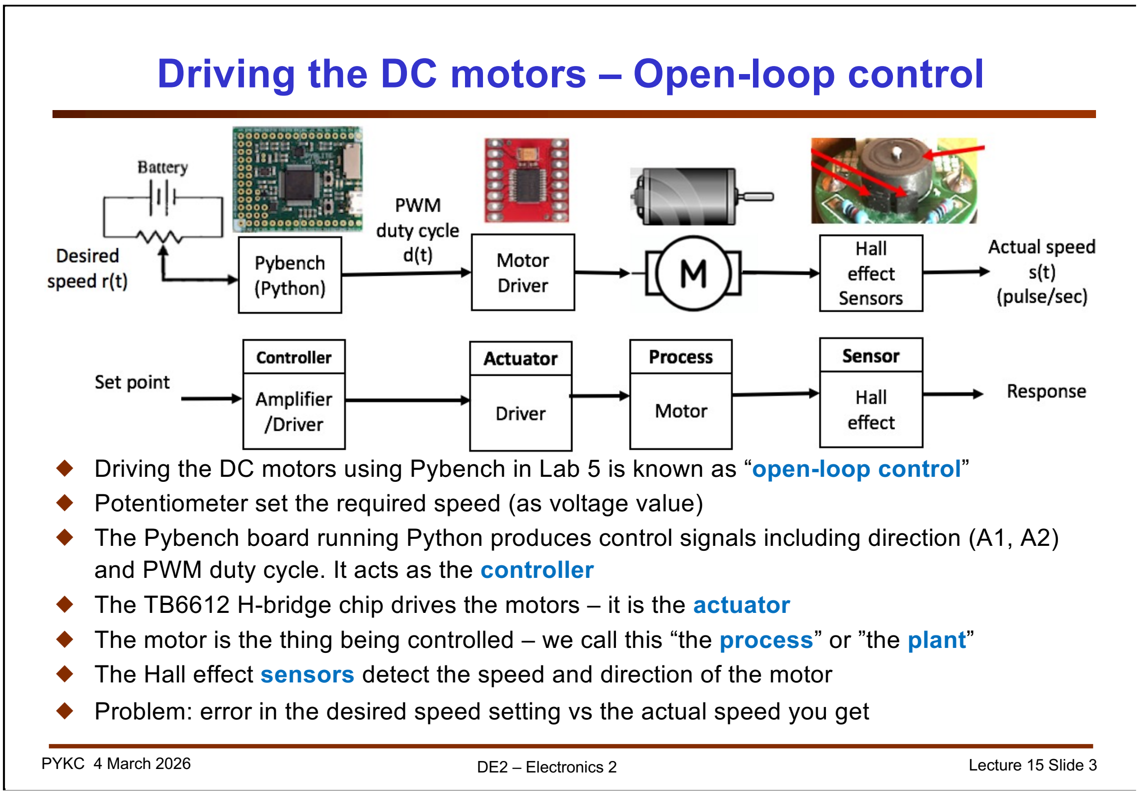
The concept of feedback is general. It is useful not only in our specific application, but it is applied in many other situations and environments. For example, you remember last year's Electronics 1 module, where you used an op amp to amplify microphone signal. We used feedback to fix the gain of the amplifier by choosing the resistor values. The fundamental principle used there is **feedback** – it is applied in the op amp circuit to fix the gain of the amplifier, no matter what the op amp characteristic is, provided that there is a **high gain** in the signal path.



What is control engineering? (a video)

Before I start this lecture, let us watch a short video explaining what is control engineering:

You can find this video on the link:
https://www.youtube.com/watch?v=Im88eVfkeBo&t=44s



Driving the DC motors – Open-loop control

- Driving the DC motors using Pybench in Lab 5 is known as "**open-loop control**"
- Potentiometer set the required speed (as voltage value)
- The Pybench board running Python produces control signals including direction (A1, A2) and PWM duty cycle. It acts as the **controller**
- The TB6612 H-bridge chip drives the motors – it is the **actuator**
- The motor is the thing being controlled – we call this "the **process**" or "the **plant**"
- The Hall effect **sensors** detect the speed and direction of the motor
- Problem: error in the desired speed setting vs the actual speed you get

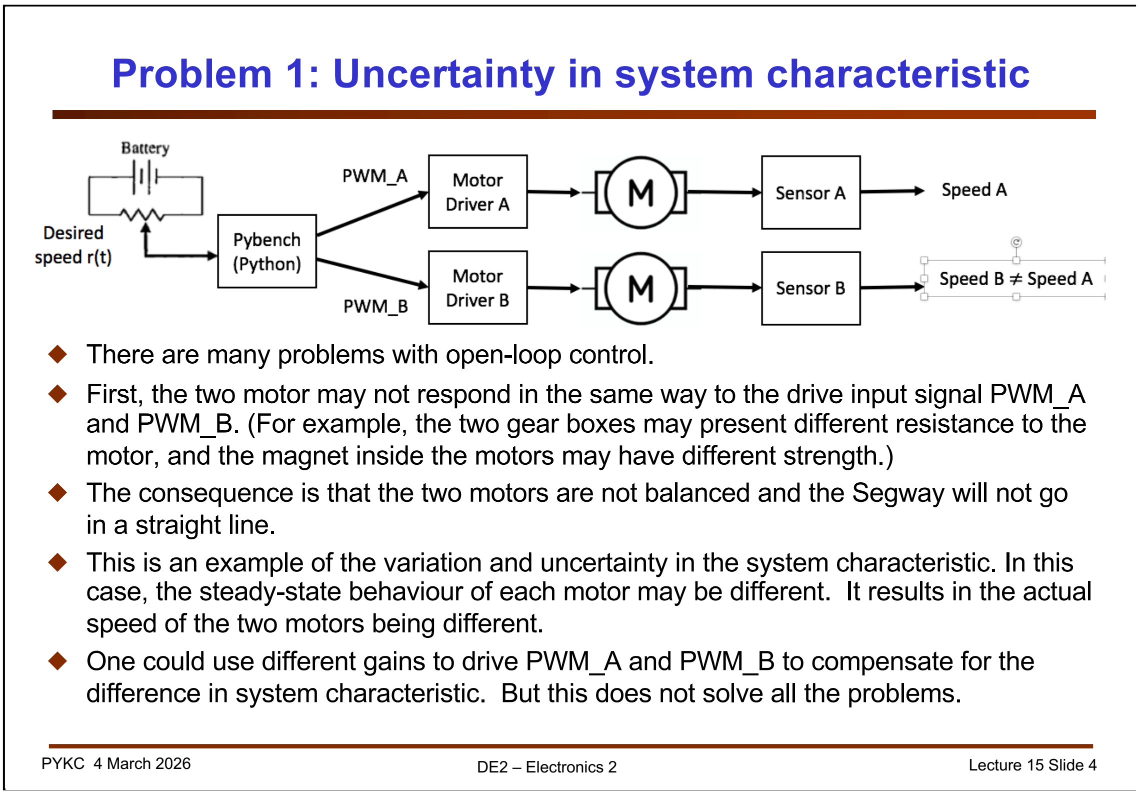
You have in the past been applying control to the DC motors, but **without feedback**. The potentiometer provide a desired motor speed. This is also known as the "**set point**". Here are what we did in Lab 5:

1. The Pybench board runs a program in Python in order to drive the motor through a PWM signal. The microcontroller running the Python program is the **controller**.
2. Then we use the TB6612 H-bridge chip to drive the motor. This is the **actuator**.
3. The output of the driver chip drives the motor directly.
4. Finally, we used the Hall effect sensors to measure the speed of the motor to provide an actual speed. This is the **sensor**.

The motor is what we want to control. In control terminology, the motor is known as the "**system**", the "**process**" or the "**plant**".

This system is subjected to "**open-loop**" control because the drive signal is independent of the output – there is no looping back of information to the drive input.

Finally, in our system, we are trying to control the speed of the motor. Therefore motor speed is the **control variable**.



Problem 1: Uncertainty in system characteristic

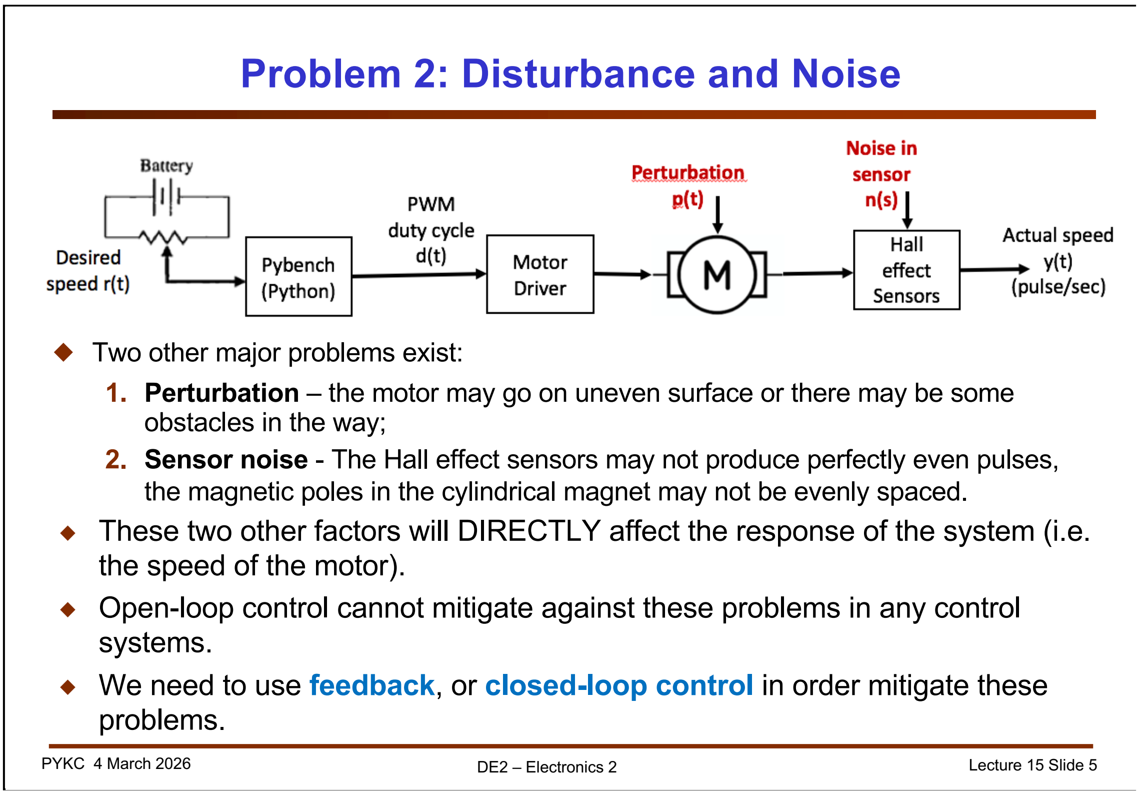
Battery
Desired speed r(t)
Pybench (Python)
PWM_A
PWM_B
Motor Driver A
Motor Driver B
M
M
Sensor A
Sensor B
Speed A
Speed B ≠ Speed A

- There are many problems with open-loop control.
- First, the two motor may not respond in the same way to the drive input signal PWM_A and PWM_B. (For example, the two gear boxes may present different resistance to the motor, and the magnet inside the motors may have different strength.)
- The consequence is that the two motors are not balanced and the Segway will not go in a straight line.
- This is an example of the variation and uncertainty in the system characteristic. In this case, the steady-state behaviour of each motor may be different. It results in the actual speed of the two motors being different.
- One could use different gains to drive PWM_A and PWM_B to compensate for the difference in system characteristic. But this does not solve all the problems.

Such a system has problems. As you have found out during the lab sessions, the two motors may be driven by the same PWM values, but the speed of the motors may be very different.

Open-loop control relies on **known system behaviour**. Any change in system behaviour (i.e. the process) will result in error in the outputs of the control variables.

However, open-loop control is not always bad. If the system characteristics is well defined and is not changing over time or under different operating conditions, open-loop control is easy to implement and will not subject to a major problem inherent in **feedback** or **closed-loop control**, which is the possibility of **instability**. Open-loop control will not cause instability, while feedback control could.



