Hands on: Reinforcement Learning

Florian Henkel
Institute of *Computational Perception*

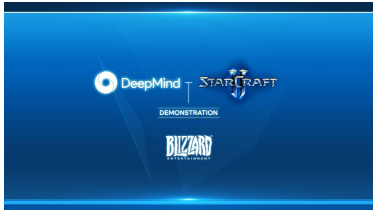
JOHANNES KEPLER UNIVERSITY LINZ

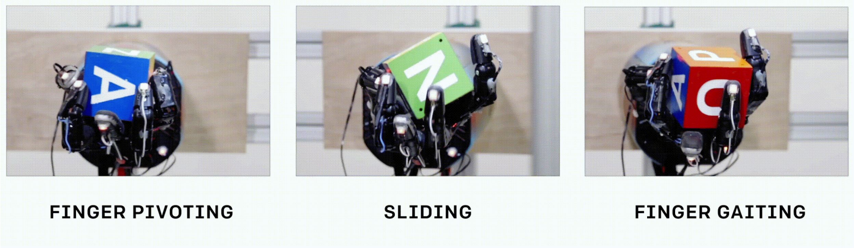
Institute of Computational Perception

Agenda

- Motivation
- Introduction to Reinforcement Learning (RL)
- Policy Gradient Methods
- RL in 100 Lines of Code
- Additional Resources

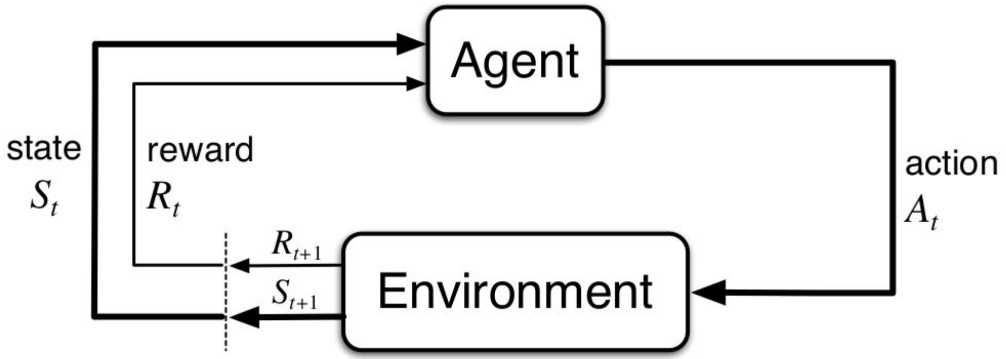
Motivation





https://deepmind.com/research/alphago/
https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/
https://openai.com/five/
https://ai.googleblog.com/2019/01/soft-actor-critic-deep-reinforcement.html
https://openai.com/blog/learning-dexterity/

Introduction to RL



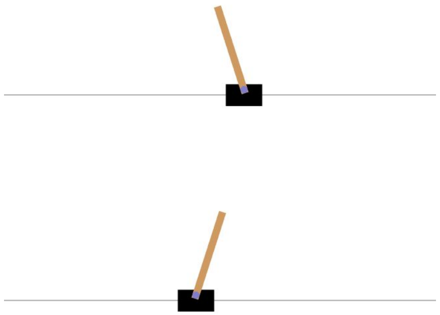
Reinforcement Learning: An Introduction (Sutton and Barto, 2018)

- Given a certain **state** within the environment the agent acts/chooses an action according to a **policy**
- The environment reacts to the action and returns a new state and a **reward**

Understanding the Policy

- Agent learns a policy $\pi(a \mid s)$
- Goal: find an optimal policy to maximize the performance of the agent i.e. get as much reward as possible (optimization problem)

Current state s



$\pi(a \mid s)$	
Push to left	Push to right
0.8	0.2
0.25	0.75

Understanding Rewards

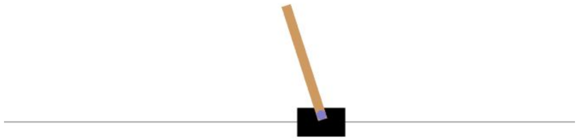
- Our goal is to get as much reward as possible
- Defined by the (discounted) return

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

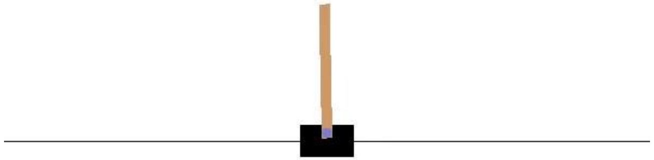
- Discounting between [0, 1]
  - < 1 keeps the sum finite (mathematical convenience)
  - Trade-off between influence of immediate and future rewards

Understanding the State-Value

- State-value function $v(s) = \mathbb{E}\big[G_t \mid S_t = s\big]$



Close to failure, we do not **expect** to get that much future reward



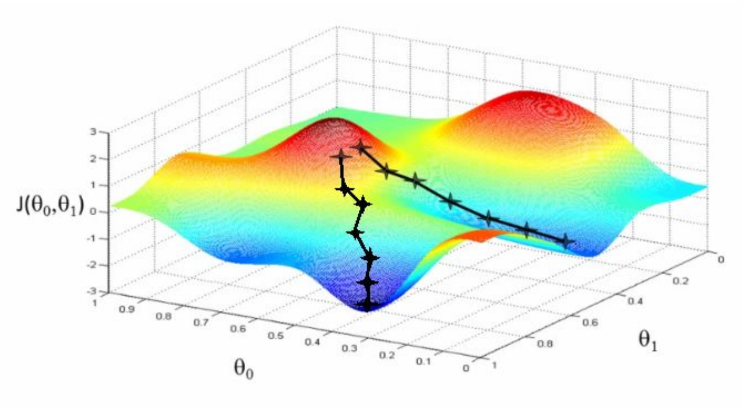
Perfect, we **expect** lots of future reward

Policy Gradient Methods

- Policy is parameterized $\pi(a \mid s; \boldsymbol{\theta})$ (e.g. with a neural network)
- Differentiability allows us to perform stochastic gradient ascend
- REINFORCE (Williams, 1992)

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha G_t \frac{\nabla \pi(A_t \mid S_t; \boldsymbol{\theta}_t)}{\pi(A_t \mid S_t; \boldsymbol{\theta}_t)}$$

- State-of-the-art methods also often learn a state-value function $v(s; \boldsymbol{\theta}_v)$



http://blog.datumbox.com/tuning-the-learning-rate-in-gradient-descent/

JOHANNES KEPLER UNIVERSITY LINZ

Policy Gradient Methods

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
    Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
    Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
        $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ $\quad (G_t)$
        $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \gamma^t G \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

Reinforcement Learning: An Introduction (Sutton and Barto, 2018)

RL in 100 Lines of Code

Additional Resources

- Two courses on RL at JKU
  - WS: Special Topics: Reinforcement Learning (CP)
  - SS: Special Topics: Deep Reinforcement Learning (ML)

- Online Resources:
  - David Silver's lecture at UCL
  - Deep Reinforcement Learning UC Berkeley
  - Deep RL Bootcamp
  - OpenAI's Spinning up in Deep RL

- Reinforcement Learning: An Introduction (Sutton and Barto, 2018)

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement number 670035).

European Research Council

Established by the European Commission

JOHANNES KEPLER UNIVERSITY LINZ