Robot Learning by Demonstration

Slides credit: Aalhad Patankar, Sozia Chernova and Aude Billard
Equivalent Terms in the Literature

Robot Programming by Demonstration
Imitation Learning in Robots
Apprenticeship Learning
Robot Learning from Demonstration
Programming robots is hard!

- Huge number of possible tasks
- Unique environmental demands
- Tasks difficult to describe formally
- Expert engineering impractical
Introduction: Why learn from demonstration?

- Natural, expressive way to program
- No expert knowledge required
- Valuable human intuition
- Program new tasks as-needed

How can robots be shown how to perform tasks?
Learning from Human Demonstrations: Principle

- Transfer to the robot skills that took years for the humans to master.
- Human can quickly re-train the robot to adapt to task changes.
- The human teaches by showing how to perform the task.
Overview of learning from demonstration (LfD)

- Learning from Demonstration: Deriving a policy from *examples* provided by a teacher
- Different from reinforcement learning, in which a policy is derived from *experience*, such as exploration of different states and actions in reinforcement learning
What is learning from demonstration (LfD)?

- Policy: a mapping between actions and world state
  - E.g. moving an actuator (action) and the location of a box near the robot (world state)
- Examples: A sequence of state-action pairs that are recorded by some sort of teacher demonstrator
Two phases of LfD

- Gathering examples: the process of recording example data to derive a policy from
- Deriving policies: analyzing examples to determine a policy
Advantages of LfD

- Does not require expert knowledge of domain dynamics, which depends heavily on the accuracy of the world model
- Intuitive, as humans already communicate knowledge in this way
- Demonstration focuses the dataset only to area in the state-space encountered during demonstration
Formal definition

- World consists of states $S$ and actions $A$
- States $Z$ are observable states which are mapped from $S$ to $Z$ by mapping $M$
- A policy $\pi : Z \rightarrow A$ is a selection of actions $A$ based on the observable world states
Design choices: demonstrator

- Choice of demonstrators have big impacts on the algorithms used for derivation of policy
- Can be broken down into who designs the demonstration, and which body executes the demonstration
  - E.g. human designer tele-operating a robot, robot designing and executing demonstration
- Human demonstrators usually used
Design choices: demonstration technique

- Whether policy is derived after all training data is obtained (batch), or is developed incrementally as data becomes available (interactive)
- Problem space continuity: whether states are discretized or continuous
  - Discretized example: states broken as “box on table,” “box held by robot,” “box on floor” etc
  - Continuous example: in same example, using 3D position of robot’s effectors and box throughout actions
- Continuity of problem space has big effects on what algorithms are used in the policy derivation stage
Building the example dataset: correspondence

- Because of differences in the teacher’s sensors and actuators (human eyes, human joints) and the robot’s sensors and actuators, a direct transfer of information from teacher to student is often difficult.
- This issue, called **correspondence**, and can be broken down into two categories:
  - Record mapping: correspondence between teacher’s actions and recorded data
  - Embodiment mapping: correspondence between recorded data and learner’s execution
Building the example dataset: correspondence

- Data acquisition for LfD can be broken down into categories based on correspondence.
- \( I(z,a) \) means identity function (direct mapping), while \( g(z,a) \) is a mapping function used for correspondence.
Teleoperation

- Human operator controls a robot teacher
- Direct record and embodiment mapping, as all recording and execution is done on the student body itself by human operator
- E.g. human controlling a robot’s movements through remote control to teach it to find a box
Shadowing

- Robotic platform shadows human teacher, and recordings are done from robotic platform
- Direct embodiment mapping because robot’s own sensors are used to record data, but record mapping required between human actions and robot demonstration in shadowing step
Sensors on Teacher

- Sensors are placed directly on teaching platform, so record correspondence issues are alleviated.
- Can come with large overhead such as specialized sensors and a customized environment.
External observation

- Sensors external to the body executing the demonstration are used to record data
- Less reliable and less precise, but comes with less overhead
Deriving a policy: mapping function

- Attempts to calculate the underlying function behind the states and actions and generalize over set of training data
- Two major categories: *classification* and *regression*
- Is heavily influenced by demonstration design choices mentioned earlier
Mapping function: Categorization

- Input is categorized into discrete classes and outputs discrete robot actions
- Many algorithms, such as k-Nearest Neighbors, Gaussian Mixture Models, and Bayesian networks are used to perform the classification, depending on the application
- Can be done for low level robot movement (controlling a car in a simulated environment), mid-level motion primitives (teaching a robot to flip an egg), and high level complex actions (ball sorting task)
Mapping function: Regression