Problem 2: Disturbance and Noise

Battery
Desired speed r(t)
Pybench (Python)
PWM duty cycle d(t)
Motor Driver
Perturbation p(t)
M
Noise in sensor n(s)
Hall effect Sensors
Actual speed y(t) (pulse/sec)

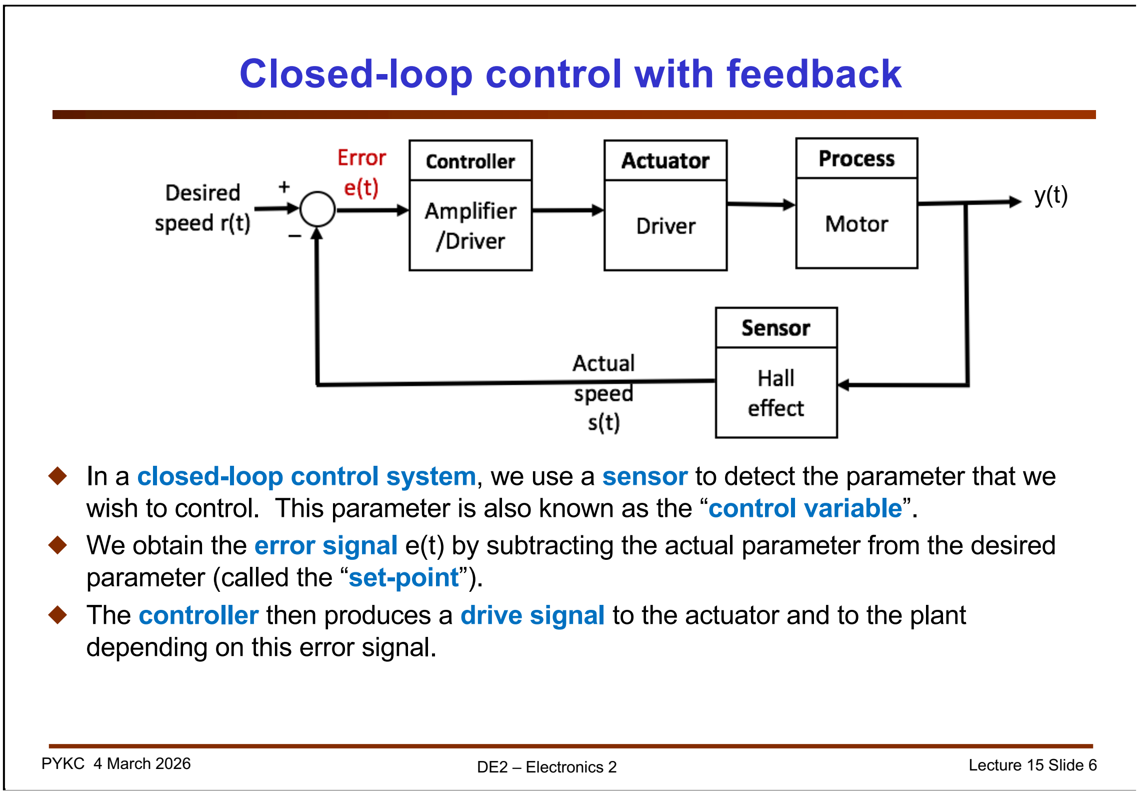
- Two other major problems exist:
  1. **Perturbation** – the motor may go on uneven surface or there may be some obstacles in the way;
  2. **Sensor noise** - The Hall effect sensors may not produce perfectly even pulses, the magnetic poles in the cylindrical magnet may not be evenly spaced.
- These two other factors will DIRECTLY affect the response of the system (i.e. the speed of the motor).
- Open-loop control cannot mitigate against these problems in any control systems.
- We need to use **feedback**, or **closed-loop control** in order mitigate these problems.

There are two further major problems with open-loop control.

The system (i.e. the motor) may experience disturbances or **perturbations**. Furthermore, the sensor may have **noise** associated with it.

These two factors p(t) and n(t) will affect the output behaviour.

A **closed-loop system** has the potential of mitigating against these undesirable factors.



Closed-loop control with feedback

- In a **closed-loop control system**, we use a **sensor** to detect the parameter that we wish to control. This parameter is also known as the "**control variable**".
- We obtain the **error signal** e(t) by subtracting the actual parameter from the desired parameter (called the "**set-point**").
- The **controller** then produces a **drive signal** to the actuator and to the plant depending on this error signal.

Here is a generic **closed-loop control system** employing **feedback**.

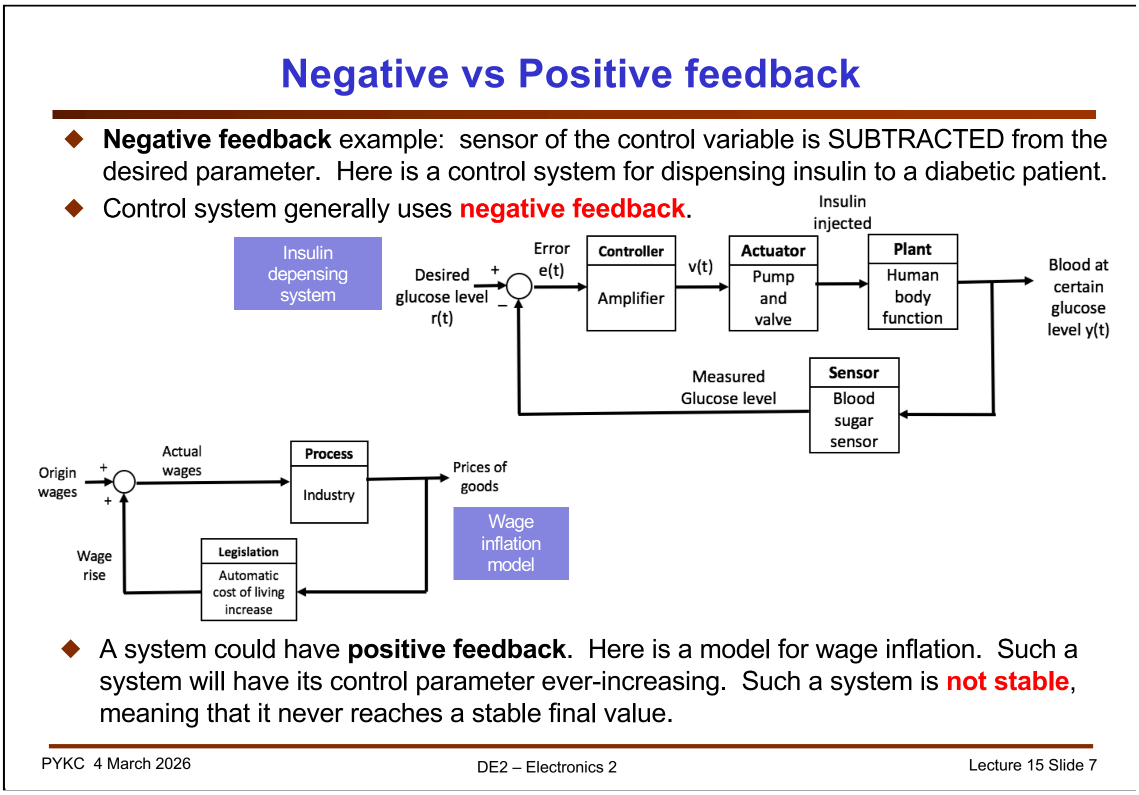
The **control variable** is detected using the **sensor**, producing the sensed value s(t). The sensed value is compared with the set point r(t) (i.e. the reference value) to produce the error signal **e(t)**.

The error signal e(t) is used as input to the **controller** (such as an amplifier) in order to provide the signal for the actuator which drives the system.

The key to this feedback control system is that the system is driven by the controller that responses to the difference between what is desired r(t) and what actually happens at the output y(t) measured by the sensor.

The purpose of this loop back (closed-loop feedback) is that by choosing or designing a proper controller, we can:

1. Provide **regulation function**, i.e. make the control variable tracks the set point.
2. Make the closed-loop system **immune to variations** in the process (e.g. motor characteristics).
3. Make the closed-loop system **less prone to disturbances** or perturbations and reduce the effect of noise.
4. **Change the system dynamic** behaviour such as its step response to what you want.
5. **Make** an inherently unstable system (such as the Segway) **stable**.



Negative vs Positive feedback

- **Negative feedback** example: sensor of the control variable is SUBTRACTED from the desired parameter. Here is a control system for dispensing insulin to a diabetic patient.
- Control system generally uses **negative feedback**.

Insulin depensing system

Desired glucose level r(t)
+
_
Error e(t)
Controller
Amplifier
v(t)
Actuator
Pump and valve
Insulin injected
Plant
Human body function
Blood at certain glucose level y(t)
Measured Glucose level
Sensor
Blood sugar sensor

Origin wages
+
+
Actual wages
Process
Industry
Prices of goods
Legislation
Automatic cost of living increase
Wage rise

Wage inflation model

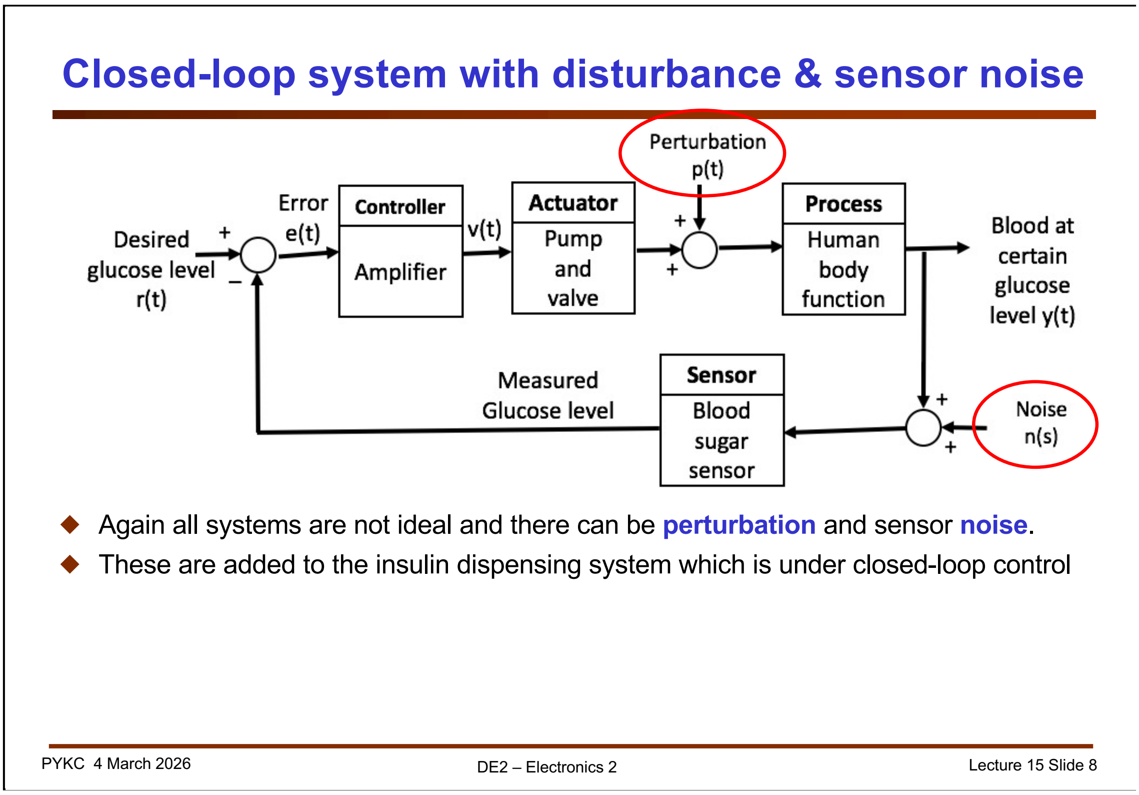
- A system could have **positive feedback**. Here is a model for wage inflation. Such a system will have its control parameter ever-increasing. Such a system is **not stable**, meaning that it never reaches a stable final value.

