ROBOTICS: ADVANCED CONCEPTS & ANALYSIS

MODULE 7 - MOTION PLANNING AND CONTROL

Ashitava Ghosal\(^1\)

\(^1\)Department of Mechanical Engineering
&
Centre for Product Design and Manufacture
Indian Institute of Science
Bangalore 560 012, India
Email: asitava@mecheng.iisc.ernet.in

NPTEL, 2010
CONTENTS

LECTURE 1
- Motion planning

LECTURE 2
- Control of a single link

LECTURE 3
- Control of a multi-link serial manipulator

LECTURE 4*
- Control of constrained and parallel manipulator
- Cartesian control of serial manipulators

LECTURE 5*
- Force control of manipulators
- Hybrid position/force control of manipulators

LECTURE 6*
- Advanced topics in non-linear control of manipulators

MODULE 7 – ADDITIONAL MATERIAL
- Problems, References and Suggested Reading
OUTLINE

1 CONTENTS

2 LECTURE 1
   • Motion planning

3 LECTURE 2
   • Control of a single link

4 LECTURE 3
   • Control of a multi-link serial manipulator

5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators

6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators

7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
### Introduction

**Overview**

- Trajectory of a robot manipulator.
- Time history of position, velocity and acceleration of *actuated joints* or the *end-effector*.
- Algorithms for *planning* and *generation*.
- Main issues:
  - Ease and flexibility of planning.
  - Planned trajectories must be *sufficiently smooth* so as not to cause vibrations or jerky motion.
  - Efficient representation of trajectory in a computer and generation of desired trajectory in *real time*.
INTRODUCTION

Overview

- Trajectory of a robot manipulator.
- Time history of position, velocity and acceleration of *actuated joints* or the *end-effector*.
- Algorithms for *planning* and *generation*.
- Main issues:
  - Ease and flexibility of planning.
  - Planned trajectories must be *sufficiently smooth* so as not to cause vibrations or jerky motion.
  - Efficient representation of trajectory in a computer and generation of desired trajectory in *real time*. 

ASHITAVA GHOSAL (IISc)  ROBOTICS: ADVANCED CONCEPTS & ANALYSIS  NPTEL, 2010 4 / 129
Introduction

Overview

- Trajectory of a robot manipulator.
- Time history of position, velocity and acceleration of actuated joints or the end-effector.
- Algorithms for planning and generation.

Main issues:
- Ease and flexibility of planning.
- Planned trajectories must be sufficiently smooth so as not to cause vibrations or jerky motion.
- Efficient representation of trajectory in a computer and generation of desired trajectory in real time.
Introduction

Overview

- Trajectory of a robot manipulator.
- Time history of position, velocity and acceleration of actuated joints or the end-effector.
- Algorithms for planning and generation.
- Main issues:
  - Ease and flexibility of planning.
  - Planned trajectories must be sufficiently smooth so as not to cause vibrations or jerky motion.
  - Efficient representation of trajectory in a computer and generation of desired trajectory in real time.
Two main ways a robot trajectory is specified:

- *Joint space schemes* – time history of a single or multiple joints.
- *Cartesian space schemes* – time history of position and/or orientation of end-effector.

Initial and final points (in joint space or Cartesian space) is specified.

Initial and final *desired* velocity is often specified.

Often via or intermediate point(s) are specified with or without desired velocity at via point(s).

Most robots require at least $C^2$ trajectories – second derivative or acceleration is continuous between initial and final points.

Trajectory updates at rates between 50 and 200 Hz – Representation and computations of trajectories must be efficient – Not a very serious issue with modern processors!!
Two main ways a robot trajectory is specified:

- *Joint space schemes* – time history of a single or multiple joints.
- *Cartesian space schemes* – time history of position and/or orientation of end-effector.

Initial and final points (in joint space or Cartesian space) is specified.

- Initial and final *desired* velocity is often specified.
- Often via or intermediate point(s) are specified with or without desired velocity at via point(s).

Most robots require at least $C^2$ trajectories – second derivative or acceleration is continuous between initial and final points.

Trajectory updates at rates between 50 and 200 Hz – Representation and computations of trajectories must be efficient – Not a very serious issue with modern processors!!
Two main ways a robot trajectory is specified:

- *Joint space schemes* – time history of a single or multiple joints.
- *Cartesian space schemes* – time history of position and/or orientation of end-effector.

Initial and final points (in joint space or Cartesian space) is specified.

Initial and final *desired* velocity is often specified.

Often *via* or intermediate point(s) are specified with or without desired velocity at via point(s).

Most robots require at least $C^2$ trajectories – second derivative or acceleration is continuous between initial and final points.

Trajectory updates at rates between 50 and 200 Hz – Representation and computations of trajectories must be efficient – Not a very serious issue with modern processors!!
Two main ways a robot trajectory is specified:

- **Joint space schemes** – time history of a single or multiple joints.
- **Cartesian space schemes** – time history of position and/or orientation of end-effector.

Initial and final points (in joint space or Cartesian space) is specified.

Initial and final *desired* velocity is often specified.

Often *via* or intermediate point(s) are specified with or without desired velocity at via point(s).

Most robots require at least $C^2$ trajectories – second derivative or acceleration is continuous between initial and final points.

Trajectory updates at rates between 50 and 200 Hz – Representation and computations of trajectories must be efficient – Not a very serious issue with modern processors!!
Two main ways a robot trajectory is specified:

- *Joint space schemes* – time history of a single or multiple joints.
- *Cartesian space schemes* – time history of position and/or orientation of end-effector.

Initial and final points (in joint space or Cartesian space) is specified.

Initial and final *desired* velocity is often specified.

Often *via* or intermediate point(s) are specified with or without desired velocity at via point(s).

Most robots require at least $C^2$ trajectories – second derivative or acceleration is continuous between initial and final points.

Trajectory updates at rates between 50 and 200 Hz – Representation and computations of trajectories must be efficient – Not a very serious issue with modern processors!!
Two main ways a robot trajectory is specified:
- *Joint space schemes* – time history of a single or multiple joints.
- *Cartesian space schemes* – time history of position and/or orientation of end-effector.

Initial and final points (in joint space or Cartesian space) is specified.

Initial and final *desired* velocity is often specified.

Often via or intermediate point(s) are specified with or without desired velocity at via point(s).

Most robots require at least \( C^2 \) trajectories – second derivative or acceleration is continuous between initial and final points.

Trajectory updates at rates between 50 and 200 Hz – Representation and computations of trajectories must be efficient – Not a very serious issue with modern processors!!
JOINT SPACE SCHEMES

- Planning trajectory of $\theta_1 - \theta_1(t_0)$ to final $\theta_1(t_f) - t_0$, $t_f$ initial and final time.
- *Infinite* number of smooth curves can connect $\theta_i(t_0)$ to $\theta_i(t_f)$.
- **Interpolation** – Choosing a smooth curve between two points – Very well studied in CAD and Geometric Modeling.
- In robotics – simple polynomials $\rightarrow$ Simplest

$$\theta_1(t) = \frac{\theta_1(t_f) - \theta_1(t_0)}{t_f - t_0} (t - t_f) + \theta_1(t_f)$$

- Not very smooth!!
JOINT SPACE SCHEMES

- Planning trajectory of $\theta_1 - \theta_1(t_0)$ to final $\theta_1(t_f) - t_0$, $t_f$ initial and final time.
- Infinite number of smooth curves can connect $\theta_i(t_0)$ to $\theta_i(t_f)$.
- Interpolation – Choosing a smooth curve between two points – Very well studied in CAD and Geometric Modeling.
  - In robotics – simple polynomials $\rightarrow$ Simplest
    \[
    \theta_1(t) = \frac{\theta_1(t_f) - \theta_1(t_0)}{t_f - t_0}(t - t_f) + \theta_1(t_f)
    \]
  - Not very smooth!!
Joint Space Schemes

- Planning trajectory of $\theta_1 - \theta_1(t_0)$ to final $\theta_1(t_f) - t_0$, $t_f$ initial and final time.
- *Infinite* number of smooth curves can connect $\theta_i(t_0)$ to $\theta_i(t_f)$.
- **Interpolation** – Choosing a smooth curve between two points – Very well studied in *CAD* and *Geometric Modeling*.

- In robotics – simple polynomials $\rightarrow$ Simplest

$$\theta_1(t) = \frac{\theta_1(t_f) - \theta_1(t_0)}{t_f - t_0}(t - t_f) + \theta_1(t_f)$$

- Not very smooth!!
JOINT SPACE SCHEMES

- Planning trajectory of $\theta_1 - \theta_1(t_0)$ to final $\theta_1(t_f) - t_0$, $t_f$ initial and final time.

- *Infinite* number of smooth curves can connect $\theta_i(t_0)$ to $\theta_i(t_f)$.

- **Interpolation** – Choosing a smooth curve between two points – Very well studied in *CAD* and *Geometric Modeling*.

- In robotics – simple polynomials $\rightarrow$ Simplest

$$\theta_1(t) = \frac{\theta_1(t_f) - \theta_1(t_0)}{t_f - t_0} (t - t_f) + \theta_1(t_f)$$

- Not very smooth!!
Joint Space Schemes

- Planning trajectory of $\theta_1 - \theta_1(t_0)$ to final $\theta_1(t_f) - t_0$, $t_f$ initial and final time.
- Infinite number of smooth curves can connect $\theta_i(t_0)$ to $\theta_i(t_f)$.
- Interpolation – Choosing a smooth curve between two points – Very well studied in CAD and Geometric Modeling.
- In robotics – simple polynomials $\rightarrow$ Simplest

$$\theta_1(t) = \frac{\theta_1(t_f) - \theta_1(t_0)}{t_f - t_0}(t - t_f) + \theta_1(t_f)$$

- Not very smooth!!
JOINT SPACE SCHEMES

PIECE-WISE LINEAR

Figure 1: Piece-wise linear joint trajectory

- 4 piece-wise linear segment – trajectory through 3 via points.
- Sign changes in $\dot{\theta}_1(t)$ between segments.
- Plot of $\ddot{\theta}_1(t)$ even worse!!
- Not even $C^1$ continuity.

ASHITAVA GHOSAL (IISc)  ROBOTICS: ADVANCED CONCEPTS & ANALYSIS  NPTEL, 2010 7 / 129
**JOINT SPACE SCHEMES**

**CUBIC TRAJECTORY**

- Simplest polynomial trajectory with $\mathcal{C}^2$ continuity
- Cubic trajectory

$$\theta_1(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$  \hfill (1)

$a_0$, $a_1$, $a_2$ and $a_3$ are four constant coefficients.

- To obtain $a_0$, $a_1$, $a_2$ and $a_3$ use given $\theta_1$ and $\dot{\theta}_1$ at $t_0$ and $t_f$.

$$\begin{align*}
\theta_1(t_0) &= \theta_1(0), \quad \theta_1(t_f) = \theta_1(f) \\
\dot{\theta}_1(t_0) &= \dot{\theta}_1(0), \quad \dot{\theta}_1(t_f) = \dot{\theta}_1(f)
\end{align*}$$  \hfill (2)

- Four *linear* equations in four unknowns $a_0$, $a_1$, $a_2$ and $a_3$ – for $t_0 = 0$

$$\begin{align*}
a_0 &= \theta_1(0), \quad a_1 = \dot{\theta}_1(0) \\
a_2 &= \frac{3}{t_f^2} (\theta_1(f) - \theta_1(0)) - \frac{2}{t_f} \dot{\theta}_1(0) - \frac{1}{t_f} \dot{\theta}_1(f) \\
a_3 &= -\frac{2}{t_f^3} (\theta_1(f) - \theta_1(0)) + \frac{1}{t_f^2} (\dot{\theta}_1(0) + \dot{\theta}_1(f))
\end{align*}$$  \hfill (3)
JOINT SPACE SCHEMES

CUBIC TRAJECTORY

- Simplest polynomial trajectory with $C^2$ continuity
- Cubic trajectory

\[ \theta_1(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \]  

(1)

$a_0$, $a_1$, $a_2$ and $a_3$ are four constant coefficients.

- To obtain $a_0$, $a_1$, $a_2$ and $a_3$ use given $\theta_1$ and $\dot{\theta}_1$ at $t_0$ and $t_f$.

\[
\begin{align*}
\theta_1(t_0) &= \theta_1(0), & \theta_1(t_f) &= \theta_1(f) \\
\dot{\theta}_1(t_0) &= \dot{\theta}_1(0), & \dot{\theta}_1(t_f) &= \dot{\theta}_1(f)
\end{align*}
\]  

(2)

- Four linear equations in four unknowns $a_0$, $a_1$, $a_2$ and $a_3$ – for $t_0 = 0$

\[
\begin{align*}
a_0 &= \theta_1(0), & a_1 &= \dot{\theta}_1(0) \\
a_2 &= \frac{3}{t_f^2}(\theta_1(f) - \theta_1(0)) - \frac{2}{t_f} \dot{\theta}_1(0) - \frac{1}{t_f} \ddot{\theta}_1(f) \\
a_3 &= -\frac{2}{t_f^3}(\theta_1(f) - \theta_1(0)) + \frac{1}{t_f^2}(\dot{\theta}_1(0) + \ddot{\theta}_1(f))
\end{align*}
\]  

(3)
JOINT SPACE SCHEMES

CUBIC TRAJECTORY

- Simplest polynomial trajectory with \( C^2 \) continuity
- Cubic trajectory

\[
\theta_1(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{1}
\]

\( a_0, a_1, a_2 \) and \( a_3 \) are four constant coefficients.
- To obtain \( a_0, a_1, a_2 \) and \( a_3 \) use given \( \theta_1 \) and \( \dot{\theta}_1 \) at \( t_0 \) and \( t_f \).

\[
\begin{align*}
\theta_1(t_0) &= \theta_1(0), \quad \theta_1(t_f) = \theta_1(f) \\
\dot{\theta}_1(t_0) &= \dot{\theta}_1(0), \quad \dot{\theta}_1(t_f) = \dot{\theta}_1(f) 
\end{align*} \tag{2}
\]

- Four linear equations in four unknowns \( a_0, a_1, a_2 \) and \( a_3 \) – for \( t_0 = 0 \)

\[
\begin{align*}
a_0 &= \theta_1(0), \quad a_1 = \dot{\theta}_1(0) \\
a_2 &= \frac{3}{t_f^2}(\theta_1(f) - \theta_1(0)) - \frac{2}{t_f} \dot{\theta}_1(0) - \frac{1}{t_f} \ddot{\theta}_1(f) \tag{3} \\
a_3 &= -\frac{2}{t_f^3}(\theta_1(f) - \theta_1(0)) + \frac{1}{t_f^2}(\dot{\theta}_1(0) + \ddot{\theta}_1(f))
\end{align*}
\]
**JOINT SPACE SCHEMES**

**CUBIC TRAJECTORY**

- Simplest polynomial trajectory with $C^2$ continuity
- Cubic trajectory

$$\theta_1(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{1}$$

$a_0$, $a_1$, $a_2$ and $a_3$ are four constant coefficients.
- To obtain $a_0$, $a_1$, $a_2$ and $a_3$ use given $\theta_1$ and $\dot{\theta}_1$ at $t_0$ and $t_f$.

$$\begin{align*}
\theta_1(t_0) &= \theta_1(0), & \theta_1(t_f) &= \theta_1(f) \\
\dot{\theta}_1(t_0) &= \dot{\theta}_1(0), & \dot{\theta}_1(t_f) &= \dot{\theta}_1(f) \tag{2}
\end{align*}$$

- Four *linear* equations in four unknowns $a_0$, $a_1$, $a_2$ and $a_3$ – for $t_0 = 0$

$$\begin{align*}
a_0 &= \theta_1(0), & a_1 &= \dot{\theta}_1(0) \\
a_2 &= \frac{3}{t_f^2} (\theta_1(f) - \theta_1(0)) - \frac{2}{t_f} \dot{\theta}_1(0) - \frac{1}{t_f} \dot{\theta}_1(f) \tag{3} \\
a_3 &= -\frac{2}{t_f^3} (\theta_1(f) - \theta_1(0)) + \frac{1}{t_f^2} (\ddot{\theta}_1(0) + \ddot{\theta}_1(f))
\end{align*}$$
JOINT SPACE SCHEMES
CUBIC TRAJECTORY – NUMERICAL EXAMPLE

- Given $\theta_1(0) = 30^\circ$, $\theta_1(3) = 60^\circ$, $\dot{\theta}_1(0) = 10\text{deg/sec}$ and $\dot{\theta}_1(3) = -30\text{deg/sec}$.
- Cubic coefficients are $a_0 = 30$, $a_1 = 10$, $a_2 = 13.34$ and $a_3 = -4.45$.
- The expressions for $\theta_1(t)$
  
  $$\theta_1(t) = 30 + 10t + 13.34t^2 - 4.45t^3$$

- Continuous $\theta_1(t)$, $\dot{\theta}_1(t)$ and $\ddot{\theta}_1(t)$ between $t = 0$ and $t = 3$ seconds.

Figure 2: Cubic joint trajectory
Joint Space Schemes

Cubic Trajectory – Non-dimensional form

- $a_2$ and $a_3$ require division by $t_f^2$ and $t_f^3$ $\rightarrow$ error prone for large $t_f$.
- Use scaling of $t$ as in geometric modeling (Mortenson, 1985).
- Define $u = t/t_f$, $u \in [0,1]$ & derivative of $(\cdot)$ with respect to $u$ by $(\cdot)'$
- Cubic $- \theta_1(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$, coefficients of cubic are

\[
\begin{align*}
  a_0 &= \theta_1(0), \quad a_1 = \theta_1'(0) \\
  a_2 &= -3\theta_1(0) + 3\theta_1(1) - 2\theta_1'(0) - \theta_1'(1) \\
  a_3 &= 2\theta_1(0) - 2\theta_1(1) + \theta_1'(0) + \theta_1'(1)
\end{align*}
\]

and

\[
\begin{align*}
  \theta_1(u) &= (2u^3 - 3u^2 + 1)\theta_1(0) + (-2u^3 + 3u^2)\theta_1(1) + \\
  &\quad (u^3 - 2u^2 + u)\theta_1'(0) + (u^3 - u^2)\theta_1'(1)
\end{align*}
\]

(4)

- Compute $\theta_1(u)$ and transform back to $t$. 

Asitava Ghosal (IISc)  
Robotics: Advanced Concepts & Analysis  
NPTEL, 2010 10 / 129
**JOINT SPACE SCHEMES**

**Cubic Trajectory – Non-dimensional form**

- $a_2$ and $a_3$ require division by $t_f^2$ and $t_f^3$ → error prone for large $t_f$.
- Use scaling of $t$ as in geometric modeling (Mortenson, 1985).
- Define $u = t/t_f$, $u \in [0,1]$ & derivative of $(\cdot)$ with respect to $u$ by $(\cdot)'$
- Cubic $- \theta_1(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$, coefficients of cubic are
  
  $a_0 = \theta_1(0), \quad a_1 = \theta_1'(0)$
  $a_2 = -3\theta_1(0) + 3\theta_1(1) - 2\theta_1'(0) - \theta_1'(1)$
  $a_3 = 2\theta_1(0) - 2\theta_1(1) + \theta_1'(0) + \theta_1'(1)$

  and

  $\theta_1(u) = (2u^3 - 3u^2 + 1)\theta_1(0) + (-2u^3 + 3u^2)\theta_1(1) + (u^3 - 2u^2 + u)\theta_1'(0) + (u^3 - u^2)\theta_1'(1)$  \hspace{1cm} (4)

- Compute $\theta_1(u)$ and transform back to $t$. 
JOINT SPACE SCHEMES
Cubic Trajectory – Non-dimensional form

- $a_2$ and $a_3$ require division by $t_f^2$ and $t_f^3 \rightarrow$ error prone for large $t_f$.
- Use scaling of $t$ as in geometric modeling (Mortenson, 1985).
- Define $u = t/t_f$, $u \in [0,1]$ & derivative of $(\cdot)$ with respect to $u$ by $(\cdot)'$
- Cubic $- \theta_1(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$, coefficients of cubic are

\[
\begin{align*}
  a_0 & = \theta_1(0), \quad a_1 = \theta_1'(0) \\
  a_2 & = -3\theta_1(0) + 3\theta_1(1) - 2\theta_1'(0) - \theta_1'(1) \\
  a_3 & = 2\theta_1(0) - 2\theta_1(1) + \theta_1'(0) + \theta_1'(1)
\end{align*}
\]

and

\[
\theta_1(u) = (2u^3 - 3u^2 + 1)\theta_1(0) + (-2u^3 + 3u^2)\theta_1(1) + \\
(u^3 - 2u^2 + u)\theta_1'(0) + (u^3 - u^2)\theta_1'(1)
\] (4)

- Compute $\theta_1(u)$ and transform back to $t$. 
**Joint Space Schemes**

**Cubic Trajectory – Non-dimensional form**

- $a_2$ and $a_3$ require division by $t_f^2$ and $t_f^3 \rightarrow$ error prone for large $t_f$.
- Use scaling of $t$ as in geometric modeling (Mortenson, 1985).
- Define $u = t/t_f$, $u \in [0,1]$ & derivative of $(\cdot)$ with respect to $u$ by $(\cdot)'$
- Cubic $- \theta_1(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$, coefficients of cubic are

$$
\begin{align*}
  a_0 &= \theta_1(0), \quad a_1 = \theta_1'(0) \\
  a_2 &= -3\theta_1(0) + 3\theta_1(1) - 2\theta_1'(0) - \theta_1'(1) \\
  a_3 &= 2\theta_1(0) - 2\theta_1(1) + \theta_1'(0) + \theta_1'(1)
\end{align*}
$$

and

$$
\theta_1(u) = (2u^3 - 3u^2 + 1)\theta_1(0) + (-2u^3 + 3u^2)\theta_1(1) + (u^3 - 2u^2 + u)\theta_1'(0) + (u^3 - u^2)\theta_1'(1)
$$

(4)

- Compute $\theta_1(u)$ and transform back to $t$. 
**Joint Space Schemes**

**Cubic Trajectory – Non-dimensional form**

- $a_2$ and $a_3$ require division by $t_f^2$ and $t_f^3$ → error prone for large $t_f$.
- Use scaling of $t$ as in geometric modeling (Mortenson, 1985).
- Define $u = t / t_f$, $u \in [0, 1]$ & derivative of $(\cdot)$ with respect to $u$ by $(\cdot)'$.
- Cubic $- \theta_1(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$, coefficients of cubic are

  
  
  \[
  a_0 = \theta_1(0), \quad a_1 = \theta_1'(0) \\
  a_2 = -3\theta_1(0) + 3\theta_1(1) - 2\theta_1'(0) - \theta_1'(1) \\
  a_3 = 2\theta_1(0) - 2\theta_1(1) + \theta_1'(0) + \theta_1'(1)
  \]

  
  and

  
  \[
  \theta_1(u) = (2u^3 - 3u^2 + 1)\theta_1(0) + (-2u^3 + 3u^2)\theta_1(1) + \\
  (u^3 - 2u^2 + u)\theta_1'(0) + (u^3 - u^2)\theta_1'(1)
  \]

  (4)

- Compute $\theta_1(u)$ and transform back to $t$. 
Cubic can be written in nested form

\[ \theta_1(u) = a_0 + u(a_1 + u(a_2 + a_3 u)) \]

Once coefficients are computed ( offline and only once!) 
- Only 3 multiplications and 3 additions required for \( \theta_1(u) \)!
- Only 3 additional multiplications and 3 additions for \( \theta_1'(u) \) and \( \theta_1''(u) \)

For \( n \) jointed robot, multiply by \( n \) → Cubic joint space scheme very efficient!!

Cubic can satisfy at most 4 constraint → No control over initial and final acceleration!

Higher-order polynomial such as quintic for control of acceleration → more computations.

---

\(^1\)Advanced control of robots use desired position, velocity and acceleration (see Lecture 3 in this module).
**Joint Space Schemes**

**Cubic Trajectory – Non-dimensional form**

- Cubic can be written in nested form

\[ \theta_1(u) = a_0 + u(a_1 + u(a_2 + a_3u)) \]

- Once coefficients are computed (**offline and only once!**)
  - Only 3 multiplications and 3 additions required for \( \theta_1(u) \)!
  - Only 3 additional multiplications and 3 additions for \( \theta'_1(u) \) and \( \theta''_1(u) \)

- For \( n \) jointed robot, multiply by \( n \rightarrow \) Cubic joint space scheme very efficient!!

- Cubic can satisfy at most 4 constraint \( \rightarrow \) No control over initial and final acceleration!

- Higher-order polynomial such as **quintic** for control of acceleration \( \rightarrow \) more computations.

---

1 Advanced control of robots use desired position, velocity and acceleration (see Lecture 3 in this module).
JOINT SPACE SCHEMES
CUBIC TRAJECTORY – NON-DIMENSIONAL FORM

- Cubic can be written in nested form

\[ \theta_1(u) = a_0 + u(a_1 + u(a_2 + a_3u)) \]

- Once coefficients are computed (offline and only once!)
  - Only 3 multiplications and 3 additions required for \( \theta_1(u) \)!
  - Only 3 additional multiplications and 3 additions for \( \theta'_1(u) \) and \( \theta''_1(u) \)

- For \( n \) jointed robot, multiply by \( n \to \) Cubic joint space scheme very efficient!!

- Cubic can satisfy at most 4 constraint \( \to \) No control over initial and final acceleration!

- Higher-order polynomial such as quintic for control of acceleration \( \to \) more computations.

---

1Advanced control of robots use desired position, velocity and acceleration (see Lecture 3 in this module).
JOINT SPACE SCHEMES
CUBIC TRAJECTORY – NON-DIMENSIONAL FORM

- Cubic can be written in nested form

\[ \theta_1(u) = a_0 + u(a_1 + u(a_2 + a_3u)) \]

- Once coefficients are computed (offline and only once!)
  - Only 3 multiplications and 3 additions required for \( \theta_1(u) \)!
  - Only 3 additional multiplications and 3 additions for \( \theta_1'(u) \) and \( \theta_1''(u) \)

- For \( n \) jointed robot, multiply by \( n \) \( \rightarrow \) Cubic joint space scheme very efficient!!

- Cubic can satisfy at most 4 constraint \( \rightarrow \) No control over initial and final acceleration!

- Higher-order polynomial such as quintic for control of acceleration \( \rightarrow \) more computations.

\(^1\)Advanced control of robots use desired position, velocity and acceleration (see Lecture 3 in this module).
Cubic can be written in nested form

\[ \theta_1(u) = a_0 + u(a_1 + u(a_2 + a_3u)) \]

Once coefficients are computed (offline and only once!)
- Only 3 multiplications and 3 additions required for \( \theta_1(u) \) !
- Only 3 additional multiplications and 3 additions for \( \theta_1'(u) \) and \( \theta_1''(u) \)

For \( n \) jointed robot, multiply by \( n \) \( \rightarrow \) Cubic joint space scheme very efficient!!

Cubic can satisfy at most 4 constraint \( \rightarrow \) No control over initial and final acceleration!

Higher-order polynomial such as quintic for control of acceleration \( \rightarrow \) more computations.

\(^1\)Advanced control of robots use desired position, velocity and acceleration (see Lecture 3 in this module).
**Joint Space Schemes**

**Cubic Trajectory with via points**

- $k$ via points specified with one of two options:
  - Case 1: Velocities at the $k$ via point(s) specified.
  - Case 2: Velocities at the $k$ via point(s) *not* specified.

- Case 1: Plan trajectories for $k+1$ segments as $k+1$ cubics.
- Solve for $a_{0i}, a_{1i}, a_{2i},$ and $a_{3i}$ ($i = 1, 2, ..., k+1$) for each of the $k+1$ segments by using equation (3).
- $C^1$ continuity *ensured* – No control on acceleration, i.e. not $C^2$. 
Joint Space Schemes
Cubic Trajectory with via points

- $k$ via points specified with one of two options:
  - Case 1: Velocities at the $k$ via point(s) specified.
  - Case 2: Velocities at the $k$ via point(s) not specified.

- Case 1: Plan trajectories for $k + 1$ segments as $k + 1$ cubics.
  - Solve for $a_{0i}$, $a_{1i}$, $a_{2i}$, and $a_{3i}$ ($i = 1, 2, ..., k + 1$) for each of the $k + 1$ segments by using equation (3).
  - $C^1$ continuity ensured – No control on acceleration, i.e. not $C^2$. 
**JOINT SPACE SCHEMES**

**CUBIC TRAJECTORY WITH VIA POINTS**

- $k$ via points specified with one of two options:
  - Case 1: Velocities at the $k$ via point(s) specified.
  - Case 2: Velocities at the $k$ via point(s) *not* specified.

- Case 1: Plan trajectories for $k+1$ segments as $k+1$ cubics.
- Solve for $a_{0i}$, $a_{1i}$, $a_{2i}$, and $a_{3i}$ ($i = 1, 2, ..., k+1$) for each of the $k+1$ segments by using equation (3).

- $C^1$ continuity *ensured* – No control on acceleration, i.e. not $C^2$. 

---

**Asitava Ghosal (IISc) Robotics: Advanced Concepts & Analysis NPTEL, 2010**
**Joint Space Schemes**

**Cubic Trajectory with via points**

- $k$ via points specified with one of two options:
  - Case 1: Velocities at the $k$ via point(s) specified.
  - Case 2: Velocities at the $k$ via point(s) *not* specified.

- Case 1: Plan trajectories for $k+1$ segments as $k+1$ cubics.
- Solve for $a_{0i}$, $a_{1i}$, $a_{2i}$, and $a_{3i}$ ($i = 1, 2, ..., k + 1$) for each of the $k + 1$ segments by using equation (3).
- $C^1$ continuity *ensured* – No control on acceleration, i.e. not $C^2$. 
JOINT SPACE SCHEMES

CUBIC TRAJECTORY WITH VIA POINT - EXAMPLE

- $\theta_1(0) = 30^\circ$, $\theta_1(3) = 60^\circ$, $\dot{\theta}_1(0) = 10\text{deg/sec}$ and $\dot{\theta}_1(3) = -30\text{deg/sec}$.
- $\theta_1(2) = 55^\circ$, $\dot{\theta}_1(2) = -10\text{deg/sec}$
- For segment 1: $a_{01} = 30$, $a_{11} = 10$, $a_{21} = 13.75$ and $a_{31} = -6.25$
- For segment 2: $a_{02} = 55$, $a_{12} = -10$, $a_{22} = 65$ and $a_{32} = -50$

**Figure 3:** Cubic joint trajectory with via point

\[
\theta_1(t) = 30 + 10t + 13.75t^2 - 6.25t^3, \quad 0 \leq t \leq 2
\]
\[
\theta_1(t) = 55 - 10t + 65t^2 - 50t^3, \quad 2 \leq t \leq 3
\]

Clearly as expected $\ddot{\theta}_1(t)$ is discontinuous!
**Joint Space Schemes**

**Cubic Trajectory with via points: Case 2**

- *k* via points specified – Velocities at the *k* via point(s) *not* specified.
- Free choices can be used to match velocity and acceleration at via points.
- Two cubics, each $0 \leq t \leq t_f, \ i = 1, 2$

$$\theta_1(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3, \ i = 1, 2$$

- From given initial, final, via point, and the initial and final velocities

  $$\theta_1(0) = a_{01}, \ \dot{\theta}_1(0) = a_{11}$$

  $$\theta_1(v) = a_{01} + a_{11}t_{f_1} + a_{21}t_{f_1}^2 + a_{31}t_{f_1}^3, \ \theta_1(v) = a_{02}$$

  $$\theta_1(f) = a_{02} + a_{12}t_{f_2} + a_{22}t_{f_2}^2 + a_{32}t_{f_2}^3$$

  $$\dot{\theta}_1(f) = a_{12} + 2a_{22}t_{f_2} + 3a_{32}t_{f_2}^2$$

  $$a_{12} = a_{11} + 2a_{21}t_{f_1} + 3a_{31}t_{f_1}^2, \ 2a_{22} = 2a_{21} + 6a_{31}t_{f_1}$$

- 8 equations in 8 unknowns $\rightarrow$ solve for 8 coefficients of 2 cubics
**Joint Space Schemes**

**Cubic Trajectory with via points: Case 2**

- $k$ via points specified – Velocities at the $k$ via point(s) *not* specified.
- Free choices can be used to match velocity and acceleration at via points.
- Two cubics, each $0 \leq t \leq t_f$, $i = 1, 2$

$$
\theta_1(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3, \ i = 1, 2
$$

- From given initial, final, via point, and the initial and final velocities

$$
\begin{align*}
\theta_1(0) &= a_{01}, \ \dot{\theta}_1(0) = a_{11} \\
\theta_1(v) &= a_{01} + a_{11}t_{f_1} + a_{21}t_{f_1}^2 + a_{31}t_{f_1}^3, \ \theta_1(v) = a_{02} \\
\theta_1(f) &= a_{02} + a_{12}t_{f_2} + a_{22}t_{f_2}^2 + a_{32}t_{f_2}^3 \\
\dot{\theta}_1(f) &= a_{12} + 2a_{22}t_{f_2} + 3a_{32}t_{f_2}^2 \\
\end{align*}
$$

$$
\begin{align*}
a_{12} &= a_{11} + 2a_{21}t_{f_1} + 3a_{31}t_{f_1}^2, \ 2a_{22} &= 2a_{21} + 6a_{31}t_{f_1} \\
\end{align*}
$$

- 8 equations in 8 unknowns $\rightarrow$ solve for 8 coefficients of 2 cubics
J O I N T  S P A C E  S C H E M E S


- $k$ via points specified – Velocities at the $k$ via point(s) not specified.
- Free choices can be used to match velocity and acceleration at via points.
- Two cubics, each $0 \leq t \leq t_i$, $i = 1, 2$

$$\theta_1(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3, \ i = 1, 2$$

- From given initial, final, via point, and the initial and final velocities

$$\theta_1(0) = a_{01}, \dot{\theta}_1(0) = a_{11}$$
$$\theta_1(v) = a_{01} + a_{11}t_{f_1} + a_{21}t_{f_1}^2 + a_{31}t_{f_1}^3, \ \theta_1(v) = a_{02}$$
$$\theta_1(f) = a_{02} + a_{12}t_{f_2} + a_{22}t_{f_2}^2 + a_{32}t_{f_2}^3$$
$$\dot{\theta}_1(f) = a_{12} + 2a_{22}t_{f_2} + 3a_{32}t_{f_2}^2$$
$$a_{12} = a_{11} + 2a_{21}t_{f_1} + 3a_{31}t_{f_1}^2, \ 2a_{22} = 2a_{21} + 6a_{31}t_{f_1}$$

- 8 equations in 8 unknowns $\rightarrow$ solve for 8 coefficients of 2 cubics
JOINT SPACE SCHEMES

CUBIC TRAJECTORY WITH VIA POINTS: CASE 2

- $k$ via points specified – Velocities at the $k$ via point(s) *not* specified.
- Free choices can be used to match velocity and acceleration at via points.
- Two cubics, each $0 \leq t \leq t_{fi}, \ i = 1, 2$

$$
\theta_1(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3, \ i = 1, 2
$$

- From given initial, final, via point, and the initial and final velocities

\begin{align*}
\theta_1(0) &= a_{01}, \quad \dot{\theta}_1(0) = a_{11} \\
\theta_1(v) &= a_{01} + a_{11}t_{f1} + a_{21}t_{f1}^2 + a_{31}t_{f1}^3, \quad \theta_1(v) = a_{02} \\
\theta_1(f) &= a_{02} + a_{12}t_{f2} + a_{22}t_{f2}^2 + a_{32}t_{f2}^3 \\
\dot{\theta}_1(f) &= a_{12} + 2a_{22}t_{f2} + 3a_{32}t_{f2}^2 \\
a_{12} &= a_{11} + 2a_{21}t_{f1} + 3a_{31}t_{f1}^2, \quad 2a_{22} = 2a_{21} + 6a_{31}t_{f1}
\end{align*}

- 8 equations in 8 unknowns $\rightarrow$ solve for 8 coefficients of 2 cubics
JOINT SPACE SCHEMES

CUBIC TRAJECTORY WITH VIA POINTS: CASE 2

- \( k \) via points specified – Velocities at the \( k \) via point(s) *not* specified.
- Free choices can be used to match velocity and acceleration at via points.
- Two cubics, each \( 0 \leq t \leq t_{f_i} \), \( i = 1, 2 \)
  \[
  \theta_1(t) = a_{0i} + a_{1i}t + a_{2i}t^2 + a_{3i}t^3, \quad i = 1, 2
  \]
- From given initial, final, via point, and the initial and final velocities
  \[
  \theta_1(0) = a_{01}, \quad \dot{\theta}_1(0) = a_{11}
  \]
  \[
  \theta_1(v) = a_{01} + a_{11}t_{f_1} + a_{21}t_{f_1}^2 + a_{31}t_{f_1}^3, \quad \theta_1(v) = a_{02}
  \]
  \[
  \theta_1(f) = a_{02} + a_{12}t_{f_2} + a_{22}t_{f_2}^2 + a_{32}t_{f_2}^3
  \]
  \[
  \dot{\theta}_1(f) = a_{12} + 2a_{22}t_{f_2} + 3a_{32}t_{f_2}^2
  \]
  \[
  a_{12} = a_{11} + 2a_{21}t_{f_1} + 3a_{31}t_{f_1}^2, \quad 2a_{22} = 2a_{21} + 6a_{31}t_{f_1}
  \]
- 8 equations in 8 unknowns \( \rightarrow \) solve for 8 coefficients of 2 cubics
JOINT SPACE SCHEMES

CUBIC TRAJECTORY WITH VIA POINT - CASE 2 EXAMPLE

\[ \theta_1(0) = 30^\circ, \theta_1(3) = 60^\circ, \]
\[ \dot{\theta}_1(0) = 10\text{deg/sec}, \]
\[ \dot{\theta}_1(3) = -30\text{deg/sec}, \text{ and} \]
\[ \theta_1(2) = 55^\circ. \]

- For segment 1: \( a_{01} = 30, a_{11} = 10, \)
  \( a_{21} = -1.04 \) and \( a_{31} = 1.15 \)
- For segment 2: \( a_{02} = 55, \)
  \( a_{12} = 19.58, a_{22} = 5.83 \) and \( a_{32} = -20.42 \)

Figure 4: Cubic joint trajectory with continuous acceleration

- Clearly as expected \( \dot{\theta}_1(t) \) and \( \ddot{\theta}_1(t) \) are continuous!
- For \( k \) via points \( 4 + 4k \) equations – sparse matrix and can be solved!!
Cartesian Space Schemes

Overview

- Joint space schemes useful if a joint or a group of joints are to be moved.
- Motion of end-effector $\rightarrow$ motion planning in terms of position and orientation $\rightarrow$ Cartesian Space schemes or motion planning.
  - More natural for the robot operator to specify.
  - Easier to see, visualize and check for obstacles.
  - Difficulty in planning orientation due to representation issues (See Module 2, Lecture 1).
- Traditionally two important Cartesian space paths used for position.
  - Linear interpolation – straight line path between two given positions
  - Circular interpolation – circular arcs between three given positions.
- All paths must be $C^2$ continuous in time $t$. 
**Cartesian Space Schemes**

**Overview**

- Joint space schemes useful if a joint or a group of joints are to be moved.
- Motion of end-effector $\rightarrow$ motion planning in terms of *position* and *orientation* $\rightarrow$ Cartesian Space schemes or motion planning.
  - More natural for the robot operator to specify.
  - Easier to see, visualize and check for obstacles.
  - Difficulty in planning orientation due to representation issues (See Module 2, Lecture 1).

Traditionally two important Cartesian space paths used for position.

- Linear interpolation – straight line path between two given positions
- Circular interpolation – circular arcs between three given positions.

All paths must be $C^2$ continuous in time $t$. 
Cartesian Space Schemes
Overview

- Joint space schemes useful if a joint or a group of joints are to be moved.
- Motion of end-effector $\rightarrow$ motion planning in terms of position and orientation $\rightarrow$ Cartesian Space schemes or motion planning.
  - More natural for the robot operator to specify.
  - Easier to see, visualize and check for obstacles.
  - Difficulty in planning orientation due to representation issues (See Module 2, Lecture 1).
- Traditionally two important Cartesian space paths used for position.
  - Linear interpolation – straight line path between two given positions
  - Circular interpolation – circular arcs between three given positions.
- All paths must be $C^2$ continuous in time $t$. 

