Control of Inverted Double Pendulum using Reinforcement Learning

Stanford University

Fredrik Gustafsson, fregu856@stanford.edu

CS 229 Final Project, Autumn 2016

Introduction

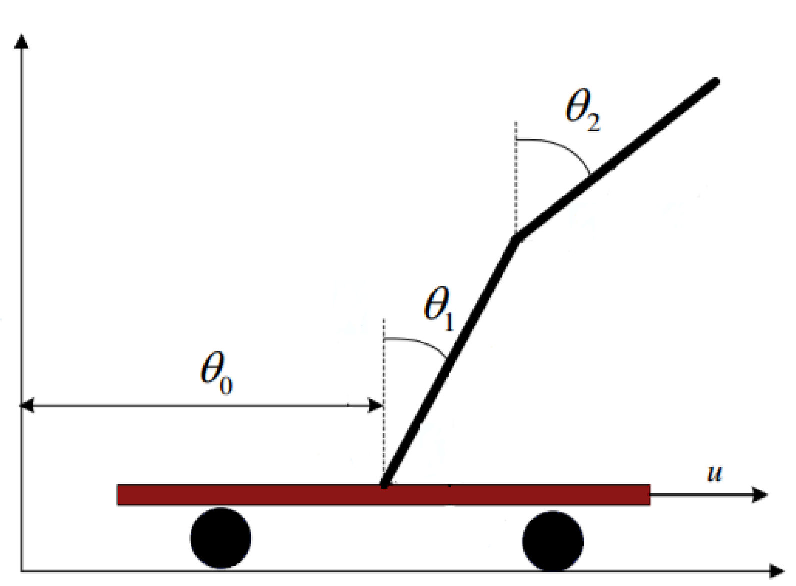
- In this project, we studied how reinforcement learning algorithms can be applied to control a complex dynamical system.
- The goal was to learn controllers for balancing and swing-up, without using any prior knowledge of the system.

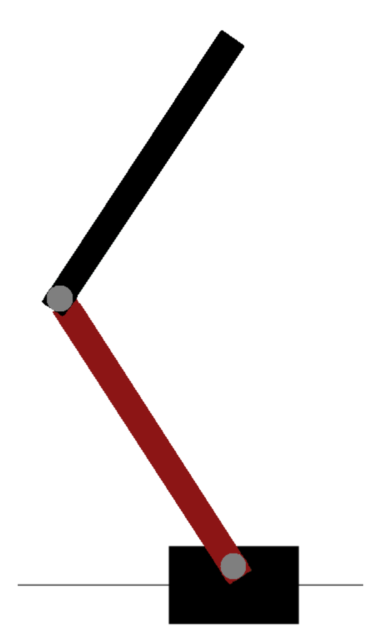
Motivation

- Control of dynamical systems normally requires a detailed mathematical model, which can be both difficult and time consuming to obtain.
- In many cases, it is virtually impossible to explicitly determine which set of actions will make the system behave as desired.
- One would thus like for an agent to explore different control approaches for an unknown system and gradually learn the optimal one: Reinforcement learning.