Note that we usually use **negative feedback** in order to control a system. We always subtract the output variable (or some form of feedback value) from the set point, not add.

If you add instead of subtract, the system will go unstable or goes to infinity – it would not settle to a final value. This is called **positive feedback.**

Show here is a **positive feedback** system modelling wage inflation – that's why in a normal economy, the wages always increase and will not stay at a steady-state level.

We only consider **negative feedback** in this module.

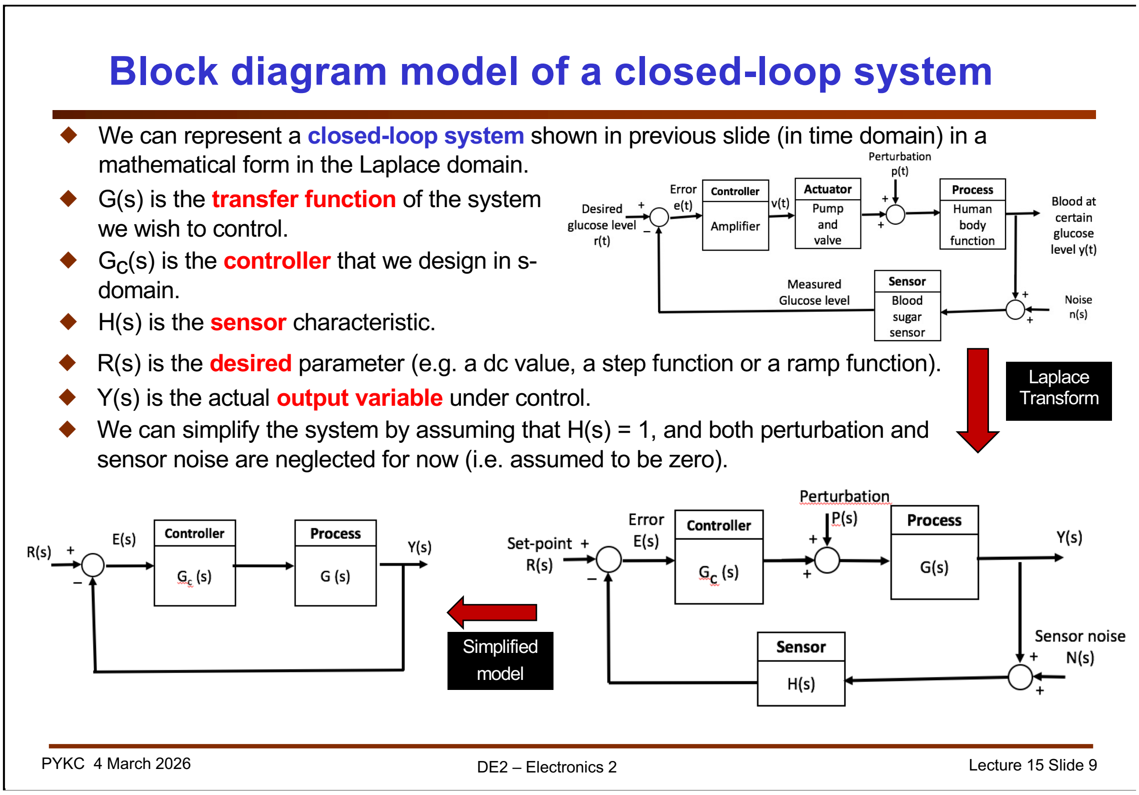


Closed-loop system with disturbance & sensor noise

- Again all systems are not ideal and there can be **perturbation** and sensor **noise**.
- These are added to the insulin dispensing system which is under closed-loop control

We must now add two other real-life factors into our closed-loop system model.

Here we add the **perturbation** or disturbance **p(t)** just before the process. Finally we add the **noise** signal **n(t)** injected into the sensor. These will have impact to the final output y(t) (as we will see later).



Block diagram model of a closed-loop system

- We can represent a **closed-loop system** shown in previous slide (in time domain) in a mathematical form in the Laplace domain.
- G(s) is the **transfer function** of the system we wish to control.
- $G_C(s)$ is the **controller** that we design in s-domain.
- H(s) is the **sensor** characteristic.
- R(s) is the **desired** parameter (e.g. a dc value, a step function or a ramp function).
- Y(s) is the actual **output variable** under control.
- We can simplify the system by assuming that H(s) = 1, and both perturbation and sensor noise are neglected for now (i.e. assumed to be zero).

So far, we have drawn our system as blocks of functions. In order to do analysis on such a system, we must model it mathematically. That's why you need to learn modeling of systems earlier.

The technique we use to analyse system in order to design a controller for it in a closed-loop fashion is through **Laplace transform**.

Here we assume that the actuator and the process (or system, or plant) together has a **system transfer function** in the Laplace domain of G(s).

The **sensor** in the feedback path may have a transfer function of H(s).

Our goal is to **design a controller** $G_C(s)$ which, when we close the loop, will make the system behaves in the way we want.

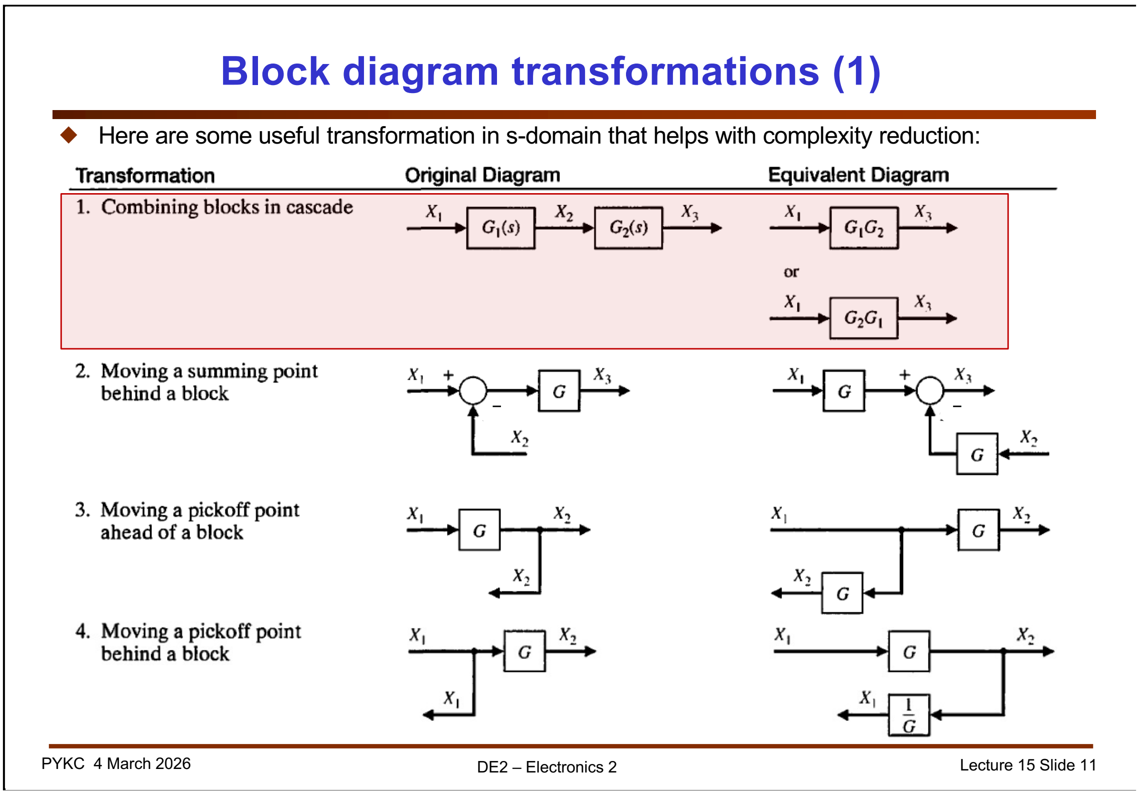
We often simplify the system by assuming that it has no disturbance, and the sensor has no noise (P(s) = N(s) = 0), and finally H(s) =1. Then we get the simplified model as shown. In this model, we basically have the process G(s) and the controller $G_C(s)$ which we need to design, and connected in a negative feedback configuration.

A video on open- & closed- loop systems

Before I move on, here is good YouTube video comparing the open-loop and the closed-loop system.

You can find this video on the link:
https://youtu.be/zjDM1qZaJ6Y

(See course webpage.)



Block diagram transformations (1)

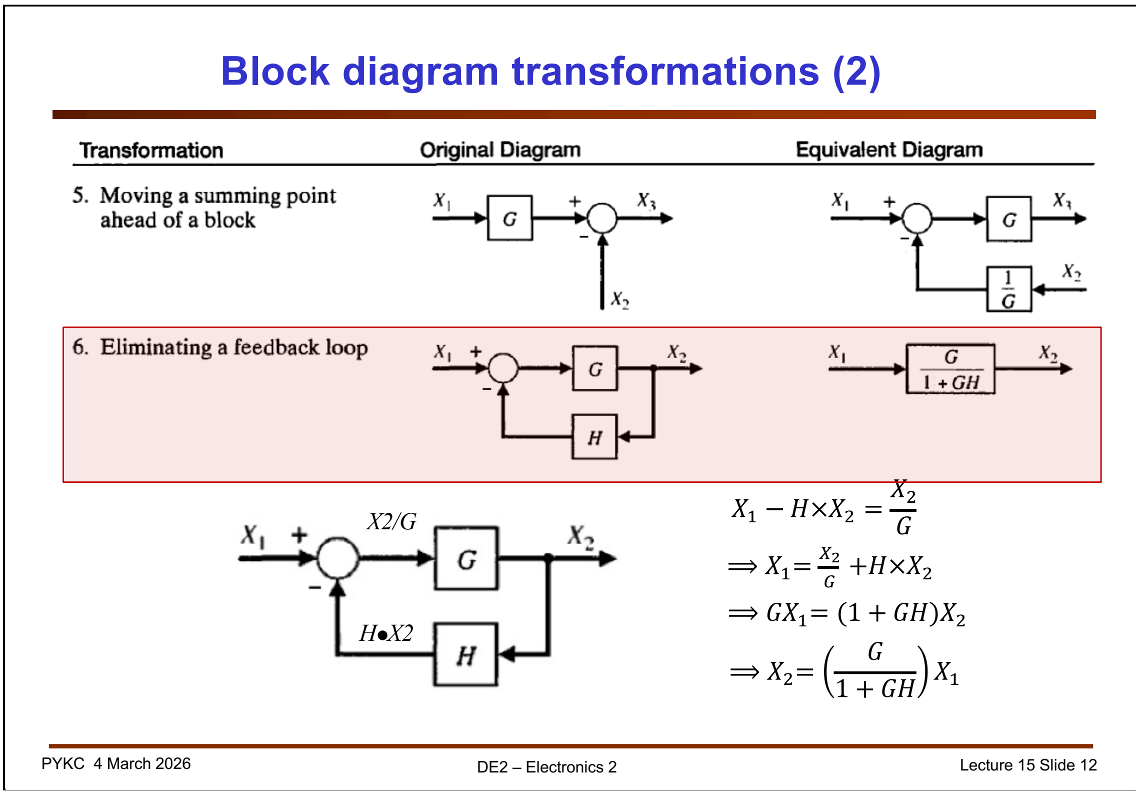
- Here are some useful transformation in s-domain that helps with complexity reduction:

Transformation	Original Diagram	Equivalent Diagram
1. Combining blocks in cascade		or
2. Moving a summing point behind a block		
3. Moving a pickoff point ahead of a block		
4. Moving a pickoff point behind a block		

Before we move on, let use consider how to take the block diagram in the Laplace domain and perform some transformations in order to achieve a simplified diagram and to derive the system level transfer function.

There are 6 transformations. However, transforms 1 and 6, highlighted in red, are the IMPORTANT ONES.

Here is shown transforms 1 to 4. They are pretty obvious.



