Efficient Reinforcement Learning for Motor Control

Marc Peter Deisenroth

joint work with Carl Edward Rasmussen

10th International PhD Workshop on Systems and Control
Hluboká nad Vltavou, Czech Republic
September 23, 2009
Why learning for control?

- machines can execute very complicated control commands
Why learning for control?

Figure: Kasparov (left) vs. DeepBlue (right), 1996/1997

but sometimes control is not so easy
Why learning for control?

but sometimes control is not so easy

→ make machines solve control tasks themselves (learning)

**Figure:** Kasparov (left) vs. DeepBlue (right), 1996/1997

with permission from http://www.chesshistory.com
Challenges in learning control

- machines typically require expert knowledge or many \(10^x, x \geq 2\) trials
  - can be a) expensive, b) not available, c) infeasible
Challenges in learning control

- Machines typically require expert knowledge or many ($10^x$, $x \geq 2$) trials
  - can be a) expensive, b) not available, c) infeasible

- Data-efficient

- Make machines learn from “scratch”
  - only general assumptions, no expert knowledge
Challenges in learning control

- machines typically require expert knowledge or many \( (10^x, x \geq 2) \) trials
  - can be a) expensive, b) not available, c) infeasible

- data-efficient

- make machines learn from “scratch”
  - only general assumptions, no expert knowledge

objective:
- find a strategy of solving a problem that satisfies these constraints
Task learning as an optimal control problem

- find a **policy**/strategy $\pi$ that yields low **expected long-term cost**

$$V^\pi(x_0) = \sum_{t=0}^{T} \mathbb{E}[c(x_t)]$$

of following policy $\pi$ for $T$ time steps (starting from $x_0$)

- $c(x_t)$: immediate/instantaneous cost function
Task learning as an optimal control problem

- find a policy/strategy $\pi$ that yields low expected long-term cost

$$V^\pi(x_0) = \sum_{t=0}^{T} \mathbb{E}[c(x_t)]$$

of following policy $\pi$ for $T$ time steps (starting from $x_0$)

- $c(x_t)$: immediate/instantaneous cost function

challenges:
- data-efficient solution (few trials)
- unknown system function
- no expert knowledge available
Task learning as an optimal control problem

- find a policy/strategy $\pi$ that yields low expected long-term cost
  
  \[
  V^\pi(x_0) = \sum_{t=0}^{T} \mathbb{E}[c(x_t)]
  \]

  of following policy $\pi$ for $T$ time steps (starting from $x_0$)

- $c(x_t)$: immediate/instantaneous cost function

challenges:

- data-efficient solution (few trials)
- unknown system function
- no expert knowledge available

two possible approaches to get $V^\pi$:

- model free $\rightarrow$ sample states and controls from real system
- model based $\rightarrow$ find a model of the system function; internal simulation
General (model-based) setup: interaction and simulation

Two phases:

- **interaction**: internal model is refined using experience from interacting with the real system.

- **simulation**: internal model is used to simulate consequences of actions in the real system, policy is refined.
General (model-based) setup: interaction and simulation

Two phases:
- **Interaction**: Internal model is refined using experience from interacting with the real system.
- **Simulation**: Internal model is used to simulate consequences of actions in the real system, policy is refined.

→ **Problem**: Model bias!
How do we get a good model?

- system identification?
How do we get a good model?

- system identification?
- extract “shape” of the system function from data with high-level assumptions (e.g. smoothness)
- model what we know and what we don’t
How do we get a good model?

- system identification?
- extract “shape” of the system function from data with high-level assumptions (e.g. smoothness)
- model what we know and what we don’t
How do we get a good model?

- system identification?
- extract “shape” of the system function from data with high-level assumptions (e.g. smoothness)
- model what we know and what we don’t

Here: Gaussian processes to find a model of the system function
Pictorial introduction to Gaussian process regression
Pictorial introduction to Gaussian process regression
Pictorial introduction to Gaussian process regression
Pictorial introduction to Gaussian process regression
Evaluation of the value function

- the GP gives us one-step transition probabilities $p(x_{t+1}|x_t)$, but we need

$$V^\pi(x_0) = \sum_{t=0}^{T} \mathbb{E}_{x_t}[c(x_t)]$$
Evaluation of the value function

- the GP gives us one-step transition probabilities $p(x_{t+1} \mid x_t)$, but we need

$$V^\pi(x_0) = \sum_{t=0}^{T} E_{x_t}[c(x_t)]$$

- cascade predictions to get $p(x_1), p(x_2), \ldots, p(x_T)$
- compute $E_{x_t}[c(x_t)]$
- add them together
Policy refinement

- expected long-term cost (value function)

\[
V^\pi(x_0) = \sum_{t=0}^{T} \mathbb{E}[c(x_t)]
\]

can be evaluated analytically using approximate Bayesian inference

- compute derivative of \(V^\pi(x_0)\) with respect to policy parameters

- iterative gradient-based method to optimize policy parameters
  → policy search
High-level algorithm

1. **init:** set policy to random
2. **loop**
3. apply policy to the real system
4. learn GP model for system function
5. **loop**
6. simulate system with policy $\pi$
7. compute value function $V^{\pi}$ for current policy
8. improve policy
9. **end loop**
10. **end loop**
Results
Wrap-up

- **data-efficient** artificial learning for control problems
- no expert knowledge
- **probabilistic model** for coherent representation of uncertainty
- explicit incorporation of uncertainty into prediction and decision-making
- gradient-based policy search
- works in simulation and hardware

http://mlg.eng.cam.ac.uk/marc