6ELEN018W - Applied Robotics
Lecture 7: Robot Dynamics - Motion upon Forces - Part II

Dr Dimitris C. Dracopoulos

Last Lecture - Newtonian and Lagrangian Mechanics

- Calculate the dynamic equations of motion for a robot, subject to (generalised) forces, i.e. linear forces and angular forces (torques), one can use either Newtonian or Lagrangian mechanics.

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.
- The equations of motion can be derived using Newton's second law and Euler's equation of rotational motion or the Lagrangian energy-based method.

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.
- The equations of motion can be derived using Newton's second law and Euler's equation of rotational motion or the Lagrangian energy-based method.

The actuator forces (joint torques) $\boldsymbol{Q}$ can be written as a set of coupled differential equations:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w} \qquad (1)$$

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.
- The equations of motion can be derived using Newton's second law and Euler's equation of rotational motion or the Lagrangian energy-based method.

The actuator forces (joint torques) $\boldsymbol{Q}$ can be written as a set of coupled differential equations:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w} \tag{1}$$

- $\boldsymbol{q}$ are the joint coordinates (angles)

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.
- The equations of motion can be derived using Newton's second law and Euler's equation of rotational motion or the Lagrangian energy-based method.

The actuator forces (joint torques) $\boldsymbol{Q}$ can be written as a set of coupled differential equations:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w} \qquad (1)$$

- $\boldsymbol{q}$ are the joint coordinates (angles)
- $\dot{\boldsymbol{q}}$ are the joint velocities

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.
- The equations of motion can be derived using Newton's second law and Euler's equation of rotational motion or the Lagrangian energy-based method.

The actuator forces (joint torques) $\boldsymbol{Q}$ can be written as a set of coupled differential equations:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w} \qquad (1)$$

- $\boldsymbol{q}$ are the joint coordinates (angles)
- $\dot{\boldsymbol{q}}$ are the joint velocities
- $\ddot{\boldsymbol{q}}$ are the joint accelerations

Robot Manipulator Rigid Body Equations of Motion

Consider serial-link manipulator and the motor which actuates each joint j, $j \in \{0, \ldots, N\}$.

- The inertia that the motor experiences is a function (depends) of the configuration of the outward links $j_{i+1}, j_{i+2}, \ldots, j_N$.
- The equations of motion can be derived using Newton's second law and Euler's equation of rotational motion or the Lagrangian energy-based method.

The actuator forces (joint torques) $\boldsymbol{Q}$ can be written as a set of coupled differential equations:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w} \tag{1}$$

- $\boldsymbol{q}$ are the joint coordinates (angles)
- $\dot{\boldsymbol{q}}$ are the joint velocities
- $\ddot{\boldsymbol{q}}$ are the joint accelerations
- $\boldsymbol{g}$ is a term which represents the torque due to the gravity acting on the manipulator. This depends only on the configuration (joint angles $\boldsymbol{q}$)

Robot Manipulator Rigid Body Equations of Motion (cont'd)

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- $\boldsymbol{M}$ is the inertia matrix and depends only on the configuration of the robot (joint angles $\boldsymbol{q}$)

Robot Manipulator Rigid Body Equations of Motion (cont'd)

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- $\boldsymbol{M}$ is the inertia matrix and depends only on the configuration of the robot (joint angles $\boldsymbol{q}$)
- $\boldsymbol{C}$ is referred to as the Coriolis and centripetal term and this represents the gyroscopic and other forces that act on the robot joints due to the rotation of other robot joints.

Robot Manipulator Rigid Body Equations of Motion (cont'd)

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- $\boldsymbol{M}$ is the inertia matrix and depends only on the configuration of the robot (joint angles $\boldsymbol{q}$)
- $\boldsymbol{C}$ is referred to as the Coriolis and centripetal term and this represents the gyroscopic and other forces that act on the robot joints due to the rotation of other robot joints.
- $\boldsymbol{f}$ is the friction force

Robot Manipulator Rigid Body Equations of Motion (cont'd)

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- $\boldsymbol{M}$ is the inertia matrix and depends only on the configuration of the robot (joint angles $\boldsymbol{q}$)
- $\boldsymbol{C}$ is referred to as the Coriolis and centripetal term and this represents the gyroscopic and other forces that act on the robot joints due to the rotation of other robot joints.
- $\boldsymbol{f}$ is the friction force
- $\boldsymbol{J}(\boldsymbol{q})$ is the manipulator Jacobian, and $\boldsymbol{w} \in \mathbb{R}^6$ is the wrench (i.e. forces and torques) applied at the end-effector.