Joint space schemes useful if a joint or a group of joints are to be moved.

Motion of end-effector → motion planning in terms of position and orientation → Cartesian Space schemes or motion planning.

- More natural for the robot operator to specify.
- Easier to see, visualize and check for obstacles.
- Difficulty in planning orientation due to representation issues (See Module 2, Lecture 1).

Traditionally two important Cartesian space paths used for position.

- Linear interpolation – straight line path between two given positions
- Circular interpolation – circular arcs between three given positions.

All paths must be $C^2$ continuous in time $t$. 
**Cartesian Space Schemes**

**Straight Line Motion**

- Given \((x_0, y_0, z_0)^T, (\dot{x}_0, \dot{y}_0, \dot{z}_0)^T\) \& \((x_f, y_f, z_f)^T, (\dot{x}_f, \dot{y}_f, \dot{z}_f)^T\)
- Equation of a straight line in the 3D Cartesian space

\[
\begin{align*}
y(t) &= \left(\frac{y_f - y_0}{x_f - x_0}\right)(x(t) - x_f) + y_f \\
z(t) &= \left(\frac{z_f - z_0}{x_f - x_0}\right)(x(t) - x_f) + z_f
\end{align*}
\]  \hspace{1cm} (5)

- Plan smooth cubic trajectory for \(x(t)\) as \(x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3\)
- Compute coefficients of cubic from given initial and final conditions

\[
\begin{align*}
a_0 &= x_0, \quad a_1 = \dot{x}_0 \\
a_2 &= \frac{3}{t_f^2}(x_f - x_0) - \frac{2}{t_f} \dot{x}_0 - \frac{1}{t_f} \dot{x}_f \\
a_3 &= -\frac{2}{t_f^3}(x_f - x_0) + \frac{1}{t_f^2} (\dot{x}_0 + \dot{x}_f)
\end{align*}
\]  \hspace{1cm} (6)

- Compute \(y(t)\) and \(z(t)\) from equation (5) \(\rightarrow x(t), y(t)\) and \(z(t)\) are all \(C^2\).
**Cartesian Space Schemes**

**Straight Line Motion**

- Given \((x_0, y_0, z_0)^T, (\dot{x}_0, \dot{y}_0, \dot{z}_0)^T\) & \((x_f, y_f, z_f)^T, (\dot{x}_f, \dot{y}_f, \dot{z}_f)^T\)
- Equation of a straight line in the 3D Cartesian space

\[
\begin{align*}
y(t) &= \left(\frac{y_f - y_0}{x_f - x_0}\right)(x(t) - x_f) + y_f \\
z(t) &= \left(\frac{z_f - z_0}{x_f - x_0}\right)(x(t) - x_f) + z_f
\end{align*}
\]  

(5)

- Plan smooth cubic trajectory for \(x(t)\) as \(x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3\)
- Compute coefficients of cubic from given initial and final conditions

\[
\begin{align*}
a_0 &= x_0, \quad a_1 = \dot{x}_0 \\
a_2 &= \frac{3}{t_f^2} (x_f - x_0) - \frac{2}{t_f} \dot{x}_0 - \frac{1}{t_f^2} \ddot{x}_f \\
a_3 &= -\frac{2}{t_f^3} (x_f - x_0) + \frac{1}{t_f^2} (\ddot{x}_0 + \ddot{x}_f)
\end{align*}
\]  

(6)

- Compute \(y(t)\) and \(z(t)\) from equation (5) \(\rightarrow x(t), y(t)\) and \(z(t)\) are all \(C^2\).
Cartesian Space Schemes

Straight Line Motion

- Given \((x_0, y_0, z_0)^T, (\dot{x}_0, \dot{y}_0, \dot{z}_0)^T\) & \((x_f, y_f, z_f)^T, (\dot{x}_f, \dot{y}_f, \dot{z}_f)^T\)
- Equation of a straight line in the 3D Cartesian space

\[
y(t) = \left(\frac{y_f - y_0}{x_f - x_0}\right)(x(t) - x_f) + y_f
\]
\[
z(t) = \left(\frac{z_f - z_0}{x_f - x_0}\right)(x(t) - x_f) + z_f
\]

(5)

- Plan smooth cubic trajectory for \(x(t)\) as \(x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3\)
- Compute coefficients of cubic from given initial and final conditions

\[
a_0 = x_0, \quad a_1 = \dot{x}_0
\]
\[
a_2 = \frac{3}{t_f^2}(x_f - x_0) - \frac{2}{t_f} \ddot{x}_0 - \frac{1}{t_f} \dddot{x}_f
\]
\[
a_3 = -\frac{2}{t_f^3}(x_f - x_0) + \frac{1}{t_f^2} (\dot{x}_0 + \dot{x}_f)
\]

(6)

- Compute \(y(t)\) and \(z(t)\) from equation (5) \(\Rightarrow x(t), y(t)\) and \(z(t)\) are all \(C^2\).
**Cartesian Space Schemes**

**Straight Line Motion**

- Given \((x_0, y_0, z_0)^T, (\dot{x}_0, \dot{y}_0, \dot{z}_0)^T\) & \((x_f, y_f, z_f)^T, (\ddot{x}_f, \ddot{y}_f, \ddot{z}_f)^T\)
- Equation of a straight line in the 3D Cartesian space

\[
y(t) = \left(\frac{y_f - y_0}{x_f - x_0}\right)(x(t) - x_f) + y_f \\
z(t) = \left(\frac{z_f - z_0}{x_f - x_0}\right)(x(t) - x_f) + z_f
\]  

(5)

- Plan smooth cubic trajectory for \(x(t)\) as \(x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3\)
- Compute coefficients of cubic from given initial and final conditions

\[
a_0 = x_0, \quad a_1 = \dot{x}_0 \\
a_2 = \frac{3}{t_f^2}(x_f - x_0) - \frac{2}{t_f} \dot{x}_0 - \frac{1}{t_f} \ddot{x}_f \\
a_3 = -\frac{2}{t_f^3}(x_f - x_0) + \frac{1}{t_f^2}(\dot{x}_0 + \ddot{x}_f)
\]  

(6)

- Compute \(y(t)\) and \(z(t)\) from equation (5) \(\rightarrow x(t), y(t)\) and \(z(t)\) are all \(C^2\)
**Cartesian Space Schemes**

**Straight Line Motion**

- Given \((x_0, y_0, z_0)^T, (\dot{x}_0, \dot{y}_0, \dot{z}_0)^T\) & \((x_f, y_f, z_f)^T, (\dot{x}_f, \dot{y}_f, \dot{z}_f)^T\)
- Equation of a straight line in the 3D Cartesian space
  
  \[
  y(t) = \left(\frac{y_f - y_0}{x_f - x_0}\right)(x(t) - x_f) + y_f \\
  z(t) = \left(\frac{z_f - z_0}{x_f - x_0}\right)(x(t) - x_f) + z_f
  \]  
  \[ (5) \]

- Plan smooth cubic trajectory for \(x(t)\) as \(x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3\)
- Compute coefficients of cubic from given initial and final conditions
  
  \[
  a_0 = x_0, \quad a_1 = \dot{x}_0  \\
  a_2 = \frac{3}{t_f^2}(x_f - x_0) - \frac{2}{t_f}\dot{x}_0 - \frac{1}{t_f}\ddot{x}_f  \\
  a_3 = -\frac{2}{t_f^3}(x_f - x_0) + \frac{1}{t_f^2}(\dot{x}_0 + \ddot{x}_f)
  \]  
  \[ (6) \]

- Compute \(y(t)\) and \(z(t)\) from equation \((5)\) \(\rightarrow\) \(x(t), y(t)\) and \(z(t)\) are all \(C^2\).
For smoothness circular arcs as opposed to piece-wise straight lines are desired.

Given points $^0p_1$, $^0p_2$, $^0p_3$, in $\mathbb{R}^3$, and velocities at these points.

Algorithm for circular interpolation

1. Compute the normal to the plane as
   
   $$^0\hat{n} = \frac{(^0p_2 - ^0p_1) \times (^0p_3 - ^0p_1)}{|(^0p_2 - ^0p_1) \times (^0p_3 - ^0p_1)|}$$

2. Compute $^0\hat{X}$, $^0\hat{Y}$ and $^0\hat{Z}$ as
   
   $$^0\hat{Z} = ^0\hat{n}$$
   $$^0\hat{X} = \frac{(^0p_2 - ^0p_1)}{|(^0p_2 - ^0p_1)|}$$
   $$^0\hat{Y} = ^0\hat{n} \times ^0\hat{X}$$

   to define coordinate system $\{CIRC\}$.

3. Obtain rotation matrix $^0_{CIRC}[R]$ with $^0\hat{X}$, $^0\hat{Y}$ and $^0\hat{n}$. 
 CARTESIAN SPACE SCHEMES

CIRCULAR MOTION

- For smoothness circular arcs as opposed to piece-wise straight lines are desired.
- Given points $0\mathbf{p}_1$, $0\mathbf{p}_2$, $0\mathbf{p}_3$, in $\mathbb{R}^3$, and velocities at these points.
- Algorithm for circular interpolation
  - Compute the normal to the plane as
    $$0\hat{n} = \frac{(0\mathbf{p}_2 - 0\mathbf{p}_1) \times (0\mathbf{p}_3 - 0\mathbf{p}_1)}{|(0\mathbf{p}_2 - 0\mathbf{p}_1) \times (0\mathbf{p}_3 - 0\mathbf{p}_1)|}$$
  - Compute $0\hat{X}$, $0\hat{Y}$ and $0\hat{Z}$ as
    $$0\hat{Z} = 0\hat{n}$$
    $$0\hat{X} = \frac{(0\mathbf{p}_2 - 0\mathbf{p}_1)}{|(0\mathbf{p}_2 - 0\mathbf{p}_1)|}$$
    $$0\hat{Y} = 0\hat{n} \times 0\hat{X}$$
  - to define coordinate system $\{CIRC\}$.
  - Obtain rotation matrix $0_{CIRC}[R]$ with $0\hat{X}$, $0\hat{Y}$ and $0\hat{n}$. 
For smoothness circular arcs as opposed to piece-wise straight lines are desired.

Given points $0p_1$, $0p_2$, $0p_3$, in $\mathbb{R}^3$, and velocities at these points.

**Algorithm for circular interpolation**

- Compute the normal to the plane as

$$0\hat{n} = \frac{(0p_2 - 0p_1) \times (0p_3 - 0p_1)}{|(0p_2 - 0p_1) \times (0p_3 - 0p_1)|}$$

- Compute $0\hat{X}$, $0\hat{Y}$ and $0\hat{Z}$ as

$$0\hat{Z} = 0\hat{n}$$

$$0\hat{X} = \frac{(0p_2 - 0p_1)}{|(0p_2 - 0p_1)|}$$

$$0\hat{Y} = 0\hat{n} \times 0\hat{X}$$

- to define coordinate system $\{CIRC\}$.

- Obtain rotation matrix $0_{CIRC}[R]$ with $0\hat{X}$, $0\hat{Y}$ and $0\hat{n}$. 
Algorithm for circular interpolation (Contd.)

- Transform $^0p_1$, $^0p_2$, $^0p_3$ to $\{CIRC\}$ using $^0_{CIRC}[R]$.
- In $\{CIRC\}$ points become $(x_1, y_1, c)$, $(x_2, y_2, c)$ and $(x_3, y_3, c)$.
- Compute centre, $(a, b)$, and radius $r$ of the circular arc in $\{CIRC\}$.
- Compute angle made by line from centre to 3 points with $\hat{X}$ axis in $\{CIRC\}$. Denote by $\phi_1$, $\phi_2$ and $\phi_3$.
- Plan a $C^2$ (cubic trajectory) for $\phi(t)$ such that $\phi_1$, $\phi_2$ and $\phi_3$ are reached at the specified $t$ and order – joint space trajectory with via points.
- Circular arc in $\{CIRC\}$ described by

$$
\begin{align*}
  x(t) &= a + r \cos(\phi(t)) \\
  y(t) &= b + r \sin(\phi(t)), \ z(t) = c
\end{align*}
$$

- Since $\phi(t)$ is $C^2 \rightarrow x(t)$, $y(t)$ and $z(t)$ is $C^2$.
- To obtain path of end-effector in $\{0\}$ use $^0_{CIRC}[R]$.

Alternate: use inverse kinematics and plan trajectory in joint space $\rightarrow$ Approximate straight line or circular trajectory in Cartesian space.
Circular Motion (Contd.)

Algorithm for circular interpolation (Contd.)

- Transform $^0p_1$, $^0p_2$, $^0p_3$ to $\{CIRC\}$ using $^0_{CIRC}[R]$.
- In $\{CIRC\}$ points become $(x_1, y_1, c)$, $(x_2, y_2, c)$ and $(x_3, y_3, c)$.
- Compute centre, $(a, b)$, and radius $r$ of the circular arc in $\{CIRC\}$.
- Compute angle made by line from centre to 3 points with $\hat{X}$ axis in $\{CIRC\}$. Denote by $\phi_1$, $\phi_2$ and $\phi_3$.
- Plan a $C^2$ (cubic trajectory) for $\phi(t)$ such that $\phi_1$, $\phi_2$ and $\phi_3$ are reached at the specified $t$ and order – joint space trajectory with via points.
- Circular arc in $\{CIRC\}$ described by

\[
\begin{align*}
x(t) &= a + r \cos(\phi(t)) \\
y(t) &= b + r \sin(\phi(t)), \\
z(t) &= c
\end{align*}
\]

- Since $\phi(t)$ is $C^2 \rightarrow x(t)$, $y(t)$ and $z(t)$ is $C^2$.
- To obtain path of end-effector in $\{0\}$ use $^0_{CIRC}[R]$.

Alternate: use inverse kinematics and plan trajectory in joint space $\rightarrow$ Approximate straight line or circular trajectory in Cartesian space.
**Cartesian Space Schemes**

**Trajectory Planning for Orientation**

- Various representation of orientation (see Module 2, Lecture 1)— all with their own advantages and disadvantages!!
- Euler parameters (see Module 2, Lecture 1) – 4 parameters + 1 constraint.
  - Given: \((0 \varepsilon_{Tool}(0), \varepsilon_4(0))^T\) and \((0 \varepsilon_{Tool}(t_f), \varepsilon_4(t_f))^T\).
  - Constraint: \(\varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 + \varepsilon_4^2 = 1\)
  - Interpolation **must** satisfy constraint at all \(t\). 
- Given: Initial angular velocity \(0 \omega_{Tool}(0)\) and final angular velocity of end-effector \(0 \omega_{Tool}(t_f)\).
- Need relationship between angular velocity and Euler parameters – not as simple as \(x(t)\) and \(\dot{x}(t)\)!
Various representation of orientation (see Module 2, Lecture 1)— all with their own advantages and disadvantages!!

Euler parameters (see Module 2, Lecture 1) – 4 parameters + 1 constraint.
- Given: $(0\varepsilon_{Tool}(0),\varepsilon_4(0))^T$ and $(0\varepsilon_{Tool}(t_f),\varepsilon_4(t_f))^T$.
- Constraint: $\varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 + \varepsilon_4^2 = 1$
- Interpolation must satisfy constraint at all $t$.

Given: Initial angular velocity $^0\omega_{Tool}(0)$ and final angular velocity of end-effector $^0\omega_{Tool}(t_f)$.

Need relationship between angular velocity and Euler parameters – not as simple as $x(t)$ and $\dot{x}(t)$!
Various representation of orientation (see Module 2, Lecture 1) – all with their own advantages and disadvantages!!

Euler parameters (see Module 2, Lecture 1) – 4 parameters + 1 constraint.
  - Given: \((0 \varepsilon_{Tool}(0), \varepsilon_4(0))^T\) and \((0 \varepsilon_{Tool}(t_f), \varepsilon_4(t_f))^T\).
  - Constraint: \(\varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 + \varepsilon_4^2 = 1\)
  - Interpolation must satisfy constraint at all \(t\).

Given: Initial angular velocity \(0 \omega_{Tool}(0)\) and final angular velocity of end-effector \(0 \omega_{Tool}(t_f)\).

Need relationship between angular velocity and Euler parameters – not as simple as \(x(t)\) and \(\dot{x}(t)\)!
Various representation of orientation (see Module 2, Lecture 1) – all with their own advantages and disadvantages!!

Euler parameters (see Module 2, Lecture 1) – 4 parameters + 1 constraint.
- Given: \( (0 \varepsilon_{Tool}(0), \varepsilon_4(0))^T \) and \( (0 \varepsilon_{Tool}(t_f), \varepsilon_4(t_f))^T \).
- Constraint: \( \varepsilon_1^2 + \varepsilon_2^2 + \varepsilon_3^2 + \varepsilon_4^2 = 1 \)
- Interpolation must satisfy constraint at all \( t \).

Given: Initial angular velocity \( {0 \omega}_{Tool}(0) \) and final angular velocity of end-effector \( {0 \omega}_{Tool}(t_f) \).

Need relationship between angular velocity and Euler parameters – not as simple as \( x(t) \) and \( \dot{x}(t) \)!
**Cartesian Space Schemes**

**Trajectory Planning for Orientation (Contd.)**

- Relationships between $^0 \omega_{\text{Tool}}(t)$ and Euler parameters

$$
^0 \omega_{\text{Tool}}(t) = 2[E(t)][^0 \varepsilon_{\text{Tool}}(t), \dot{\varepsilon}_4(t)]^T \\
(0 \varepsilon_{\text{Tool}}(t), \dot{\varepsilon}_4(t))^T = \frac{1}{2}[E(t)]^T^0 \omega_{\text{Tool}}(t)
$$

where $[E(t)]$ is given

$$
[E(t)] = \begin{pmatrix}
-\varepsilon_1 & \varepsilon_4 & -\varepsilon_3 & \varepsilon_2 \\
-\varepsilon_2 & \varepsilon_3 & \varepsilon_4 & -\varepsilon_1 \\
-\varepsilon_3 & -\varepsilon_2 & \varepsilon_1 & \varepsilon_4
\end{pmatrix}
$$

- Plan $C^2$ trajectories from given $^0 \varepsilon_{\text{Tool}}$ and $^0 \dot{\varepsilon}_{\text{Tool}}$ at $t = 0$ and $t = t_f$.
- Compute the trajectory for $\varepsilon_4(t)$ from

$$
\varepsilon_4(t) = \pm \sqrt{1 - (^0 \varepsilon_{\text{Tool}}(t) \cdot ^0 \dot{\varepsilon}_{\text{Tool}}(t))}
$$

- From $(\varepsilon(t), \varepsilon_4(t))$ obtain any required representation of the orientation of the end-effector at each instant of time.
Cartesian Space Schemes

Trajectory Planning for Orientation (Contd.)

- Relationships between $^0\omega_{Tool}(t)$ and Euler parameters

\[
^0\omega_{Tool}(t) = 2[E(t)](0\epsilon_{Tool}(t), \dot{\epsilon}_4(t))^T
\]

\[
(0\epsilon_{Tool}(t), \dot{\epsilon}_4(t))^T = \frac{1}{2}[E(t)]^T{^0\omega}_{Tool}(t)
\]

where $[E(t)]$ is given

\[
[E(t)] = \begin{pmatrix}
-\epsilon_1 & \epsilon_4 & -\epsilon_3 & \epsilon_2 \\
-\epsilon_2 & \epsilon_3 & \epsilon_4 & -\epsilon_1 \\
-\epsilon_3 & -\epsilon_2 & \epsilon_1 & \epsilon_4
\end{pmatrix}
\]

- Plan $C^2$ trajectories from given $^0\epsilon_{Tool}$ and $^0\dot{\epsilon}_{Tool}$ at $t = 0$ and $t = t_f$.
- Compute the trajectory for $\epsilon_4(t)$ from

\[
\epsilon_4(t) = \pm \sqrt{1 - (0\epsilon_{Tool}(t) \cdot 0\epsilon_{Tool}(t))}
\]

- From $(\epsilon(t), \epsilon_4(t))$ obtain any required representation of the orientation of the end-effector at each instant of time.
Relationships between $\mathbf{0} \mathbf{w}_{\text{Tool}}(t)$ and Euler parameters

$$
\mathbf{0} \mathbf{w}_{\text{Tool}}(t) = 2[E(t)](\mathbf{0} \mathbf{e}_{\text{Tool}}(t), \dot{\mathbf{e}}_4(t))^T
$$

$$(\mathbf{0} \mathbf{e}_{\text{Tool}}(t), \dot{\mathbf{e}}_4(t))^T = \frac{1}{2}[E(t)]^T \mathbf{0} \mathbf{w}_{\text{Tool}}(t)
$$

where $[E(t)]$ is given

$$
[E(t)] = \begin{pmatrix}
-\mathbf{e}_1 & \mathbf{e}_4 & -\mathbf{e}_3 & \mathbf{e}_2 \\
-\mathbf{e}_2 & \mathbf{e}_3 & \mathbf{e}_4 & -\mathbf{e}_1 \\
-\mathbf{e}_3 & -\mathbf{e}_2 & \mathbf{e}_1 & \mathbf{e}_4
\end{pmatrix}
$$

Plan $C^2$ trajectories from given $\mathbf{0} \mathbf{e}_{\text{Tool}}$ and $\mathbf{0} \mathbf{e}_{\text{Tool}}$ at $t = 0$ and $t = t_f$.

Compute the trajectory for $\mathbf{e}_4(t)$ from

$$
\mathbf{e}_4(t) = \pm \sqrt{1 - (\mathbf{0} \mathbf{e}_{\text{Tool}}(t) \cdot \mathbf{0} \mathbf{e}_{\text{Tool}}(t))}
$$

From $(\mathbf{e}(t), \mathbf{e}_4(t))$ obtain any required representation of the orientation of the end-effector at each instant of time.
**Cartesian Space Schemes**

**Trajectory Planning for Orientation (Contd.)**

- Relationships between \( ^0 \omega_{Tool}(t) \) and Euler parameters

\[
^0 \omega_{Tool}(t) = 2[E(t)][^0 \varepsilon_{Tool}(t), \dot{\varepsilon}_4(t)]^T
\]

\[
(0 \varepsilon_{Tool}(t), \dot{\varepsilon}_4(t))^T = \frac{1}{2} [E(t)]^T \omega_{Tool}(t)
\]

where \([E(t)]\) is given

\[
[E(t)] = \begin{pmatrix}
-\varepsilon_1 & \varepsilon_4 & -\varepsilon_3 & \varepsilon_2 \\
-\varepsilon_2 & \varepsilon_3 & \varepsilon_4 & -\varepsilon_1 \\
-\varepsilon_3 & -\varepsilon_2 & \varepsilon_1 & \varepsilon_4
\end{pmatrix}
\]

- Plan \( C^2 \) trajectories from given \(^0 \varepsilon_{Tool} \) and \(^0 \dot{\varepsilon}_{Tool} \) at \( t = 0 \) and \( t = t_f \).

- Compute the trajectory for \( \varepsilon_4(t) \) from

\[
\varepsilon_4(t) = \pm \sqrt{1 - (^0 \varepsilon_{Tool}(t) \cdot ^0 \dot{\varepsilon}_{Tool}(t))}
\]

- From \((\varepsilon(t), \varepsilon_4(t))\) obtain any required representation of the orientation of the end-effector at each instant of time.
**Summary of Motion Planning**

- Joint space schemes can be applied for all *actuated* joints in a robot, independently.
- In parallel manipulators with passive joints, interpolated actuated joint values *must satisfy* constraint equations containing passive and actuated joints.
- Straight line or circular trajectories may pass through singularities or points not in workspace *even though* initial and final points are in workspace or far away from singularities!
- Straight line and circular trajectories must be checked for singularities, workspace and joint limits!!
- End-effector trajectories need to take into account dynamics and torque limits at joints (Bobrow et al., 1983).
**Summary of Motion Planning**

- Joint space schemes can be applied for all *actuated* joints in a robot, independently.
- In parallel manipulators with passive joints, interpolated actuated joint values *must satisfy* constraint equations containing passive and actuated joints.
- Straight line or circular trajectories may pass through singularities or points not in workspace *even though* initial and final points are in workspace or far away from singularities!
- Straight line and circular trajectories must be checked for singularities, workspace and joint limits!!
- End-effector trajectories need to take into account dynamics and torque limits at joints (Bobrow et al., 1983).
Summary of Motion Planning

- Joint space schemes can be applied for all *actuated* joints in a robot, independently.

- In parallel manipulators with passive joints, interpolated actuated joint values *must satisfy* constraint equations containing passive and actuated joints.

- Straight line or circular trajectories may pass through singularities or points not in workspace *even though* initial and final points are in workspace or far away from singularities!

- Straight line and circular trajectories must be checked for singularities, workspace and joint limits!!

- End-effector trajectories need to take into account dynamics and torque limits at joints (Bobrow et al., 1983).
Summary of Motion Planning

- Joint space schemes can be applied for all *actuated* joints in a robot, independently.
- In parallel manipulators with passive joints, interpolated actuated joint values *must satisfy* constraint equations containing passive and actuated joints.
- Straight line or circular trajectories may pass through singularities or points not in workspace *even though* initial and final points are in workspace or far away from singularities!
- Straight line and circular trajectories must be checked for singularities, workspace and joint limits!!
- End-effector trajectories need to take into account dynamics and torque limits at joints (Bobrow et al., 1983).
**Summary of Motion Planning**

- Joint space schemes can be applied for all *actuated* joints in a robot, independently.
- In parallel manipulators with passive joints, interpolated actuated joint values *must satisfy* constraint equations containing passive and actuated joints.
- Straight line or circular trajectories may pass through singularities or points not in workspace *even though* initial and final points are in workspace or far away from singularities!
- Straight line and circular trajectories must be checked for singularities, workspace and joint limits!!
- End-effector trajectories need to take into account dynamics and torque limits at joints (Bobrow et al., 1983).
OUTLINE

1 CONTENTS

2 LECTURE 1
   • Motion planning

3 LECTURE 2
   • Control of a single link

4 LECTURE 3
   • Control of a multi-link serial manipulator

5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators

6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators

7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
Introduction
Overview

- Desired joint motion $\theta_d(t)$ available from motion planning.
- Goal of control
  - Make the joint follow desired $\theta_d(t)$ accurately.
  - In spite of external disturbances and internal parameter changes.
- To minimise error between desired and actual or measured motion feedback used.
- Feedback requires use of sensors to measure actual motion and a control scheme.
- Linear control very well known and studied – often a basis for advanced nonlinear control schemes.
Desired joint motion $\theta_d(t)$ available from motion planning.

Goal of control

- Make the joint follow desired $\theta_d(t)$ accurately.
- *In spite of external* disturbances and *internal* parameter changes.

To minimise error between *desired* and *actual or measured* motion feedback used.

Feedback requires use of sensors to measure actual motion and a control scheme.

Linear control very well known and studied – often a basis for advanced *nonlinear* control schemes.
Desired joint motion $\theta_d(t)$ available from motion planning.

Goal of control
- Make the joint follow desired $\theta_d(t)$ accurately.
- *In spite of* external disturbances and *internal* parameter changes.

To minimise error between *desired* and *actual or measured* motion, *feedback* used.

- Feedback requires use of sensors to measure actual motion and a control scheme.
- Linear control very well known and studied – often a basis for advanced *nonlinear* control schemes.
Desired joint motion $\theta_d(t)$ available from motion planning.

Goal of control

- Make the joint follow desired $\theta_d(t)$ accurately.
- In spite of external disturbances and internal parameter changes.

To minimise error between desired and actual or measured motion feedback used.

Feedback requires use of sensors to measure actual motion and a control scheme.

Linear control very well known and studied – often a basis for advanced nonlinear control schemes.
Desired joint motion $\theta_d(t)$ available from motion planning.

Goal of control
- Make the joint follow desired $\theta_d(t)$ accurately.
- *In spite of* external disturbances and *internal* parameter changes.

To minimise error between desired and actual or measured motion feedback used.

Feedback requires use of sensors to measure actual motion and a control scheme.

Linear control very well known and studied – often a basis for advanced nonlinear control schemes.
**CONTROL OF A SINGLE LINK**

**MODEL**

![Diagram of a single link model with gears, bearings, and a motor.](image)

**Figure 5:** Model of a single link

- Single link driven by a DC motor through a gear shown in Figure 5.
  - Rated speed of typical DC motor $\rightarrow$ 2000 rpm or more.
  - Required speed about 60 rpm $\rightarrow$ need large speed reduction!
  - Analysis assume two spur gears giving the required speed reduction $\rightarrow$ gear ratio $n << 1$. 
CONTROL OF A SINGLE LINK

Model (Contd.)

- Link rotation $\theta_l$ related to motor rotation $\theta_m$ by $\frac{\theta_l}{\theta_m} = n$
- One-degree-of-freedom system
  \[
  \theta_l = n\theta_m, \quad \dot{\theta}_l = n\dot{\theta}_m, \quad \ddot{\theta}_l = n\ddot{\theta}_m
  \]
- Equation of motion of Gear 1
  \[
  J_m \ddot{\theta}_m + f_m \dot{\theta}_m + T_1 = T_m
  \]
  $J_m$, $f_m$ and $T_m$ are the inertia of the motor, the viscous friction at the motor shaft, and the torque output of the motor, respectively. $T_1$ denotes the torque acting on gear 1 from gear 2 and the link.
- Equation of motion of link + Gear 2
  \[
  J_l \ddot{\theta}_l + f_l \dot{\theta}_l = T_2 + T_l
  \]
  where $J_l$, $f_l$ and $T_l$ are the inertia of the load (link and gear), the viscous friction at the load, and any external disturbance torque acting on the link, respectively. $T_2$ denotes the torque transmitted to gear 2 by gear 1.
CONTROL OF A SINGLE LINK

Model (Contd.)

- Link rotation $\theta_l$ related to motor rotation $\theta_m$ by $\frac{\theta_l}{\theta_m} = n$
- One-degree-of-freedom system

$$\theta_l = n\theta_m, \quad \dot{\theta}_l = n\dot{\theta}_m, \quad \ddot{\theta}_l = n\ddot{\theta}_m$$

- Equation of motion of Gear 1

$$J_m\ddot{\theta}_m + f_m\dot{\theta}_m + T_1 = T_m$$

$J_m$, $f_m$ and $T_m$ are the inertia of the motor, the viscous friction at the motor shaft, and the torque output of the motor, respectively. $T_1$ denotes the torque acting on gear 1 from gear 2 and the link.

- Equation of motion of link + Gear 2

$$J_l\ddot{\theta}_l + f_l\dot{\theta}_l = T_2 + T_l$$

where $J_l$, $f_l$ and $T_l$ are the inertia of the load (link and gear), the viscous friction at the load, and any external disturbance torque acting on the link, respectively. $T_2$ denotes the torque transmitted to gear 2 by gear 1.
CONTROL OF A SINGLE LINK

Model (Contd.)

- Link rotation $\theta_l$ related to motor rotation $\theta_m$ by $\frac{\theta_l}{\theta_m} = n$
- One-degree-of-freedom system

\[
\begin{align*}
\theta_l &= n\theta_m, \\
\dot{\theta}_l &= n\dot{\theta}_m, \\
\ddot{\theta}_l &= n\ddot{\theta}_m
\end{align*}
\]

- Equation of motion of Gear 1

\[
J_m\ddot{\theta}_m + f_m\dot{\theta}_m + T_1 = T_m
\]

$J_m$, $f_m$ and $T_m$ are the inertia of the motor, the viscous friction at the motor shaft, and the torque output of the motor, respectively. $T_1$ denotes the torque acting on gear 1 from gear 2 and the link.

- Equation of motion of link + Gear 2

\[
J_l\ddot{\theta}_l + f_l\dot{\theta}_l = T_2 + T_l
\]

where $J_l$, $f_l$ and $T_l$ are the inertia of the load (link and gear), the viscous friction at the load, and any external disturbance torque acting on the link, respectively. $T_2$ denotes the torque transmitted to gear 2 by gear 1.
CONTROL OF A SINGLE LINK

Model (Contd.)

- Link rotation $\theta_l$ related to motor rotation $\theta_m$ by $\frac{\theta_l}{\theta_m} = n$
- One-degree-of-freedom system

\[
\theta_l = n\theta_m, \quad \dot{\theta}_l = n\dot{\theta}_m, \quad \ddot{\theta}_l = n\ddot{\theta}_m
\]

- Equation of motion of Gear 1

\[
J_m\ddot{\theta}_m + f_m\dot{\theta}_m + T_1 = T_m
\]

where $J_m$, $f_m$ and $T_m$ are the inertia of the motor, the viscous friction at the motor shaft, and the torque output of the motor, respectively. $T_1$ denotes the torque acting on gear 1 from gear 2 and the link.

- Equation of motion of link + Gear 2

\[
J_l\ddot{\theta}_l + f_l\dot{\theta}_l = T_2 + T_l
\]

where $J_l$, $f_l$ and $T_l$ are the inertia of the load (link and gear), the viscous friction at the load, and any external disturbance torque acting on the link, respectively. $T_2$ denotes the torque transmitted to gear 2 by gear 1.
**CONTROL OF A SINGLE LINK**

**Model (Contd.)**

- **Assuming** no energy loss at gear tooth contacts: \( T_1 \theta_m = T_2 \theta_I \)

Equations of motion for system

\[
(J_m + n^2 J_I) \ddot{\theta}_m + (f_m + n^2 f_I) \dot{\theta}_m = T_m + n T_I
\]  

(7)

- \( n \) is small (around 0.01), the effect of the load inertia and load friction, *as seen from the motor*, is reduced by a factor of \( n^2 \).
- Effect of \( T_I \) is also reduced by a factor of \( n \).
- Multi-link robots with gear reduction at joints \( \rightarrow \) effect of the coupling torques from motion of other links (see [Module 6, Lecture 2]) (contributing to \( T_I \)) is reduced.
- One of the reason why linear control schemes work in industrial robots!!.
CONTROL OF A SINGLE LINK
Model (Contd.)

- Assuming no energy loss at gear tooth contacts \( T_1 \theta_m = T_2 \theta_l \)
- Equations of motion for system

\[
(J_m + n^2 J_l) \ddot{\theta}_m + (f_m + n^2 f_l) \dot{\theta}_m = T_m + nT_l
\]  \( (7) \)

- \( n \) is small (around 0.01), the effect of the load inertia and load friction, as seen from the motor, is reduced by a factor of \( n^2 \).
- Effect of \( T_l \) is also reduced by a factor of \( n \).
- Multi-link robots with gear reduction at joints \( \rightarrow \) effect of the coupling torques from motion of other links (see Module 6, Lecture 2) (contributing to \( T_l \)) is reduced.
- One of the reason why linear control schemes work in industrial robots!!.
CONTROL OF A SINGLE LINK
Model (Contd.)

Figure 6: Model of a permanent magnet DC servo-motor

- Model of a permanent magnet DC motor shown in Figure 6.
- Stationary armature of resistance and inductance $R_a$ and $L_a$ respectively.
- Rotor is a permanent magnet (rare earth material).
- Voltage applied $V_a$ and current in coil $i_a$. 
CONTROL OF A SINGLE LINK
MODEL (CONTD.)

- Torque generated by motor
  \[ T_m = K_t i_a \]

- Back emf generated by coil rotating at \( \dot{\theta}_m \)
  \[ V = K_g \dot{\theta}_m \]

\( K_t \) and \( K_g \rightarrow \) torque and back emf constant (available in motor specifications).

- Dynamics of a motor
  \[ L_a \ddot{i}_a + R_a i_a + K_g \dot{\theta}_m = V_a \]

- For small DC servo motors, \( L_a \) is small and can be ignored.
CONTROL OF A SINGLE LINK
Model (Contd.)

- Torque generated by motor
  \[ T_m = K_t i_a \]

- *Back emf* generated by coil rotating at \( \dot{\theta}_m \)
  \[ V = K_g \dot{\theta}_m \]

*\( K_t \) and \( K_g \to \) torque and back emf constant (available in motor specifications).

- Dynamics of a motor
  \[ L_a \dot{i}_a + R_a i_a + K_g \dot{\theta}_m = V_a \]

- For small DC servo motors, \( L_a \) is small and can be ignored.
CONTROL OF A SINGLE LINK
Model (Contd.)

- Torque generated by motor

\[ T_m = K_t i_a \]

- Back emf generated by coil rotating at \( \dot{\theta}_m \)

\[ V = K_g \dot{\theta}_m \]

\( K_t \) and \( K_g \) → torque and back emf constant (available in motor specifications).

- Dynamics of a motor

\[ L_a \dot{i}_a + R_a i_a + K_g \dot{\theta}_m = V_a \]

- For small DC servo motors, \( L_a \) is small and can be ignored.
CONTROL OF A SINGLE LINK
MODEL (CONTD.)

- Torque generated by motor
  \[ T_m = K_t i_a \]

- Back emf generated by coil rotating at \( \dot{\theta}_m \)
  \[ V = K_g \dot{\theta}_m \]

\( K_t \) and \( K_g \) \( \rightarrow \) torque and back emf constant (available in motor specifications).

- Dynamics of a motor
  \[ L_a \dot{i}_a + R_a i_a + K_g \dot{\theta}_m = V_a \]

- For small DC servo motors, \( L_a \) is small and can be ignored.
CONTROL OF A SINGLE LINK

Model (Contd.)

- Combining equations of motion and the dynamics of motor with $L_a = 0$

\[
(J_m + n^2 J_l) \ddot{\theta}_m + (f_m + n^2 f_l) \dot{\theta}_m = K_t \left( \frac{V_a - K_g \dot{\theta}_m}{R_a} \right) + n T_l
\]

- In a compact form

\[
J \dot{\Omega} + F \Omega = KV_a + T_d \tag{8}
\]

\[
K = \frac{K_t}{R_a}, \quad F = (f_m + n^2 f_l) + K_t K_g / R_a
\]

\[
J = J_m + n^2 J_l, \quad T_d = n T_l, \quad \Omega = \dot{\theta}_m
\]

- Equation (8) describes the *mechatronic* behavior of the single-link manipulator.
  - Dynamics *in terms of angular velocity* → linear first-order ODE.
  - Back emf → increases the damping of the system.
  - Link will rotate if a) voltage is applied or b) an external disturbance torque acts on the link.
CONTROL OF A SINGLE LINK

Model (Contd.)

- Combining equations of motion and the dynamics of motor with $L_a = 0$

\[(J_m + n^2 J_l) \ddot{\theta}_m + (f_m + n^2 f_l) \dot{\theta}_m = K_t \left( \frac{V_a - K_g \dot{\theta}_m}{R_a} \right) + nT_l\]

- In a compact form

\[J \dot{\Omega} + F \Omega = KV_a + T_d \] (8)

\[K = K_t / R_a, \quad F = (f_m + n^2 f_l) + K_t K_g / R_a\]

\[J = J_m + n^2 J_l, \quad T_d = nT_l, \quad \Omega = \dot{\theta}_m\]

- Equation (8) describes the mechatronic behavior of the single-link manipulator.
  - Dynamics \textit{in terms of angular velocity} $\rightarrow$ linear first-order ODE.
  - Back emf $\rightarrow$ increases the damping of the system.
  - Link will rotate if a) voltage is applied or b) an external disturbance torque acts on the link.
CONTROL OF A SINGLE LINK
MODEL (Contd.)

