Fraunhofer IIS

Fraunhofer-Institut für Integrierte Schaltungen IIS

Reinforcement Learning

Lecture 1: Introduction to RL

Christopher Mutschler

Opening Remarks
Class Logistics

- We (will start with) use StudOn for main communication (forum + messages, announcements)
- If you have any questions, you can also write to

Christopher
christopher.mutschler@iis.fraunhofer.de

Sebastian
sebastian.rietsch@iis.fraunhofer.de

Nico
nico.meyer@iis.fraunhofer.de

- If we will ever use a password somewhere, it will be **FAU_RL_2023**

Opening Remarks

Syllabus

Week	Lecture		Exercises		
1.	19.04.	Intro to RL, MDPs			
2.	26.04.	Dynamic Programming	28.04.	MDPs, DP	→ 05.07.
3.	03.05.	Model-free Prediction	05.05.	OpenAI Gym, TD-Learning	→ 12.05.
4.	10.05.	Model-free Control	12.05.	TD-Control	→ 19.05.
5.	17.05.	Value Function Approximation	19.05.	PyTorch, DQNs	→ 02.06.
6.	24.05.	Policy-based RL #1	26.05.		
7.	31.05.	Policy-based RL #2	02.06.	VPG, A2C, PPO	→ 16.06.
8.	07.06	Guest Lecture: Quantum Reinforcement Learning	09.06.		
9.	14.06.	Model-based RL #1 (Discrete Actions)	16.06.	MCTS	→ 23.06.
10.	21.06.	Model-based RL #2 (Continuous Actions)	23.06.	MPC, CEM	→ 30.06.
11.	28.06.	Exploration-Exploitation, Regret, Bandits	30.06.	Multi-armed Bandits	→ 07.07.
12.	05.07.	Exploration in Deep RL, Intrinsic Motivation	07.07.	RND/ICM	→ 14.07.
13.	12.07.	Offline RL	14.07.	BCQ	→ 19.07. (lecture slot)
14.	19.07.	Guest Lecture: ChatGPT Course Wrap-Up, Evaluation Results			

Opening Remarks
Syllabus

Tips & Tricks: How to be successful in this class

- Play with the Jupyter notebooks (if we provide them)
- Attend the classes, follow the content, and ask questions
- Attend the exercises and implement them
- ..one more thing
  - This is not a class to obtain 5 ECTS "as easy as it might be"

Opening Remarks
Exam

- Currently, the plan is to have **a written** exam.

- Final scheduling and logistics of the exam will be done around June/July
  - Exam will (most likely) take place in the first examination period! (03.08.2023)

- The exams will cover topics from both **lectures** and **exercises**
  - Exams will focus on **basic understanding** of concepts
  - Exams will have **theoretical parts** (but we will not include proofs)
  - Exams will focus on **practical aspects** (i.e., implementation w.r.t. exercise content)

- Exam is in English (in the unlikely case of oral exams we can also do it in German)

Opening Remarks

What's your study program?

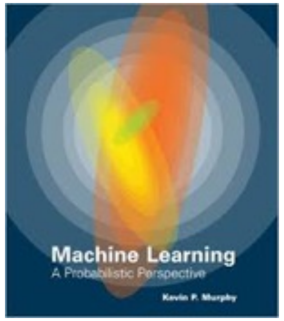
https://www.menti.com

Opening Remarks

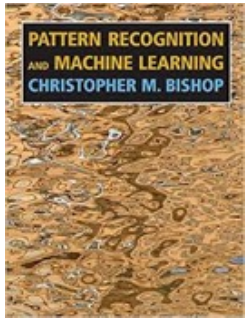
Prerequisites

- What pre-requisites do we expect?
  - Analysis/Calculus
  - Multivariate Statistics
  - Machine Learning, Deep Learning
  - Python (to get used to the exercises)
- How can you get them?
  - You attended the recommended basic lectures: MLTS, Pattern Recognition, Deep Learning
  - Or/And dive into one of those books (better today than tomorrow):

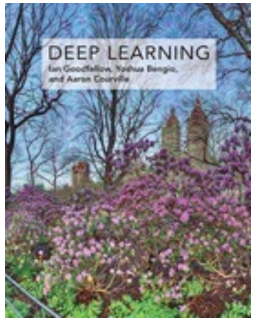
Do not be so casual about it!



Kevin Murphy: Machine learning; a probabilistic perspective.



Christopher Bishop: Pattern Recognition and Machine Learning



Ian Goodfellow and Yoshua Bengio and Aaron Courville: Deep Learning

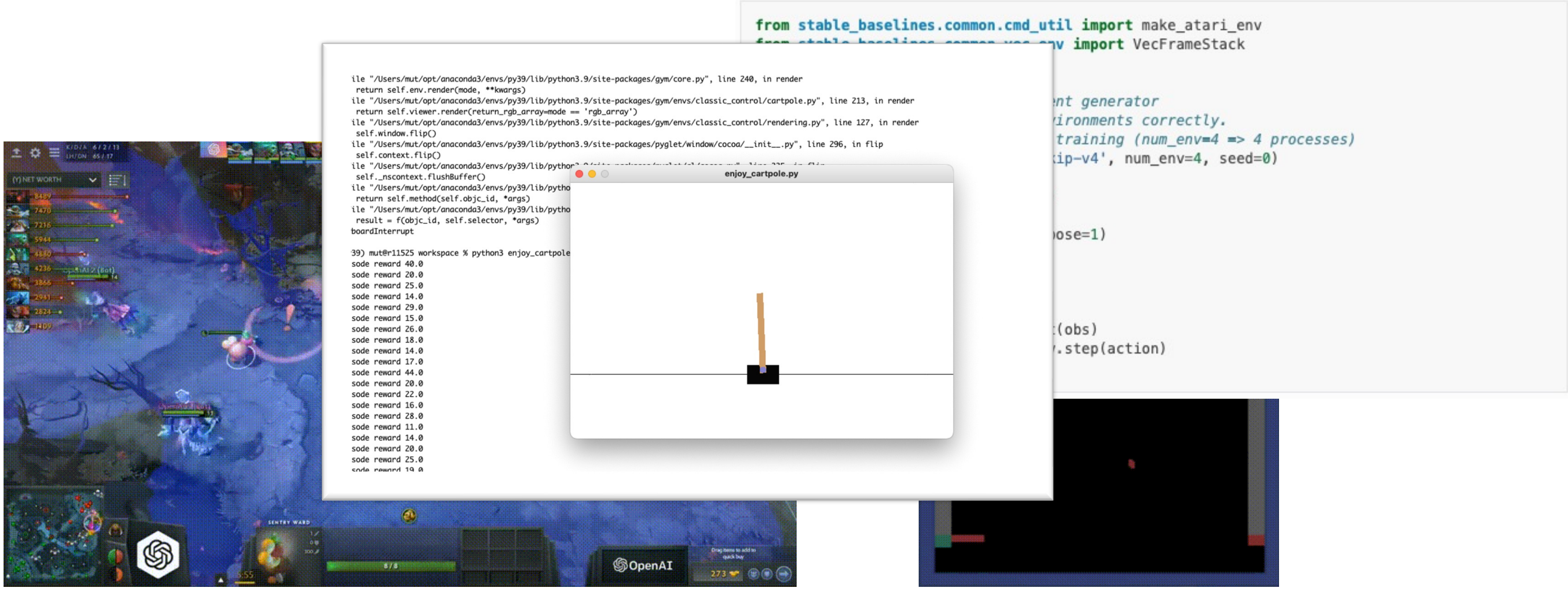
- You will find RL literature in Lecture 1.02 (later today)

Fraunhofer IIS

Introduction to Reinforcement Learning

Playing games with RL

https://stable-baselines.readthedocs.io/en/master/guide/examples.html



https://www.youtube.com/watch?v=lc1fl5bdZdA

https://www.youtube.com/watch?v=V1eYniJ0Rnk

Also watch the nice marketing video on AlphaGo: https://www.youtube.com/watch?v=I2WFvGl4y8c

Introduction to Reinforcement Learning

Finding multi-agent soccer strategies with RL

https://www.youtube.com/watch?v=F8DcgFDT9sc

see also: https://github.com/google-research/football

Fraunhofer IIS

Introduction to Reinforcement Learning

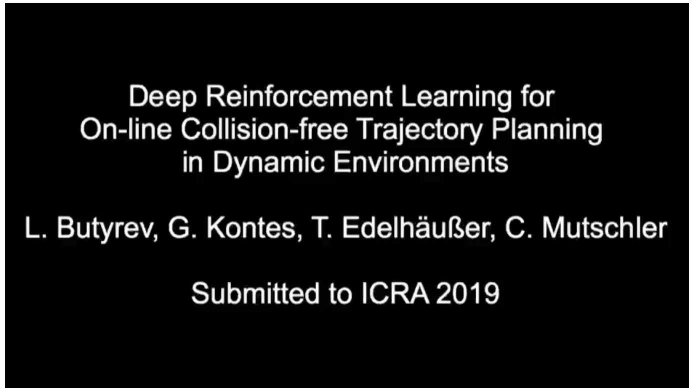
Controlling robots with RL

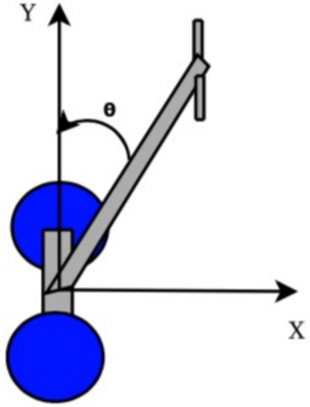
https://www.youtube.com/watch?v=0JL04JJjocc

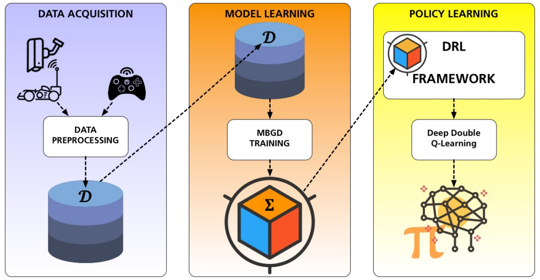
https://www.youtube.com/watch?v=W_gxLKSsSIE

