Learning Low-level Motion Trajectory

Jane Li

Assistant Professor
Mechanical Engineering Department, Robotic Engineering Program
Worcester Polytechnic Institute
Quiz (10 point)

• (4 pts) List the four levels of a robot knowledge hierarchy

• (2 pts) Describe a situation that direct mapping cannot be applied to learning from demonstration

• (4 pts) What are the limitation of kinesthetic teaching?
Robot’s knowledge hierarchy
Mapping from Teacher to Learner

• Direct mapping
  • No correspondence problem
  • Demonstration is recorded in robot’s sensing states

• Not available option to all systems
  • Complex, coordinated motion on high degree of freedom
  • Controlling the robot physically may not be natural
Kinesthetic teaching using force control
Reward shaping and occasional bias
Reward shaping

• How to use Reward signals?
  • Initially, reinforce tendency to correct behavior
  • Gradually, reward more difficult elements of the task \( \rightarrow \) **shaping**

• Significantly more efficiently compare to feedback-only approach
Occasional Bias

• Instead of directly controlling all the agent’s actions
  • Human teacher occasionally bias the action selection

• Advantages?
  • Human doesn’t have to know all about how to perform the task
  • No need for undivided attention from human teacher
Low-level Skill Learning
Overview

• Low-level motion learning
  • Skills, motor skills, primitive actions, low-level motions ... 
• Goal
  • Build an accurate model of a primitive action such that it can generalize across a variety of domain specific tasks
Methods

• Dynamic movement primitives (DMPs)
  • Deterministic – learn from individual demo
  • More recent variety
    • Probabilistic DMP – Integration of sensing uncertainty

• Other probabilistic modeling methods
  • HMMs, GMR - Learn from multiple demos
State spaces for motion learning

• An important choice

• Possible frames
  • Joint space frame
  • Task space frame
  • Object-directed frame

• How to make the choice?
  • Generalization
More than kinematic motions

• Choice is also biased by the capabilities of the hardware

• Learning motion in the sensing and actuation space of robots
  • Learning force or compliance profiles
Dynamic Movement Primitives (DMPs)

- Inspiration of motor primitives
- Algorithm
- Example
- Integration with RL
- Limitation
Motor primitives

- Motor primitives
  - Building blocks (control modules) of complex motions and behaviors

- Properties of motor primitives
  - Exist at different levels of motor hierarchy
  - Limited number, yet flexible task-based combination
Motor primitives

• Kinematic motor primitives
  • Strokes, sub-movements

• Dynamic motor primitives
  • Static force filed, muscle and joint torque synergies

• Neural motor primitives
  • A neuron assembly (spinal or cortical neuron), central pattern generators (CPGs)
Example of movement primitives
Motor primitives at behavior level

- Mental templates of motions
  - Control policy

- Examples
  - Bell-shape velocity profile in human motion
  - Fitts’ law – relationship of motion accuracy and speed

\[ MT = a + b \cdot ID = a + b \cdot \log_2 \left( \frac{2D}{W} \right) \]
Motion primitives at muscle level

- Co-activation of multiple muscles
Motor Primitives at neural level

- Micro-stimulation of an inter-neuronal region in spinal cord
- Microcircuits are organized into discrete modules, each generating a specific force field
Central pattern generator (CPGs)
How to model goal-directed behaviors of non-linear dynamical system?

- Biological motor control
- Robotics
- Economies
- Traffic predictions
- Etc. ...
Appropriate model for movement primitives

- A generic modeling for \textit{discrete} and \textit{periodic} motions
- Autonomous, time-independent
- Coordinate multiple DOFs, in a stable way
- Model parameters are easy to learn
Appropriate model for movement primitives

• Motion synchronization
  • Phase difference

• Fast to compute
  • Online trajectory modulation

• Temporal and spatial scaling
A damped spring model

\[ \tau \ddot{y} = \alpha_z (\beta_z (g - y) - \dot{y}) + f. \]

- How does the system behave if \( f = 0 \) ?
- Goal of the system?
- How does this system behave if it is critically damped?
Recap: 2\textsuperscript{nd} order dynamic system
System Modeling

• How to model this system?

