Lecturer: Prof. Sethu Vijayakumar

[Course URL: http://www.inf.ed.ac.uk/teaching/courses/rlsc]

Monday 10:00-11:50: 3.2, Lister Learning and Teaching Centre
Thursday 10:00-11:50: G.06, 50 George Square
Scoring and Grades
30% of marks are allocated to the homework assignment, 10% to the class presentation and 60% to the final exam.

Homework Assignment
They are individual assignments and have to be done individually !! It will involve a combination of software implementation, some statistical analysis and written analytical homework.

Class Presentation
You will be asked to review a topic and make a short presentation in class in addition to writing a short report. The topic will be allocated from a predefined list on motor control and the ‘seed’ paper will be provided to you. You are free to use your discretion in looking for additional materials to facilitate your understanding and presentation.

Class Participation
... is equally important as homeworks and exams in determining your final grade.
Lecture I – Introduction & Basics

Contents:

• The *What & Why* of Machine Learning
• Why do we need ML for control?
What is Machine Learning?

Definition:
- Process by which a computer algorithm finds a solution to a problem

  Here ...
  - ‘machine’ = computer
  - ‘learning’ = finding a good solution

Example:
- Learn to drive a car
- Play chess / backgammon
- Classify handwritten characters
Uses of ‘learning’

- **Knowledge extraction.**
  - Find a simpler model that explains data and hence, have an *explanation* about the data generation process.

- **Compression.**
  - Represent the generated data by the underlying principle and hence, use *less memory* to store and process.

- **Prediction.**
  - Predict future data based on past fits.
Classification of Machine Learning

- **Based on level of feedback**
  - **Supervised** – True output provided
  - **Reinforcement** – Only indirect output provided (reward/punishment)
  - **Unsupervised** – No feedback & no output

- **Based on type of output**
  - **Concept Learning** – Binary output based on +ve/-ve examples
  - **Classification** – Classifying into one among many classes
  - **Regression** – Numeric, ordered output

- **Based on sampling**
  - **Independent (iid) samples**
  - **Dependent samples**
Computational Motor Control

**PLAN**

- **Controller**
  - Motor Command
  - Efference Copy
  - Estimated State
  - Sensory Data

- **Estimator**
  - Sensory Data
  - Noise
  - Estimated State

- **Biomechanical Plant**
  - State
  - Noise

- **Sensory Apparatus**
  - State
  - Noise
Biomimetic robotics

How can we get robots to make fast, accurate, adaptive movements while being compliant?

Low stiffness, give-in, low gains

Let us look at the apparent conflict between accuracy and safety!
Control: basics

| θ      | Desired robot posture       |
| ~θ     | Actual robot posture        |
| u      | Command (torque)            |

- θ: Desired robot posture
- ~θ: Actual robot posture
- u: Command (torque)

PID: Proportional Integral Control
Problem: we know that increasing $u$ (the voltage to the motor) increases $\theta$ (the position), but it is difficult to predict by how much.

$\Rightarrow$ Need for a PID control loop
Control: basics (PID Control)

\[ e = \tilde{\theta} - \theta \]

\[ u = K_p \cdot e + K_i \cdot \int e \cdot dt + K_d \cdot \frac{\partial e}{\partial t} \]

- **\( \theta \)**: Desired robot posture
- **\( \tilde{\theta} \)**: Actual robot posture
- **\( u \)**: Command (torque)
- **\( e \)**: Error

K<sub>p</sub> = Proportional gain
K<sub>i</sub> = Integral gain
K<sub>d</sub> = Derivative gain
Proportional Control

- Set the torque \( u \) proportional to the position error

\[
u = K_p \cdot e
\]

Low gain?  
High gain?

\( \theta \quad \tilde{\theta} \quad \text{Desired Output} \)

\( t \)
Proportional Control: low gain

- Set the torque \( u \) proportional to the position error

\[ u = K_p \cdot e \]

Problem: Long time to desired state
Proportional Control: high gain

- Set the torque \( u \) proportional to the position error

\[ u = K_p \cdot e \]

Problem: Overshoot
Steady state error

- Set the torque \((u)\) proportional to the position error

\[ u = K_p \cdot e \]

Problem: State does not converge to desired output

Solution: Integral or derivative control
PID control

\[ u = K_p \cdot e + K_i \cdot \int e \cdot dt + K_d \cdot \frac{\partial e}{\partial t} \]

- Without derivative action
- With derivative action, the controller output is proportional to the rate of change of the measurement or error: braking effect.
- Adds damping → brakes the dynamics: reduces overshoot and tends to reduce the settling time.
- Problem: if too large, will slow down the dynamics, and increase the rising and settling time.
PID control summary

- A proportional controller ($K_p$) will have the effect of reducing the rise time and will reduce, but never eliminate, the steady-state error.
- An integral control ($K_i$) will have the effect of eliminating the steady-state error, but it may make the transient response worse.
- A derivative control ($K_d$) will have the effect of increasing the stability of the system, reducing the overshoot, and improving the transient response.

