ELEC4410

Control System Design

Lecture 5: Reference & Disturbance Feedforward. Cascade Control

School of Electrical Engineering and Computer Science
The University of Newcastle

Outline

- Review Affine Parameterisation.

Outline

- Review Affine Parameterisation.

- Reference Feedforward.

Outline

- Review Affine Parameterisation.
- Reference Feedforward.
- Disturbance Feedforward.

Outline

- Review Affine Parameterisation.
- Reference Feedforward.
- Disturbance Feedforward.
- Cascade Control.

Outline

- Review Affine Parameterisation.
- Reference Feedforward.
- Disturbance Feedforward.
- Cascade Control.

Reference: Control System Design, Goodwin, Graebe & Salgado.

Review Affine Parameterisation - Open Loop Stable Model

- All sensitivity functions are affine in $Q(s)$.

$T_o(s) = Q(s)G_o(s)$	Complementary Sensitivity
$S_o(s) = 1 - Q(s)G_o(s)$	Sensitivity
$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$	Input Disturbance Sensitivity
$S_{uo}(s) = Q(s)$	Control Sensitivity

Unlike the case of $C(s)$, which is nonlinear in the sensitivity functions, making it difficult to tune $C(s)$ to achieve a desired closed loop performance i.e.

$$T_o(s) = \frac{G_o(s)C(s)}{1 + G_o(s)C(s)}$$

Review Affine Parameterisation - Open Loop Stable Model

- All sensitivity functions are affine in $Q(s)$.

$T_o(s) = Q(s)G_o(s)$	Complementary Sensitivity
$S_o(s) = 1 - Q(s)G_o(s)$	Sensitivity
$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$	Input Disturbance Sensitivity
$S_{uo}(s) = Q(s)$	Control Sensitivity

Unlike the case of $C(s)$, which is nonlinear in the sensitivity functions, making it difficult to tune $C(s)$ to achieve a desired closed loop performance i.e.

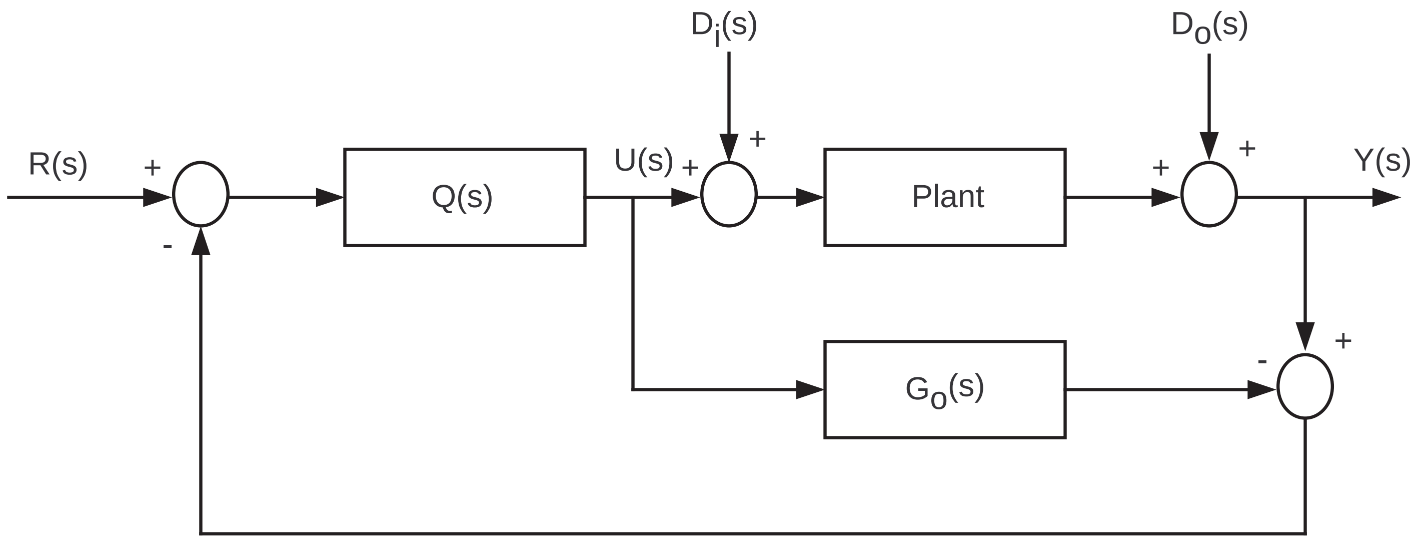
$$T_o(s) = \frac{G_o(s)C(s)}{1 + G_o(s)C(s)}$$

- The nominal loop is internally stable if and only if $Q(s)$ is a stable and proper transfer function and $C(s)$ is

$$C(s) = \frac{Q(s)}{1 - Q(s)G_o(s)}$$

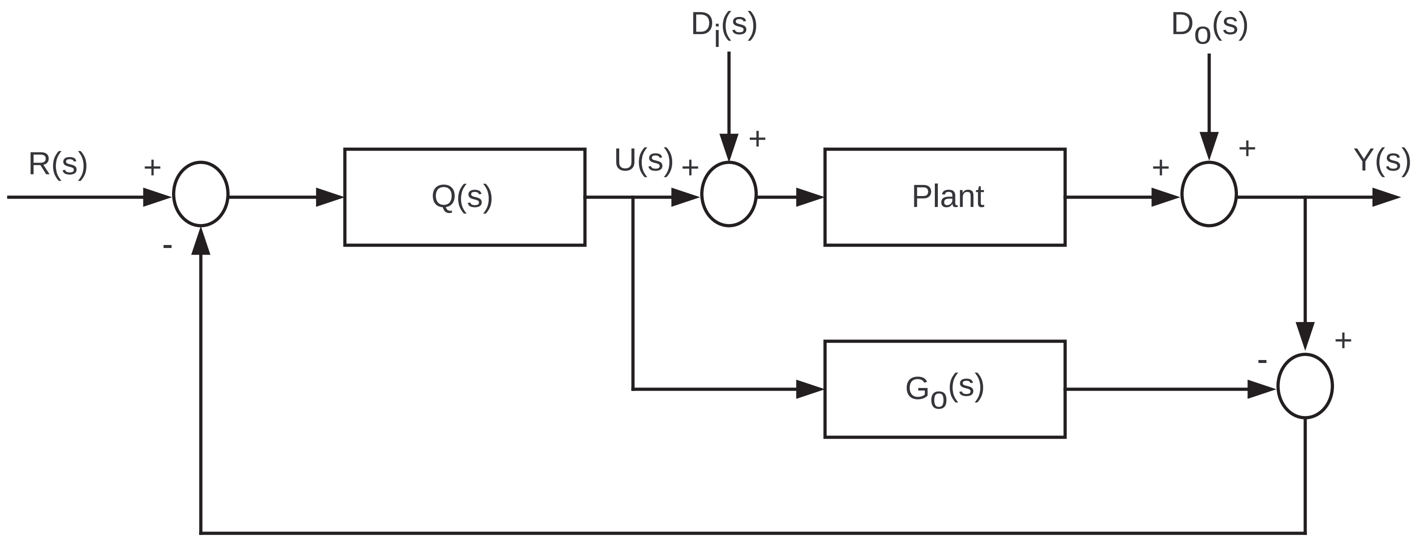
Review Affine Parameterisation - Open Loop Stable Model