Block diagram transformations (2)

Transformation	Original Diagram	Equivalent Diagram
5. Moving a summing point ahead of a block		
6. Eliminating a feedback loop		

$$X_1 - H\times X_2 = \frac{X_2}{G}$$

$$\Rightarrow X_1 = \frac{X_2}{G} + H\times X_2$$

$$\Rightarrow GX_1 = (1+GH)X_2$$

$$\Rightarrow X_2 = \left(\frac{G}{1+GH}\right)X_1$$

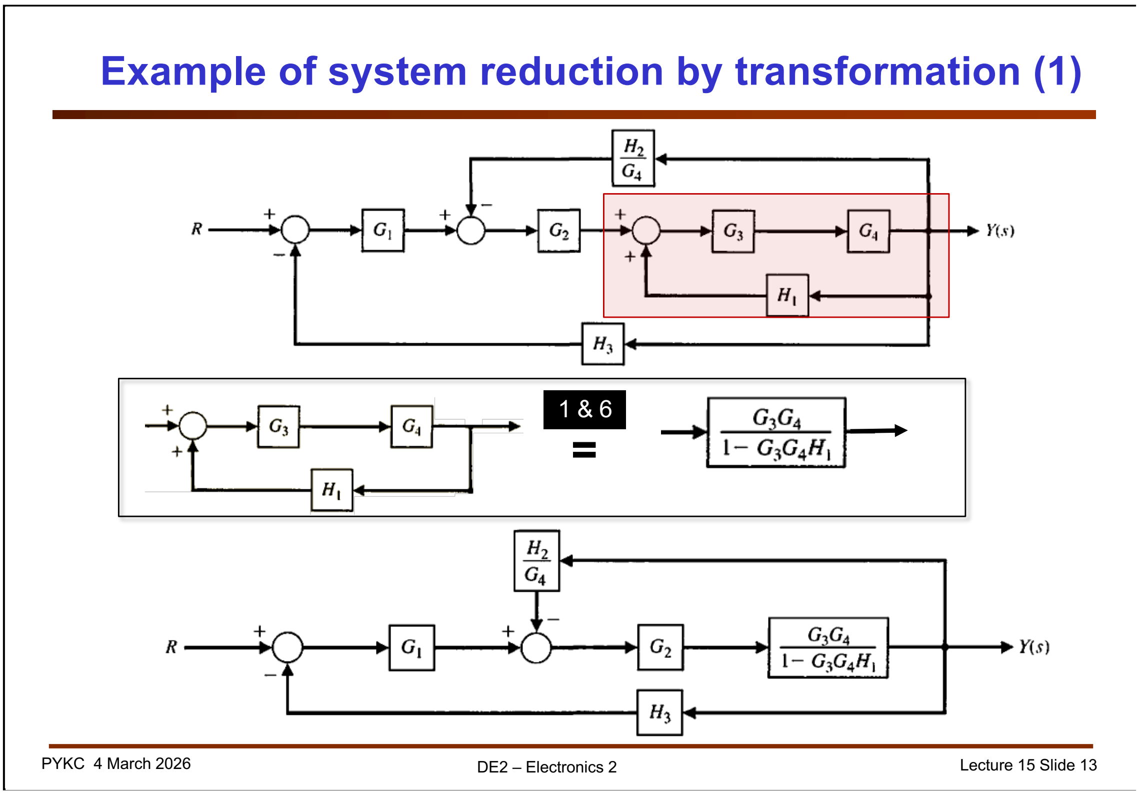
The most important transform is number 6 – transforming a generic feedback loop into a transfer function.

What I have shown here is the derivation of this transformation. Make sure that you are able to do the derivation yourself.

The transfer function for a feedback system with forward gain G(s) and feedback gain H(s) is:

$$\frac{G(s)}{1+G(s)H(s)}$$

This equation is one of the very few that are worth memorizing!

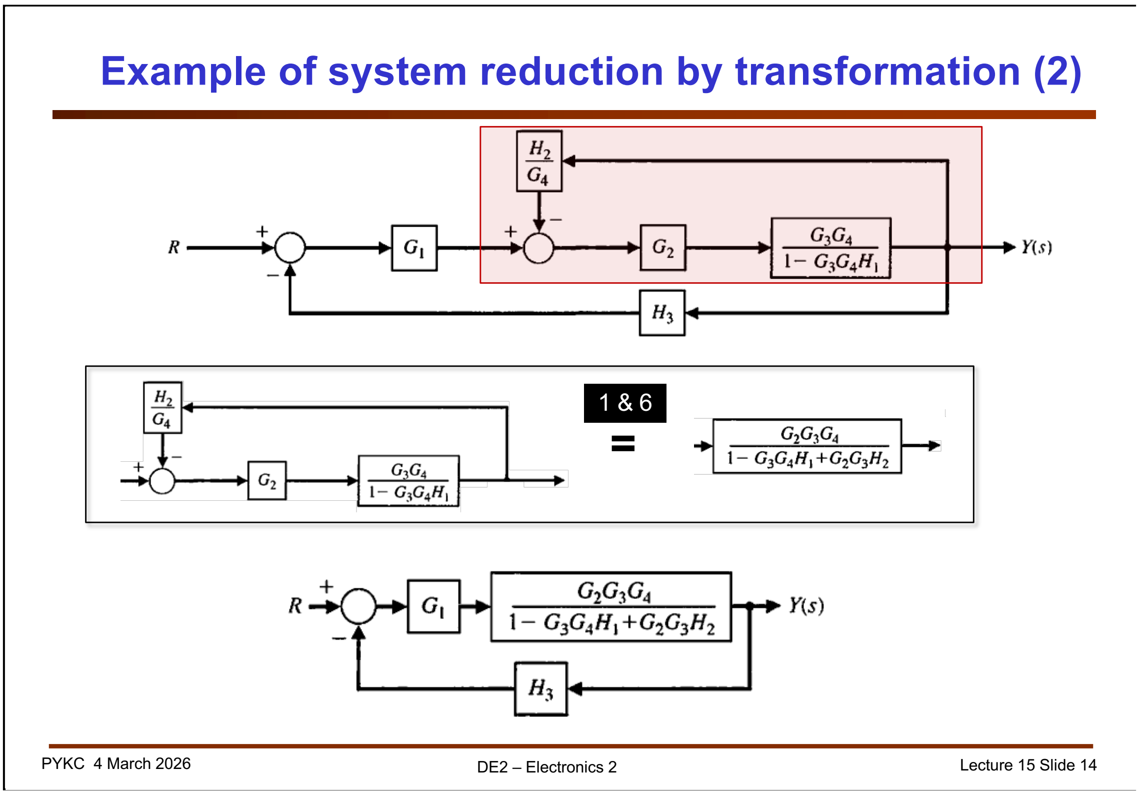


Example of system reduction by transformation (1)

PYKC 4 March 2026 DE2 – Electronics 2 Lecture 15 Slide 13

Now let me go through step-by-step how a complex system block diagram can be reduced to a transfer function in the s-domain using an example.

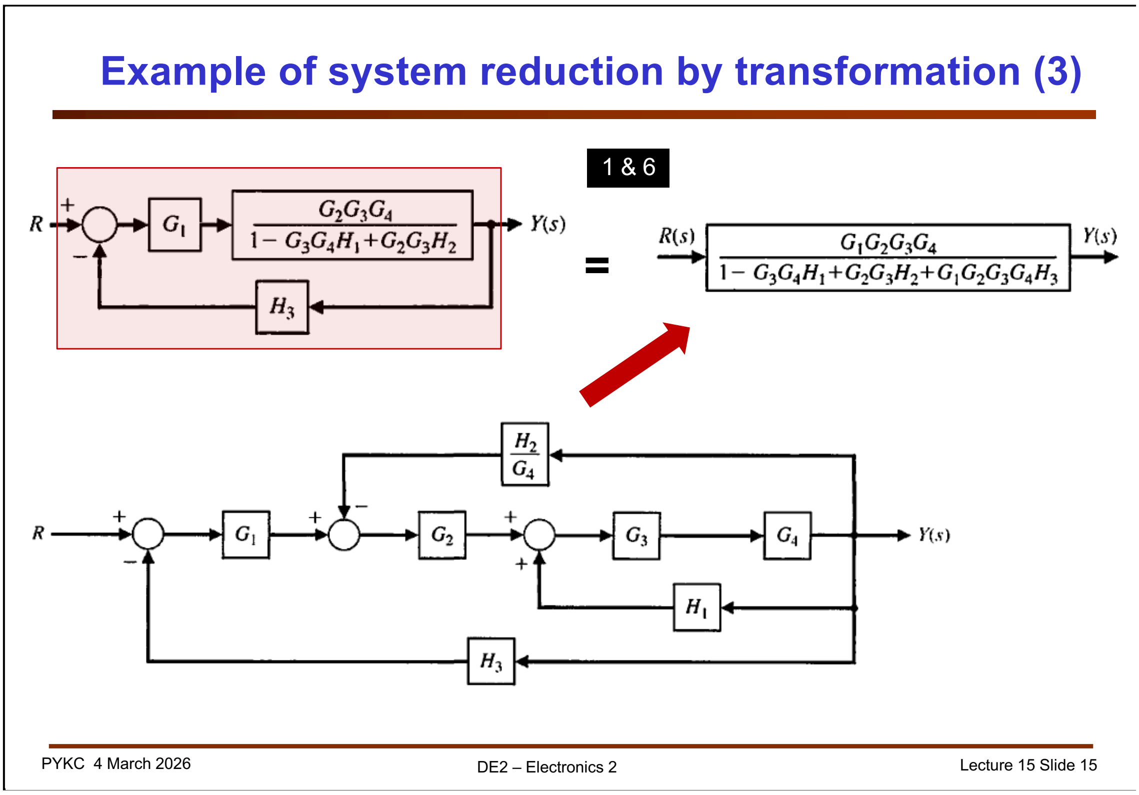
Make sure that you can follow each step.



Example of system reduction by transformation (2)

H_2/G_4
+
−
R
+
−
G_1
G_2
$\frac{G_3G_4}{1-G_3G_4H_1}$
$Y(s)$
H_3
1 & 6
=
$\frac{G_2G_3G_4}{1-G_3G_4H_1+G_2G_3H_2}$

Keep following the steps.



Example of system reduction by transformation (3)

1 & 6

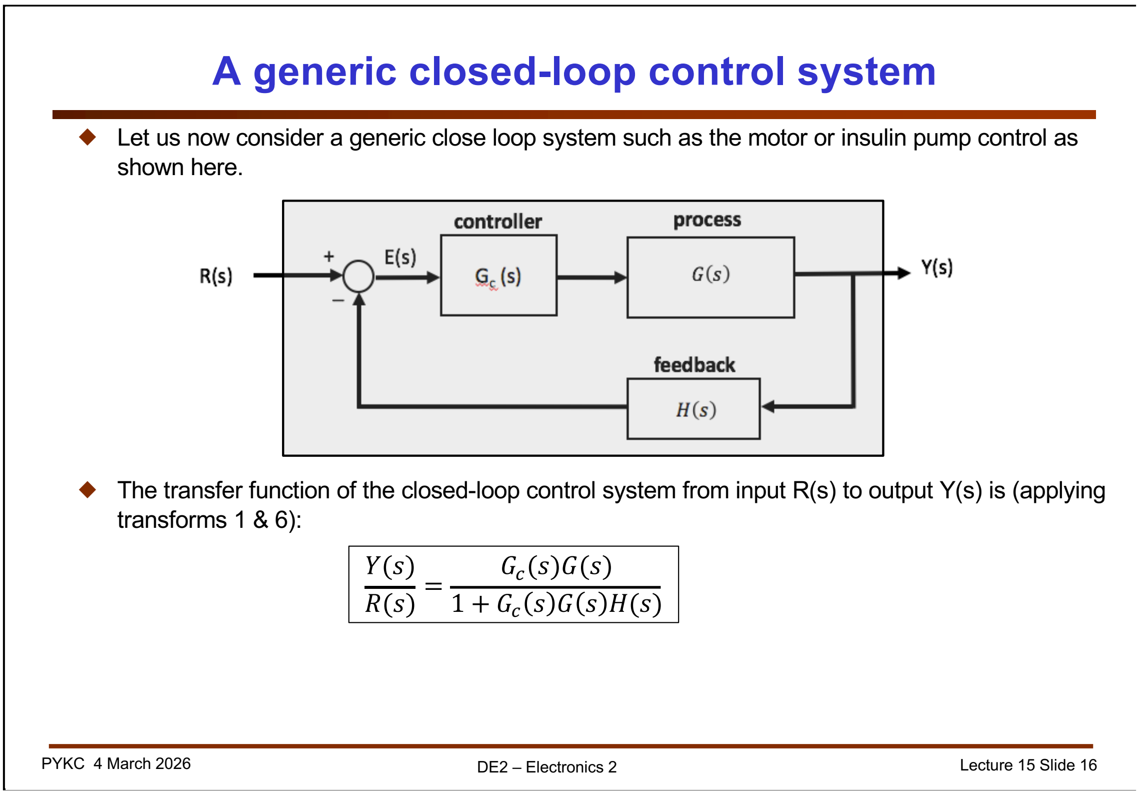
$$\frac{G_2G_3G_4}{1-G_3G_4H_1+G_2G_3H_2}$$

=

$$R(s) \rightarrow \boxed{\frac{G_1G_2G_3G_4}{1-G_3G_4H_1+G_2G_3H_2+G_1G_2G_3G_4H_3}} \rightarrow Y(s)$$

Here we show how the complex block diagram is reduced to the expression shown. Each term in the expression Gx and Hx is a transfer function block in the original diagram.