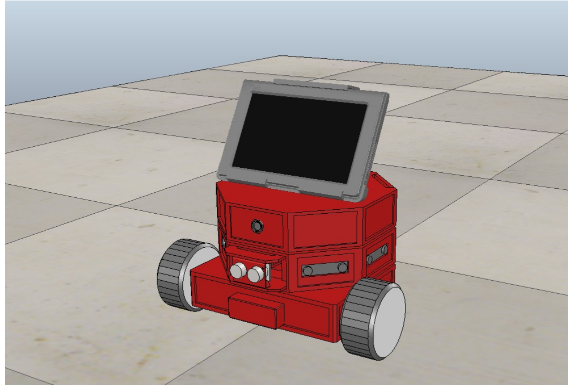
Introduction to Reinforcement Kearning

Controlling robots with RL



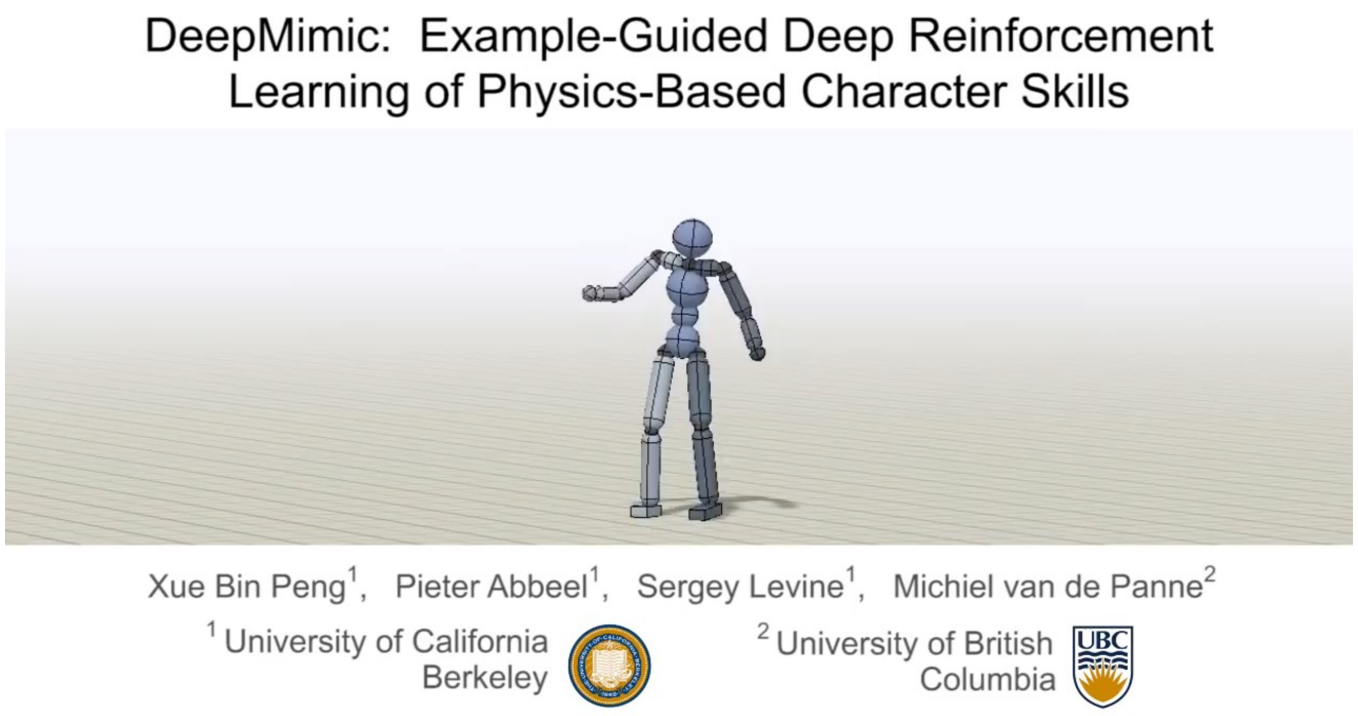






Introduction to Reinforcement Learning

Advanced robot control in simulation



https://www.youtube.com/watch?v=vppFvq2quQ0

Fraunhofer IIS

Introduction to Reinforcement Learning

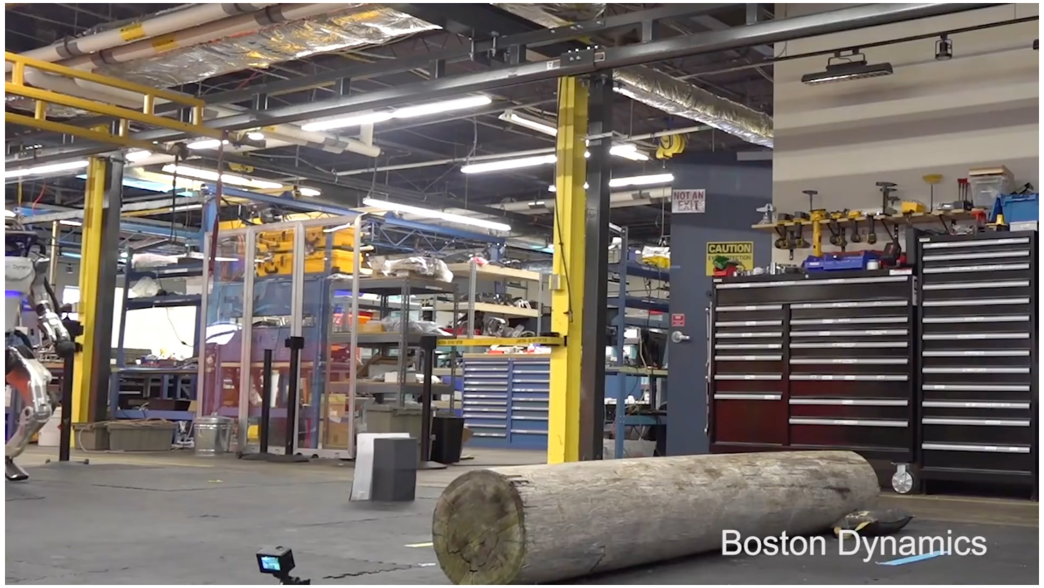
Advanced robot control in reality

https://www.youtube.com/watch?v=x4O8pojMF0w

Introduction to Reinforcement Learning

Advanced robot control in reality

https://www.youtube.com/watch?v=8RILnqPxo1s

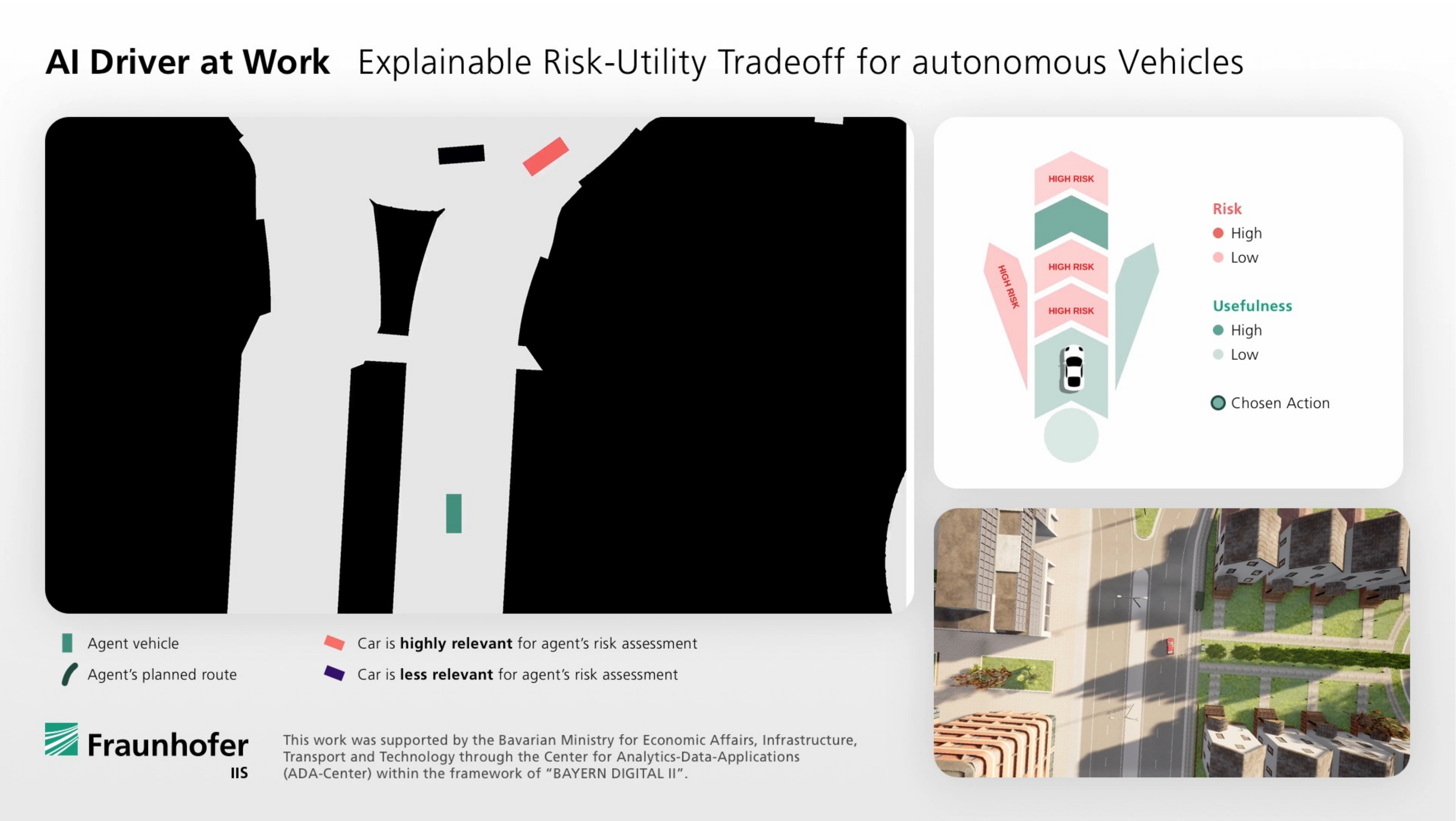


https://www.youtube.com/watch?v=LikxFZZO2sk

Not RL ☺

Fraunhofer IIS

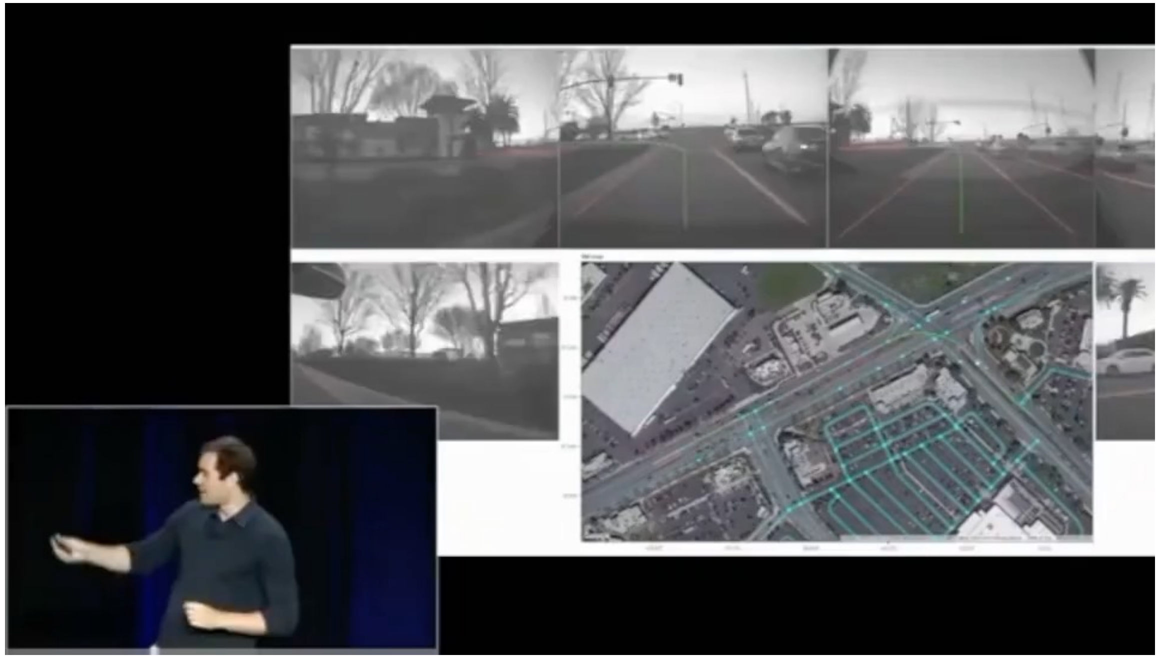
Introduction to Reinforcement Learning
Driving cars with RL



https://youtu.be/0IWjE_8xj6Q

Introduction to Reinforcement Learning

Path planning/navigation



https://www.youtube.com/watch?v=v5I-jPsAK7k

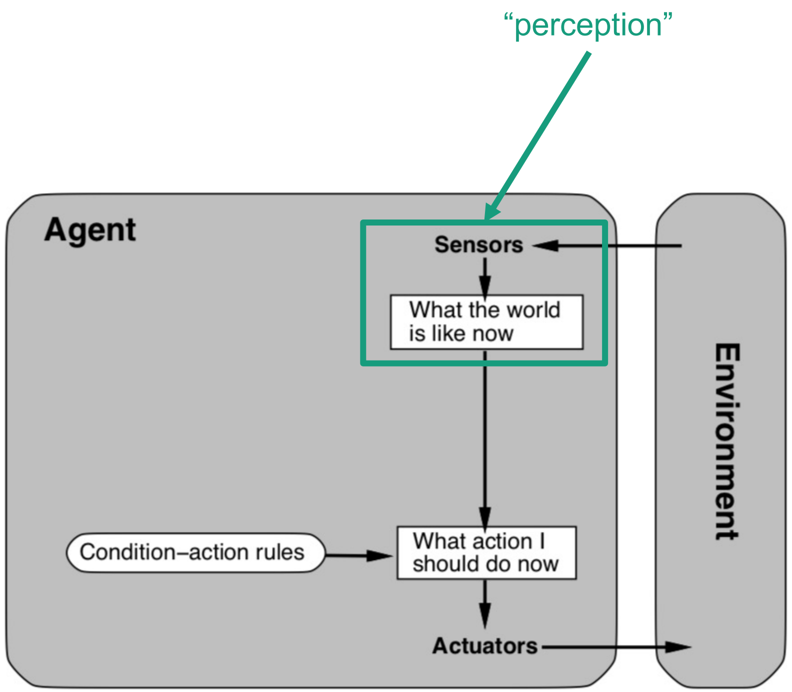
https://www.youtube.com/watch?v=H7Ym3DMSGms

Introduction to Reinforcement Learning

What are “autonomous systems”?

- **Autonomous Agent (Simple Reflex Agent)**
- An Autonomous Agent is **anything** that:
  - Perceives its environment via sensors
  - Acts on it with actuators
  - Operates without any interference (autonomously)

→ **Percept & Act**



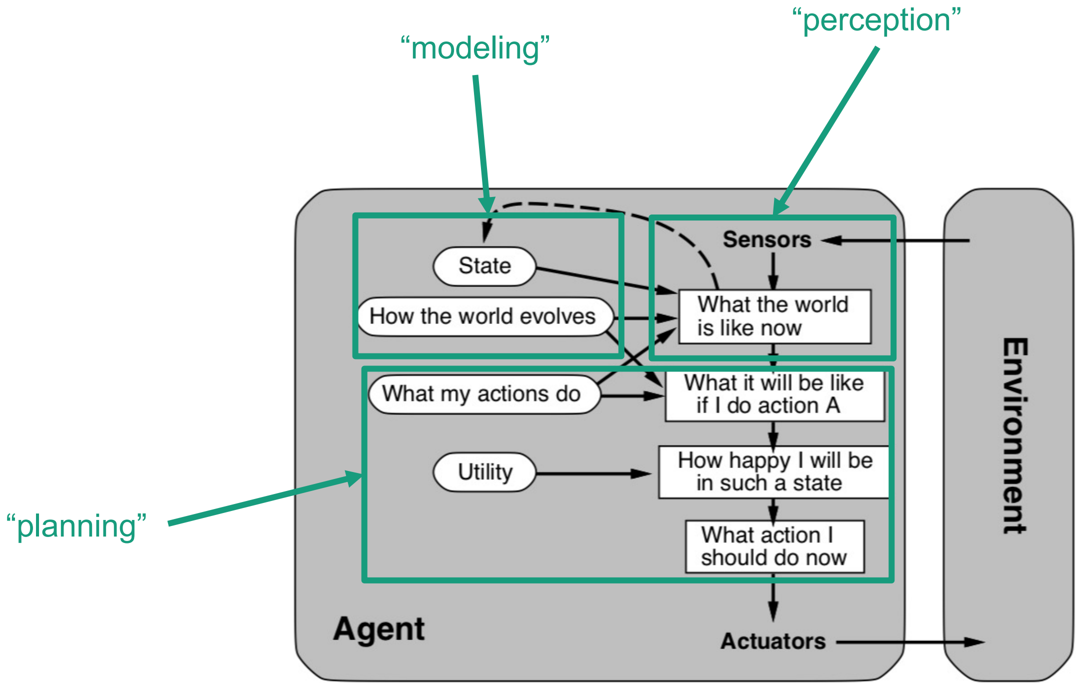
Russell, S. J., & Norvig, P. (2016). Artificial intelligence: A Modern Approach. Malaysia; Pearson Education Limited.