Affine Parameterisation in terms of Q

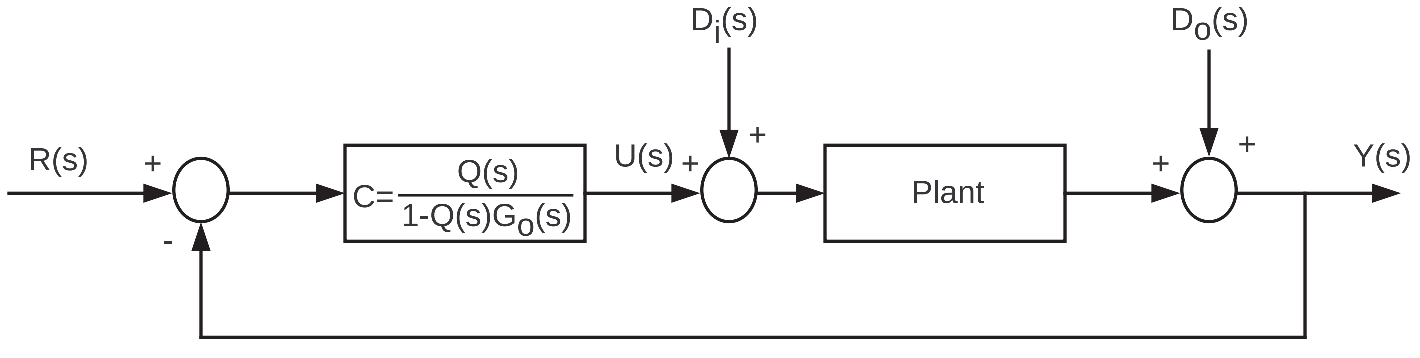


Review Affine Parameterisation - Open Loop Stable Model

Affine Parameterisation in terms of Q



Affine Parameterisation in terms of C



Review Affine Parameterisation - Open Loop Stable Model

- By use of $Q(s)$ we can shape 1 of 4 nominal sensitivities.

$T_o(s) = Q(s)G_o(s)$	Complementary Sensitivity
$S_o(s) = 1 - Q(s)G_o(s)$	Sensitivity
$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$	Input Disturbance Sensitivity
$S_{uo}(s) = Q(s)$	Control Sensitivity

Review Affine Parameterisation - Open Loop Stable Model

- By use of $Q(s)$ we can shape 1 of 4 nominal sensitivities.

$T_o(s) = Q(s)G_o(s)$	Complementary Sensitivity
$S_o(s) = 1 - Q(s)G_o(s)$	Sensitivity
$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$	Input Disturbance Sensitivity
$S_{uo}(s) = Q(s)$	Control Sensitivity

- Some trade-offs with respect to bandwidth of the closed loop that need to be considered are: reference tracking (B.W ↑), measurement noise (B.W ↓), modelling errors (B.W ↓), output disturbance rejection (B.W ↑) and the controller output (B.W ↓).

Review Affine Parameterisation - Open Loop Stable Model

- By use of $Q(s)$ we can shape 1 of 4 nominal sensitivities.

$T_o(s) = Q(s)G_o(s)$	Complementary Sensitivity
$S_o(s) = 1 - Q(s)G_o(s)$	Sensitivity
$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$	Input Disturbance Sensitivity
$S_{uo}(s) = Q(s)$	Control Sensitivity

- Some trade-offs with respect to bandwidth of the closed loop that need to be considered are: reference tracking (B.W ↑), measurement noise (B.W ↓), modelling errors (B.W ↓), output disturbance rejection (B.W ↑) and the controller output (B.W ↓).

- We know inversion is a key idea of control.

Review Affine Parameterisation - Open Loop Stable Model

- One way to design $Q(s)$ is

$$Q(s) = F_Q(s)[G_o(s)]^{-1}$$

However, recall, it is not always possible to invert $G_o(s)$ exactly.
Therefore use $G_o^i(s)$ which is a stable approximation to $[G_o(s)]^{-1}$

$$Q(s) = F_Q(s)G_o^i(s)$$

Review Affine Parameterisation - Open Loop Stable Model

- One way to design $Q(s)$ is

$$Q(s) = F_Q(s)[G_o(s)]^{-1}$$

However, recall, it is not always possible to invert $G_o(s)$ exactly. Therefore use $G_o^i(s)$ which is a stable approximation to $[G_o(s)]^{-1}$

$$Q(s) = F_Q(s)G_o^i(s)$$

- Use $F_Q(s)$ to ensure properness of $Q(s)$.

Review Affine Parameterisation - Open Loop Stable Model

- One way to design $Q(s)$ is

$$Q(s) = F_Q(s)[G_o(s)]^{-1}$$

However, recall, it is not always possible to invert $G_o(s)$ exactly. Therefore use $G_o^i(s)$ which is a stable approximation to $[G_o(s)]^{-1}$

$$Q(s) = F_Q(s)G_o^i(s)$$

- Use $F_Q(s)$ to ensure properness of $Q(s)$.

- Note that the characteristic equation of $F_Q(s)$ will also be the characteristic equation of $T_o(s)$ (and of $S_o(s)$) if all the stable poles of $G_o(s)$ are included in the approximate inversion.

Review Affine Parameterisation - Open Loop Stable Model

- What about $S_{io}(s)$? We can see that the poles of $G_o(s)$ will appear in it. These poles will only be controllable from the disturbance!

$$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$$

Review Affine Parameterisation - Open Loop Stable Model

- What about $S_{io}(s)$? We can see that the poles of $G_o(s)$ will appear in it. These poles will only be controllable from the disturbance!

$$S_{io}(s) = (1 - Q(s)G_o(s))G_o(s)$$

- What can we do about this? Slow poles in $G_o(s)$ will cause a transient associated with an input disturbance to decay at a rate dictated by these modes. The fix, essentially adding zeros to $S_o(s)$ at the location of the poles in $G_o(s)$ to be cancelled.

Review Affine Parameterisation - Open Loop Stable Model

- What about time delay, $e^{-s\tau}$? For small τ, use Padé approximation to model delay. Otherwise use Smith controller design, where $Q(s)$ is based on the rational part of the model only.