Learning problem

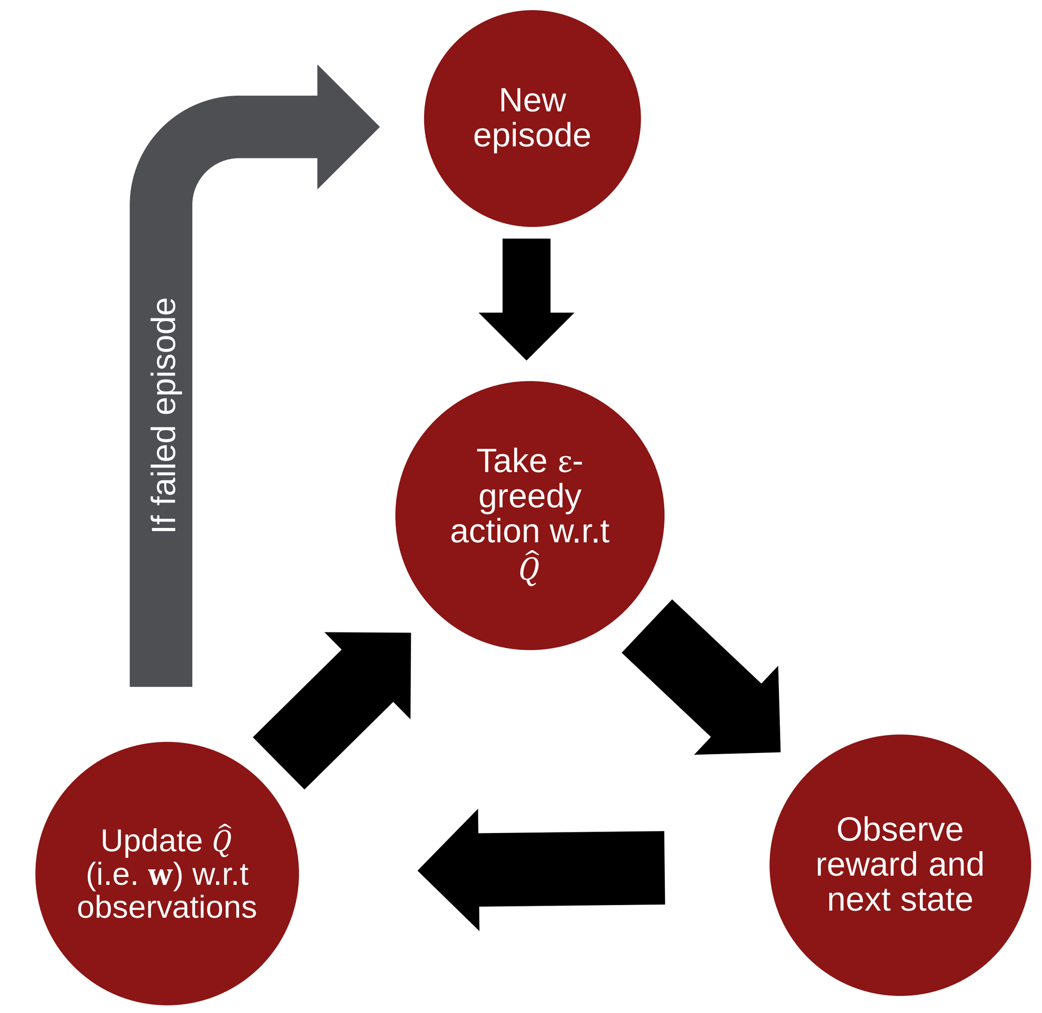
- The **state** $\mathbf{s} = \left(\theta_0, \theta_1, \theta_2, \dot{\theta}_0, \dot{\theta}_1, \dot{\theta}_2\right) \in \mathbb{R}^6$.
- The **action** $a = u$, applied force to the cart.
- Two types of negative **rewards** are defined. The first penalizes every failed episode, the second penalizes any position that is not perfectly vertical.
- A simulator is used in place of an actual physical system.





