Unit #1: Introduction to Reinforcement Learning

W26: Introduction to Reinforcement Learning (W26M-RELAD)

Opening Remarks

Class Logistics

- Communication
  - There are no scheduled office hours
  - We will use Canvas Course Q&A forum for main communication
  - Canvas announcements for general information

Please plan ahead when working on assignments. Questions sent shortly before a deadline, may not receive a response before the submission deadline. Immediate responses cannot be guaranteed.

- **Teaching team:**


Christopher Mutschler


Claire Vernade


Alexander Mattick


Gloria Desideri


Alexander von Rohr

Opening Remarks

Syllabus

Week	Lecture		Exercises	
1.	15.10.	Intro to RL, MDPs	21.10.	MDPs (no assignments)
2.	22.10.	Dynamic Programming	28.10.	*Graded Exercise Presentation: T.B.D.*
3.	29.10.	Model-free Prediction	04.11.	*Graded Exercise Presentation: T.B.D.*
4.	05.11.	Model-free Control I	11.11.	*Graded Exercise Presentation: T.B.D.*
5.	12.11.	Model-free Control II	18.11.	*Graded Exercise Presentation: T.B.D.*
6.	19.11.	*Self-Study and Preparation for Mid-Term Assessment*	25.11.	**Mid-term Assessment**
7.	26.11	Policy Gradient Methods I	02.12.	*Graded Exercise Presentation: T.B.D.*
8.	03.12.	Policy Gradient Methods II	09.12.	*Graded Exercise Presentation: T.B.D.*
9.	10.12.	Policy Gradient Methods III	16.12.	*Graded Exercise Presentation: T.B.D.*
10.	17.12.	Guest Lecture T.B.D.	23.12.	*Graded Exercise Presentation: T.B.D.*
Christmas break				
11.	07.01.	Model-based RL I	13.01.	*Graded Exercise Presentation: T.B.D.*
12.	14.01.	Model-based RL II	20.01.	*Graded Exercise Presentation: T.B.D.*
13.	21.01.	Introduction to Exploration	27.01.	*Graded Exercise Presentation: T.B.D.*
14.	28.01.	Exploration in Deep RL and Modern RL Theory	03.02.	*Graded Exercise Presentation: T.B.D.*
15.	04.02.	Q&A		

Opening Remarks

Assessment Overview

Unit	Assignment	Submission mode	Assignment type	Points	Weight to the final grade	Learning outcome(s) evaluated
3	Mid-term assessment	Individual	Written	20	20%	LO1, LO2, LO3
depends on group	Graded Exercise Presentation	Group	Graded exercise presentation	20	20%	LO2 - LO7
	Oral examination	Individual	Oral	60	60%	LO1 - LO7

- Graded Exercise Presentation
  - Students present exercise assignments as a group, each students should present a dedicated sub-part
  - TAs will ask follow-up questions
  - Scheduling of groups + more information on Oct 21
- Final oral exam: scheduling and logistics will be finalized around January
  - In-person during the weeks February 15 - 19 and February 22 - 26.
  - Oral exam can be in English
  - It will cover topics from lectures and exercises
  - Exams will focus on core concepts, algorithmic ideas, and theory

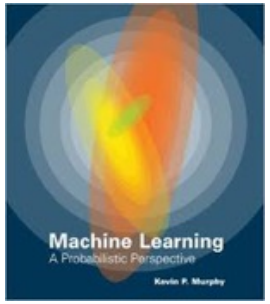
Opening Remarks

Prerequisites

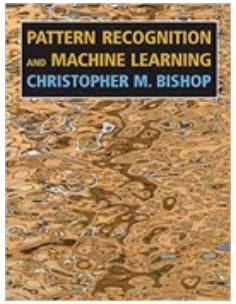
- What pre-requisites do we expect?
  - Analysis/Calculus
  - Multivariate Statistics
  - Machine Learning, Deep Learning
  - Python (to get used to the exercises)

Don’t be too casual about it!

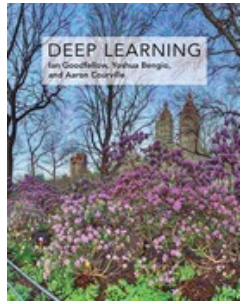
- How can you obtain them?
  - Dive into one of those books (better today than tomorrow):



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 on some later slides today

Introduction to Reinforcement Learning

Playing games with RL

UTN

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

```python
from stable_baselines.common.cmd_util import make_atari_env
from stable_baselines.common.vec_env import VecFrameStack
from stable_baselines import ACER

# There already exists an environment generator
# that will make and wrap atari environments correctly.
# Here we are also multiprocessing training (num_env=4 => 4 processes)
env = make_atari_env('PongNoFrameskip-v4', num_env=4, seed=0)
# Frame-stacking with 4 frames
env = VecFrameStack(env, n_stack=4)

model = ACER('CnnPolicy', env, verbose=1)
model.learn(total_timesteps=25000)

obs = env.reset()
while True:
    action, _states = model.predict(obs)
    obs, rewards, dones, info = env.step(action)
    env.render()
```



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

Introduction to Reinforcement Learning

Controlling robots with RL



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



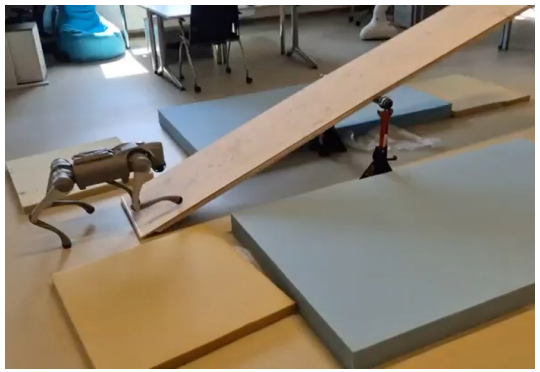
https://www.youtube.com/watch?v=W_gxLKSsSIE

