5ELEN018W - Robotic Principles
Lecture 6: Control - Part 1

Dr Dimitris C. Dracopoulos

What is Robot Control?

What is Robot Control?

- A robot needs to move its joints to achieve tasks

What is Robot Control?

- A robot needs to move its joints to achieve tasks
- A mobile robot moves to different locations

What is Robot Control?

- A robot needs to move its joints to achieve tasks
- A mobile robot moves to different locations

The movement of a robot (joints) is done using actuators.

What is Robot Control?

- A robot needs to move its joints to achieve tasks
- A mobile robot moves to different locations

The movement of a robot (joints) is done using actuators.

In general, everything can be considered as **control**:

- Decisions we make affect (control) our future

What is Robot Control?

- A robot needs to move its joints to achieve tasks
- A mobile robot moves to different locations

The movement of a robot (joints) is done using actuators.

In general, everything can be considered as **control**:

- Decisions we make affect (control) our future
- Decision while driving affect (control) the next position and the final location

What is Robot Control?

- A robot needs to move its joints to achieve tasks
- A mobile robot moves to different locations

The movement of a robot (joints) is done using actuators.

In general, everything can be considered as **control**:

- Decisions we make affect (control) our future
- Decision while driving affect (control) the next position and the final location
- Control theory is a big area used not only in engineering and robotics, but in computer science

What is Robot Control?

- A robot needs to move its joints to achieve tasks
- A mobile robot moves to different locations

The movement of a robot (joints) is done using actuators.

In general, everything can be considered as **control**:

- Decisions we make affect (control) our future
- Decision while driving affect (control) the next position and the final location
- Control theory is a big area used not only in engineering and robotics, but in computer science
- Can be seen as what is <u>the best next action to take</u> (given a specific state) so as to <u>achieve (optimise) specific objectives!</u>

Actuators (Motors)

An actuator is a device that causes motion.

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

The output of an actuator can be:

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

The output of an actuator can be:

- Speed

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

The output of an actuator can be:

- Speed
- Force (torque in the case of rotation)

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

The output of an actuator can be:

- Speed
- Force (torque in the case of rotation)

Types of actuators:

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

The output of an actuator can be:

- Speed
- Force (torque in the case of rotation)

Types of actuators:

- Hydraulic (using compressed oil)

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

The output of an actuator can be:

- Speed
- Force (torque in the case of rotation)

Types of actuators:

- Hydraulic (using compressed oil)
- Pneumatic (using compressed air)

Actuators (Motors)

An actuator is a device that causes motion.

- Linear motion
- Rotary

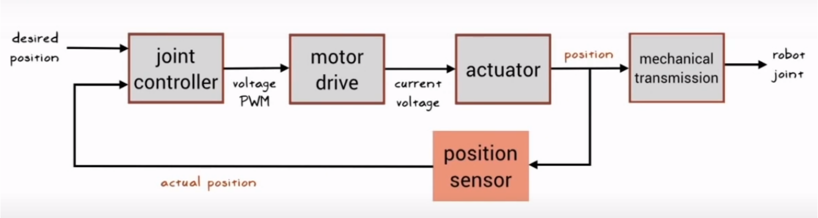
The output of an actuator can be:

- Speed
- Force (torque in the case of rotation)

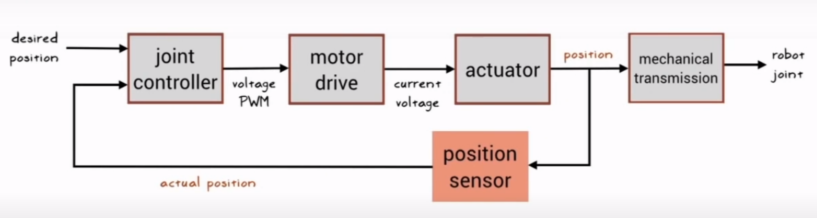
Types of actuators:

- Hydraulic (using compressed oil)
- Pneumatic (using compressed air)
- Electric (using current)

Components of a Robot Joint Control System



Components of a Robot Joint Control System



The dynamic system that is to be controlled is called the **plant**.

Open-loop Control

In open-loop (feedforward) control the control action from the controller is independent of the process (dynamic system or plant) output.

Open-loop Control

In open-loop (feedforward) control the control action from the controller is independent of the process (dynamic system or plant) output.

Example:

Open-loop Control

In open-loop (feedforward) control the control action from the controller is independent of the process (dynamic system or plant) output.