Introduction to Reinforcement Learning

What are “autonomous systems”?

- **Intelligent (Utility-based) Agent**
- An intelligent agent is **anything** that:
  - Perceives its environment via sensors
  - Acts on it with actuators
  - Operates without any interference (autonomously)
  - Directs its activity towards achieving goals or maximizing a utility function

→ Percept & Plan to Control



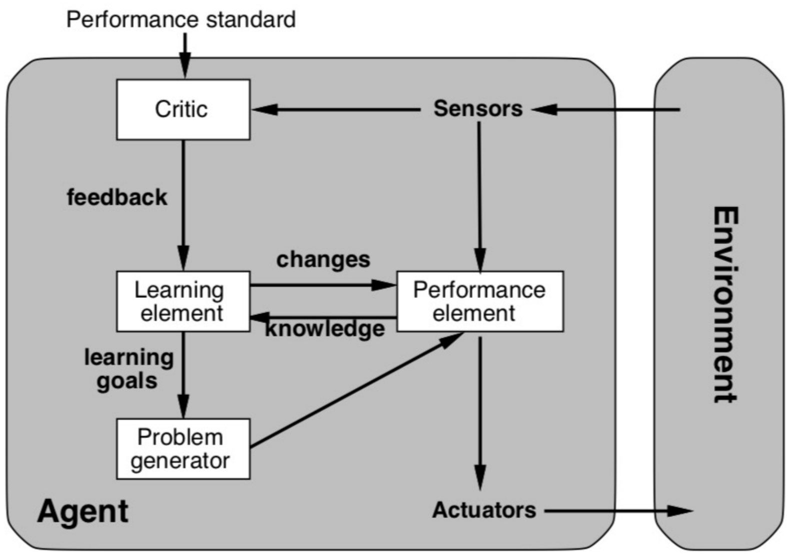
Russell, S. J., & Norvig, P. (2016). Artificial intelligence: A Modern Approach. Malaysia; Pearson Education Limited.

Introduction to Reinforcement Learning

What are “autonomous systems”?

- **Learning Agent**
- A learning agent is **anything** that:
  - Perceives its environment via sensors
  - Acts on it with actuators
  - Operates without any interference (autonomously)
  - **Learns** how to better achieve goals or maximize a **utility function**

→ Percept & Learn to Control
(not necessarily separate)



Russell, S. J., & Norvig, P. (2016). Artificial intelligence: A Modern Approach. Malaysia; Pearson Education Limited.

Introduction to Reinforcement Learning

What are "autonomous systems"?

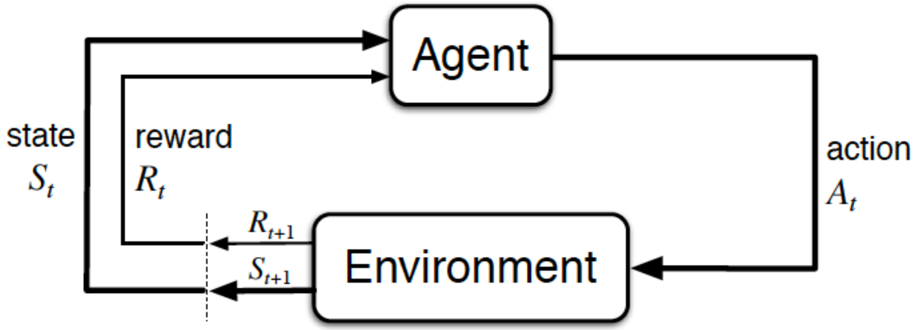
- Autonomous Cars
- Smart Homes/Buildings that adapt to occupants
- Intelligent traffic lights control
- Software trading agents
- Virtual assistants that manage appointments or answer emails automatically
- Recommender systems, e.g., for movies (Netflix), consumer products (Amazon), advertisements (Google), content (Facebook) or music (Spotify) recommendations
- Player Modelling and Content Generation in Computer Games

Introduction to Reinforcement Learning

The RL Paradigm (reward hypothesis)

- Do you agree with following statement?

"All goals can be described by the maximization of expected cumulative ***reward****."*

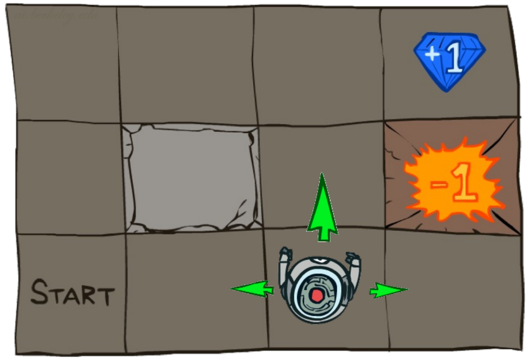


Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

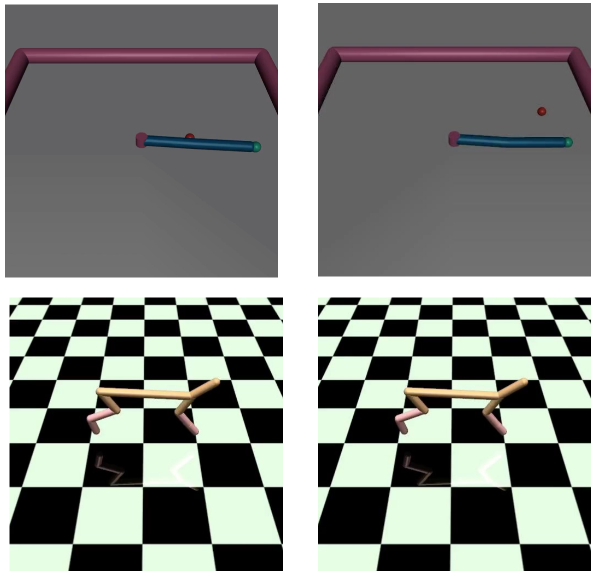
Introduction to Reinforcement Learning

Goals for different applications

- Control a robot in the Gridworld
  - Getting to the treasure
  - Falling into traps
- Play videogames
  - Increasing the score
  - Decreasing the score
- Fly stunt maneuvers in a helicopter
  - Following desired trajectory
  - Crashing
- Humanoid walk
  - Forward motion
  - Falling over

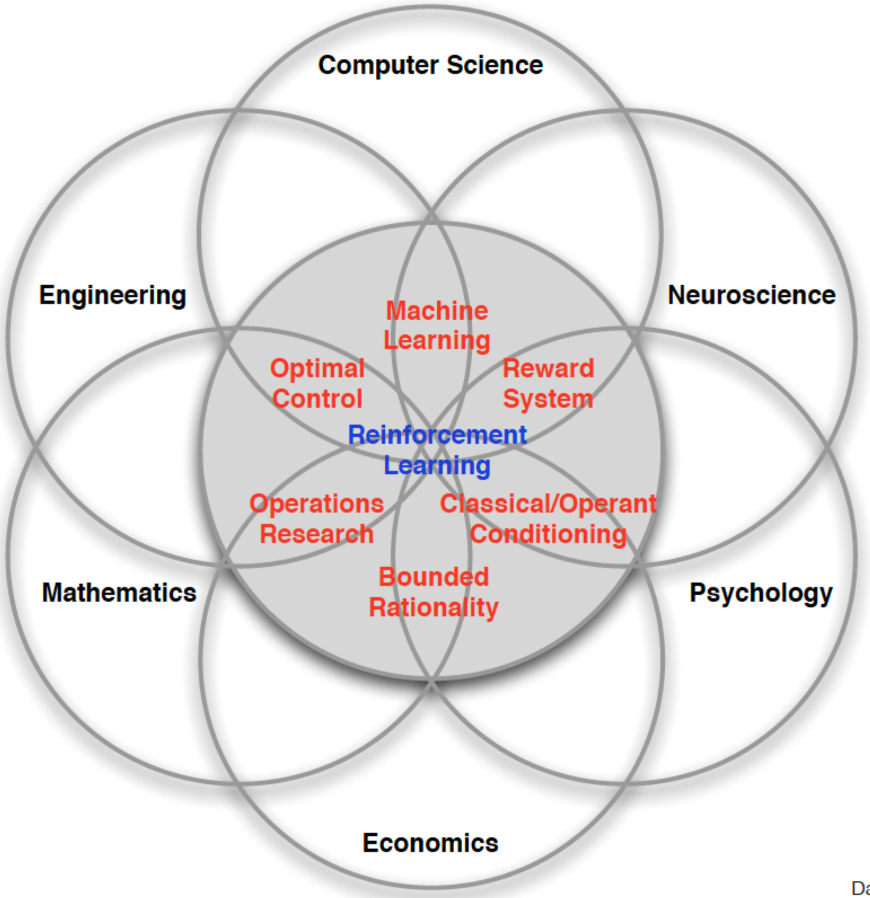


http://ai.berkeley.edu/lecture_slides.html



Introduction to Reinforcement Learning

RL vs the world: the many faces of Reinforcement Learning



David Silver 2015

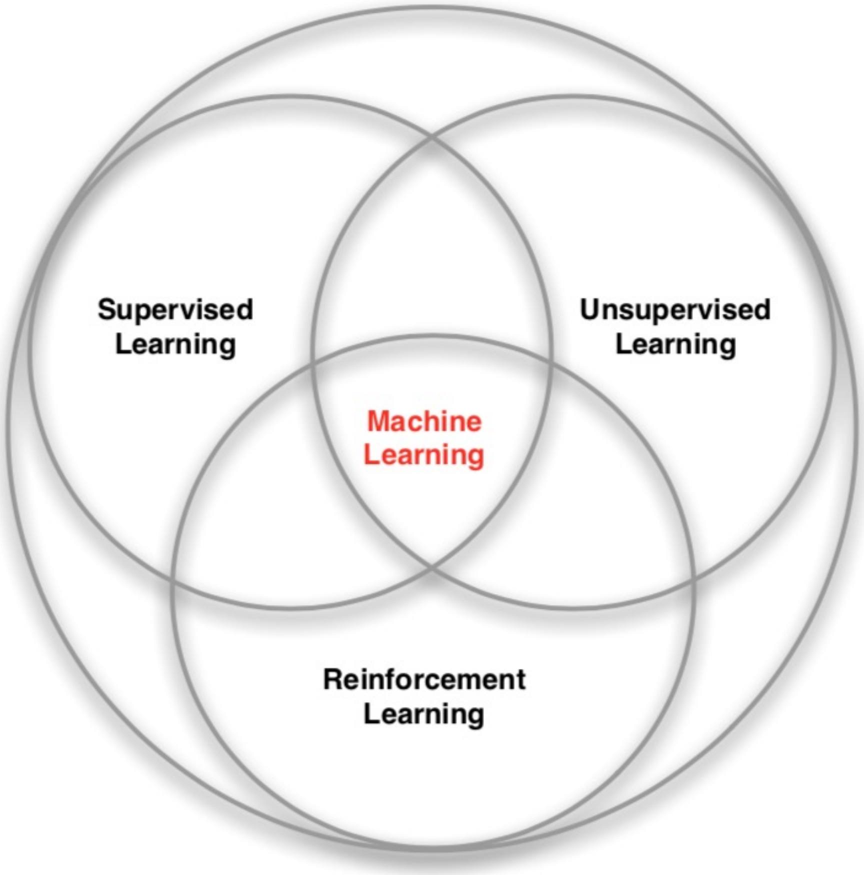
see also: https://www.youtube.com/watch?v=-63ysqT5nu0

Fraunhofer IIS

Introduction to Reinforcement Learning

RL vs other ML branches

- No teacher/supervisor, only reward signals.
- Delayed feedback, not instantaneous (credit assignment problem).
- Learning by interaction between environment and agent over time.
- Agent's actions affect the environment: Actions have consequences!!! → non i.i.d.!
- Active Learning process: the actions that the agent takes affect the subsequent data the agent receives

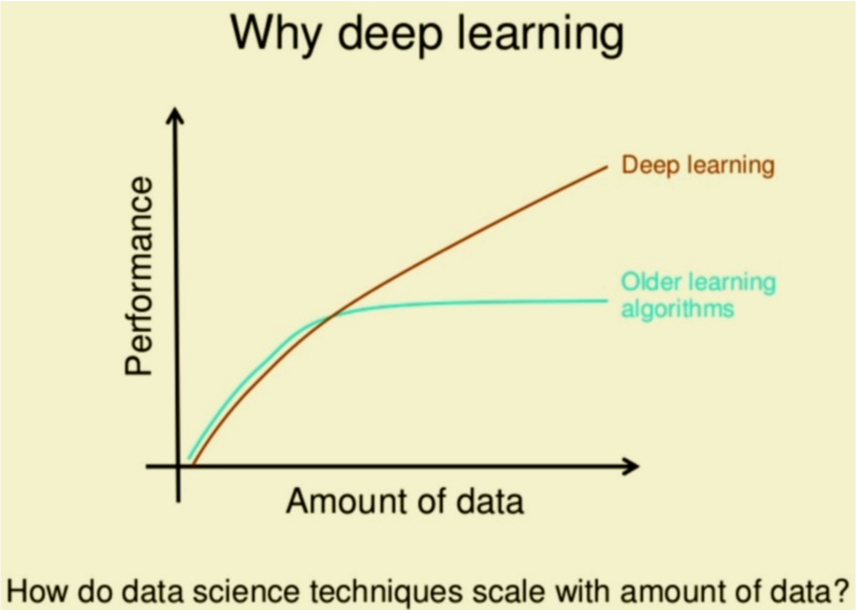


see also: https://www.youtube.com/watch?v=-63ysqT5nu0

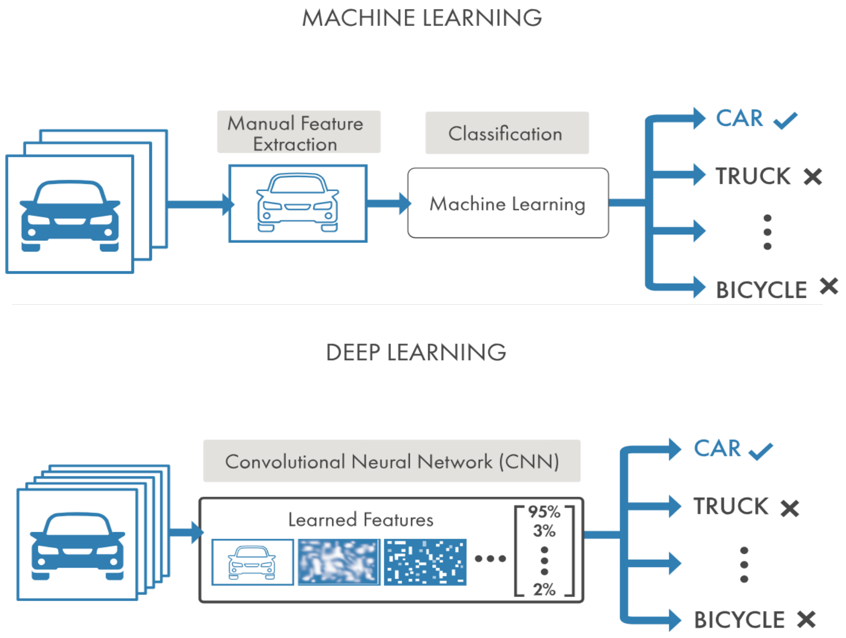
Introduction to Reinforcement Learning

Why RL now?