In practice we will not be dealing with any system that is as complex as the one shown here. Nevertheless, it is worth learning the technique to do such simplification for ANY system.



A generic closed-loop control system

- Let us now consider a generic close loop system such as the motor or insulin pump control as shown here.

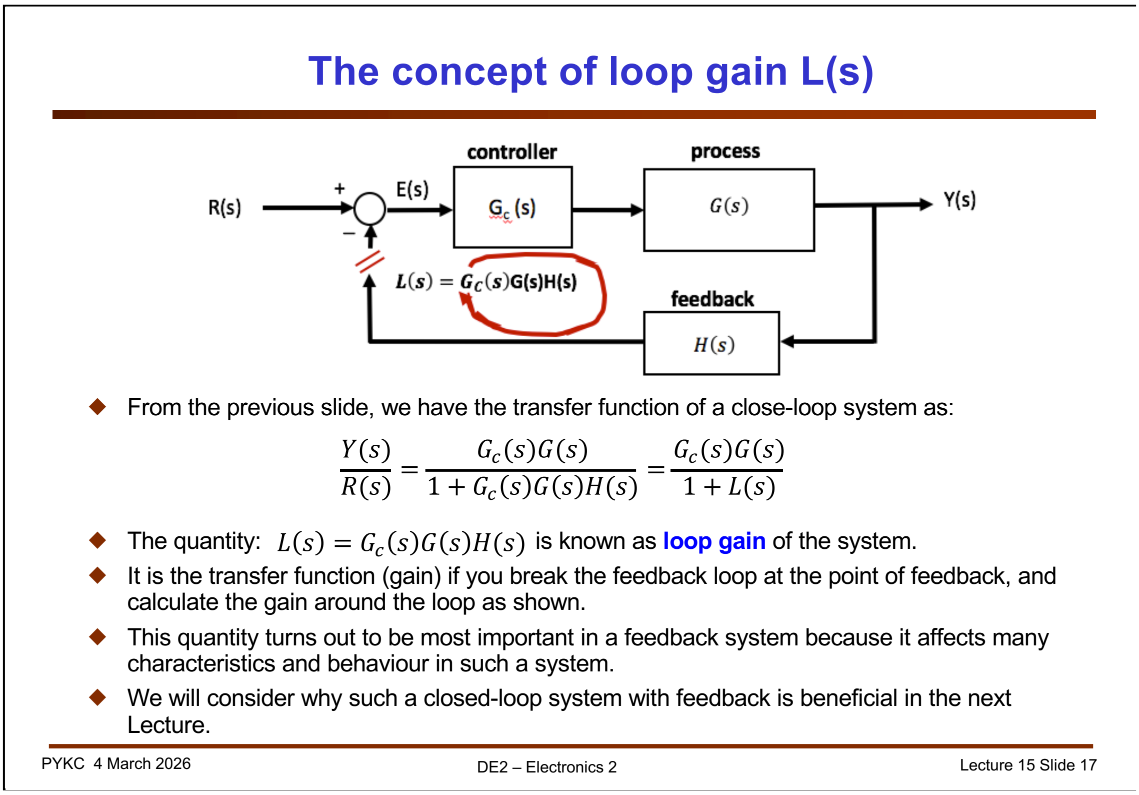
- The transfer function of the closed-loop control system from input R(s) to output Y(s) is (applying transforms 1 & 6):

$$\boxed{\frac{Y(s)}{R(s)} = \frac{G_c(s)G(s)}{1 + G_c(s)G(s)H(s)}}$$

Let us apply these transformation technique to our closed-loop system of the motor or the insulin pump.

We can easily derive (using 1 and 6) that the transfer function from R(s) to Y(s) is as shown here.

We will see in the next lecture why this is significant.



The concept of loop gain L(s)

- From the previous slide, we have the transfer function of a close-loop system as:

$$\frac{Y(s)}{R(s)} = \frac{G_c(s)G(s)}{1 + G_c(s)G(s)H(s)} = \frac{G_c(s)G(s)}{1 + L(s)}$$

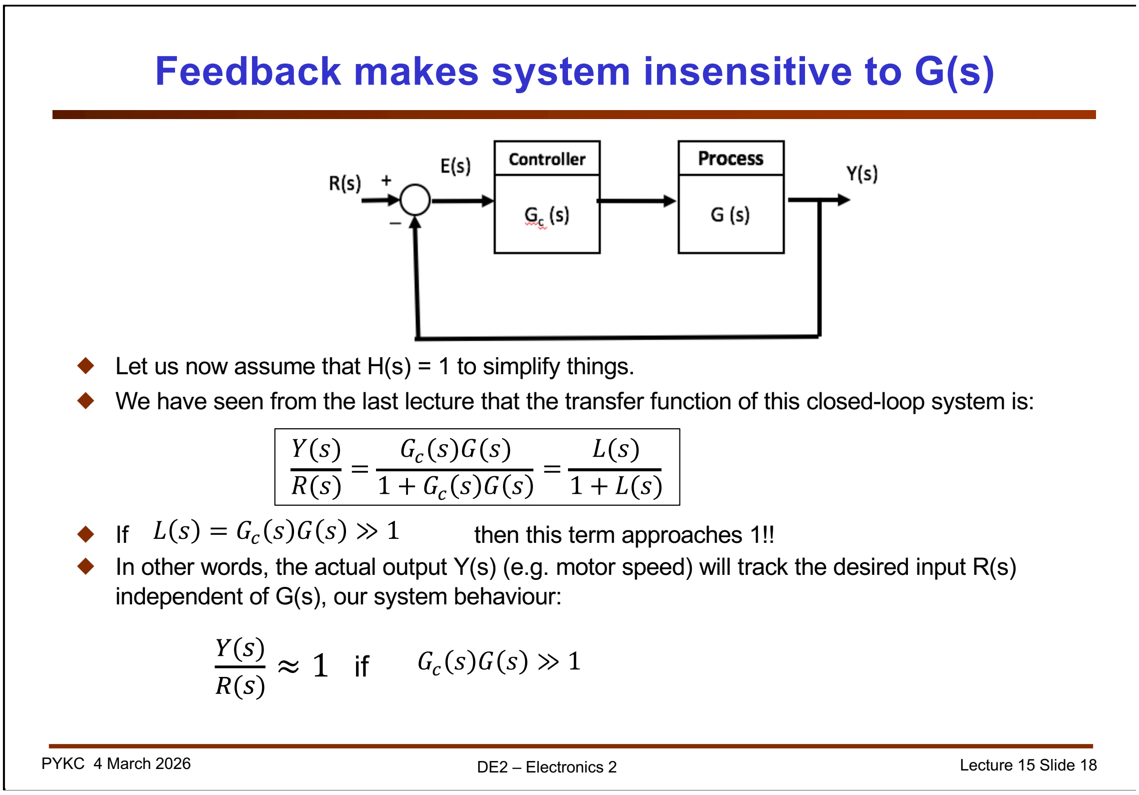
- The quantity: $L(s) = G_c(s)G(s)H(s)$ is known as **loop gain** of the system.
- It is the transfer function (gain) if you break the feedback loop at the point of feedback, and calculate the gain around the loop as shown.
- This quantity turns out to be most important in a feedback system because it affects many characteristics and behaviour in such a system.
- We will consider why such a closed-loop system with feedback is beneficial in the next Lecture.

Loop gain is an important concept.

If we break the feedback loop at just BEFORE we subtract the feedback signal from the desired input R(s), the gain of the system around the loop (without feedback) is known as **loop gain**. In the generic system shown here, the loop gain:

$$L(s) = G_C(s)\ G(s)\ H(s).$$

You will find this quantity popping up all over the place in any feedback systems. In many cases, the higher the loop gain, the “better” is the control system. However, increasing the loop gain also make the system more prone to **instability** (i.e. it can become oscillatory).



Feedback makes system insensitive to G(s)

R(s)
+
−
E(s)
Controller
G_c (s)
Process
G (s)
Y(s)

- Let us now assume that H(s) = 1 to simplify things.
- We have seen from the last lecture that the transfer function of this closed-loop system is:

$$\boxed{\frac{Y(s)}{R(s)} = \frac{G_c(s)G(s)}{1 + G_c(s)G(s)} = \frac{L(s)}{1 + L(s)}}$$

- If $L(s) = G_c(s)G(s) \gg 1$ then this term approaches 1!!
- In other words, the actual output Y(s) (e.g. motor speed) will track the desired input R(s) independent of G(s), our system behaviour:

$$\frac{Y(s)}{R(s)} \approx 1 \quad \text{if} \quad G_c(s)G(s) \gg 1$$

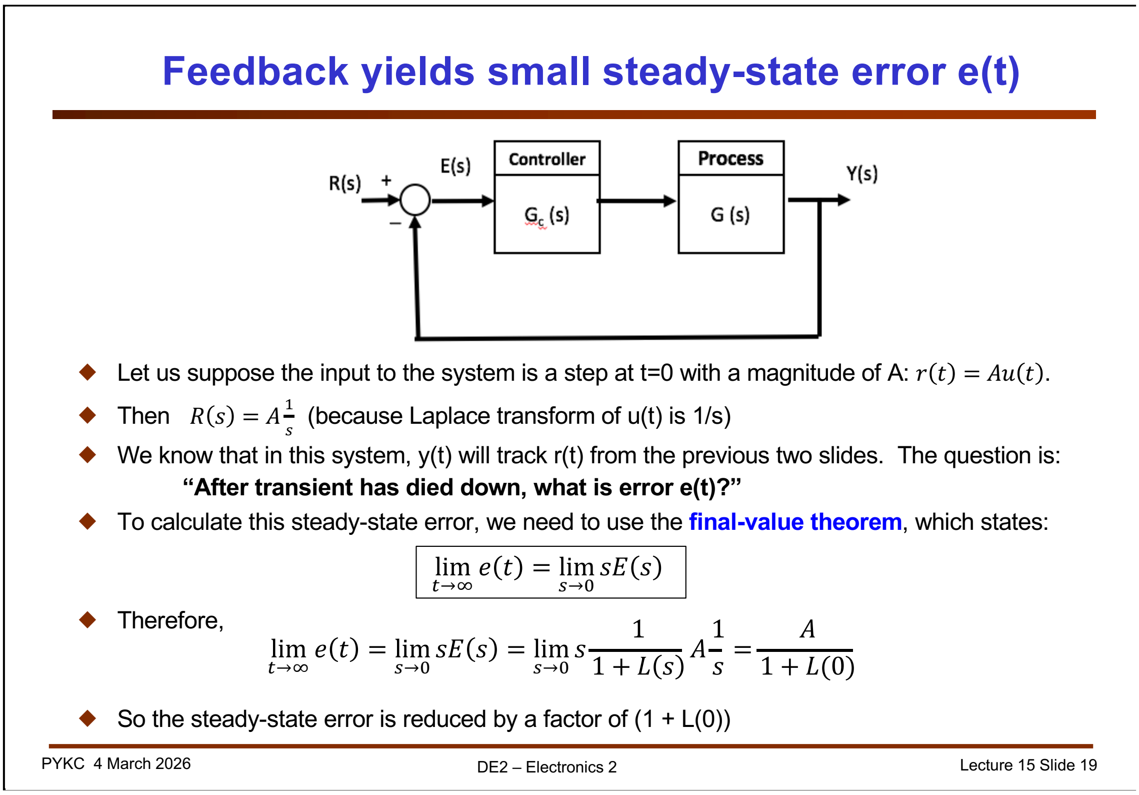
Here is the feedback system we used in the last lecture. The controller is under our control. The purpose is that with the negative feedback, we can make the system behaviour in a better way than an open-loop system.