- Combining equations of motion and the dynamics of motor with $L_a = 0$

$$(J_m + n^2 J_l)\ddot{\theta}_m + (f_m + n^2 f_l)\dot{\theta}_m = K_t \left(\frac{V_a - K_g \dot{\theta}_m}{R_a}\right) + nT_l$$

- In a compact form

$$J\dot{\Omega} + F\Omega = KV_a + T_d \quad (8)$$

$$K = \frac{K_t}{R_a}, \quad F = (f_m + n^2 f_l) + \frac{K_t K_g}{R_a}$$

$$J = J_m + n^2 J_l, \quad T_d = nT_l, \quad \Omega = \dot{\theta}_m$$

- Equation (8) describes the mechatronic behavior of the single-link manipulator.
  - Dynamics in terms of angular velocity → linear first-order ODE.
  - Back emf → increases the damping of the system.
  - Link will rotate if a) voltage is applied or b) an external disturbance torque acts on the link.
CONTROL OF A SINGLE LINK
ANALYSIS – s–domain Approach

- Laplace Transforms (See any undergraduate mathematics textbook)
  - Definition – $F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t) dt$
  - Laplace of derivative: $\mathcal{L}\{\frac{d}{dt} f(t)\} = sF(s) - F(0)$
  - For zero initial conditions, converts ODE to polynomial in $s$ – ODE in equation (8) in Laplace domain is

$$Js\Omega(s) + F\Omega(s) = KV_a(s) + T_d(s)$$

- Transfer Function → Ratio of output to input in Laplace domain
- Two inputs $V_a(s)$ and $T_d(s) →$ two transfer functions

$$\frac{\Omega(s)}{V_a(s)} = \frac{K}{Js + F}, \quad \frac{\Omega(s)}{T_d(s)} = \frac{1}{Js + F}$$

with $\Omega(s)$ as output.
CONTROL OF A SINGLE LINK
ANALYSIS – s–domain Approach

- Laplace Transforms (See any undergraduate mathematics textbook)
  - Definition – \( F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st} f(t) dt \)
  - Laplace of derivative: \( \mathcal{L}\left\{ \frac{df}{dt} \right\} = sF(s) - F(0) \)
  - For zero initial conditions, converts ODE to polynomial in \( s \) – ODE in equation (8) in Laplace domain is
    \[ Js\Omega(s) + F\Omega(s) = KV_a(s) + T_d(s) \]

- Transfer Function → Ratio of output to input in Laplace domain
  - Two inputs \( V_a(s) \) and \( T_d(s) \) → two transfer functions
    \[ \frac{\Omega(s)}{V_a(s)} = \frac{K}{Js + F}, \quad \frac{\Omega(s)}{T_d(s)} = \frac{1}{Js + F} \]

    with \( \Omega(s) \) as output.
CONTROL OF A SINGLE LINK

ANALYSIS – s–domain Approach

- Laplace Transforms (See any undergraduate mathematics textbook)
  - Definition – $F(s) = \mathcal{L}\{f(t)\} = \int_0^\infty e^{-st}f(t)\,dt$
  - Laplace of derivative: $\mathcal{L}\left\{\frac{d}{dt}f(t)\right\} = sF(s) - F(0)$
  - For zero initial conditions, converts ODE to polynomial in $s$ – ODE in equation (8) in Laplace domain is
    $$Js\Omega(s) + F\Omega(s) = KV_a(s) + T_d(s)$$

- Transfer Function $\rightarrow$ Ratio of output to input in Laplace domain
- Two inputs $V_a(s)$ and $T_d(s) \rightarrow$ two transfer functions
  $$\frac{\Omega(s)}{V_a(s)} = \frac{K}{Js + F}, \quad \frac{\Omega(s)}{T_d(s)} = \frac{1}{Js + F}$$

with $\Omega(s)$ as output.
CONTROL OF A SINGLE LINK

ANALYSIS – s–DOMAIN APPROACH (Contd.)

\[ \frac{K}{Js + F} \]

\[ \frac{1}{Js + F} \]

\[ \frac{K}{Js + F} \]

Figure 7: Transfer functions of a single link manipulator
CONTROL OF A SINGLE LINK
ANALYSIS – s–domain APPROACH(Contd.)

- Figures (7) (a) & (b) are called Open-loop Transfer Functions.
- Figure (7) (c) is called Closed-loop Transfer Functions – motor output is measured and fed back as another input to controller.
- Feedback $\rightarrow$ robustness to internal parameter change and external disturbances.
  - Assume $V_a(s) = K_p(\Omega_d(s) - \Omega(s))$ – simplest possible controller, $D(s) = K_p$ a constant!
  - Controller gain, $K_p$, can be chosen but once chosen is fixed (factory setting!)
  - For open-loop (without feedback), $V_a(s) = K_p\Omega_d(s)$.
  - Open-loop – Choose $K_p = 1/K_0$ where $K_0 = K/F$. 
Control of a Single Link
Analysis – s-domain Approach (Contd.)

- Figures (7) (a) & (b) are called Open-loop Transfer Functions.
- Figure (7) (c) is called Closed-loop Transfer Functions – motor output is measured and fed back as another input to controller.
- Feedback → robustness to internal parameter change and external disturbances.
  - Assume \( V_a(s) = K_p(\Omega_d(s) - \Omega(s)) \) – simplest possible controller, \( D(s) = K_p \) a constant!
  - Controller gain, \( K_p \), can be chosen but once chosen is fixed (factory setting!)
  - For open-loop (without feedback), \( V_a(s) = K_p\Omega_d(s) \).
  - Open-loop – Choose \( K_p = 1/K_0 \) where \( K_0 = K/F \).
Figures (7) (a) & (b) are called *Open-loop Transfer Functions*.

Figure (7) (c) is called *Closed-loop Transfer Functions* – motor output is measured and *fed back* as another input to *controller*.

*Feedback → robustness* to internal parameter change and external disturbances.

- Assume $V_a(s) = K_p(\Omega_d(s) - \Omega(s))$ – simplest possible controller, $D(s) = K_p$ a constant!
- Controller gain, $K_p$, can be chosen but *once chosen is fixed* (factory setting!)
- For *open-loop* (without feedback), $V_a(s) = K_p \Omega_d(s)$.
- Open-loop – Choose $K_p = 1/K_0$ where $K_0 = K/F$. 
**Control of a Single Link**

**Analysis – s–domain Approach (Contd.)**

- With $T_d = 0$ and steady state, i.e., $s \to 0$,

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{K}{Js + F} V_a(s) \Rightarrow \Omega = (K/F)V_a = K_0K_p\Omega_d$$

- For $K_p = 1/K_0$, $\Omega = \Omega_d$ as desired in any controller!

- For closed-loop $V_a(s) = K_p(\Omega_d(s) - \Omega(s))$ and for $s \to 0$

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{KK_p}{Js + F + KK_p} V_a(s) \Rightarrow \Omega = \frac{K_0K_p}{1 + K_0K_p}\Omega_d$$

- Best possible choice $K_0K_p >> 1$ and best possible outcome $\Omega \approx \Omega_d$.

- Apparently with feedback, the situation is worse!
CONTROL OF A SINGLE LINK
ANALYSIS – s–domain APPROACH (Contd.)

- With $T_d = 0$ and steady state, i.e., $s \to 0$,

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{K}{Js + F} V_a(s) \Rightarrow \Omega = (K/F) V_a = K_0 K_p \Omega_d$$

- For $K_p = 1/K_0$, $\Omega = \Omega_d$ as desired in any controller!
- For closed-loop $V_a(s) = K_p (\Omega_d(s) - \Omega(s))$ and for $s \to 0$

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{K K_p}{Js + F + KK_p} V_a(s) \Rightarrow \Omega = \frac{K_0 K_p}{1 + K_0 K_p} \Omega_d$$

- Best possible choice $K_0 K_p >> 1$ and best possible outcome $\Omega \approx \Omega_d$.
- Apparently with feedback, the situation is worse!
With $T_d = 0$ and steady state, i.e., $s \to 0$,

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{K}{Js + F} V_a(s) \Rightarrow \Omega = \left(\frac{K}{F}\right) V_a = K_0 K_p \Omega_d$$

For $K_p = 1/K_0$, $\Omega = \Omega_d$ as desired in any controller!

For closed-loop $V_a(s) = K_p (\Omega_d(s) - \Omega(s))$ and for $s \to 0$

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{KK_p}{Js + F + KK_p} V_a(s) \Rightarrow \Omega = \frac{K_0 K_p}{1 + K_0 K_p} \Omega_d$$

Best possible choice $K_0 K_p \gg 1$ and best possible outcome $\Omega \approx \Omega_d$.

Apparently with feedback, the situation is worse!
CONTROL OF A SINGLE LINK

Analysis – s–domain Approach (Contd.)

- With $T_d = 0$ and steady state, i.e., $s \to 0$,

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{K}{Js + F} V_a(s) \Rightarrow \Omega = \left(K/F\right)V_a = K_0K_p\Omega_d$$

- For $K_p = 1/K_0$, $\Omega = \Omega_d$ as desired in any controller!

- For closed-loop $V_a(s) = K_p(\Omega_d(s) - \Omega(s))$ and for $s \to 0$

$$\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{KK_p}{Js + F + KK_p} V_a(s) \Rightarrow \Omega = \frac{K_0K_p}{1 + K_0K_p}\Omega_d$$

- Best possible choice $K_0K_p >> 1$ and best possible outcome $\Omega \approx \Omega_d$.

- Apparently with feedback, the situation is worse!
**Control of a Single Link**

**Analysis – s-domain Approach (Contd.)**

- With \( T_d = 0 \) and steady state, i.e., \( s \to 0 \),

\[
\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{K}{Js + F} V_a(s) \Rightarrow \Omega = (K/F)V_a = K_0 K_p \Omega_d
\]

- For \( K_p = 1/K_0 \), \( \Omega = \Omega_d \) as desired in any controller!
- For closed-loop \( V_a(s) = K_p(\Omega_d(s) - \Omega(s)) \) and for \( s \to 0 \)

\[
\lim_{s \to 0} \Omega(s) = \lim_{s \to 0} \frac{KK_p}{Js + F + KK_p} V_a(s) \Rightarrow \Omega = \frac{K_0 K_p}{1 + K_0 K_p} \Omega_d
\]

- Best possible choice \( K_0 K_p >> 1 \) and best possible outcome \( \Omega \approx \Omega_d \).
- Apparently with feedback, the situation is worse!
Consider *change in internal* parameter due to environmental changes – $K_0$ changes to $K_0 + \delta K_0$

- For open-loop in steady-state
  \[ \Omega + \delta \Omega = (K_0 + \delta K_0)K_p \Omega_d \]
  Since $K_p$ is set to $1/K_0$, \[ \delta \Omega = (\delta K_0/K_0) \Omega_d \]

- For closed-loop with $K_0 K_p \gg 1$,
  \[
  \frac{\delta \Omega' / \Omega'}{\Omega'} = \frac{1}{1 + K_0 K_p} (\delta K_0 / K_0)
  \]
  where $\Omega' = \frac{K_0 K_p}{1 + K_0 K_p} \Omega_d \approx \Omega_d$

- An $x\%$ change in $K_0 \rightarrow \frac{1}{1 + K_0 K_p} \times x\%$ change in $\Omega'$.

- Since $1 + K_0 K_p \gg 1$, the change in output *greatly reduced* by feedback $\rightarrow$ *Robustness!*
Consider change in internal parameter due to environmental changes –
$K_0$ changes to $K_0 + \delta K_0$

For open-loop in steady-state
- $\Omega + \delta \Omega = (K_0 + \delta K_0)K_p\Omega_d$
- Since $K_p$ is set to $1/K_0$, $\delta \Omega = (\delta K_0/K_0)\Omega_d$

For closed-loop with $K_0K_p >> 1$,

$$\frac{\delta \Omega'}{\Omega'} = \frac{1}{1 + K_0K_p}(\delta K_0/K_0)$$

where $\Omega' = \frac{K_0K_p}{1+K_0K_p}\Omega_d \approx \Omega_d$

- An $x\%$ change in $K_0 \rightarrow \frac{1}{1+K_0K_p} \times x\%$ change in $\Omega'$.
- Since $1 + K_0K_p >> 1$, the change in output greatly reduced by feedback $\rightarrow$ Robustness!
CONTROL OF A SINGLE LINK

Analysis – s–domain Approach (Contd.)

- Consider *change* in *internal* parameter due to environmental changes – $K_0$ changes to $K_0 + \delta K_0$

- For open-loop in steady-state
  - $\Omega + \delta \Omega = (K_0 + \delta K_0) K_p \Omega_d$
  - Since $K_p$ is set to $1/K_0$, $\delta \Omega = (\delta K_0/K_0) \Omega_d$

- For closed-loop with $K_0 K_p >> 1$,

\[
\frac{\delta \Omega' / \Omega'}{\Omega'} = \frac{1}{1 + K_0 K_p} (\delta K_0/K_0)
\]

where $\Omega' = \frac{K_0 K_p}{1 + K_0 K_p} \Omega_d \approx \Omega_d$

- An $x\%$ change in $K_0 \rightarrow \frac{1}{1 + K_0 K_p} \times x\%$ change in $\Omega'$.

- Since $1 + K_0 K_p >> 1$, the change in output *greatly reduced* by feedback $\rightarrow$ Robustness!
**CONTROL OF A SINGLE LINK**

**Analysis – s-domain Approach (Contd.)**

- Consider *change* in *internal* parameter due to environmental changes – $K_0$ changes to $K_0 + \delta K_0$

- For open-loop in steady-state
  - $\Omega + \delta \Omega = (K_0 + \delta K_0) K_p \Omega_d$
  - Since $K_p$ is set to $1/K_0$, $\delta \Omega = (\delta K_0/K_0) \Omega_d$

- For closed-loop with $K_0 K_p \gg 1$,

$$\frac{\delta \Omega'}{\Omega'} = \frac{1}{1 + K_0 K_p} (\delta K_0/K_0)$$

where $\Omega' = \frac{K_0 K_p}{1+K_0 K_p} \Omega_d \approx \Omega_d$

- An $x\%$ change in $K_0 \rightarrow \frac{1}{1+K_0 K_p} \times x\%$ change in $\Omega'$.

- Since $1 + K_0 K_p \gg 1$, the change in output *greatly reduced* by feedback $\rightarrow$ Robustness!
Consider change in internal parameter due to environmental changes – $K_0$ changes to $K_0 + \delta K_0$

For open-loop in steady-state
- $\Omega + \delta \Omega = (K_0 + \delta K_0)K_p\Omega_d$
- Since $K_p$ is set to $1/K_0$, $\delta \Omega = (\delta K_0/K_0)\Omega_d$

For closed-loop with $K_0K_p \gg 1$,

$$\frac{\delta \Omega' / \Omega'}{\Omega'} = \frac{1}{1 + K_0K_p}(\delta K_0/K_0)$$

where $\Omega' = \frac{K_0K_p}{1 + K_0K_p}\Omega_d \approx \Omega_d$

An $x\%$ change in $K_0 \rightarrow \frac{1}{1 + K_0K_p} \times x\%$ change in $\Omega'$.

Since $1 + K_0K_p \gg 1$, the change in output greatly reduced by feedback → Robustness!
CONTROL OF A SINGLE LINK

Analysis – s-domain Approach (Contd.)

- If $T_d \neq 0$

\[ \Omega = K_0 K_c \Omega_d + K_0 \left( \frac{T_d}{K} \right), \quad \text{Controller gain is } K_c \]

- For $K_0 K_c = 1$, \[ \Omega = \Omega_d + K_0 \left( \frac{T_d}{K} \right) \rightarrow \text{Change in output proportional to } T_d \]

- With feedback, steady-state output

\[ \Omega = \frac{K_0 K_c}{1 + K_0 K_c} \Omega_d + \frac{K_0}{1 + K_0 K_c} \left( \frac{T_d}{K} \right) \]

- Choose $K_0 K_c \gg 1$ and $K_0 K_c \gg (K_0/K)$ (or $K_c \gg 1/K$).

- Effect of $T_d$ is reduced due to feedback!!

ASHITAVA GHOSAL (IISc)  ROBOTICS: ADVANCED CONCEPTS & ANALYSIS  NPTEL, 2010  36 / 129
CONTROL OF A SINGLE LINK
ANALYSIS – s–domain APPROACH (Contd.)

- If \( T_d \neq 0 \)
  \[
  \Omega = K_0 K_c \Omega_d + K_0 (T_d / K), \quad \text{Controller gain is } K_c
  \]
- For \( K_0 K_c = 1 \), \( \Omega = \Omega_d + K_0 (T_d / K) \rightarrow \text{Change in output proportional to } T_d \)
- With feedback, steady-state output
  \[
  \Omega = \frac{K_0 K_c}{1 + K_0 K_c} \Omega_d + \frac{K_0}{1 + K_0 K_c} (T_d / K)
  \]
- Choose \( K_0 K_c >> 1 \) and \( K_0 K_c >> (K_0 / K) \) (or \( K_c >> 1 / K \)).
- Effect of \( T_d \) is \textit{reduced} due to feedback!!
If $T_d \neq 0$

$$\Omega = K_0 K_c \Omega_d + K_0 \left( \frac{T_d}{K} \right),$$
Controller gain is $K_c$

For $K_0 K_c = 1$, $\Omega = \Omega_d + K_0 \left( \frac{T_d}{K} \right) \rightarrow$ Change in output proportional to $T_d$

With feedback, steady-state output

$$\Omega = \frac{K_0 K_c}{1 + K_0 K_c} \Omega_d + \frac{K_0}{1 + K_0 K_c} \left( \frac{T_d}{K} \right)$$

Choose $K_0 K_c >> 1$ and $K_0 K_c >> \left( \frac{K_0}{K} \right)$ (or $K_c >> 1/K$).

Effect of $T_d$ is reduced due to feedback!!
CONTROL OF A SINGLE LINK
ANALYSIS – s–domain Approach (Contd.)

- If $T_d \neq 0$
  \[ \Omega = K_0 K_c \Omega_d + K_0 \frac{T_d}{K}, \quad \text{Controller gain is } K_c \]

- For $K_0 K_c = 1$, $\Omega = \Omega_d + K_0 \frac{T_d}{K} \rightarrow$ Change in output proportional to $T_d$

- With feedback, steady-state output
  \[ \Omega = \frac{K_0 K_c}{1 + K_0 K_c} \Omega_d + \frac{K_0}{1 + K_0 K_c} \left( T_d / K \right) \]

- Choose $K_0 K_c \gg 1$ and $K_0 K_c \gg \left( K_0 / K \right)$ (or $K_c \gg 1/K$).

- Effect of $T_d$ is reduced due to feedback!!
CONTROL OF A SINGLE LINK

Analysis – s-domain Approach (Contd.)

- If $T_d \neq 0$
  
  $\Omega = K_0 K_c \Omega_d + K_0 (T_d/K)$,  \hspace{1cm} \text{Controller gain is } K_c

- For $K_0 K_c = 1$, $\Omega = \Omega_d + K_0 (T_d/K) \rightarrow \text{Change in output proportional to } T_d$

- With feedback, steady-state output

  \[
  \Omega = \frac{K_0 K_c}{1 + K_0 K_c} \Omega_d + \frac{K_0}{1 + K_0 K_c} (T_d/K)
  \]

- Choose $K_0 K_c >> 1$ and $K_0 K_c >> (K_0/K)$ (or $K_c >> 1/K$).  

- Effect of $T_d$ is reduced due to feedback!!
CONTROL OF A SINGLE LINK

ANALYSIS – FIRST-ORDER SYSTEM

First-order system as governing ODE is first-order.

- Several ways to analyse control systems \( \rightarrow s \)-plane analysis
- Closed-loop transfer function between output \( \Omega(s) \) and desired speed \( \Omega_d(s) \)

\[
\frac{\Omega(s)}{\Omega_d(s)} = \left(\frac{KK_p}{J}\right)\left(\frac{1}{s + (F + KK_p)/J}\right)
\]

- Step response \( \Omega(s) \) for \( \Omega_d(s) = 1/s \).

- \( \Omega(t) \) is of the form \( 1 - e^{-\left(\frac{F+KK_p}{J}\right)t} \) \( \rightarrow F, K, K_p \) and \( J \) are all positive
  \( \rightarrow \Omega(t) \) always bounded and approaches \( \Omega_d(t) \) as \( t \rightarrow \infty \).

- System stable as bounded output for a bounded input.

- Increasing \( K_p \) makes \( \Omega(t) \) approach 1 faster!!

Figure 8: Block diagram of single link manipulator under feedback and \( T_d = 0 \)
CONTROL OF A SINGLE LINK

ANALYSIS – SECOND-ORDER SYSTEM

- For control of angular rotation, open-loop transfer function with $T_d = 0$ is

$$\frac{\theta(s)}{V_a(s)} = \frac{K}{s(Js + F)}$$

- Transfer function is second-order as the governing ODE is second-order (denominator polynomial is second degree in $s$).

- Closed-loop transfer function between output $\theta(s)$ and desired input $\theta_d(s)$

$$\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p}{s(Js + F) + KK_p} = \frac{\omega_n^2}{s^2 + 2\xi \omega_n s + \omega_n^2}$$

where $\omega_n^2 = (KK_p/J)$, $F/J = 2\xi \omega_n$ and $\xi = \frac{F}{2\sqrt{JKK_p}}$.

- For second-order systems, $\omega_n$ is called the natural frequency of the system and $\xi$ is called the damping.

- The parameters $\omega_n$ and $\xi$ completely determine the behaviour of a second-order system.
CONTROL OF A SINGLE LINK

ANALYSIS – second-order system

- For control of angular rotation, open-loop transfer function with $T_d = 0$ is

$$\frac{\theta(s)}{V_a(s)} = \frac{K}{s(Js + F)}$$

- Transfer function is second-order as the governing ODE is second-order (denominator polynomial is second degree in $s$).

- Closed-loop transfer function between output $\theta(s)$ and desired input $\theta_d(s)$

$$\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p}{s(Js + F) + KK_p} = \frac{\omega_n^2}{s^2 + 2\xi \omega_n s + \omega_n^2}$$

where $\omega_n^2 = (KK_p / J)$, $F / J = 2\xi \omega_n$ and $\xi = \frac{F}{2\sqrt{JKK_p}}$.

- For second-order systems, $\omega_n$ is called the natural frequency of the system and $\xi$ is called the damping.

- The parameters $\omega_n$ and $\xi$ completely determine the behaviour of a second-order system.
CONTROL OF A SINGLE LINK

ANALYSIS – second-order system

- For control of angular rotation, open-loop transfer function with $T_d = 0$ is

$$\frac{\theta(s)}{V_a(s)} = \frac{K}{s(Js + F)}$$

- Transfer function is second-order as the governing ODE is second-order (denominator polynomial is second degree in $s$).

- Closed-loop transfer function between output $\theta(s)$ and desired input $\theta_d(s)$

$$\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p}{s(Js + F) + KK_p} = \frac{\omega_n^2}{s^2 + 2\xi \omega_n s + \omega_n^2}$$

where $\omega_n^2 = (KK_p/J)$, $F/J = 2\xi \omega_n$ and $\xi = \frac{F}{2\sqrt{JKK_p}}$.

- For second-order systems, $\omega_n$ is called the natural frequency of the system and $\xi$ is called the damping.

- The parameters $\omega_n$ and $\xi$ completely determine the behaviour of a second-order system.
CONTROL OF A SINGLE LINK

ANALYSIS – second-order system

- For control of angular rotation, open-loop transfer function with $T_d = 0$ is
  \[\frac{\theta(s)}{V_a(s)} = \frac{K}{s(Js + F)}\]

- Transfer function is second-order as the governing ODE is second-order (denominator polynomial is second degree in $s$).

- Closed-loop transfer function between output $\theta(s)$ and desired input $\theta_d(s)$
  \[\frac{\theta(s)}{\theta_d(s)} = \frac{\frac{KK_p}{s(Js + F) + KK_p}} = \frac{\omega_n^2}{s^2 + 2\xi \omega_n s + \omega_n^2}\]
  where $\omega_n^2 = \frac{KK_p}{J}$, $F/J = 2\xi \omega_n$ and $\xi = \frac{F}{2\sqrt{JKK_p}}$.

- For second-order systems, $\omega_n$ is called the natural frequency of the system and $\xi$ is called the damping.

- The parameters $\omega_n$ and $\xi$ completely determine the behaviour of a second-order system.
CONTROL OF A SINGLE LINK

ANALYSIS – SECOND-ORDER SYSTEM

- For control of angular rotation, open-loop transfer function with $T_d = 0$ is

$$\frac{\theta(s)}{V_a(s)} = \frac{K}{s(Js + F)}$$

- Transfer function is second-order as the governing ODE is second-order (denominator polynomial is second degree in $s$).

- Closed-loop transfer function between output $\theta(s)$ and desired input $\theta_d(s)$

$$\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p}{s(Js + F) + KK_p} = \frac{\omega_n^2}{s^2 + 2\xi \omega_n s + \omega_n^2}$$

where $\omega_n^2 = (KK_p/J)$, $F/J = 2\xi \omega_n$ and $\xi = \frac{F}{2\sqrt{JKK_p}}$.

- For second-order systems, $\omega_n$ is called the natural frequency of the system and $\xi$ is called the damping.

- The parameters $\omega_n$ and $\xi$ completely determine the behaviour of a second-order system.
Three possible kinds of behaviour

0 < \( \xi \) < 1 – under-damped systems.

- Output oscillates about the desired input before settling down in infinite time.
- Settling time \( t_s \) – Time taken for output to reach within ±5% (or ±2%) of the input → For ±5% \( t_s \approx \frac{3}{\xi \omega_n} \) and is \( \approx \frac{4}{\xi \omega_n} \) for ±2%.
- The maximum overshoot is large for low damping \( \xi \), and small for high \( \xi \) → Peak overshoot is \( e^{-\left(\frac{\xi}{\sqrt{1-\xi^2}}\right)\pi} \).
- The roots of the denominator closed-loop polynomial are complex with negative real parts.
- Roots are in the left-half of the s plane → second-order system is stable.

Output \( \Omega(t) \) to a step input shown in Figure 9(b).
Three possible kinds of behaviour

$0 < \xi < 1$ – under-damped systems.

- Output oscillates about the desired input before settling down in infinite time.
- Settling time $t_s$ – Time taken for output to reach within $\pm 5\%$ (or $\pm 2\%$) of the input $\rightarrow$ For $\pm 5\%$ $t_s \approx \frac{3}{\xi \omega_n}$ and is $\approx \frac{4}{\xi \omega_n}$ for $\pm 2\%$.
- The maximum overshoot is large for low damping $\xi$, and small for high $\xi \rightarrow$ Peak overshoot is $e^{-(\xi/\sqrt{1-\xi^2})\pi}$.
- The roots of the denominator closed-loop polynomial are complex with negative real parts.
- Roots are in the left-half of the $s$ plane $\rightarrow$ second-order system is stable.

Output $\Omega(t)$ to a step input shown in Figure 9(b).
Three possible kinds of behaviour

0 < $\xi$ < 1 – under-damped systems.

- Output oscillates about the desired input before settling down in *infinite* time.
- Settling time $t_s$ – Time taken for output to reach within $\pm 5\%$ (or $\pm 2\%$) of the input → For $\pm 5\%$ $t_s \approx \frac{3}{\xi \omega_n}$ and is $\approx \frac{4}{\xi \omega_n}$ for $\pm 2\%$.
- The maximum overshoot is large for low damping $\xi$, and small for high $\xi$ → Peak overshoot is $e^{-\left(\frac{\xi}{\sqrt{1-\xi^2}}\right)\pi}$.
- The roots of the denominator closed-loop polynomial are complex with negative real parts.
- Roots are in the left-half of the $s$ plane → second-order system is stable.

Output $\Omega(t)$ to a step input shown in Figure 9(b).
CONTROL OF A SINGLE LINK
Analysis – second-order system (Contd.)

- $\xi = 1$ – critically damped systems.
  - Output shows no oscillations and can cross input at most once.
  - Settling time can be defined similar to the under-damped case.
  - The roots of the denominator polynomial are real and repeated, and lie in the left-half of the $s$ plane.
  - Output $\Omega(t)$ for a step input shown in Figure 9(b).

- $\xi > 1$ – over-damped systems.
  - Output $\Omega(t)$ can never cross the input and is the sum of two exponential functions.
  - The roots of the denominator polynomial are real and distinct, and lie on the left-half of the $s$ plane.
  - Figure 9(b) shows a typical response of an over-damped system.
CONTROL OF A SINGLE LINK
ANALYSIS – second-order system (Contd.)

- $\xi = 1$ -- critically damped systems.
  - Output shows no oscillations and can cross input at most once.
  - Settling time can be defined similar to the under-damped case.
  - The roots of the denominator polynomial are real and repeated, and lie in the left-half of the $s$ plane.
  - Output $\Omega(t)$ for a step input shown in Figure 9(b).

- $\xi > 1$ -- over-damped systems.
  - Output $\Omega(t)$ can never cross the input and is the sum of two exponential functions.
  - The roots of the denominator polynomial are real and distinct, and lie on the left-half of the $s$ plane.
  - Figure 9(b) shows a typical response of an over-damped system.
Control of a Single Link
Analysis – Second-Order System (Contd.)

Figure 9: Second-order system and its step response \((K = J = F = 1)\)
CONTROL OF A SINGLE LINK
Analysis – second-order system (Contd.)

- For one link manipulator $\omega_n$ and $\xi$ depends on controller gain $K_p$

$$\omega_n^2 = \left(\frac{KK_p}{J}\right), \quad \xi = \frac{F}{2\sqrt{JKK_p}}$$

- Changing $K_p$ changes both $\omega_n$ and $\xi$.
- Can make the output under-damped, critically damped or over-damped by choosing $K_p$!!
- Simplest possible controller → Proportional Controller
- To choose $\omega_n$ and $\xi$ arbitrarily, two parameters needed → Proportional plus Derivative (PD) controller.
CONTROL OF A SINGLE LINK
ANALYSIS – second-order system (Contd.)

For one link manipulator $\omega_n$ and $\xi$ depends on controller gain $K_p$

$$\omega_n^2 = (K K_p / J), \quad \xi = \frac{F}{2 \sqrt{J K K_p}}$$

Changing $K_p$ changes both $\omega_n$ and $\xi$.

Can make the output under-damped, critically damped or over-damped by choosing $K_p$!!

Simplest possible controller $\rightarrow$ Proportional Controller

To choose $\omega_n$ and $\xi$ arbitrarily, two parameters needed $\rightarrow$ Proportional plus Derivative (PD) controller.
CONTROL OF A SINGLE LINK

ANALYSIS – second-order system (Contd.)

- For one link manipulator $\omega_n$ and $\xi$ depends on controller gain $K_p$

$$\omega_n^2 = (KK_p/J), \quad \xi = \frac{F}{2\sqrt{JKK_p}}$$

- Changing $K_p$ changes both $\omega_n$ and $\xi$.
- Can *make the output* under-damped, critically damped or over-damped by choosing $K_p$!
- Simplest possible controller → *Proportional Controller*
- To choose $\omega_n$ and $\xi$ *arbitrarily*, two parameters needed → Proportional plus Derivative (PD) controller.
CONTROL OF A SINGLE LINK

Analysis – second-order system (Contd.)

- For one link manipulator $\omega_n$ and $\xi$ depends on controller gain $K_p$

\[ \omega_n^2 = \frac{KK_p}{J}, \quad \xi = \frac{F}{2\sqrt{JKK_p}} \]

- Changing $K_p$ changes both $\omega_n$ and $\xi$.

- Can make the output under-damped, critically damped or over-damped by choosing $K_p$.

- Simplest possible controller $\rightarrow$ Proportional Controller

- To choose $\omega_n$ and $\xi$ arbitrarily, two parameters needed $\rightarrow$ Proportional plus Derivative (PD) controller.
For one link manipulator $\omega_n$ and $\xi$ depends on controller gain $K_p$

\[ \omega_n^2 = \frac{KK_p}{J}, \quad \xi = \frac{F}{2\sqrt{JKK_p}} \]

- Changing $K_p$ changes both $\omega_n$ and $\xi$.
- Can make the output under-damped, critically damped or over-damped by choosing $K_p$!!
- Simplest possible controller $\rightarrow$ Proportional Controller
- To choose $\omega_n$ and $\xi$ arbitrarily, two parameters needed $\rightarrow$ Proportional plus Derivative (PD) controller.
Controller transfer function $D(s) = K_p + K_v s$, $K_v$ derivative gain.

The closed-loop transfer function

$$\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p + sKK_v}{Js^2 + s(F + KK_v) + KK_p}$$

- $\omega_n$ and $\xi$ related to $K_p$ and $K_v$ and can be set arbitrarily.
- Increasing $K_v$ decreases overshoot but $t_s$ becomes larger! For critical damping $K_v = 2\sqrt{K_p}$
- To obtain desired performance, need to use (computer) tools developed by researchers (see Franklin et al., 1991).
CONTROL OF A SINGLE LINK
PID CONTROL

- Controller transfer function \( D(s) = K_p + K_v s \), \( K_v \) derivative gain.
- The closed-loop transfer function

\[
\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p + sKK_v}{Js^2 + s(F + KK_v) + KK_p}
\]

- \( \omega_n \) and \( \xi \) related to \( K_p \) and \( K_v \) and can be set arbitrarily.
- Increasing \( K_v \) decreases overshoot but \( t_s \) becomes larger! For critical damping \( K_v = 2\sqrt{K_p} \)
- To obtain desired performance, need to use (computer) tools developed by researchers (see Franklin et al., 1991).
**CONTROL OF A SINGLE LINK**

**PID CONTROL**

- Controller transfer function $D(s) = K_p + K_v s$, $K_v$ *derivative* gain.
- The closed-loop transfer function

\[
\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p + sKK_v}{Js^2 + s(F + KK_v) + KK_p}
\]

- $\omega_n$ and $\xi$ related to $K_p$ and $K_v$ and can be set *arbitrarily*.
- Increasing $K_v$ decreases overshoot but $t_s$ becomes larger! For critical damping $K_v = 2\sqrt{K_p}$
- To obtain desired performance, need to use (computer) tools developed by researchers (see Franklin et al., 1991).
CONTROL OF A SINGLE LINK

PID CONTROL

- Controller transfer function \( D(s) = K_p + K_v s \), \( K_v \) derivative gain.
- The closed-loop transfer function

\[
\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p + sKK_v}{Js^2 + s(F + KK_v) + KK_p}
\]

- \( \omega_n \) and \( \xi \) related to \( K_p \) and \( K_v \) and can be set arbitrarily.
- Increasing \( K_v \) decreases overshoot but \( t_s \) becomes larger! For critical damping \( K_v = 2\sqrt{K_p} \)
- To obtain desired performance, need to use (computer) tools developed by researchers (see Franklin et al., 1991).
CONTROL OF A SINGLE LINK
PID CONTROL

- Controller transfer function \( D(s) = K_p + K_v s \), \( K_v \) derivative gain.
- The closed-loop transfer function

\[
\frac{\theta(s)}{\theta_d(s)} = \frac{KK_p + sKK_v}{Js^2 + s(F + KK_v) + KK_p}
\]

- \( \omega_n \) and \( \xi \) related to \( K_p \) and \( K_v \) and can be set \textit{arbitrarily}.
- Increasing \( K_v \) decreases overshoot but \( t_s \) becomes larger! For critical damping \( K_v = 2\sqrt{K_p} \)
- To obtain desired performance, need to use (computer) tools developed by researchers (see Franklin et al., 1991).
CONTROL OF A SINGLE LINK

PID Control

- To decrease steady state error (from backlash, friction/stiction), *integral* term is used.
- Integral term $K_i/s - K_i$ is called the controller gain, must be chosen carefully $\rightarrow$ large $K_i$ can make system unstable!
- $sK_v$ term is not allowed$^2 \rightarrow$ PID controller

$$D(s) = K_p + \frac{K_i}{s} + \frac{K_v s}{1 + T_v s}$$

$T_v$ is a (chosen) time constant and $s/(1 + T_v s)$ represents a filter.

- In time domain $V_a(t) = K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$.
- Often *feed-forward* term added for improved *trajectory tracking* $\rightarrow$ modified PID controller

$$V_a(t) = \ddot{\theta}_d(t) + K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$$

---

$^2$In classical control, the numerator polynomial degree *must* be less than or equal to the denominator polynomial.
**CONTROL OF A SINGLE LINK**

**PID Control**

- To decrease steady state error (from backlash, friction/stiction), *integral* term is used.
- Integral term $K_i/s - K_i$ is called the controller gain, must be chosen carefully → large $K_i$ can make system unstable!
- $sK_v$ term is not allowed → PID controller

$$D(s) = K_p + \frac{K_i}{s} + \frac{K_v s}{1 + T_v s}$$

$T_v$ is a (chosen) time constant and $s/(1 + T_v s)$ represents a filter.

- In time domain $V_a(t) = K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$.
- Often *feed-forward* term added for improved *trajectory tracking* → modified PID controller

$$V_a(t) = \ddot{\theta}_d(t) + K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$$

---

2 In classical control, the numerator polynomial degree *must* be less than or equal to the denominator polynomial.
CONTROL OF A SINGLE LINK

PID CONTROL

- To decrease steady state error (from backlash, friction/stiction), integral term is used.
- Integral term $K_i/s - K_i$ is called the controller gain, must be chosen carefully $\rightarrow$ large $K_i$ can make system unstable!
- $sK_v$ term is not allowed$^2 \rightarrow$ PID controller

$$D(s) = K_p + \frac{K_i}{s} + \frac{K_v s}{1 + T_v s}$$

$T_v$ is a (chosen) time constant and $s/(1 + T_v s)$ represents a filter.

- In time domain $V_a(t) = K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t)dt$.
- Often feed-forward term added for improved trajectory tracking $\rightarrow$ modified PID controller

$$V_a(t) = \ddot{\theta}_d(t) + K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t)dt$$

$^2$In classical control, the numerator polynomial degree must be less than or equal to the denominator polynomial.
CONTROL OF A SINGLE LINK

PID CONTROL

- To decrease steady state error (from backlash, friction/stiction), *integral* term is used.
- Integral term $K_i/s - K_i$ is called the controller gain, must be chosen carefully → large $K_i$ can make system unstable!
- $sK_v$ term is not allowed$^2$ → PID controller
  
  $$D(s) = K_p + \frac{K_i}{s} + \frac{K_v s}{1 + T_v s}$$

  $T_v$ is a (chosen) time constant and $s/(1 + T_v s)$ represents a filter.
- In time domain $V_a(t) = K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$.
- Often *feed-forward* term added for improved *trajectory tracking* → modified PID controller
  
  $$V_a(t) = \ddot{\theta}_d(t) + K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$$

$^2$In classical control, the numerator polynomial degree *must* be less than or equal to the denominator polynomial.
CONTROL OF A SINGLE LINK

PID CONTROL

- To decrease steady state error (from backlash, friction/stiction), *integral* term is used.
- Integral term $K_i/s - K_i$ is called the controller gain, must be chosen carefully → large $K_i$ can make system unstable!
- $sK_v$ term is not allowed$^2$ → PID controller

$$D(s) = K_p + \frac{K_i}{s} + \frac{K_v s}{1 + T_v s}$$

$T_v$ is a (chosen) time constant and $s/(1 + T_v s)$ represents a filter.

- In time domain $V_a(t) = K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$.
- Often *feed-forward* term added for improved *trajectory tracking* → modified PID controller

$$V_a(t) = \ddot{\theta}_d(t) + K_p e(t) + K_v \dot{e}(t) + K_i \int_0^t e(t) dt$$

$^2$In classical control, the numerator polynomial degree *must* be less than or equal to the denominator polynomial.
CONTROL OF A SINGLE LINK

DIGITAL CONTROL

- Most modern controllers are implemented using digital microprocessors.
- No longer *continuous* time control → *discrete-time* control – Sampling.

![Figure 10: Discretisation of $\theta_d(t)$](image)

- Desired input $\theta_d(t)$ and the output $\theta(t)$ are *not* continuous → only dashed lines available.
- Analog to digital conversion is done electronically
- Typical sampling time, $T_s$, is between 1 and 10 milli-seconds and typically 8 – 12 bits used in A/D conversion.
- Less difference if number of bits in A/D conversion is more.
Sampling performed by an *independent* clock which *interrupts* the microprocessor.

- $\theta_d(kT_s)$ and $\theta(kT_s)$ are the $k$th desired and measured $\theta$.
- Error $e(kT_s) = \theta_d(kT_s) - \theta(kT_s)$ computed as a digital value.
CONTROL OF A SINGLE LINK

DIGITAL CONTROL (Contd.)

- Error is input to the controller $D(z) \rightarrow$ output is \textit{discretised} voltage.
- Discretised voltage \textit{converted} to analog in a D/A converter and using a zero order hold ZOH.
- The D/A and ZOH introduces \textit{delay} $\rightarrow$ source of many complications!
- Output of microprocessor in milliamperes $\rightarrow$ needs to be amplified to drive motor.
- Controller designed using \textit{discrete controls} and $z$ transform (see textbook by Franklin et al., 1990)

$$D(z) = K_p + \frac{K_i T_s}{1 - z^{-1}} + \frac{K_v (1 - z^{-1})}{T_s + T_v (1 - z^{-1})}$$
CONTROL OF A SINGLE LINK

DIGITAL CONTROL (Contd.)