<table>
<thead>
<tr>
<th>Gain</th>
<th>Rise Time</th>
<th>Overshoot</th>
<th>Settling Time</th>
<th>S-S Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$K_p$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Small Change</td>
<td>Decrease</td>
</tr>
<tr>
<td>$K_i$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Increase</td>
<td>Eliminate</td>
</tr>
<tr>
<td>$K_d$</td>
<td>Small increase</td>
<td>Decrease</td>
<td>Decrease (might increase)</td>
<td>Small Change</td>
</tr>
</tbody>
</table>
A feedforward model (also called ‘internal models’, ‘inverse dynamics’) is essential to make fast movements with high compliance—low stiffness (why?)

\[ u_{fb} = k_p (x_{cur} - x_{des}) + k_d (\dot{x}_{cur} - \dot{x}_{des}) \]
Various Types of Internal Models

Motor Command Generation

Sensory Feedback Generation

State Transition

[task, state, context] → motor command

[state, motor command, context] → sensory feedback

[previous state, motor command, context] → state
Evidence for use and adaptation of internal models in humans

Manipulandum with force fields

Start
Inverse Dynamics: 2DOF arm

\[ \tau_1 = (I_1 + m_1 L^2_{G_1} + I_2 + m_2 L^2_{G_2} + m_2 L^2_{l_1}) \ddot{\theta}_1 + (I_2 + m_2 L^2_{G_2}) \ddot{\theta}_2 + (2m_2 L_1 L^2_{G_2} \ddot{\theta}_1 + m_2 L_1 L^2_{G_2} \ddot{\theta}_2) \cos \theta_2 - (m_1 L_1 L^2_{G_2} \ddot{\theta}_2^2 + m_1 L_1 L_{G_2} \dot{\theta}_1 \dot{\theta}_2) \sin \theta_2 - (m_1 L_{G_1} + m_2 L_1) g \sin \theta_1 - m_2 L_{G_1} g \sin \theta_1 \cos \theta_2 - m_2 L_{G_2} g \sin \theta_2 \cos \theta_1 \]

\[ \tau_2 = (I_2 + m_2 L^2_{G_2}) \ddot{\theta}_1 + (I_2 + m_2 L^2_{G_2}) \ddot{\theta}_2 + m_2 L_1 L_{G_2} \ddot{\theta}_1 \cos \theta_2 + m_2 L_1 L_{G_2} \dot{\theta}_2 \sin \theta_2 - m_2 L_{G_2} g \sin \theta_1 \cos \theta_2 - m_2 L_{G_2} g \sin \theta_2 \cos \theta_1 \]

Inverse Dynamics derived based on equations of motion and rigid body dynamics assumptions.

Can be used as a forward model and minor deviations corrected through feedback control.
Feedforward model for a 7DOF robot

- Inverse dynamics of a 7DOF anthropomorphic robot arm

\[ \tau = f(\theta, \dot{\theta}, \ddot{\theta}) \]

\[ f : \mathbb{R}^{21} \rightarrow \mathbb{R}^{7} \]

SARCOS dexterous arm
Learning Internal Models or Control Policies is essentially performing function approximation.

Supervised Learning
Adaptive Control
Dynamic Programming
Reinforcement Learning

\[ \tau = f(\theta, \dot{\theta}, \ddot{\theta}) \]
Online learning with Robot Arm

Reaching & pole balancing

Learning Trials

Trial 1
Adaptation to Changed Dynamics

Learning new dynamics online

To demonstrate on-line adaptation, the tennis ball at the top of the pole can be removed to change the pole's physical properties...

Adaptation in Humanoid

\[ f : \mathbb{R}^{90} \rightarrow \mathbb{R}^{30} \]
Biological vs. Biomimetic motor control

- **Biological motor control:**
  - Has to deal with huge sensorimotor delays ~200ms
  - And yet be able to react *fast* with relatively *low gains*
  - *Adapt* to changed dynamics and variable loads
    ...case for *adaptive internal models*

- **Biomimetic systems:**
  - Have to operate with relatively *low feedback gains in order to be compliant* [not stiff, but give-in]
  - Are highly non-linear systems that are *hard to model analytically*
  - Yet has to react *fast* !!
  - *Adapt* to wear and tear, friction/viscous forces and changing loads
    ...case for *learning feedforward control*
Next: Linear Algebra revisited

- LWPR (for the ambitious)...
  - http://ipab.inf.ed.ac.uk/slmc/research/software-lwpr