If you derive the transfer function from R(s) to Y(s), you find that the transfer function of this closed-loop system becomes:

$$\boxed{\frac{Y(s)}{R(s)} = \frac{G_c(s)G(s)}{1 + G_c(s)G(s)} = \frac{L(s)}{1 + L(s)}}$$

Provided that the loop gain L(s) is large compared to 1, this give us a value of 1 (for all s values). That means the output Y(s) is tracking the set point (desired control variable value) R(s) because of this feedback loop.

This is an important result. The closed-loop system behaviour is now **INDEPENDENT** of G(s), the system we are controlling, as long as the loop gain L(s) = Gc(s) G(s) is large as compared to 1.


R(s)
+
−
E(s)
Controller
G_c (s)
Process
G (s)
Y(s)

Feedback yields small steady-state error e(t)

- Let us suppose the input to the system is a step at t=0 with a magnitude of A: $r(t) = Au(t)$.
- Then $R(s) = A\frac{1}{s}$ (because Laplace transform of u(t) is 1/s)
- We know that in this system, y(t) will track r(t) from the previous two slides. The question is:
  "After transient has died down, what is error e(t)?"
- To calculate this steady-state error, we need to use the **final-value theorem**, which states:

$$\boxed{\lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s)}$$

- Therefore,

$$\lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s) = \lim_{s\to 0} s\frac{1}{1+L(s)} A\frac{1}{s} = \frac{A}{1+L(0)}$$

- So the steady-state error is reduced by a factor of (1 + L(0))

Now let us consider the error e(t) or E(s) in the s-domain. Ideally you want Y(s) to be exactly R(s), that is, what we set as desired, is what we get.
However, in any systems, there may be an error.

E(s) = R(s) – Y(s) = R(s) – E(s) G_c(s)G(s) = R(s) – E(s) L(s), where L(s)=loop gain

Hence, E(s) = R(s) x [1/1+L(s)].

Let us now consider the case that the input r(t) is a step function with a step value of A, i.e. r(t) = A u(t) at t = 0.
This is modelling the case that you may be controlling a robot arm to move from one point to any other at time = 0, or changing the motor speed from one to an other.

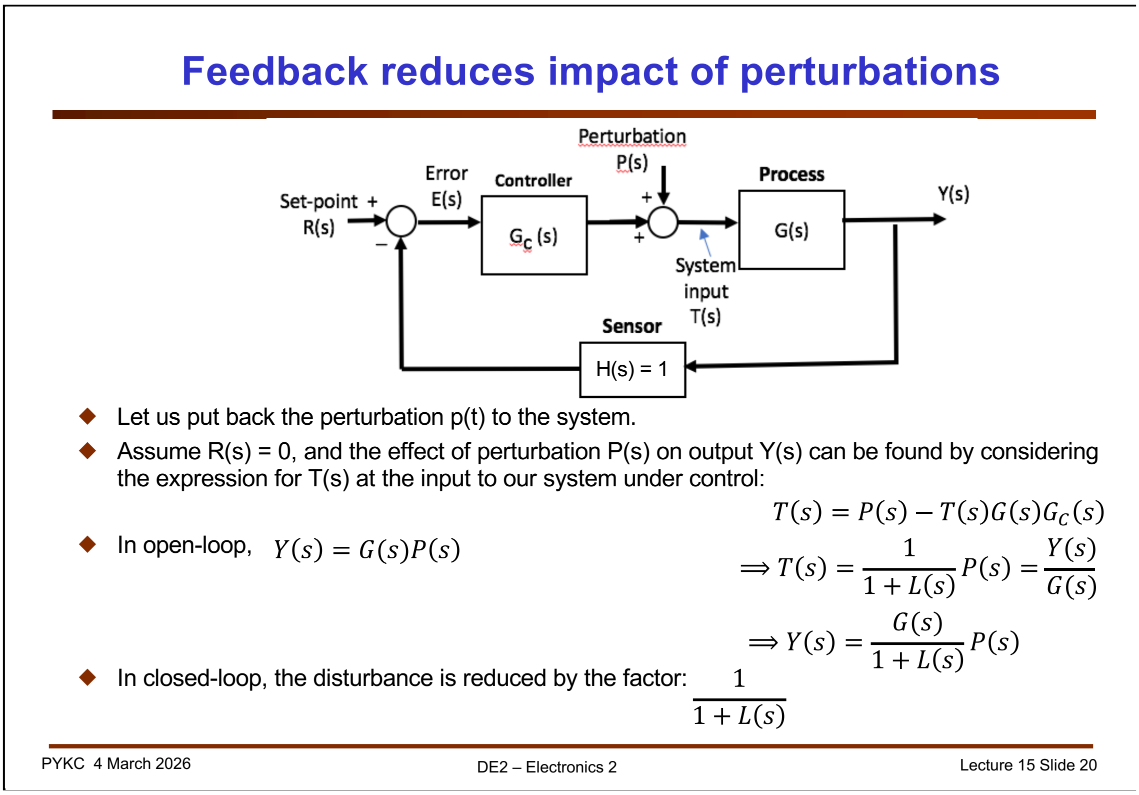
Remember from earlier lecture that the Laplace transform of A u(t) is:

$$R(s) = L\{Au(t)\} = A\frac{1}{s}$$

The question is: after such as step, what will the output eventually settle down to? In order to answer this question, we have to use some important theorem, known as the final-value theorem, which states:

$$\boxed{\lim_{t\to\infty} e(t) = \lim_{s\to 0} sE(s)}$$

I don't want to prove this, but it allows you to computer the final value of a signal using only the s-domain equations (i.e. the transfer function). As shown in the slide here, the final value of the error e(t) = A / (1 + L(0)), where L(0) is the loop gain at s=0, or the DC gain of L(s).
In other words, the steady-state error (when s=0) is A reduced by a factor of (1+L(0))



Feedback reduces impact of perturbations

- Let us put back the perturbation p(t) to the system.
- Assume R(s) = 0, and the effect of perturbation P(s) on output Y(s) can be found by considering the expression for T(s) at the input to our system under control:

$$T(s) = P(s) - T(s)G(s)G_C(s)$$

$$\Rightarrow T(s) = \frac{1}{1+L(s)}P(s) = \frac{Y(s)}{G(s)}$$

$$\Rightarrow Y(s) = \frac{G(s)}{1+L(s)}P(s)$$

- In open-loop, $Y(s) = G(s)P(s)$
- In closed-loop, the disturbance is reduced by the factor: $\dfrac{1}{1+L(s)}$

Let us now consider the impact of closed-loop on perturbations P(s) or p(t).

This is added as shown in the block diagram.
We are interested in the relationship between Y(s), the output, and the perturbation P(s).
To work this out, there is a good trick – always evaluate at the signal point AFTER the summer. In this case, it is the input to the system G(s). Let us call this T(s).

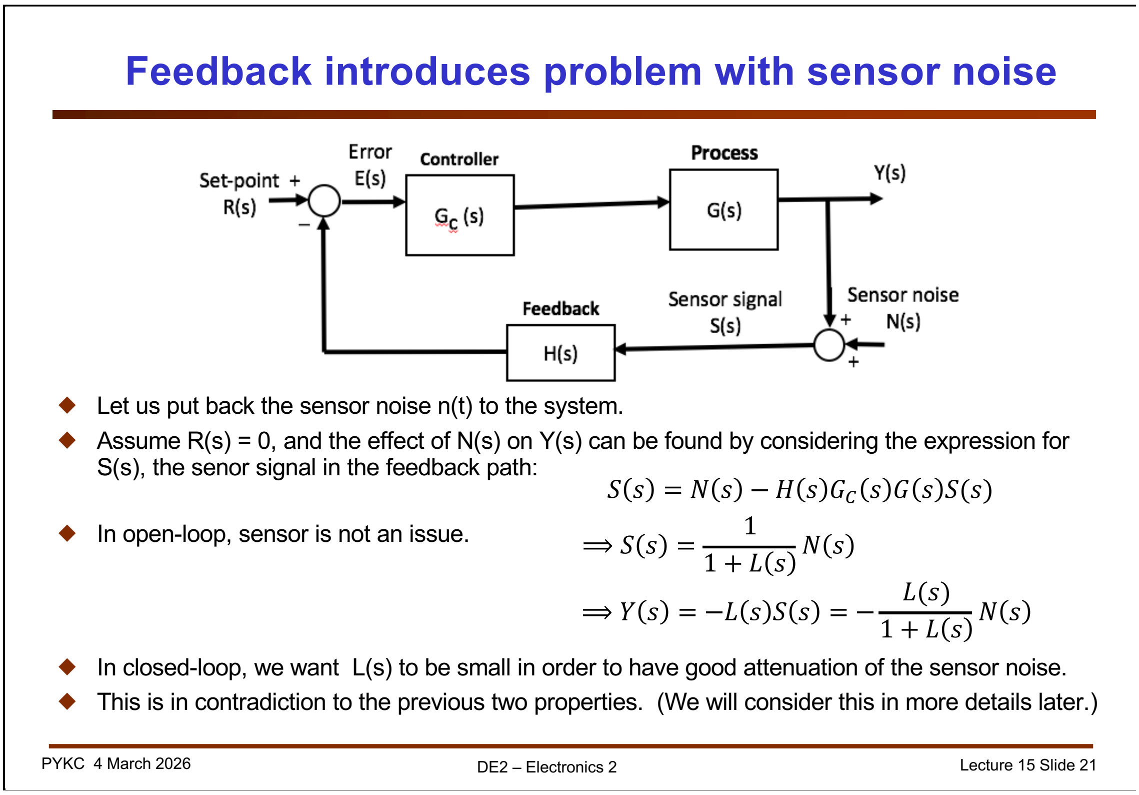
Some simple algebra yield the answer:

$$Y(s) = \frac{G(s)}{1+L(s)}P(s)$$

If we did not have closed-loop, but have a simple open-loop system, then Y(s) = G(s)P(s). That is, the perturbation is passed to the output through our system as-is.

However, putting the system G(s) in this feedback loop, we reduce this effect by the factor $\dfrac{1}{1+L(s)}$. Note that this factor comes up all the time!

If loop gain L(s) is relatively large as compared to 1, then the perturbation is reduced by the factor of L(s) at all value of s.



Feedback introduces problem with sensor noise

- Let us put back the sensor noise n(t) to the system.
- Assume R(s) = 0, and the effect of N(s) on Y(s) can be found by considering the expression for S(s), the senor signal in the feedback path:

$$S(s) = N(s) - H(s)G_C(s)G(s)S(s)$$

- In open-loop, sensor is not an issue.

$$\Rightarrow S(s) = \frac{1}{1+L(s)}N(s)$$

$$\Rightarrow Y(s) = -L(s)S(s) = -\frac{L(s)}{1+L(s)}N(s)$$

- In closed-loop, we want L(s) to be small in order to have good attenuation of the sensor noise.
- This is in contradiction to the previous two properties. (We will consider this in more details later.)