- Error is input to the controller $D(z) \rightarrow$ output is discretised voltage.
- Discretised voltage converted to analog in a D/A converter and using a zero order hold ZOH.
- The D/A and ZOH introduces delay $\rightarrow$ source of many complications!
- Output of microprocessor in milliamperes $\rightarrow$ needs to be amplified to drive motor.
- Controller designed using discreet controls and z transform (see textbook by Franklin et al., 1990)

$$D(z) = K_p + \frac{K_i T_s}{1 - z^{-1}} + \frac{K_v (1 - z^{-1})}{T_s + T_v (1 - z^{-1})}$$
Error is input to the controller $D(z) \rightarrow$ output is *discretised* voltage.

Discretised voltage converted to analog in a D/A converter and using a zero order hold ZOH.

The D/A and ZOH introduces *delay* → source of many complications!

Output of microprocessor in milliamperes → needs to be amplified to drive motor.

Controller designed using *discrete controls* and $z$ transform (see textbook by Franklin et al., 1990)

$$D(z) = K_p + \frac{K_i T_s}{1 - z^{-1}} + \frac{K_v (1 - z^{-1})}{T_s + T_v (1 - z^{-1})}$$
CONTROL OF A SINGLE LINK
DIGITAL CONTROL (Contd.)

- Error is input to the controller $D(z) \rightarrow$ output is discretised voltage.
- Discretised voltage converted to analog in a D/A converter and using a zero order hold ZOH.
- The D/A and ZOH introduces delay $\rightarrow$ source of many complications!
- Output of microprocessor in milliamperes $\rightarrow$ needs to be amplified to drive motor.

Controller designed using discreet controls and $z$ transform (see textbook by Franklin et al., 1990)

$$D(z) = K_p + \frac{K_i T_s}{1 - z^{-1}} + \frac{K_v (1 - z^{-1})}{T_s + T_v (1 - z^{-1})}$$
Control of a Single Link
Digital Control (Contd.)

- Error is input to the controller $D(z) \rightarrow$ output is discretised voltage.
- Discretised voltage converted to analog in a D/A converter and using a zero order hold ZOH.
- The D/A and ZOH introduces delay $\rightarrow$ source of many complications!
- Output of microprocessor in milliamperes $\rightarrow$ needs to be amplified to drive motor.
- Controller designed using discreet controls and $z$ transform (see textbook by Franklin et al., 1990)

\[
D(z) = K_p + \frac{K_i T_s}{1 - z^{-1}} + \frac{K_v (1 - z^{-1})}{T_s + T_v (1 - z^{-1})}
\]
OUTLINE

1 CONTENTS

2 LECTURE 1
   • Motion planning

3 LECTURE 2
   • Control of a single link

4 LECTURE 3
   • Control of a multi-link serial manipulator

5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators

6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators

7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
CONTROL OF A MULTI-LINK SERIAL MANIPULATOR

Overview

- Multi-link $\rightarrow n$ joint variables – $\mathbf{q}$.
- Desired joint motion, $\mathbf{q}_d(t)$, available from motion planning.
- Assume $\dot{\mathbf{q}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ also available – see cubic trajectory plan!
- PD control of multi-link manipulator – actual implementation is PID.
- Non-linear control of multi-link manipulator.
- Simulation and experimental results.
CONTROL OF A MULTI-LINK SERIAL MANIPULATOR

OVERVIEW

- Multi-link $\rightarrow n$ joint variables – $\mathbf{q}$.
- Desired joint motion, $\mathbf{q}_d(t)$, available from motion planning.
  - Assume $\dot{\mathbf{q}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ also available – see cubic trajectory plan!
- PD control of multi-link manipulator – actual implementation is PID.
- Non-linear control of multi-link manipulator.
- Simulation and experimental results.
CONTROL OF A MULTI-LINK SERIAL MANIPULATOR

Overview

- Multi-link $\rightarrow n$ joint variables – $\mathbf{q}$.
- Desired joint motion, $\mathbf{q}_d(t)$, available from motion planning.
- Assume $\dot{\mathbf{q}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ also available – see cubic trajectory plan!
- PD control of multi-link manipulator – actual implementation is PID.
- Non-linear control of multi-link manipulator.
- Simulation and experimental results.
CONTROL OF A MULTI-LINK SERIAL MANIPULATOR

Overview

- Multi-link $\rightarrow$ $n$ joint variables – $q$.
- Desired joint motion, $q_d(t)$, available from motion planning.
- Assume $\dot{q}_d(t)$ and $\ddot{q}_d(t)$ also available – see cubic trajectory plan!
- PD control of multi-link manipulator – actual implementation is PID.
- Non-linear control of multi-link manipulator.
- Simulation and experimental results.
Multi-link $\rightarrow n$ joint variables – $\mathbf{q}$.

Desired joint motion, $\mathbf{q}_d(t)$, available from motion planning.

Assume $\dot{\mathbf{q}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ also available – see cubic trajectory plan!

PD control of multi-link manipulator – actual implementation is PID.

Non-linear control of multi-link manipulator.

Simulation and experimental results.
CONTROL OF A MULTI-LINK SERIAL MANIPULATOR

OVERVIEW

• Multi-link → $n$ joint variables – $\mathbf{q}$.
• Desired joint motion, $\mathbf{q}_d(t)$, available from motion planning.
• Assume $\dot{\mathbf{q}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ also available – see cubic trajectory plan!
• PD control of multi-link manipulator – actual implementation is PID.
• Non-linear control of multi-link manipulator.
• Simulation and experimental results.
PD CONTROL OF MULTI-LINK SERIAL MANIPULATOR

INTRODUCTION

- Extend continuous time control of single link manipulator.
- Feed-forward plus PD instead of PID control algorithm for analysis

\[ V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

Implemented control will also have a integral term!
- Use torque \( \tau \) acting at the joint instead of voltage \( V_a \) in analysis\(^3\).
- Control law used in analysis

\[ \tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

- Linear control law applied to a non-linear system!

---

\(^3\)Joint torque is related to the applied voltage at the motor terminals since 
\[ T_m = K_t i_a = \left( K_t / R_a \right) (V_a - K_g \dot{\theta}_m) \] and \( \tau = T_m / n \). One can also find \( V_a \) from motor characteristics curves.
PD CONTROL OF MULTI-LINK SERIAL MANIPULATOR

INTRODUCTION

● Extend continuous time control of single link manipulator.
● Feed-forward plus PD instead of PID control algorithm for analysis

\[ V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

Implemented control will also have a integral term!
● Use torque \( \tau \) acting at the joint instead of voltage \( V_a \) in analysis\(^3\).
● Control law used in analysis

\[ \tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

● Linear control law applied to a non-linear system!

---

\(^3\)Joint torque is related to the applied voltage at the motor terminals since \( T_m = K_t i_a = (K_t/R_a)(V_a - K_g \dot{\theta}_m) \) and \( \tau = T_m/n \). One can also find \( V_a \) from motor characteristics curves.
PD Control of Multi-Link Serial Manipulator

Introduction

- Extend continuous time control of single link manipulator.
- Feed-forward plus PD instead of PID control algorithm for analysis

\[ V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

Implemented control will also have a integral term!
- Use torque \( \tau \) acting at the joint instead of voltage \( V_a \) in analysis\(^3\).
- Control law used in analysis

\[ \tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

- Linear control law applied to a non-linear system!

---

\(^3\)Joint torque is related to the applied voltage at the motor terminals since \( T_m = K_t i_a = (K_t / R_a)(V_a - K_g \dot{\theta}_m) \) and \( \tau = T_m / n \). One can also find \( V_a \) from motor characteristics curves.
PD Control of Multi-Link Serial Manipulator

Introduction

- Extend continuous time control of single link manipulator.
- Feed-forward plus PD instead of PID control algorithm for analysis

$$V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t)$$

Implemented control will also have a integral term!

- Use torque $\tau$ acting at the joint instead of voltage $V_a$ in analysis$^3$.
- Control law used in analysis

$$\tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t)$$

- Linear control law applied to a non-linear system!

$^3$Joint torque is related to the applied voltage at the motor terminals since $T_m = K_t i_a = (K_t/R_a)(V_a - K_g \dot{\theta}_m)$ and $\tau = T_m/n$. One can also find $V_a$ from motor characteristics curves.
PD Control of Multi-link Serial Manipulator

Introduction

- Extend continuous time control of single link manipulator.
- Feed-forward plus PD instead of PID control algorithm for analysis

\[ V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

Implemented control will also have an integral term!
- Use torque \( \tau \) acting at the joint instead of voltage \( V_a \) in analysis\(^3\).
- Control law used in analysis

\[ \tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t) \]

- Linear control law applied to a non-linear system!

\(^3\)Joint torque is related to the applied voltage at the motor terminals since \( T_m = K_t i_a = (K_t / R_a)(V_a - K_g \dot{\theta}_m) \) and \( \tau = T_m / n \). One can also find \( V_a \) from motor characteristics curves.
**Figure 12:** PD control of a multi-link robot

- Each joint or motor independently controlled.
- All quantities, $q_d$, $q$, $\tau$ are $n \times 1$ vectors ($n$ DOF manipulator)
- $[K_p]$ and $[K_v]$ are $n \times n$ positive-definite proportional and derivative controller gain matrices.
PD Control of multi-link serial manipulator

Figure 12: PD control of a multi-link robot

- Each joint or motor independently controlled.
- All quantities, $q_d$, $q$, $\tau$ are $n \times 1$ vectors ($n$ DOF manipulator)
- $[K_p]$ and $[K_v]$ are $n \times n$ positive-definite proportional and derivative controller gain matrices.
Each joint or motor independently controlled.

All quantities, $q_d$, $q$, $\tau$ are $n \times 1$ vectors ($n$ DOF manipulator)

$[K_p]$ and $[K_v]$ are $n \times n$ positive-definite proportional and derivative controller gain matrices.
PD Control of Multi-Link Serial Manipulator

Introduction

- Multi-link manipulator is a non-linear system $\rightarrow$ Cannot expect uniform damping and settling time everywhere in workspace.
- Reason for working – slow speed and large gear ratio at joints!
- Linear control law implemented using one or more microprocessors
- Two main kinds of architecture commonly used.
  - Joint parallel – each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
  - Functional parallel – Each/group of function(s)/task(s) handled by a processor.
- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, $\theta_d$ available every 28 msec and $T_s$ for joint processor was 0.875 msec, high-level language VAL for robot programming.
- Modern solution – add-on cards for industrial PC’s to control several joints.
PD CONTROL OF MULTI-LINK SERIAL MANIPULATOR

INTRODUCTION

- Multi-link manipulator is a non-linear system → Cannot expect *uniform* damping and settling time *everywhere* in workspace.
- Reason for working – slow speed and large gear ratio at joints!
- Linear control law implemented using one or more microprocessors
- Two main kinds of architecture commonly used.
  - *Joint parallel* – each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
  - *Functional parallel* – Each/group of function(s)/task(s) handled by a processor.
- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, \( \theta_d \) available every 28 msec and \( T_s \) for joint processor was 0.875 msec, high-level language VAL for robot programming.
- Modern solution – add-on cards for industrial PC’s to control several joints.
PD Control of Multi-link Serial Manipulator

Introduction

- Multi-link manipulator is a non-linear system → Cannot expect uniform damping and settling time everywhere in workspace.

- Reason for working – slow speed and large gear ratio at joints!

- Linear control law implemented using one or more microprocessors

- Two main kinds of architecture commonly used.
  - Joint parallel – each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
  - Functional parallel – Each/group of function(s)/task(s) handled by a processor.

- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, $\theta_d$ available every 28 msec and $T_s$ for joint processor was 0.875 msec, high-level language VAL for robot programming.

- Modern solution – add-on cards for industrial PC’s to control several joints.
PD Control of Multi-link Serial Manipulator

Introduction

- Multi-link manipulator is a non-linear system → Cannot expect uniform damping and settling time everywhere in workspace.
- Reason for working – slow speed and large gear ratio at joints!
- Linear control law implemented using one or more microprocessors
- Two main kinds of architecture commonly used.
  - Joint parallel – each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
  - Functional parallel – Each/group of function(s)/task(s) handled by a processor.

- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, \( \theta_d \) available every 28 msec and \( T_s \) for joint processor was 0.875 msec, high-level language VAL for robot programming.
- Modern solution – add-on cards for industrial PC’s to control several joints.
PD CONTROL OF MULTI-LINK SERIAL MANIPULATOR

INTRODUCTION

- Multi-link manipulator is a non-linear system → Cannot expect uniform damping and settling time everywhere in workspace.
- Reason for working – slow speed and large gear ratio at joints!
- Linear control law implemented using one or more microprocessors
- Two main kinds of architecture commonly used.
  - *Joint parallel* – each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
  - *Functional parallel* – Each/group of function(s)/task(s) handled by a processor.
- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, $\theta_d$ available every 28 msec and $T_s$ for joint processor was 0.875 msec, high-level language VAL for robot programming.
- Modern solution – add-on cards for industrial PC’s to control several joints.
**PD Control of Multi-Link Serial Manipulator**

**Introduction**

- Multi-link manipulator is a non-linear system → Cannot expect *uniform* damping and settling time *everywhere* in workspace.
- Reason for working – slow speed and large gear ratio at joints!
- Linear control law implemented using one or more microprocessors
- Two main kinds of architecture commonly used.
  - *Joint parallel* – each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
  - *Functional parallel* – Each/group of function(s)/task(s) handled by a processor.

- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, $\theta_d$ available every 28 msec and $T_s$ for joint processor was 0.875 msec, high-level language VAL for robot programming.
- Modern solution – add-on cards for industrial PC’s to control several joints.
Non-linear control of multi-link serial manipulator

Introduction

- Non-linear control – a vast field!
- One particular kind of non-linear controller – *computed torque* (also called *feedback linearizing*) control scheme.
- In ideal situations can give *uniform* performance *everywhere* in workspace!
- Uses dynamic model in the control scheme.
- The better the estimate of the dynamic model, better the performance.
- Large amount of literature – first popularised by Freund (1982) in turn uses results of Singh and Rugh (1972)
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Introduction

- Non-linear control – a vast field!
- One particular kind of non-linear controller – *computed torque* (also called *feedback linearizing*) control scheme.
- In ideal situations can give *uniform* performance everywhere in workspace!
- Uses dynamic model in the control scheme.
- The better the estimate of the dynamic model, better the performance.
- Large amount of literature – first popularised by Freund (1982) in turn uses results of Singh and Rugh(1972)
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Introduction

- Non-linear control – a vast field!
- One particular kind of non-linear controller – *computed torque* (also called *feedback linearizing*) control scheme.
- In ideal situations can give *uniform* performance *everywhere* in workspace!
- Uses dynamic model in the control scheme.
- The better the estimate of the dynamic model, better the performance.
- Large amount of literature – first popularised by Freund (1982) in turn uses results of Singh and Rugh (1972)
Non-linear control of multi-link serial manipulator

Introduction

- Non-linear control – a vast field!
- One particular kind of non-linear controller – *computed torque* (also called *feedback linearizing*) control scheme.
- In ideal situations can give *uniform* performance *everywhere* in workspace!
- Uses dynamic model in the control scheme.
  - The better the estimate of the dynamic model, better the performance.
  - Large amount of literature – first popularised by Freund (1982) in turn uses results of Singh and Rugh (1972)
Non-linear control – a vast field!

One particular kind of non-linear controller – computed torque (also called feedback linearizing) control scheme.

In ideal situations can give uniform performance everywhere in workspace!

Uses dynamic model in the control scheme.

The better the estimate of the dynamic model, better the performance.

Large amount of literature – first popularised by Freund (1982) in turn uses results of Singh and Rugh (1972)
Non-linear control – a vast field!

One particular kind of non-linear controller – *computed torque* (also called *feedback linearizing*) control scheme.

In ideal situations can give *uniform* performance *everywhere* in workspace!

Uses dynamic model in the control scheme.

The better the estimate of the dynamic model, better the performance.

Large amount of literature – first popularised by Freund (1982) in turn uses results of Singh and Rugh (1972)
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Control Law Partitioning

- Dynamic equations of motion for a serial manipulator (see Module 6, Lecture 1)
  \[ \tau = [M(q)]\ddot{q} + C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]
  
  \([M(q)]\) is an \(n \times n\) mass matrix and \(C(q, \dot{q}), G(q),\) and \(F(q, \dot{q})\) are \(n \times 1\) vectors representing Coriolis/centripetal, gravity, and friction terms, respectively.

- Write \(n \times 1\) vector \(\tau\) of joint torques as,
  \[ \tau = [\alpha]\tau' + \beta \]

- Choose
  \[ [\alpha] = [M(q)], \quad \beta = C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]

- Using equations of motion,
  \[ \tau' = \ddot{q} \]
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Control law partitioning

- Dynamic equations of motion for a serial manipulator (see Module 6, Lecture 1)

\[ \tau = [M(q)]\ddot{q} + C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]

- \([M(q)]\) is an \(n \times n\) mass matrix and \(C(q, \dot{q}), G(q),\) and \(F(q, \dot{q})\) are \(n \times 1\) vectors representing Coriolis/centripetal, gravity, and friction terms, respectively.

- Write \(n \times 1\) vector \(\tau\) of joint torques as,

\[ \tau = [\alpha]\tau' + \beta \]

- Choose

\[ [\alpha] = [M(q)], \quad \beta = C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]

- Using equations of motion,

\[ \tau' = \ddot{q} \]
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

CONTROL LAW PARTITIONING

- Dynamic equations of motion for a serial manipulator (see Module 6, Lecture 1)

\[ \tau = [M(q)]\ddot{q} + C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]

- \([M(q)]\) is an \(n \times n\) mass matrix and \(C(q, \dot{q})\), \(G(q)\), and \(F(q, \dot{q})\) are \(n \times 1\) vectors representing Coriolis/centripetal, gravity, and friction terms, respectively.

- Write \(n \times 1\) vector \(\tau\) of joint torques as,

\[ \tau = [\alpha] \tau' + \beta \]

- Choose \([\alpha] = [M(q)]\), \(\beta = C(q, \dot{q}) + G(q) + F(q, \dot{q})\)

- Using equations of motion,

\[ \tau' = \ddot{q} \]
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

CONTROL LAW PARTITIONING

- Dynamic equations of motion for a serial manipulator (see Module 6, Lecture 1)
  \[ \tau = [M(q)]\ddot{q} + C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]
  
  
  - $[M(q)]$ is an $n \times n$ mass matrix and $C(q, \dot{q})$, $G(q)$, and $F(q, \dot{q})$ are $n \times 1$ vectors representing Coriolis/centripetal, gravity, and friction terms, respectively.

- Write $n \times 1$ vector $\tau$ of joint torques as,
  \[ \tau = [\alpha]\tau' + \beta \]

- Choose
  \[ [\alpha] = [M(q)], \quad \beta = C(q, \dot{q}) + G(q) + F(q, \dot{q}) \]

- Using equations of motion,
  \[ \tau' = \ddot{q} \]
Non-linear control of multi-link serial manipulator

Control law partitioning (Contd.)

- The equation $\tau' = \ddot{q}$ represents a unit inertia system with input $\tau'$.
- The dynamics represented by $[\alpha]$ and $\beta$ are used.
- All non-linearities & coupling are ‘canceled’ and original non-linear equations transformed to $n$ decoupled linear equations.
- Choose
  $$\tau' = \ddot{q}_d(t) + [K_v]\dot{e}(t) + [K_p]e(t)$$
- Error equation becomes
  $$\ddot{e}(t) + [K_v]\dot{e}(t) + [K_p]e(t) = 0$$
- Choose positive-definite, diagonal matrices $[K_p]$ and $[K_v]$, to get critical damping at every point in the workspace!!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Control law partitioning (Contd.)

The equation \( \tau' = \ddot{q} \) represents a unit inertia system with input \( \tau' \).

- The dynamics represented by \( [\alpha] \) and \( \beta \) are used.

- All non-linearities & coupling are ‘canceled’ and original non-linear equations transformed to \( n \) decoupled linear equations.

- Choose

\[
\tau' = \ddot{q}_d(t) + [K_v]\dot{e}(t) + [K_p]e(t)
\]

- Error equation becomes

\[
\dot{e}(t) + [K_v]\dot{e}(t) + [K_p]e(t) = 0
\]

- Choose positive-definite, diagonal matrices \([K_p]\) and \([K_v]\), to get critical damping at every point in the workspace!!
The equation $\tau' = \ddot{q}$ represents a unit inertia system with input $\tau'$. The dynamics represented by $[\alpha]$ and $\beta$ are used. All non-linearities & coupling are ‘canceled’ and original non-linear equations transformed to $n$ decoupled linear equations.

Choose

$$\tau' = \ddot{q}_d(t) + [K_v]\dot{e}(t) + [K_p]e(t)$$

Error equation becomes

$$\ddot{e}(t) + [K_v]\dot{e}(t) + [K_p]e(t) = 0$$

Choose positive-definite, diagonal matrices $[K_p]$ and $[K_v]$, to get critical damping at every point in the workspace!!
The equation $\tau' = \ddot{q}$ represents a unit inertia system with input $\tau'$. The dynamics represented by $[\alpha]$ and $\beta$ are used. All non-linearities & coupling are ‘canceled’ and original non-linear equations transformed to $n$ decoupled linear equations. Choose

$$\tau' = \ddot{q}_d(t) + [K_v]\dot{e}(t) + [K_p]e(t)$$

Error equation becomes

$$\ddot{e}(t) + [K_v]\dot{e}(t) + [K_p]e(t) = 0$$

Choose positive-definite, diagonal matrices $[K_p]$ and $[K_v]$, to get critical damping at every point in the workspace!!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Control law partitioning (Contd.)

• The equation \( \tau' = \ddot{q} \) represents a unit inertia system with input \( \tau' \).
• The dynamics represented by \([\alpha]\) and \(\beta\) are used.
• All non-linearities & coupling are ‘canceled’ and original non-linear equations transformed to \(n\) decoupled linear equations.
• Choose
  \[
  \tau' = \ddot{q}_d(t) + [K_v]\dot{e}(t) + [K_p]e(t)
  \]
• Error equation becomes
  \[
  \ddot{e}(t) + [K_v]\dot{e}(t) + [K_p]e(t) = 0
  \]
• Choose positive-definite, diagonal matrices \([K_p]\) and \([K_v]\), to get critical damping at every point in the workspace!!
The equation $\tau' = \ddot{q}$ represents a *unit* inertia system with input $\tau'$.

The dynamics represented by $[\alpha]$ and $\beta$ are used.

All *non-linearities & coupling* are ‘canceled’ and original non-linear equations *transformed* to $n$ *decoupled linear* equations.

Choose

$$\tau' = \ddot{q}_d(t) + [K_v]\dot{e}(t) + [K_p]e(t)$$

Error equation becomes

$$\ddot{e}(t) + [K_v]\dot{e}(t) + [K_p]e(t) = 0$$

Choose positive-definite, diagonal matrices $[K_p]$ and $[K_v]$, to get *critical* damping at *every* point in the workspace!!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Figure 13: Computed torque control scheme for robots

- Two partitions – Error driven PD control and Model-based

\[ M(q) \hat{C}(q, \dot{q}) + \hat{G}(q) + \hat{F}(q, \dot{q}) \]

\[ \tau = \hat{M}(q) \dot{\dot{q}} + \hat{C}(q, \dot{q}) + \hat{G}(q) + \hat{F}(q, \dot{q}) \]
Non-linear control of multi-link serial manipulator

Control law partitioning (Contd.)

- “Ideal” performance not possible
  - Time required to compute $[\alpha]$ and $\beta \rightarrow$ during this time $q$ changes!
  - Manipulator parameters such as mass, inertia etc. not known exactly!
- Only estimates of $[M(q)]$, $C(q, \dot{q})$, $G(q)$ and $F(q, \dot{q})$ available $\rightarrow$ symbol $[\hat{M}(q)]$ etc. used in figure.
- Estimates $\rightarrow$ Error equation no longer linear and decoupled.
- If $[\alpha] = [\hat{M}(q)]$ and $\beta = C(q, \dot{q}) + G(q) + F(q, \dot{q})$, then error equation

\[
\ddot{e} + [K_v] \dot{e} + [K_p] e = [\hat{M}]^{-1} \left( ([M] - [\hat{M}]) \ddot{q} + (C - \hat{C}) + (G - \hat{G}) + (F - \hat{F}) \right)
\]

- If $([M] - [\hat{M}]) = [0]$ etc. then “exact cancellation”.

Ashitava Ghosal (IISc) ROBOTICS: ADVANCED CONCEPTS & ANALYSIS NPTEL, 2010 57 / 129
**NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR**

**Control Law Partitioning (Contd.)**

- “Ideal” performance not possible
  - Time required to compute $[\alpha]$ and $\beta \rightarrow$ during this time $q$ changes!
  - Manipulator parameters such as mass, inertia etc. not known *exactly*!

- Only estimates of $[M(q)]$, $C(q, \dot{q})$, $G(q)$ and $F(q, \dot{q})$ available $\rightarrow$
  - symbol $[\hat{M}(q)]$ etc. used in figure.

- Error equation no longer linear and decoupled.

- If $[\alpha] = [\hat{M}(q)]$ and $\beta = C(q, \dot{q}) + G(q) + F(q, \dot{q})$, then error equation

\[
\ddot{e} + [K_v] \dot{e} + [K_p] e = [\hat{M}]^{-1} \left( ([M] - [\hat{M}]) \ddot{q} + (C - \hat{C}) + (G - \hat{G}) + (F - \hat{F}) \right)
\]

- If $([M] - [\hat{M}]) = [0]$ etc. then “exact cancellation”.

---

Ashitava Ghosal (IISc)  
Robotics: Advanced Concepts & Analysis  
NPTEL, 2010  
57 / 129
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

**Control law partitioning (Contd.)**

- “Ideal” performance not possible
  - Time required to compute $[\alpha]$ and $\beta \rightarrow$ during this time $q$ changes!
  - Manipulator parameters such as mass, inertia etc. not known *exactly*!

- Only estimates of $[M(q)]$, $C(q, \dot{q})$, $G(q)$ and $F(q, \dot{q})$ available $\rightarrow$
symbol $[\hat{M}(q)]$ etc. used in figure.

- Estimates $\rightarrow$ Error equation no longer linear and decoupled.

  If $[\alpha] = [\hat{M}(q)]$ and $\beta = C(q, \dot{q}) + G(q) + F(q, \dot{q})$, then error equation

  $\ddot{e} + [K_v] \dot{e} + [K_p] e$

  $= [\hat{M}]^{-1} \left( ([M] - [\hat{M}]) \ddot{q} + (C - \hat{C}) + (G - \hat{G}) + (F - \hat{F}) \right)$

  If $([M] - [\hat{M}]) = [0]$ etc. then “exact cancellation”. 

Asitava Ghosal (IISc)  ROBOTICS: ADVANCED CONCEPTS & ANALYSIS  NPTEL, 2010  57 / 129
Non-linear control of multi-link serial manipulator

Control law partitioning (Contd.)

- “Ideal” performance not possible
  - Time required to compute $[\alpha]$ and $\beta \rightarrow$ during this time $q$ changes!
  - Manipulator parameters such as mass, inertia etc. not known exactly!

- Only estimates of $[M(q)]$, $C(q, \dot{q})$, $G(q)$ and $F(q, \dot{q})$ available $\rightarrow$
  symbol $\hat{M}(q)$ etc. used in figure.

- Estimates $\rightarrow$ Error equation no longer linear and decoupled.

- If $[\alpha] = [\hat{M}(q)]$ and $\beta = C(q, \dot{q}) + \hat{G}(q) + \hat{F}(q, \dot{q})$, then error equation

  $\ddot{e} + [K_v] \dot{e} + [K_p] e$

  $$= [\hat{M}]^{-1} \left( ([M] - [\hat{M}])\ddot{q} + (C - \hat{C}) + (G - \hat{G}) + (F - \hat{F}) \right)$$

- If $([M] - [\hat{M}]) = [0]$ etc. then “exact cancellation”.

Ashitava Ghosal (IISc)           Robotics: Advanced Concepts & Analysis  NPTEL, 2010  57 / 129
ONLINE CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Control law partitioning (Contd.)

- “Ideal” performance not possible
  - Time required to compute $[\alpha]$ and $\beta \rightarrow$ during this time $q$ changes!
  - Manipulator parameters such as mass, inertia etc. not known exactly!

- Only estimates of $[M(q)]$, $C(q, \dot{q})$, $G(q)$ and $F(q, \dot{q})$ available $\rightarrow$ symbol $[\hat{M}(q)]$ etc. used in figure.

- Estimates $\rightarrow$ Error equation no longer linear and decoupled.

- If $[\alpha] = [\hat{M}(q)]$ and $\beta = C(q, \dot{q}) + G(q) + F(q, \dot{q})$, then error equation
  $$\ddot{e} + [K_v]\dot{e} + [K_p]e = [\hat{M}]^{-1} \left( ([M] - [\hat{M}])\ddot{q} + (C - \hat{C}) + (G - \hat{G}) + (F - \hat{F}) \right)$$

- If $([M] - [\hat{M}]) = [0]$ etc. then “exact cancellation”.
Non-linear control of multi-link serial manipulator

Control law partitioning (Contd.)

- Special cases of computed torque scheme
  - \([\alpha] = [U] \) and \(\beta = G(q)\) → Gravity compensation.
  - No model used → \([\alpha] = [U] \) and \(\beta = 0\) → PD control scheme.
  - Feed-forward control law

\[
[\alpha] = \[\hat{M}(q_d)\], \quad \beta = \[\hat{C}(q_d, \dot{q_d}) + \hat{G}(q_d) + \hat{F}(q_d, \dot{q_d})\]
\]

- Model terms computed according to desired trajectory and not in the feed-back loop.
- Model terms can be computed off-line → Almost no issue of computation time.

- No “exact” cancellation in special cases → No decoupling or linearity.
- If estimates are good, then right-hand side is small! → performance better than PD.
- Borne out by simulations and experiments.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Control law partitioning (Contd.)

- Special cases of computed torque scheme
  - \([\alpha] = [U] \text{ and } \beta = G(q) \rightarrow \text{ Gravity compensation.}\)
  - No model used \(\rightarrow [\alpha] = [U] \text{ and } \beta = 0 \rightarrow \text{ PD control scheme.}\)
  - Feed-forward control law
    \[
    [\alpha] = [\hat{M}(q_d)], \quad \beta = C(q_d, \dot{q}_d) + \hat{G}(q_d) + \hat{F}(q_d, \dot{q}_d)
    \]
  - Model terms computed according to desired trajectory and not in the feed-back loop.
  - Model terms can be computed off-line \(\rightarrow \text{ Almost no issue of computation time.}\)
  - No “exact” cancellation in special cases \(\rightarrow \text{ No decoupling or linearity.}\)
    - If estimates are good, then right-hand side is small! \(\rightarrow \) performance better than PD.
    - Borne out by simulations and experiments.
Special cases of computed torque scheme

- \([\alpha] = [U] \) and \(\beta = G(q)\) → Gravity compensation.
- No model used → \([\alpha] = [U] \) and \(\beta = 0\) → PD control scheme.
- Feed-forward control law

\[
[\alpha] = [\overrightarrow{M(q_d)}], \quad \beta = C(q_d, \dot{q}_d) + \overrightarrow{G(q_d)} + \overrightarrow{F(q_d, \dot{q}_d)}
\]

- Model terms computed according to desired trajectory and not in the feed-back loop.
- Model terms can be computed off-line → Almost no issue of computation time.

- No “exact” cancellation in special cases → No decoupling or linearity.
- If estimates are good, then right-hand side is small! → performance better than PD.

- Borne out by simulations and experiments.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

CONTROL LAW PARTITIONING (Contd.)

- Special cases of computed torque scheme
  - \([\alpha] = [U] \text{ and } \beta = G(q) \rightarrow \text{Gravity compensation.}\)
  - No model used \(\rightarrow [\alpha] = [U] \text{ and } \beta = 0 \rightarrow \text{PD control scheme.}\)
  - \text{Feed-forward control law}
    \[
    [\alpha] = [\hat{M}(q_d)], \quad \beta = C(q_d, \dot{q}_d) + \hat{G}(\dot{q}_d) + \hat{F}(q_d, \dot{q}_d)
    \]
  - Model terms computed according to \textit{desired} trajectory and \textit{not} in the feed-back loop.
  - Model terms can be computed off-line \(\rightarrow \text{Almost no issue of computation time.}\)

- No “exact” cancellation in special cases \(\rightarrow \text{No decoupling or linearity.}\)
- If estimates are good, then right-hand side is small! \(\rightarrow \text{performance better than PD.}\)
- Borne out by simulations and experiments.
**Non-linear control of multi-link serial manipulator**

Simulation results

- Planar 2R robot shown in 2 configurations.
- Link 1 parameters – $l_1 = 1m$, $r_1 = 0.773m$, $m_1 = 12.456kg$ and $l_1 = 1.042$ kg – m$^2$.
- Link 2 parameters – $l_1 = 1m$, $r_1 = 0.583m$, $m_1 = 12.456kg$ and $l_1 = 1.042$ kg – m$^2$.
- Payload at the end 2.5 kg.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

- Tip moves up from (0, 0.55m) to (0, 1.45m) and back to (0, 0.55m).
- Two cases: (a) *fast*: total time is 2 sec, (b) *slow*: total time is 2 min.
- Smooth Cartesian cubic trajectories generated.

**Figure 15:** Desired Cartesian trajectory

**Figure 16:** Desired $\theta_1(t)$ and $\theta_2(t)$
**Non-linear Control of Multi-link Serial Manipulator**

Simulation Results (Contd.)

- Desired $\theta_{id}(t)$, $i = 1, 2$ and derivatives obtained using *inverse kinematics*

- Simulation results presented for
  - PD control scheme
  - Feed-forward controller with an *exact* knowledge of the model parameters,
  - Model-based controller with 10% error in $m_i$ and 5% error in $r_i$
  - Cartesian control scheme (discussed later).

- Gain values $K_{pi}$, $K_{vi}$ are chosen such that $\omega_1 = 85.0$, $\omega_2 = 75.0$, and $\xi_i$ are 2.0 → system over-damped.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Simulation results (Contd.)

- Desired $\theta_{id}(t), \ i = 1, 2$ and derivatives obtained using inverse kinematics
- Simulation results presented for
  - PD control scheme
  - Feed-forward controller with an exact knowledge of the model parameters,
  - Model-based controller with 10% error in $m_i$ and 5% error in $r_i$
  - Cartesian control scheme (discussed later).
- Gain values $K_{pi}, K_{vi}$ are chosen such that $\omega_1 = 85.0, \omega_2 = 75.0$, and $\xi_i$ are 2.0 → system over-damped.
Desired $\theta_{id}(t), \ i = 1, 2$ and derivatives obtained using inverse kinematics

Simulation results presented for
- PD control scheme
- Feed-forward controller with an exact knowledge of the model parameters,
- Model-based controller with 10% error in $m_i$ and 5% error in $r_i$
- Cartesian control scheme (discussed later).

Gain values $K_{pi}, K_{vi}$ are chosen such that $\omega_1 = 85.0, \ \omega_2 = 75.0$, and $\xi_i$ are 2.0 $\rightarrow$ system over-damped.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

SIMULATION RESULTS – PD CONTROL

(a) Error in $\theta_1$, $\theta_2$ for fast motion

(b) Error in $x$, $y$ for fast motion

(c) Torque at two joints for fast motion

(d) Error in $\theta_1$, $\theta_2$ for slow motion

(e) Error in $x$, $y$ for slow motion

(f) Torque at two joints for slow motion
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

SIMULATION RESULTS – PD CONTROL

- Maximum error in joint variables larger in case of \textit{fast} motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.

- Fast motion $\rightarrow$ Non-linear inertia, centripetal/Coriolis terms larger
- Linear PD control less effective \textit{as expected}!
- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m
- Torque larger in fast motion due to non-linear terms in equations of motion!
- Curves much smoother in slow motion.
- Non-linear controller results next!!
ON-LINE CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Simulation results – PD control

- Maximum error in joint variables larger in case of *fast* motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.

- Fast motion $\rightarrow$ Non-linear inertia, centripetal/Coriolis terms larger

- Linear PD control less effective *as expected!*

- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m

- Torque larger in fast motion due to non-linear terms in equations of motion!

- Curves much smoother in slow motion.

- Non-linear controller results next!!
ON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Simulation results – PD control

- Maximum error in joint variables larger in case of *fast* motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.

- Fast motion → Non-linear inertia, centripetal/Coriolis terms larger

- Linear PD control less effective *as expected*!
  - Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m
  - Torque larger in fast motion due to non-linear terms in equations of motion!

- Curves much smoother in slow motion.
- Non-linear controller results next!!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

SIMULATION RESULTS – PD CONTROL

- Maximum error in joint variables larger in case of fast motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.
- Fast motion → Non-linear inertia, centripetal/Coriolis terms larger
- Linear PD control less effective as expected!
- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m
  - Torque larger in fast motion due to non-linear terms in equations of motion!
- Curves much smoother in slow motion.
- Non-linear controller results next!!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

SIMULATION RESULTS – PD CONTROL

- Maximum error in joint variables larger in case of *fast* motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.

- Fast motion → Non-linear inertia, centripetal/Coriolis terms larger

- Linear PD control less effective *as expected!*

- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m

- Torque larger in fast motion due to non-linear terms in equations of motion!

- Curves much smoother in slow motion.

- Non-linear controller results next!!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Simulation results – PD control

- Maximum error in joint variables larger in case of fast motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.
- Fast motion → Non-linear inertia, centripetal/Coriolis terms larger
- Linear PD control less effective as expected!
- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m
- Torque larger in fast motion due to non-linear terms in equations of motion!
- Curves much smoother in slow motion.
- Non-linear controller results next!!
Non-linear control of multi-link serial manipulator

Simulation results – PD control

- Maximum error in joint variables larger in case of fast motion.
  - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
  - Approximately 0.023 m in fast versus 0.016 m in slow motion.

- Fast motion → Non-linear inertia, centripetal/Coriolis terms larger

- Linear PD control less effective as expected!

- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m

- Torque larger in fast motion due to non-linear terms in equations of motion!

- Curves much smoother in slow motion.

- Non-linear controller results next!!
Non-linear control of multi-link serial manipulator

Simulation results – Non-linear controllers (fast motions)

(g) Trajectory errors and torques for feed-forward controller

(h) Trajectory errors and torques for computed torque controller with uncertainties

(i) Trajectory errors and torques for Cartesian controller

- Feed-forward controller without model uncertainties is very accurate.
- Computed torque with 10% uncertainties more accurate than PD.
- Torque profiles are smoother – similar to PD control for slow motion → effect of non-linearities reduced!!
The PD (PID) control scheme is not suitable for high-speed applications and the errors can be large. To reduce errors, we need to perform trial and error. The performance for slow-speed operation is better and one can get smooth torque profiles.

Model-based schemes show improved performance in simulation. The torques are lower and the profile is also smoother. The lack of the knowledge of parameters degrades the performance only to a small extent.

The computation times for the model-based control are larger, but can be easily handled by newer processors.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results

- Five DOF pink-and-place robot, all DOF rotary, $\theta_i, \ i = 1, ..., 5$.
- A four-bar linkage drive joint 3 – Motors for joint 2 and 3 are on platform rotated by Motor 1 $\rightarrow$ Motor 2 “see” less inertia
- All motors are two-phase AC motors with large gear reduction.
- Significant backlash and friction in the gears.
- Encoders and tacho-generators measure joint rotation and velocity.

Figure 17: Schematic of a five-axis servo manipulator
Non-linear control of multi-link serial manipulator

Experimental results (Contd.)