$$G_o(s) = e^{-s\tau}\overline{G}_o(s) \text{ and}$$

$$Q(s) = F_Q(s)\overline{G}_o^{\,i}(s)$$

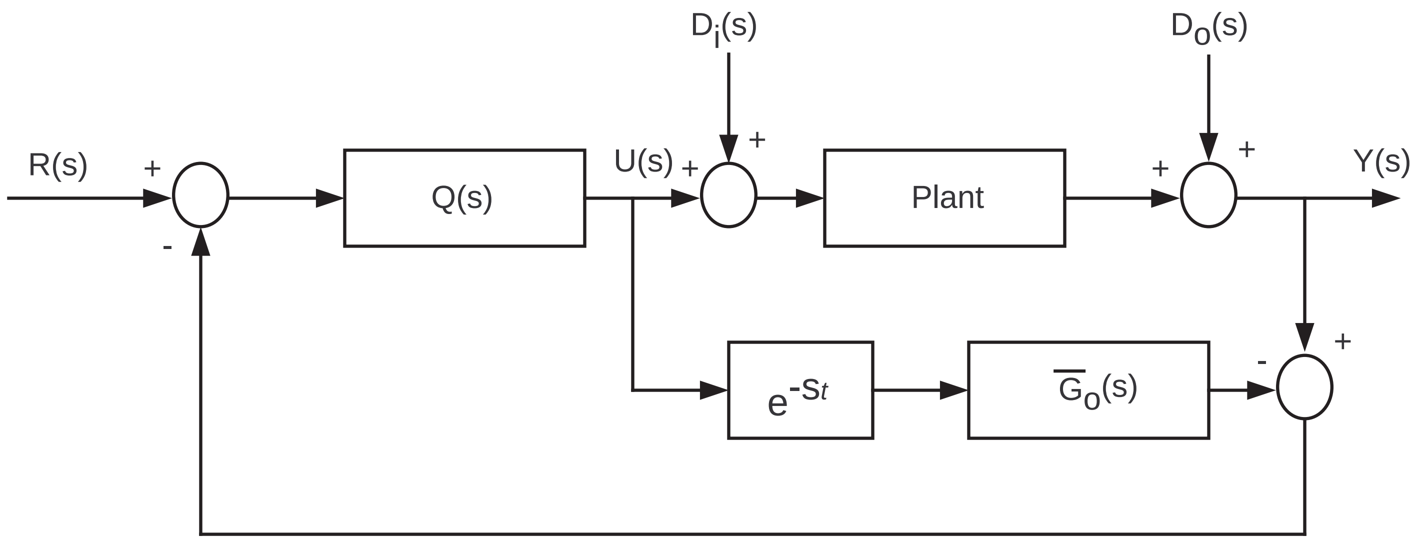
Review Affine Parameterisation - Open Loop Stable Model

- What about time delay, $e^{-s\tau}$? For small τ, use Padé approximation to model delay. Otherwise use Smith controller design, where $Q(s)$ is based on the rational part of the model only.

$$G_o(s) = e^{-s\tau}\overline{G}_o(s) \text{ and}$$

$$Q(s) = F_Q(s)\overline{G}_o^{\,i}(s)$$

Smith Controller in Q Parameterisation Form



Reference Feedforward

- A disadvantage of a one-degree-of-freedom control system is that the performance criteria that can be realised are limited.

Reference Feedforward

- A disadvantage of a one-degree-of-freedom control system is that the performance criteria that can be realised are limited.
- For example: if the roots of the characteristic equation are selected to provide a certain amount of relative damping, the maximum overshoot of the step response may still be excessive, owing to the zeros in the closed loop transfer function.

Reference Feedforward

- A disadvantage of a one-degree-of-freedom control system is that the performance criteria that can be realised are limited.
- For example: if the roots of the characteristic equation are selected to provide a certain amount of relative damping, the maximum overshoot of the step response may still be excessive, owing to the zeros in the closed loop transfer function.
- Reference Feedforward can help reduce this and other effects as we will see.

Reference Feedforward

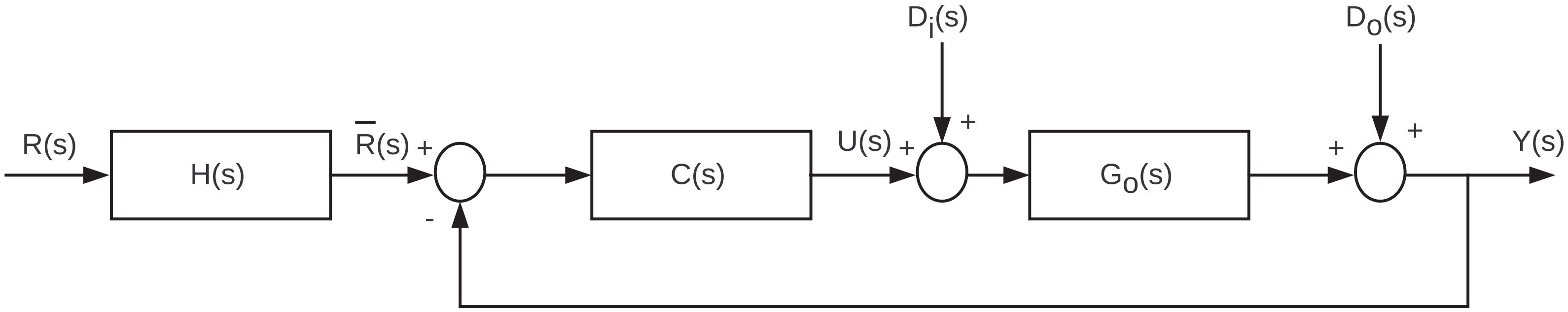
- Reference Feedforward is also known as the 2nd degree-of-freedom control.

Reference Feedforward

- Reference Feedforward is also known as the 2nd degree-of-freedom control.
- We can use a two-degree-of-freedom architecture to improve reference tracking.

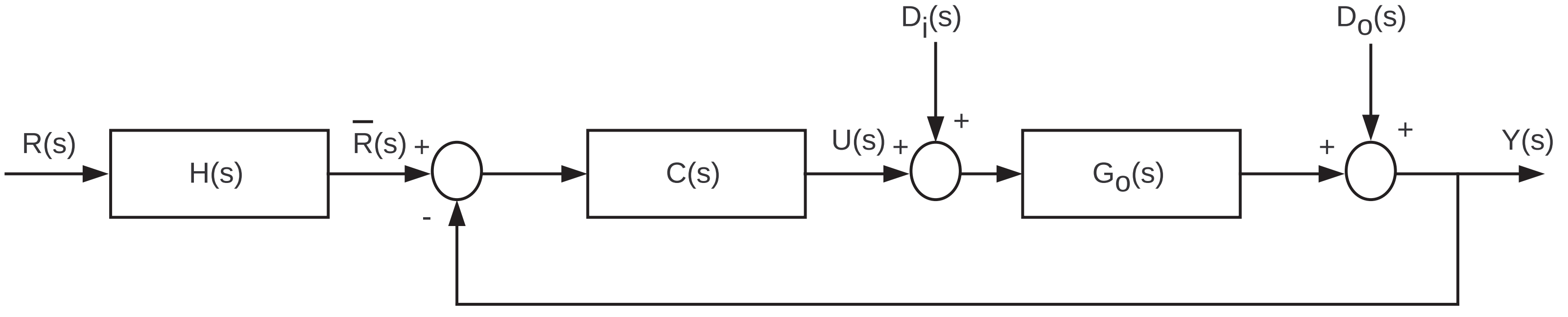
Reference Feedforward

- Reference Feedforward is also known as the 2nd degree-of-freedom control.
- We can use a two-degree-of-freedom architecture to improve reference tracking.
- Consider the two-degree-of-freedom architecture:



Reference Feedforward

- Reference Feedforward is also known as the 2nd degree-of-freedom control.
- We can use a two-degree-of-freedom architecture to improve reference tracking.
- Consider the two-degree-of-freedom architecture:



- The Feedforward control element is in the forward path of the feedback loop.

