Optimal Control: Introduction and Overview

Jonathan Wilder Lavington

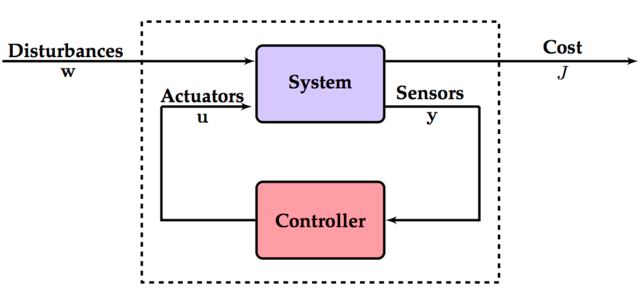
July 8, 2020

University of British Columbia,
Department of Computer Science

What is Optimal Control?

We define Optimal Control as the active manipulation of dynamical systems to achieve a given engineering goal.

Core Idea: Closed Loop Feed Back Control



Simple Control Example

Temperature Control

Create a control policy to keep the internal temperature of a freezer at a reference temperature.

Governing ODE

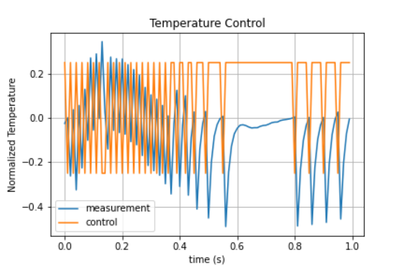
$$f(x,t,\epsilon) = \alpha x + f(t) + \epsilon \quad (1)$$

Control Problem

$$\hat{f}(x,t,\epsilon,c) = f(x,t,\epsilon) + c \quad (2)$$

$$\hat{f}^* = \min_{c \in C} ||f(x,t,\epsilon,c)|| \quad (3)$$

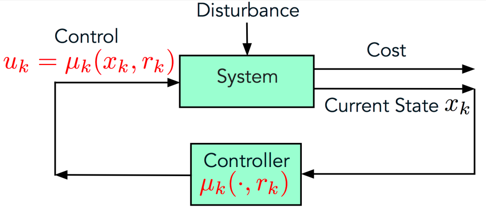
Bang Bang Control



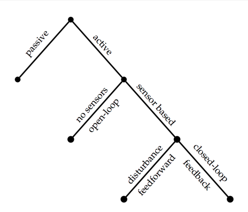
What will we focus on?

This term will focus on Closed Loop Feedback Control in both discrete and continuous systems. We will also for the most part focus on leaning a parameterized control policy.

Parameterized Control

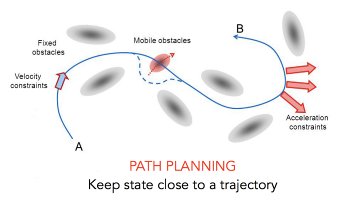


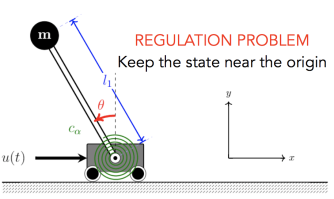
Forms of Control



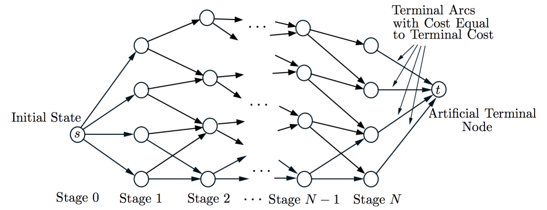
Optimal Control Problem Classes

Continuous Control





Discrete Control



What are some applications

1. **Fluid dynamics:** Improve drag reduction, lift increase, and noise reduction in aeronautics.
2. **Finance:** Maximize profit given a level of risk tolerance.
3. **Epidemiology:** Effectively suppress a disease with constraints of sensing (blood samples, clinics, etc.) and actuation (vaccines, bed nets, etc.).
4. **Industry:** Increasing productivity subject to constraints like labor and work safety laws, and enviro impact.
5. **Autonomy and robotics:** self-driving cars and autonomous robots is to achieve a task while interacting safely with a complex environment, including cooperating with human agents.