- Maps demonstration states to continuous action outputs
- Lazy learning: function approximation is done “on demand” whenever a current observation needs to be mapped at run-time
- At opposite end, all function approximation done prior to run-time
  - No adjustments to policy done at run-time
  - Very computationally expensive
Plans

- Actions are composed of *pre-conditions*, the state that must take place before an action can occur, and *post-conditions*, the state immediately after the action.
- Non state-action information, such as *intentions* and *annotations* can be provided by the teacher to the learner in addition to demonstration data.
Example with plans: clearing a table

- Task: clearing a table
- Pre-programmed actions: pick, drop, search, etc. available to robot
- After demonstration, robot learns how these actions relate to objects and states, and learns mapping between sequence of actions and states

Overview of current research areas

Low-Level Skills
• Trajectories
• Force profiles

High-Level Skills
• Combination of actions
• Speech-directed teaching

Combined with other techniques
• Bootstrap reinforcement learning
• Inverse optimal control

User-studies to assess:
• Interfaces
• Effectiveness of algorithm

Batch learning versus incremental learning

http://www.scholarpedia.org/article/Robot_learning_by_demonstration
Not Just Record and Replay: Generalize!

Recording demonstration via kinesthetic teaching
Correspondence Problem

Demonstrator

\[
(\theta_1, \theta_2, \theta_3)
\]

\[
(\theta_5, \theta_6, \theta_7)
\]

\[
\vec{x} = (x_1, x_2, x_3)
\]

Imitator

\[
(\theta'_1, \theta'_2)
\]

\[
(\theta'_3, \theta'_4)
\]

\[
(\theta'_5, \theta'_6, \theta'_7)
\]

\[
\vec{x}' = (x'_1, x'_2, x'_3)
\]

Establish a correspondence across degrees of freedom when feasible.
The correspondence problem

state-action mapping?
Sensing: Motion capture

Phasespace

Vicon
Motion sensors:

**Pros:**
- Real-time kinematic information
- Solve correspondence problem

**Cons:**
- Require to wear the system
- No haptic information
Sensing: RGB(D) cameras, depth sensors

- Standard RGB cameras
- Stereo: Bumblebee
- **RGB-D**: Microsoft Kinect
- Time of flight: Swiss Ranger
- **LIDAR**: SICK
Sensing: Visual fiducials

AR tags
http://wiki.ros.org/ar_track_alvar

RUNE-129 tags
Sensing: Wearable sensors

SARCOS Sensuit:
- Record 35-DOF poses at 100 Hz

Other wearables:
- Accelerometers
- Pressure sensors
- First-person video
**Haptic devices:**

**Pros:**
- Solve correspondence problem
- Transmit kinematic & haptic information

**Cons:**
- Requires training
- User far from task location

Which interface?
Learning by doing: Teleoperation
Learning by doing: Kinesthetic demonstration
Kinesthetic Teaching:

Pros:
- Solve correspondence problem
- Transmit kinematic & haptic information

Cons:
- Need two hands to teach movements of a few DOFs
Learning by doing: Keyframe demonstration
Kinesthetic Teaching using Tactile Sensing

By contrast, when no model is used, the posture does not adapt.

Tactile corrections:
1) Improve contact.
2) Show posture adaptation.

No Adaptation

Teaching adaptive behavior

Sauser, Argall, Metta and Billard, Autonomous Robots, 2011
Learn a probabilistic mapping $p(\phi, s, \theta)$ between contact signature of the object (normal force $\phi$ and tactile response $s$) and fingers' posture $\theta$. 

Sauser, Argall, Metta and Billard, Autonomous Robots, 2011
Learning by watching: Shadowing
Learning by watching: Simplified mimicry

Object-based

End effector-based
Supplementary information: Speech and critique

Interpreting and grounding natural language commands

Realtime user feedback given to RL system
Learning a task plan: STRIPS-style plans

When I say deliver message
  If Person1 is present
    Give message to Person1
  Otherwise
    If Person2 is present
      Give message to Person2
    Otherwise
      Report message delivery failure
  Before
  Before
  Goto home

[Rybski et al. 2007]
Learning a task plan: STRIPS-style plans

>> When I say deliver message
>> If Person1 is present
>>   Give message to Person1
>> Otherwise
>>   If Person2 is present
>>     Give message to Person2
>> Otherwise
>>   Report message delivery failure
>> Before
>> Before
>> Goto home