- Existing control law \( V_i(t) = K_{p_i}(\theta_{i_d} - \theta_i) - K_{v_i} \dot{\theta}_i, \quad i = 1, ..., 5 \)
- Voltage \( V_i(t) \) applied at motor \( i \).
- Subset of PD control law – available \( \dot{\theta}_{i_d} \) and \( \ddot{\theta}_{i_d} \) not used.
- Modify existing desired joint rotation to

\[
\theta_{i_d}^* = \theta_{i_d} + \frac{1}{K_{p_i}} \ddot{\theta}_{i_d} + \frac{K_{v_i}}{K_{p_i}} \dot{\theta}_{i_d}, \quad i = 1, ..., 5
\]

- Modified control law with \( \theta_{i_d}^* \) → PD Control Law.

\[
V_i(t) = K_{p_i}(\theta_{i_d}^* - \theta_i) - K_{v_i} \dot{\theta}_i \\
= \ddot{\theta}_{i_d} + K_{p_i}(\theta_{i_d} - \theta_i) + K_{v_i}(\dot{\theta}_{i_d} - \dot{\theta}_i), \quad i = 1, ..., 5
\]
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Existing control law \( V_i(t) = K_{pi}(\theta_{id} - \theta_i) - K_{vi}\dot{\theta}_i, \quad i = 1, \ldots, 5 \)
- Voltage \( V_i(t) \) applied at motor \( i \).
- Subset of PD control law – available \( \dot{\theta}_{id} \) and \( \ddot{\theta}_{id} \) not used.
- Modify existing desired joint rotation to

\[
\theta_{id}^* = \theta_{id} + \frac{1}{K_{pi}} \ddot{\theta}_{id} + \frac{K_{vi}}{K_{pi}} \dot{\theta}_{id}, \quad i = 1, \ldots, 5
\]

- Modified control law with \( \theta_{id}^* \rightarrow \) PD Control Law.

\[
V_i(t) = K_{pi}(\theta_{id}^* - \theta_i) - K_{vi}\dot{\theta}_i \\
= \ddot{\theta}_{id} + K_{pi}(\theta_{id} - \theta_i) + K_{vi}(\dot{\theta}_{id} - \dot{\theta}_i), \quad i = 1, \ldots, 5
\]
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Existing control law \( V_i(t) = K_{pi}(\theta_{id} - \theta_i) - K_{vi} \dot{\theta}_i, \quad i = 1, \ldots, 5 \)
- Voltage \( V_i(t) \) applied at motor \( i \).
- Subset of PD control law – available \( \dot{\theta}_{id} \) and \( \ddot{\theta}_{id} \) not used.
- Modify existing desired joint rotation to

\[
\theta_{id}^* = \theta_{id} + \frac{1}{K_{pi}} \ddot{\theta}_{id} + \frac{K_{vi}}{K_{pi}} \dot{\theta}_{id}, \quad i = 1, \ldots, 5
\]

- Modified control law with \( \theta_{id}^* \rightarrow \) PD Control Law.

\[
V_i(t) = K_{pi}(\theta_{id}^* - \theta_i) - K_{vi} \dot{\theta}_i \\
= \ddot{\theta}_{id} + K_{pi}(\theta_{id} - \theta_i) + K_{vi}(\dot{\theta}_{id} - \dot{\theta}_i), \quad i = 1, \ldots, 5
\]
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Existing control law $V_i(t) = K_{p_i}(\theta_{i_d} - \theta_i) - K_{v_i} \dot{\theta}_i$, $i = 1, \ldots, 5$
- Voltage $V_i(t)$ applied at motor $i$.
- Subset of PD control law – available $\dot{\theta}_{i_d}$ and $\ddot{\theta}_{i_d}$ not used.
- Modify existing desired joint rotation to

$$\theta_{i_d}^* = \theta_{i_d} + \frac{1}{K_{p_i}} \ddot{\theta}_{i_d} + \frac{K_{v_i}}{K_{p_i}} \dot{\theta}_{i_d}, \quad i = 1, \ldots, 5$$

- Modified control law with $\theta_{i_d}^* \rightarrow$ PD Control Law.

$$V_i(t) = K_{p_i}(\theta_{i_d}^* - \theta_i) - K_{v_i} \dot{\theta}_i$$
$$= \ddot{\theta}_{i_d} + K_{p_i}(\theta_{i_d} - \theta_i) + K_{v_i}(\dot{\theta}_{i_d} - \dot{\theta}_i), \quad i = 1, \ldots, 5$$
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Existing control law \( V_i(t) = K_{p_i}(\theta_{i_d} - \theta_i) - K_{v_i} \dot{\theta_i}, \quad i = 1, ..., 5 \)
- Voltage \( V_i(t) \) applied at motor \( i \).
- Subset of PD control law – available \( \dot{\theta}_{i_d} \) and \( \ddot{\theta}_{i_d} \) not used.
- Modify existing desired joint rotation to
  \[
  \theta_{i_d}^* = \theta_{i_d} + \frac{1}{K_{p_i}} \ddot{\theta}_{i_d} + \frac{K_{v_i}}{K_{p_i}} \dot{\theta}_{i_d}, \quad i = 1, ..., 5
  \]
- Modified control law with \( \theta_{i_d}^* \rightarrow \) PD Control Law.
  \[
  V_i(t) = K_{p_i}(\theta_{i_d}^* - \theta_i) - K_{v_i} \dot{\theta_i} \\
  = \ddot{\theta}_{i_d} + K_{p_i}(\theta_{i_d} - \theta_i) + K_{v_i}(\dot{\theta}_{i_d} - \dot{\theta_i}), \quad i = 1, ..., 5
  \]
Non-linear control of multi-link serial manipulator  

Experimental results (Contd.)  

- Similar idea used to modify existing controller to a *model-based* control scheme – Modify desired $\theta_{i_d}$ with  

$$
\theta_{i_d}^* = \frac{V_{i_{mdl}}}{K_{p_i}} + \theta_{i_d} + \frac{1}{K_{p_i}} \dot{\theta}_{i_d} + \frac{K_{v_i}}{K_{p_i}} \ddot{\theta}_{i_d}, \quad i = 1, \ldots, 5
$$

where $V_{i_{mdl}}$, corresponding to $\tau_{i_{mdl}}$, computed from  

$$
\tau_{mdl} = [M(\theta_d)]\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d) + G(\theta_d)
$$

with available motor characteristics chart.  

- Above control law is analogous to *feed-forward* law

$$
\tau = \tau_{model} + \ddot{\theta}_d + [K_p](\theta_d - \theta) + [K_v](\dot{\theta}_d - \dot{\theta})
$$

- Model parameters required for $\theta_{i_d}^*$ from CAD model of robot.  
- Computed $\theta_{i_d}^*$ instead of $\theta_{i_d}$ used as reference input.  
- Above approach does *not* change any electronics or hardware!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Similar idea used to modify existing controller to a *model-based* control scheme – Modify desired $\theta_{id}$ with

$$\theta_{id}^* = \frac{V_{mdl}}{K_{pi}} + \theta_{id} + \frac{K_v}{K_{pi}} \dot{\theta}_{id} + \frac{K_p}{K_{pi}} \ddot{\theta}_{id}, \quad i = 1, \ldots, 5$$

where $V_{mdl}$, corresponding to $\tau_{mdl}$ computed from

$$\tau_{mdl} = [M(\theta_d)] \ddot{\theta}_d + C(\theta_d, \dot{\theta}_d) + G(\theta_d)$$

with available motor characteristics chart.

- Above control law is analogous to *feed-forward* law

$$\tau = \tau_{model} + \ddot{\theta}_d + [K_p](\theta_d - \theta) + [K_v](\dot{\theta}_d - \dot{\theta})$$

- Model parameters required for $\theta_{id}^*$ from CAD model of robot.
- Computed $\theta_{id}^*$ instead of $\theta_{id}$ used as reference input.
- Above approach *does not* change any electronics or hardware!
Similar idea used to modify existing controller to a model-based control scheme – Modify desired $\theta_{id}$ with

$$\theta_{id}^* = \frac{V_{mdl}}{K_p} + \theta_{id} + \frac{1}{K_p} \dot{\theta}_{id} + \frac{K_v}{K_p} \ddot{\theta}_{id}, \quad i = 1, \ldots, 5$$

where $V_{mdl}$, corresponding to $\tau_{mdl}$ computed from

$$\tau_{mdl} = [M(\theta_d)]\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d) + G(\theta_d)$$

with available motor characteristics chart.

Above control law is analogous to feed-forward law

$$\tau = \tau_{model} + \ddot{\theta}_d + [K_p](\theta_d - \theta) + [K_v](\dot{\theta}_d - \dot{\theta})$$

Model parameters required for $\theta_{id}^*$ from CAD model of robot.

Computed $\theta_{id}^*$ instead of $\theta_{id}$ used as reference input.

Above approach does not change any electronics or hardware!
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Similar idea used to modify existing controller to a *model-based* control scheme – Modify desired $\theta_{id}$ with

$$\theta_{id}^* = \frac{V_{mdl}}{K_{pi}} + \theta_{id} + \frac{1}{K_{pi}} \dot{\theta}_{id} + \frac{K_v}{K_{pi}} \ddot{\theta}_{id}, \quad i = 1, \ldots, 5$$

where $V_{mdl}$, corresponding to $\tau_{mdl}$ computed from

$$\tau_{mdl} = [M(\theta_d)]\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d) + G(\theta_d)$$

with available motor characteristics chart.

- Above control law is analogous to *feed-forward* law

$$\tau = \tau_{model} + \ddot{\theta}_d + [K_p](\theta_d - \theta) + [K_v](\dot{\theta}_d - \dot{\theta})$$

- Model parameters required for $\theta_{id}^*$ from CAD model of robot.
- Computed $\theta_{id}^*$ *instead* of $\theta_{id}$ used as reference input.
- Above approach *does not* change any electronics or hardware!
Non-linear control of multi-link serial manipulator

Experimental results (Contd.)

- Similar idea used to modify existing controller to a model-based control scheme – Modify desired $\theta_{id}$ with

$$\theta_{id}^* = \frac{V_{mdl}}{K_{pi}} + \theta_{id} + \frac{1}{K_{pi}} \ddot{\theta}_{id} + \frac{K_v}{K_{pi}} \dot{\theta}_{id}, \quad i = 1, \ldots, 5$$

where $V_{mdl}$, corresponding to $\tau_{mdl}$ computed from

$$\tau_{mdl} = [M(\theta_d)]\ddot{\theta}_d + C(\theta_d, \dot{\theta}_d) + G(\theta_d)$$

with available motor characteristics chart.

- Above control law is analogous to feed-forward law

$$\tau = \tau_{model} + \ddot{\theta}_d + [K_p](\theta_d - \theta) + [K_v](\dot{\theta}_d - \dot{\theta})$$

- Model parameters required for $\theta_{id}^*$ from CAD model of robot.
- Computed $\theta_{id}^*$ instead of $\theta_{id}$ used as reference input.
- Above approach does not change any electronics or hardware!
Non-linear control of multi-link serial manipulator

Experimental results (Contd.)

- Desired trajectory – traverse from \((0°, 0°, -90°, 180°, 0°)\) to \((30°, 40°, -60°, 180°, 0°)\) and back
- Total time 4 seconds – going 2 seconds and coming back 2 seconds.
- Initial 2 seconds against gravity and final two seconds aided by gravity.
- Smooth cubic trajectories generated (see Lecture 1) with zero initial and final velocity.
- Sampling time is 5 ms or a set-points generated at a frequency of 200 Hz.
- Trajectory faster than typical usage for the robot.
Non-linear control of multi-link serial manipulator

Experimental results (Contd.)

- Desired trajectory – traverse from \((0^\circ, 0^\circ, -90^\circ, 180^\circ, 0^\circ)\) to \((30^\circ, 40^\circ, -60^\circ, 180^\circ, 0^\circ)\) and back
- Total time 4 seconds – going 2 seconds and coming back 2 seconds.
- Initial 2 seconds against gravity and final two seconds aided by gravity.
- Smooth cubic trajectories generated (see Lecture 1) with zero initial and final velocity.
- Sampling time is 5 ms or a set-points generated at a frequency of 200 Hz.
- Trajectory faster than typical usage for the robot.
Non-linear control of multi-link serial manipulator

Experimental results (Contd.)

- Desired trajectory – traverse from \((0^\circ, 0^\circ, -90^\circ, 180^\circ, 0^\circ)\) to \((30^\circ, 40^\circ, -60^\circ, 180^\circ, 0^\circ)\) and back
- Total time 4 seconds – going 2 seconds and coming back 2 seconds.
- Initial 2 seconds against gravity and final two seconds aided by gravity.
- Smooth cubic trajectories generated (see Lecture 1) with zero initial and final velocity.
- Sampling time is 5 ms or a set-points generated at a frequency of 200 Hz.
- Trajectory faster than typical usage for the robot.
Desired trajectory – traverse from \((0^\circ, 0^\circ, -90^\circ, 180^\circ, 0^\circ)\) to \((30^\circ, 40^\circ, -60^\circ, 180^\circ, 0^\circ)\) and back

- Total time 4 seconds – going 2 seconds and coming back 2 seconds.
- Initial 2 seconds against gravity and final two seconds aided by gravity.
- Smooth cubic trajectories generated (see Lecture 1) with zero initial and final velocity.
- Sampling time is 5 ms or a set-points generated at a frequency of 200 Hz.
- Trajectory faster than typical usage for the robot.
NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR

Experimental results (Contd.)

- Desired trajectory – traverse from \((0^\circ, 0^\circ, -90^\circ, 180^\circ, 0^\circ)\) to \((30^\circ, 40^\circ, -60^\circ, 180^\circ, 0^\circ)\) and back
- Total time 4 seconds – going 2 seconds and coming back 2 seconds.
- Initial 2 seconds against gravity and final two seconds aided by gravity.
- Smooth cubic trajectories generated (see Lecture 1) with zero initial and final velocity.
- Sampling time is 5 ms or a set-points generated at a frequency of 200 Hz.
- Trajectory faster than typical usage for the robot.
Desired trajectory – traverse from \((0^\circ, 0^\circ, -90^\circ, 180^\circ, 0^\circ)\) to \((30^\circ, 40^\circ, -60^\circ, 180^\circ, 0^\circ)\) and back.

Total time 4 seconds – going 2 seconds and coming back 2 seconds.

Initial 2 seconds against gravity and final two seconds aided by gravity.

Smooth cubic trajectories generated (see Lecture 1) with zero initial and final velocity.

Sampling time is 5 ms or a set-points generated at a frequency of 200 Hz.

Trajectory faster than typical usage for the robot.
Non-linear control of multi-link serial manipulator

Experimental results (Contd.)

- Solid line is $\theta_{1d}$, Dotted line is $\theta_1$ using PD control.
- Dashed line is achieved trajectory of joint 1 using model-based control.

**Figure 18:** Controller performance in following the desired trajectory of joint 1

**Figure 19:** Comparison of errors at joint 1
**NON-LINEAR CONTROL OF MULTI-LINK SERIAL MANIPULATOR**

![Graphs showing comparison of errors at joint 2 and joint 3.](image)

**Figure 20:** Comparison of errors at joint 2

- Maximum \( \theta_1 \) error reduce from \(-5^\circ\) to \(2^\circ\).
- \( \theta_2 \) error also reduces for model-based, not much difference in \( \theta_3 \).
- In joint 4 and 5 (not shown), there is almost no difference!
- Joints 4 and 5 “see” less inertial, centripetal/Coriolis effects!

**Figure 21:** Comparison of errors at joint 3
OUTLINE

1 CONTENTS

2 LECTURE 1
   • Motion planning

3 LECTURE 2
   • Control of a single link

4 LECTURE 3
   • Control of a multi-link serial manipulator

5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators

6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators

7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
Control of Constrained and Parallel Manipulator

Overview

- Till now – control of serial manipulator *without* any constraint on joint trajectory $q(t)$.
- End-effector of a serial manipulator tracing a desired path *while maintaining* contact with a surface.
- Parallel manipulators – passive and active variables related by loop-closure equations.
- Joint space and Cartesian space approaches.
- Leads to *force* and *hybrid* position/force control – End-effector of a serial manipulator tracing a path on a surface and *applying* a force.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Overview

- Till now – control of serial manipulator *without* any constraint on joint trajectory $q(t)$.
- End-effector of a serial manipulator tracing a desired path *while maintaining* contact with a surface.
- Parallel manipulators – passive and active variables related by loop-closure equations.
- Joint space and Cartesian space approaches.
- Leads to *force* and *hybrid* position/force control – End-effector of a serial manipulator tracing a path on a surface and *applying* a force.
Till now – control of serial manipulator *without* any constraint on joint trajectory $q(t)$.

End-effector of a serial manipulator tracing a desired path *while maintaining* contact with a surface.

Parallel manipulators – passive and active variables related by loop-closure equations.

Joint space and Cartesian space approaches.

Leads to *force* and *hybrid* position/force control – End-effector of a serial manipulator tracing a path on a surface and *applying* a force.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Overview

- Till now – control of serial manipulator *without* any constraint on joint trajectory $q(t)$.
- End-effector of a serial manipulator tracing a desired path *while maintaining* contact with a surface.
- Parallel manipulators – passive and active variables related by loop-closure equations.
- Joint space and Cartesian space approaches.
- Leads to *force* and *hybrid* position/force control – End-effector of a serial manipulator tracing a path on a surface and *applying* a force.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

OVERVIEW

- Till now – control of serial manipulator *without* any constraint on joint trajectory $q(t)$.
- End-effector of a serial manipulator tracing a desired path *while maintaining* contact with a surface.
- Parallel manipulators – passive and active variables related by loop-closure equations.
- Joint space and Cartesian space approaches.
- Leads to *force* and *hybrid* position/force control – End-effector of a serial manipulator tracing a path on a surface and *applying* a force.
**CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR**

**Example of constrained motion**

- Tip of planar 2R manipulator to keep in contact with the curve \( f(x, y) = 0 \).
- In joint space

\[
F(\theta_1, \theta_2) = f(l_1 c_1 + l_2 c_{12}, l_1 s_1 + l_2 s_{12}) = 0
\]

Since \( x = l_1 c_1 + l_2 c_{12} \) and \( y = l_1 s_1 + l_2 s_{12} \)

- See direct kinematic equations for the planar 2R manipulator (See Module 3, Lecture 1).

---

**Figure 22:** Constrained motion of a 2R planar manipulator

[Ashitava Ghosal (IISc)](https://www.nptel.ac.in/courses/106115105/) | [Robotics: Advanced Concepts & Analysis](https://www.nptel.ac.in/courses/106115105/) | NPTEL, 2010 | 74 / 129
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- From \( f(x, y) = 0 \) obtain \( x = f_1(\phi) \) and \( y = f_2(\phi) \) → parametric equation of the curve \( f(x, y) = 0 \) in terms of parameter \( \phi \).
- Obtain from the parametric form
  \[
  \theta_1 = h_1(\phi), \quad \theta_2 = h_2(\phi), \quad \text{or} \quad \Theta = h(\phi), \quad \Theta = (\theta_1, \theta_2)^T
  \]
- Inverse kinematics of the planar 2R manipulator\(^4\).
- If \( f(x, y) = 0 \) is a simple curve such as circle, then possible to use direct kinematics of a parallel manipulator/mecanism.
- For a circle centered at \((l_0, 0)\) and radius \(l_3\), parametric equations (from the equations of a four-bar) are
  \[
  x = l_1 c_1 + l_2 c_{12} = l_0 + l_3 \cos \phi, \quad y = l_1 s_1 + l_2 s_{12} = l_3 \sin \phi
  \]

\(^4\)For other manipulators, it may not be easy to obtain analytical expressions.
CONTROL OF CONSTRAINED AND PARALLEL
MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- From \( f(x, y) = 0 \) obtain \( x = f_1(\phi) \) and \( y = f_2(\phi) \) → parametric
  equation of the curve \( f(x, y) = 0 \) in terms of parameter \( \phi \).
- Obtain from the parametric form
  \[
  \theta_1 = h_1(\phi), \quad \theta_2 = h_2(\phi), \quad \text{or} \quad \Theta = h(\phi), \quad \Theta = (\theta_1, \theta_2)^T
  \]
- **Inverse** kinematics of the planar 2R manipulator\(^4\).
- If \( f(x, y) = 0 \) is a simple curve such as circle, then possible to use
  direct kinematics of a parallel manipulator/mechanism.
- For a circle centered at \((l_0, 0)\) and radius \(l_3\), parametric equations
  (from the equations of a four-bar) are
  \[
  x = l_1 c_1 + l_2 c_{12} = l_0 + l_3 \cos \phi, \quad y = l_1 s_1 + l_2 s_{12} = l_3 \sin \phi
  \]

\(^4\)For other manipulators, it may not be easy to obtain analytical expressions.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Analysis of constrained motion of planar 2R manipulator

- From \( f(x, y) = 0 \) obtain \( x = f_1(\phi) \) and \( y = f_2(\phi) \) → parametric equation of the curve \( f(x, y) = 0 \) in terms of parameter \( \phi \).
- Obtain from the parametric form
  \[
  \theta_1 = h_1(\phi), \quad \theta_2 = h_2(\phi), \quad \text{or} \quad \Theta = h(\phi), \quad \Theta = (\theta_1, \theta_2)^T
  \]
- *Inverse* kinematics of the planar 2R manipulator\(^4\).
- If \( f(x, y) = 0 \) is a simple curve such as circle, then possible to use direct kinematics of a parallel manipulator/mechanism.
- For a circle centered at \((l_0, 0)\) and radius \(l_3\), parametric equations (from the equations of a four-bar) are
  \[
  x = l_1 c_1 + l_2 c_{12} = l_0 + l_3 \cos \phi, \quad y = l_1 s_1 + l_2 s_{12} = l_3 \sin \phi
  \]

\(^4\)For other manipulators, it may not be easy to obtain analytical expressions.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- From $f(x, y) = 0$ obtain $x = f_1(\phi)$ and $y = f_2(\phi) \rightarrow$ parametric equation of the curve $f(x, y) = 0$ in terms of parameter $\phi$.
- Obtain from the parametric form

$$\theta_1 = h_1(\phi), \quad \theta_2 = h_2(\phi), \quad \text{or} \quad \Theta = h(\phi), \ \Theta = (\theta_1, \theta_2)^T$$

- Inverse kinematics of the planar 2R manipulator\(^4\).
- If $f(x, y) = 0$ is a simple curve such as circle, then possible to use direct kinematics of a parallel manipulator/mechanism.

- For a circle centered at $(l_0, 0)$ and radius $l_3$, parametric equations (from the equations of a four-bar) are

$$x = l_1 c_1 + l_2 c_{12} = l_0 + l_3 \cos \phi, \quad y = l_1 s_1 + l_2 s_{12} = l_3 \sin \phi$$

\(^4\)For other manipulators, it may not be easy to obtain analytical expressions.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- From \( f(x, y) = 0 \) obtain \( x = f_1(\phi) \) and \( y = f_2(\phi) \) → parametric equation of the curve \( f(x, y) = 0 \) in terms of parameter \( \phi \).
- Obtain from the parametric form
  \[
  \theta_1 = h_1(\phi), \quad \theta_2 = h_2(\phi), \quad \text{or} \quad \Theta = h(\phi), \quad \Theta = (\theta_1, \theta_2)^T
  \]
- *Inverse* kinematics of the planar 2R manipulator\(^4\).
- If \( f(x, y) = 0 \) is a simple curve such as circle, then possible to use *direct kinematics* of a parallel manipulator/mechanism.
- For a circle centered at \((l_0, 0)\) and radius \( l_3 \), parametric equations (from the equations of a four-bar) are
  \[
  x = l_1 c_1 + l_2 c_{12} = l_0 + l_3 \cos \phi, \quad y = l_1 s_1 + l_2 s_{12} = l_3 \sin \phi
  \]

\(^4\)For other manipulators, it may not be easy to obtain analytical expressions.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- From $\theta_1 = h_1(\phi)$, $\theta_2 = h_2(\phi)$, obtain

$$
\dot{\theta}_i = \frac{\partial h_i}{\partial \phi} \dot{\phi}, \quad i = 1, 2
$$

$$
\ddot{\theta}_i = \frac{\partial h_i}{\partial \phi} \ddot{\phi} + \left( \frac{\partial^2 h_i}{\partial \phi^2} \right) \dot{\phi} \quad i = 1, 2
$$

- Substitute $\theta_i$, $\dot{\theta}_i$ and $\ddot{\theta}_i \ (i = 1, 2)$ in the equations of motion of a planar 2R manipulator (see Module 6, Lecture 2) to get

$$
[M(\Theta)][J_h]\ddot{\phi} + (C(\Theta, \dot{\Theta}) + [M(\Theta)][J_h] \dot{\phi}) + G(\Theta) = \tau
$$

$[J_h]$ denotes the Jacobian of the transformation $\Theta = h(\phi)$ and $[\dot{J}_h]$ is its time derivative.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- From \( \theta_1 = h_1(\phi) \), \( \theta_2 = h_2(\phi) \), obtain
  \[
  \dot{\theta}_i = \frac{\partial h_i}{\partial \phi} \dot{\phi}, \quad i = 1, 2
  \]
  \[
  \ddot{\theta}_i = \frac{\partial h_i}{\partial \phi} \ddot{\phi} + (\frac{\partial^2 h_i}{\partial \phi^2} \dot{\phi}) \dot{\phi} \quad i = 1, 2
  \]

- Substitute \( \theta_i \), \( \dot{\theta}_i \) and \( \ddot{\theta}_i \) \((i = 1, 2)\) in the equations of motion of a planar 2R manipulator (see Module 6, Lecture 2) to get
  \[
  [M(\Theta)][J_h]\ddot{\phi} + (C(\Theta, \dot{\Theta}) + [M(\Theta)][\dot{J}_h]\dot{\phi}) + G(\Theta) = \tau
  \]

[\( J_h \)] denotes the Jacobian of the transformation \( \Theta = h(\phi) \) and [\( \dot{J}_h \)] is its time derivative.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Analysis of constrained motion of planar 2R manipulator

- Pre-multiply the left- and the right-hand side by $[J_h]^T$ to get

$$\ddot{\bar{M}}(\phi)\dot{\phi} + \bar{C}(\phi, \dot{\phi}) + \bar{G}(\phi) = [J_h]^T \tau$$

where

$$\bar{M}(\phi) = [J_h]^T [M(h(\phi))] [J_h]$$
$$\bar{C}(\phi, \dot{\phi}) = C(h(\phi), [J_h] \dot{\phi}) + [M(h(\phi))] [J_h] \dot{\phi}$$
$$\bar{G}(\phi) = G(h(\phi))$$

- Above represents a unconstrained one DOF system which satisfies $f(x, y) = 0$.
- The single ODE can be used to “design” model-based control schemes.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- Pre-multiply the left- and the right-hand side by $[J_h]^T$ to get

$$\ddot{\bar{M}}(\phi)\ddot{\phi} + \bar{C}(\phi, \dot{\phi}) + \bar{G}(\phi) = [J_h]^T \tau$$

where

$$\begin{align*}
\ddot{\bar{M}}(\phi) &= [J_h]^T[M(h(\phi))][J_h] \\
\bar{C}(\phi, \dot{\phi}) &= C(h(\phi), [J_h]\dot{\phi}) + [M(h(\phi))][J_h]\dot{\phi} \\
\bar{G}(\phi) &= G(h(\phi))
\end{align*}$$

- Above represents a unconstrained one DOF system which satisfies $f(x, y) = 0$.
- The single ODE can be used to “design” model-based control schemes.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Analysis of constrained motion of planar 2R manipulator

- Pre-multiply the left- and the right-hand side by $[J_h]^T$ to get

$$\ddot{M}(\phi)\ddot{\phi} + \bar{C}(\phi, \dot{\phi}) + \bar{G}(\phi) = [J_h]^T \tau$$

where

$$\ddot{M}(\phi) = [J_h]^T [M(h(\phi))][J_h]$$
$$\bar{C}(\phi, \dot{\phi}) = C(h(\phi), [J_h]\dot{\phi}) + [M(h(\phi))][J_h]\dot{\phi}$$
$$\bar{G}(\phi) = G(h(\phi))$$

- Above represents a *unconstrained* one DOF system which satisfies $f(x, y) = 0$.
- The single ODE can be used to “design” model-based control schemes.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- \([J_h]^T\) removes all information about the force normal to curve \(\rightarrow\) 
  Single ODE not useful to “design” control scheme for applying force.
- The normal is along gradient \(\nabla f(x, y)\).
- Force normal to \(f(x, y) = 0\) is of the form \(\tau_n = \lambda \nabla F(\theta_1, \theta_2)\) where \(\lambda(t)\) is the desired force.
- \(\tau_n\) does not do any work while tracing \(f(x, y) = 0\)
  \[
  \tau_n \cdot \dot{\theta} = \lambda \left( \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_1} \dot{\theta}_1 + \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_2} \dot{\theta}_2 \right)
  \]
  \[
  = \lambda \frac{d}{dt} (F(\theta_1, \theta_2)) = 0
  \]
- Combined joint torque
  \[
  \tau = \lambda(t) \nabla F(\theta_1, \theta_2) + \tau_{\phi}
  \]
  \(\tau_{\phi}\) can be utilised to trace a desired path without violating the constraint \(f(x, y) = 0\).
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- \([J_h]^T\) removes all information about the force normal to curve \(\rightarrow\)
  Single ODE not useful to “design” control scheme for applying force.
- The normal is along gradient \(\nabla f(x, y)\).
- Force normal to \(f(x, y) = 0\) is of the form \(\tau_n = \lambda \nabla F(\theta_1, \theta_2)\) where \(\lambda(t)\) is the desired force.
- \(\tau_n\) does not do any work while tracing \(f(x, y) = 0\)
  \[
  \tau_n \cdot \dot{\Theta} = \lambda \left( \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_1} \dot{\theta}_1 + \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_2} \dot{\theta}_2 \right)
  \]
  \[
  = \lambda \frac{d}{dt} (F(\theta_1, \theta_2)) = 0
  \]
- Combined joint torque
  \[
  \tau = \lambda(t)\nabla F(\theta_1, \theta_2) + \tau_{\phi}
  \]
  \(\tau_{\phi}\) can be utilised to trace a desired path without violating the constraint \(f(x, y) = 0\).
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Analysis of constrained motion of planar 2R manipulator

- $[J_h]^T$ removes all information about the force normal to curve $\rightarrow$
  Single ODE not useful to "design" control scheme for applying force.
- The normal is along gradient $\nabla f(x, y)$.
- Force normal to $f(x, y) = 0$ is of the form $\tau_n = \lambda \nabla F(\theta_1, \theta_2)$ where $\lambda(t)$ is the desired force.
- $\tau_n$ does not do any work while tracing $f(x, y) = 0$
  
  $$\tau_n \cdot \dot{\Theta} = \lambda \left( \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_1} \dot{\theta}_1 + \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_2} \dot{\theta}_2 \right)$$
  
  $$= \lambda \frac{d}{dt} (F(\theta_1, \theta_2)) = 0$$

- Combined joint torque
  
  $$\tau = \lambda(t) \nabla F(\theta_1, \theta_2) + \tau_\phi$$

$\tau_\phi$ can be utilised to trace a desired path without violating the constraint $f(x, y) = 0$. 
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- $[J_h]^T$ removes all information about the force normal to curve $\rightarrow$
  Single ODE not useful to “design” control scheme for applying force.
- The normal is along gradient $\nabla f(x, y)$.
- Force normal to $f(x, y) = 0$ is of the form $\tau_n = \lambda \nabla F(\theta_1, \theta_2)$ where $\lambda(t)$ is the desired force.
- $\tau_n$ does not do any work while tracing $f(x, y) = 0$
  \[ \tau_n \cdot \dot{\Theta} = \lambda \left( \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_1} \dot{\theta}_1 + \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_2} \dot{\theta}_2 \right) \]
  \[ = \lambda \frac{d}{dt} (F(\theta_1, \theta_2)) = 0 \]

- Combined joint torque
  \[ \tau = \lambda(t) \nabla F(\theta_1, \theta_2) + \tau_\phi \]
  $\tau_\phi$ can be utilised to trace a desired path without violating the constraint $f(x, y) = 0$. 
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- \([J_h]^T\) removes all information about the force normal to curve \(\rightarrow\)
  Single ODE not useful to “design” control scheme for applying force.
- The normal is along gradient \(\nabla f(x, y)\).
- Force normal to \(f(x, y) = 0\) is of the form \(\tau_n = \lambda \nabla F(\theta_1, \theta_2)\) where \(\lambda(t)\) is the desired force.
- \(\tau_n\) does not do any work while tracing \(f(x, y) = 0\)

\[
\tau_n \cdot \dot{\Theta} = \lambda \left( \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_1} \dot{\theta_1} + \frac{\partial F(\theta_1, \theta_2)}{\partial \theta_2} \dot{\theta_2} \right)
\]

\[
= \lambda \frac{d}{dt} (F(\theta_1, \theta_2)) = 0
\]

- Combined joint torque

\[
\tau = \lambda(t)\nabla F(\theta_1, \theta_2) + \tau_\phi
\]

\(\tau_\phi\) can be utilised to trace a desired path without violating the constraint \(f(x, y) = 0\).

ASHITAVA GHOSAL (IISc)  ROBOTICS: ADVANCED CONCEPTS & ANALYSIS  NPTEL, 2010  78 / 129
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Analysis of constrained motion of planar 2R manipulator

- Using concept of computed torque control
  \[
  \tau_\phi = [\alpha]_\phi \tau'_\phi + \beta_\phi
  \]

  with

  \[
  [\alpha]_\phi = [M(\Theta)][J_h] \\
  \beta_\phi = (C(\Theta, \dot{\Theta}) + [M(\Theta)][J_h] \dot{\phi}) + G(\Theta) \\
  \tau'_\phi = \ddot{\phi}_d + K_v(\dot{\phi}_d - \dot{\phi}) + K_p(\phi_d - \phi)
  \]

- Choose controller gains $K_p$ and $K_v$ to meet performance requirement.
- Manipulator always keeps in contact with $f(x, y) = 0$.
- The terms $\lambda(t)\nabla F(\theta_1, \theta_2)$ and $\tau_\phi$ do not affect each other!
- Fairly complicated – not practical for 6 DOF manipulator → Cartesian control schemes much better!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- Using concept of computed torque control

\[ \tau_\phi = [\alpha]_\phi \tau'_\phi + \beta_\phi \]

with

\[ [\alpha]_\phi = [M(\Theta)] [J_h] \]
\[ \beta_\phi = (C(\Theta, \dot{\Theta}) + [M(\Theta)] [J_h] \dot{\phi}) + G(\Theta) \]
\[ \tau'_\phi = \ddot{\phi}_d + K_v (\dot{\phi}_d - \dot{\phi}) + K_p (\phi_d - \phi) \]

- Choose controller gains \( K_p \) and \( K_v \) to meet performance requirement.
- Manipulator always keeps in contact with \( f(x, y) = 0 \).
- The terms \( \lambda(t) \nabla F(\theta_1, \theta_2) \) and \( \tau_\phi \) do not affect each other!
- Fairly complicated – not practical for 6 DOF manipulator \( \rightarrow \) Cartesian control schemes much better!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- Using concept of computed torque control

\[ \tau_\phi = [\alpha]_\phi \tau'_\phi + \beta_\phi \]

with

\[ [\alpha]_\phi = [M(\Theta)][J_h] \]
\[ \beta_\phi = (C(\Theta, \dot{\Theta}) + [M(\Theta)][J_h]\dot{\phi}) + G(\Theta) \]
\[ \tau'_\phi = \ddot{\phi}_d + K_v (\dot{\phi}_d - \dot{\phi}) + K_p (\phi_d - \phi) \]

- Choose controller gains \( K_p \) and \( K_v \) to meet performance requirement.
- Manipulator always keeps in contact with \( f(x, y) = 0 \).
- The terms \( \lambda(t)\nabla F(\theta_1, \theta_2) \) and \( \tau_\phi \) do not affect each other!
- Fairly complicated – not practical for 6 DOF manipulator \( \rightarrow \) Cartesian control schemes much better!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Analysis of constrained motion of planar 2R manipulator

- Using concept of computed torque control

\[ \tau_\phi = [\alpha]_\phi \tau'_\phi + \beta_\phi \]

with

\[ [\alpha]_\phi = [M(\Theta)][J_h] \]
\[ \beta_\phi = (C(\Theta, \dot{\Theta}) + [M(\Theta)][J_h]\dot{\phi}) + G(\Theta) \]
\[ \tau'_\phi = \ddot{\phi}_d + K_v(\dot{\phi}_d - \dot{\phi}) + K_p(\phi_d - \phi) \]

- Choose controller gains \( K_p \) and \( K_v \) to meet performance requirement.
- Manipulator always keeps in contact with \( f(x, y) = 0 \).
- The terms \( \lambda(t)\nabla F(\theta_1, \theta_2) \) and \( \tau_\phi \) do not affect each other!
- Fairly complicated – not practical for 6 DOF manipulator → Cartesian control schemes much better!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

ANALYSIS OF CONSTRAINED MOTION OF PLANAR 2R MANIPULATOR

- Using concept of computed torque control

\[ \tau_\phi = [\alpha]_\phi \tau'_\phi + \beta_\phi \]

with

\[ [\alpha]_\phi = [M(\Theta)][J_h] \]
\[ \beta_\phi = (C(\Theta, \dot{\Theta}) + [M(\Theta)][J_h] \dot{\phi}) + G(\Theta) \]
\[ \tau'_\phi = \ddot{\phi}_d + K_v(\dot{\phi}_d - \dot{\phi}) + K_p(\phi_d - \phi) \]

- Choose controller gains \( K_p \) and \( K_v \) to meet performance requirement.
- Manipulator always keeps in contact with \( f(x, y) = 0 \).
- The terms \( \lambda(t) \nabla F(\theta_1, \theta_2) \) and \( \tau_\phi \) do not affect each other!
- Fairly complicated – not practical for 6 DOF manipulator → Cartesian control schemes much better!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Parallel manipulators

- In parallel manipulator *loop-closure* constraint.
- Equations of motion can be derived using Lagrange multipliers (see Module 6, Lecture 1).

\[
[M(q)]\ddot{q} + [C(q, \dot{q})]\dot{q} + G(q) = \tau + [\Psi(q)]^T \lambda
\]

- \([\Psi(q)]\) and \(\lambda\) are similar to the Jacobian matrix \([J_h]\) and \(\lambda\) for 2R serial manipulators with constraints.
- Key difference – no need to control constraint forces arising out of loop-closure constraints!
In parallel manipulator *loop-closure* constraint.

Equations of motion can be derived using Lagrange multipliers (see Module 6, Lecture 1).

\[
[M(q)]\ddot{q} + [C(q, \dot{q})]\dot{q} + G(q) = \tau + [\Psi(q)]^T \lambda
\]

- \([\Psi(q)]\) and \(\lambda\) are similar to the Jacobian matrix \([J_h]\) and \(\lambda\) for 2R serial manipulators with constraints.
- Key difference – no need to control constraint forces arising out of loop-closure constraints!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

PARALLEL MANIPULATORS

- In parallel manipulator *loop-closure* constraint.
- Equations of motion can be derived using Lagrange multipliers (see Module 6, Lecture 1).

\[
[M(q)]\ddot{q} + [C(q, \dot{q})] \dot{q} + G(q) = \tau + [\Psi(q)]^T \lambda
\]

- \([\Psi(q)]\) and \(\lambda\) are similar to the Jacobian matrix \([J_h]\) and \(\lambda\) for 2R serial manipulators with constraints.
- Key difference – no need to control constraint forces arising out of loop-closure constraints!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

PARALLEL MANIPULATORS

- In parallel manipulator loop-closure constraint.
- Equations of motion can be derived using Lagrange multipliers (see Module 6, Lecture 1).

\[
[M(q)]\ddot{q} + [C(q, \dot{q})]q + G(q) = \tau + [\Psi(q)]^T \lambda
\]

- \([\Psi(q)]\) and \(\lambda\) are similar to the Jacobian matrix \([J_h]\) and \(\lambda\) for 2R serial manipulators with constraints.
- Key difference – no need to control constraint forces arising out of loop-closure constraints!
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Parallel manipulators (Contd.)

- \( \tau \) has non-zero elements only for the \( n \) actuated joints.
- Can directly use the equations obtained after eliminating \( \lambda \) (see Module 6, Lecture 1).

\[
[M] \ddot{q} = f - [\psi]^T ([\psi][M]^{-1}[\psi]^T)^{-1} ([\psi][M]^{-1}f + [\dot{\psi}] \dot{q})
\]

\( f \) denotes \((\tau - [C] \dot{q} - G)\).
- The \( n + m \) equations of motion can be written as

\[
[M] \ddot{q} + B(q, \dot{q}) = [A(q)] \tau
\]

- From control law partitioning

\[
[A(q)] \tau = [\alpha] \tau' + \beta
\]

Choose \([\alpha]\) and \(\beta\) as \([M(q)]\) and \(B(q, \dot{q})\), respectively, for the model based control part.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

PARALLEL MANIPULATORS (Contd.)