Introduction to Reinforcement Learning

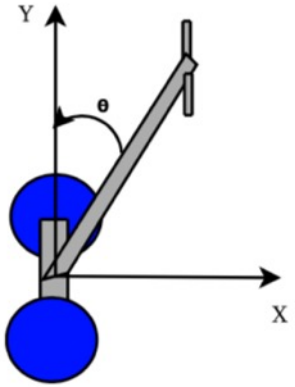
Controlling robots with RL

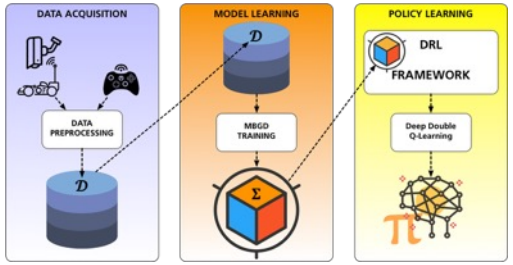


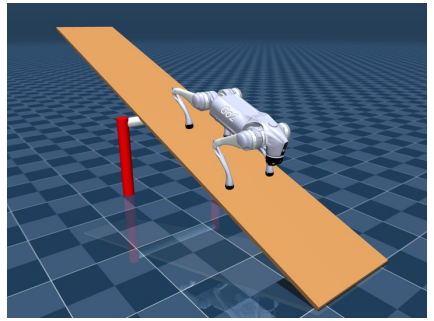








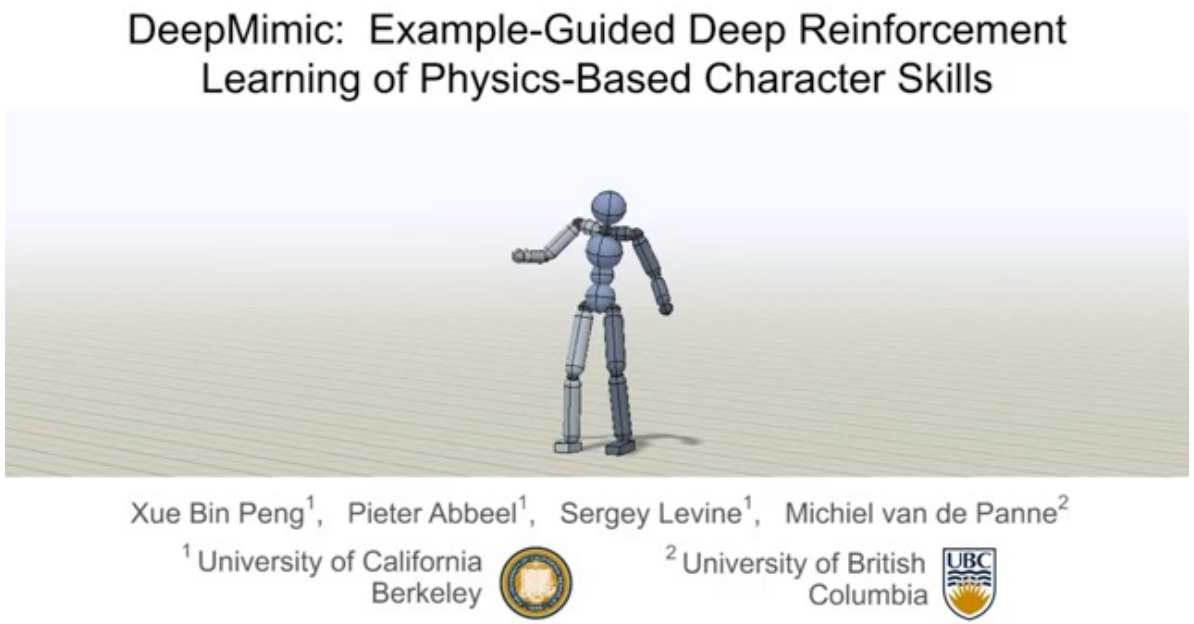




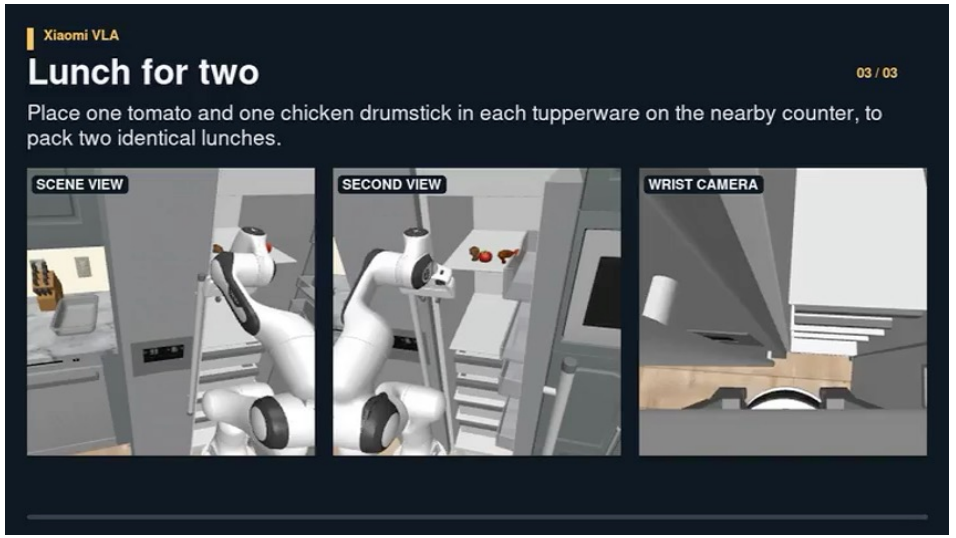


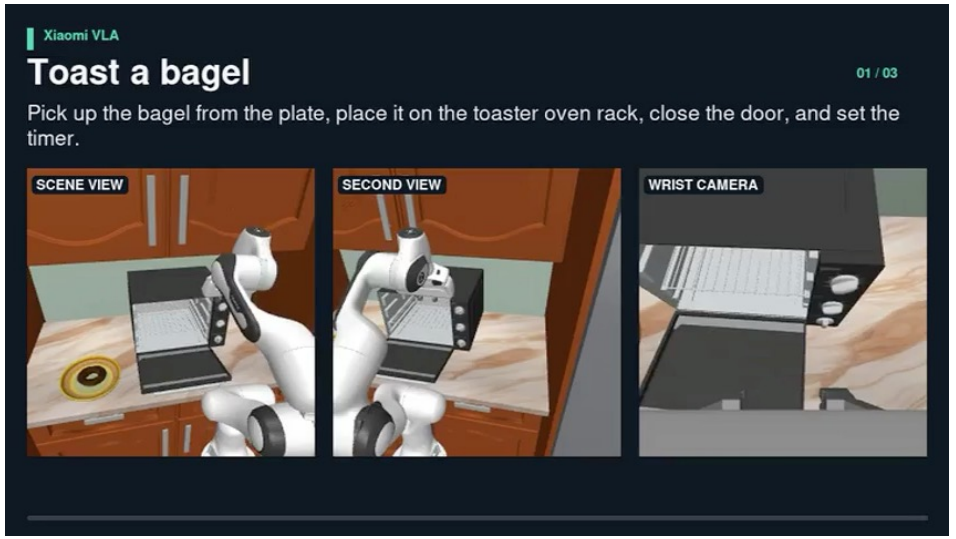
Introduction to Reinforcement Learning

Advanced robot control in simulation



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





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



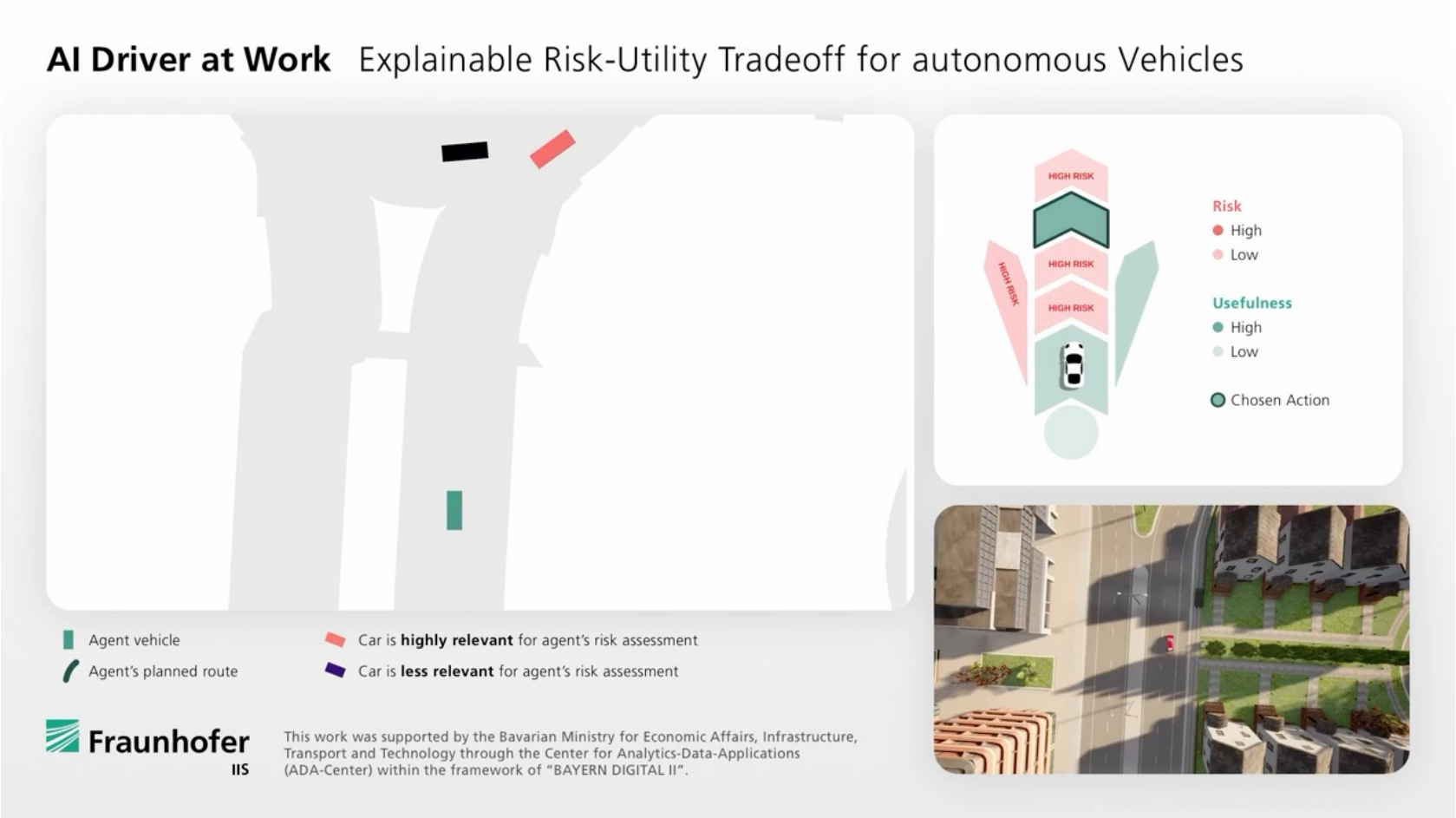
https://www.youtube *atch?v=-e1_QhJ1EhQ*

Walk, Run, Crawl, RL Fun | Boston Dynamics | Atlas

https://www.youtube.com/watch?v=I44_zbEwz_w

Introduction to Reinforcement Learning

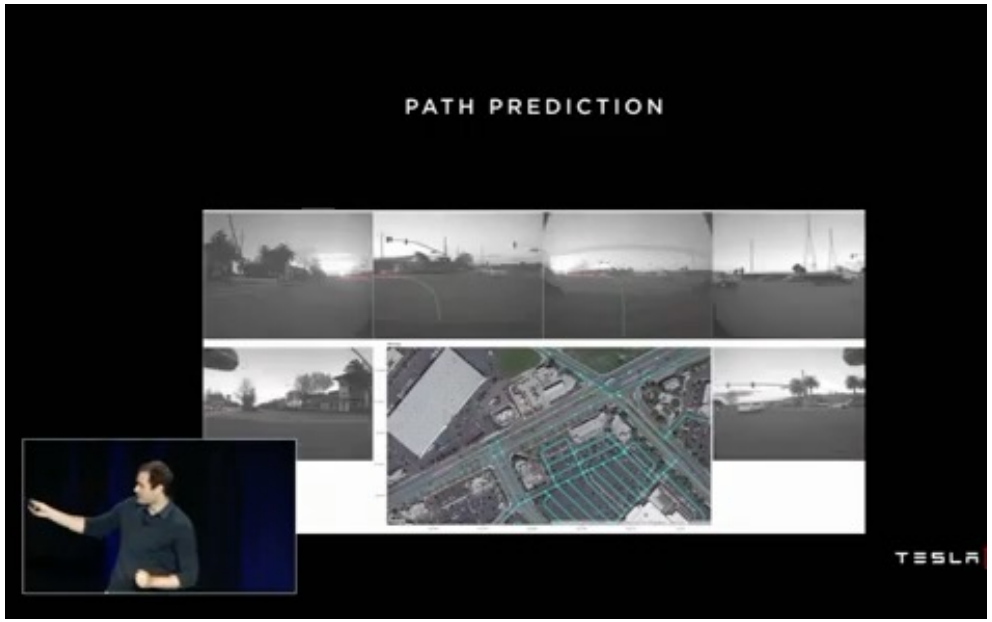
Driving cars with RL



https://youtu.be/0IWjE_8xj6Q

Introduction to Reinforcement Learning

Path planning/navigation



https://www.youtube.com/watch?v=v5l-jPsAK7k



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

Introduction to Reinforcement Learning

Practical Implementations



Google DeepMind

Energy efficiency at Google's data centers

- Every 5 minutes, AI draws snapshots of data center cooling system through thousands of sensors
- Information fed into deep neural network, →defines optimal action to reduce energy usage while keeping data center reliable
- Actions verified by second system and then implemented
- Significant energy and cost savings achieved