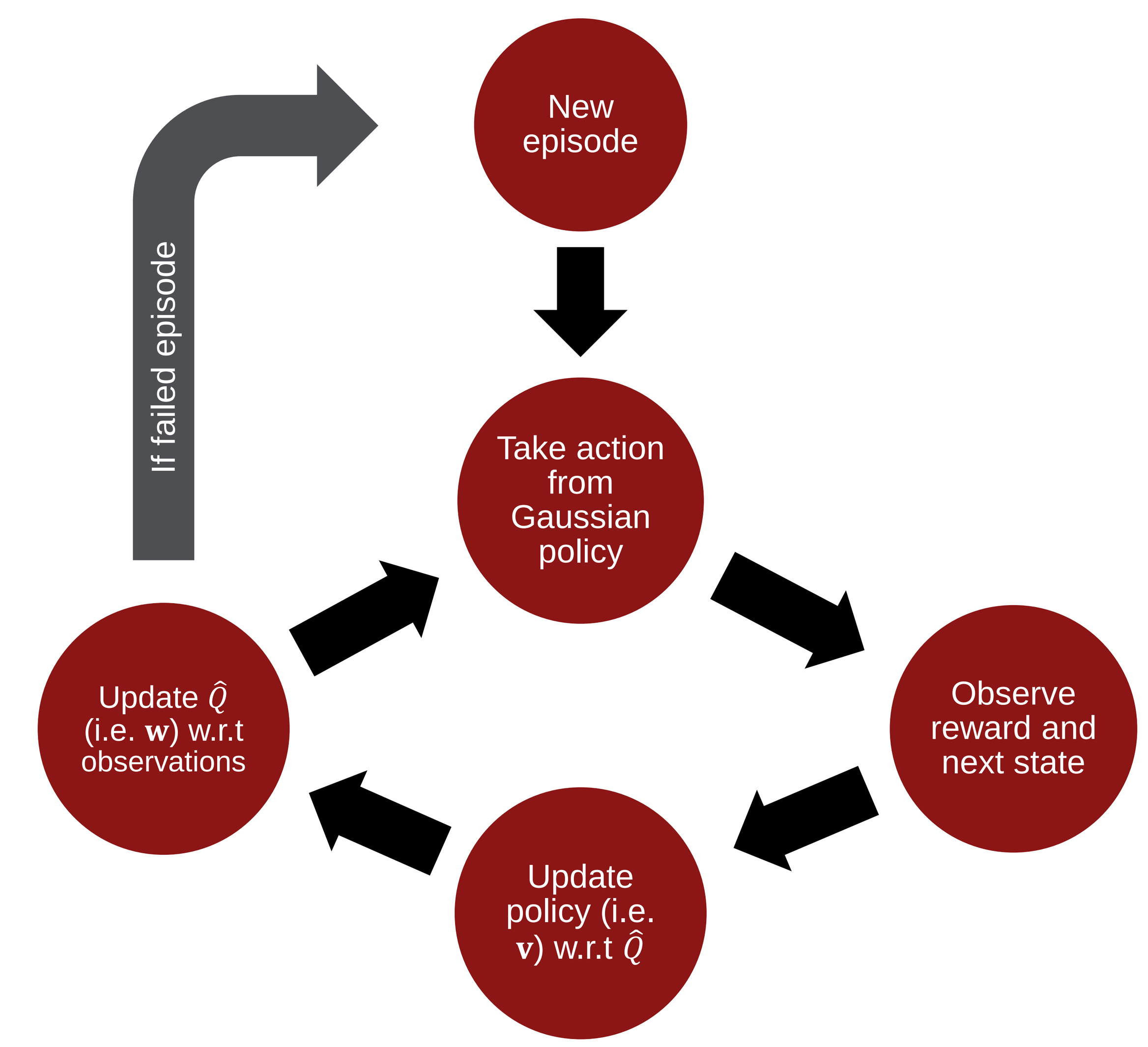
Linear Q-learning

- Q-learning with linear function approximation, $Q(\mathbf{s},\mathrm{a}) \approx \hat{Q}(\mathbf{s},\mathrm{a},\mathbf{w}) = \mathbf{w}^{\mathrm{T}}\mathbf{x}(\mathbf{s},\mathrm{a})$.
- $\mathbf{x}(\mathbf{s},\mathrm{a})$ is a vector of n state-action features.