Reference Feedforward

- The tracking performance can be quantified through the following equations (assuming the disturbances $D_i(s)$ and $D_o(s)$ are zero):

$$Y(s) = H(s)T_o(s)R(s)$$

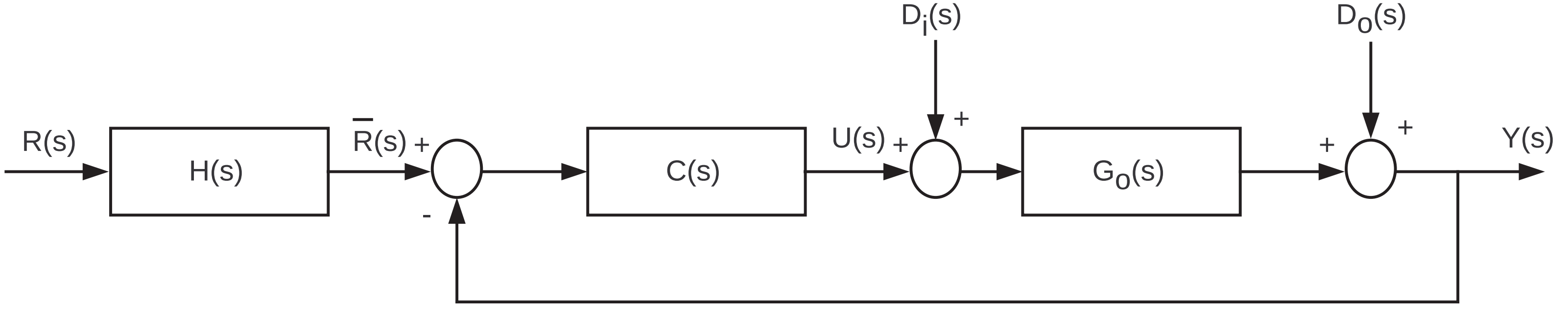
$$U(s) = H(s)S_{uo}(s)R(s)$$

Reference Feedforward

- The tracking performance can be quantified through the following equations (assuming the disturbances $D_i(s)$ and $D_o(s)$ are zero):

$$Y(s) = H(s)T_o(s)R(s)$$

$$U(s) = H(s)S_{uo}(s)R(s)$$



Reference Feedforward

- Another way of describing the flexibility of the two-degree-of-freedom controller is that the controller $C(s)$ is usually designed to provide a certain degree of system stability and performance, but since the zeros of $C(s)$ always become the zeros of the closed loop transfer function, unless some zeros are cancelled by the poles of the process, these zeros may cause a large overshoot in the system output even when the relative damping as determined by the characteristic equation is satisfactory.

Reference Feedforward

- Another way of describing the flexibility of the two-degree-of-freedom controller is that the controller $C(s)$ is usually designed to provide a certain degree of system stability and performance, but since the zeros of $C(s)$ always become the zeros of the closed loop transfer function, unless some zeros are cancelled by the poles of the process, these zeros may cause a large overshoot in the system output even when the relative damping as determined by the characteristic equation is satisfactory.
- In this case and for other reasons, the transfer function $H(s)$ may be used for the control or cancellation of the undesirable poles or zeros of the closed loop transfer function, while keeping the characteristic equation intact.

Reference Feedforward

- Another way of describing the flexibility of the two-degree-of-freedom controller is that the controller $C(s)$ is usually designed to provide a certain degree of system stability and performance, but since the zeros of $C(s)$ always become the zeros of the closed loop transfer function, unless some zeros are cancelled by the poles of the process, these zeros may cause a large overshoot in the system output even when the relative damping as determined by the characteristic equation is satisfactory.
- In this case and for other reasons, the transfer function $H(s)$ may be used for the control or cancellation of the undesirable poles or zeros of the closed loop transfer function, while keeping the characteristic equation intact.
- Of course we could introduce zeros in $H(s)$ to cancel some of the undesirable poles of the closed loop transfer function that result from the controller $C(s)$.

Reference Feedforward

- The key to the reference feedforward controller is that the controller $H(s)$ is not in the loop of the system, so that it does not affect the roots of the characteristic equation of the original system.

Reference Feedforward

- The key to the reference feedforward controller is that the controller $H(s)$ is not in the loop of the system, so that it does not affect the roots of the characteristic equation of the original system.
- The poles and zeros of $H(s)$ may be selected to add to or cancel poles and zeros of the closed loop transfer function, $T_o(s)$.

Reference Feedforward

- The key to the reference feedforward controller is that the controller $H(s)$ is not in the loop of the system, so that it does not affect the roots of the characteristic equation of the original system.

- The poles and zeros of $H(s)$ may be selected to add to or cancel poles and zeros of the closed loop transfer function, $T_o(s)$.

- The essential idea of reference feedforward is to use $H(s)$ to invert $T_o(s)$ at certain key frequencies.

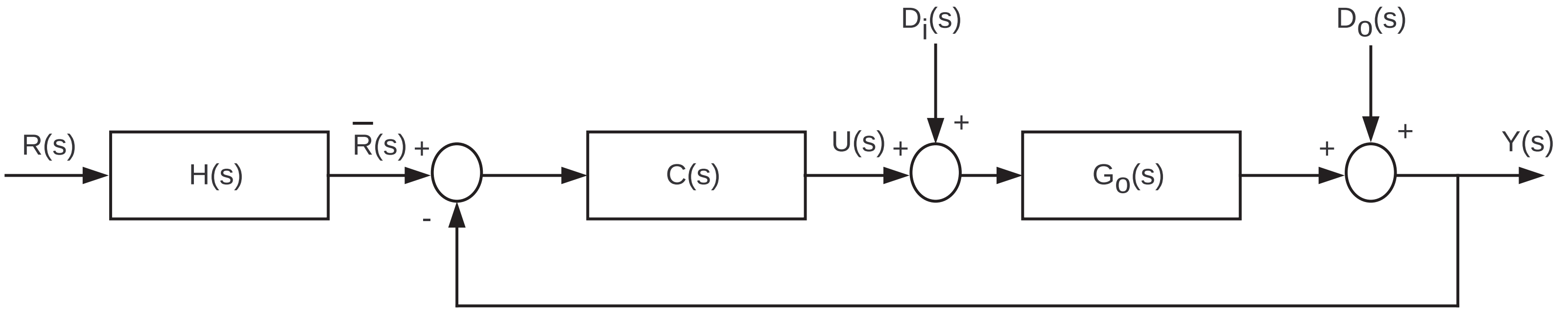
Reference Feedforward

- The key to the reference feedforward controller is that the controller $H(s)$ is not in the loop of the system, so that it does not affect the roots of the characteristic equation of the original system.
- The poles and zeros of $H(s)$ may be selected to add to or cancel poles and zeros of the closed loop transfer function, $T_o(s)$.
- The essential idea of reference feedforward is to use $H(s)$ to invert $T_o(s)$ at certain key frequencies.
- Note that, by this strategy, one can avoid using high gain feedback to bring $T_o(\omega_i)$ to $\mathbf{1}$.

Reference Feedforward

- The key to the reference feedforward controller is that the controller $H(s)$ is not in the loop of the system, so that it does not affect the roots of the characteristic equation of the original system.
- The poles and zeros of $H(s)$ may be selected to add to or cancel poles and zeros of the closed loop transfer function, $T_o(s)$.
- The essential idea of reference feedforward is to use $H(s)$ to invert $T_o(s)$ at certain key frequencies.
- Note that, by this strategy, one can avoid using high gain feedback to bring $T_o(\omega_i)$ to $\mathbf{1}$.
- Note, however, that use of reference feedforward in this way does not provide perfect tracking if there is a change in the model.