Source: https://deepmind.google/discover/blog/safety-first-ai-for-autonomous-data-centre-cooling-and-industrial-control/



DB

InstaDeep™

Optimizing Rail Network with RL

- In case of unexpected events, trains need to be redirected in real time
- Ensure optimal capacity planning and traffic management in real time
- RL trained with real data and then further improved with millions of simulations
- Through optimized capacity planning and traffic management
  → higher train frequencies
  → delays can be better avoided

Source: https://www.instadeep.com/2023/04/instadeep-explores-the-benefits-of-deep-reinforcement-learning-technology-in-transportation-at-interchange-event/



LOON

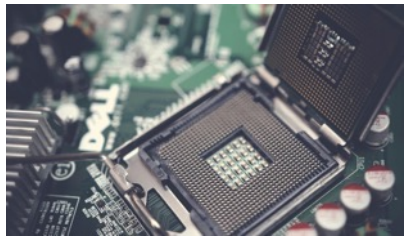
Ballons steering in stratosphere

- Ballons that autonomously operate in stratosphere for months serving various purposes
- Complex steering involves considering cues like wind speed, visibility, solar elevation, and (imperfect) weather forecasts
- RL trained with millions of simulated flight hours to make optimal real-time decisions
- RL surpasses previous algorithms and withstands natural diversity

Source: https://blog.x.company/drifting-efficiently-through-the-stratosphere-using-deep-reinforcement-learning-c38723ee2e90

Introduction to Reinforcement Learning

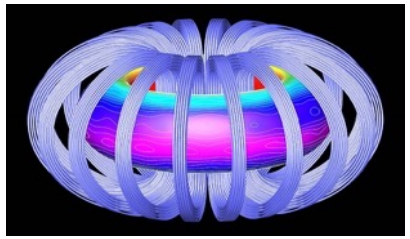
Practical Implementations



RL to optimize the manufacturing flow of semiconductors at Infineon

- RL-driven optimization of semiconductor manufacturing flow
- RL agent automates order dispatching in wafer fabrication
- Wafers automatically directed specific equipment for processing, considering limited capacity and storage space
- Ensures smooth flow with minimal delays
- Outperformed traditional heuristics

Source: https://www.sciencedirect.com/science/article/pii/S0007850618300659?via%3Dihub



Optimizing nuclear fusion with Deep RL at Swiss Plasma Center in Lausanne

- Utilization of RL for precise control of plasma (heated hydrogen) in tokamak for nuclear fusion
- Plasma is confined within a vacuum room with magnetic coils
- Plasma instability requires constant adjust-ment of magnetic coils & other conditions
- RL optimizes adjustments considering multiple conditions simultaneously
- Potential clean energy source

Sources: https://deepmind.google/discover/blog/accelerating-fusion-science-through-learned-plasma-control/; https://www.nature.com/articles/s41586-021-04301-9



RL to solve American Airlines inventory control & overbooking problems

- Utilization of RL to manage overbookings
- Considers multiple factors to maximize revenue and minimize bumping costs
- Ensures flights remain full despite cancellations
- RL agent surpasses traditional methods, yielding close to maximum profits per flight

Source: https://arxiv.org/pdf/1902.06824.pdf

Introduction to Reinforcement Learning

Practical Implementations



RL for inventory optimization and shortened lead times at Zara

- AI-driven system optimizes production and supply chain
- RL agent leverages sales data, customer feedback, and social media trends to predict demand
- RL-driven recommendations allow optimization of inventory levels due
- Shortened lead & delivery times enhance customer satisfaction by ensuring goods are available when needed

Source: https://www.tokinomo.com/blog/artificial-intelligence-in-retail



Optimizing crop management efficiency with open-source RL

- Farmers must balance variables for optimal crop production, e.g., weather, fertilization, & soil conditions
- RL agent simultaneously considers various factors to provide recommendations for optimal crop management
- Facilitates reaching production objectives while optimizing resource utilization
- Open-source RL environment provided by Inria centre at the Univ. of Lille

Source: https://arxiv.org/pdf/2207.03270v1.pdf



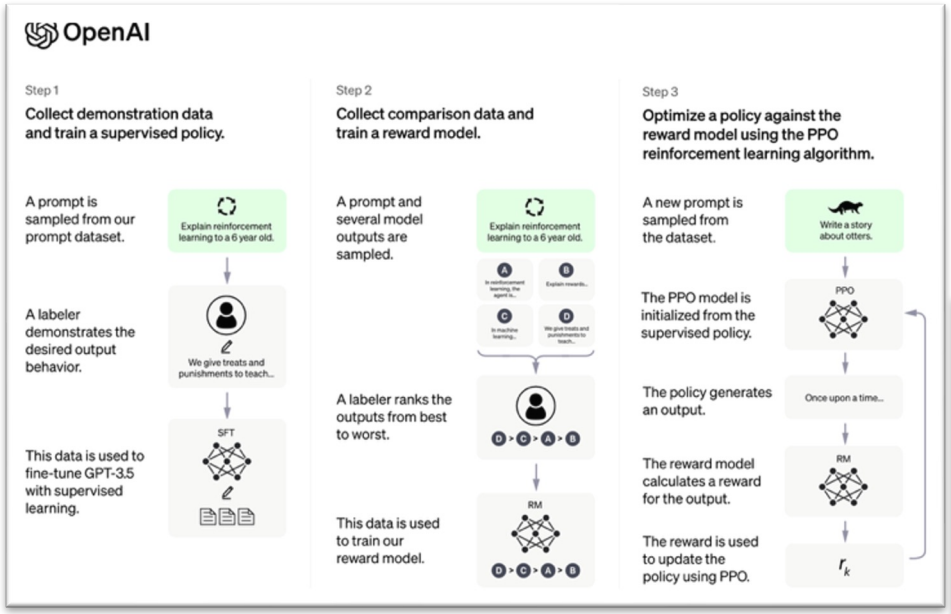
Utilization of deep RL to optimize energy costs in a flexible production machine

- RL employed to ensure most effective control policy for production machines considering varying framework conditions (e.g., energy prices)
- Deep learning architecture forecasts load profiles of future manufacturing schedules from past production time series
- RL algorithm trained to optimize machine load and speed for long-term energy cost reduction

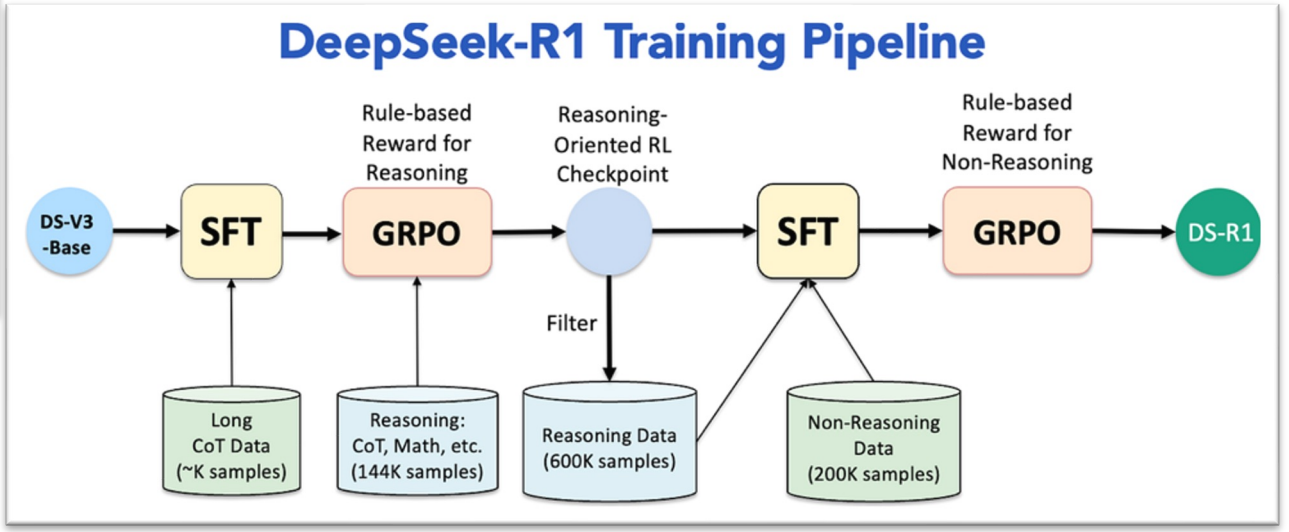
Source; https://www.scientific.net/AMM.882.96.pdf