- Taking advantage of advances in:
  - **Deep Learning Algorithms (DL)**



https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063



https://www.mathworks.com/discovery/deep-learning.html

Introduction to Reinforcement Learning

Why RL now?

- Taking advantage of advances in:
  - Deep Learning Algorithms (DL)
  - **Software for DL and RL**

Caffe2

Keras

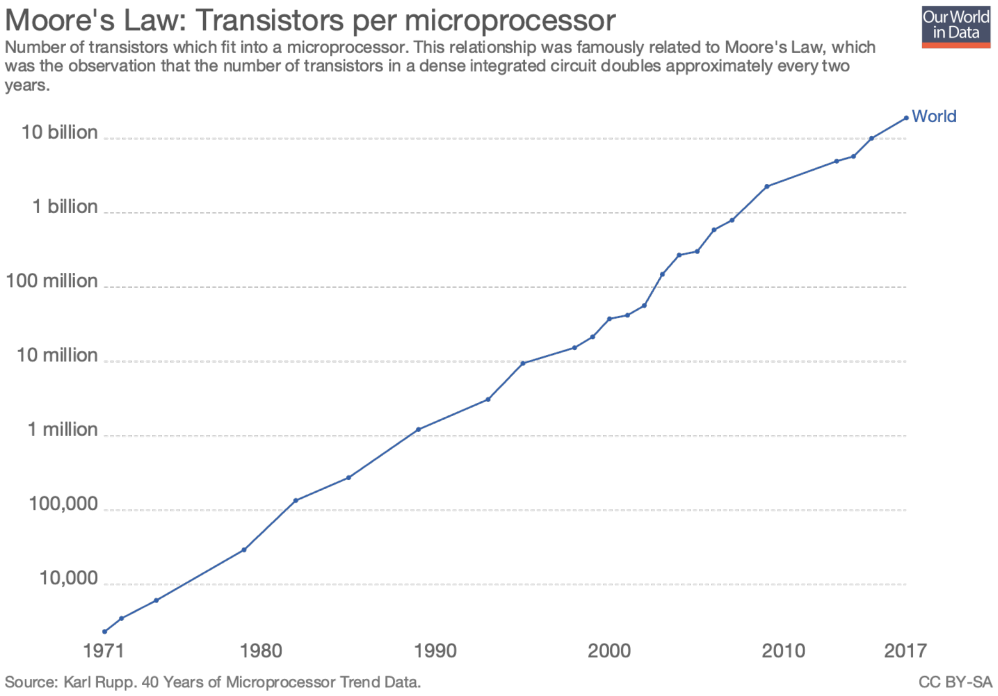
Introduction to Reinforcement Learning

Why RL now?

- Taking advantage of advances in:
  - Deep Learning Algorithms (DL)
  - Software for DL and RL
  - **Hardware (CPU & Memory)**

	OPENAI 1V1 BOT	OPENAI FIVE
CPUs	60,000 CPU cores on Azure	128,000 preemptible CPU cores on GCP
GPUs	256 K80 GPUs on Azure	256 P100 GPUs on GCP
Experience collected	~300 years per day	~180 years per day (~900 years per day counting each hero separately)

https://blog.openai.com/openai-five/

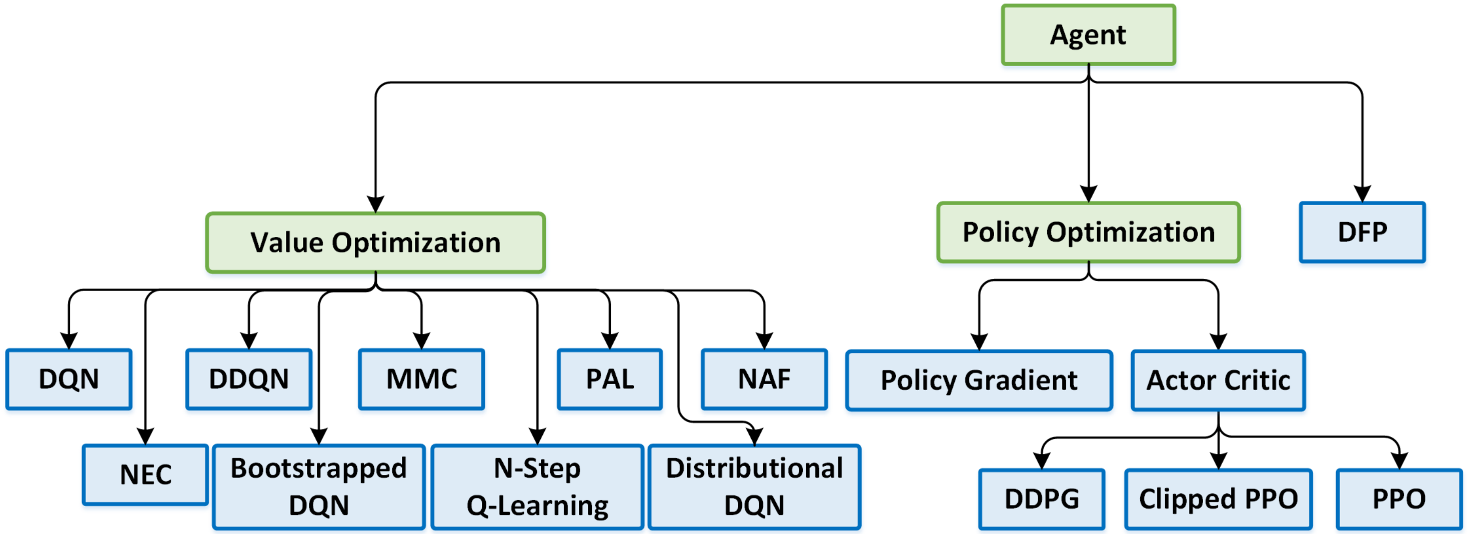


https://ourworldindata.org/technological-progress

Introduction to Reinforcement Learning

Why RL now?

- Taking advantage of advances in:
  - Deep Learning Algorithms (DL)
  - Software for DL and RL
  - Hardware (CPU & Memory)
  - **Deep RL**



https://ai.intel.com/reinforcement-learning-coach-intel/

Introduction to Reinforcement Learning

Why RL now?

- Taking advantage of advances in:
  - Deep Learning Algorithms (DL)
  - Software for DL and RL
  - Hardware (CPU & Memory)
  - Deep RL
  - **(Really good) Open Source Algorithm Implementations**

Introduction to Reinforcement Learning

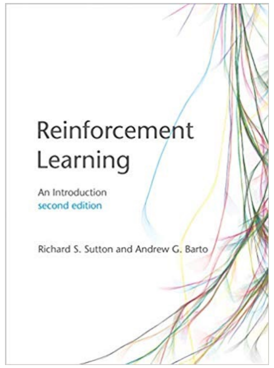
Why RL now?

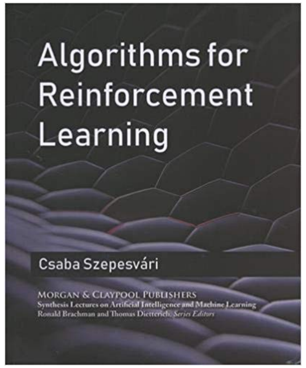
- Taking advantage of advances in:
  - Deep Learning Algorithms (DL)
  - Software for DL and RL
  - Hardware (CPU & Memory)
  - Deep RL
  - (Really good) Open Source Algorithm Implementations
  - **You!**

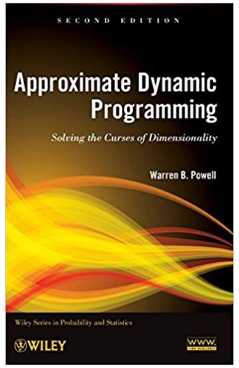
Fraunhofer IIS

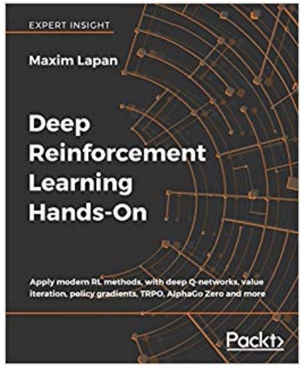
Introduction to Reinforcement Learning

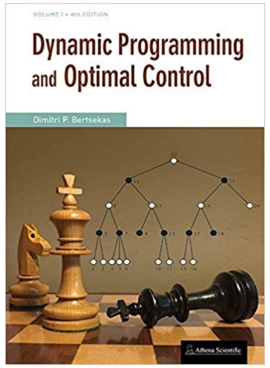
Literature

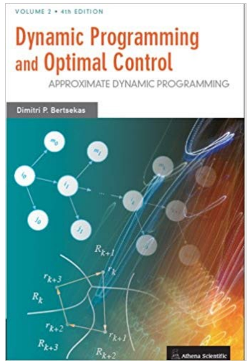


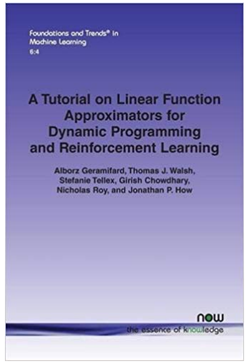










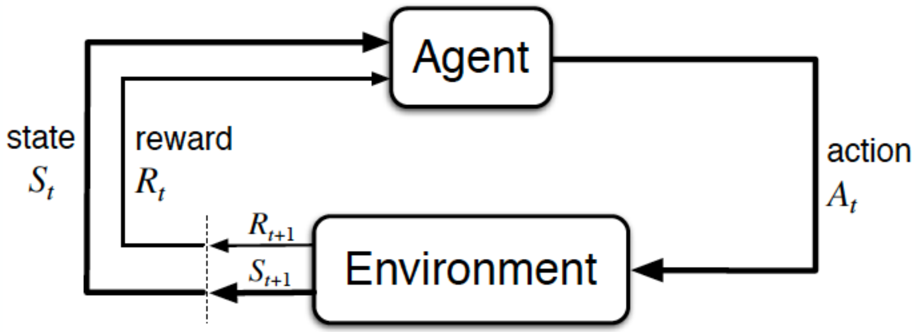


Introduction to Reinforcement Learning

The RL Paradigm (revisited)

- Do you agree with following statement?

"All goals can be described by the maximization of expected cumulative ***reward****."*

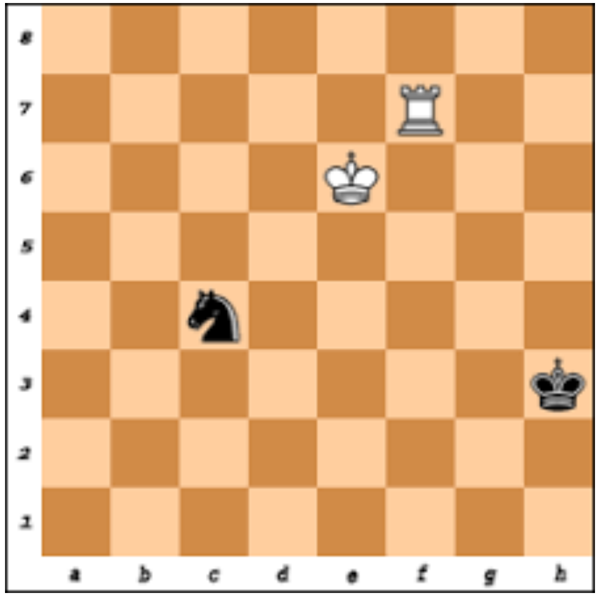


Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

Introduction to Reinforcement Learning

Challenges of sequential decision making

- **Goal: select actions to maximize total future reward**
- Actions may have long-term consequences
- Reward may be delayed
- It may be better to sacrifice immediate reward to gain more long-term reward
- Examples:
  - Financial investments
  - Refueling the helicopter
  - Game playing?