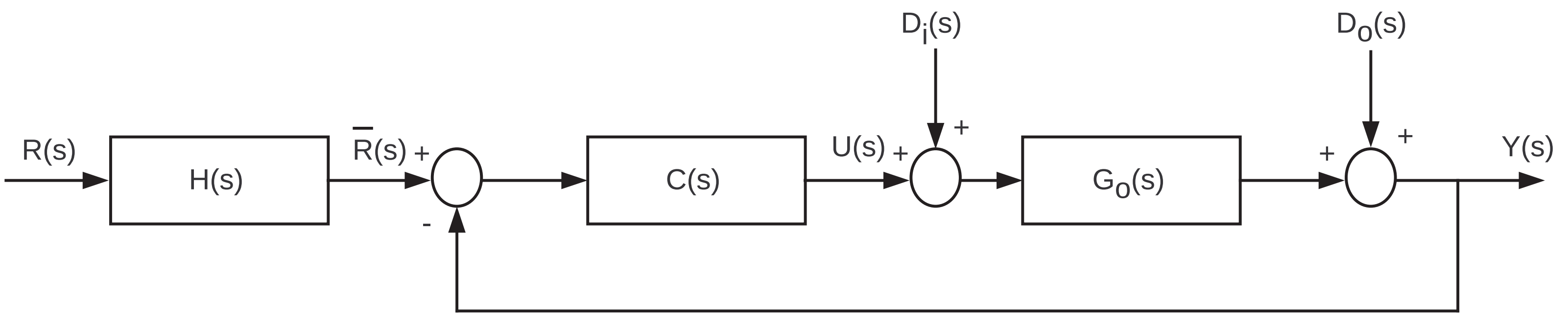
Reference Feedforward



- Recall,

$$\frac{Y(s)}{R(s)} = H(s)T_o(s) \quad \text{and} \quad T_o(s) = \frac{G(s)C(s)}{1 + G(s)C(s)}$$

Reference Feedforward



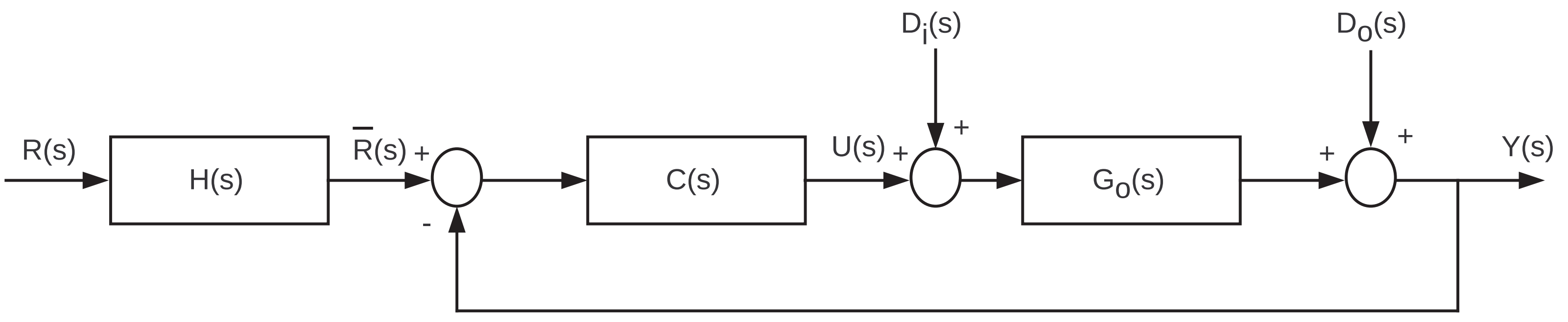
- Recall,

$$\frac{Y(s)}{R(s)} = H(s)T_o(s) \quad \text{and} \quad T_o(s) = \frac{G(s)C(s)}{1 + G(s)C(s)}$$

- Also,

$$\frac{U(s)}{R(s)} = H(s)S_{uo}(s)$$

Reference Feedforward



- Recall,

$$\frac{Y(s)}{R(s)} = H(s)T_o(s) \quad \text{and} \quad T_o(s) = \frac{G(s)C(s)}{1 + G(s)C(s)}$$

- Also,

$$\frac{U(s)}{R(s)} = H(s)S_{uo}(s)$$

- Can now shape the sensitivity from R(s) to Y(s) independent of the other sensitivities.

Reference Feedforward

- Performance of loop can be made robust by choosing B.W. of $T_o(s)$ small.

Reference Feedforward

- Performance of loop can be made robust by choosing B.W. of $T_o(s)$ small.
- Again key idea ⇒ Inversion. However we now invert $T_o(s)$

$$H(s) = F_R(s)[T_o(s)]^{-1}$$

Reference Feedforward

- Performance of loop can be made robust by choosing B.W. of $T_o(s)$ small.

- Again key idea $\Rightarrow$ Inversion. However we now invert $T_o(s)$

$$H(s) = F_R(s)[T_o(s)]^{-1}$$

- $H(s)$ needs to be stable and proper.

Reference Feedforward

- Performance of loop can be made robust by choosing B.W. of $T_o(s)$ small.

- Again key idea ⇒ Inversion. However we now invert $T_o(s)$

$$H(s) = F_R(s)[T_o(s)]^{-1}$$

- $H(s)$ needs to be stable and proper.

- For regulators, which have constant set points, has no benefit.

Reference Feedforward

- Performance of loop can be made robust by choosing B.W. of $T_o(s)$ small.
- Again key idea ⇒ Inversion. However we now invert $T_o(s)$

$$H(s) = F_R(s)[T_o(s)]^{-1}$$

- $H(s)$ needs to be stable and proper.
- For regulators, which have constant set points, has no benefit.
- Good for set point tracking loops.

Disturbance Feedforward

- Disturbance Feedforward is also known as the 3rd degree-of-freedom control. Note that it only uses two-degrees-of-freedom in the control.

Disturbance Feedforward

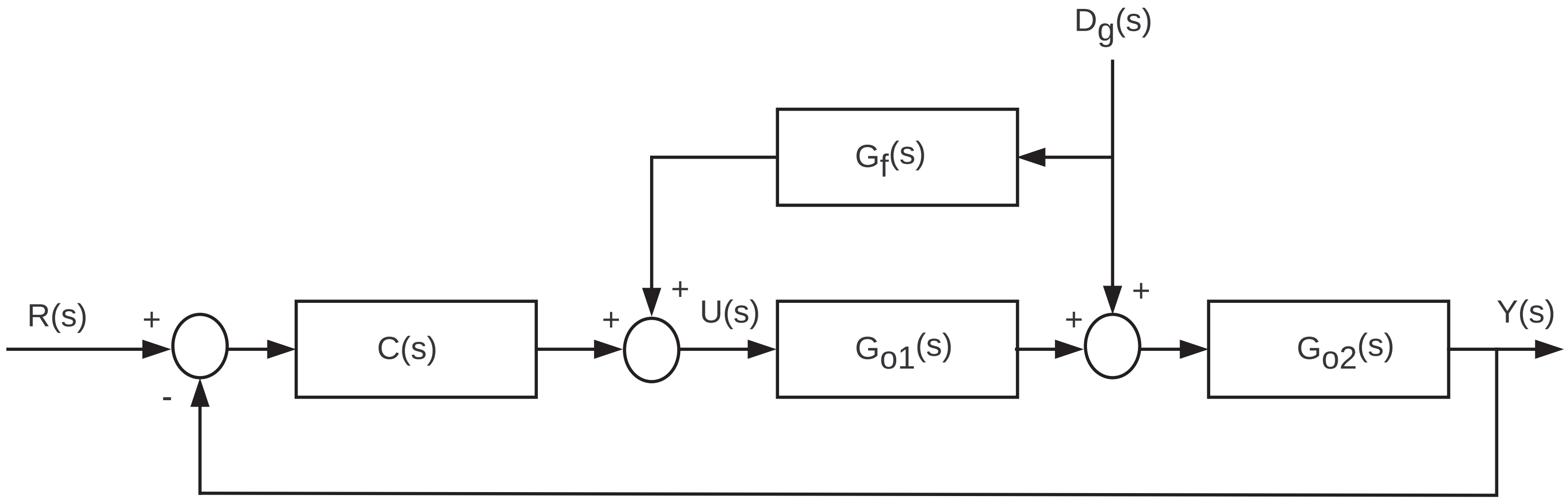
- Disturbance Feedforward is also known as the 3rd degree-of-freedom control. Note that it only uses two-degrees-of-freedom in the control.
- If a disturbance can be measured, then feedforward can be applied to improve disturbance rejection.