• Time Domain
  • ODE
  • State space

• Frequency Domain
  • Laplace transform
System Modeling

$$m\ddot{q} + c(\dot{q}) + kq = 0$$

ODE – How to solve?
Characteristic Polynomial

\[ a y'' + b y' + c y = 0, \quad a \neq 0. \]

• Let \( y = e^{rt} \) be a solution,
  \[ y' = r e^{rt} \quad y'' = r^2 e^{rt} \]
  \[ a r^2 e^{rt} + b r e^{rt} + c e^{rt} = 0 \]
  \[ e^{rt} (ar^2 + br + c) = 0 \]
Solutions to Characteristic Equations

\[ ar^2 + br + c = 0 \]

- \( b^2 - 4ac > 0 \) \rightarrow two distinct real roots \( r_1, r_2 \)
- \( b^2 - 4ac < 0 \) \rightarrow two complex conjugate roots \( r = \lambda \pm \mu i \)
- \( b^2 - 4ac = 0 \) \rightarrow one repeated real root \( r \)
Solution to ODE – Case 1

Two distinct real roots: $b^2 - 4ac > 0$ leads to two distinct real roots $r_1, r_2$

- Two distinct real roots:
  
  \[ y_1 = e^{r_1 t} \quad y_2 = e^{r_2 t} \]

- General solution
  
  \[ y = C_1 y_1 + C_2 y_2 = C_1 e^{r_1 t} + C_2 e^{r_2 t}. \]
Solution to ODE – Case 2

- Two complex conjugate roots

\[ r_1 = \lambda + \mu i \quad r_2 = \lambda - \mu i \]

- Euler formula

For any real number \( \theta \),
\[ e^{\theta i} = \cos \theta + i \sin \theta \]

\[ e^{rt} = e^{(\lambda + \mu i)t} = e^{ \lambda t} e^{\mu i t} = e^{ \lambda t} (\cos \mu t + i \sin \mu t) \]
Solution to ODE – Case 2

\[ e^{rt} = e^{(\lambda + \mu i)t} = e^{\lambda t} e^{\mu i t} = e^{\lambda t} (\cos \mu t + i \sin \mu t) \]

\[ y = C_1 e^{\lambda t} (\cos \mu t + i \sin \mu t) + C_2 e^{\lambda t} (\cos \mu t - i \sin \mu t) \]

- General solution

\[ y = C_1 e^{\lambda t} \cos \mu t + C_2 e^{\lambda t} \sin \mu t. \]
Solution to ODE – Case 3

\[ b^2 - 4ac = 0 \quad \rightarrow \quad \text{one repeated real root } r \]

- Repeated real root

\[ r = \frac{-b}{2a} \]

- General solution

\[ y = C_1 e^{rt} + C_2 t e^{rt} \]
When do these solutions go unstable?

\[ b^2 - 4ac > 0 \quad \Rightarrow \quad y = C_1 y_1 + C_2 y_2 = C_1 e^{r_1 t} + C_2 e^{r_2 t}. \]

\[ b^2 - 4ac < 0 \quad \Rightarrow \quad y = C_1 e^{\lambda t} \cos \mu t + C_2 e^{\lambda t} \sin \mu t. \]

\[ b^2 - 4ac = 0 \quad \Rightarrow \quad y = C_1 e^{rt} + C_2 t e^{rt}. \]

Solutions have negative real parts
Behavior of Stable Solution

• Let \( x(t; a) \) be a solution with initial condition \( a \)

• A solution is **stable** if other solutions that start near \( a \) stay close to \( x(t; a) \)

• A solution is **asymptotically stable** if all the nearby solutions converge to this stable solution for large time

\[
x(t; b) \rightarrow x(t; a) \text{ as } t \rightarrow \infty
\]
Behavior of asymptotically stable solutions

\[ m\ddot{q} + c(\dot{q}) + kq = 0 \]

\[ b^2 - 4ac < 0 \]

\[ y = C_1 e^{\lambda t} \cos \mu t + C_2 e^{\lambda t} \sin \mu t. \]
Over, critical, under damped systems

\[ b^2 - 4ac > 0 \]  \quad \text{Over}

\[ b^2 - 4ac < 0 \]  \quad \text{Under}

\[ b^2 - 4ac = 0 \]  \quad \text{Critical}

All asymptotically stable
Autonomous VS forced systems

\[ m\ddot{q} + c(\dot{q}) + kq = 0 \]

\[ m\ddot{q} + c\dot{q} + kq = u \]
Forced System

\[ m\ddot{q} + c\dot{q} + kq = u \]

\[ u(t) = A \sin \omega t \]
Back to DMP model
A damped spring model

\[ \tau \ddot{y} = \alpha_z (\beta_z (g - y) - \dot{y}) + f. \]