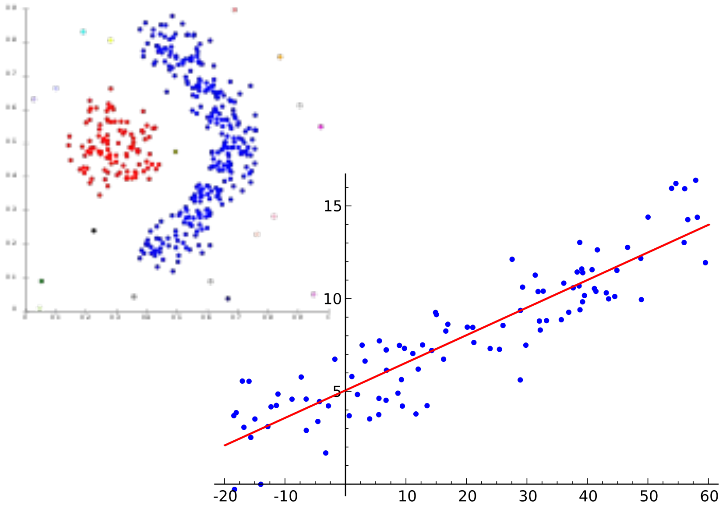


Introduction to Reinforcement Learning

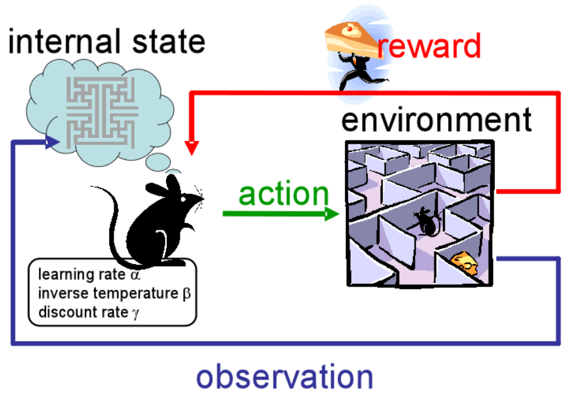
Challenges of understanding/adopting RL

- Counter-Intuitive Visualization!!!

Supervised
Learning



Reinforcement
Learning

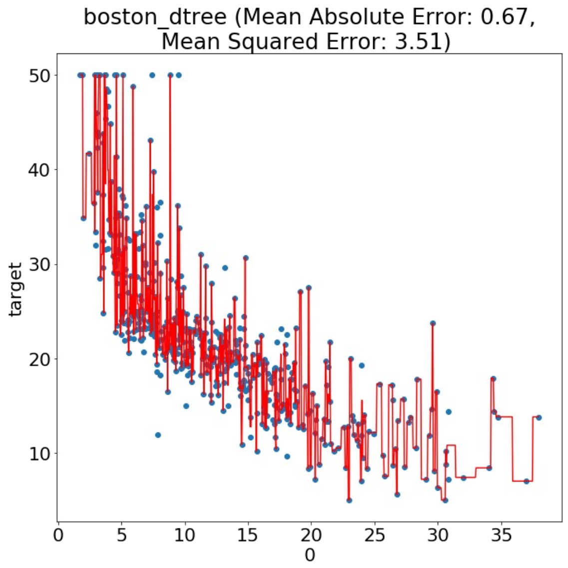


Introduction to Reinforcement Learning

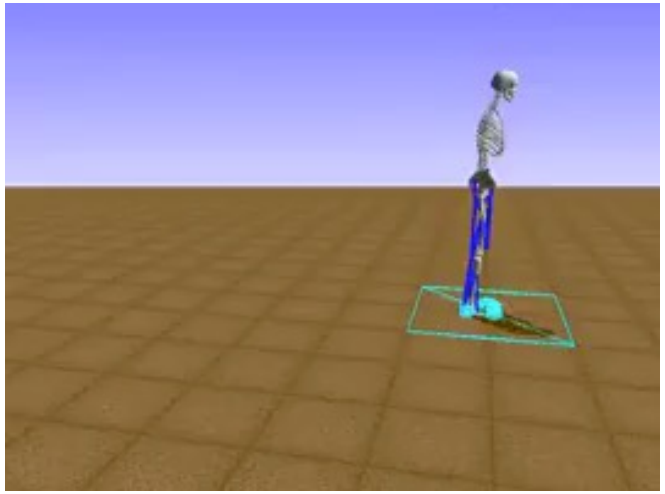
Challenges of understanding/adopting RL

- Example: what went wrong here?

Supervised Learning



Reinforcement Learning

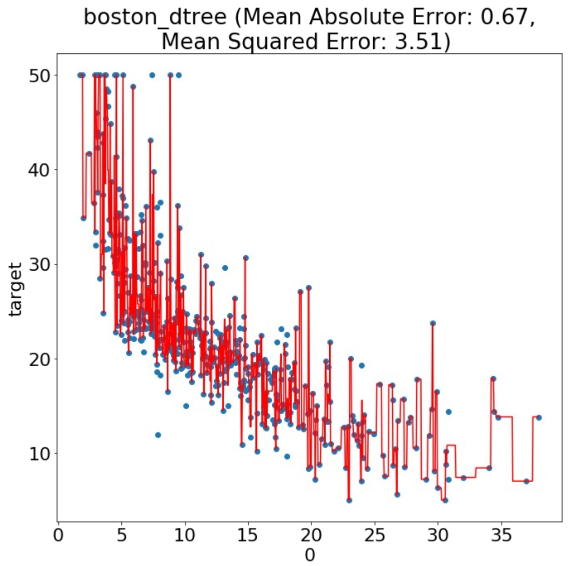


Introduction to Reinforcement Learning

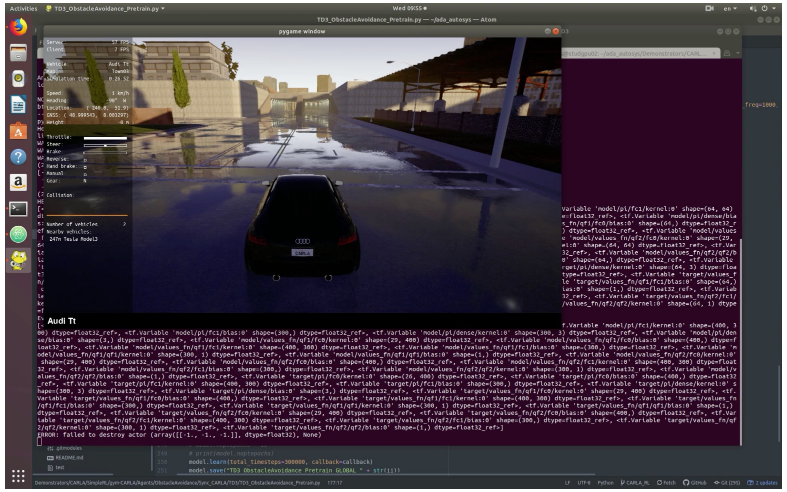
Challenges of understanding/adopting RL

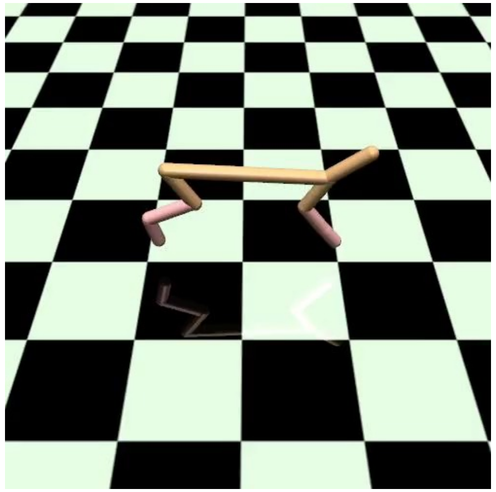
- Example: what went wrong here?

Supervised Learning



Reinforcement Learning





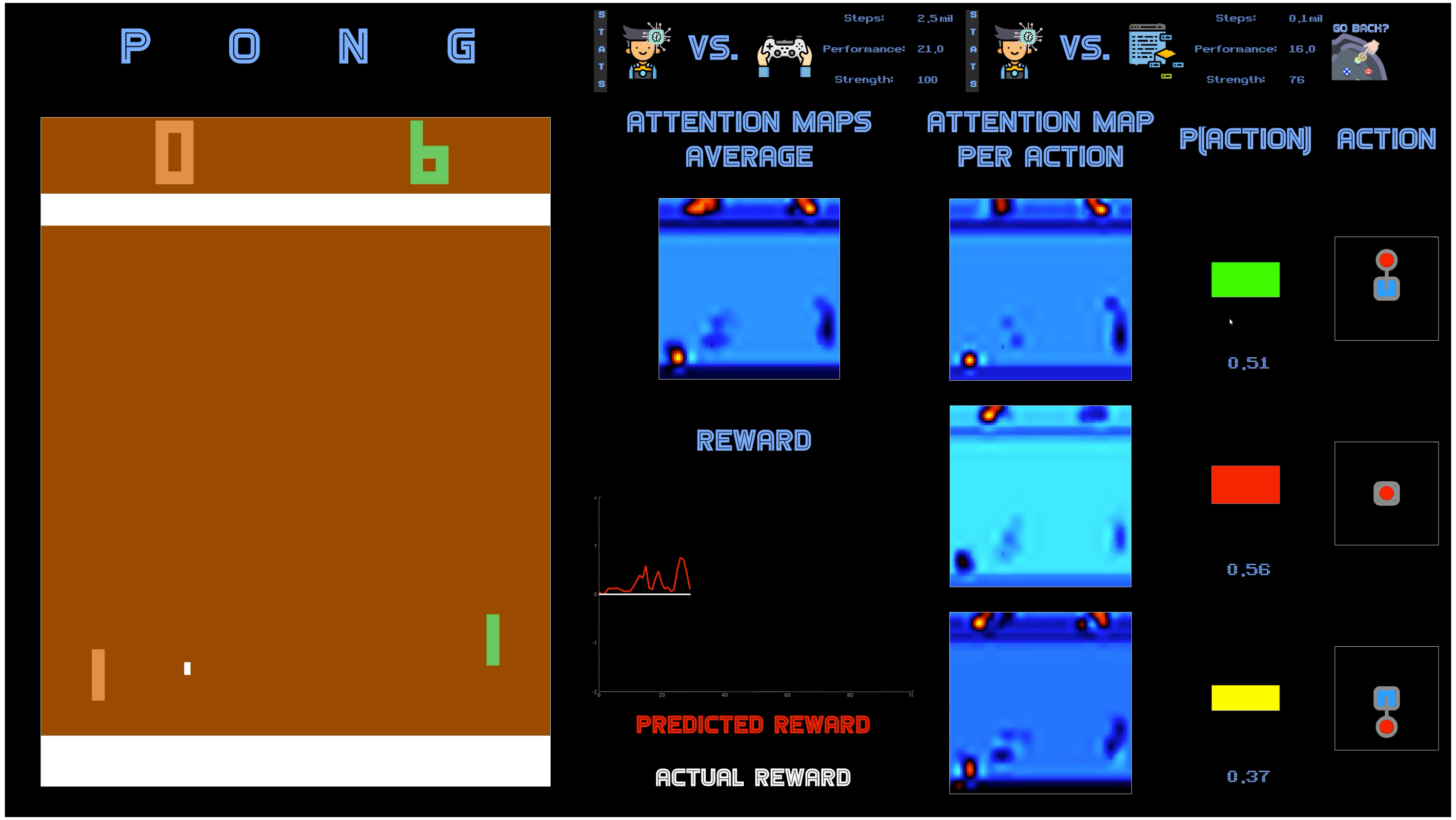
http://ai.berkeley.edu/lecture_slides.html

Fraunhofer IIS

Introduction to Reinforcement Learning

Challenges of understanding/adopting RL

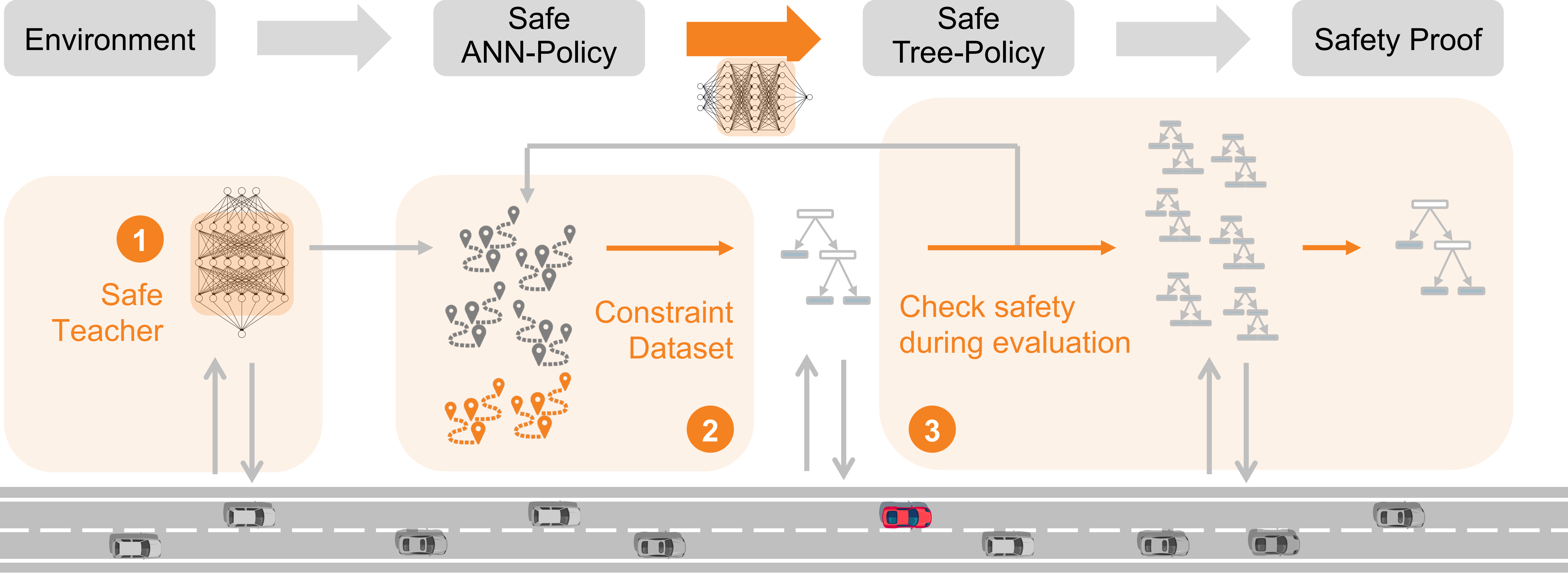
- Idea: Saliency Maps



Introduction to Reinforcement Learning

Challenges of understanding/adopting RL

- Idea: explainable decision rules



Fraunhofer IIS

Introduction to Reinforcement Learning

Challenges of understanding/adopting RL

- Simple algorithms don't scale!!!
  - k-means → time-series clustering
  - Linear/polynomial regression → house/car pricing prediction
  - Tabular Q-Learning/SARSA → very specialized applications