- \( \tau \) has non-zero elements only for the \( n \) actuated joints.
- Can directly use the equations obtained after eliminating \( \lambda \) (see Module 6, Lecture 1).

\[
[M] \ddot{q} = f - [\psi]^T ([\psi][M]^{-1}[\psi]^T)^{-1} \{[\psi][M]^{-1}f + [\dot{\psi}] \dot{q}\}
\]

\( f \) denotes \( (\tau - [C]\dot{q} - G) \).
- The \( n + m \) equations of motion can be written as

\[
[M] \ddot{q} + B(q, \dot{q}) = [A(q)] \tau
\]

- From control law partitioning

\[
[A(q)] \tau = [\alpha] \tau' + \beta
\]

Choose \([\alpha]\) and \( \beta \) as \([M(q)]\) and \( B(q, \dot{q})\), respectively, for the model based control part.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Parallel manipulators (Contd.)

- $\tau$ has non-zero elements only for the $n$ actuated joints.
- Can directly use the equations obtained after eliminating $\lambda$ (see Module 6, Lecture 1).

\[
[M]\ddot{q} = f - [\psi]^T([\psi][M]^{-1}[\psi]^T)^{-1}\{[\psi][M]^{-1}f + [\psi]\dot{q}\}
\]

$f$ denotes $(\tau - [C]\dot{q} - G)$.
- The $n + m$ equations of motion can be written as

\[
[M]\ddot{q} + B(q, \dot{q}) = [A(q)]\tau
\]

- From control law partitioning

\[
[A(q)]\tau = [\alpha]\tau' + \beta
\]

Choose $[\alpha]$ and $\beta$ as $[M(q)]$ and $B(q, \dot{q})$, respectively, for the model based control part.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

PARALLEL MANIPULATORS (Contd.)

- $\tau$ has non-zero elements only for the $n$ actuated joints.
- Can directly use the equations obtained after eliminating $\lambda$ (see Module 6, Lecture 1).

$$[M]\ddot{q} = f - [\psi]^T([\psi][M]^{-1}[\psi]^T)^{-1}\{[\psi][M]^{-1}f + [\psi]\dot{q}\}$$

- $f$ denotes $(\tau - [C]\dot{q} - G)$.
- The $n + m$ equations of motion can be written as

$$[M]\ddot{q} + B(q, \dot{q}) = [A(q)]\tau$$

- From control law partitioning

$$[A(q)]\tau = [\alpha]\tau' + \beta$$

Choose $[\alpha]$ and $\beta$ as $[M(q)]$ and $B(q, \dot{q})$, respectively, for the model based control part.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Parallel manipulators (Contd.)

- Choose non-zero elements of $\tau'$ for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$.
- Motion of actuated joints will not violate loop-closure constraints!
- Model-based terms involve *active* and *passive* variables!
- Typically *passive* variables not measured $\rightarrow$ Passive variables must be estimated using direct-kinematics equations.
- Use of direct kinematics for estimating passive joint variables and their rates make model-based control of parallel manipulators much more complex.
- As in serial manipulators cannot avoid issues arising out “lack of knowledge” of parameters.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

Parallel manipulators (Contd.)

- Choose non-zero elements of $\tau'$ for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$.
- Motion of actuated joints will not violate loop-closure constraints!
- Model-based terms involve active and passive variables!
- Typically passive variables not measured $\rightarrow$ Passive variables must be estimated using direct-kinematics equations
- Use of direct kinematics for estimating passive joint variables and their rates make model-based control of parallel manipulators much more complex.
- As in serial manipulators cannot avoid issues arising out “lack of knowledge” of parameters.
Control of Constrained and Parallel Manipulator

Parallel manipulators (Contd.)

- Choose non-zero elements of $\tau'$ for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$.
- Motion of actuated joints will not violate loop-closure constraints!
- Model-based terms involve active and passive variables!
  - Typically passive variables not measured $\rightarrow$ Passive variables must be estimated using direct-kinematics equations.
  - Use of direct kinematics for estimating passive joint variables and their rates make model-based control of parallel manipulators much more complex.
- As in serial manipulators cannot avoid issues arising out “lack of knowledge” of parameters.
Choose non-zero elements of $\tau'$ for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$.  

Motion of actuated joints will not violate loop-closure constraints!  

Model-based terms involve active and passive variables!  

Typically passive variables not measured $\rightarrow$ Passive variables must be estimated using direct-kinematics equations  

Use of direct kinematics for estimating passive joint variables and their rates make model-based control of parallel manipulators much more complex.  

As in serial manipulators cannot avoid issues arising out “lack of knowledge” of parameters.
CONTROL OF CONSTRAINED AND PARALLEL MANIPULATOR

PARALLEL MANIPULATORS (Contd.)

- Choose non-zero elements of $\tau'$ for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$.
- Motion of actuated joints will not violate loop-closure constraints!
- Model-based terms involve active and passive variables!
- Typically passive variables not measured $\rightarrow$ Passive variables must be estimated using direct-kinematics equations
- Use of direct kinematics for estimating passive joint variables and their rates make model-based control of parallel manipulators much more complex.
- As in serial manipulators cannot avoid issues arising out “lack of knowledge” of parameters.
Control of Constrained and Parallel Manipulator

Parallel Manipulators (Contd.)

- Choose non-zero elements of $\tau'$ for PD control with appropriate gain matrices $[K_p]$ and $[K_v]$.
- Motion of actuated joints will not violate loop-closure constraints!
- Model-based terms involve active and passive variables!
- Typically passive variables not measured $\rightarrow$ Passive variables must be estimated using direct-kinematics equations
- Use of direct kinematics for estimating passive joint variables and their rates make model-based control of parallel manipulators much more complex.
- As in serial manipulators cannot avoid issues arising out “lack of knowledge” of parameters.
OUTLINE

1 CONTENTS

2 LECTURE 1
   ● Motion planning

3 LECTURE 2
   ● Control of a single link

4 LECTURE 3
   ● Control of a multi-link serial manipulator

5 LECTURE 4*
   ● Control of constrained and parallel manipulator
   ● Cartesian control of serial manipulators

6 LECTURE 5*
   ● Force control of manipulators
   ● Hybrid position/force control of manipulators

7 LECTURE 6*
   ● Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   ● Problems, References and Suggested Reading
**Cartesian control schemes**

**Overview**

- Very difficult to implement joint space control of serial manipulators with constraint
  - The constraint is *almost always* in terms of end-effector position and/or orientation.
  - More often than not, closed-form expressions for inverse kinematics do not exist!
  - Except for simple curves, not possible to convert to a simple parallel mechanism.

- Need to develop control schemes which use desired trajectories specified in terms of Cartesian/task space variables.
- Scheme should not use inverse kinematics as it is computationally intensive.
- A model-based/feedback linearization type of control scheme is desirable.
**Cartesian Control Schemes**

**Overview**

- Very difficult to implement joint space control of serial manipulators with constraint
  - The constraint is *almost always* in terms of end-effector position and/or orientation.
  - More often than not, closed-form expressions for inverse kinematics do not exist!
  - Except for simple curves, not possible to convert to a simple parallel mechanism.

- Need to develop control schemes which use desired trajectories specified in terms of Cartesian/task space variables.
  - Scheme should not use inverse kinematics as it is computationally intensive.
  - A model-based/feedback linearization type of control scheme is desirable.
Cartesian control schemes

Overview

• Very difficult to implement joint space control of serial manipulators with constraint
  • The constraint is *almost always* in terms of end-effector position and/or orientation.
  • More often than not, closed-form expressions for inverse kinematics do not exist!
  • Except for simple curves, not possible to convert to a simple parallel mechanism.

• Need to develop control schemes which use desired trajectories specified in terms of Cartesian/task space variables.

• Scheme should not use inverse kinematics as it is computationally intensive.

• A model-based/feedback linearization type of control scheme is desirable.
 CARTESIAN CONTROL SCHEMES

OVERVIEW

- Very difficult to implement joint space control of serial manipulators with constraint
  - The constraint is *almost always* in terms of end-effector position and/or orientation.
  - More often than not, closed-form expressions for inverse kinematics do not exist!
  - Except for simple curves, not possible to convert to a simple parallel mechanism.

- Need to develop control schemes which use desired trajectories specified in terms of Cartesian/task space variables.

- Scheme should not use inverse kinematics as it is computationally intensive.

- A model-based/feedback linearization type of control scheme is desirable.
**Cartesian control schemes**

**Cartesian equations of motion**

- Equations of motion in terms of Cartesian/task space variables $\mathcal{X}$ (See Module 6, Lecture 1).

\[
\mathcal{F} = [M_{\mathcal{X}}(q)] \ddot{\mathcal{X}} + C_{\mathcal{X}}(q, \dot{q}) + G_{\mathcal{X}}(q)
\]

where $\mathcal{F}$ is a $6 \times 1$ entity of force & moment acting on the end-effector and

\[
[J(q)]^T \mathcal{F} = \tau
\]

\[
[M_{\mathcal{X}}(q)] = [J(q)]^{-T}[M(q)][J(q)]^{-1}
\]

\[
C_{\mathcal{X}}(q, \dot{q}) = [J(q)]^{-T}(C(q, \dot{q}) - [M(q)][J(q)]^{-1}[J(q)]\dot{q})
\]

\[
G_{\mathcal{X}}(q) = [J(q)]^{-T}G(q)
\]

and where $[J(q)]^{-T}$ denotes the inverse of $[J(q)]^T$.

- Inverse kinematics is not required in the control.
- Inverse Jacobian required to obtain Cartesian mass matrix and other model-based terms → the model-based terms can be obtained symbolically once.
Cartesian control schemes

Cartesian equations of motion

- Equations of motion in terms of Cartesian/task space variables $\mathbf{X}$ (See Module 6, Lecture 1).

$$ \mathbf{F} = [M_{\mathbf{X}}(\mathbf{q})] \ddot{\mathbf{X}} + C_{\mathbf{X}}(\mathbf{q}, \dot{\mathbf{q}}) + G_{\mathbf{X}}(\mathbf{q}) $$

where $\mathbf{F}$ is a $6 \times 1$ entity of force & moment acting on the end-effector and

$$ [J(\mathbf{q})]^T \mathbf{F} = \tau $$

$$ [M_{\mathbf{X}}(\mathbf{q})] = [J(\mathbf{q})]^{-T} [M(\mathbf{q})][J(\mathbf{q})]^{-1} $$

$$ C_{\mathbf{X}}(\mathbf{q}, \dot{\mathbf{q}}) = [J(\mathbf{q})]^{-T} (C(\mathbf{q}, \dot{\mathbf{q}}) - [M(\mathbf{q})][J(\mathbf{q})]^{-1}[J(\mathbf{q})]\dot{\mathbf{q}}) $$

$$ G_{\mathbf{X}}(\mathbf{q}) = [J(\mathbf{q})]^{-T} G(\mathbf{q}) $$

and where $[J(\mathbf{q})]^{-T}$ denotes the inverse of $[J(\mathbf{q})]^T$.

- Inverse kinematics is not required in the control.

- Inverse Jacobian required to obtain Cartesian mass matrix and other model-based terms → the model-based terms can be obtained symbolically once.
Cartesian control schemes

Cartesian equations of motion

- Equations of motion in terms of Cartesian/task space variables $\mathcal{X}$ (See Module 6, Lecture 1).
  \[
  \mathcal{F} = [M_\mathcal{X}(q)] \ddot{\mathcal{X}} + C_\mathcal{X}(q, \dot{q}) + G_\mathcal{X}(q)
  \]
  where $\mathcal{F}$ is a $6 \times 1$ entity of force & moment acting on the end-effector and
  \[
  [J(q)]^T \mathcal{F} = \tau \\
  [M_\mathcal{X}(q)] = [J(q)]^{-T} [M(q)] [J(q)]^{-1} \\
  C_\mathcal{X}(q, \dot{q}) = [J(q)]^{-T} (C(q, \dot{q}) - [M(q)] [J(q)]^{-1} [J(q)] \dot{q}) \\
  G_\mathcal{X}(q) = [J(q)]^{-T} G(q)
  \]
  and where $[J(q)]^{-T}$ denotes the inverse of $[J(q)]^T$.

- Inverse kinematics is not required in the control.
- Inverse Jacobian required to obtain Cartesian mass matrix and other model-based terms → the model-based terms can be obtained symbolically once.
Cartesian Control Schemes

Model-based Cartesian control

- Similar to joint space control scheme, assume a control law of the form

\[ \mathcal{F} = [\alpha \dot{\mathbf{x}}] \mathcal{F}' + \beta \mathbf{x} \]

- Choose \([\alpha \dot{\mathbf{x}}] = [M \dot{\mathbf{x}}(\mathbf{q})]\) and \(\beta \mathbf{x} = \mathbf{C} \dot{\mathbf{x}}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G} \mathbf{x}(\mathbf{q})\).
- To get \(\mathcal{F}' = \ddot{\mathbf{x}} \rightarrow\) Unit mass system with new input \(\mathcal{F}'\)
- Choose

\[ \mathcal{F}' = \ddot{\mathbf{x}}_d(t) + [K_v] \mathbf{x} \dot{e}(t) + [K_p] \mathbf{x} e(t) \]

to get linear, decoupled error equation of the form

\[ \ddot{e}(t) + [K_v] \mathbf{x} \dot{e}(t) + [K_p] \mathbf{x} e(t) = 0 \]

and appropriate choice of \([K_p] \mathbf{x}\) and \([K_v] \mathbf{x}\) will give required performance!

- To obtain required Cartesian actuation force & moment \(\mathcal{F}\), use joint torque as \(\tau = [J(\mathbf{q})]^T \mathcal{F}\)
**Cartesian control schemes**

**Model-based Cartesian control**

- Similar to joint space control scheme, assume a control law of the form
  \[ \mathcal{F} = [\alpha \mathbf{x}] \mathcal{F}' + \beta \mathbf{x} \]

- Choose \([\alpha \mathbf{x}] = [M \mathbf{x}(\mathbf{q})]\) and \(\beta \mathbf{x} = C \mathbf{x}(\mathbf{q}, \dot{\mathbf{q}}) + G \mathbf{x}(\mathbf{q})\).

- To get \(\mathcal{F}' = \ddot{\mathbf{x}}\) \(\rightarrow\) Unit mass system with new input \(\mathcal{F}'\)

- Choose
  \[ \mathcal{F}' = \ddot{\mathbf{x}}_d(t) + [K_v \mathbf{x}] \dot{\mathbf{e}}(t) + [K_p \mathbf{x}] \mathbf{e}(t) \]
  to get linear, decoupled error equation of the form
  \[ \ddot{\mathbf{e}}(t) + [K_v \mathbf{x}] \dot{\mathbf{e}}(t) + [K_p \mathbf{x}] \mathbf{e}(t) = 0 \]
  and appropriate choice of \([K_p \mathbf{x}]\) and \([K_v \mathbf{x}]\) will give required performance!

- To obtain required Cartesian actuation force & moment \(\mathcal{F}\), use joint torque as \(\tau = [J(\mathbf{q})]^T \mathcal{F}\)
**Cartesian control schemes**

**Model-based Cartesian control**

- Similar to joint space control scheme, assume a control law of the form
  \[
  \mathbb{F} = [\alpha \dot{x}] \mathbb{F}' + \beta \dot{x}
  \]

- Choose \([\alpha \dot{x}] = [M \dot{x}(q)]\) and \(\beta \dot{x} = C \dot{x}(q, \dot{q}) + G \dot{x}(q)\).

- To get \(\mathbb{F}' = \ddot{X} \rightarrow \) Unit mass system with new input \(\mathbb{F}'\)

- Choose
  \[
  \mathbb{F}' = \ddot{X}_d(t) + [K_v] \dot{X} \dot{e}(t) + [K_p] \dot{X} e(t)
  \]
  to get linear, decoupled error equation of the form
  \[
  \dot{e}(t) + [K_v] \dot{X} \dot{e}(t) + [K_p] \dot{X} e(t) = 0
  \]
  and appropriate choice of \([K_p] \dot{X}\) and \([K_v] \dot{X}\) will give required performance!

- To obtain required Cartesian actuation force & moment \(\mathbb{F}\), use joint torque as \(\tau = [J(q)]^T \mathbb{F}\)
**Cartesian Control Schemes**

**Model-based Cartesian Control**

- Similar to joint space control scheme, assume a control law of the form
  \[
  \mathcal{F} = [\alpha \dot{x}] \mathcal{F}' + \beta \dot{x}
  \]

- Choose \([\alpha \dot{x}] = [M \ddot{x}(q)]\) and \(\beta \dot{x} = C \ddot{x}(q, \dot{q}) + G \dot{x}(q)\).

- To get \(\mathcal{F}' = \ddot{X} \rightarrow\) Unit mass system with new input \(\mathcal{F}'\)

- Choose
  \[
  \mathcal{F}' = \ddot{X}_d(t) + [K_v] \dot{X} \dot{e}(t) + [K_p] \dot{X} e(t)
  \]
  to get linear, decoupled error equation of the form
  \[
  \dot{e}(t) + [K_v] \dot{X} \dot{e}(t) + [K_p] X e(t) = 0
  \]
  and appropriate choice of \([K_p] \dot{X}\) and \([K_v] \dot{X}\) will give required performance!

- To obtain required Cartesian actuation force & moment \(\mathcal{F}\), use joint torque as \(\tau = [J(q)]^T \mathcal{F}\)
Similar to joint space control scheme, assume a control law of the form

$$\mathbf{F} = [\alpha \mathbf{x}] \mathbf{F}' + \beta \mathbf{x}$$

Choose $$[\alpha \mathbf{x}] = [M \mathbf{x}(\mathbf{q})]$$ and $$\beta \mathbf{x} = \mathbf{C} \mathbf{x}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G} \mathbf{x}(\mathbf{q})$$.

To get $$\mathbf{F}' = \ddot{\mathbf{x}} \rightarrow$$ Unit mass system with new input $$\mathbf{F}'$$

Choose

$$\mathbf{F}' = \ddot{\mathbf{x}}_d(t) + [K_v \mathbf{x}] \dot{\mathbf{e}}(t) + [K_p \mathbf{x}] \mathbf{e}(t)$$

to get linear, decoupled error equation of the form

$$\ddot{\mathbf{e}}(t) + [K_v \mathbf{x}] \dot{\mathbf{e}}(t) + [K_p \mathbf{x}] \mathbf{e}(t) = 0$$

and appropriate choice of $$[K_p \mathbf{x}]$$ and $$[K_v \mathbf{x}]$$ will give required performance!

To obtain required Cartesian actuation force & moment $$\mathbf{F}$$, use joint torque as $$\mathbf{\tau} = [J(\mathbf{q})]^T \mathbf{F}$$
**Cartesian Control Schemes**

**Model-Based Cartesian Control**

![Diagram](image)

**Figure 23:** Cartesian model-based control scheme
**Cartesian control schemes**

**Model-based Cartesian control (Contd.)**

- No inverse kinematics used → Direct kinematics used to estimate $\dot{X}$ and $\ddot{X}$ in figure.
- Vision or other sensors can also be used to measure $X$ and $\dot{X}$.
- Khatib (1986) used the Cartesian controller for *real-time* obstacle avoidance – Synthetic force $F_r$ obtained as

\[ F_r = \sum_{i=1}^{N} F_i \propto K_i / r_i^n \]

$N$ is the number of obstacles and $r_i$ is the distance from the $i^{th}$ obstacle (see figure).
- $F_r$ is repulsive and $K_i$ and $n$ chosen so that it falls off quickly!
- $F$ drives the robot along a desired trajectory, when near obstacle $F_r$ is more dominant → repels robot away from obstacles!
No inverse kinematics used → Direct kinematics used to estimate $X$ and $\dot{X}$ in figure.

Vision or other sensors can also be used to measure $X$ and $\dot{X}$.

Khatib (1986) used the Cartesian controller for real-time obstacle avoidance – Synthetic force $F_r$ obtained as

$$F_r = \sum_{i}^{N} F_i \propto K_i / r_i^n$$

$N$ is the number of obstacles and $r_i$ is the distance from the $i^{th}$ obstacle (see figure).

$F_r$ is repulsive and $K_i$ and $n$ chosen so that it falls off quickly!

$F_r$ drives the robot along a desired trajectory, when near obstacle $F_r$ is more dominant → repels robot away from obstacles!
No inverse kinematics used → Direct kinematics used to estimate $\dot{X}$ and $\ddot{X}$ in figure.

Vision or other sensors can also be used to measure $X$ and $\dot{X}$.

Khatib (1986) used the Cartesian controller for real-time obstacle avoidance – Synthetic force $F_r$ obtained as

$$F_r = \sum_i^{N} F_i \propto K_i / r_i^n$$

$N$ is the number of obstacles and $r_i$ is the distance from the $i^{th}$ obstacle (see figure).

$F_r$ is repulsive and $K_i$ and $n$ chosen so that it falls off quickly!

$F$ drives the robot along a desired trajectory, when near obstacle $F_r$ is more dominant → repels robot away from obstacles!
No inverse kinematics used $\rightarrow$ Direct kinematics used to estimate $\dot{X}$ and $\ddot{X}$ in figure.

Vision or other sensors can also be used to measure $\dot{X}$ and $\ddot{X}$.

Khatib (1986) used the Cartesian controller for *real-time* obstacle avoidance – Synthetic force $\mathcal{F}_r$ obtained as

$$\mathcal{F}_r = \sum_{i=1}^{N} \mathcal{F}_i \propto K_i / r_i^n$$

$N$ is the number of obstacles and $r_i$ is the distance from the $i^{th}$ obstacle (see figure).

$\mathcal{F}_r$ is *repulsive* and $K_i$ and $n$ chosen so that it falls off quickly!

$\mathcal{F}$ drives the robot along a desired trajectory, when near obstacle $\mathcal{F}_r$ is more dominant $\rightarrow$ *repels* robot away from obstacles!
No inverse kinematics used $\rightarrow$ Direct kinematics used to estimate $\dot{X}$ and $\ddot{X}$ in figure.

- Vision or other sensors can also be used to measure $\dot{X}$ and $\ddot{X}$.

Khatib (1986) used the Cartesian controller for real-time obstacle avoidance – Synthetic force $\mathcal{F}_r$ obtained as

$$\mathcal{F}_r = \sum_{i}^{N} \mathcal{F}_i \propto K_i / r_i^n$$

$N$ is the number of obstacles and $r_i$ is the distance from the $i^{th}$ obstacle (see figure).

- $\mathcal{F}_r$ is repulsive and $K_i$ and $n$ chosen so that it falls off quickly!

- $\mathcal{F}$ drives the robot along a desired trajectory, when near obstacle $\mathcal{F}_r$ is more dominant $\rightarrow$ repels robot away from obstacles!
OUTLINE

1 CONTENTS
2 LECTURE 1
   • Motion planning
3 LECTURE 2
   • Control of a single link
4 LECTURE 3
   • Control of a multi-link serial manipulator
5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators
6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators
7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators
8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
FORCE CONTROL OF MANIPULATORS

OVERVIEW

- Manipulator moving in free space → position control.
- Robotic assembly, grinding and manufacturing → Position control not enough → Need to apply desired force/moment on environment!
- Apply force/moment with passive stiffness in end-effector → Plan a trajectory such that it is ‘just inside’ the contacting surface.
- Difficult to apply desired and changing force/moment.
  - Error in position control can result in not touching or excessive interference!
  - Not possible to apply desired force/moment to environment if stiffness of environment is high → Very small strains and displacements difficult to measure.

- Joint space control, similar to constrained motion, not suitable
- Cartesian control strategy easily extended for force control!
FORCE CONTROL OF MANIPULATORS

OVERVIEW

- Manipulator moving in free space → position control.
- Robotic assembly, grinding and manufacturing → Position control not enough → Need to apply desired force/moment on environment!
  - Apply force/moment with passive stiffness in end-effector → Plan a trajectory such that it is ‘just inside’ the contacting surface.
  - Difficult to apply desired and changing force/moment.
    - Error in position control can result in not touching or excessive interference!
    - Not possible to apply desired force/moment to environment if stiffness of environment is high → Very small strains and displacements difficult to measure.
- Joint space control, similar to constrained motion, not suitable
- Cartesian control strategy easily extended for force control!
FORCE CONTROL OF MANIPULATORS

Overview

- Manipulator moving in *free space* → position control.
- Robotic *assembly, grinding* and manufacturing → Position control not enough → Need to apply desired force/moment on environment!
- Apply force/moment with *passive* stiffness in end-effector → Plan a trajectory such that it is ‘just inside’ the contacting surface.
- Difficult to apply *desired and changing* force/moment.
  - Error in position control can result in *not touching* or *excessive interference*!
  - Not possible to *apply* desired force/moment to environment if stiffness of environment is high → Very small strains and displacements difficult to measure.
- Joint space control, similar to constrained motion, not suitable
- Cartesian control strategy easily extended for force control!
FORCE CONTROL OF MANIPULATORS

OVERVIEW

- Manipulator moving in *free space* → position control.
- Robotic *assembly*, *grinding* and manufacturing → Position control not enough → Need to apply desired force/moment on environment!
- Apply force/moment with *passive* stiffness in end-effector → Plan a trajectory such that it is ‘just inside’ the contacting surface.
- Difficult to apply *desired and changing* force/moment.
  - Error in position control can result in *not touching* or *excessive* interference!
  - Not possible to *apply* desired force/moment to environment if stiffness of environment is high → Very small strains and displacements difficult to measure.

- Joint space control, similar to constrained motion, not suitable
- Cartesian control strategy easily extended for force control!
FORCE CONTROL OF MANIPULATORS

OVERVIEW

- Manipulator moving in free space $\rightarrow$ position control.
- Robotic assembly, grinding and manufacturing $\rightarrow$ Position control not enough $\rightarrow$ Need to apply desired force/moment on environment!
- Apply force/moment with passive stiffness in end-effector $\rightarrow$ Plan a trajectory such that it is ‘just inside’ the contacting surface.
- Difficult to apply desired and changing force/moment.
  - Error in position control can result in not touching or excessive interference!
  - Not possible to apply desired force/moment to environment if stiffness of environment is high $\rightarrow$ Very small strains and displacements difficult to measure.
- Joint space control, similar to constrained motion, not suitable
- Cartesian control strategy easily extended for force control!
FORCE CONTROL OF MANIPULATORS

OVERVIEW

- Manipulator moving in *free space* → position control.
- Robotic *assembly, grinding* and manufacturing → Position control not enough → Need to apply desired force/moment on environment!
- Apply force/moment with *passive* stiffness in end-effector → Plan a trajectory such that it is ‘just inside’ the contacting surface.
- Difficult to apply *desired and changing* force/moment.
  - Error in position control can result in *not touching* or *excessive* interference!
  - Not possible to *apply* desired force/moment to environment if stiffness of environment is high → Very small strains and displacements difficult to measure.

- Joint space control, similar to constrained motion, not suitable
- Cartesian control strategy easily extended for force control!
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS

Figure 24: Force control of a mass along one direction

- Applied force from an actuator $f(t)$.
- Disturbance force $f_{\text{dist}}(t)$
- Displacement of mass $x(t)$
- Environment stiffness $K_e$
- Force exerted by environment $f_e(t) = K_e x(t)$
- Aim is to control $f_e(t)$ to a desired value $f_{e_d}(t)$ by $f(t)$. 
The equation of motion of the system is given by

\[ f = m\ddot{x} + K_e x + f_{\text{dist}} \]

Written in terms of \( f_e \),

\[ f = mK_e^{-1}\ddot{f}_e + f_e + f_{\text{dist}} \]

Similar to a second-order ODE for a single-link manipulator.

Can use PD or PID control scheme.

Model-based control scheme is better!
The equation of motion of the system is given by

\[ f = m\ddot{x} + K_e x + f_{dist} \]

Written in terms of \( f_e \),

\[ f = mK_e^{-1}\ddot{f}_e + f_e + f_{dist} \]

Similar to a second-order ODE for a single-link manipulator.

Can use PD or PID control scheme.

Model-based control scheme is better!
The equation of motion of the system is given by

\[ f = m\ddot{x} + K_e x + f_{\text{dist}} \]

Written in terms of \( f_e \),

\[ f = mK_e^{-1}\ddot{f}_e + f_e + f_{\text{dist}} \]

Similar to a second-order ODE for a single-link manipulator.

- Can use PD or PID control scheme.
- Model-based control scheme is better!
The equation of motion of the system is given by

\[ f = m\ddot{x} + K_e x + f_{\text{dist}} \]

Written in terms of \( f_e \),

\[ f = mK_e^{-1}\ddot{f}_e + f_e + f_{\text{dist}} \]

Similar to a second-order ODE for a single-link manipulator.
Can use PD or PID control scheme.
Model-based control scheme is better!
The equation of motion of the system is given by

\[ f = m\ddot{x} + K_e x + f_{\text{dist}} \]

Written in terms of \( f_e \),

\[ f = mK_e^{-1}\ddot{f}_e + f_e + f_{\text{dist}} \]

Similar to a second-order ODE for a single-link manipulator.
- Can use PD or PID control scheme.
- Model-based control scheme is better!
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- Following control law partitioning concept

\[
\begin{align*}
f &= \alpha f' + \beta \\
\alpha &= mK_e^{-1} \\
\beta &= f_e + f_{\text{dist}} \\
f' &= \ddot{f}_e + K_{vf} \dot{e}_f + K_{pf} e_f
\end{align*}
\]

force error is \( e_f = f_{ed} - f_e \) & \( f_e \) (measured) force acting on the environment.

- Closed-loop force error equation is

\[
\ddot{e}_f + K_{vf} \dot{e}_f + K_{pf} e_f = 0
\]

- \( K_{vf}, K_{pf} \) – derivative and proportional gains – set for required performance.
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- Following control law partitioning concept

\[
\begin{align*}
    f &= \alpha f' + \beta \\
    \alpha &= mK_e^{-1} \\
    \beta &= f_e + f_{dist} \\
    f' &= \ddot{f}_e + K_v \dot{f_e} + K_p f_e
\end{align*}
\]

force error is \( e_f = f_{ed} - f_e \) & \( f_e \) (measured) force acting on the environment.

- Closed-loop force error equation is

\[
\ddot{e}_f + K_v \dot{e}_f + K_p e_f = 0
\]

- \( K_v, K_p \) – derivative and proportional gains – set for required performance.
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- Following control law partitioning concept

\[ f = \alpha f' + \beta \]
\[ \alpha = mK_e^{-1} \]
\[ \beta = f_e + f_{\text{dist}} \]
\[ f' = \ddot{f}_e + K_{vf} \dot{e}_f + K_{pf} e_f \]

force error is \( e_f = f_{ed} - f_e \) & \( f_e \) (measured) force acting on the environment.

- Closed-loop force error equation is

\[ \ddot{e}_f + K_{vf} \dot{e}_f + K_{pf} e_f = 0 \]

- \( K_{vf}, K_{pf} \) – derivative and proportional gains – set for required performance.
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- No knowledge of $f_{\text{dist}} \rightarrow$ cannot use in model-based term!
- Set $\beta = f_{\text{ed}} \rightarrow$ Steady-state error not zero!

$$e_f = \frac{f_{\text{dist}}}{1 + mK_e^{-1}}$$

- Since $K_e$ is typically large $\rightarrow$ $e_f \approx f_{\text{dist}}$ – best possible!
- $\dot{f}_{\text{ed}}$ and $\ddot{f}_{\text{ed}}$ not specified – no physical sense in derivative of desired force!
- $f_e$ measured but $\dot{f}_e$ very difficult to measure $\rightarrow$ $\dot{f}_e = K_e \dot{x}$.
- Control law with above constraints

$$f = m[K_{p_f}K_e^{-1}e_f - K_{v_f}\dot{x}] + f_{\text{ed}}$$
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- No knowledge of $f_{\text{dist}} \rightarrow$ cannot use in model-based term!
- Set $\beta = f_{e_d} \rightarrow$ Steady-state error not zero!

\[
e_f = \frac{f_{\text{dist}}}{1 + mK_e^{-1}}
\]

- Since $K_e$ is typically large $\rightarrow e_f \approx f_{\text{dist}}$ – best possible!
- $f_{e_d}$ and $f_{e_d}$ not specified – no physical sense in derivative of desired force!

- $f_e$ measured but $\dot{f}_e$ very difficult to measure $\rightarrow \dot{f}_e = K_e \dot{x}$.
- Control law with above constraints

\[
f = m[K_p f_{e} K_e^{-1} e_f - K_v f_{e_d}] + f_{e_d}
\]
FORCE CONTROL OF MANIPULATORS
FORCE CONTROL OF A SINGLE MASS (Contd.)

- No knowledge of $f_{\text{dist}} \rightarrow$ cannot use in model-based term!
- Set $\beta = f_{ed} \rightarrow$ Steady-state error not zero!

$$e_f = \frac{f_{\text{dist}}}{1 + mK_e^{-1}}$$

- Since $K_e$ is typically large $\rightarrow e_f \approx f_{\text{dist}}$ – best possible!
- $f_{ed}$ and $f_{ed}''$ not specified – no physical sense in derivative of desired force!
- $f_e$ measured but $\dot{f}_e$ very difficult to measure $\rightarrow \dot{f}_e = K_e \dot{x}$.
- Control law with above constraints

$$f = m[K_p f K_e^{-1} e_f - K_v \dot{x}] + f_{ed}$$
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- No knowledge of \( f_{\text{dist}} \) → cannot use in model-based term!
- Set \( \beta = f_{e_d} \) → Steady-state error not zero!

\[
e_f = \frac{f_{\text{dist}}}{1 + mK_e^{-1}}
\]

- Since \( K_e \) is typically large → \( e_f \approx f_{\text{dist}} \) – best possible!
- \( f_{e_d} \) and \( f_{e_d} \) not specified – no physical sense in derivative of desired force!
- \( f_e \) measured but \( \dot{f}_e \) very difficult to measure → \( \dot{f}_e = K_e \dot{x} \).
- Control law with above constraints

\[
f = m[K_p f K_e^{-1} e_f - K_v \dot{x}] + f_{e_d}
\]
No knowledge of $f_{\text{dist}} \rightarrow$ cannot use in model-based term!

Set $\beta = f_{ed} \rightarrow$ Steady-state error not zero!

$$e_f = \frac{f_{\text{dist}}}{1 + mK_e^{-1}}$$

Since $K_e$ is typically large $\rightarrow e_f \simeq f_{\text{dist}}$ – best possible!

$f_{ed}$ and $f_{ed}$ not specified – no physical sense in derivative of desired force!

$f_e$ measured but $\dot{f}_e$ very difficult to measure $\rightarrow \dot{f}_e = K_e \dot{x}$.

Control law with above constraints

$$f = m[K_p f_{ed} K_e^{-1} e_f - K_v f \dot{x}] + f_{ed}$$
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL OF A SINGLE MASS (Contd.)

- No knowledge of $f_{\text{dist}} \rightarrow$ cannot use in model-based term!
- Set $\beta = f_{e_d} \rightarrow$ Steady-state error not zero!

$$e_f = \frac{f_{\text{dist}}}{1 + mK_e^{-1}}$$

- Since $K_e$ is typically large $\rightarrow e_f \approx f_{\text{dist}}$ – best possible!
- $f_{\dot{e}_d}$ and $f_{\ddot{e}_d}$ not specified – no physical sense in derivative of desired force!
- $f_e$ measured but $\dot{f}_e$ very difficult to measure $\rightarrow \dot{f}_e = K_e \dot{x}$.
- Control law with above constraints

$$f = m[K_{pf}K_e^{-1}e_f - K_{vf}\dot{x}] + f_{e_d}$$
FORCE CONTROL OF MANIPULATORS
FORCE CONTROL OF A SINGLE MASS (Contd.)

Equation: \[ f = m\ddot{x} + K_e x + f_{dist} \]

Figure 25: A force control scheme for a spring-mass system
Difficult to estimate $K_e$ — can change with time.

Choose $K_e$ large as most environments are “stiff”.

Terms in $\beta$ and derivatives of $f_{ed}$ dropped $\rightarrow e_f$ does not go to zero as in a second-order system!

Six DOF manipulator, $\mathcal{F}$ and $\mathcal{K}$ are $6 \times 1$ entities (not vectors!), $m$ is the Cartesian mass matrix and $K_e$ is a $6 \times 6$ positive-definite (diagonal) stiffness matrix.

The gain matrices $[K_{pf}]$ and $[K_{vf}]$ are $6 \times 6$ positive definite and diagonal matrices.
Difficult to estimate $K_e$ – can change with time.

Choose $K_e$ large as most environments are “stiff”.

Terms in $\beta$ and derivatives of $f_{ed}$ dropped $\rightarrow e_f$ does not go to zero as in a second-order system!

Six DOF manipulator, $F$ and $H$ are $6 \times 1$ entities (not vectors!), $m$ is the Cartesian mass matrix and $K_e$ is a $6 \times 6$ positive-definite (diagonal) stiffness matrix.

The gain matrices $[K_{pf}]$ and $[K_{vf}]$ are $6 \times 6$ positive definite and diagonal matrices.
FORCE CONTROL OF MANIPULATORS

Force control of a single mass (Contd.)

- Difficult to estimate $K_e$ – can change with time.
- Choose $K_e$ large as most environments are “stiff”.
- Terms in $\beta$ and derivatives of $f_{ed}$ dropped $\rightarrow e_f$ does not go to zero as in a second-order system!
- Six DOF manipulator, $\mathcal{F}$ and $\mathcal{K}$ are $6 \times 1$ entities (not vectors!), $m$ is the Cartesian mass matrix and $K_e$ is a $6 \times 6$ positive-definite (diagonal) stiffness matrix.
- The gain matrices $[K_{pf}]$ and $[K_{vf}]$ are $6 \times 6$ positive definite and diagonal matrices.
Difficult to estimate $K_e$ – can change with time.

Choose $K_e$ large as most environments are “stiff”.

Terms in $\beta$ and derivatives of $f_{ed}$ dropped $\rightarrow e_f$ does not go to zero as in a second-order system!

Six DOF manipulator, $F$ and $X$ are $6 \times 1$ entities (not vectors!), $m$ is the Cartesian mass matrix and $K_e$ is a $6 \times 6$ positive-definite (diagonal) stiffness matrix.

The gain matrices $[K_{pf}]$ and $[K_{vf}]$ are $6 \times 6$ positive definite and diagonal matrices.
Difficult to estimate $K_e$ – can change with time.

Choose $K_e$ large as most environments are “stiff”.

Terms in $\beta$ and derivatives of $f_{ed}$ dropped $\rightarrow e_f$ does not go to zero as in a second-order system!

Six DOF manipulator, $\mathcal{T}$ and $\mathcal{X}$ are $6 \times 1$ entities (not vectors!), $m$ is the Cartesian mass matrix and $K_e$ is a $6 \times 6$ positive-definite (diagonal) stiffness matrix.

The gain matrices $[K_{pf}]$ and $[K_{vf}]$ are $6 \times 6$ positive definite and diagonal matrices.
Force control of manipulators

Force control for 6DOF system

- Principle of *duality* – Cannot control force and velocity(or position) in the same direction.
- Since force/torque and linear/angular velocity are related through power.
- Example – in robotic grinding, force can be controlled normal to surface being ground and velocity can be controlled tangent to the surface being ground.
- Duality is analogous to the partitioning of control torque in the planar 2R robot moving while satisfying a constraint (see Lecture 4).
- Cartesian control schemes naturally extend for force control!

---

5 A more accurate description can be given using advanced kinematic concepts of screws, wrenches, and the principle of reciprocity (see papers by Mason (1981), Raibert and Craig (1981) and others listed at the end of this module for a more detailed treatment.

6 Even if friction is taken into account, the force tangent to the surface cannot be arbitrary!
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL FOR 6DOF SYSTEM

- Principle of *duality* – Cannot control force *and* velocity(or position) in the same direction.

- Since force/torque and linear/angular velocity are related through power\(^5\).

- Example – in robotic grinding, force can be controlled normal to the surface being ground and velocity can be controlled tangent to the surface being ground\(^6\).