Example:

- Control of a boiler using a timer

Open-loop Control

In open-loop (feedforward) control the control action from the controller is independent of the process (dynamic system or plant) output.

Example:

- Control of a boiler using a timer
- The controller (timer) switches the boiler on or off based on specific times and independent of the temperatures that the boiler has reached

Open-loop Control

In open-loop (feedforward) control the control action from the controller is independent of the process (dynamic system or plant) output.

Example:

- Control of a boiler using a timer
- The controller (timer) switches the boiler on or off based on specific times and independent of the temperatures that the boiler has reached

There is no feedback from the plant output to the controller on what actual temperature the boiler reached.

Open-loop Control

In open-loop (feedforward) control the control action from the controller is independent of the process (dynamic system or plant) output.

Example:

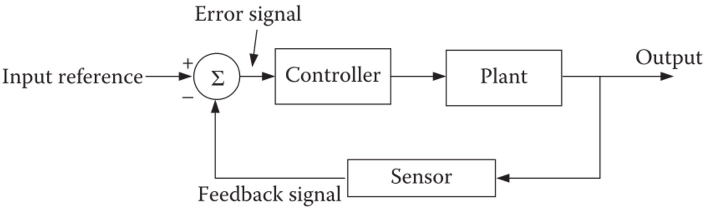
- Control of a boiler using a timer
- The controller (timer) switches the boiler on or off based on specific times and independent of the temperatures that the boiler has reached

There is no feedback from the plant output to the controller on what actual temperature the boiler reached.

Closed-loop (Feedback) control

Closed-loop (Feedback) control

Block diagram of closed-loop control system:



Error Response

- How to measure the current error?

Error Response

- How to measure the current error?

The desired behaviour (reference input) is compared with the current actual output value of the plant:

Error Response

- How to measure the current error?

The desired behaviour (reference input) is compared with the current actual output value of the plant:

Example:

Error Response

- How to measure the current error?

The desired behaviour (reference input) is compared with the current actual output value of the plant:

Example:
the desired joint position of a multi-joint robot is $\theta_d(t)$ and the actual joint position is $\theta(t)$, then the joint error is:

Error Response

- How to measure the current error?

The desired behaviour (reference input) is compared with the current actual output value of the plant:

Example:
the desired joint position of a multi-joint robot is $\theta_d(t)$ and the actual joint position is $\theta(t)$, then the joint error is:

$$\theta_e = \theta_d(t) - \theta(t) \tag{1}$$

Error Response - Characteristics of a Good Controller

Error Response - Characteristics of a Good Controller

An ideal controller would drive the error to 0 instantly and keep it 0 forever.

Error Response - Characteristics of a Good Controller

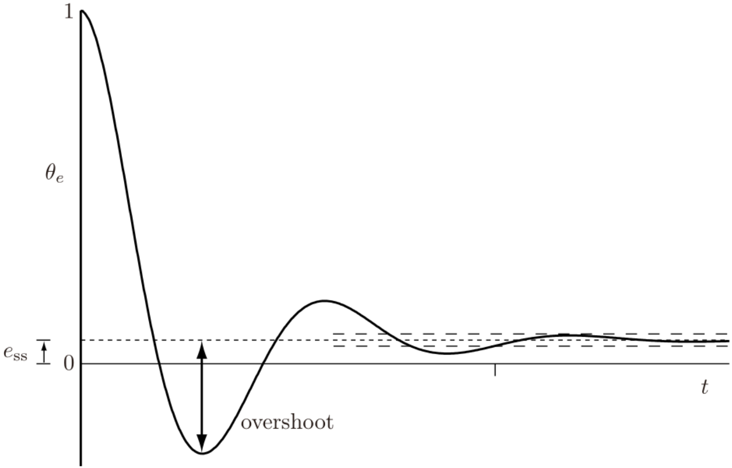
An ideal controller would drive the error to 0 instantly and keep it 0 forever.

- In practice, it takes time to reduce the error and a value of 0 might never be achieved.

Error Response - Characteristics of a Good Controller

An ideal controller would drive the error to 0 instantly and keep it 0 forever.

- In practice, it takes time to reduce the error and a value of 0 might never be achieved.



Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \to \infty$

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \to \infty$
- A *transient response*

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \rightarrow \infty$
- A *transient response*
  - $\rightarrow$ *overshoot*

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \to \infty$
- A *transient response*
  - $\rightarrow$ *overshoot*
  - $\rightarrow$ *settling time*

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \to \infty$
- A *transient response*
  - $\rightarrow$ *overshoot*
  - $\rightarrow$ *settling time*

A good controller achieves an error response $\theta_e(t)$ with:

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \to \infty$
- A *transient response*
  - $\rightarrow$ *overshoot*
  - $\rightarrow$ *settling time*

A good controller achieves an error response $\theta_e(t)$ with:

- small or no steady-state error

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \rightarrow \infty$
- A *transient response*
  - $\rightarrow$ *overshoot*
  - $\rightarrow$ *settling time*

A good controller achieves an error response $\theta_e(t)$ with:

- small or no steady-state error
- little or no overshoot

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \to \infty$
- A *transient response*
  - $\to$ *overshoot*
  - $\to$ *settling time*

A good controller achieves an error response $\theta_e(t)$ with:

- small or no steady-state error
- little or no overshoot
- a short settling time

Error Response - Characteristics of a good Controller (cont'd)

An error response $\theta_e(t)$ can be described by:

- A *steady-state response*: the error as time goes to infinity $t \rightarrow \infty$
- A *transient response*
  - $\rightarrow$ *overshoot*
  - $\rightarrow$ *settling time*

A good controller achieves an error response $\theta_e(t)$ with:

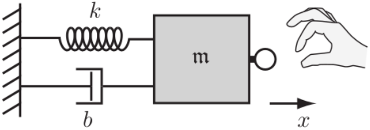
- small or no steady-state error
- little or no overshoot
- a short settling time
- stability: a steady state error is achieved (no oscillations)

A Surgical Robot - Impedance Control (Mass-Spring-Damper Example)

Simulation of a robot used as a haptic surgical simulator, mimicking the mass, stiffness and damping properties of a virtual surgical instrument in contact with virtual tissue.

A Surgical Robot - Impedance Control (Mass-Spring-Damper Example)

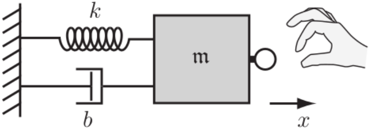
Simulation of a robot used as a haptic surgical simulator, mimicking the mass, stiffness and damping properties of a virtual surgical instrument in contact with virtual tissue.



- *surgical instrument* $\longrightarrow$ mass

A Surgical Robot - Impedance Control (Mass-Spring-Damper Example)

Simulation of a robot used as a haptic surgical simulator, mimicking the mass, stiffness and damping properties of a virtual surgical instrument in contact with virtual tissue.



- *surgical instrument* $\longrightarrow$ mass
- *tissue* $\longrightarrow$ spring

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

$$m\ddot{x} + b\dot{x} + kx = f \tag{2}$$

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

$$m\ddot{x} + b\dot{x} + kx = f \tag{2}$$

where:

- x: is the position

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

$$m\ddot{x} + b\dot{x} + kx = f \quad (2)$$

where:

- x: is the position
- m: is the mass (surgical instrument)

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

$$m\ddot{x} + b\dot{x} + kx = f \quad (2)$$

where:

- x: is the position
- m: is the mass (surgical instrument)
- b: is the damping

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

$$m\ddot{x} + b\dot{x} + kx = f \qquad (2)$$

where:

- x: is the position
- m: is the mass (surgical instrument)
- b: is the damping
- k: is the stiffness (of the tissue)

The Surgical Robot (cont'd)

The dynamics of a 1-degree of freedom robot rendering an impedance is described by:

$$m\ddot{x} + b\dot{x} + kx = f \tag{2}$$

where:

- x: is the position
- m: is the mass (surgical instrument)
- b: is the damping
- k: is the stiffness (of the tissue)
- f: is the force

The Surgical Robot (cont'd)

To simplify, in the case that the robot force $f = 0$, the above second order differential equation can be written in the equivalent form of an algebraic equation:

The Surgical Robot (cont'd)

To simplify, in the case that the robot force $f = 0$, the above second order differential equation can be written in the equivalent form of an algebraic equation:

$$ms^2 + bs + k = 0 \tag{3}$$

The Surgical Robot (cont'd)

To simplify, in the case that the robot force $f = 0$, the above second order differential equation can be written in the equivalent form of an algebraic equation:

$$ms^2 + bs + k = 0 \qquad (3)$$

- One can solve for s, called the *poles* of the system

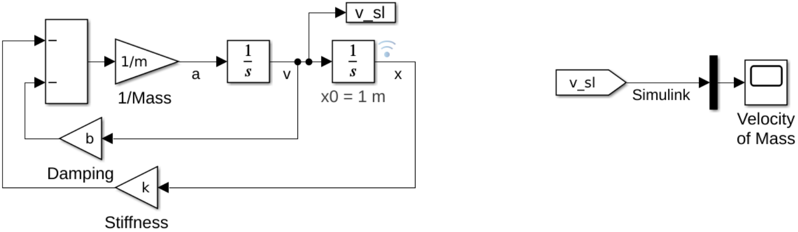
The Surgical Robot (cont'd)