Introduction to Reinforcement Learning

Myth vs. Reality

1. AI is RL
   → NO! Many AI methods exist

2. RL can solve only games
   → NO! We will see several examples

3. RL is just "fancy" search
   → NO! We will compare to fancy search methods and see this

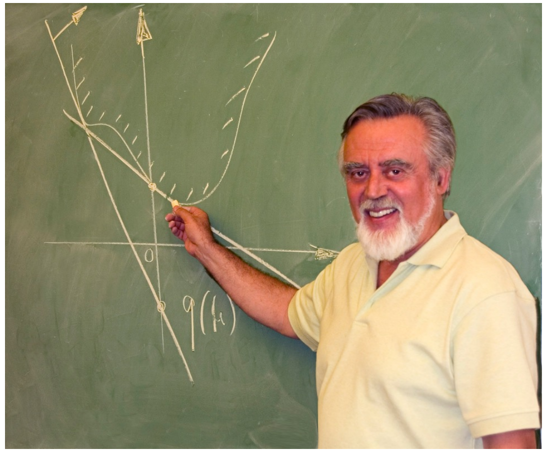
4. (Deep) RL can solve any problem, without any domain knowledge
   → NO!

Introduction to Reinforcement Learning

Myth vs. Reality

- Deep RL can solve anything vs Deep RL does not work
  (see https://www.alexirpan.com/2018/02/14/rl-hard.html)

  NO and NO!

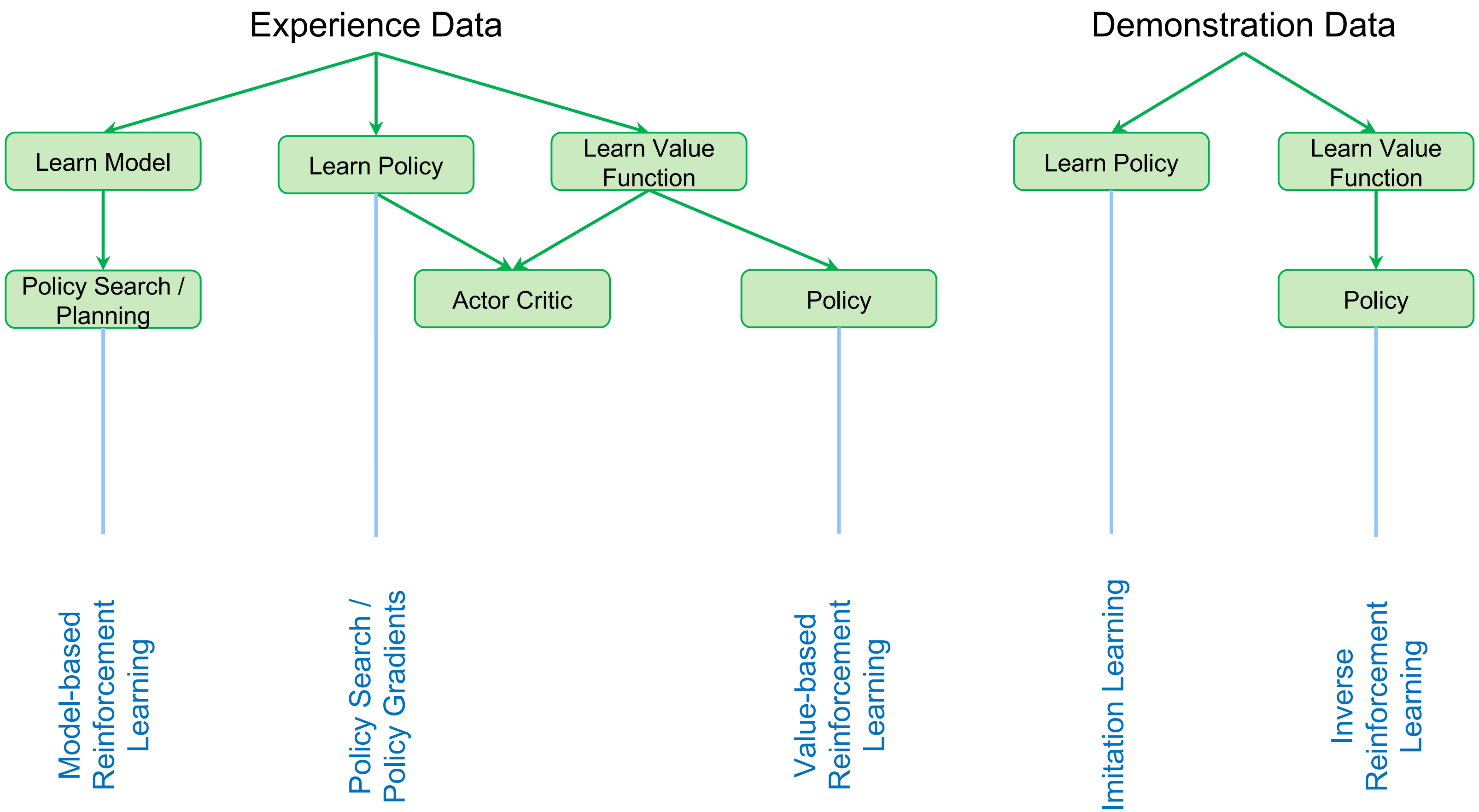


Bertsekas, 2019:

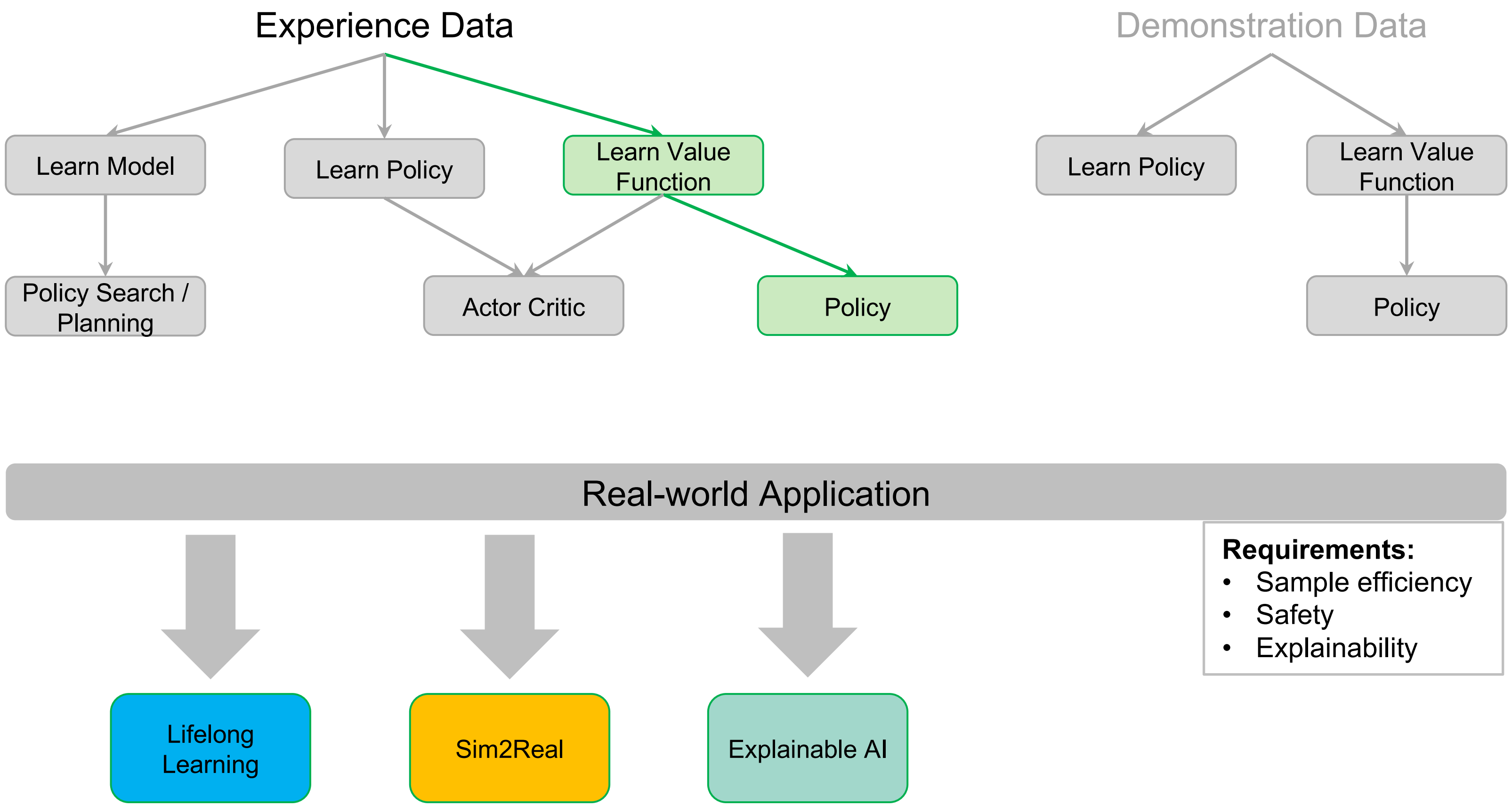
State of the art:

- Broadly applicable methodology: Can address a very broad range of challenging problems. Deterministic-stochastic-dynamic, discrete-continuous, games, etc
- There are no methods that are guaranteed to work for all or even most problems
- There are enough methods to try with a reasonable chance of success for most types of optimization problems
- Role of the theory: Guide the art, delineate the sound ideas

Introduction to Reinforcement Learning

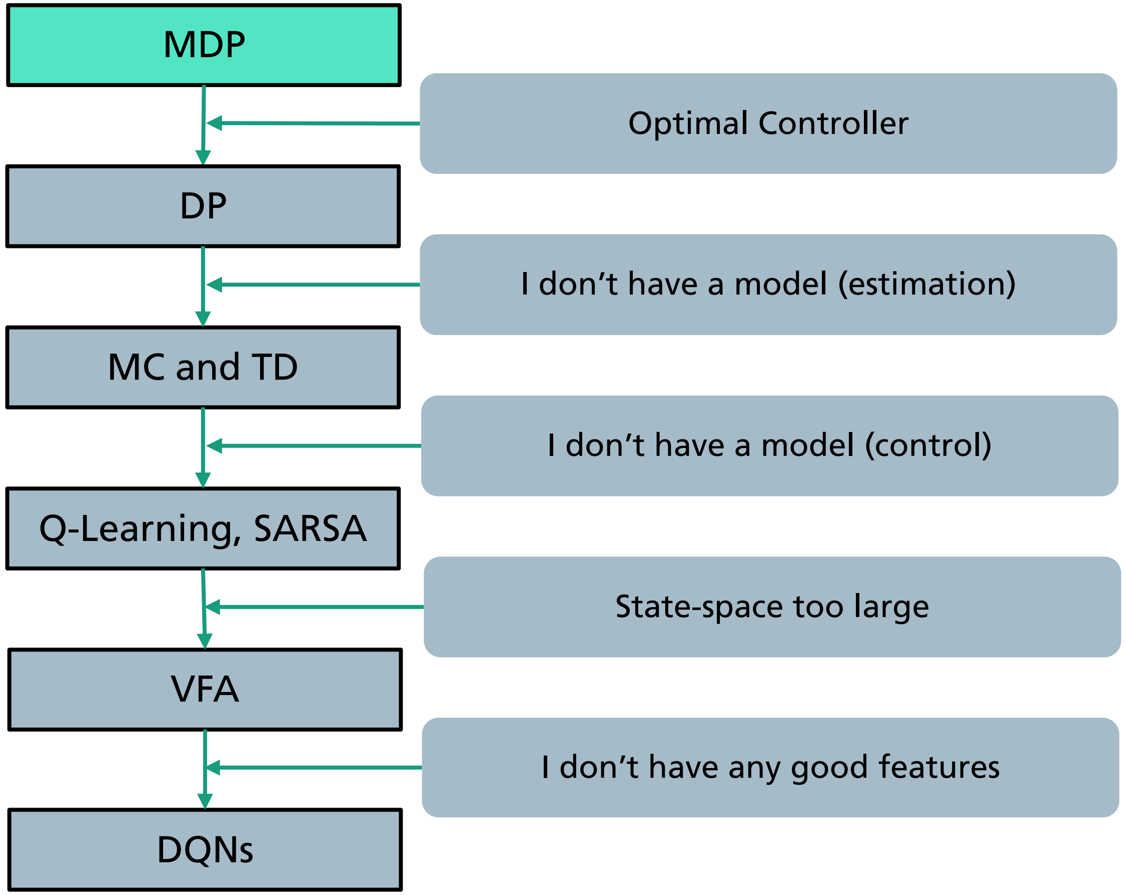


Introduction to Reinforcement Learning



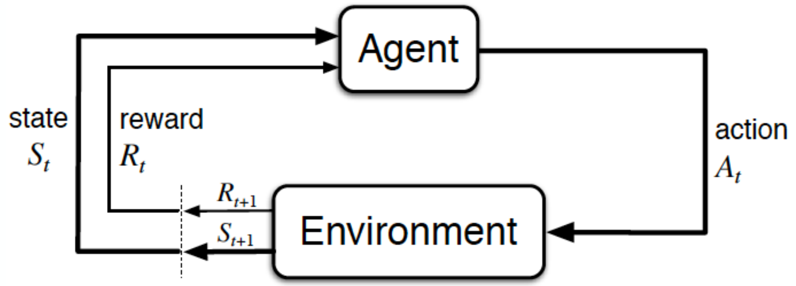
Markov Decision Processes

Overview



Markov Decision Processes

- Agent learns by interacting with an environment over many time-steps:
- Markov Decision Process (MDP) is a tool to formulate RL problems
  - Description of an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:



Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

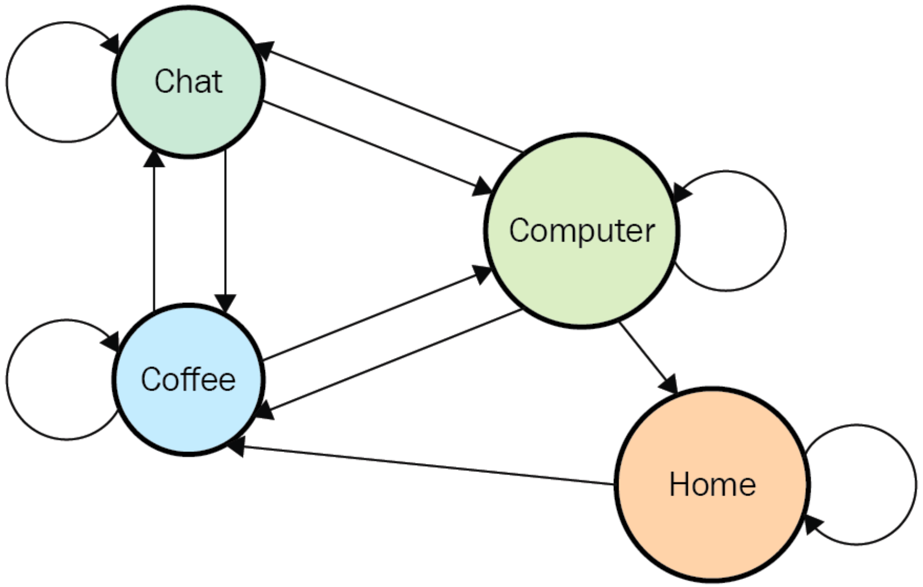
Note:
If the interaction does stop at some point in time (T) then we have an *episodic RL problem.*

- At each step t, the agent:
  - is at state S_t,
  - performs action A_t,
  - receives reward R_t.

- At each step t, the environment:
  - receives action A_t from the agent,
  - provides reward R_t,
  - moves at state S_{t+1},
  - increments time $t \leftarrow t + 1$.

Markov Decision Processes

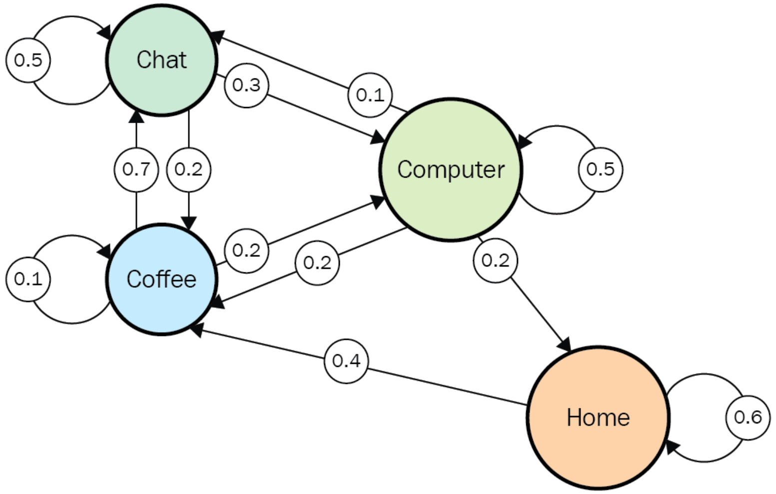
- Markov Process (MP)
  - Description of an MP $(\mathcal{S}, \mathcal{P})$:



Lapan, M. (2018). Deep Reinforcement Learning Hands-On. Packt Publishing Ltd.

Markov Decision Processes

- Markov Process (MP)
  - Description of an MP $(\mathcal{S}, \mathcal{P})$:



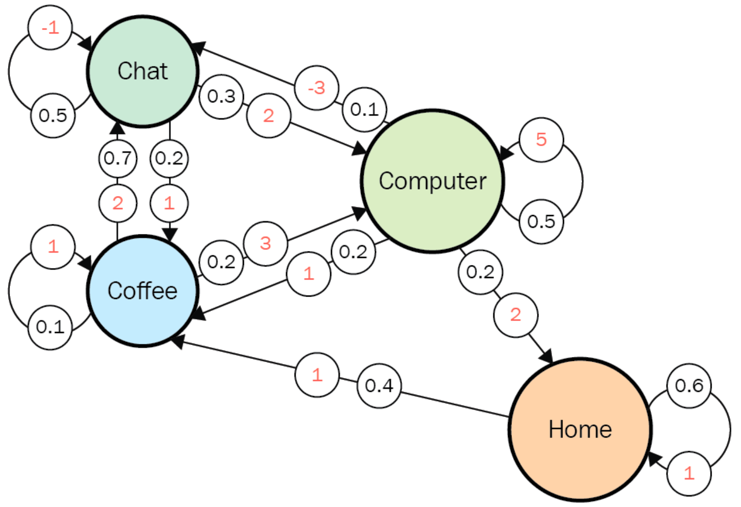
	Home	Coffee	Chat	Computer
Home	60%	40%	0%	0%
Coffee	0%	10%	70%	20%
Chat	0%	20%	50%	30%
Computer	20%	20%	10%	50%

Lapan, M. (2018). Deep Reinforcement Learning Hands-On. Packt Publishing Ltd.

Markov Decision Processes

- Markov Reward Process (MRP)
  - Description of an MRP $(\mathcal{S}, \mathcal{P}, \mathcal{R})$:
  - $\mathcal{R}$ is a reward function:

$$\mathcal{R}_S = \mathbb{E}[R_{t+1} | \, S_t = s]$$

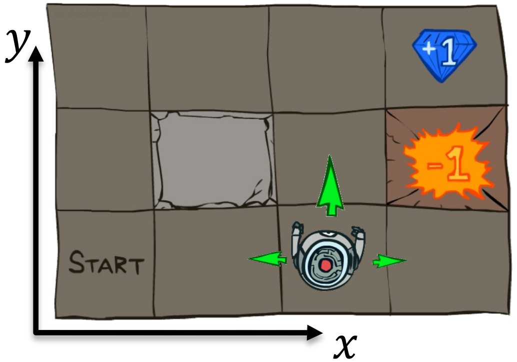


	Home	Coffee	Chat	Computer
Home	1	1		
Coffee		1	2	3
Chat		1	-1	2
Computer	2	1	-3	5

Lapan, M. (2018). Deep Reinforcement Learning Hands-On. Packt Publishing Ltd.

Markov Decision Processes

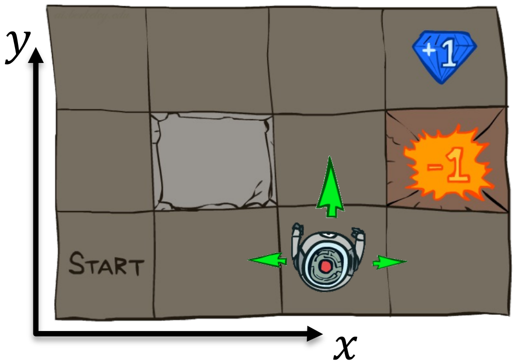
- Markov Decision Process (MDP)
  - Description of an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:



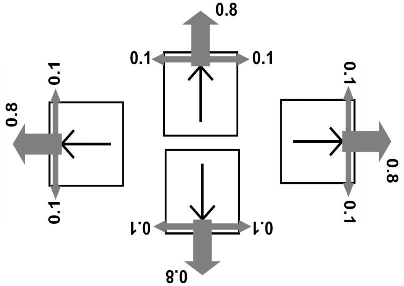
http://ai.berkeley.edu/lecture_slides.html

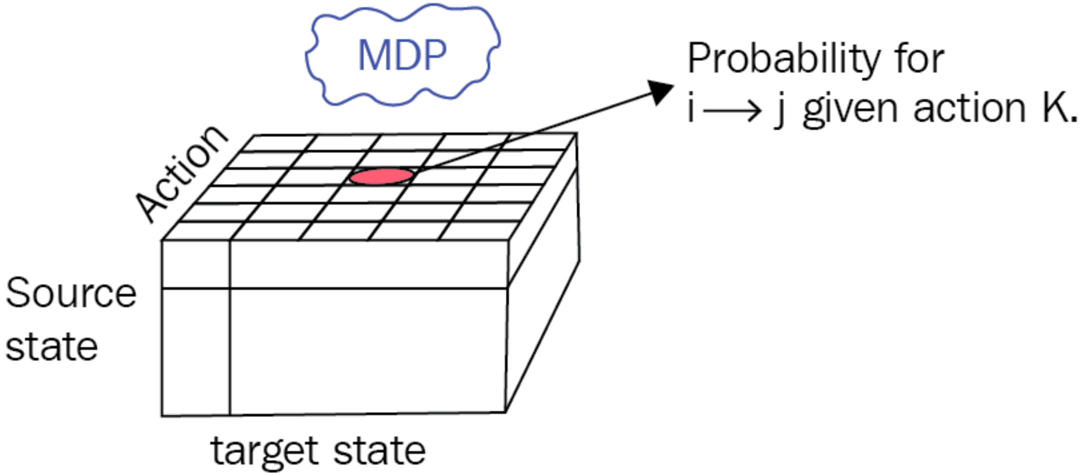
Markov Decision Processes

- Markov Decision Process (MDP)
  - Description of an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:
  - State transition model:
    - A state transition probability matrix $\mathcal{P}$ helps to model the true state transition function $T(S_{t+1}|\, S_t, A_t)$ of a real-world environment.
    - For each action $A^i \in \mathcal{A}$, we have a state transition matrix $\mathcal{P}^{A^i}$ at any time-step t



http://ai.berkeley.edu/lecture_slides.html





Lapan, M. (2018). Deep Reinforcement Learning Hands-On. Packt Publishing Ltd.

Markov Decision Processes

- Markov Decision Process (MDP)
  - Description of an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:
  - State transition model:
    - A state transition probability matrix $\mathcal{P}$ helps to model the true state transition function $T(S_{t+1}|\, S_t, A_t)$ of a real-world environment.
    - For each action $A^i \in \mathcal{A}$, we have a state transition matrix $\mathcal{P}^{A^i}$ at any time-step t as follows:

$$\begin{bmatrix} \mathcal{P}_{11} & \cdots & \mathcal{P}_{1n} \\ \vdots & \ddots & \vdots \\ \mathcal{P}_{n1} & \cdots & \mathcal{P}_{nn} \end{bmatrix}$$

Notes:
- Rows sum up to 1.0.
- $\mathcal{P}$ could change over time.