Why should we care about optimal control?

It Connects us with powerful tools from different fields

- Approximate Dynamic Programming
- Reinforcement Learning
- Model Predictive Control
- Online Control / System Identification

Areas that have been researched depending on focus:

- Do you need algorithmic guarantees?
- What can you approximate safely?
- How quickly do you need to produce control online?
- Do you have access to a model?
- Can you make assumptions about the dynamical system?

Why should we care about Optimal Control

THE UNIVERSITY OF BRITISH COLUMBIA

Learning from imperfect experts

Sometimes we can reduce an RL/IL problem to something simpler, like an online learning problem.

- "A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning"
- "Reinforcement and Imitation Learning via Interactive No-Regret Learning"
- "Truncated Horizon Policy Search: Combining Reinforcement Learning Imitation Learning"

Simple efficient algorithms (DAgger)

Initialize $\mathcal{D} \leftarrow \emptyset$.
Initialize $\hat{\pi}_1$ to any policy in Π.
for $i = 1$ **to** N **do**
  Let $\pi_i = \beta_i \pi^* + (1 - \beta_i)\hat{\pi}_i$.
  Sample T-step trajectories using π_i.
  Get dataset $\mathcal{D}_i = \{(s, \pi^*(s))\}$ of visited states by π_i and actions given by expert.
  Aggregate datasets: $\mathcal{D} \leftarrow \mathcal{D} \bigcup \mathcal{D}_i$.
  Train classifier $\hat{\pi}_{i+1}$ on $\mathcal{D}$.
end for
Return best $\hat{\pi}_i$ on validation.

Algorithm 3.1: DAGGER Algorithm.

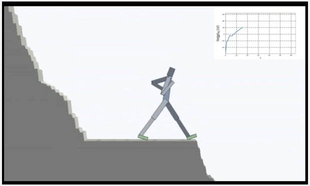
Why should we care about Optimal Control

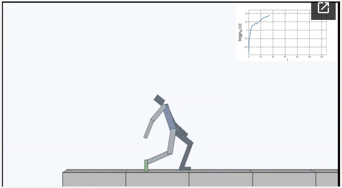
THE UNIVERSITY OF BRITISH COLUMBIA

Learning via some notion of intrinsic stability

If the goal is actually to produce an agent which just needs to "survive" in the environment, then the usual reward mechanisms / deep RL might not be the right tool. (some results from `https://sites.google.com/view/surpriseminimization`)

Emergent behavior from stability seeking algorithms

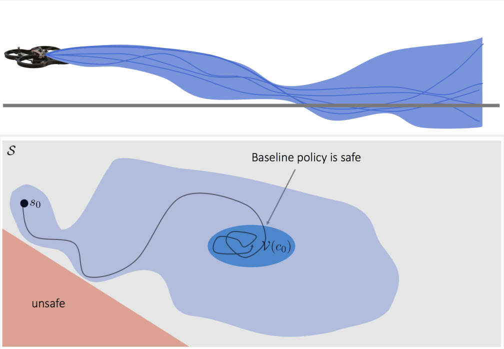




Safe Policy Learning with Model Predictive Control

Planning/MPC often provides guarantees and improved performance over "constrained policy learning" approach.

Staying in safe regions of state-space

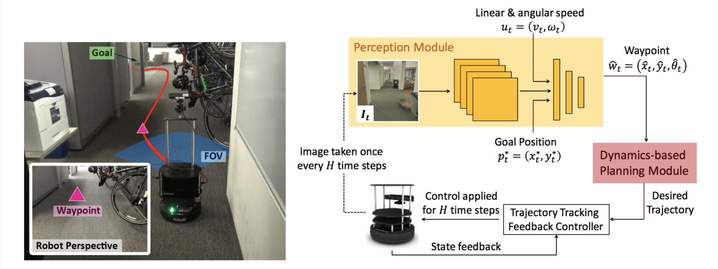


THE UNIVERSITY OF BRITISH COLUMBIA

Exploration and Learning In Novel Environments

When actually interacting with the environment, how do we deal with new information while still maintaining performance?