Demonstrated behavior

[Rybski et al. 2007]
Learning a task plan: Finite state automata

Unsegmented demonstrations of multi-step tasks

Finite-state task representation

[Niekum et al. 2013]
Learning a task plan: Finite state automata

[Niekum et al. 2013]
Ikea Assembly:
Overview of the iterative learning from demonstration framework
Learning a task plan: Finite state automata

Skills

\[ x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5 \rightarrow x_6 \rightarrow x_7 \rightarrow x_8 \]

Observations

\[ y_1 \rightarrow y_2 \rightarrow y_3 \rightarrow y_4 \rightarrow y_5 \rightarrow y_6 \rightarrow y_7 \rightarrow y_8 \]

Standard Hidden Markov Model

[Niekum et al. 2013]
Learning a task plan: Finite state automata

\[ y_t^{(i)} = \sum_{j=1}^{r} A_{j,z_t^{(i)}} y_{t-j}^{(i)} + e_t^{(i)}(z_t^{(i)}) \]

Skills

Observations

Autoregressive Hidden Markov Model

[ Niekum et al. 2013 ]
Learning a task plan: Finite state automata

\[ y^{(i)}_t = \sum_{j=1}^{r} A_{j, z^{(i)}_t} y^{(i)}_{t-j} + e^{(i)}_t(z^{(i)}_t) \]

Skills

6 \rightarrow 6 \rightarrow 3 \rightarrow 1 \rightarrow 1 \rightarrow 3 \rightarrow 11 \rightarrow 10

Observations

y_1 \rightarrow y_2 \rightarrow y_3 \rightarrow y_4 \rightarrow y_5 \rightarrow y_6 \rightarrow y_7 \rightarrow y_8

Autoregressive Hidden Markov Model

[Niekum et al. 2013]
Learning a task plan: Finite state automata

\[ y_{t}^{(i)} = \sum_{j=1}^{r} A_{j, z_{t}^{(i)}} y_{t-j}^{(i)} + e_{t}^{(i)}(z_{t}^{(i)}) \]

[unknown number!]

Skills

Observations

Beta Process Autoregressive Hidden Markov Model

[Niekum et al. 2013]
Learning a task plan: Finite state automata

[Niekum et al. 2013]
Interactive corrections

[Niekum et al. 2013]
Replay with corrections: missed grasp

[Niekum et al. 2013]
Replay with corrections: too far away

[Niekum et al. 2013]
Replay with corrections: full run

[Niekum et al. 2013]
Learning object affordances: Action + object

Can we learn to recognize actions based on their effects on objects?

Object features: Color, shape, size
Actions: Grasp, tap, touch
Effects: Velocity, contact, object-hand distance

[Lopes et al. 2007]
Learning object affordances: Articulation models

Prismatic - drawer
Revolute - cabinet
Gaussian process - garage door
Infer full kinematic chain via Bayes net

[Sturm et al. 2011]
Combining LfD with Reinforcement Learning

- Use human demonstrations to initialize the parameters of the controller.
- One cannot use directly demonstration as the dynamics of robot differ from human dynamics.
- User reinforcement learning to search for solutions nearby the demonstrations.

Kormushev et al, Int. Conf. on Robotics and Intelligent Systems, 2010
Refinement through verbal interaction

- Robot has initial set of reaching skills
- Robot provided with a dialogue system to query the teacher
- Teacher modifies the controller through verbal guidance

Cakmak & Thomaz, Intern. Conf. on Human-Robot Interaction, 2012
Future directions

- **Feature selection**
  - selecting too many features is computationally expensive and can “confuse” learning process, while too few features might lead to insufficient data for policy inference
  - What is an intuitive way to select the right features?

- **Including temporal data**
  - Currently, most algorithms discard temporal data
  - Repetitive tasks become difficult to sequentialize
  - Actions that have no perceivable effect on the states are difficult to learn from
  - Temporal data could alleviate both these issues
Future directions

- Multi-robot demonstration learning
  - Both agents could request advice from human teacher or provide demonstrations for one another
- Refined evaluation metrics
  - Currently, LfD projects are highly domain and task specific
  - Field lacks a cross-domain standard for evaluating performance
Future directions

- Multiple tasks, libraries of skills, skill hierarchies
- Parameterized skills (pick up any object, hit ball to any location, etc.)
- ‘Common sense’ understanding of physics, actions, etc.
- Bridge the gap between low-level observations and high-level concepts
- Novel ways to leverage human insight (natural language + demonstrations, learning to ‘play’, etc.)