$\Longrightarrow$ This is the **inverse dynamics problem**:
Given the motion find the torques: $(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \rightarrow \boldsymbol{Q}$

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.
  - Use Netwon's second law for translational motion.

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.
  - Use Netwon's second law for translational motion.
  - Use Euler's law for rotational motion.

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.
  - Use Netwon's second law for translational motion.
  - Use Euler's law for rotational motion.
- Start at the base of the robot and work outwards to:

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.
  - Use Netwon's second law for translational motion.
  - Use Euler's law for rotational motion.
- Start at the base of the robot and work outwards to:
  - Determine the translational and angular velocity of the centre of mass for each link in turn.

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.
  - Use Netwon's second law for translational motion.
  - Use Euler's law for rotational motion.
- Start at the base of the robot and work outwards to:
  - Determine the translational and angular velocity of the centre of mass for each link in turn.
- Once we reach the end of the robot: start at the tip and work inwards:

The Newton-Euler Recursive Formula

Solve the equations of motion for the robot serial-link manipulator.
How it works?

- Determine the translational and rotational velocity and acceleration for the centre of mass of each link.
  - Use Netwon's second law for translational motion.
  - Use Euler's law for rotational motion.
- Start at the base of the robot and work outwards to:
  - Determine the translational and angular velocity of the centre of mass for each link in turn.
- Once we reach the end of the robot: start at the tip and work inwards:
  - Determine the force and moment each link exerts on the inboard link

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot

zero = np.zeros(6)
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot

zero = np.zeros(6)

# print nominal configuration
print(puma.qn)
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot

zero = np.zeros(6)

# print nominal configuration
print(puma.qn)

puma.plot(puma.qn)
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot

zero = np.zeros(6)

# print nominal configuration
print(puma.qn)

puma.plot(puma.qn)

Q = puma.rne(puma.qn, zero, zero)
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot

zero = np.zeros(6)

# print nominal configuration
print(puma.qn)

puma.plot(puma.qn)

Q = puma.rne(puma.qn, zero, zero)
```

The robot is not moving ($\boldsymbol{q} = 0, \dot{\boldsymbol{q}} = 0$), therefore these torques must be those required to hold the robot up against gravity.
Without gravity:

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

```
puma = models.DH.Puma560() # 6-joint robot

zero = np.zeros(6)

# print nominal configuration
print(puma.qn)

puma.plot(puma.qn)

Q = puma.rne(puma.qn, zero, zero)
```

The robot is not moving ($\boldsymbol{q} = 0, \dot{\boldsymbol{q}} = 0$), therefore these torques must be those required to hold the robot up against gravity.
Without gravity:

```
Q = puma.rne(puma.qn, zero, zero, gravity=[0, 0, 0])
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

Consider now a case where the robot is moving, joint 1 has a velocity of 1 rad/s^{-1}. In the absence of gravity, the required joint torques are:

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

Consider now a case where the robot is moving, joint 1 has a velocity of 1 rad/s^{-1}. In the absence of gravity, the required joint torques are:

```
puma.rne(puma.qn, [1, 0, 0, 0, 0, 0], zero, gravity=[0, 0, 0])
```

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

Consider now a case where the robot is moving, joint 1 has a velocity of 1 rad/s^{-1}. In the absence of gravity, the required joint torques are:

```
puma.rne(puma.qn, [1, 0, 0, 0, 0, 0], zero, gravity=[0, 0, 0])
```

The torque on joint 0 is that needed to overcome friction which always opposes the motion.