To simplify, in the case that the robot force $f = 0$, the above second order differential equation can be written in the equivalent form of an algebraic equation:

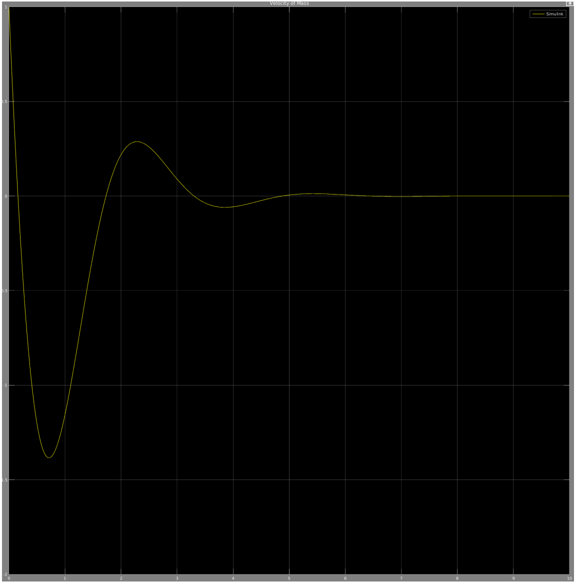
$$ms^2 + bs + k = 0 \qquad (3)$$

- One can solve for s, called the *poles* of the system
- The poles define the response (position) of the system as a function of time

Simulink Model of the Surgical Robot



Running the Simulink Model



Multi-joint Manipulator Robot Control - Independent Joint Control

Multi-joint Manipulator Robot Control - Independent Joint Control

A common robot control strategy for manipulators:

Multi-joint Manipulator Robot Control - Independent Joint Control

A common robot control strategy for manipulators:

- Each joint has its own controller

Multi-joint Manipulator Robot Control - Independent Joint Control

A common robot control strategy for manipulators:

- Each joint has its own controller
- Only a few parameters of the software controller need to be changed

Multi-joint Manipulator Robot Control - Independent Joint Control

A common robot control strategy for manipulators:

- Each joint has its own controller
- Only a few parameters of the software controller need to be changed
- The parameters changed are done in a way so that they can control the different size motors attached to each joint.

Discrete vs Continuous Dynamic Systems

- Discrete dynamic systems are described by difference equations.

Discrete vs Continuous Dynamic Systems

- Discrete dynamic systems are described by difference equations.

$$x(n) = 5 * x(n-1) + 6 * x(n-2) + 2$$

- Continuous dynamic systems are described by differential equations.

Discrete vs Continuous Dynamic Systems

- Discrete dynamic systems are described by difference equations.

$$x(n) = 5 * x(n-1) + 6 * x(n-2) + 2$$

- Continuous dynamic systems are described by differential equations.

$$\ddot{x} = 5 * \dot{x} + 10 * x + 10$$

Linear vs Non-Linear Control

Linear vs Non-Linear Control

Although in practice many robotic systems follow non-linear dynamics:

Linear vs Non-Linear Control

Although in practice many robotic systems follow non-linear dynamics:

- In many cases, a linear model can be developed

Linear vs Non-Linear Control

Although in practice many robotic systems follow non-linear dynamics:

- In many cases, a linear model can be developed
  - $\rightarrow$ It is much easier to develop controllers

Linear vs Non-Linear Control

Although in practice many robotic systems follow non-linear dynamics:

- In many cases, a linear model can be developed
  - $\rightarrow$ It is much easier to develop controllers
  - $\rightarrow$ It is much easier to analyse mathematically

Linear vs Non-Linear Control

Although in practice many robotic systems follow non-linear dynamics:

- In many cases, a linear model can be developed
  - $\rightarrow$ It is much easier to develop controllers
  - $\rightarrow$ It is much easier to analyse mathematically
  - $\rightarrow$ It is much easier to prove mathematically the stability

Linear vs Non-Linear Control

Although in practice many robotic systems follow non-linear dynamics:

- In many cases, a linear model can be developed
  - $\rightarrow$ It is much easier to develop controllers
  - $\rightarrow$ It is much easier to analyse mathematically
  - $\rightarrow$ It is much easier to prove mathematically the stability

Not possible to linearise complex dynamic systems!