Perception learning + closed loop feedback control

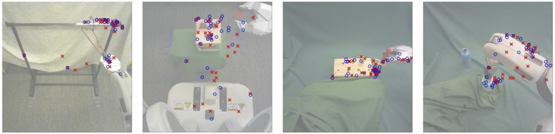


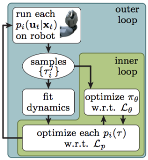
THE UNIVERSITY OF BRITISH COLUMBIA

Model Stacking Stacks versus End-to-End

When the perception problem can be detached from learning the policy, we can take advantage of extremely efficient, low sample complexity control methods, but can we do better ("End-to-End Training of Deep Visuomotor Policies")?

Pipeline Example


(a) hanger (b) cube (c) hammer (d) bottle

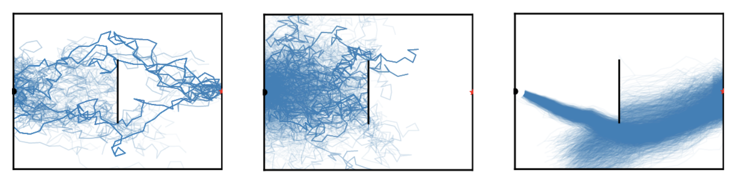


Why should we care about Optimal Control

A deep connection to model based RL

Depending on on your definition of control, many approaches to planning stem from dynamic programming principles: "Probabilistic Planning with Sequential Monte-Carlo Methods".

Multi-model behavior in SAC using fewer samples



Why should we care about Optimal Control

THE UNIVERSITY OF BRITISH COLUMBIA

Improved Algorithmic and Performance Guarantees

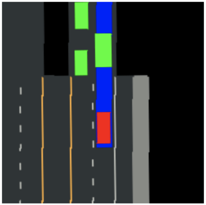
There has been a **huge**, amount of work done in this area. Here is a list of papers by a prominent control researcher:

- “Finite-time Analysis of Approximate Policy Iteration for the Linear Quadratic Regulator”
- “Learning Linear Dynamical Systems with Semi-Parametric Least Squares”
- “Regret Bounds for Robust Adaptive Control of the Linear Quadratic Regulator”
- “Least-Squares Temporal Difference Learning for the Linear Quadratic Regulator”
- “On the Sample Complexity of the Linear Quadratic Regulator”

Efficient Expert Learning in Asymmetric Algorithms

In AV, we can use information such as a top-down view, or a condensed numerical format is used to train models that are not used at test time (from "Learning by Cheating" - here).

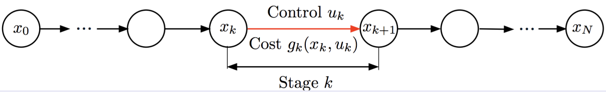
Learning From Asymmetric Information



$$\begin{pmatrix} x_1 & y_1 & z_1 & v_1 & a_1 \\ x_2 & y_2 & z_2 & v_2 & a_2 \\ x_3 & y_3 & z_3 & v_3 & a_3 \\ x_4 & y_4 & z_4 & v_4 & a_4 \\ t_1 & t_2 & t_3 & t_4 & l_d \end{pmatrix}$$

Different Types of Control Problems

Finite Horizon-Deterministic Problems



- Discrete-time system:

$$x_{k+1} = f_k(x_k, u_k), \qquad k = 0, 1, \ldots, N-1$$

  where x_k: State, u_k: Control chosen from some constraint set $U_k(x_k)$

- Cost function:

$$g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

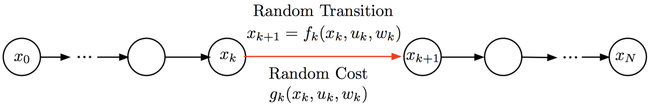
- For given initial state x_0, minimize over control sequences $\{u_0, \ldots, u_{N-1}\}$

$$J(x_0; u_0, \ldots, u_{N-1}) = g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k)$$