Gaussian QAC

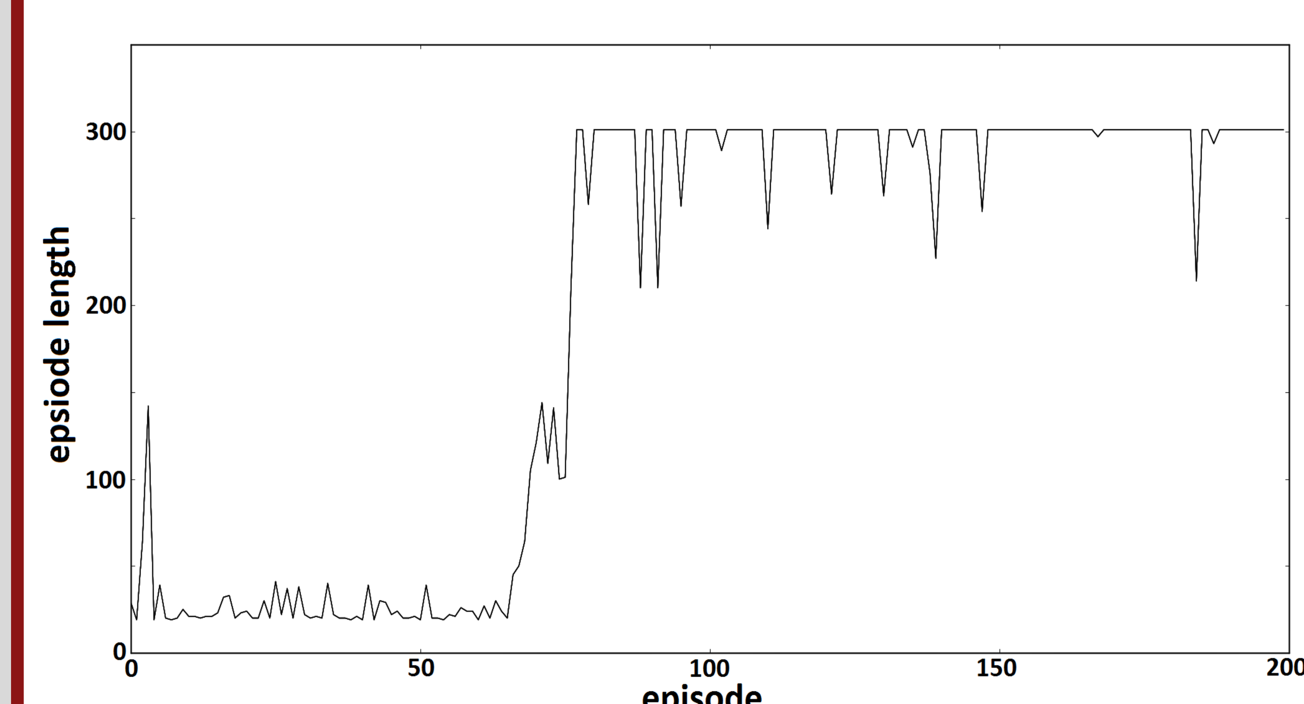
- Actor learns a Gaussian policy using $\hat{Q}$.
- $a \sim N(\boldsymbol{\mu}(\mathbf{s}), \sigma^2)$, $\boldsymbol{\mu}(\mathbf{s}) = \mathbf{v}^{\mathrm{T}}\boldsymbol{\varphi}(\mathbf{s})$.
- $\boldsymbol{\varphi}(\mathbf{s})$ is a vector of m state features.
- Critic learns $\hat{Q}(\mathbf{s},\mathrm{a},\mathbf{w}) = \mathbf{w}^{\mathrm{T}}\mathbf{x}(\mathbf{s},\mathrm{a})$.