Introduction to Reinforcement Learning

Practical Implementations



https://openai.com/blog/chatgpt



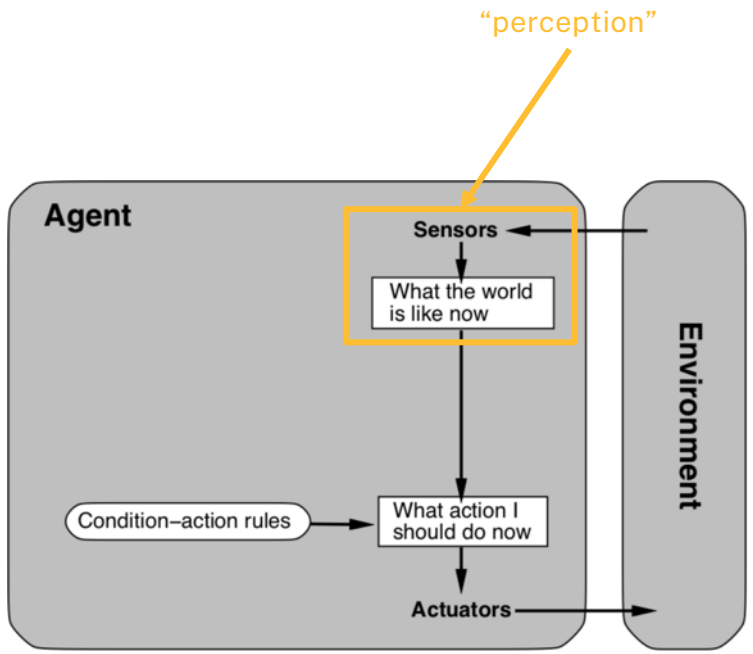
https://medium.com/@lmpo/deepseek-r1-affordable-efficient-and-state-of-the-art-ai-reasoning-f293b0bd8d65

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**



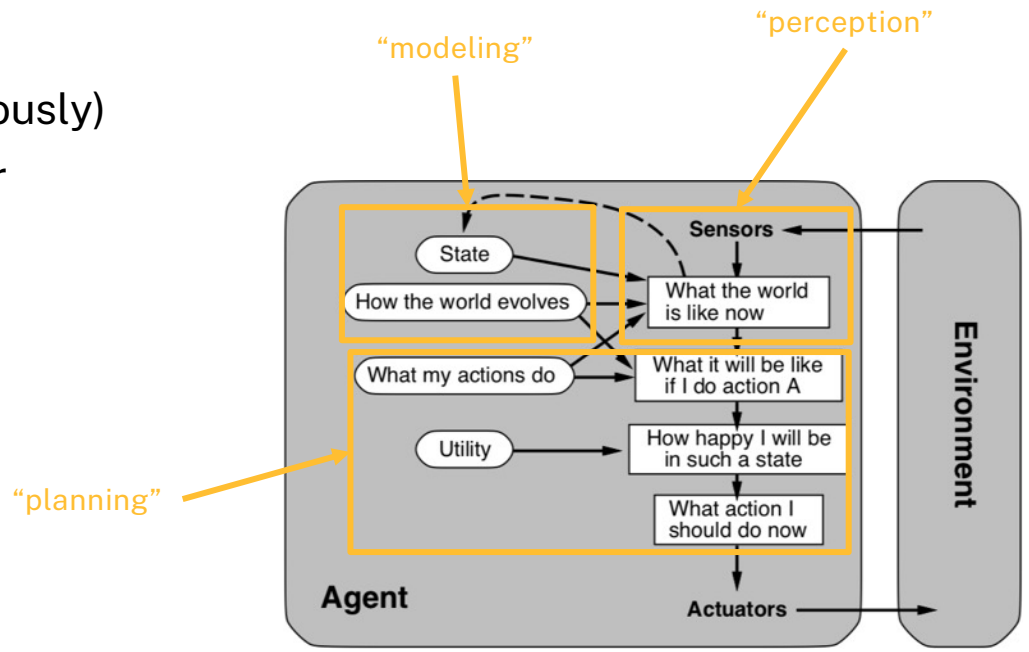
Russell, S. J., & Norvig, P. (2016). Artificial intelligence: A Modern Approach. Malaysia; Pearson Education Limited.

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**



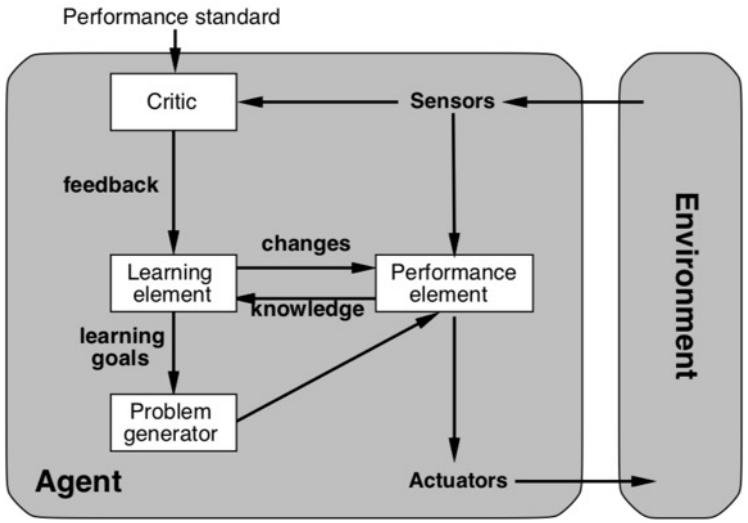
Russell, S. J., & Norvig, P. (2016). Artificial intelligence: A Modern Approach. Malaysia; Pearson Education Limited.

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”?

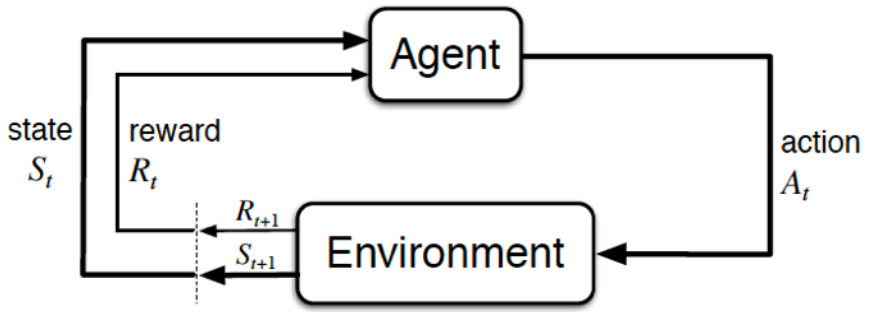
- Autonomous Cars
- Smart Homes/Buildings that adapt to occupants
- Intelligent traffic lights control
- Software trading agents
- Virtual assistants that manage appointments or answer emails automatically
- Recommender systems, e.g., for movies (Netflix), consumer products (Amazon), advertisements (Google), 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**.”*