- Control sequences correspond to paths from start node to end node in the graph
- Optimal cost function $J^*(x_0) = \min_{\substack{u_k \in U_k(x_k) \\ k=0,\ldots,N-1}} J(x_0; u_0, \ldots, u_{N-1})$

THE UNIVERSITY OF BRITISH COLUMBIA

Finite Horizon-Stochastic Problems



- Stochasticity in the form of a random "disturbance" w_k (e.g., physical noise, market uncertainties, demand for inventory, unpredictable breakdowns, etc)
- Cost function:

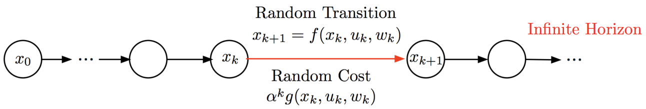
$$E\left\{g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, u_k, w_k)\right\}$$

- Policies $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$, where μ_k is a "closed-loop control law" or "feedback policy"/a function of x_k. Specifies control $u_k = \mu_k(x_k)$ to apply when at x_k.
- For given initial state x_0, minimize over all $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$ the cost

$$J_\pi(x_0) = E\left\{g_N(x_N) + \sum_{k=0}^{N-1} g_k(x_k, \mu_k(x_k), w_k)\right\}$$

- Optimal cost function $J^*(x_0) = \min_\pi J_\pi(x_0)$

Infinite Horizon-Discounted Problems



Infinite number of stages, and stationary system and cost

- System $x_{k+1} = f(x_k, u_k, w_k)$ with state, control, and random disturbance.
- Policies $\pi = \{\mu_0, \mu_1, \ldots\}$ with $\mu_k(x) \in U(x)$ for all x and k.
- Optimal cost function $J^*(x_0) = \min_\pi J_\pi(x_0)$ satisfies Bellman's equation

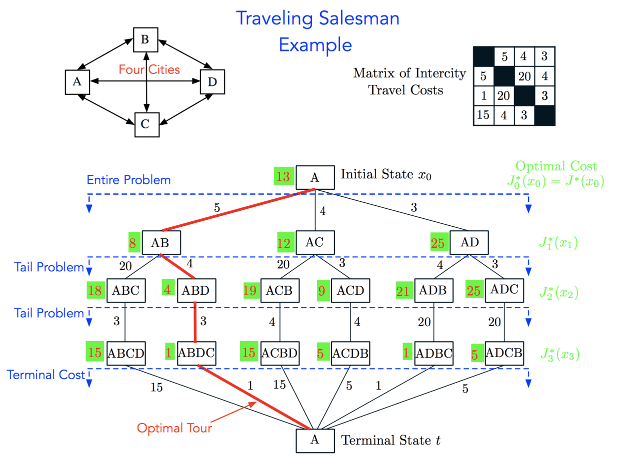
$$J^*(x) = \min_{u \in U(x)} E\Big\{ g(x, u, w) + \alpha J^*\big(f(x, u, w)\big) \Big\}$$

- Optimal policy: Applies at x the minimizing u above, regardless of stage k.
- When there are finitely many states, $i = 1, \ldots, n$, Bellman's equation is written in terms of the $i \to j$ transition probabilities $p_{ij}(u)$ as

$$J^*(i) = \min_{u \in U(i)} \sum_{i=1}^{n} p_{ij}(u)\big(g(i, u, j) + \alpha J^*(j)\big)$$

- Approximation possibility: Use $\tilde{J}$ in place of J^*, and approximate $E\{\cdot\}$ and $\min_u$

Stochastic Shortest Path Problems

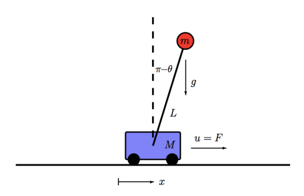


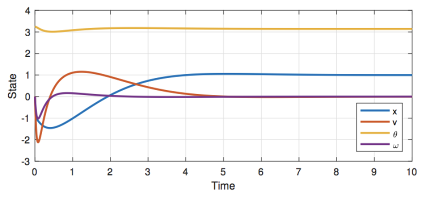
	5	4	3
5		20	4
1	20		3
15	4	3	

Inducing stability in dynamical systems

Typically this requires finding the minimum cost control to remain within a region of stability with respect to the system dynamics, and (provided the system is linear) the Eigen values of the transition matrix.