Now let us put into the system the sensor noise n(t) or N(s).
With some manipulation, we found that:

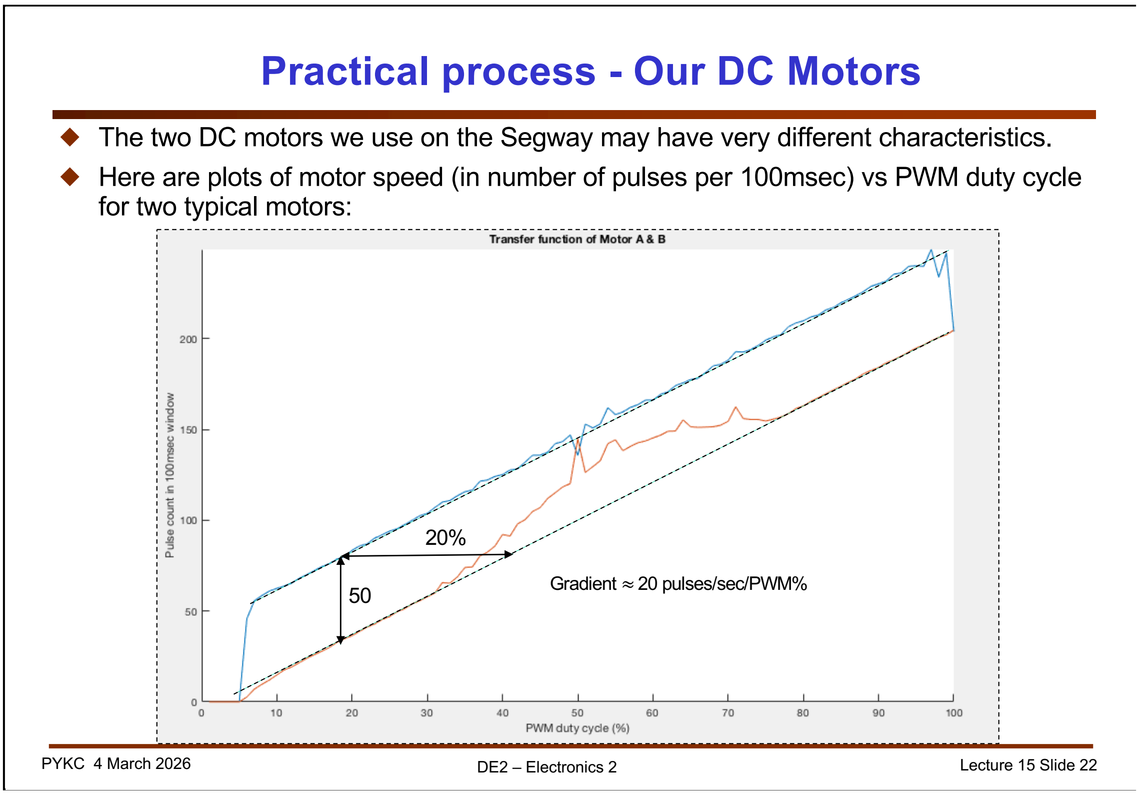
$$Y(s) = -L(s)S(s) = -\frac{L(s)}{1+L(s)}N(s)$$

Unfortunately the output is VERY MUCH affected by the noise. If L(s) is large, as we have assumed in previous slides, then N(s) is more or less passed to the output Y(s).

Fortunately, in many practical systems, N(s), the noise, generally is at high frequency. If G(s) or Gc(s) has low gain at high frequency (lowpass filter), L(s) << 1 at high s, then,

$$Y(s) \approx -L(s)N(s)$$

Since L(s) << 1, N(s) can be suppressed.



Practical process - Our DC Motors

- The two DC motors we use on the Segway may have very different characteristics.
- Here are plots of motor speed (in number of pulses per 100msec) vs PWM duty cycle for two typical motors:

Gradient ≈ 20 pulses/sec/PWM%

So far, we have been considering the theoretical foundation of a feedback system. Let us know consider something that is both real and that you are familiar with.

Plotted here is the speed of two typical DC motors used for our project. These are motors from the mini-Segway that I have been personally using.
The plot is the pulse count from the hall effect sensor (in a 100ms window) vs the PWM duty cycle driving each motor.

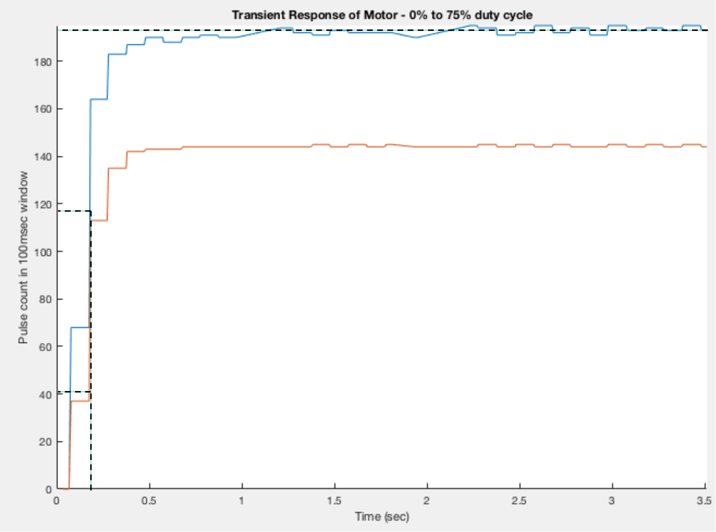
We can make the following observations:

1. The characteristic is relatively linear – that is speed is proportional to PWM value.
2. The two motors have more or less the same gradient of around 20 pulses/sec/PWM%. In other words, if you increase PWM value by 10, you can expect the pulse count to increase by 20 in the duration of 1sec.
3. The two motors are not matched. The BLUE motor is consistently faster (offset) from the RED motor by 50 pulses per second. Or in order for the two motor to go at the same speed, the RED motor needs to be driven with an extra 20% in PWM!
4. Both motors do not start turning unless the PWM value exceeds around 6 or 7%.
5. Something funny happens to the RED motor in mid-range. Not sure why!

This characteristic really demonstrate why we may need to use feedback control in order to make the motor drive at desired speed independ its own characteristics.

Step response of the motor

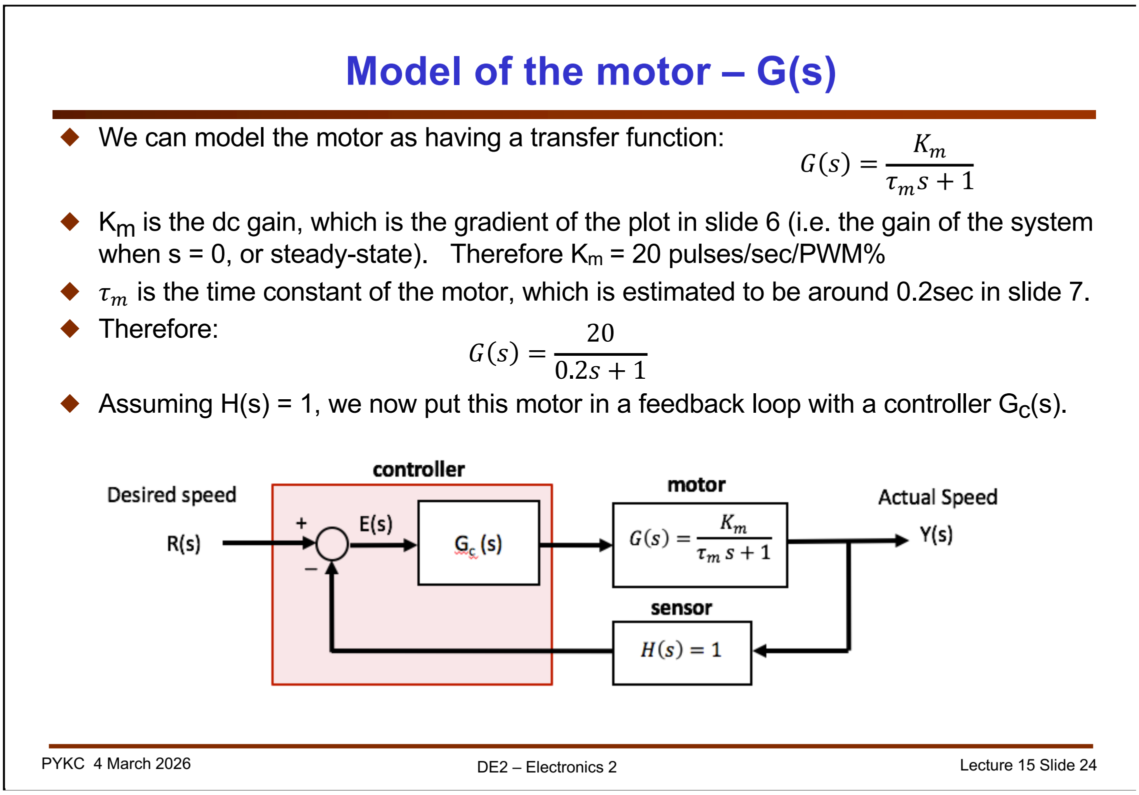
- Here is the plot of the step response of two typical motors.
- The time constant (time it takes to reach 63% of final speed) is around 0.2sec.



I also test the response of the motors to a step input (going from PWM=0% to 75%).

This shows that both motor behaves approximately to a first order function with an exponential rise of time-constant = 0.2sec.

Any first order system responses to a step inout as a exponent rising signal with a time constant τ, where τ is the time it takes to reach 63% of final value. (Remember the RC time constant from Year 1?)



Model of the motor – G(s)

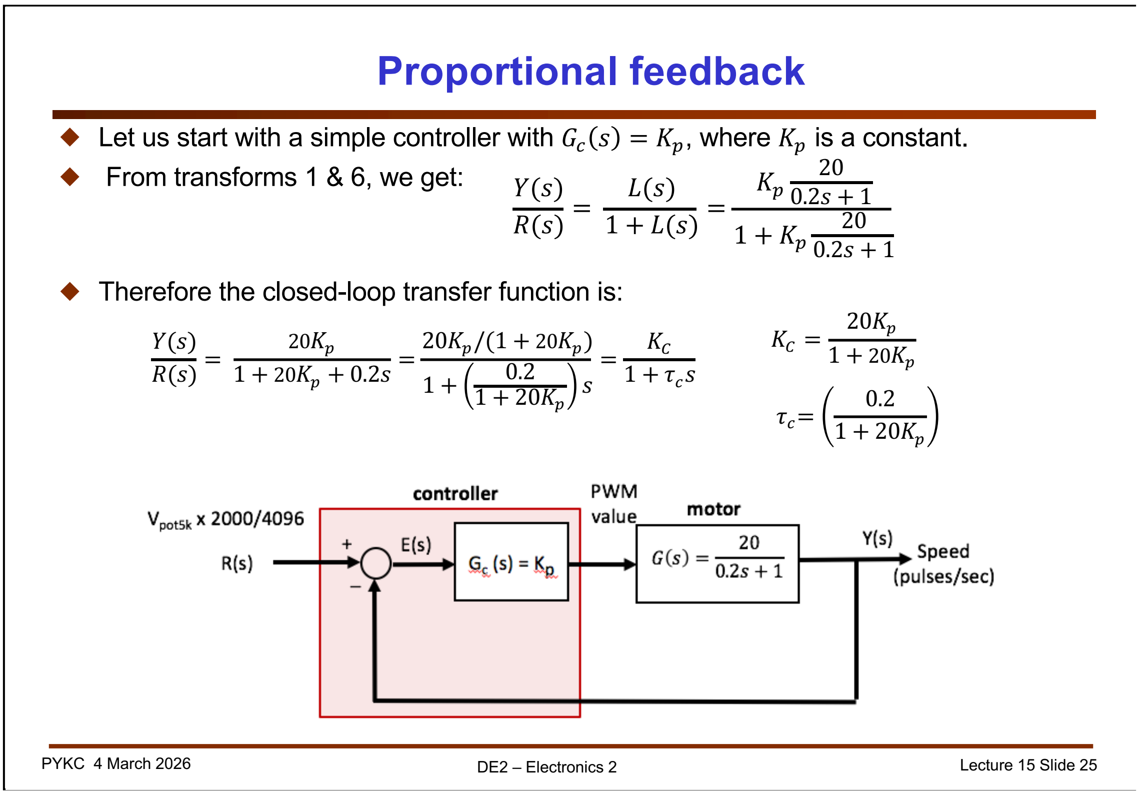
- We can model the motor as having a transfer function: $G(s) = \dfrac{K_m}{\tau_m s + 1}$
- K_m is the dc gain, which is the gradient of the plot in slide 6 (i.e. the gain of the system when s = 0, or steady-state). Therefore K_m = 20 pulses/sec/PWM%
- τ_m is the time constant of the motor, which is estimated to be around 0.2sec in slide 7.
- Therefore: $$G(s) = \frac{20}{0.2s + 1}$$
- Assuming H(s) = 1, we now put this motor in a feedback loop with a controller $G_C(s)$.