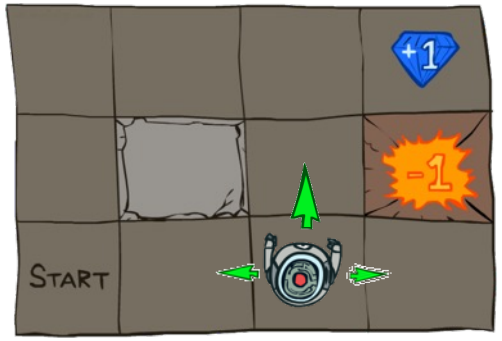


Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

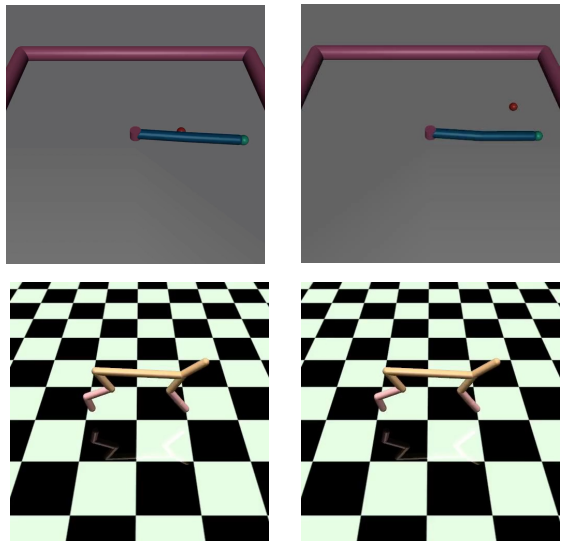
Introduction to Reinforcement Learning

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

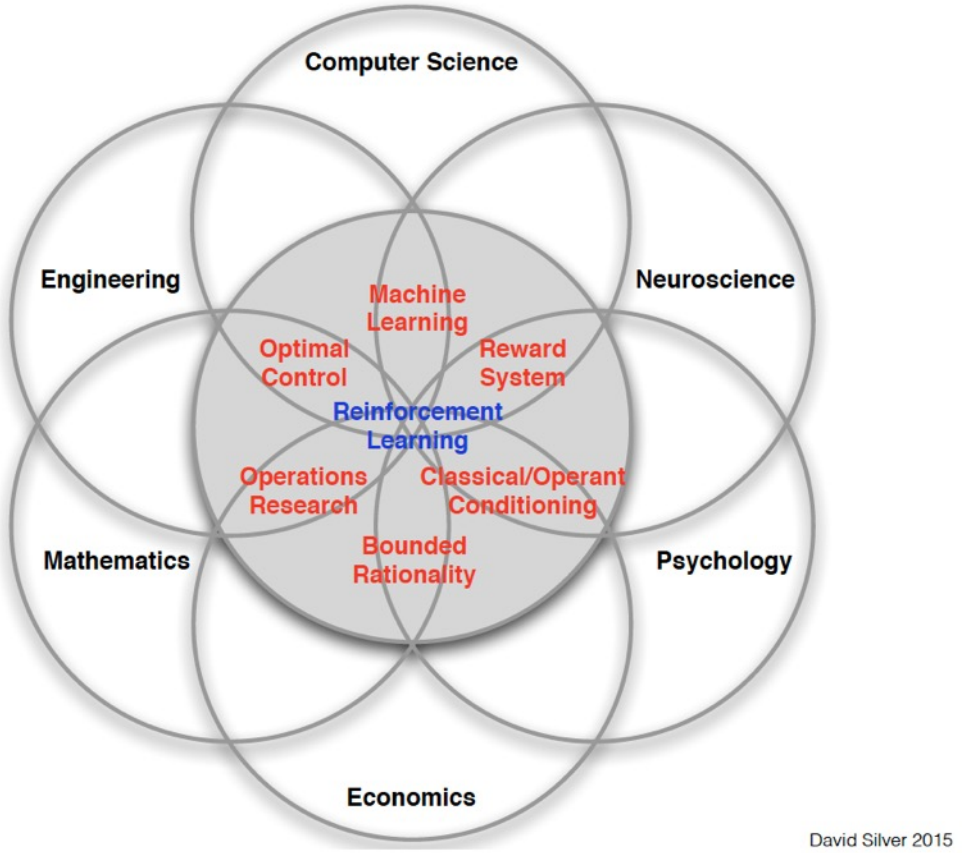


http://ai.berkeley.edu/lecture_slides.html



Introduction to Reinforcement Learning

RL vs the world: the many faces of Reinforcement Learning

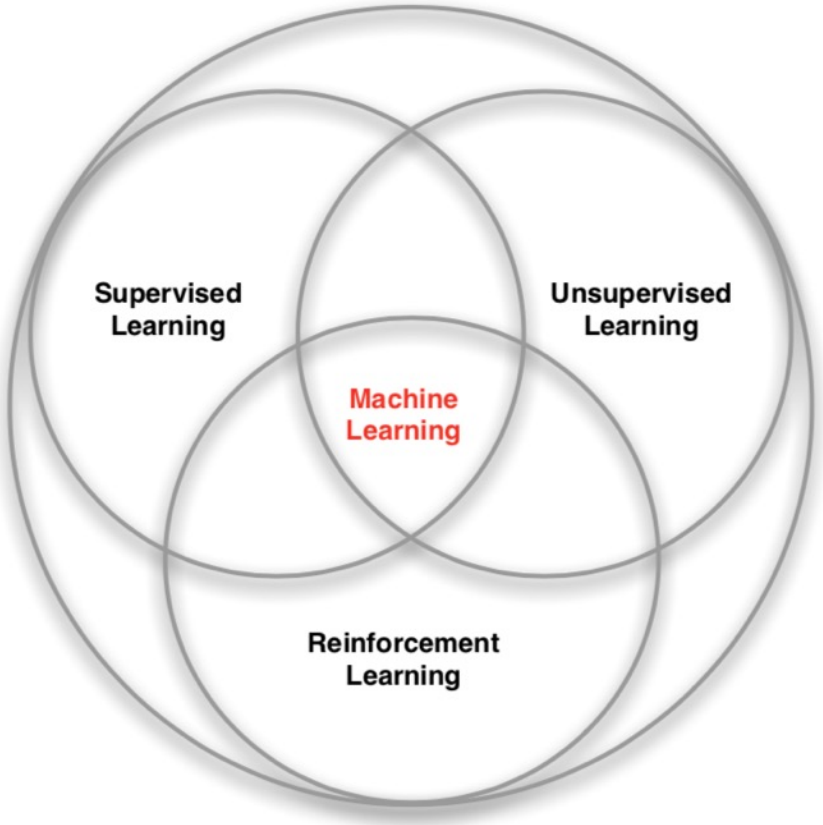


see also: https://www.youtube.com/watch?v=-63ysqT5nu0

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

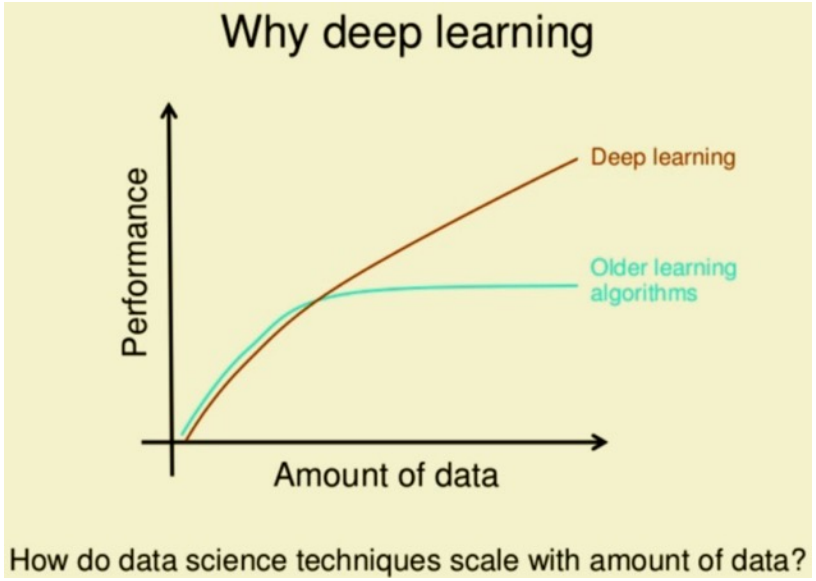


see also: https://www.youtube.com/watch?v=-63ysqT5nu0

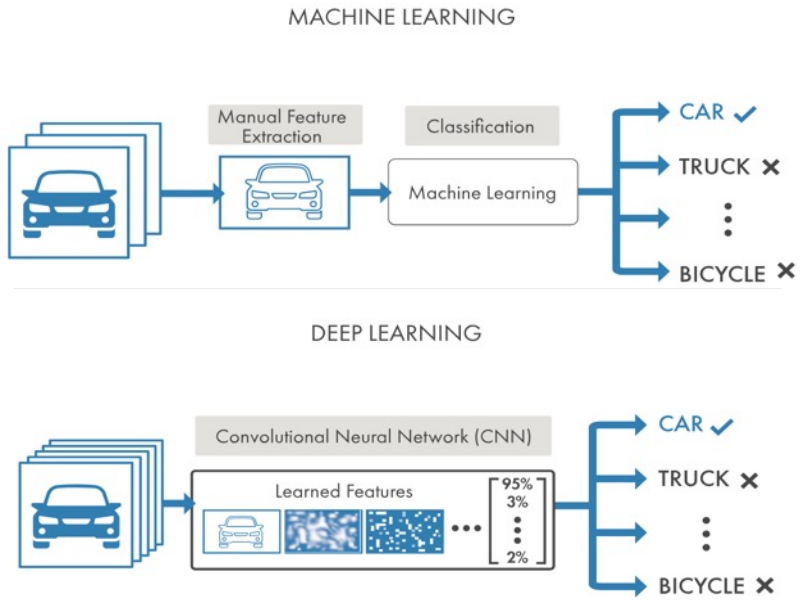
Introduction to Reinforcement Learning

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

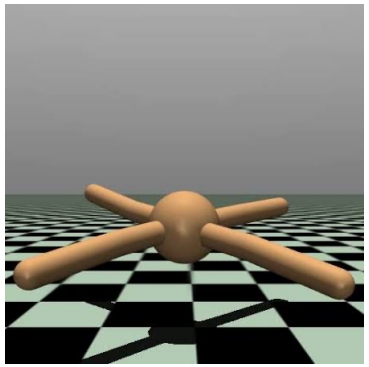
Introduction to Reinforcement Learning

Why RL now?

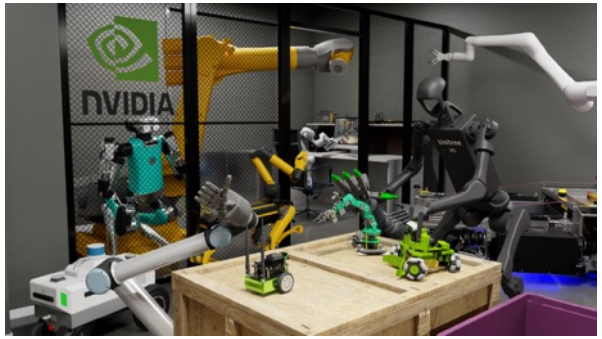
- Taking advantage of advances in:
  - Deep Learning Algorithms (DL)
  - **Software for DL and RL**