Simple Pole-Balancing Example





Optimal Control Overview

Important Algorithm Classes

Different Forms of Approximation

- Approximation in Value space
- Approximation in Policy space
- Approximation in Value space **and** Policy space

Different Algorithm Classes

- Look-ahead algorithms
- Roll-out Algorithms

Why do we care

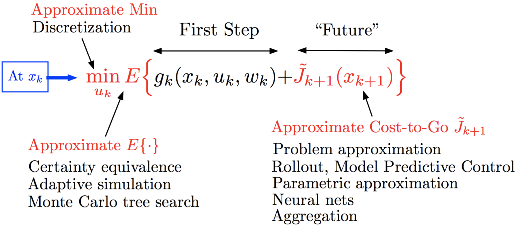
- Simple, efficient algorithms
- Improvement bounds

Approximation in Value Space

THE UNIVERSITY OF BRITISH COLUMBIA

Look ahead algorithms (1-step look ahead)

At state x_k, use $\tilde{J}_{k+1}$ (in place of J^*_{k+1}) to compute a (suboptimal) control



THE THREE APPROXIMATIONS: (They can be designed separately)

- How to construct $\tilde{J}_k$ [an important example is parametric approximation $\tilde{J}_k(x_k, r_k)$ with parameter vector r_k, e.g., neural nets].
- How to simplify $E\{\cdot\}$ operation.
- How to simplify min operation.

Approximation in Value Space

THE UNIVERSITY OF BRITISH COLUMBIA

Look ahead algorithms (k-step look ahead)

First ℓ Steps ⟷ "Future" ⟷

$$\text{At } x_k \rightarrow \min_{u_k,\mu_{k+1},\ldots,\mu_{k+\ell-1}} E\left\{ g_k(x_k,u_k,w_k) + \sum_{m=k+1}^{k+\ell-1} g_m\big(x_m,\mu_m(x_m),w_m\big) + \tilde{J}_{k+\ell}(x_{k+\ell}) \right\}$$

- At state x_k, solve an ℓ-stage version of the DP problem with x_k as the initial state and $\tilde{J}_{k+\ell}$ as the terminal cost function.
- Use the first control of the ℓ-stage policy thus obtained, and discard the others.

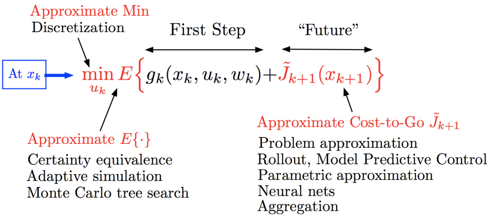
We can view ℓ-step lookahead as a special case of one-step lookahead:

The "effective" one-step lookahead function is the optimal cost function of an $(\ell - 1)$-stage DP problem with terminal cost $\tilde{J}_{k+\ell}$.

Approximation in Value Space

THE UNIVERSITY OF BRITISH COLUMBIA

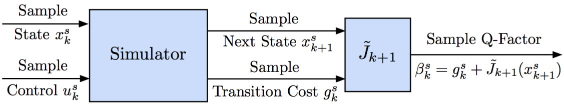
Online vs Offline look ahead algorithms



- Off-line methods: All the functions $\tilde{J}_{k+1}$ are computed for every k, before the control process begins.
- Examples of off-line methods: Neural net and other parametric approximations.
- On-line methods: The values $\tilde{J}_{k+1}(x_{k+1})$ are computed only at the relevant next states x_{k+1}, and are used to compute the control to be applied at the N time steps.
- Examples of on-line methods: Rollout and model predictive control.
- On-line methods are well-suited for on-line replanning (but require more on-line computation).

Reference Point (Policy Gradients)

I only do RL - what are these terms?