Using the Robotics Toolbox for the Newton-Euler Recursive Formula

Consider now a case where the robot is moving, joint 1 has a velocity of 1 rad/s^{-1}. In the absence of gravity, the required joint torques are:

```
puma.rne(puma.qn, [1, 0, 0, 0, 0, 0], zero, gravity=[0, 0, 0])
```

The torque on joint 0 is that needed to overcome friction which always opposes the motion. The nonzero torques need to be exerted on the joints to oppose the gyroscopic torques that joint 0 motion is exerting on those joints.

Gravity and Payload

The equations of motion (1) of the serial link robot manipulator:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \underline{\boldsymbol{g}(\boldsymbol{q})} + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- Gravity is the force that acts on the robot even if it's not moving.

Gravity and Payload

The equations of motion (1) of the serial link robot manipulator:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \underline{\boldsymbol{g}(\boldsymbol{q})} + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- Gravity is the force that acts on the robot even if it's not moving.
- The torque that counteracts gravity and stops the arm from collapsing under its own weight.

Gravity and Payload

The equations of motion (1) of the serial link robot manipulator:

$$\boldsymbol{Q} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \underline{\boldsymbol{g}(\boldsymbol{q})} + \boldsymbol{f}(\dot{\boldsymbol{q}}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{w}$$

- Gravity is the force that acts on the robot even if it's not moving.
- The torque that counteracts gravity and stops the arm from collapsing under its own weight.

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

- The last link in the chain of the robot has to hold the payload.

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

- The last link in the chain of the robot has to hold the payload.
- This propagates down the chain towards the base of the robot.

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

- The last link in the chain of the robot has to hold the payload.
- This propagates down the chain towards the base of the robot.
- All joints of the robot need to help hold up the payload to stop it being pulled down by the force of gravity.

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

- The last link in the chain of the robot has to hold the payload.
- This propagates down the chain towards the base of the robot.
- All joints of the robot need to help hold up the payload to stop it being pulled down by the force of gravity.

Effect of the payload:

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

- The last link in the chain of the robot has to hold the payload.
- This propagates down the chain towards the base of the robot.
- All joints of the robot need to help hold up the payload to stop it being pulled down by the force of gravity.

Effect of the payload:

- As the mass of the object (payload) increases:

$\Longrightarrow$ *One joint will hit its torque limit it will become overloaded. And that's the maximum payload that the robot can hold.*

Payload

A robot needs has to carry an object and place it somewhere else. This is the end-effector's payload.

- The last link in the chain of the robot has to hold the payload.
- This propagates down the chain towards the base of the robot.
- All joints of the robot need to help hold up the payload to stop it being pulled down by the force of gravity.

Effect of the payload:

- As the mass of the object (payload) increases:

$\Longrightarrow$ *One joint will hit its torque limit it will become overloaded. And that's the maximum payload that the robot can hold*.

- The maximum payload of the robot is a function of the torque capabilities of the motors but it is also a function of the configuration (angles) of the robot links.

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)
```

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)

# The default gravitational force in Earth
puma.gravity
```

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)

# The default gravitational force in Earth
puma.gravity


# Gravity in moon:
```

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)

# The default gravitational force in Earth
puma.gravity


# Gravity in moon:
print(puma.gravity / 6)
```

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)

# The default gravitational force in Earth
puma.gravity


# Gravity in moon:
print(puma.gravity / 6)

# Place the robot in moon
puma.gravity = puma.gravity / 6
```

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)

# The default gravitational force in Earth
puma.gravity


# Gravity in moon:
print(puma.gravity / 6)

# Place the robot in moon
puma.gravity = puma.gravity / 6

# In moon the torques required are reduced:
```

Calculating the Gravity load (torques) Using the Robotics Toolbox

```
Q = puma.gravload(puma.qn)

# The default gravitational force in Earth
puma.gravity


# Gravity in moon:
print(puma.gravity / 6)

# Place the robot in moon
puma.gravity = puma.gravity / 6

# In moon the torques required are reduced:

print(puma.gravload(puma.qn))
```