Gymnasium

https://github.com/farama-foundation/gymnasium



https://mujoco.org



https://developer.nvidia.com/isaac/lab

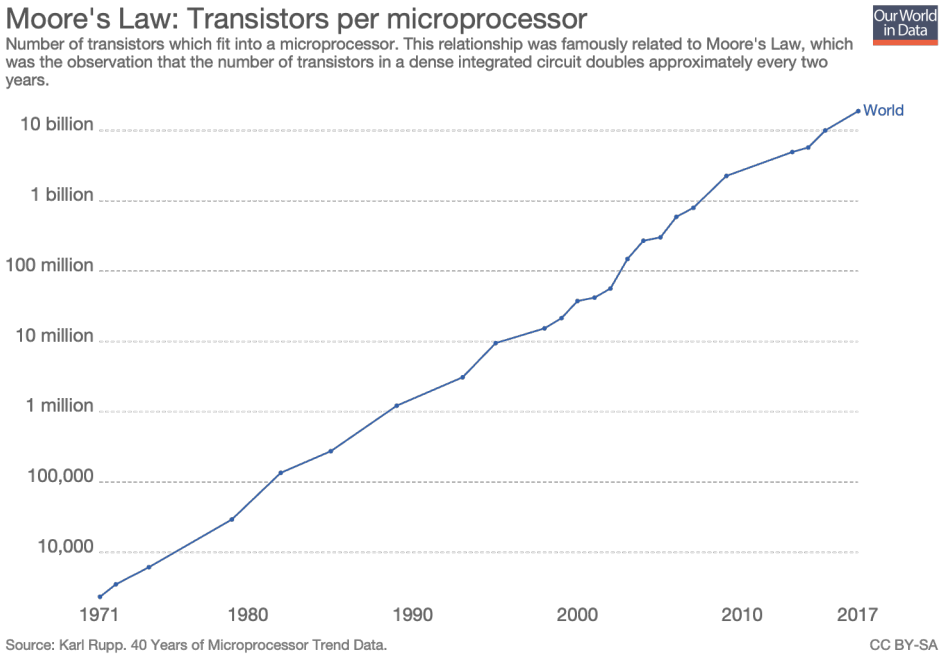
Introduction to Reinforcement Learning

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/

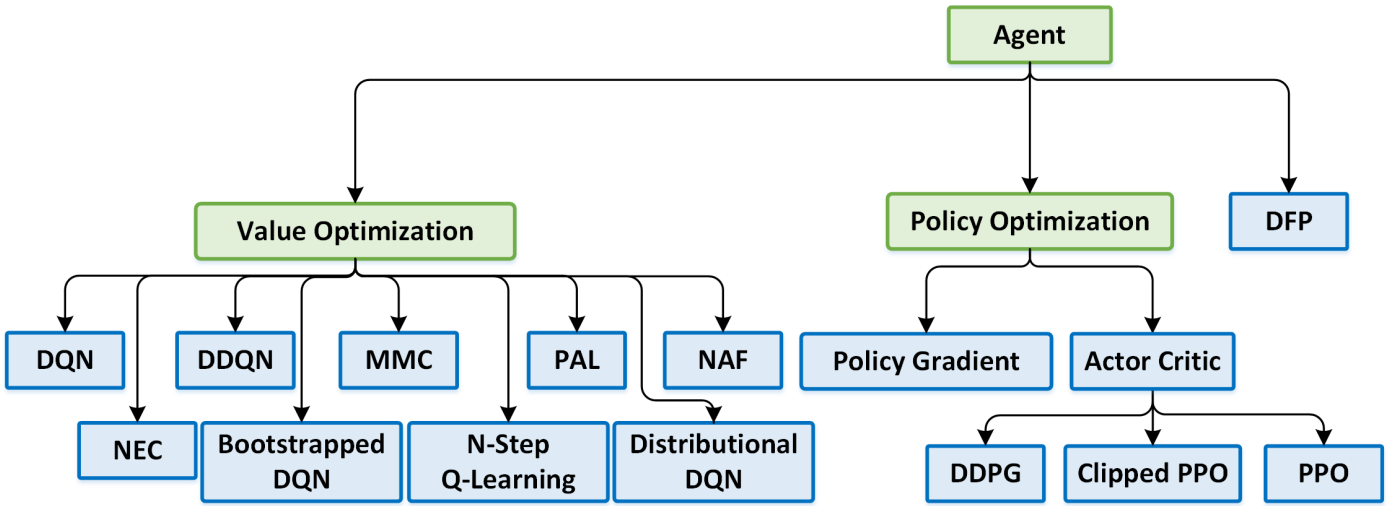


https://ourworldindata.org/technological-progress

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**



https://github.com/vwxyzjn/cleanrl

TorchRL

https://docs.pytorch.org/rl/stable/index.html



https://github.com/DLR-RM/stable-baselines3

Introduction to Reinforcement Learning

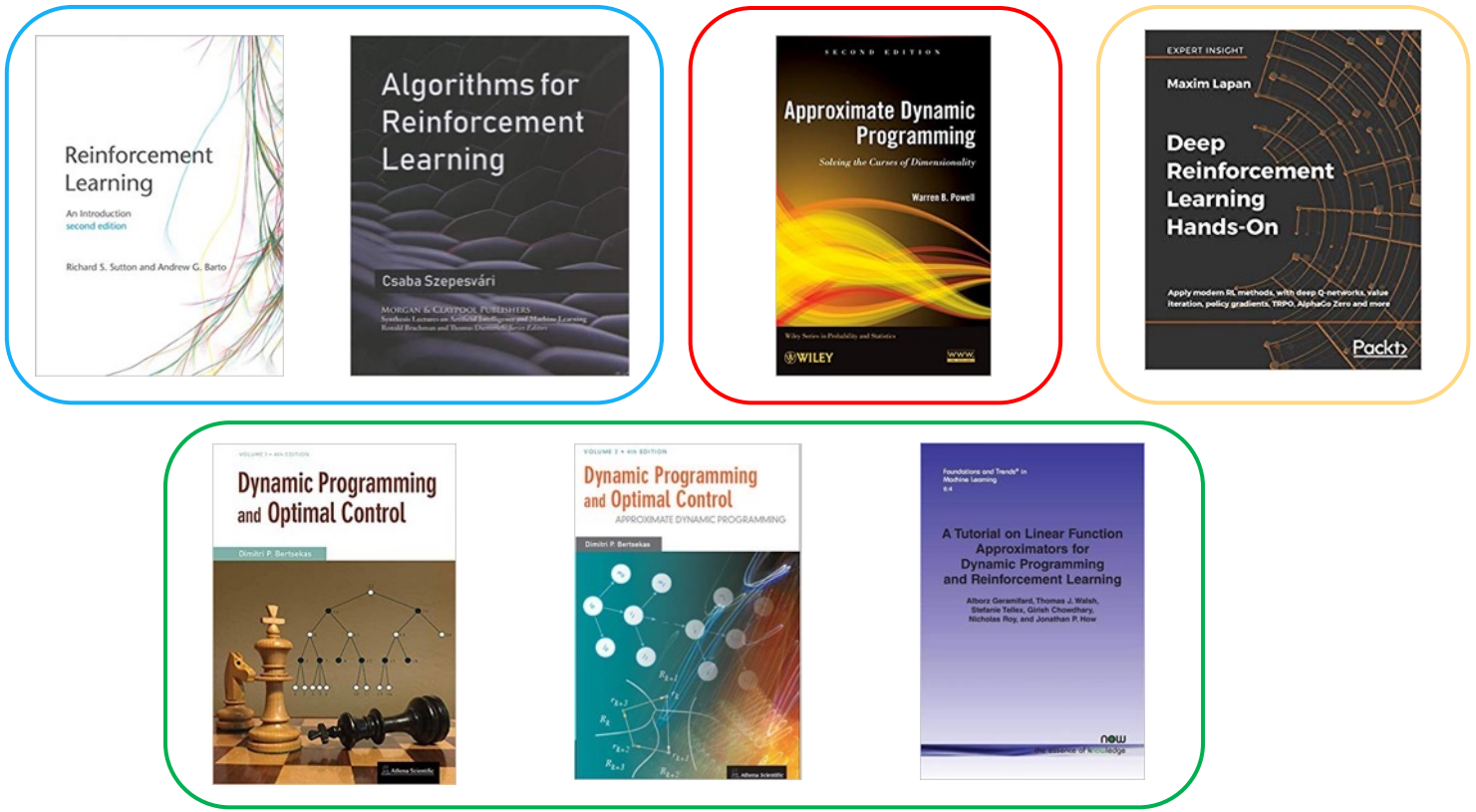
Why RL now?

- 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!**



Introduction to Reinforcement Learning

Literature

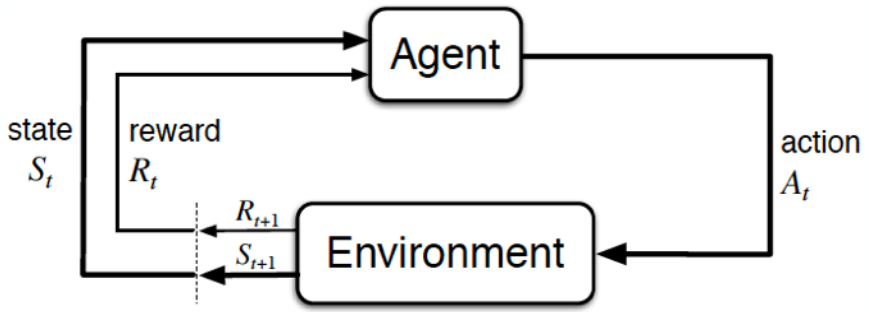


Introduction to Reinforcement Learning

The RL Paradigm (revisited)

- Do you agree with following statement?