With these information, we can write down the transfer function of the motor. The dc gain is Km, and it is 20 pulses/sec/PWM%.
The time constant is 0.2sec.

Therefore the motor transfer function (from PWM duty cycle in %) to output speed (in pulses/sec) is:

$$G(s) = \frac{20}{0.2s + 1}$$

In this discussion, we assume the feedback transfer H(s) = 1.



Proportional feedback

- Let us start with a simple controller with $G_c(s) = K_p$, where K_p is a constant.
- From transforms 1 & 6, we get: $\frac{Y(s)}{R(s)} = \frac{L(s)}{1+L(s)} = \frac{K_p \frac{20}{0.2s+1}}{1+K_p \frac{20}{0.2s+1}}$
- Therefore the closed-loop transfer function is:

$$\frac{Y(s)}{R(s)} = \frac{20K_p}{1+20K_p+0.2s} = \frac{20K_p/(1+20K_p)}{1+\left(\frac{0.2}{1+20K_p}\right)s} = \frac{K_C}{1+\tau_c s}$$

$$K_C = \frac{20K_p}{1+20K_p}$$

$$\tau_c = \left(\frac{0.2}{1+20K_p}\right)$$

controller
PWM value
motor
V_{pot5k} x 2000/4096
R(s)
+
−
E(s)
$G_c(s) = K_p$
$G(s) = \frac{20}{0.2s+1}$
Y(s)
Speed (pulses/sec)

Now let us plug this motor into our simple feedback system with the control Gc(s). Furthermore, let us assume that the controller simply multiply the error signal e(t) by a constant gain Kp. In other words, the drive signal is proportional to the error signal.

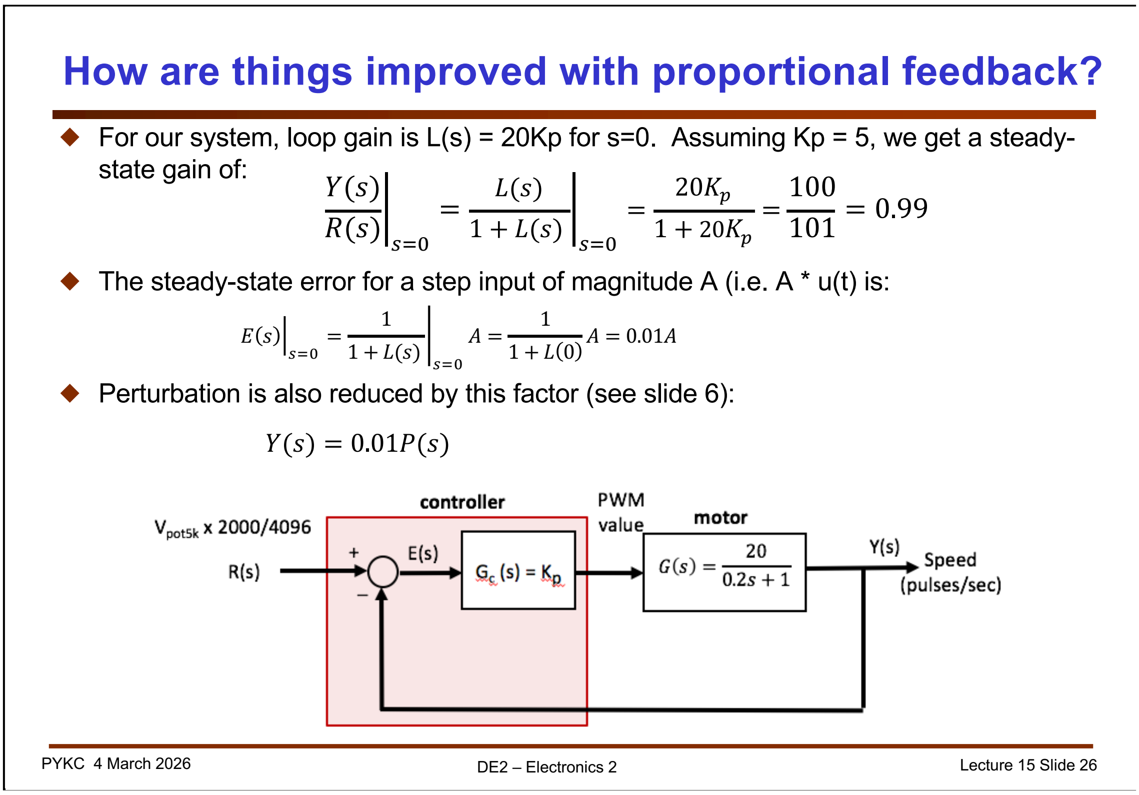
This is known as "**proportional control**" and the proportional gain is Kp.

Some algebraic manipulation give us:

$$\frac{Y(s)}{R(s)} = \frac{20K_p}{1+20K_p+0.2s} = \frac{20K_p/(1+20K_p)}{1+\left(\frac{0.2}{1+20K_p}\right)s} = \frac{K_C}{1+\tau_c s}$$

$$K_C = \frac{20K_p}{1+20K_p} \qquad \tau_c = \left(\frac{0.2}{1+20K_p}\right)$$

We will consider the implication of this result next.



How are things improved with proportional feedback?

- For our system, loop gain is L(s) = 20Kp for s=0. Assuming Kp = 5, we get a steady-state gain of:

$$\left.\frac{Y(s)}{R(s)}\right|_{s=0} = \left.\frac{L(s)}{1+L(s)}\right|_{s=0} = \frac{20K_p}{1+20K_p} = \frac{100}{101} = 0.99$$

- The steady-state error for a step input of magnitude A (i.e. A * u(t) is:

$$E(s)\Big|_{s=0} = \left.\frac{1}{1+L(s)}\right|_{s=0} A = \frac{1}{1+L(0)} A = 0.01A$$

- Perturbation is also reduced by this factor (see slide 6):

$$Y(s) = 0.01P(s)$$

Remember: $$\frac{Y(s)}{R(s)} = \frac{20K_p}{1+20K_p+0.2s} = \frac{20K_p/(1+20K_p)}{1+\left(\frac{0.2}{1+20K_p}\right)s} = \frac{K_C}{1+\tau_C s}$$

$$K_C = \frac{20K_p}{1+20K_p} \qquad \tau_C = \left(\frac{0.2}{1+20K_p}\right)$$

Let us assume the proportional gain Kp = 5 (not unreasonable), then the steady-state transfer function:

Y(0) = 0.99 R(0)

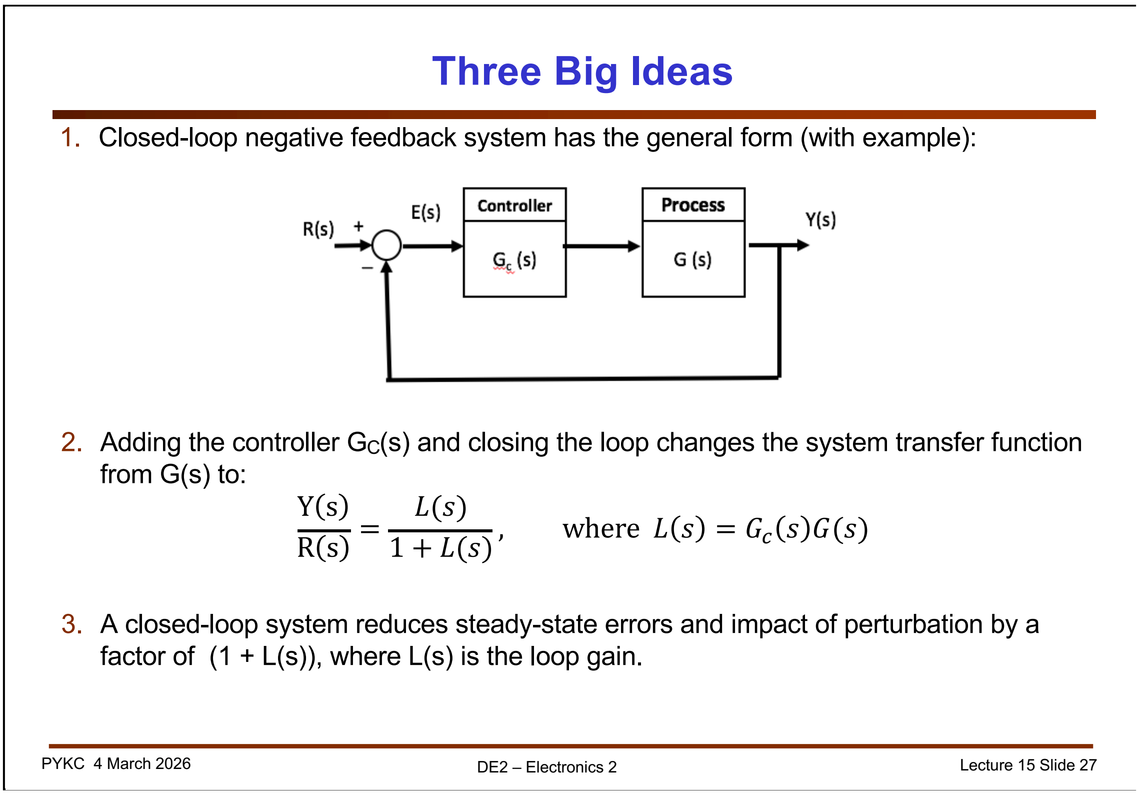
In other words, at steady state, the output tracks the input to 1%.

Also the steady-state error for a step of magnitude A, is also 1% of A – very small.

The sensitivity to perturbation is also reduced by a factor of 100.

Finally, if you rearrange to closed-loop transfer function to the form shown, you can see that the closed-loop time constant τ_C is also reduced from 0.2 to around 0.002, also around 100 times.

This is the wonderful world of feedback control!
One word of warning – everything is not as rosy as it may seem. We have not considered an important factor: the stability of the system. As we increase the proportional gain Kp, the system could go unstable if the system is NOT strictly first order. We will consider stability issue later.



Three Big Ideas

1. Closed-loop negative feedback system has the general form (with example):

R(s)
+
−
E(s)
Controller
G_c (s)
Process
G (s)
Y(s)

2. Adding the controller $G_C(s)$ and closing the loop changes the system transfer function from G(s) to:

$$\frac{\mathrm{Y(s)}}{\mathrm{R(s)}} = \frac{L(s)}{1+L(s)}, \qquad \text{where } L(s) = G_c(s)G(s)$$

3. A closed-loop system reduces steady-state errors and impact of perturbation by a factor of (1 + L(s)), where L(s) is the loop gain.

PYKC 4 March 2026 DE2 – Electronics 2 Lecture 15 Slide 27

The three main ideas of this lectures are:

1. Negative feedback involves adding a controller and then closing the loop by subtracting the output Y(s) to the setpoint R(s). The subtraction is important to ensure that the system is stable. Positive feedback is when the feedback signal is ADDED instead of subtracted, and generally result in an unstable system.

2. Loop gain is the system gain going around the loop when the feedback loop is broken. In our case, the loop gain L(s) is $G_C(s)$ G(s). This is gain is usually much larger than 1 for the feedback loop to provide benefits. This is because of 3) below.

3. Closed-loop system provide benefits if loop gain is large. Instead of having a large system gain, the gain is being used to reduce undesirable parameters such as steady-state error or noise, or disturbances.

   Generally, the steady-state error is reduced by a factor of ~L(s). Further any perturbations (disturbances) is also reduced by this factor.