Disturbance Feedforward

- Disturbance Feedforward is also known as the 3rd degree-of-freedom control. Note that it only uses two-degrees-of-freedom in the control.
- If a disturbance can be measured, then feedforward can be applied to improve disturbance rejection.
- Once again, a key concept is inversion.

Disturbance Feedforward

- Disturbance Feedforward is also known as the 3rd degree-of-freedom control. Note that it only uses two-degrees-of-freedom in the control.
- If a disturbance can be measured, then feedforward can be applied to improve disturbance rejection.
- Once again, a key concept is inversion.
- We want to use the control signal, $U(s)$, to cancel the disturbance, $D_g(s)$, at the point where it enters the process.



Disturbance Feedforward

- Assuming zero reference,

$$Y(s) = S_o(s)G_{o2}(s)(1 + G_{o1}G_f(s))D_g(s).$$

Disturbance Feedforward

- Assuming zero reference,

$$Y(s) = S_o(s)G_{o2}(s)(1 + G_{o1}G_f(s))D_g(s).$$

- Also,

$$U(s) = -S_{uo}(s)(G_{o2}(s) + G_f(s))D_g(s).$$

Disturbance Feedforward

- Assuming zero reference,

$$Y(s) = S_o(s)G_{o2}(s)(1 + G_{o1}G_f(s))D_g(s).$$

- Also,

$$U(s) = -S_{uo}(s)(G_{o2}(s) + G_f(s))D_g(s).$$

- $G_f(s)$ must be stable and proper (it is open loop control).

Disturbance Feedforward

- Assuming zero reference,

$$Y(s) = S_o(s)G_{o2}(s)(1 + G_{o1}G_f(s))D_g(s).$$

- Also,

$$U(s) = -S_{uo}(s)(G_{o2}(s) + G_f(s))D_g(s).$$

- $G_f(s)$ must be stable and proper (it is open loop control).
- What should $G_f(s)$ be? To reject disturbances, i.e. $Y(s) = 0$, ideally

$$G_{o1}(s)G_f(s) = -1$$

$$\therefore G_f(s) = -[G_{o1}]^{-1}$$

Disturbance Feedforward

- Assuming zero reference,

$$Y(s) = S_o(s)G_{o2}(s)(1 + G_{o1}G_f(s))D_g(s).$$

- Also,

$$U(s) = -S_{uo}(s)(G_{o2}(s) + G_f(s))D_g(s).$$

- $G_f(s)$ must be stable and proper (it is open loop control).
- What should $G_f(s)$ be? To reject disturbances, i.e. $Y(s) = 0$, ideally

$$G_{o1}(s)G_f(s) = -1$$

$$\therefore G_f(s) = -[G_{o1}]^{-1}$$

- $G_f(s)$ would be expected to be high pass as $G_{o1}(s)$ will typically possess a low pass characteristic. Therefore will have to include "fast" poles in $G_f(s)$ to make proper.

Disturbance Feedforward

- Assuming zero reference,

$$Y(s) = S_o(s)G_{o2}(s)(1 + G_{o1}G_f(s))D_g(s).$$

- Also,

$$U(s) = -S_{uo}(s)(G_{o2}(s) + G_f(s))D_g(s).$$

- $G_f(s)$ must be stable and proper (it is open loop control).
- What should $G_f(s)$ be? To reject disturbances, i.e. $Y(s) = 0$, ideally

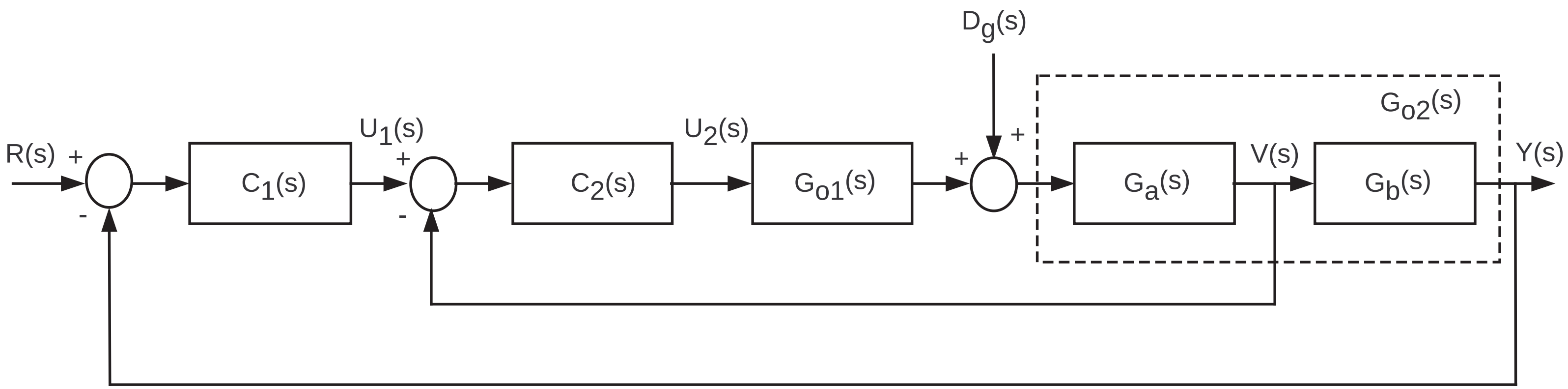
$$G_{o1}(s)G_f(s) = -1$$

$$\therefore G_f(s) = -[G_{o1}]^{-1}$$

- $G_f(s)$ would be expected to be high pass as $G_{o1}(s)$ will typically possess a low pass characteristic. Therefore will have to include “fast” poles in $G_f(s)$ to make proper.
- Gives more flexibility in the design as trade-offs can be relaxed.