- Duality is analogous to the partitioning of control torque in the planar 2R robot moving while satisfying a constraint (see Lecture 4).

- Cartesian control schemes naturally extends for force control!

---

\(^5\)A more accurate description can be given using advanced kinematic concepts of screws, wrenches, and the principle of reciprocity (see papers by Mason (1981), Raibert and Craig (1981) and others listed at the end of this module for a more detailed treatment.

\(^6\)Even if friction is taken into account, the force tangent to the surface cannot be arbitrary!
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL FOR 6DOF SYSTEM

- Principle of duality – Cannot control force and velocity (or position) in the same direction.

- Since force/torque and linear/angular velocity are related through power\(^5\).

- Example – in robotic grinding, force can be controlled normal to surface being ground and velocity can be controlled tangent to the surface being ground\(^6\).

- Duality is analogous to the partitioning of control torque in the planar 2R robot moving while satisfying a constraint (see Lecture 4).

- Cartesian control schemes naturally extend for force control!

---

\(^5\) A more accurate description can be given using advanced kinematic concepts of screws, wrenches, and the principle of reciprocity (see papers by Mason (1981), Raibert and Craig (1981) and others listed at the end of this module for a more detailed treatment.

\(^6\) Even if friction is taken into account, the force tangent to the surface cannot be arbitrary!
FORCE CONTROL OF MANIPULATORS

Force control for 6DOF system

- Principle of *duality* – Cannot control force *and* velocity (or position) in the same direction.

- Since force/torque and linear/angular velocity are related through power\(^5\).

- Example – in robotic grinding, force can be controlled normal to the surface being ground and velocity can be controlled tangent to the surface being ground\(^6\).

- Duality is analogous to the partitioning of control torque in the planar 2R robot moving while satisfying a constraint (see Lecture 4).

- Cartesian control schemes naturally extend for force control!

\(^5\)A more accurate description can be given using advanced kinematic concepts of screws, wrenches, and the principle of reciprocity (see papers by Mason (1981), Raibert and Craig (1981) and others listed at the end of this module for a more detailed treatment.

\(^6\)Even if friction is taken into account, the force tangent to the surface cannot be arbitrary!
FORCE CONTROL OF MANIPULATORS

FORCE CONTROL FOR 6DOF SYSTEM

- Principle of *duality* – Cannot control force and velocity (or position) in the same direction.
- Since force/torque and linear/angular velocity are related through power\(^5\).
- Example – in robotic grinding, force can be controlled normal to surface being ground and velocity can be controlled tangent to the surface being ground\(^6\).
- Duality is analogous to the partitioning of control torque in the planar 2R robot moving while satisfying a constraint (see Lecture 4).
- Cartesian control schemes naturally extend for force control!

\(^5\) A more accurate description can be given using advanced kinematic concepts of screws, wrenches, and the principle of reciprocity (see papers by Mason (1981), Raibert and Craig (1981) and others listed at the end of this module for a more detailed treatment.

\(^6\) Even if friction is taken into account, the force tangent to the surface cannot be arbitrary!
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS

- Robotic tasks divided into subtasks – (a) in contact with environment or (b) in free space.
- Tasks in contact with environment – position control and force controlled ‘directions’.
- Natural constraint on position and force when manipulator in contact with a surface\(^7\) – involve variables that cannot be controlled.
  - Manipulator cannot go through surface – natural position constraint.
  - Manipulator cannot apply arbitrary force tangent to surface – natural force constraint.
- Natural position constraints normal to surface and natural force constraint tangent to surface.
- Can generate natural position and force constraints for any robotic task where robot in contact with environment.

\(^7\)We follow Craig (1989) for this treatment.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS

- Robotic tasks divided into subtasks – (a) in contact with environment or (b) in free space.
- Tasks in contact with environment – position control and force controlled ‘directions’.
  - *Natural constraint* on position and force when manipulator in contact with a surface\(^7\) – involve variables that *cannot be controlled*.
    - Manipulator cannot go through surface – natural position constraint.
    - Manipulator cannot apply arbitrary force tangent to surface – natural force constraint.
  - Natural position constraints *normal* to surface and natural force constraint *tangent* to surface.
  - Can generate natural position and force constraints for any robotic task where robot in contact with environment.

\(^7\)We follow Craig (1989) for this treatment.
FORCE CONTROL OF MANIPULATORS
PARTITIONING OF TASKS

- Robotic tasks divided into subtasks – (a) in contact with environment or (b) in free space.
- Tasks in contact with environment – position control and force controlled ‘directions’.
- *Natural constraint* on position and force when manipulator in contact with a surface\(^7\) – involve variables that *cannot be controlled*.
  - Manipulator cannot go through surface – natural position constraint.
  - Manipulator cannot apply arbitrary force tangent to surface – natural force constraint.

- Natural position constraints *normal* to surface and natural force constraint *tangent* to surface.
- Can generate natural position and force constraints for any robotic task where robot in contact with environment.

\(^7\)We follow Craig (1989) for this treatment.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS

- Robotic tasks divided into subtasks – (a) in contact with environment or (b) in free space.
- Tasks in contact with environment – position control and force controlled ‘directions’.
- Natural constraint on position and force when manipulator in contact with a surface\(^7\) – involve variables that cannot be controlled.
  - Manipulator cannot go through surface – natural position constraint.
  - Manipulator cannot apply arbitrary force tangent to surface – natural force constraint.
- Natural position constraints normal to surface and natural force constraint tangent to surface.
- Can generate natural position and force constraints for any robotic task where robot in contact with environment.

\(^7\)We follow Craig (1989) for this treatment.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS

- Robotic tasks divided into subtasks – (a) in contact with environment or (b) in free space.
- Tasks in contact with environment – position control and force controlled ‘directions’.

*Natural constraint* on position and force when manipulator in contact with a surface⁷ – involve variables that *cannot be controlled*.
  - Manipulator cannot go through surface – natural position constraint.
  - Manipulator cannot apply arbitrary force tangent to surface – natural force constraint.

- Natural position constraints *normal* to surface and natural force constraint *tangent* to surface.
- Can generate natural position and force constraints for any robotic task where robot in contact with environment.

---

⁷We follow Craig (1989) for this treatment.
**Force Control of Manipulators**

Partitioning of Tasks (Contd.)

- *Artificial constraints* – all position and force variables that *can be controlled*.

  - Manipulator in contact with environment
    - Position variables in the tangent direction can be controlled.
    - Force variables in the normal direction can be controlled.

- *Natural* and *Artificial* constraints partition position and force variables in two complementary sets.

- Follows from principle of duality.

- Typical examples shown next!
FORCE CONTROL OF MANIPULATORS

Partitioning of tasks (Contd.)

- *Artificial constraints* – all position and force variables that *can be controlled*.

- Manipulator in contact with environment
  - Position variables in the tangent direction can be controlled.
  - Force variables in the normal direction can be controlled.

- *Natural* and *Artificial* constraints partition position and force variables in two complementary sets.

- Follows from principle of duality.

- Typical examples shown next!
**FORCE CONTROL OF MANIPULATORS**

**PARTITIONING OF TASKS (Contd.)**

- *Artificial constraints* – all position and force variables that *can be controlled*.

- Manipulator in contact with environment
  - Position variables in the tangent direction can be controlled.
  - Force variables in the normal direction can be controlled.

- *Natural* and *Artificial* constraints partition position and force variables in two complementary sets.
  - Follows from principle of duality.
  - Typical examples shown next!
**FORCE CONTROL OF MANIPULATORS**

**PARTITIONING OF TASKS (Contd.)**

- *Artificial constraints* – all position and force variables that *can be controlled*.

- Manipulator in contact with environment
  - Position variables in the tangent direction can be controlled.
  - Force variables in the normal direction can be controlled.

- *Natural* and *Artificial* constraints partition position and force variables in two complementary sets.

- Follows from principle of duality.

- Typical examples shown next!
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS (Contd.)

- **Artificial constraints** – all position and force variables that *can be controlled*.
- Manipulator in contact with environment
  - Position variables in the tangent direction can be controlled.
  - Force variables in the normal direction can be controlled.
- **Natural** and **Artificial** constraints partition position and force variables in two complementary sets.
- Follows from principle of duality.
- Typical examples shown next!
**FORCE CONTROL OF MANIPULATORS**

(a) Grinding a Surface

(b) Turning a Crank

**Figure 26:** Natural and artificial constraints for two tasks
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 1

- Manipulator holding a grinding wheel grinding a surface.
- Define constraint frame \( \{C\} \) at end-effector
  - \( \hat{C}Z \) is parallel to the normal \( \mathbf{n} \)
  - \( \hat{C}X \) and \( \hat{C}Y \) determine the tangent plane at the point of contact on the surface.
- Grinding – a desired force along the normal and a desired trajectory on the surface.
- All constraints described in \( \{C\} \) using linear velocity components \( V_x, V_y, V_z \), angular velocity components \( \omega_x, \omega_y, \omega_z \), force components \( f_x, f_y, f_z \), and moment components \( n_x, n_y, n_z \).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 1

- Manipulator holding a grinding wheel grinding a surface.
- Define constraint frame \{C\} at end-effector
  - \(C\hat{Z}\) is parallel to the normal \(n\)
  - \(C\hat{X}\) and \(C\hat{Y}\) determine the tangent plane at the point of contact on the surface.
- Grinding – a desired force along the normal and a desired trajectory on the surface.
- All constraints described in \{C\} using linear velocity components \(V_x, V_y, V_z\), angular velocity components \(\omega_x, \omega_y, \omega_z\), force components \(f_x, f_y, f_z\), and moment components \(n_x, n_y, n_z\).
Manipulator holding a grinding wheel grinding a surface.

Define constraint frame \{ C \} at end-effector

- \( C\hat{Z} \) is parallel to the normal \( \mathbf{n} \)
- \( C\hat{X} \) and \( C\hat{Y} \) determine the tangent plane at the point of contact on the surface.

Grinding – a desired force along the normal and a desired trajectory on the surface.

All constraints described in \{ C \} using linear velocity components \( V_x, V_y, V_z \), angular velocity components \( \omega_x, \omega_y, \omega_z \), force components \( f_x, f_y, f_z \), and moment components \( n_x, n_y, n_z \).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 1

- Manipulator holding a grinding wheel grinding a surface.
- Define constraint frame \{C\} at end-effector
  - \(\hat{C}Z\) is parallel to the normal \(n\)
  - \(\hat{C}X\) and \(\hat{C}Y\) determine the tangent plane at the point of contact on the surface.
- Grinding – a desired force along the normal and a desired trajectory on the surface.
- All constraints described in \{C\} using linear velocity components \(V_x, V_y, V_z\), angular velocity components \(\omega_x, \omega_y, \omega_z\), force components \(f_x, f_y, f_z\), and moment components \(n_x, n_y, n_z\).
Force Control of Manipulators

Partitioning of Tasks – Example 1

- Cannot lose contact or interfere $\rightarrow V_z = 0$.
- Grinding wheel has area contact $\rightarrow \omega_x = \omega_y = 0$ so as not too loose contact.
- $f_x$, $f_y$ and $n_z$ determined by the friction $\rightarrow$ not arbitrary!
- $V_x$ and $V_y$ determine desired trajectory $\rightarrow$ artificial constraint.
- Desired force $f_z$ $\rightarrow$ artificial constraint.
- $\omega_z$, $n_x$ and $n_y$ from principle of duality $\rightarrow$ artificial constraints!
Cannot lose contact or interfere $\rightarrow V_z = 0$.

Grinding wheel has area contact $\rightarrow \omega_x = \omega_y = 0$ so as not too loose contact.

$f_x, f_y$ and $n_z$ determined by the friction – not arbitrary!

$V_x$ and $V_y$ determine desired trajectory $\rightarrow$ artificial constraint.

Desired force $f_z$ $\rightarrow$ artificial constraint.

$\omega_z, n_x$ and $n_y$ from principle of duality $\rightarrow$ artificial constraints!
Cannot loose contact or interfere → $V_z = 0$.

Grinding wheel has area contact → $\omega_x = \omega_y = 0$ so as not too loose contact.

$f_x$, $f_y$ and $n_z$ determined by the friction – not arbitrary!

$V_x$ and $V_y$ determine desired trajectory → artificial constraint.

Desired force $f_z$ → artificial constraint.

$\omega_z$, $n_x$ and $n_y$ from principle of duality → artificial constraints!
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 1

- Cannot lose contact or interfere $\rightarrow V_z = 0$.
- Grinding wheel has area contact $\rightarrow \omega_x = \omega_y = 0$ so as not too loose contact.
- $f_x$, $f_y$ and $n_z$ determined by the friction $\rightarrow$ not arbitrary!
- $V_x$ and $V_y$ determine \textit{desired} trajectory $\rightarrow$ artificial constraint.
- Desired force $f_z$ $\rightarrow$ artificial constraint.
- $\omega_z$, $n_x$ and $n_y$ from principle of duality $\rightarrow$ artificial constraints!
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 1

- Cannot lose contact or interfere \( \rightarrow V_z = 0 \).
- Grinding wheel has area contact \( \rightarrow \omega_x = \omega_y = 0 \) so as not too loose contact.
- \( f_x, f_y \) and \( n_z \) determined by the friction \( \rightarrow \) not arbitrary!
- \( V_x \) and \( V_y \) determine *desired* trajectory \( \rightarrow \) artificial constraint.
- Desired force \( f_z \) \( \rightarrow \) artificial constraint.
- \( \omega_z, n_x \) and \( n_y \) from principle of duality \( \rightarrow \) artificial constraints!
Cannot loose contact or interfere $\rightarrow V_z = 0$.

Grinding wheel has area contact $\rightarrow \omega_x = \omega_y = 0$ so as not too loose contact.

$f_x$, $f_y$ and $n_z$ determined by the friction $\rightarrow$ not arbitrary!

$V_x$ and $V_y$ determine *desired* trajectory $\rightarrow$ artificial constraint.

Desired force $f_z$ $\rightarrow$ artificial constraint.

$\omega_z$, $n_x$ and $n_y$ from principle of duality $\rightarrow$ artificial constraints!
Force Control of Manipulators
Partitioning of Tasks – Example 2

- Robot turning a crank – \( \{C\} \) as shown in figure.
  - \( V_x = V_z = 0 \) – No motion possible along \( \hat{C}\hat{X} \) or \( \hat{C}\hat{Z} \) direction.
  - \( \omega_x = \omega_y = 0 \) – No rotation possible along \( \hat{C}\hat{X} \)- and \( \hat{C}\hat{Y} \)-axis.
  - Cannot apply any force along the \( \hat{C}\hat{Y} \)-axis or apply moment about the \( \hat{C}\hat{Z} \)-axis.

- Artificial position constraints \( \rightarrow \) Controlled position/orientation variables \( V_y \) and \( \omega_z \).

- Artificial force constraints \( \rightarrow \) Controlled force/moment variables – \( f_x, f_z, n_x \) and \( n_y \).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 2

- Robot turning a crank – \{C\} as shown in figure.
- \(V_x = V_z = 0\) – No motion possible along \(C\hat{X}\) or \(C\hat{Z}\) direction.
- \(\omega_x = \omega_y = 0\) – No rotation possible along \(C\hat{X}\)- and \(C\hat{Y}\)-axis.
- Cannot apply any force along the \(C\hat{Y}\)-axis or apply moment about the \(C\hat{Z}\)-axis.
- Artificial position constraints \(\rightarrow\) Controlled position/orientation variables \(V_y\) and \(\omega_z\).
- Artificial force constraints \(\rightarrow\) Controlled force/moment variables – \(f_x\), \(f_z\), \(n_x\) and \(n_y\).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 2

- Robot turning a crank – \{C\} as shown in figure.
- \( V_x = V_z = 0 \) – No motion possible along \( \hat{c}X \) or \( \hat{c}Z \) direction.
- \( \omega_x = \omega_y = 0 \) – No rotation possible along \( \hat{c}X \)- and \( \hat{c}Y \)-axis.
- Cannot apply any force along the \( \hat{c}Y \)-axis or apply moment about the \( \hat{c}Z \)-axis.
- Artificial position constraints \( \rightarrow \) Controlled position/orientation variables \( V_y \) and \( \omega_z \).
- Artificial force constraints \( \rightarrow \) Controlled force/moment variables – \( f_x \), \( f_z \), \( n_x \) and \( n_y \).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 2

- Robot turning a crank – \{C\} as shown in figure.
- \(V_x = V_z = 0\) – No motion possible along \(C\hat{X}\) or \(C\hat{Z}\) direction.
- \(\omega_x = \omega_y = 0\) – No rotation possible along \(C\hat{X}\)- and \(C\hat{Y}\)-axis.
- Cannot apply any force along the \(C\hat{Y}\)-axis or apply moment about the \(C\hat{Z}\)-axis.

- Artificial position constraints \(\rightarrow\) Controlled position/orientation variables \(V_y\) and \(\omega_z\).
- Artificial force constraints \(\rightarrow\) Controlled force/moment variables – \(f_x\), \(f_z\), \(n_x\) and \(n_y\).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 2

- Robot turning a crank – \{C\} as shown in figure.
- \( V_x = V_z = 0 \) – No motion possible along \( C\hat{X} \) or \( C\hat{Z} \) direction.
- \( \omega_x = \omega_y = 0 \) – No rotation possible along \( C\hat{X} \)- and \( C\hat{Y} \)-axis.
- Cannot apply any force along the \( C\hat{Y} \)-axis or apply moment about the \( C\hat{Z} \)-axis.
- Artificial position constraints \(\rightarrow\) Controlled position/orientation variables \(V_y\) and \(\omega_z\).
- Artificial force constraints \(\rightarrow\) Controlled force/moment variables – \(f_x, f_z, n_x\) and \(n_y\).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE 2

- Robot turning a crank – \( \{C\} \) as shown in figure.
- \( V_x = V_z = 0 \) – No motion possible along \( C\hat{X} \) or \( C\hat{Z} \) direction.
- \( \omega_x = \omega_y = 0 \) – No rotation possible along \( C\hat{X} \)– and \( C\hat{Y} \)-axis.
- Cannot apply any force along the \( C\hat{Y} \)-axis or apply moment about the \( C\hat{Z} \)-axis.
- Artificial position constraints \( \rightarrow \) Controlled position/orientation variables \( V_y \) and \( \omega_z \).
- Artificial force constraints \( \rightarrow \) Controlled force/moment variables – \( f_x, f_z, n_x \) and \( n_y \).
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

Figure 27: Peg-in-hole assembly
Classic problem in robotic assembly – Assembly of a peg in a hole.

Assumptions:
- 2D motion of peg.
- No friction between peg and hole surface.
- Sensors available to find hole.

Can be divided into 4 stages.
- Stage 1 – motion in free space – figure (a)
- Stage 2 – motion while touching surface – figure (b)
- Stage 3 – insertion of peg in hole – figure (c)
- Stage 4 – completion of assembly – figure (d)
Classic problem in robotic assembly – Assembly of a peg in a hole.

Assumptions:
- 2D motion of peg.
- No friction between peg and hole surface.
- Sensors available to find hole.

Can be divided into 4 stages.
- Stage 1 – motion in free space – figure (a)
- Stage 2 – motion while touching surface – figure (b)
- Stage 3 – insertion of peg in hole – figure (c)
- Stage 4 – completion of assembly – figure (d)
Classic problem in robotic assembly – Assembly of a peg in a hole.

Assumptions:
- 2D motion of peg.
- No friction between peg and hole surface.
- Sensors available to find hole.

Can be divided into 4 stages.
- Stage 1 – motion in free space – figure (a)
- Stage 2 – motion while touching surface – figure (b)
- Stage 3 – insertion of peg in hole – figure (c)
- Stage 4 – completion of assembly – figure (d)
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- **Stage 1 – natural constraints**
  \[ \mathbf{C F} = [f_x, f_y, f_z; n_x, n_y, n_z]^T = 0 \]
  Motion in free space → no forces/moments on end-effector.

- **Stage 1 – artificial constraints**
  \[ \mathbf{C V} = [0, 0, v_a; 0, 0, 0]^T \]
  \( v_a \) is a desired approach velocity → manipulator under pure position control.

- **Stage 2 – natural constraints**
  - Once peg touches surface, no motion along \( \mathbf{C \hat{Z}} \) or rotation about \( \mathbf{C \hat{X}} \)- or \( \mathbf{C \hat{Y}} \)-axis.
  - Cannot apply force along the direction of sliding.

\[
\begin{align*}
V_z &= 0, \quad \omega_x = 0, \quad \omega_y = 0 \\
f_x &= 0, \quad f_y = 0, \quad n_z = 0
\end{align*}
\]
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- Stage 1 – natural constraints

\[ C \mathcal{F} = [f_x, f_y, f_z; n_x, n_y, n_z]^T = 0 \]

Motion in free space \(\rightarrow\) no forces/moments on end-effector.

- Stage 1 – artificial constraints

\[ C \psi = [0, 0, v_a; 0, 0, 0]^T \]

\(v_a\) is a desired approach velocity \(\rightarrow\) manipulator under *pure position* control.

- Stage 2 – natural constraints

Once peg touches surface, no motion along \(C\hat{Z}\) or rotation about \(C\hat{X}\)-
or \(C\hat{Y}\)-axis.

Cannot apply force along the direction of sliding.

\[ \begin{align*}
V_z &= 0, \quad \omega_x = 0, \quad \omega_y = 0 \\
        f_x &= 0, \quad f_y = 0, \quad n_z = 0
\end{align*} \]
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- Stage 1 – natural constraints
  \[ C \mathcal{F} = [f_x, f_y, f_z; n_x, n_y, n_z]^T = 0 \]
  Motion in free space → no forces/moments on end-effector.

- Stage 1 – artificial constraints
  \[ C \mathcal{V} = [0, 0, v_a; 0, 0, 0]^T \]
  \( v_a \) is a desired approach velocity → manipulator under pure position control.

- Stage 2 – natural constraints
  - Once peg touches surface, no motion along \( C \mathcal{Z} \) or rotation about \( C \mathcal{X} \)- or \( C \mathcal{Y} \)-axis.
  - Cannot apply force along the direction of sliding.

\[
\begin{align*}
V_z &= 0, & \omega_x &= 0, & \omega_y &= 0 \\
f_x &= 0, & f_y &= 0, & n_z &= 0
\end{align*}
\]
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

• Stage 2 – artificial constraints
  • Apply a small force along the $C\hat{Z}$-axis to keep it in contact.
  • Control the velocity along the $C\hat{X}$-axis.

\[
V_x = v_s, \quad V_y = 0, \quad \omega_z = 0
\]
\[
f_z = f_c, \quad n_x = 0, \quad n_y = 0
\]

$v_s$ and $f_c$ are the sliding velocity and the contact force.

• Stage 3 – natural constraints

\[
V_x = 0, \quad V_y = 0, \quad \omega_x = 0, \quad \omega_y = 0
\]
\[
f_z = 0, \quad n_z = 0
\]

After some motion along the $C\hat{X}$-axis, the peg will fall into the hole.

• Stage 3 – artificial constraints or controlled variables are

\[
V_z = v_i, \quad \omega_z = 0
\]
\[
f_x = 0, \quad f_y = 0, \quad n_x = 0, \quad n_y = 0
\]

$v_i$ is the insertion speed of the peg.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- **Stage 2 – artificial constraints**
  - Apply a small force along the $C\hat{Z}$-axis to keep it in contact.
  - Control the velocity along the $C\hat{X}$-axis.

\[
\begin{align*}
V_x &= v_s, \quad V_y = 0, \quad \omega_z = 0 \\
V_x &= f_c, \quad n_x = 0, \quad n_y = 0
\end{align*}
\]

$v_s$ and $f_c$ are the sliding velocity and the contact force.

- **Stage 3 – natural constraints**

\[
\begin{align*}
V_x &= 0, \quad V_y = 0, \quad \omega_x = 0, \quad \omega_y = 0 \\
f_z &= 0, \quad n_z = 0
\end{align*}
\]

After some motion along the $C\hat{X}$-axis, the peg will fall into the hole.

- **Stage 3 – artificial constraints or controlled variables are**

\[
\begin{align*}
V_z &= v_i, \quad \omega_z = 0 \\
f_x &= 0, \quad f_y = 0, \quad n_x = 0, \quad n_y = 0
\end{align*}
\]

$v_i$ is the insertion speed of the peg.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- Stage 2 – artificial constraints
  - Apply a small force along the $C\hat{Z}$-axis to keep it in contact.
  - Control the velocity along the $C\hat{X}$-axis.

\[
V_x = v_s, \quad V_y = 0, \quad \omega_z = 0
\]
\[
f_z = f_c, \quad n_x = 0, \quad n_y = 0
\]

$v_s$ and $f_c$ are the sliding velocity and the contact force.

- Stage 3 – natural constraints

\[
V_x = 0, \quad V_y = 0, \quad \omega_x = 0, \quad \omega_y = 0
\]
\[
f_z = 0, \quad n_z = 0
\]

After some motion along the $C\hat{X}$-axis, the peg will fall into the hole.

- Stage 3 – artificial constraints or controlled variables are

\[
V_z = v_i, \quad \omega_z = 0
\]
\[
f_x = 0, \quad f_y = 0, \quad n_x = 0, \quad n_y = 0
\]

$v_i$ is the insertion speed of the peg.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- Stage 4 – natural constraints

\[ \mathbf{c}_Y = [V_x, V_y, V_z; \omega_x, \omega_y, \omega_z]^T = 0 \]

No motion after a full insertion.

- Stage 4 – controlled variables

\[ \mathbf{c}_F = [f_x, f_y, f_z; n_x, n_y, n_z]^T = 0 \]

No force should be applied after assembly is over!

- Switching between stages decided by monitoring changes in natural constraints – not the variable being controlled!
  - Stage 1 to 2 – monitor force along \( \mathbf{c}_Z \rightarrow \) This should cross (from 0) a chosen threshold value.
  - Stage 2 to 3 – monitor motion along \( \mathbf{c}_Z \rightarrow \) This should cross (from 0) a chosen threshold value.
  - Stage 3 to 4 – monitor force along \( \mathbf{c}_Z \rightarrow \) This should cross (from 0) a chosen threshold value.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- **Stage 4 – natural constraints**

\[ C\mathcal{Y} = [V_x, V_y, V_z; \omega_x, \omega_y, \omega_z]^T = 0 \]

No motion after a full insertion.

- **Stage 4 – controlled variables**

\[ C\mathcal{F} = [f_x, f_y, f_z; n_x, n_y, n_z]^T = 0 \]

No force should be applied after assembly is over!

- Switching between stages decided by monitoring changes in natural constraints – not the variable being controlled!
  - Stage 1 to 2 – monitor force along \( C\mathcal{Z} \) → This should cross (from 0) a chosen threshold value.
  - Stage 2 to 3 – monitor motion along \( C\mathcal{Z} \) → This should cross (from 0) a chosen threshold value.
  - Stage 3 to 4 – monitor force along \( C\mathcal{Z} \) → This should cross (from 0) a chosen threshold value.
FORCE CONTROL OF MANIPULATORS

PARTITIONING OF TASKS – EXAMPLE OF ASSEMBLY

- Stage 4 – natural constraints

\[ C \psi = [V_x, V_y, V_z; \omega_x, \omega_y, \omega_z]^T = 0 \]

No motion after a full insertion.

- Stage 4 – controlled variables

\[ C \Phi = [f_x, f_y, f_z; n_x, n_y, n_z]^T = 0 \]

No force should be applied after assembly is over!

- Switching between stages decided by monitoring changes in natural constraints – not the variable being controlled!

  - Stage 1 to 2 – monitor force along \( C \hat{Z} \) → This should cross (from 0) a chosen threshold value.
  - Stage 2 to 3 – monitor motion along \( C \hat{Z} \) → This should cross (from 0) a chosen threshold value.
  - Stage 3 to 4 – monitor force along \( C \hat{Z} \) → This should cross (from 0) a chosen threshold value.
OUTLINE

1 CONTENTS

2 LECTURE 1
   • Motion planning

3 LECTURE 2
   • Control of a single link

4 LECTURE 3
   • Control of a multi-link serial manipulator

5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators

6 LECTURE 5*
   • Force control of manipulators
   • **Hybrid position/force control of manipulators**

7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
Many robotic tasks require position and force control at the same time.

- Position and force control not in the same direction!
- Joint space scheme shown for planar 2R with constraint not feasible for spatial and multi-DOF motions.
- Cartesian position and force control algorithms can be combined.
- Choose position and force control variables using a task planner as shown in examples.
- Form a selector switch to select appropriate position and force variables for control!
Many robotic tasks require position and force control at the same time.

Position and force control not in the same direction!

Joint space scheme shown for planar 2R with constraint not feasible for spatial and multi-DOF motions.

Cartesian position and force control algorithms can be combined.

Choose position and force control variables using a task planner as shown in examples.

Form a selector switch to select appropriate position and force variables for control!
Many robotic tasks require position and force control at the same time.

- Position and force control not in the same direction!
- Joint space scheme shown for planar 2R with constraint not feasible for spatial and multi-DOF motions.

- Cartesian position and force control algorithms can be combined.
- Choose position and force control variables using a task planner as shown in examples.
- Form a selector switch to select appropriate position and force variables for control!
Many robotic tasks require position and force control at the same time.

Position and force control not in the same direction!

Joint space scheme shown for planar 2R with constraint not feasible for spatial and multi-DOF motions.

Cartesian position and force control algorithms can be combined.

Choose position and force control variables using a task planner as shown in examples.

Form a selector switch to select appropriate position and force variables for control!
HYBRID POSITION/FORCE CONTROL OF MANIPULATORS

OVERVIEW

- Many robotic tasks require position and force control at the same time.
- Position and force control not in the same direction!
- Joint space scheme shown for planar 2R with constraint not feasible for spatial and multi-DOF motions.
- Cartesian position and force control algorithms can be combined.
- Choose position and force control variables using a task planner as shown in examples.
- Form a selector switch to select appropriate position and force variables for control!
HYBRID POSITION/FORCE CONTROL OF MANIPULATORS

Overview

- Many robotic tasks require position and force control at the same time.
- Position and force control not in the same direction!
- Joint space scheme shown for planar 2R with constraint not feasible for spatial and multi-DOF motions.
- Cartesian position and force control algorithms can be combined.
- Choose position and force control variables using a task planner as shown in examples.
- Form a selector switch to select appropriate position and force variables for control!
**Figure 28:** A hybrid position/force controller
Hybrid position/force control of manipulators

- Top half of figure implements Cartesian position control & Bottom half of figure implements force control.
- Output of both controllers are $F$ and can be combined!
- Matrix $[S]$ and $[S']$ selector switches to select position and force variables, according to principle of duality.
- For Stage 2 in peg-in-hole assembly

\[
[S] = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}, \quad [S'] = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]
Top half of figure implements Cartesian position control & Bottom half of figure implements force control.

Output of both controllers are $\mathcal{F}$ and can be combined!

Matrix $[S]$ and $[S']$ selector switches to select position and force variables, according to principle of duality.

For Stage 2 in peg-in-hole assembly

$$
[S] = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix},
\quad
[S'] = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
$$
**Hybrid position/force control of manipulators**

- Top half of figure implements Cartesian position control & Bottom half of figure implements force control.
- Output of both controllers are $F$ and can be combined!
- Matrix $[S]$ and $[S']$ selector switches to select position and force variables, according to principle of duality.
- For Stage 2 in peg-in-hole assembly

\[
[S] = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix}, \quad [S'] = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
\end{pmatrix}
\]
Hybrid position/force control of manipulators

- Top half of figure implements Cartesian position control & Bottom half of figure implements force control.
- Output of both controllers are $\mathcal{F}$ and can be combined!
- Matrix $[S]$ and $[S']$ selector switches to select position and force variables, according to principle of duality.
- For Stage 2 in peg-in-hole assembly

$$[S] = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}, \quad [S'] = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$
OUTLINE

1 CONTENTS

2 LECTURE 1
   • Motion planning

3 LECTURE 2
   • Control of a single link

4 LECTURE 3
   • Control of a multi-link serial manipulator

5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators

6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators

7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
**Advanced Topics - Stability**

**Overview**

- **Stability** – *bounded input gives bounded output.*
- Linear controller – easy to analyse
  - Single-link manipulator under a proportional control scheme.
  - $\Omega(t) \to \Omega_d(t)$ as $t \to \infty$
  - Proportional controller is stable.
  - PD is also stable but PID can become unstable!
- Non-linear controllers – difficult to analyse for stability.
- Stability analysis using Lyapunov’s method.
- Controllability.
Stability – *bounded input gives bounded output*.

Linear controller – easy to analyse
- Single-link manipulator under a proportional control scheme.
- $\Omega(t) \rightarrow \Omega_d(t)$ as $t \rightarrow \infty$
- Proportional controller is stable.
- PD is also stable but PID can become unstable!

Non-linear controllers – difficult to analyse for stability.
- Stability analysis using Lyapunov’s method.
- Controllability.
**Advanced Topics - Stability**

**Overview**

- Stability – *bounded input gives bounded output.*
- Linear controller – easy to analyse
  - Single-link manipulator under a proportional control scheme.
  - $\Omega(t) \to \Omega_d(t)$ as $t \to \infty$
  - Proportional controller is stable.
  - PD is also stable but PID can become unstable!
- Non-linear controllers – difficult to analyse for stability.
  - Stability analysis using Lyapunov’s method.
  - Controllability.
Stability – *bounded input gives bounded output*.

- Linear controller – easy to analyse
  - Single-link manipulator under a proportional control scheme.
  - $\Omega(t) \rightarrow \Omega_d(t)$ as $t \rightarrow \infty$
  - Proportional controller is stable.
  - PD is also stable but PID can become unstable!

- Non-linear controllers – difficult to analyse for stability.
- Stability analysis using Lyapunov’s method.
  - Controllability.
Stability – *bounded input gives bounded output*.

Linear controller – easy to analyse
- Single-link manipulator under a proportional control scheme.
- $\Omega(t) \rightarrow \Omega_d(t)$ as $t \rightarrow \infty$
- Proportional controller is stable.
- PD is also stable but PID can become unstable!

Non-linear controllers – difficult to analyse for stability.

Stability analysis using Lyapunov’s method.

Controllability.
**Stability analysis using Lyapunov’s method**

- Lyapunov (1892) Russian mathematician.
- One of the few general and widely used result for non-linear systems.
- Non-linear system described in the form
  \[ \dot{X} = f(X, t) \]
  
  \( X \in \mathbb{R}^n, f(X, t) \) are \( n \) vector functions, and \( t \) denotes the time
- ODE has a unique solution starting at a given initial condition \( X_0 \).
- Robot manipulators no *explicit* dependence on \( t \) and \( \dot{X} = f(X) \).


Stability analysis using Lyapunov’s method

- Lyapunov (1892) Russian mathematician.
- One of the few general and widely used result for non-linear systems.
- Non-linear system described in the form

\[ \dot{X} = f(X, t) \]

\[ X \in \mathbb{R}^n, \ f(X, t) \] are \( n \) vector functions, and \( t \) denotes the time
- ODE has a unique solution starting at a given initial condition \( X_0 \).
- Robot manipulators no \textit{explicit} dependence on \( t \) and \( \dot{X} = f(X) \).
Lyapunov (1892) Russian mathematician.

One of the few general and widely used result for non-linear systems.

Non-linear system described in the form

\[
\dot{X} = f(X, t)
\]

\(X \in \mathbb{R}^n, f(X, t)\) are \(n\) vector functions, and \(t\) denotes the time

- ODE has a unique solution starting at a given initial condition \(X_0\).
- Robot manipulators no explicit dependence on \(t\) and \(\dot{X} = f(X)\).
Stability analysis using Lyapunov’s method

- Lyapunov (1892) Russian mathematician.
- One of the few general and widely used result for non-linear systems.
- Non-linear system described in the form

\[ \dot{X} = f(X, t) \]

where \( X \in \mathbb{R}^n \), \( f(X, t) \) are \( n \) vector functions, and \( t \) denotes the time.
- ODE has a unique solution starting at a given initial condition \( X_0 \).
- Robot manipulators no explicit dependence on \( t \) and \( \dot{X} = f(X) \).
Stability analysis using Lyapunov’s method

- Lyapunov (1892) Russian mathematician.
- One of the few general and widely used result for non-linear systems.
- Non-linear system described in the form

\[ \dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}, t) \]

\( \mathbf{X} \in \mathbb{R}^n, \mathbf{f}(\mathbf{X}, t) \) are \( n \) vector functions, and \( t \) denotes the time
- ODE has a unique solution starting at a given initial condition \( \mathbf{X}_0 \).
- Robot manipulators no *explicit* dependence on \( t \) and \( \dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}) \).
Stability analysis using Lyapunov’s method

Stability analysis is performed at equilibrium point(s) or state(s).

- $X_e$ is called an equilibrium point or state when it satisfies $f(X) = 0$

- $X_e$ can be solved from $n$ non-linear algebraic/trigonometric equations.

- $f(X) = 0$ can have more than one solution.

- Need to investigate stability at all equilibrium points!

---

8 In a linear system $\dot{X} = [A]X$, only one equilibrium point $X = 0$ when $[A]$ is a constant non-singular matrix.
Stability analysis using Lyapunov’s method

- Stability analysis is performed at \( \text{equilibrium} \) point(s) or state(s).
- \( X_e \) is called an \( \text{equilibrium} \) point or state when it satisfies
  \[
  f(X) = 0
  \]
- \( X_e \) can be solved from \( n \) non-linear algebraic/trigonometric equations.
- \( f(X) = 0 \) can have \textit{more than one} solution\(^8\).
- Need to investigate stability at \textit{all} equilibrium points!

\(^8\)In a linear system \( \dot{X} = [A]X \), only one equilibrium point \( X = 0 \) when \( [A] \) is a constant non-singular matrix.
Stability analysis using Lyapunov’s method

- Stability analysis is performed at equilibrium point(s) or state(s).
- \( X_e \) is called an equilibrium point or state when it satisfies
  \[
  f(X) = 0
  \]
- \( X_e \) can be solved from \( n \) non-linear algebraic/trigonometric equations.
- \( f(X) = 0 \) can have more than one solution\(^8\).
- Need to investigate stability at all equilibrium points!

---

\(^8\) In a linear system \( \dot{X} = [A]X \), only one equilibrium point \( X = 0 \) when \( [A] \) is a constant non-singular matrix.
Stability analysis is performed at equilibrium point(s) or state(s).

\( X_e \) is called an equilibrium point or state when it satisfies

\[ f(X) = 0 \]

\( X_e \) can be solved from \( n \) non-linear algebraic/trigonometric equations.

\( f(X) = 0 \) can have more than one solution\(^8\).

Need to investigate stability at all equilibrium points!

---

\(^8\)In a linear system \( \dot{X} = [A]X \), only one equilibrium point \( X = 0 \) when \( [A] \) is a constant non-singular matrix.
Stability analysis is performed at *equilibrium* point(s) or state(s).

$X_e$ is called an *equilibrium* point or state when it satisfies

$$f(X) = 0$$

$X_e$ can be solved from $n$ non-linear algebraic/trigonometric equations.

$f(X) = 0$ can have *more than one* solution\(^8\).

Need to investigate stability at *all* equilibrium points!

---

\(^8\)In a linear system $\dot{X} = [A]X$, only one equilibrium point $X = 0$ when $[A]$ is a constant non-singular matrix.
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

LYAPUNOV’S SECOND METHOD

- Statement: A non-linear system $\dot{X} = f(X)$ is said to be asymptotically stable (in the sense of Lyapunov) if there exists a positive-definite, differentiable, scalar function of the state variables $V(X)$, with $\dot{V}(X)$ being negative definite.

- A function $f(x)$ is positive definite if $f(x) > 0$ for all $x \neq 0$ and is zero only when $x = 0$.

- Positive semi-definite, if $f(x) \geq 0$ & Negative definite if $f(x) < 0$.
  - $f(x) = x_1^2 + x_2^2$ is positive-definite
  - $f(x) = (x_1 - x_2)^2$ is positive semi-definite, and
  - $f(x) = -(x_1^2 + x_2^2)$ is negative definite.

- Motivation of Lyapunov’s theorem: Spring-mass-damper system is stable
  - Energy of system is positive-definite
  - Energy continuously decreases with time.
Stability analysis using Lyapunov’s method

Lyapunov’s Second Method

- Statement: A non-linear system $\dot{X} = f(X)$ is said to be asymptotically stable (in the sense of Lyapunov) if there exists a positive-definite, differentiable, scalar function of the state variables $V(X)$, with $\dot{V}(X)$ being negative definite.