Markov Decision Processes

about the state space $\mathcal{S}$

- History is the sequence of observations, actions, rewards:

$$H_t = O_0, A_0, R_0, O_1, A_1, R_1, O_2, \dots, O_{t-1}, A_{t-1}, R_{t-1}, O_t$$

- 3 different definitions of s_t
  - **(Full) Environmental state S_t^e**
    - Private to the environment, not visible, maybe irrelevant information
    - Uses H_t to pick observation and reward
  - **Agent state S_t^a (actually used)**
    - Private to the agent, history of observations, rewards, and actions
    - Uses function of history $S_t^a = f(H_t)$ to select next action
  - **Information state (we will define it soon)**
    - Basically, S_t^a with special constraints in $f(H_t)$

Fraunhofer IIS

Markov Decision Processes

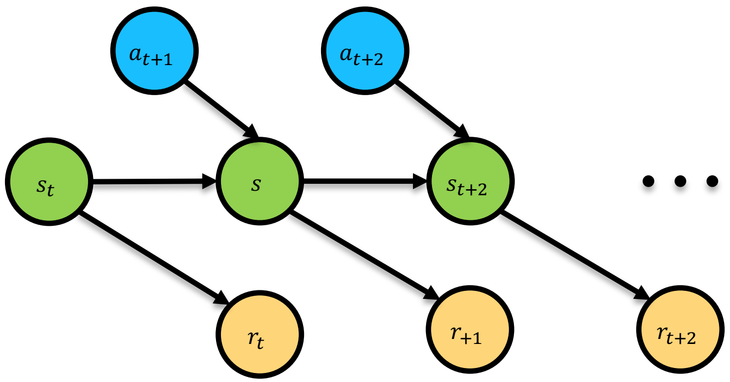
about the state space $\mathcal{S}$

- Assumption of MDPs: Markov Property
  - A state S_t is Markov if and only if

$$\mathbb{P}[S_{t+1} \mid S_1, \ldots, S_{t-1}, S_t] = \mathbb{P}[S_{t+1} \mid S_t]$$

  - Past states $S_1, \ldots, S_{t-1}$ do not change the outcome for the next state S_{t+1}.
  - The current state S_t captures all relevant information from the history.
  - **"The future is independent of the past given the present"**

$$H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty}$$

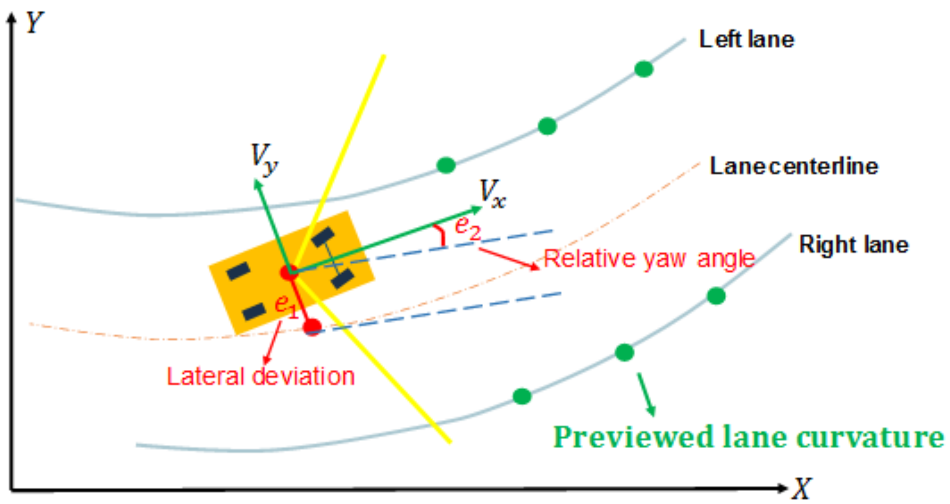


- State is the information used to determine what happens next
  - Direct (fully observable): $O_t = S_t^e$
  - Indirect (partially observable): $O_t = f(S_t^e)$

Markov Decision Processes

about the state space $\mathcal{S}$

- Assumption of MDPs: Markov Property
  - How can we ensure/construct such a Markov state?



Sensor Measurements:
- Speed, Angle

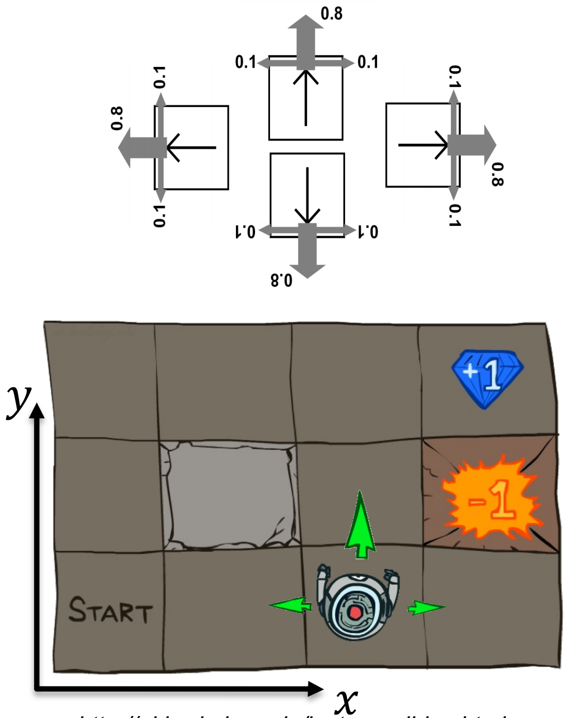
Requirements:
- Lateral acceleration
- Angular velocity

Fraunhofer IIS

Markov Decision Processes

about the action space $\mathcal{A}$

- MDP example: Gridworld, episodic task



http://ai.berkeley.edu/lecture_slides.html

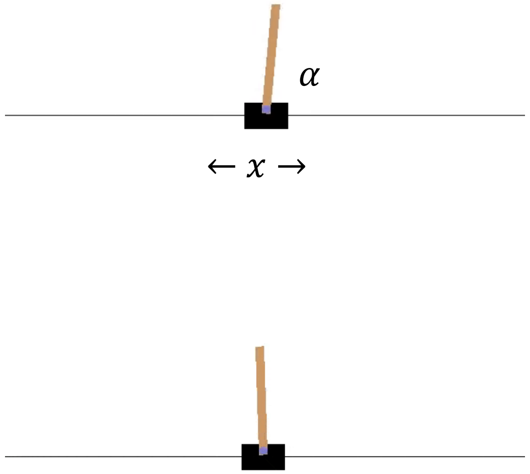
	Values
$\mathcal{S}$	(x, y) with $x \in \{0, 1, 2, 3\}$ and $y \in \{0, 1, 2\}$
$\mathcal{A}$	LEFT, RIGHT, UP, DOWN,

Fraunhofer IIS

Markov Decision Processes ($\mathcal{A}$)

about the action space $\mathcal{A}$

- MDP example: Cartpole, episodic or continuing task

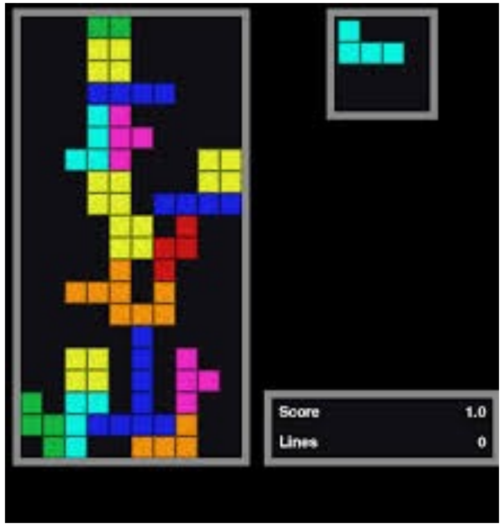


	Values
$\mathcal{S}$	$(x, \theta, \dot{x}, \dot{\theta})$ with $x \in \mathbb{R}$ and $\alpha \in [0°, 360°]$
$\mathcal{A}$	LEFT, RIGHT

Fraunhofer IIS

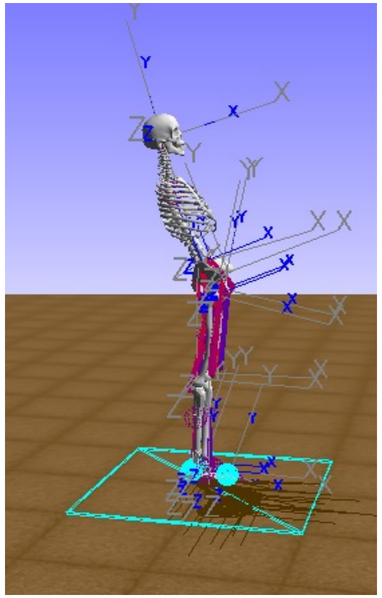
Markov Decision Processes

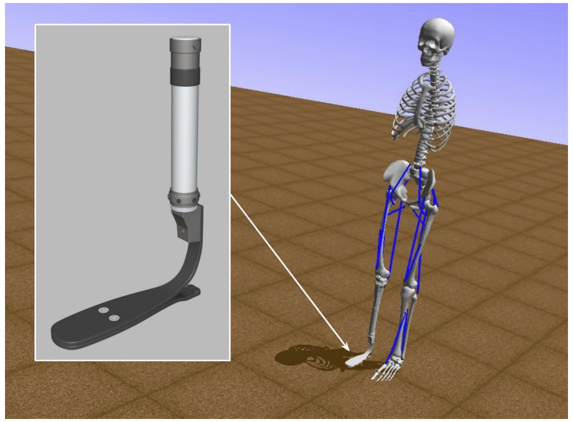
MDP example: Tetris, *episodic task*



Markov Decision Processes

MDP example: Running with a prosthetic leg, *episodic task*



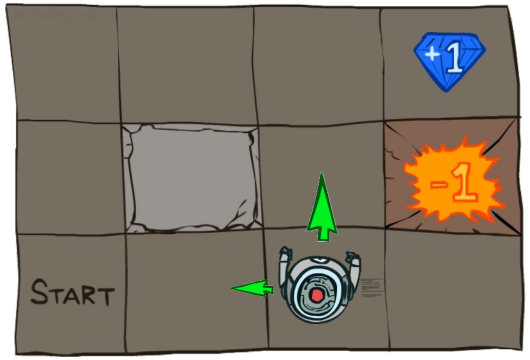


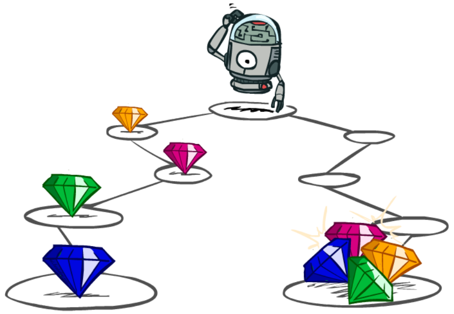
# of muscles	19
# degrees of freedom	14
reward	negative distance from requested velocity

Markov Decision Processes

- Markov Decision Process (MDP) is a tool to formulate RL problems
  - Description of MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$
  - Recall: Actions have consequences!
  - Choosing an action $A^i \in \mathcal{A}$ for A_t at timestep t yields different reward sequences
  - How do we know which sequence to prefer?

  - Idea: Decay value of rewards over time.
    - γ is a discount factor: $\gamma \in [0,1]$





http://ai.berkeley.edu/lecture_slides.html

Fraunhofer IIS

Markov Decision Processes

- We want to "solve" the MDP, by maximizing future rewards.
  - We see the episodes in the form of

$$S_0 \xrightarrow{(A_0, R_0)} S_1 \xrightarrow{(A_1, R_1)} S_2 \xrightarrow{(A_2, R_2)} S_3 \ldots \; S_{t-1} \xrightarrow{(A_{t-1}, R_{t-1})} S_t$$

  - **Question:** what happens if our problem never stops (i.e., $T = \infty$)?
    - Examples: data center cooling, recommender systems, etc.

  - Total discounted (γ) reward (**return**) (of one sample)

$$\mathrm{G} = R_0 + \gamma R_1 + \gamma^2 R_2 + \gamma^3 R_3 + \cdots = \sum_{t=0}^{\infty} \gamma^t R_t$$

Markov Decision Processes

- Markov Decision Process (MDP) is a tool to formulate RL problems
  - Description of MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$

- **Why discount rewards with γ?**
  - Mathematically convenient to discount rewards (true reason).
  - Avoids infinite returns in non-episodic problems
    - Datacenter cooling
    - Recommender system
  - Uncertainty about the future may not be fully represented (model uncertainty, our model is not perfect).

- Can I use $\gamma = 1$?
  - Yes, if you have an episodic setting or you definitely know that there is a terminal absorbing state.
- Should I use $\gamma = 1$?
  - **NO!**

Markov Decision Processes

about the policy π

- Expected long-term value of state s:

$$v(s) = \mathbb{E}(\mathrm{G}) = \mathbb{E}(R_0 + \gamma R_1 + \gamma^2 R_2 + \gamma^3 R_3 + \ldots + \gamma^t R_t)$$

- **Goal: maximize the expected return $\mathbb{E}(\mathrm{G})$.**

- We need a controller that helps us select the actions to maximize $\mathbb{E}(G)$!
- A policy π represents this controller:
  - π determines the agent's behavior, i.e., its way of acting
  - π is a mapping from state space $\mathcal{S}$ to action space $\mathcal{A}$

$$\pi : \mathcal{S} \mapsto \mathcal{A}$$

  - Two types of policies:
    - Deterministic policy: $a = \pi(s)$.
    - Stochastic policy: $\pi(a \mid s) = \mathbb{P}[A_t = a \mid S_t = s]$.

- **New goal: find a policy that maximizes the expected return!**

Fraunhofer IIS

Markov Decision Processes

Some remarks about terminology

$\mathbf{s}_t$ – state
$\mathbf{a}_t$ – action
$r(\mathbf{s}, \mathbf{a})$ – reward function

$\mathbf{x}_t$ – state
$\mathbf{u}_t$ – action
$c(\mathbf{x}, \mathbf{u})$ – cost function

$$r(\mathbf{s}, \mathbf{a}) = -c(\mathbf{x}, \mathbf{u})$$

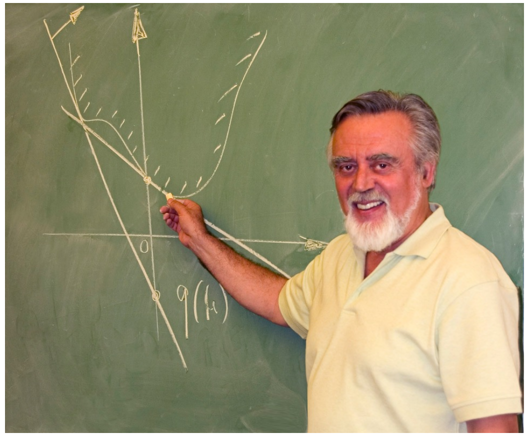
Richard Bellman

Lev Pontryagin

http://rail.eecs.berkeley.edu/deeprlcourse/static/slides/lec-2.pdf

Markov Decision Processes

Some remarks about terminology



Bertsekas, 2019:

RL uses Max/Value, DP uses Min/Cost

- Reward of a stage = (Opposite of) Cost of a stage.
- State value = (Opposite of) State cost.
- Value (or state-value) function = (Opposite of) Cost function.

Controlled system terminology

- Agent = Decision maker or controller.
- Action = Decision or control.
- Environment = Dynamic system.

Methods terminology

- Learning = Solving a DP-related problem using simulation.
- Self-learning (or self-play in the context of games) = Solving a DP problem using simulation-based policy iteration.
- Planning vs Learning distinction = Solving a DP problem with model-based vs model-free simulation.