- Use the simulator to collect a large number of "representative" samples of state-control-successor states-stage cost quadruplets $(x_k^s, u_k^s, x_{k+1}^s, g_k^s)$, and corresponding sample Q-factors

$$\beta_k^s = g_k^s + \tilde{J}_{k+1}(x_{k+1}^s), \qquad s = 1, \ldots, q$$

- Introduce a parametric family of Q-factors $\tilde{Q}_k(x_k, u_k, r_k)$.
- Determine the parameter vector $\bar{r}_k$ by the least-squares fit

$$\bar{r}_k \in \arg\min_{r_k} \sum_{s=1}^{q} \left(\tilde{Q}_k(x_k^s, u_k^s, r_k) - \beta_k^s\right)^2$$

- Use the policy

$$\tilde{\mu}_k(x_k) \in \arg\min_{u_k \in U_k(x_k)} \tilde{Q}_k(x_k, u_k, \bar{r}_k)$$

The Roll-out Algorithm

First ℓ Steps ⟷ "Future" ⟷

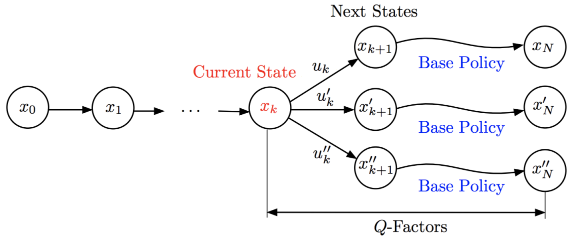
At x_k → $$\min_{u_k, \mu_{k+1}, \ldots, \mu_{k+\ell-1}} E\Big\{ g_k(x_k, u_k, w_k) + \sum_{m=k+1}^{k+\ell-1} g_m\big(x_m, \mu_m(x_m), w_m\big) + \tilde{J}_{k+\ell}(x_{k+\ell}) \Big\}$$

Rollout Control — Lookahead Minimization — Cost of Base Policy

Use the cost of the base/suboptimal policy at the end of ℓ-step lookahead

- Assume a base policy is available and can be simulated.
- The control $\tilde{\mu}_k(x_k)$ of the lookahead policy, can be computed at any x_k. It defines the rollout policy.
- The rollout policy performs better than the base policy. (Intuition: Using optimization in the first ℓ steps instead of using the base policy should work better.)
- In practice rollout performs well, is very reliable, is very simple to implement, can be model-free (particularly in the case $\ell = 1$).
- Rollout in its "standard" form involves simulation and on-line implementation.
- The simulation can be prohibitively expensive (so further approximations may be needed); particularly for stochastic problems and multistep lookahead.

The Roll-out Algorithm (deterministic)



- At state x_k, for every pair (x_k, u_k), $u_k \in U_k(x_k)$, we generate a Q-factor

$$\tilde{Q}_k(x_k, u_k) = g_k(x_k, u_k) + H_{k+1}(f_k(x_k, u_k))$$

  using the base policy [$H_{k+1}(x_{k+1})$ is the base policy cost starting from x_{k+1}].
- We select the control u_k with minimal Q-factor.
- We move to the next state x_{k+1}, and continue.
- Multistep lookahead versions (length of lookahead limited by the branching factor of the lookahead tree).

The Roll-out Algorithm (stochastic)

First ℓ Steps — "Future"

$$\text{At } x_k \rightarrow \min_{u_k,\mu_{k+1},\ldots,\mu_{k+\ell-1}} E\Big\{ g_k(x_k,u_k,w_k) + \sum_{m=k+1}^{k+\ell-1} g_m\big(x_m,\mu_m(x_m),w_m\big) + \tilde{J}_{k+\ell}(x_{k+\ell}) \Big\}$$

Rollout Control — Lookahead Minimization — Cost of Base Policy

- Start with a base policy $\pi = \{\mu_0, \ldots, \mu_{N-1}\}$.
- Let the rollout policy be $\tilde{\pi} = \{\tilde{\mu}_0, \ldots, \tilde{\mu}_{N-1}\}$. Then cost improvement is obtained

$$J_{k,\tilde{\pi}}(x_k) \le J_{k,\pi}(x_k), \qquad \text{for all } x_k \text{ and } k.$$

- This fundamental property carries over to policy iteration, which can be viewed as perpetual rollout:

$$\text{Start Policy} \implies \text{Rollout Policy} \implies \text{Rollout of Rollout Policy} \implies \cdots$$

- Approximate policy iteration (or self-learning): Use of simulation, and approximation in policy and/or value space, to learn sequentially improved policies.
- Many variants: Actor only, critic only, actor-critic, Q-learning methods.