Results: balancing

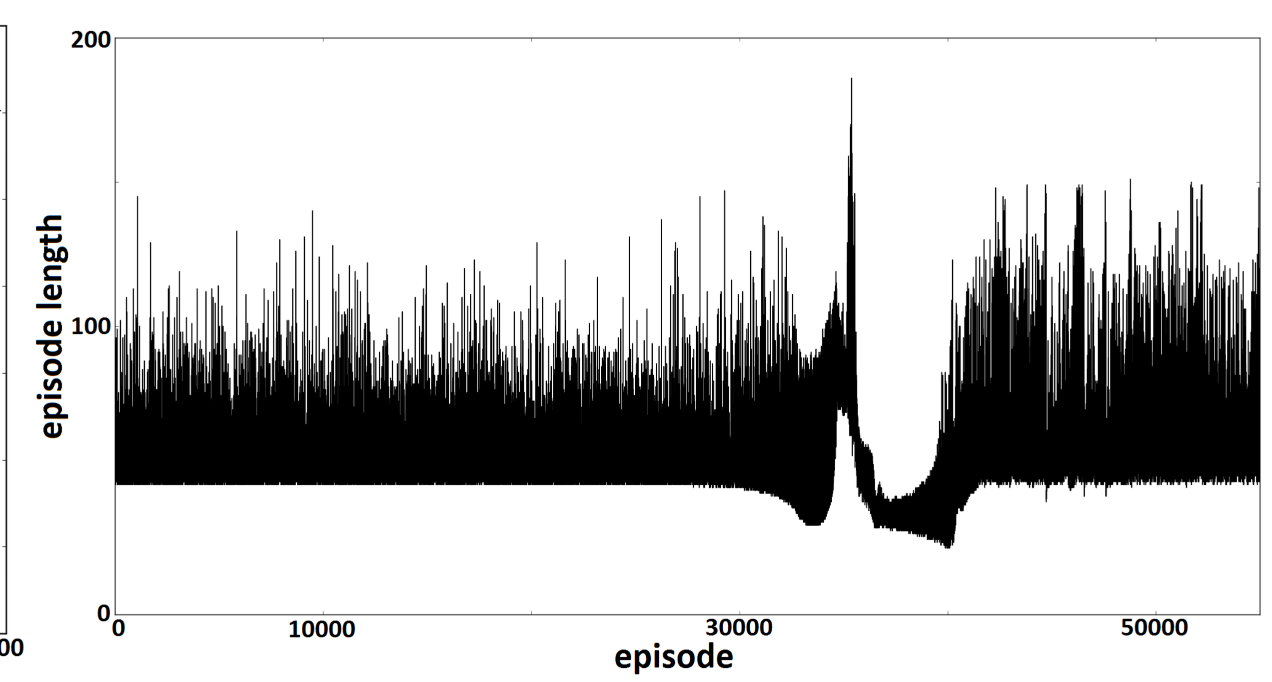
Linear Q-learning

- Converged in **70%** of trials.
- Learned controller capable of balancing for 300+ time steps in **95%** of converged trials.
- **Video**: goo.gl/yLZRT0
- Typical learning curve:



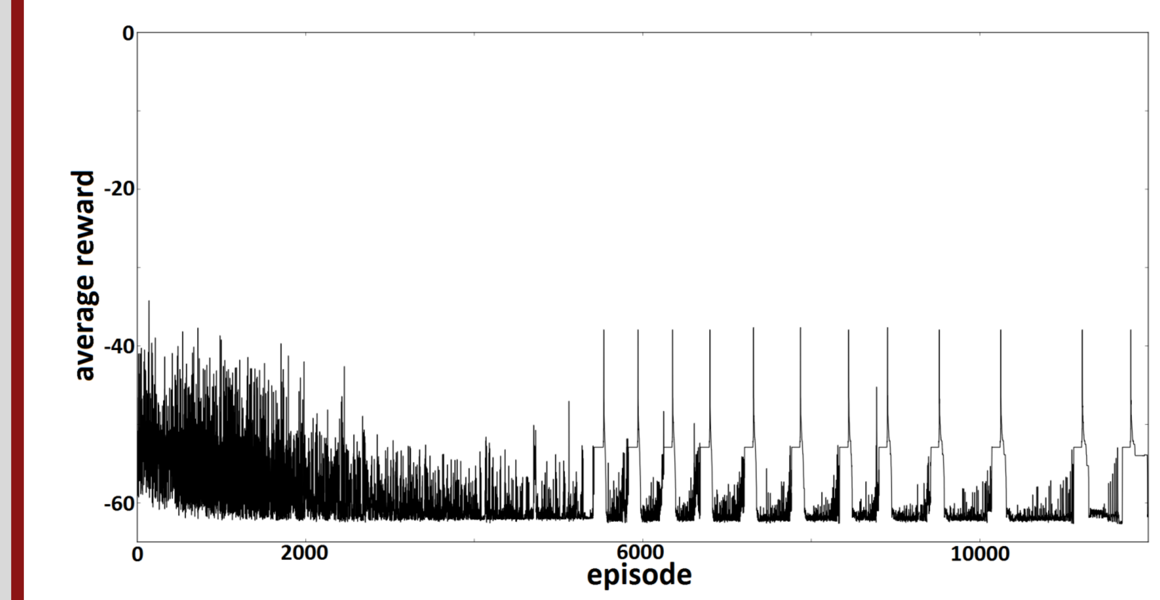
Gaussian QAC

- Never reached convergence in a reasonable number of episodes.
- Never observed episode of 300+ time steps.
- Typical learning curve:



Results: swing-up

- Never reached convergence in a reasonable number of episodes.
- Never observed successful episode.
- Typical learning curve for linear Q-learning:



Conclusions

- The results for balancing are promising and implementation on a physical system seems feasible.
- Swing-up control proved to be a more difficult problem than expected.
- Using only linear function approximators seem to require cleverly designed features. However, **choosing features is difficult** and not always intuitive.
- Linear Q-learning clearly outperformed Gaussian QAC, likely because the latter is more sensitive to the choice of features.
- Future work would study the application of deep reinforcement learning algorithms to swing-up control.

References

- Alexander Bogdanov. Optimal Control of a Double Inverted Pendulum on a Cart. Technical Report CSE-04-006, 2004.
- RL course by David Silver. http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html