- How does the system behave if \( f = 0 \) ?
- Goal of the system?
- How does this system behave if it is critically damped?
The forcing term

\[ \tau \ddot{y} = \alpha_z (\beta_z (g - y) - \dot{y}) + f \]

- How does the system behave if \( f \neq 0 \) ?
- How to set the forcing term such that the system will eventually stabilize at the goal?
- What can the forcing term encode?
The forcing term

\[ \dot{\ddot{y}} = \alpha_z (\beta_z (g - y) - \dot{y}) + f \]

\[ f(t) = \frac{\sum_{i=1}^{N} \Psi_i(t) w_i}{\sum_{i=1}^{N} \Psi_i(t)} \]

- The forcing term preserves the shape of a trajectory
- Fixed basis function
- Adjustable weight
- How to get rid of the explicit time dependence?
Change basis by introducing a canonical system

\[ \frac{\tau \dot{x}}{\dot{\alpha}_x} = x, \]

\[ f(t) \rightarrow f(x) \]

- \( t = 0 \) motion starts
- \( t = \infty \) motion finishes
Reformulation of Basis

\[ f(t) = \frac{\sum_{i=1}^{N} \Psi_i(t) w_i}{\sum_{i=1}^{N} \Psi_i(t)} \]

\[ f(x) = \frac{\sum_{i=1}^{N} \Psi_i(x) w_i}{\sum_{i=1}^{N} \Psi_i(x)} x(g - y_0) \]

- Where

\[ \Psi_i(x) = \exp\left( -\frac{1}{2\sigma_i^2} (x - c_i)^2 \right) \]
Basis and weights

Psi activations

x-domain

Psi activations

t-domain

Weighted summation

Weighted summation

<table>
<thead>
<tr>
<th>activation</th>
<th>weighted summation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.8</td>
</tr>
<tr>
<td>0.8</td>
<td>0.6</td>
</tr>
<tr>
<td>0.6</td>
<td>0.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>activation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
</tr>
<tr>
<td>0.8</td>
</tr>
<tr>
<td>0.6</td>
</tr>
<tr>
<td>0.4</td>
</tr>
</tbody>
</table>

11/12/2018
How to use DMP to learn from a demonstration

- Transformation system:
  \[ \tau \ddot{y} = \alpha_z (\beta_z (g - y) - \dot{y}) + f \]

- Forcing term:
  \[ f_{\text{target}} = \tau^2 \ddot{y}_{\text{demo}} - \alpha_z (\beta_z (g - y_{\text{demo}}) - \tau \dot{y}_{\text{demo}}) \]
How to use DMP to learn from a demonstration

• Estimate the weights using locally weighted regression (LWR)

\[ w_i = \frac{s^T \Gamma_i f_{\text{target}}}{s^T \Gamma_i s} \]

• Where

\[
\begin{align*}
s &= \begin{pmatrix} \xi(1) \\ \xi(2) \\ \vdots \\ \xi(P) \end{pmatrix} \quad \Gamma_i &= \begin{pmatrix} \Psi_i(1) & 0 \\ 0 & \Psi_i(2) \end{pmatrix} \\ f_{\text{target}} &= \begin{pmatrix} f_{\text{target}}(1) \\ f_{\text{target}}(2) \\ \vdots \\ f_{\text{target}}(P) \end{pmatrix}
\end{align*}
\]
Apply DMP on a single DOF
Reading


- Understand how to generate
  - Periodic motion
  - Multi-DOF synchronization
  - Obstacle avoidance
Assignment 9 – Due Nov 21

• Download and exercise with DMP Matlab code
  • [http://www-clmc.usc.edu/software/git/gitweb.cgi?p=matlab/dmp.git;a=summary](http://www-clmc.usc.edu/software/git/gitweb.cgi?p=matlab/dmp.git;a=summary)
  • Click the “snapshot” of master branch to download

• Temporal & spatial scaling (45 pts)
  • Generate a trajectory and learn it as a DMP
  • Use the learned DMP to generate a trajectory twice as slow to the original goal
  • Use the learned DMP to generate a trajectory to a new goal

Reference

• Study wolf’s DMP lecture notes
  • Part 1: https://studywolf.wordpress.com/2013/11/16/dynamic-movement-primitives-part-1-the-basics/
  • Part 2: https://studywolf.wordpress.com/2013/12/05/dynamic-movement-primitives-part-2-controlling-a-system-and-comparison-with-direct-trajectory-control/
  • Part 3: https://studywolf.wordpress.com/2014/03/07/dynamic-movement-primitives-part-3-rhythmic-movements/