Picking Presenters

Topics

THE UNIVERSITY OF BRITISH COLUMBIA

Presentation List

1. Background and overview (Engineering Perspective)
2. Background and overview (Optimization Perspective)
3. Applied versions of LQR in deep learning (ILQR / Guided Policy Search)
4. Learning Non-linear system dynamics (LQR Sample Complexity / Koopman Theory)
5. Model Predictive Control (Safe-exploration + Tutorial)
6. Learning End to End Visuomotor Policies (high-dim control Under Partial Information)
7. Vision Based Navigation in Novel Environments (high-dim control + exploration)

Topics

THE UNIVERSITY OF BRITISH COLUMBIA

Presentation 1:
Linear Control In Engineering Applications

Read chapter 8 of "Data Driven Science Engineering Machine Learning, Dynamical Systems, and Control" (pg 326-352) from here

Major Topics

- Closed loop feedback control
- Controllability and observability
- Optimal full state control: the linear quadratic regulator
- Optimal full state estimation: the Kalman filter

Presentation 2: Control Theory From RL / Optimization Perspective

Read “Optimal Control Theory” (pg 1-23) from here

Major Topics

- Discrete Control / Dynamic Programming
- Continuous Control / HJB equations
- Pontryagin’s Maximum Principle
- Linear quadratic Guassian
- Duality of optimal control and optimal estimation

Topics

Presentation 3: Applications of LQR

Read "Learning Neural Network Policies with Guided Policy Search under Unknown Dynamics" - link and if your up for it, an important reference "Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems" - link

Major Topics

- Iterative LQR
- Guided Policy Search
- Learning Unknown System dynamics

Topics

Presentation 4: Theory / Sample Complexity of LQR

Read "On the Sample Complexity of the Linear Quadratic Regulator" - link

Major Topics

- Sample Complexity Bounds in LQR
- Computing Unknown Model Dynamics
- Optimization Theory for Control
- System Identification

Topics

Presentation 5: Safe Model Predictive Control

Read "Learning-based Model Predictive Control for Safe Exploration and Reinforcement Learning" - link, and if you want an additional resource for MPC see "Model predictive control: Recent developments and future promise" - link, a complete review of safe RL see: link, or a nice set of slides - here

Major Topics

- Safe exploration
- Model predictive control (MPC)
- combining MPC with reinforcement learning

Topics

Presentation 6:

Read "End-to-End Training of Deep Visuomotor Policies" - link

Major Topics

- Partial Observation
- High dimensional control
- Learning from Images
- Asymmetric Information

Presentation 7: Learning online in high dimensional state-spaces with simple control algorithms

Read "Combining Optimal control and Learning for Visual Navigation in Novel Environments" - link

Major Topics

- Trajectory planning
- Learning perception
- online navigation in environments

Additional reading:

THE UNIVERSITY OF BRITISH COLUMBIA

Lawrence Evans Mini-Textbook

Partial textbook provided for free online at https://math.berkeley.edu/~evans/control.course.pdf.

Bertsekas RL+OC slides

http://web.mit.edu/dimitrib/www/RLbook.html

Two interesting control theory papers

- Lyapunov Functions and Feedback in Nonlinear Control - link
- The O.D.E. Method for Convergence of Stochastic Approximation and Reinforcement Learning - link

Additional reading:

THE UNIVERSITY OF BRITISH COLUMBIA

Control BootCamp (Engineering)

YouTube series:link

Nice tutorial from the perspective of control

Tour of Reinforcement Learning and Control - link