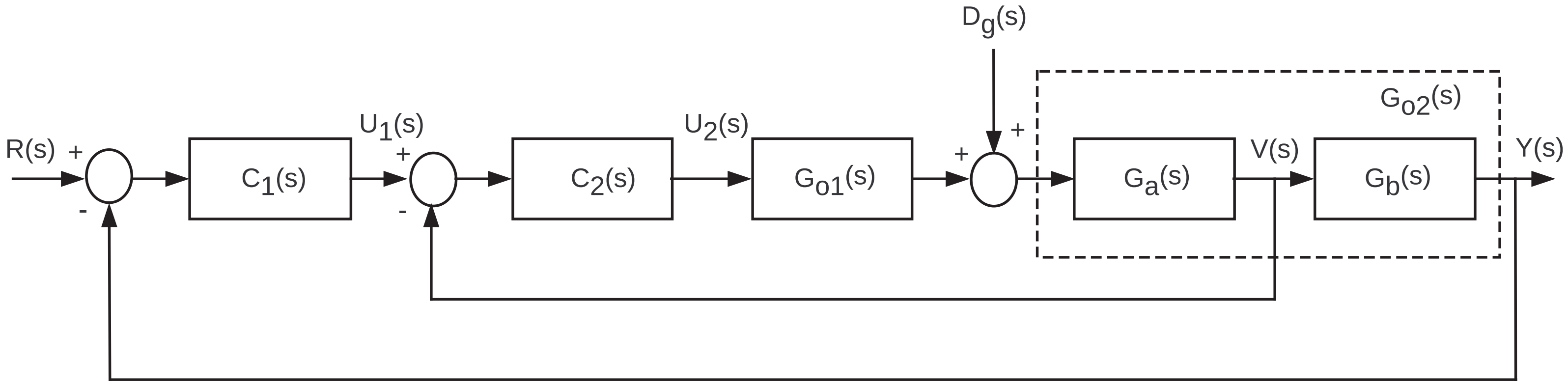
Cascade Control

- If a measurement of a variable can be made between the point where a disturbance enters the process and the output of the process, then we can utilise feedback for disturbance rejection. This gives rise to “cascade control”.



Cascade Control

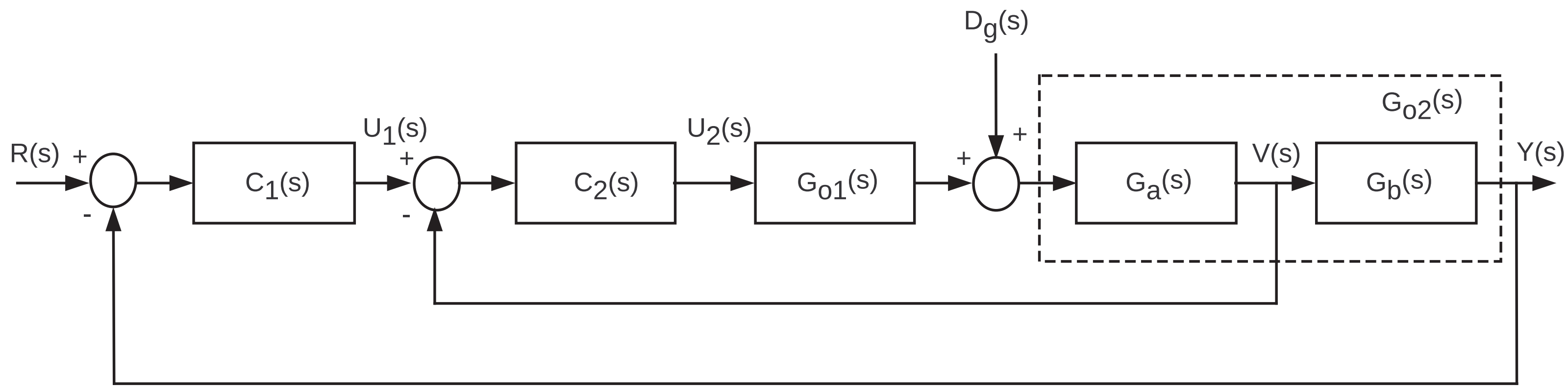
- If a measurement of a variable can be made between the point where a disturbance enters the process and the output of the process, then we can utilise feedback for disturbance rejection. This gives rise to "cascade control".



- Cascade control usually consists of two feedback loops

Cascade Control

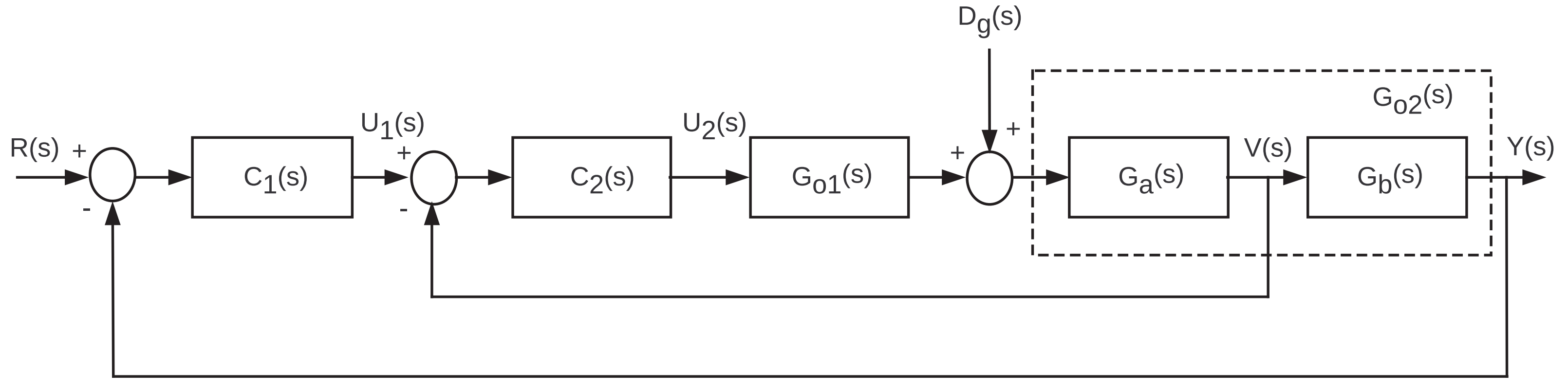
- If a measurement of a variable can be made between the point where a disturbance enters the process and the output of the process, then we can utilise feedback for disturbance rejection. This gives rise to "cascade control".



- Cascade control usually consists of two feedback loops
  - Primary (outer) controlled by C_1,

Cascade Control

- If a measurement of a variable can be made between the point where a disturbance enters the process and the output of the process, then we can utilise feedback for disturbance rejection. This gives rise to “cascade control”.



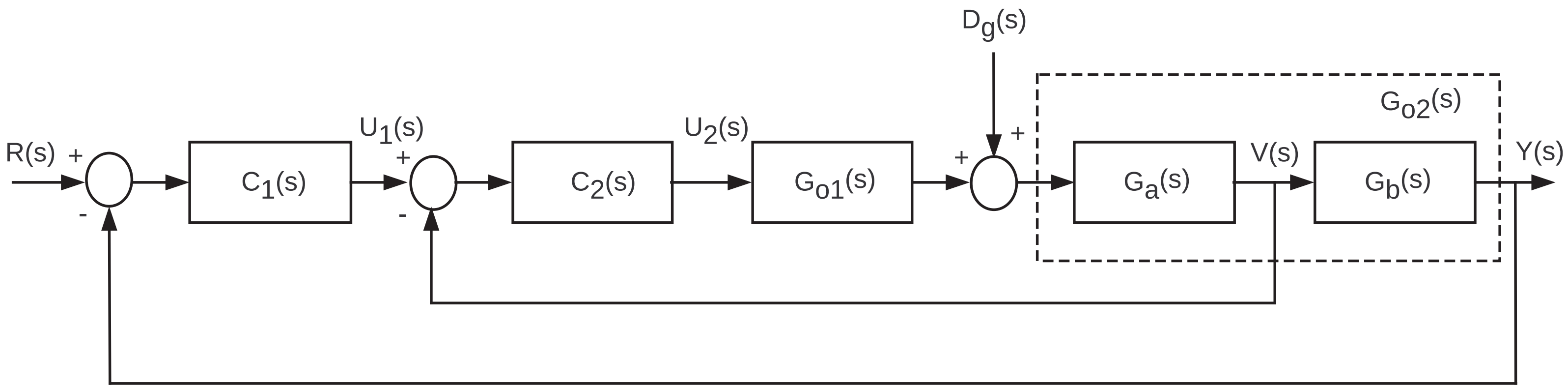
- Cascade control usually consists of two feedback loops
  - Primary (outer) controlled by C_1,
  - Secondary (inner) controlled by C_2.