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

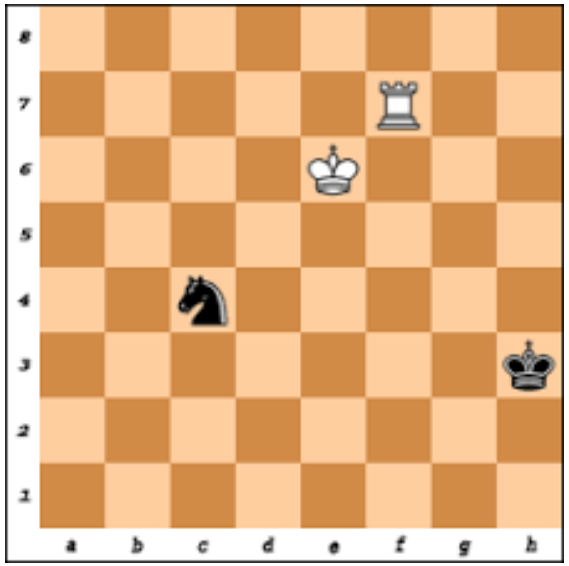


Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

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?

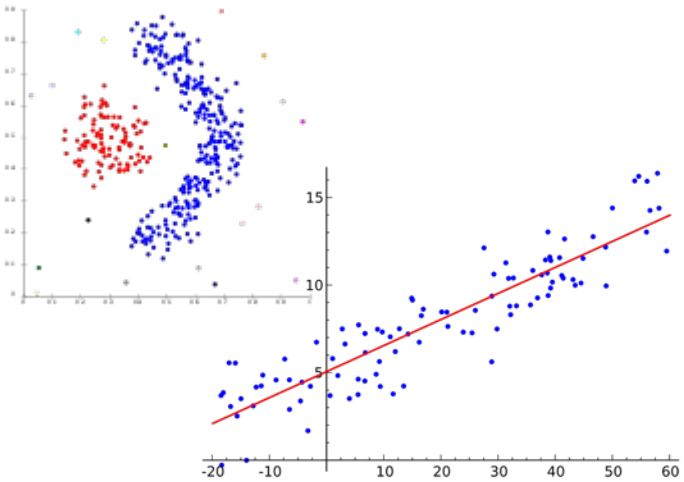


Introduction to Reinforcement Learning

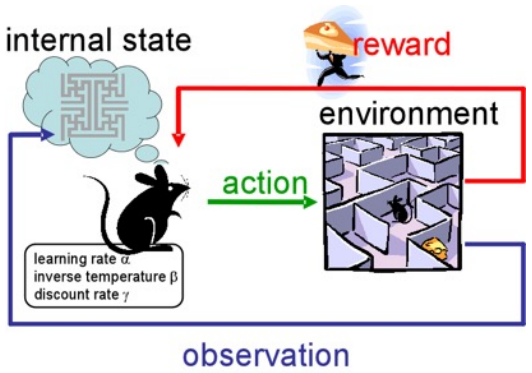
Challenges of understanding/adopting RL

- Counter-Intuitive Visualization!!!

Supervised Learning



Reinforcement Learning

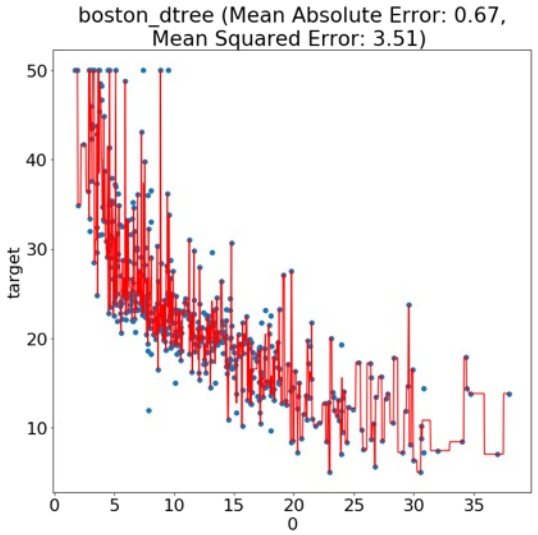


Introduction to Reinforcement Learning

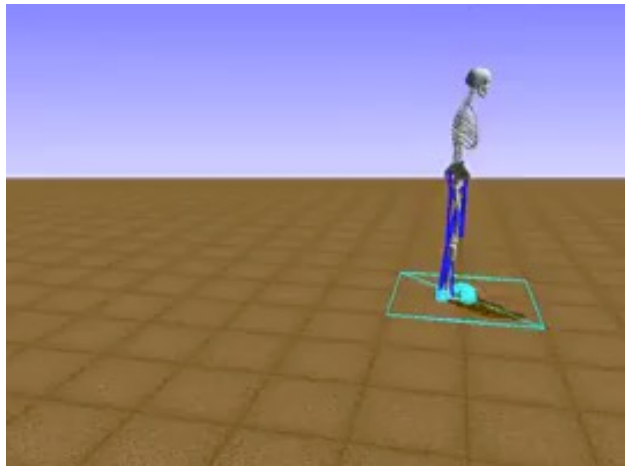
Challenges of understanding/adopting RL

- Example: what went wrong here?

Supervised Learning



Reinforcement Learning

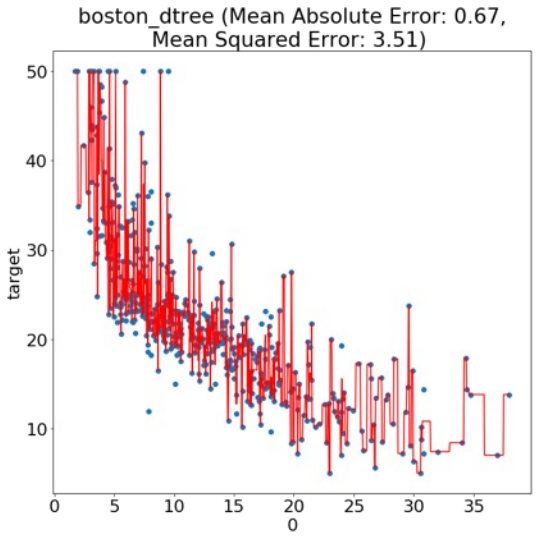


Introduction to Reinforcement Learning

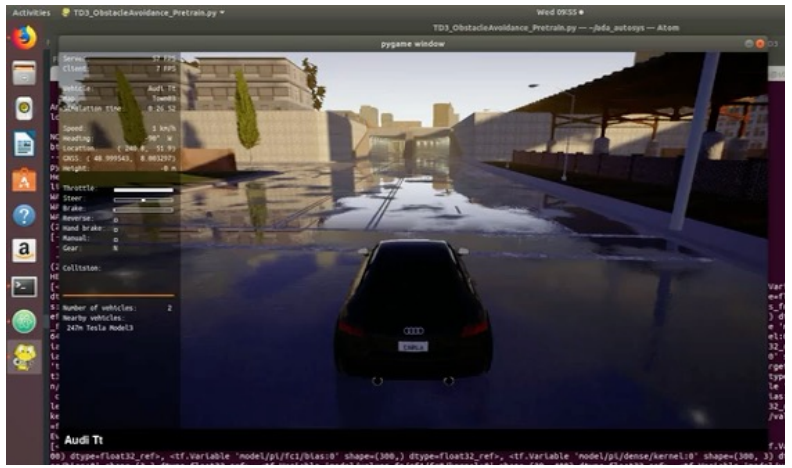
Challenges of understanding/adopting RL

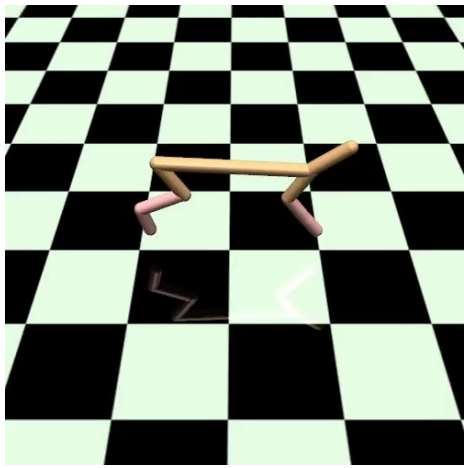
- Example: what went wrong here?

Supervised Learning



Reinforcement Learning





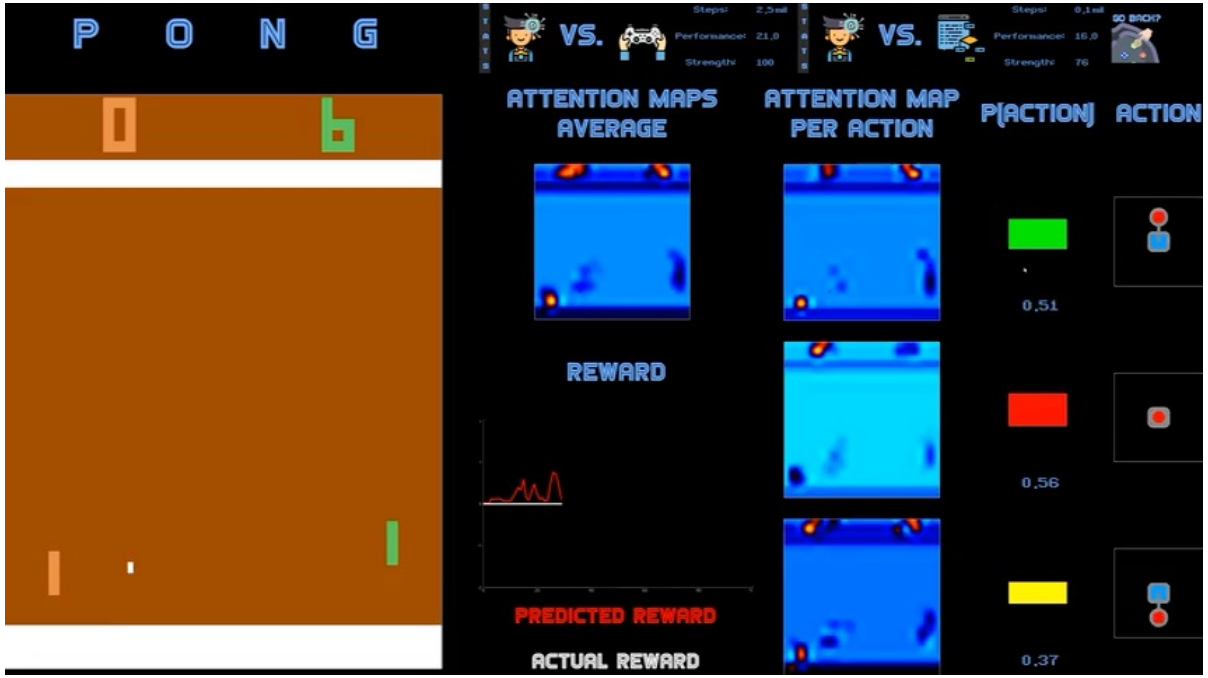
http://ai.berkeley.edu/lecture_slides.html