- A function $f(x)$ is positive definite if $f(x) > 0$ for all $x \neq 0$ and is zero only when $x = 0$.

- Positive semi-definite, if $f(x) \geq 0$ & Negative definite if $f(x) < 0$.
  - $f(x) = x_1^2 + x_2^2$ is positive-definite
  - $f(x) = (x_1 - x_2)^2$ is positive semi-definite, and
  - $f(x) = -(x_1^2 + x_2^2)$ is negative definite.

- Motivation of Lyapunov’s theorem: Spring-mass-damper system is stable
  - Energy of system is positive-definite
  - Energy continuously decreases with time.
Stability Analysis Using Lyapunov’s Method

Lyapunov’s Second Method

- **Statement:** A non-linear system $\dot{X} = f(X)$ is said to be asymptotically stable (in the sense of Lyapunov) if there exists a *positive-definite*, differentiable, scalar function of the state variables $V(X)$, with $\dot{V}(X)$ being *negative definite*.

- A function $f(x)$ is positive definite if $f(x) > 0$ for all $x \neq 0$ and is zero only when $x = 0$.

- Positive semi-definite, if $f(x) \geq 0$ & Negative definite if $f(x) < 0$.
  - $f(x) = x_1^2 + x_2^2$ is positive-definite
  - $f(x) = (x_1 - x_2)^2$ is positive semi-definite, and
  - $f(x) = -(x_1^2 + x_2^2)$ is negative definite.

- **Motivation of Lyapunov’s theorem:** Spring-mass-damper system is *stable*
  - Energy of system is positive-definite
  - Energy continuously decreases with time.
**Statement:** A non-linear system $\dot{X} = f(X)$ is said to be asymptotically stable (in the sense of Lyapunov) if there exists a *positive-definite*, differentiable, scalar function of the state variables $V(X)$, with $\dot{V}(X)$ being *negative definite*.

- A function $f(x)$ is positive definite if $f(x) > 0$ for all $x \neq 0$ and is zero only when $x = 0$.
- Positive semi-definite, if $f(x) \geq 0$ & Negative definite if $f(x) < 0$.
  - $f(x) = x_1^2 + x_2^2$ is positive-definite
  - $f(x) = (x_1 - x_2)^2$ is positive semi-definite, and
  - $f(x) = -(x_1^2 + x_2^2)$ is negative definite.

**Motivation of Lyapunov’s theorem:** Spring-mass-damper system is *stable*

- Energy of system is positive-definite
- Energy continuously *decreases* with time.
**Stability analysis using Lyapunov’s method**

**Comments on Lyapunov stability**

- **Sufficient condition** for stability *not a necessary conditions*!
  - A single $V(X) > 0$ such that $\dot{V}(X) < 0 \Rightarrow$ Asymptotic stability!
  - For a $V(X) > 0$, if $\dot{V}(X) \neq 0 \Rightarrow$ system is *not* stable (or unstable) – Choice of $V(X)$ was not proper!!

- If $V(X) > 0$ and $\dot{V}(X) \leq 0 \rightarrow$ Asymptotic stability under certain conditions (LaSalle and Lefschetz (1961)).

- Local result – $X_e$ is asymptotically stable if any trajectory starting in a region around the point converges to $X_e$ as $t \to \infty$ (see Khalil (1992) or Vidyasagar (1993) for a more formal definition). Region of asymptotic stability or *domain of attraction* is more difficult to obtain!

- Lyapunov’s method is also applicable for *non-autonomous systems* $\dot{X} = f(X, t)$ (see Khalil (1992) and Vidyasagar (1993)).

- Main difficulty – finding appropriate Lyapunov function, $V(X)$. 
Sufficient condition for stability not a necessary conditions!

- A single $V(X) > 0$ such that $\dot{V}(X) < 0 \Rightarrow$ Asymptotic stability!
- For a $V(X) > 0$, if $\dot{V}(X) \neq 0 \not\Rightarrow$ system is not stable (or unstable) – Choice of $V(X)$ was not proper!!

- If $V(X) > 0$ and $\dot{V}(X) \leq 0 \rightarrow$ Asymptotic stability under certain conditions (LaSalle and Lefschetz (1961)).

- Local result – $X_e$ is asymptotically stable if any trajectory starting in a region around the point converges to $X_e$ as $t \to \infty$ (see Khalil (1992) or Vidyasagar (1993) for a more formal definition). Region of asymptotic stability or domain of attraction is more difficult to obtain!

- Lyapunov’s method is also applicable for non-autonomous systems $\dot{X} = f(X, t)$ (see Khalil (1992) and Vidyasagar (1993)).

- Main difficulty – finding appropriate Lyapunov function, $V(X)$. 
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

COMMENTS ON LYAPUNOV STABILITY

- **Sufficient condition** for stability *not a necessary conditions*!.
  - A single $V(X) > 0$ such that $\dot{V}(X) < 0 \Rightarrow$ Asymptotic stability!
  - For a $V(X) > 0$, if $\dot{V}(X) \not< 0 \nRightarrow$ system is *not* stable (or unstable) – Choice of $V(X)$ was not proper!!

- If $V(X) > 0$ and $\dot{V}(X) \leq 0$ $\rightarrow$ Asymptotic stability under certain conditions (LaSalle and Lefschetz (1961)).

- Local result – $X_e$ is asymptotically stable if any trajectory *starting in a region around the point* converges to $X_e$ as $t \rightarrow \infty$ (see Khalil (1992) or Vidyasagar (1993) for a more formal definition). Region of asymptotic stability or domain of attraction is more difficult to obtain!

- Lyapunov’s method is also applicable for *non-autonomous systems* $\dot{X} = f(X,t)$ (see Khalil (1992) and Vidyasagar (1993)).

- Main difficulty – finding appropriate Lyapunov function, $V(X)$. 

Asitava Ghosal  (IISc)  ROBOTICS: ADVANCED CONCEPTS & ANALYSIS  NPTEL, 2010  118 / 129
**STABILITY ANALYSIS USING LYAPUNOV’S METHOD**

**COMMENTS ON LYAPUNOV STABILITY**

- **Sufficient condition** for stability *not a necessary conditions*!.
  - A single $V(X) > 0$ such that $\dot{V}(X) < 0 \Rightarrow$ Asymptotic stability!
  - For a $V(X) > 0$, if $\dot{V}(X) \neq 0 \Rightarrow$ system is *not* stable (or unstable) – Choice of $V(X)$ was not proper!!

- If $V(X) > 0$ and $\dot{V}(X) \leq 0 \rightarrow$ Asymptotic stability under certain conditions (LaSalle and Lefschetz (1961)).

- Local result – $X_e$ is asymptotically stable if any trajectory *starting in a region around the point* converges to $X_e$ as $t \rightarrow \infty$ (see Khalil (1992) or Vidyasagar (1993) for a more formal definition). Region of asymptotic stability or *domain of attraction* is more difficult to obtain!

- Lyapunov’s method is also applicable for *non-autonomous systems* $\dot{X} = f(X, t)$ (see Khalil (1992) and Vidyasagar (1993)).

- Main difficulty – finding appropriate Lyapunov function, $V(X)$. 

---

**Ashitava Ghosal (IISc)**

**Robotics: Advanced Concepts & Analysis**

**NPTEL, 2010 118 / 129**
**Stability analysis using Lyapunov’s method**

**Comments on Lyapunov stability**

- **Sufficient condition** for stability *not a necessary conditions*!.
  - A single $V(\mathbf{X}) > 0$ such that $\dot{V}(\mathbf{X}) < 0 \Rightarrow$ Asymptotic stability!
  - For a $V(\mathbf{X}) > 0$, if $\dot{V}(\mathbf{X}) \neq 0 \Rightarrow$ system is *not* stable (or unstable) – Choice of $V(\mathbf{X})$ was not proper!!

- If $V(\mathbf{X}) > 0$ and $\dot{V}(\mathbf{X}) \leq 0 \Rightarrow$ Asymptotic stability under certain conditions (LaSalle and Lefschetz (1961)).

- Local result – $\mathbf{X}_e$ is asymptotically stable if any trajectory starting in a region around the point converges to $\mathbf{X}_e$ as $t \to \infty$ (see Khalil (1992) or Vidyasagar (1993) for a more formal definition). Region of asymptotic stability or *domain of attraction* is more difficult to obtain!

- Lyapunov’s method is also applicable for *non-autonomous systems* $\dot{\mathbf{X}} = f(\mathbf{X}, t)$ (see Khalil (1992) and Vidyasagar (1993)).

- Main difficulty – finding appropriate Lyapunov function, $V(\mathbf{X})$. 

Stability analysis using Lyapunov’s method

Example: Single link manipulator

- Equation of motion

\[ \ddot{\theta}_1 + \left(\frac{g}{l_1}\right) \sin \theta_1 = u(t) \]

where \( u(t) = \frac{\tau_1(t)}{(m_1 l_1^2)} \) and \( \theta_1 \) is angle as shown.

- State equation with

\[ (X_1, X_2)^T = (\theta_1, \dot{\theta}_1)^T \]

\[ \dot{X}_1 = X_2, \quad \dot{X}_2 = u(t) - \left(\frac{g}{l_1}\right) \sin(X_1) \]

- Equilibrium points: \( u(t) = 0, \theta_1 = 0 \) and \( \theta_1 = \pi \).

Figure 29: A single link manipulator
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

- Investigate stability at $\theta_1 = 0$.
- Candidate Lyapunov function
  \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) \]
- $V(X_1, X_2) = \text{Total Energy} > 0 \text{ & Zero only when } X_1 = X_2 = 0 - \text{Zero potential energy at } y = -l_1$.
- $\dot{V}(X_1, X_2)$ at equilibrium point $\theta_1 = 0$ is given by
  \[ \dot{V}(X_1, X_2) = m_1 l_1^2 X_2 \dot{X}_2 + m_1 g l_1 \sin(X_1) \dot{X}_1 = 0 \]
- Not asymptotically stable!!
Investigate stability at $\theta_1 = 0$.

Candidate Lyapunov function

$$V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1))$$

$V(X_1, X_2)$ = Total Energy $> 0$ & Zero only when $X_1 = X_2 = 0$ – Zero potential energy at $y = -l_1$.

$\dot{V}(X_1, X_2)$ at equilibrium point $\theta_1 = 0$ is given by

$$\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 \dot{X}_2 + m_1 g l_1 \sin(X_1) \dot{X}_1 = 0$$

Not asymptotically stable!!
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

- Investigate stability at \( \theta_1 = 0 \).
- Candidate Lyapunov function
  \[
  V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1))
  \]
- \( V(X_1, X_2) = \text{Total Energy} > 0 \) & Zero only when \( X_1 = X_2 = 0 \) – Zero potential energy at \( y = -l_1 \).
- \( \dot{V}(X_1, X_2) \) at equilibrium point \( \theta_1 = 0 \) is given by
  \[
  \dot{V}(X_1, X_2) = m_1 l_1^2 X_2 \dot{X}_2 + m_1 g l_1 \sin(X_1) \dot{X}_1 = 0
  \]
- Not asymptotically stable!!
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

EXAMPLE: SINGLE LINK MANIPULATOR (Contd.)

- Investigate stability at $\theta_1 = 0$.
- Candidate Lyapunov function

\[
V(X_1, X_2) = \frac{1}{2}m_1(l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1))
\]

- $V(X_1, X_2) = \text{Total Energy} > 0 \ & \text{Zero only when } X_1 = X_2 = 0 - \text{Zero potential energy at } y = -l_1$.
- $\dot{V}(X_1, X_2)$ at equilibrium point $\theta_1 = 0$ is given by

\[
\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 \dot{X}_2 + m_1 g l_1 \sin(X_1) \dot{X}_1 = 0
\]

- Not asymptotically stable!!
Investigate stability at $\theta_1 = 0$.

Candidate Lyapunov function

$$V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1))$$

$V(X_1, X_2) = \text{Total Energy} > 0$ & Zero only when $X_1 = X_2 = 0$ – Zero potential energy at $y = -l_1$.

$\dot{V}(X_1, X_2)$ at equilibrium point $\theta_1 = 0$ is given by

$$\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 \dot{X}_2 + m_1 g l_1 \sin(X_1) \dot{X}_1 = 0$$

Not asymptotically stable!!
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

Consider added damping

\[ \dot{X}_1 = X_2, \quad \dot{X}_2 = -(g/l_1)\sin(X_1) - cX_2, \quad c > 0 \]

For above state equations, \( \dot{V}(X_1, X_2) \) at \( \theta_1 = 0 \) is

\[ \dot{V}(X_1, X_2) = -m_1/l_1^2 cX_2^2 < 0 \]

Single link manipulator with damping is asymptotically stable!

Consider actuator output \( \tau_1(t) = K_p(X_{1d} - X_1), \quad K_p > 0 \), or \( u(t) \) given by

\[ u(t) = K_p(X_{1d} - X_1)/(m_1/l_1^2), \quad K_p > 0 \]

where \( X_{1d} \) denotes a desired \( \theta_1 \).
**Stability analysis using Lyapunov’s method**

**Example: Single link manipulator (Contd.)**

- Consider added damping

\[
\dot{X}_1 = X_2, \quad \dot{X}_2 = -\left(\frac{g}{l_1}\right)\sin(X_1) - cX_2, \quad c > 0
\]

- For above state equations, \(\dot{V}(X_1, X_2)\) at \(\theta_1 = 0\) is

\[
\dot{V}(X_1, X_2) = -m_1 l_1^2 cX_2^2 < 0
\]

- Single link manipulator with damping is asymptotically stable!

- Consider actuator output \(\tau_1(t) = K_p(X_{1d} - X_1), \quad K_p > 0\), or \(u(t)\) given by

\[
\dot{u}(t) = K_p(X_{1d} - X_1)/(m_1 l_1^2), \quad K_p > 0
\]

where \(X_{1d}\) denotes a desired \(\theta_1\).
**STABILITY ANALYSIS USING LYAPUNOV’S METHOD**

**EXAMPLE: SINGLE LINK MANIPULATOR (Contd.)**

- Consider added damping
  \[
  \dot{X}_1 = X_2, \quad \dot{X}_2 = -(g/l_1) \sin(X_1) - cX_2, \quad c > 0
  \]

- For above state equations, \( \dot{V}(X_1, X_2) \) at \( \theta_1 = 0 \) is
  \[
  \dot{V}(X_1, X_2) = -m_1 l_1^2 c X_2^2 < 0
  \]

- Single link manipulator *with damping* is asymptotically stable!

- Consider actuator output \( \tau_1(t) = K_p(X_{1d} - X_1), \; K_p > 0 \), or \( u(t) \) given by
  \[
  u(t) = K_p(X_{1d} - X_1)/(m_1 l_1^2), \quad K_p > 0
  \]

  where \( X_{1d} \) denotes a desired \( \theta_1 \).
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

Consider added damping

$$\dot{X}_1 = X_2, \quad \dot{X}_2 = -(g/l_1)\sin(X_1) - cX_2, \quad c > 0$$

For above state equations, $$\dot{V}(X_1, X_2)$$ at $$\theta_1 = 0$$ is

$$\dot{V}(X_1, X_2) = -m_1 l_1^2 c X_2^2 < 0$$

Single link manipulator with damping is asymptotically stable!

Consider actuator output $$\tau_1(t) = K_p(X_{1d} - X_1), \quad K_p > 0$$, or $$u(t)$$ given by

$$u(t) = K_p(X_{1d} - X_1)/(m_1 l_1^2), \quad K_p > 0$$

where $$X_{1d}$$ denotes a desired $$\theta_1$$. 
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

- Investigate stability for $X_{1d} = 0$.
- Choose candidate Lyapunov function as

$$V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2$$

- $V(X_1, X_2)$ is positive definite.
- For the undamped state equations,

$$\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 u(t) + K_p X_1 X_2$$

- For $u(t) = -K_p X_1 / (m_1 l_1^2) \rightarrow \dot{V}(X_1, X_2) = 0 \Rightarrow$ Not asymptotically stable!
- With damping, $\dot{V}(X_1, X_2) < 0 \Rightarrow$ Asymptotic stability at $X_{1d}$.

\[\text{If } X_{1d} \neq 0, \text{ perform a change of coordinates } X_1' = X_{1d} - X_1.\]
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

- Investigate stability for $X_{1d} = 0^9$.
- Choose candidate Lyapunov function as

$$V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2$$

- $V(X_1, X_2)$ is positive definite.
- For the undamped state equations,

$$\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 u(t) + K_p X_1 X_2$$

- For $u(t) = -K_p X_1 / (m_1 l_1^2) \rightarrow \dot{V}(X_1, X_2) = 0 \Rightarrow$ Not asymptotically stable!

- With damping, $\dot{V}(X_1, X_2) < 0 \Rightarrow$ Asymptotic stability at $X_{1d}$.

\[9\text{If } X_{1d} \neq 0, \text{ perform a change of coordinates } X_1' = X_{1d} - X_1.\]
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

- Investigate stability for $X_{1d} = 0^9$.
- Choose candidate Lyapunov function as

$$V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2$$

- $V(X_1, X_2)$ is positive definite.
- For the undamped state equations,

$$\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 u(t) + K_p X_1 X_2$$

- For $u(t) = -K_p X_1 / (m_1 l_1^2) \rightarrow \dot{V}(X_1, X_2) = 0 \Rightarrow$ Not asymptotically stable!
- With damping, $\dot{V}(X_1, X_2) < 0 \Rightarrow$ Asymptotic stability at $X_{1d}$.

If $X_{1d} \neq 0$, perform a change of coordinates $X_1' = X_{1d} - X_1$. 
**Stability analysis using Lyapunov’s method**

**Example: Single link manipulator (Contd.)**

- Investigate stability for $X_{1_d} = 0^9$.
- Choose candidate Lyapunov function as
  \[
  V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2
  \]
- $V(X_1, X_2)$ is positive definite.
- For the *undamped* state equations,
  \[
  \dot{V}(X_1, X_2) = m_1 l_1^2 X_2 u(t) + K_p X_1 X_2
  \]
- For $u(t) = -K_p X_1/(m_1 l_1^2) \rightarrow \dot{V}(X_1, X_2) = 0 \Rightarrow$ Not asymptotically stable!
- With damping, $\dot{V}(X_1, X_2) < 0 \Rightarrow$ Asymptotic stability at $X_{1_d}$.

---

$^9$If $X_{1_d} \neq 0$, perform a change of coordinates $X_1' = X_{1_d} - X_1$.
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

EXAMPLE: SINGLE LINK MANIPULATOR (CONT'D.)

- Investigate stability for $X_{1d} = 0^9$.
- Choose candidate Lyapunov function as
  
  $$V(X_1, X_2) = \frac{1}{2} m_1(l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2$$

- $V(X_1, X_2)$ is positive definite.
- For the undamped state equations,
  
  $$\dot{V}(X_1, X_2) = m_1 l_1^2 X_2 u(t) + K_p X_1 X_2$$

- For $u(t) = -K_p X_1/(m_1 l_1^2) \rightarrow \dot{V}(X_1, X_2) = 0 \Rightarrow$ Not asymptotically stable!
- With damping, $\dot{V}(X_1, X_2) < 0 \Rightarrow$ Asymptotic stability at $X_{1d}$.

---

$^9$ If $X_{1d} \neq 0$, perform a change of coordinates $X_1' = X_{1d} - X_1$.
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

1. Investigate stability for $X_1^d = 0^9$.
2. Choose candidate Lyapunov function as

   \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

3. $V(X_1, X_2)$ is positive definite.
4. For the undamped state equations,

   \[ \dot{V}(X_1, X_2) = m_1 l_1^2 X_2 u(t) + K_p X_1 X_2 \]

5. For $u(t) = -K_p X_1 / (m_1 l_1^2) \rightarrow \dot{V}(X_1, X_2) = 0 \Rightarrow$ Not asymptotically stable!
6. With damping, $\dot{V}(X_1, X_2) < 0 \Rightarrow$ Asymptotic stability at $X_1^d$.

---

9If $X_1^d \neq 0$, perform a change of coordinates $X_1' = X_1^d - X_1$.
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

Consider a (modified) proportional plus derivative (PD) control

\[ \tau_1(t) = -K_p X_1 - K_v \dot{X}_1, \quad K_p, K_v > 0 \]

Consider the candidate Lyapunov function

\[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

For the undamped system \[ \dot{V}(X_1, X_2) = -K_v X_2^2 < 0 \Rightarrow \text{Asymptotically stable!} \]

\( \dot{X}_{1_d} \) is assumed zero – Cannot prove asymptotic stability for trajectory following when \( \dot{X}_{1_d} \) is non-zero!

Not possible to prove stability for second equilibrium point \( \theta_1 = \pi \) using Lyapunov’s second method.

Recall \( V(X) > 0 \) and \( \dot{V}(X) < 0 \) is a sufficient condition for stability – \( \theta_1 = \pi \) is known to be unstable!!
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

EXAMPLE: SINGLE LINK MANIPULATOR (Contd.)

- Consider a (modified) proportional plus derivative (PD) control
  \[ \tau_1(t) = -K_p X_1 - K_v \dot{X}_1, \quad K_p, K_v > 0 \]

- Consider the candidate Lyapunov function
  \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

- For the undamped system \( \dot{V}(X_1, X_2) = -K_v X_2^2 < 0 \Rightarrow \text{Asymptotically stable!} \)
- \( \dot{X}_{1_d} \) is assumed zero – Cannot prove asymptotic stability for trajectory following when \( \dot{X}_{1_d} \) is non-zero!
- Not possible to prove stability for second equilibrium point \( \theta_1 = \pi \) using Lyapunov’s second method.
- Recall \( V(X) > 0 \) and \( \dot{V}(X) < 0 \) is a sufficient condition for stability – \( \theta_1 = \pi \) is known to be unstable!!
Stability Analysis Using Lyapunov’s Method

Example: Single Link Manipulator (Contd.)

- Consider a (modified) proportional plus derivative (PD) control
  \[ \tau_1(t) = -K_p X_1 - K_v \dot{X}_1, \quad K_p, K_v > 0 \]

- Consider the candidate Lyapunov function
  \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

- For the undamped system \( \dot{V}(X_1, X_2) = -K_v X_2^2 < 0 \Rightarrow \text{Asymptotically stable!} \)

- \( \dot{X}_{1d} \) is assumed zero – Cannot prove asymptotic stability for trajectory following when \( \dot{X}_{1d} \) is non-zero!

- Not possible to prove stability for second equilibrium point \( \theta_1 = \pi \) using Lyapunov’s second method.

- Recall \( V(X) > 0 \) and \( \dot{V}(X) < 0 \) is a sufficient condition for stability – \( \theta_1 = \pi \) is known to be unstable!!
Stability analysis using Lyapunov’s method

Example: Single link manipulator (Contd.)

- Consider a (modified) proportional plus derivative (PD) control
  \[ \tau_1(t) = -K_p X_1 - K_v \dot{X}_1, \quad K_p, K_v > 0 \]

- Consider the candidate Lyapunov function
  \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

- For the undamped system \( \dot{V}(X_1, X_2) = -K_v X_2^2 < 0 \) \( \Rightarrow \) Asymptotically stable!

- \( \dot{X}_{1_d} \) is *assumed* zero – Cannot prove asymptotic stability for trajectory following when \( \dot{X}_{1_d} \) is non-zero!

- *Not possible* to prove stability for second equilibrium point \( \theta_1 = \pi \) using Lyapunov’s second method.

- Recall \( V(X) > 0 \) and \( \dot{V}(X) < 0 \) is a *sufficient condition for stability* – \( \theta_1 = \pi \) is known to be unstable!!
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

EXAMPLE: SINGLE LINK MANIPULATOR (Contd.)

- Consider a (modified) proportional plus derivative (PD) control
  \[ \tau_1(t) = -K_p X_1 - K_v \dot{X}_1, \ K_p, K_v > 0 \]

- Consider the candidate Lyapunov function
  \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 gl_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

- For the undamped system \( \dot{V}(X_1, X_2) = -K_v X_2^2 < 0 \Rightarrow \) Asymptotically stable!

- \( \dot{X}_{1d} \) is assumed zero – Cannot prove asymptotic stability for trajectory following when \( \dot{X}_{1d} \) is non-zero!

- Not possible to prove stability for second equilibrium point \( \theta_1 = \pi \) using Lyapunov’s second method.

- Recall \( V(X) > 0 \) and \( \dot{V}(X) < 0 \) is a sufficient condition for stability – \( \theta_1 = \pi \) is known to be unstable!!
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

Example: Single link manipulator (Contd.)

- Consider a (modified) proportional plus derivative (PD) control
  \[ \tau_1(t) = -K_p X_1 - K_v \dot{X}_1, \quad K_p, K_v > 0 \]

- Consider the candidate Lyapunov function
  \[ V(X_1, X_2) = \frac{1}{2} m_1 (l_1 X_2)^2 + m_1 g l_1 (1 - \cos(X_1)) + \frac{1}{2} K_p X_1^2 \]

- For the undamped system \( \dot{V}(X_1, X_2) = -K_v X_2^2 < 0 \Rightarrow \text{Asymptotically stable!} \)

- \( \dot{X}_{1_d} \) is assumed zero – Cannot prove asymptotic stability for trajectory following when \( \dot{X}_{1_d} \) is non-zero!

- Not possible to prove stability for second equilibrium point \( \theta_1 = \pi \) using Lyapunov’s second method.

- Recall \( V(X) > 0 \) and \( \dot{V}(X) < 0 \) is a sufficient condition for stability – \( \theta_1 = \pi \) is known to be unstable!!
Stability analysis using Lyapunov’s method

PD control scheme

- Equations of motion of $n$-DOF manipulator without gravity

$$\tau = [M(q)]\ddot{q} + [C(q, \dot{q})]\dot{q}$$

- Consider a PD control of the form $\tau = -[K_p]q(t) - [K_v]\dot{q}(t)$. Note: $\dot{q}_d(t) = 0$ and $q_d = 0^{10}$.

- Consider a candidate Lyapunov function

$$V(q, \dot{q}) = \frac{1}{2} \dot{q}^T [M(q)] \ddot{q} + \frac{1}{2} q^T [K_p]q$$

- Evaluate $\dot{V}(q, \dot{q})$ to get

$$\dot{V}(q, \dot{q}) = \dot{q}^T [M(q)] \ddot{q} + \frac{1}{2} \dot{q}^T [\dot{M}(q)] \dot{q} + \dot{q}^T [K_p]q$$

$$= -\dot{q}^T [K_v] \dot{q} + \frac{1}{2} \dot{q}^T ([\ddot{M}(q)] - 2[C(q, \dot{q})]) \dot{q}$$

$[\ddot{M}]$ denotes the derivative of $[M]$ with respect to time.

$^{10}$Setting $q_d = 0$ is not a serious issue – perform a linear transformation $q' = q_d - q$ and investigate the stability at $q' = 0$. 
Stability analysis using Lyapunov’s method

PD control scheme

- Equations of motion of $n$-DOF manipulator without gravity
  \[ \tau = [M(q)]\ddot{q} + [C(q, \dot{q})]\dot{q} \]

- Consider a PD control of the form $\tau = -[K_p]q(t) - [K_v]\dot{q}(t)$. Note: $\dot{q}_d(t) = 0$ and $q_d = 0$.

- Consider a candidate Lyapunov function
  \[ V(q, \dot{q}) = \frac{1}{2} \dot{q}^T [M(q)]\ddot{q} + \frac{1}{2} q^T [K_p]q \]

- Evaluate $\dot{V}(q, \dot{q})$ to get
  \[ \dot{V}(q, \dot{q}) = \dot{q}^T [M(q)]\ddot{q} + \frac{1}{2} \dot{q}^T [\dot{M}(q)]\dot{q} + \dot{q}^T [K_p]q \]
  \[ = -\dot{q}^T [K_v]q + \frac{1}{2} \dot{q}^T ([\dot{M}(q)] - 2[C(q, \dot{q})])\dot{q} \]

$[\dot{M}]$ denotes the derivative of $[M]$ with respect to time.

\(^{10}\)Setting $q_d = 0$ is not a serious issue – perform a linear transformation $q' = q_d - q$ and investigate the stability at $q' = 0$. 
Stability analysis using Lyapunov’s method

PD control scheme

- Equations of motion of $n$-DOF manipulator \textit{without} gravity
  \[
  \tau = [\mathbf{M}(\mathbf{q})] \ddot{\mathbf{q}} + [\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})] \dot{\mathbf{q}}
  \]

- Consider a PD control of the form $\tau = -[K_p] \dot{\mathbf{q}}(t) - [K_v] \dot{\mathbf{q}}(t)$. Note: $\dot{\mathbf{q}}_d(t) = 0$ and $\mathbf{q}_d = 0^{10}$.

- Consider a candidate Lyapunov function
  \[
  \mathcal{V}(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^T [\mathbf{M}(\mathbf{q})] \dot{\mathbf{q}} + \frac{1}{2} \mathbf{q}^T [K_p] \mathbf{q}
  \]

- Evaluate $\dot{\mathcal{V}}(\mathbf{q}, \dot{\mathbf{q}})$ to get
  \[
  \dot{\mathcal{V}}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{\mathbf{q}}^T [\mathbf{M}(\mathbf{q})] \ddot{\mathbf{q}} + \frac{1}{2} \dot{\mathbf{q}}^T [\dot{\mathbf{M}}(\mathbf{q})] \dot{\mathbf{q}} + \dot{\mathbf{q}}^T [K_p] \mathbf{q}
  \]
  \[
  = -\dot{\mathbf{q}}^T [K_v] \dot{\mathbf{q}} + \frac{1}{2} \dot{\mathbf{q}}^T ([\dot{\mathbf{M}}(\mathbf{q})] - 2[\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})]) \dot{\mathbf{q}}
  \]

$[\dot{\mathbf{M}}]$ denotes the derivative of $[\mathbf{M}]$ with respect to time.

$^{10}$Setting $\mathbf{q}_d = 0$ is not a serious issue – perform a linear transformation $\mathbf{q}' = \mathbf{q}_d - \mathbf{q}$ and investigate the stability at $\mathbf{q}' = 0$. 
Stability analysis using Lyapunov’s method

PD control scheme

- Equations of motion of $n$-DOF manipulator without gravity
  \[ \tau = [M(q)]\ddot{q} + [C(q, \dot{q})]\dot{q} \]

- Consider a PD control of the form $\tau = -[K_p]q(t) - [K_v]\dot{q}(t)$. Note: $\dot{q}_d(t) = 0$ and $q_d = 0^{10}$.

- Consider a candidate Lyapunov function
  \[ V(q, \dot{q}) = \frac{1}{2} \dot{q}^T [M(q)]\ddot{q} + \frac{1}{2} q^T [K_p]q \]

- Evaluate $\dot{V}(q, \dot{q})$ to get
  \[ \dot{V}(q, \dot{q}) = \dot{q}^T [M(q)]\ddot{q} + \frac{1}{2} \dot{q}^T [\dot{M}(q)]\dot{q} + \dot{q}^T [K_p]q \]
  \[ = -\dot{q}^T [K_v]\dot{q} + \frac{1}{2} \dot{q}^T ([\dot{M}(q)] - 2[C(q, \dot{q})])\dot{q} \]

$[\dot{M}]$ denotes the derivative of $[M]$ with respect to time.

$^{10}$Setting $q_d = 0$ is not a serious issue — perform a linear transformation $q' = q_d - q$ and investigate the stability at $q' = 0$. 
Stability analysis using Lyapunov’s method

PD control scheme

- \([\dot{M}(q)] - 2[C(q, \dot{q})]\) is skew-symmetric \(\rightarrow\) the second quadratic form is zero, and
  
  \[\dot{V}(q, \dot{q}) = -\dot{q}^T[K_v]\dot{q}\]

- Since \(\dot{V}(q, \dot{q})\) can be zero even for non-zero \(q\), \(\dot{V}(q, \dot{q})\) is negative semi-definite

- By LaSalle’s invariance principle (LaSalle and Lefschetz 1961) \(\rightarrow\) equilibrium point \((q, \dot{q}) = 0\) is asymptotically stable.

- Assumption: \(\ddot{q}_d = \dot{q}_d = 0\) \(\rightarrow\) PD control scheme is not proved asymptotically stable for trajectory following!
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

PD CONTROL SCHEME

- \([\dot{M}(q)] - 2[C(q, \dot{q})]\) is skew-symmetric → the second quadratic form is zero, and
  \[\dot{V}(q, \dot{q}) = -\dot{q}^T[K_v]\dot{q}\]

- Since \(\dot{V}(q, \dot{q})\) can be zero even for non-zero \(q\), \(\dot{V}(q, \dot{q})\) is negative semi-definite

- By LaSalle’s invariance principle (LaSalle and Lefschetz 1961) → equilibrium point \((q, \dot{q}) = 0\) is asymptotically stable.

- Assumption: \(\ddot{q}_d = \dot{q}_d = 0\) → PD control scheme is not proved asymptotically stable for trajectory following!
Stability analysis using Lyapunov’s method

PD control scheme

- \([\dot{M}(q)] - 2[C(q, \dot{q})]\) is skew-symmetric → the second quadratic form is zero, and

\[
\dot{V}(q, \dot{q}) = -\dot{q}^T[K_v] \dot{q}
\]

- Since \(\dot{V}(q, \dot{q})\) can be zero even for non-zero \(q\), \(\dot{V}(q, \dot{q})\) is negative semi-definite

- By LaSalle’s invariance principle (LaSalle and Lefschetz 1961) → equilibrium point \((q, \dot{q}) = 0\) is asymptotically stable.

- Assumption: \(\ddot{q}_d = \dot{q}_d = 0\) → PD control scheme is not proved asymptotically stable for trajectory following!
STABILITY ANALYSIS USING LYAPUNOV’S METHOD

PD control scheme

- \([\dot{M}(q)] - 2[C(q, \dot{q})]\) is skew-symmetric \(\rightarrow\) the second quadratic form is zero, and
  \[
  \dot{V}(q, \dot{q}) = -\dot{q}^T [K_v] \dot{q}
  \]
- Since \(\dot{V}(q, \dot{q})\) can be zero even for non-zero \(q\), \(\dot{V}(q, \dot{q})\) is negative semi-definite
- By LaSalle’s invariance principle (LaSalle and Lefschetz 1961) \(\rightarrow\) equilibrium point \((q, \dot{q}) = 0\) is asymptotically stable.
- Assumption: \(\ddot{q}_d = \dot{q}_d = 0\) \(\rightarrow\) PD control scheme is not proved asymptotically stable for trajectory following!
**Stability analysis using Lyapunov’s method**

**Model-based control schemes**

- Very little about stability can be proved!
- PD and exact gravity cancellation

\[ \tau = -[K_p]q(t) - [K_v]\ddot{q}(t) + G(q) \]

Equilibrium point \((q, \dot{q}) = 0\) is stable!

- Computed torque with *exact cancellation*: Error equation

\[ \ddot{e}_i + K_v \dot{e}_i + K_p e = 0, \quad i = 1, \ldots, n \]

*damped second-order linear ODE’s* → asymptotically stable!

- Stability analysis of non-linear control systems is *unsolved* problem!
- In **Module 10**, Lecture 1, possibility of *chaotic* motions shown for trajectory following.
Stability analysis using Lyapunov’s method

Model-based control schemes

- Very little about stability can be proved!
- PD and exact gravity cancellation

\[ \tau = -[K_p]q(t) - [K_v]\dot{q}(t) + G(q) \]

equilibrium point \((q, \dot{q}) = 0\) is stable!

- Computed torque with exact cancellation: Error equation

\[ \ddot{e}_i + K_v \dot{e}_i + K_p e = 0, \quad i = 1, \ldots, n \]

*damped second-order linear ODE’s* → asymptotically stable!

- Stability analysis of non-linear control systems is unsolved problem!
- In Module 10, Lecture 1, possibility of chaotic motions shown for trajectory following.
Stability analysis using Lyapunov’s method

Model-based control schemes

- Very little about stability can be proved!
- PD and exact gravity cancellation

\[ \tau = -[K_p]q(t) - [K_v]\dot{q}(t) + G(q) \]

equilibrium point \((q, \dot{q}) = 0\) is stable!

- Computed torque with exact cancellation: Error equation

\[ \ddot{e}_i + K_v \dot{e}_i + K_p e = 0, \quad i = 1, \ldots, n \]

*damped second-order linear ODE’s* → asymptotically stable!

- Stability analysis of non-linear control systems is *unsolved* problem!
- In Module 10, Lecture 1, possibility of chaotic motions shown for
  trajectory following.
STABILITY ANALYSIS USING LYAPUNOV’S METHOD
MODEL-BASED CONTROL SCHEMES

- Very little about stability can be proved!
- PD and exact gravity cancellation
  
  \[ \tau = -[K_p]q(t) - [K_v]\dot{q}(t) + G(q) \]

  equilibrium point \((q, \dot{q}) = 0\) is stable!

- Computed torque with exact cancellation: Error equation
  
  \[ \ddot{e}_i + K_v \dot{e}_i + K_p e = 0, \quad i = 1, \ldots, n \]

  damped second-order linear ODE’s → asymptotically stable!

- Stability analysis of non-linear control systems is unsolved problem!

- In Module 10, Lecture 1, possibility of chaotic motions shown for trajectory following.
Stability analysis using Lyapunov’s method
Model-based control schemes

- Very little about stability can be proved!
- PD and exact gravity cancellation

\[ \tau = -[K_p]q(t) - [K_v]\dot{q}(t) + G(q) \]

equilibrium point \((q, \dot{q}) = 0\) is stable!
- Computed torque with exact cancellation: Error equation

\[ \ddot{e}_i + K_v \dot{e}_i + K_p e = 0, \quad i = 1, \ldots, n \]

Damped second-order linear ODE’s \(\rightarrow\) asymptotically stable!
- Stability analysis of non-linear control systems is unsolved problem!
- In Module 10, Lecture 1, possibility of chaotic motions shown for trajectory following.
Lack of knowledge of model parameters
- No "exact" cancellation and difficult to predict evolution of error $e(t)$.
- Model parameters can be obtained using adaptive control schemes (see Craig (1988), Ortega and Spong(1989) for more on adaptive control schemes for robots).

Mathematical notion of controllability of a system.
- A system $\dot{X} = f(X)$ is said to be controllable if it is possible to transfer the system from any initial state $X(0)$ to any desired state $X(t_f)$ in finite time $t_f$ by the application of the control input $u(t)$.
- In a linear system ($n$ state variables and $m$ inputs)

$$\dot{X} = [F]X + [G]u$$

the system is controllable if the controllability matrix

$$[Q_c] = \left[ [G][F][G][F]^2G|...|[F]^{n-1}[G] \right]$$

has rank $n$.
- A non-linear robot is not controllable at a singularity!
Lack of knowledge of model parameters
- No “exact” cancellation and difficult to predict evolution of error $e(t)$.
- Model parameters can be obtained using adaptive control schemes (see Craig (1988), Ortega and Spong(1989) for more on adaptive control schemes for robots).

Mathematical notion of controllability of a system.
- A system $\dot{X} = f(X)$ is said to be controllable if it is possible to transfer the system from any initial state $X(0)$ to any desired state $X(t_f)$ in finite time $t_f$ by the application of the control input $u(t)$.
- In a linear system ($n$ state variables and $m$ inputs)

$$\dot{X} = [F]X + [G]u$$

the system is controllable if the controllability matrix

$$[Q_c] = [ [G][F][G][F]^2G|...|[F]^{n-1}[G] ]$$

has rank $n$.
- A non-linear robot is not controllable at a singularity!
OUTLINE

1 CONTENTS
2 LECTURE 1
   • Motion planning
3 LECTURE 2
   • Control of a single link
4 LECTURE 3
   • Control of a multi-link serial manipulator
5 LECTURE 4*
   • Control of constrained and parallel manipulator
   • Cartesian control of serial manipulators
6 LECTURE 5*
   • Force control of manipulators
   • Hybrid position/force control of manipulators
7 LECTURE 6*
   • Advanced topics in non-linear control of manipulators

8 MODULE 7 – ADDITIONAL MATERIAL
   • Problems, References and Suggested Reading
Module 7 – Additional Material

- Exercise Problems
- References & Suggested Reading