Cascade Control

- $C_2(s)$ can be designed to attenuate $D_g(s)$ before it affects the output.

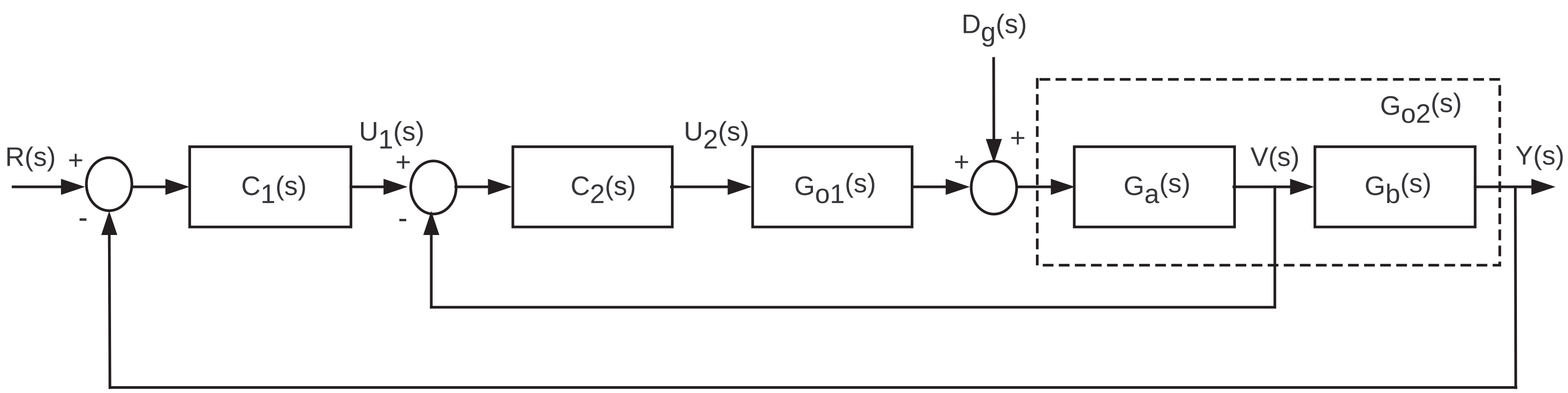
Cascade Control

- $C_2(s)$ can be designed to attenuate $D_g(s)$ before it affects the output.
- Main benefits arise when



Cascade Control

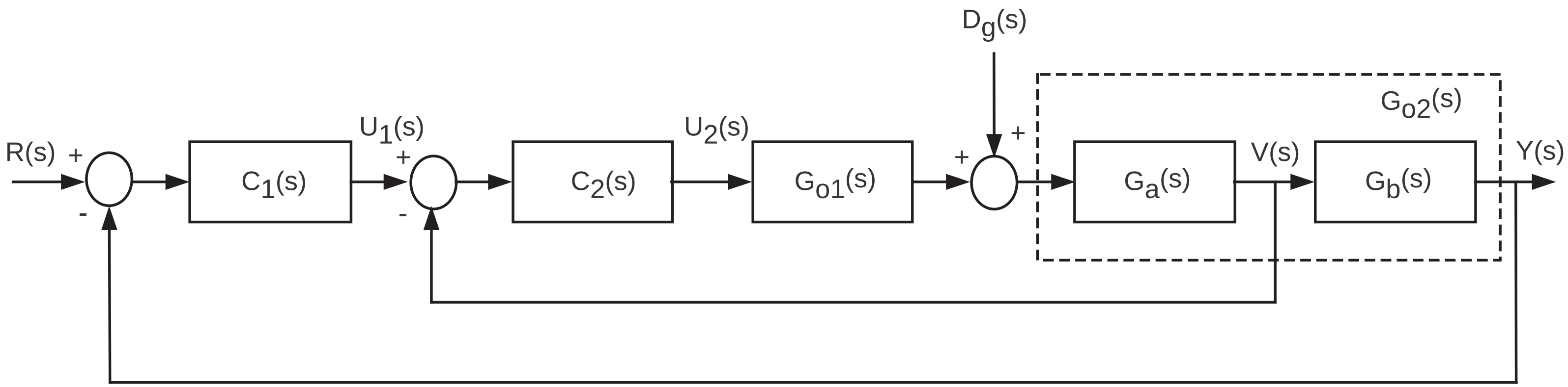
- $C_2(s)$ can be designed to attenuate $D_g(s)$ before it affects the output.
- Main benefits arise when
  - $G_a(s)$ contains nonlinearities that limit the loop performance



The University of Newcastle

Cascade Control

- $C_2(s)$ can be designed to attenuate $D_g(s)$ before it affects the output.
- Main benefits arise when
  - $G_a(s)$ contains nonlinearities that limit the loop performance
  - $G_b(s)$ is N.M.P. and / or contains time delays that limit B.W.



Cascade Control

- The output of the system is given by:

$$Y(s) = C_2(s)G_o(s)S_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s)$$

$$G_o(s) = G_{o1}(s)G_{o2}(s)$$

Cascade Control

- The output of the system is given by:

$$Y(s) = C_2(s)G_o(s)S_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s)$$
$$G_o(s) = G_{o1}(s)G_{o2}(s)$$

- This can be re-written as:

$$Y(s) = G_b(s)T_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s) \tag{3}$$

Cascade Control

- The output of the system is given by:

$$Y(s) = C_2(s)G_o(s)S_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s)$$
$$G_o(s) = G_{o1}(s)G_{o2}(s)$$

- This can be re-written as:

$$Y(s) = G_b(s)T_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s) \qquad (5)$$

- If we don't have an inner loop, the output is given by

$$Y(s) = G_o(s)U(s) + G_{o2}(s)D_g(s) \qquad (6)$$

Cascade Control

- The output of the system is given by:

$$Y(s) = C_2(s)G_o(s)S_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s)$$

$$G_o(s) = G_{o1}(s)G_{o2}(s)$$

- This can be re-written as:

$$Y(s) = G_b(s)T_{o2}(s)U_1(s) + G_{o2}(s)S_{o2}(s)D_g(s) \tag{7}$$

- If we don't have an inner loop, the output is given by

$$Y(s) = G_o(s)U(s) + G_{o2}(s)D_g(s) \tag{8}$$

- It can be seen in (1) that the disturbance will be somewhat attenuated when compared to (2).

Cascade Control

- The secondary controller is usually designed first.

Cascade Control

- The secondary controller is usually designed first.
- The primary controller is then designed based on an equivalent plant.

$$G_{eq} \triangleq G_b(s)T_{o2}(s)$$

Cascade Control

- The secondary controller is usually designed first.
- The primary controller is then designed based on an equivalent plant.

$$G_{eq} \triangleq G_b(s)T_{o2}(s)$$

- Generally, the secondary controller is designed such that

$$\text{B.W of } T_{o2}(s) > \text{B.W of } T_{o1}(s)$$