Introduction to Reinforcement Learning

Challenges of understanding/adopting RL

- Idea: Saliency Maps

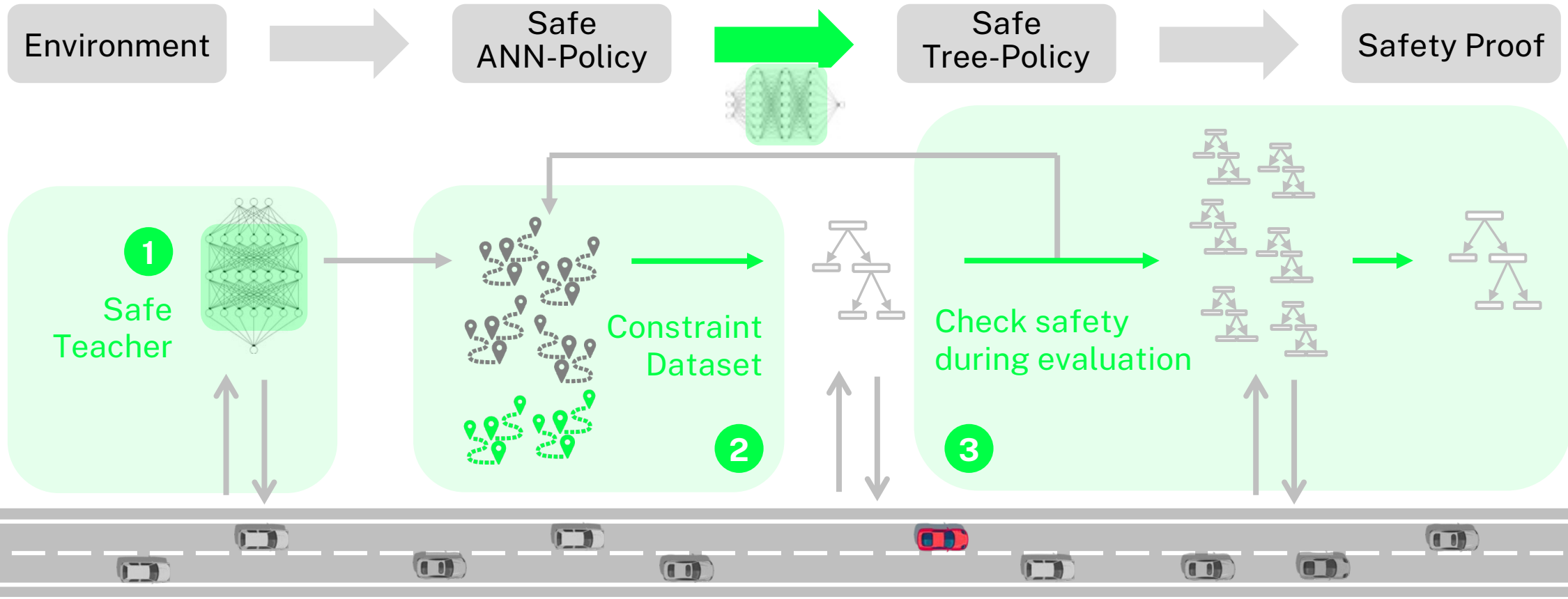




Introduction to Reinforcement Learning

Challenges of understanding/adopting RL

- Idea: explainable decision rules



Schmidt, L.; Kontes, G.; Plinge, A.; Mutschler, C.: Can You Trust Your Autonomous Car? Interpretable and Verifiably Safe Reinforcement Learning. IV2021

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!**

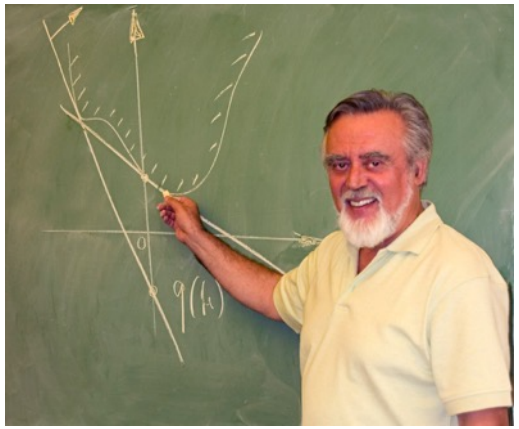
Introduction to Reinforcement Learning

Myth vs. Reality

UTN

- 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!



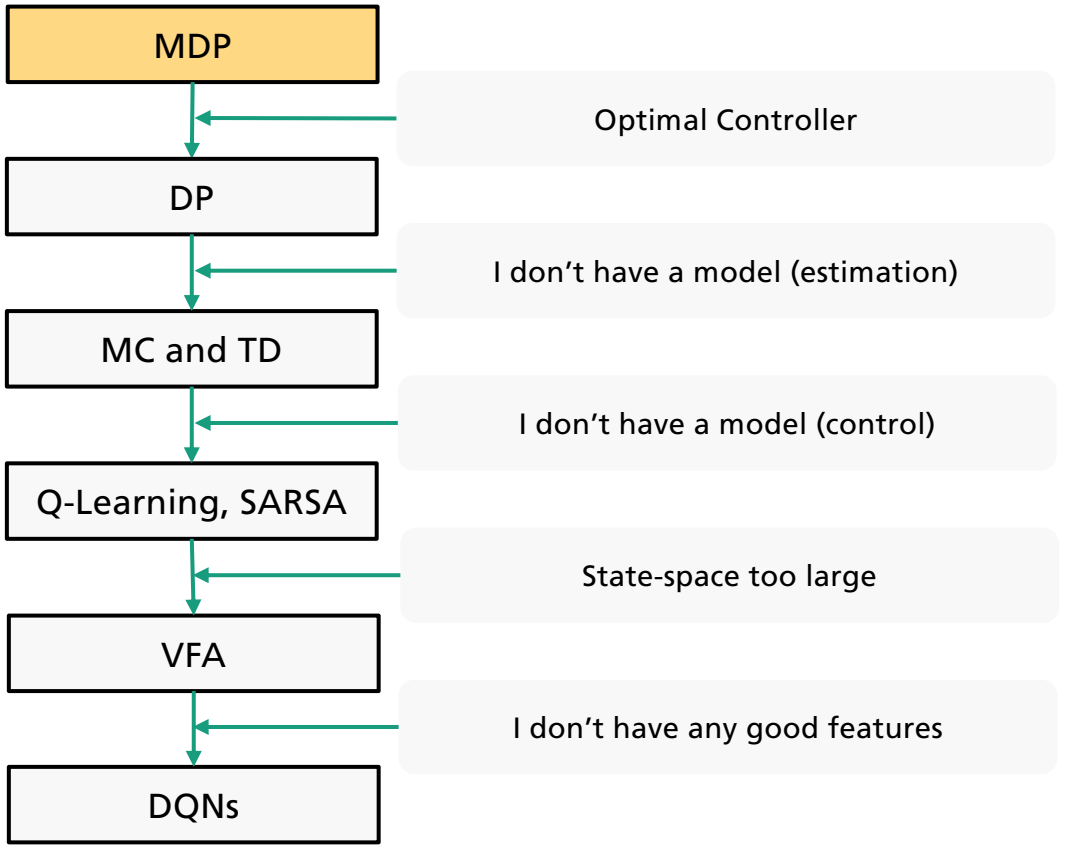
Bertsekas, 2019:

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

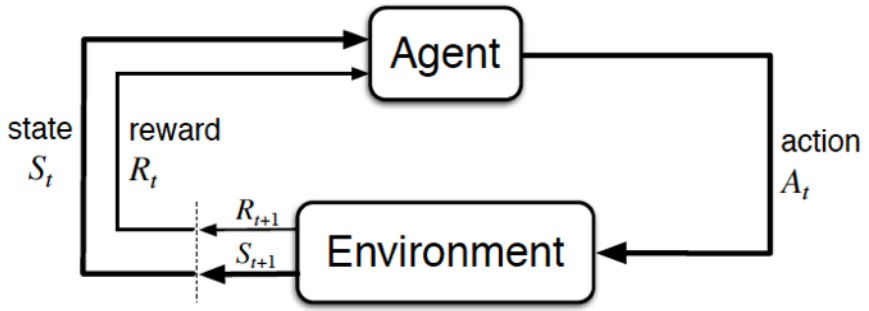
Markov Decision Processes

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.*

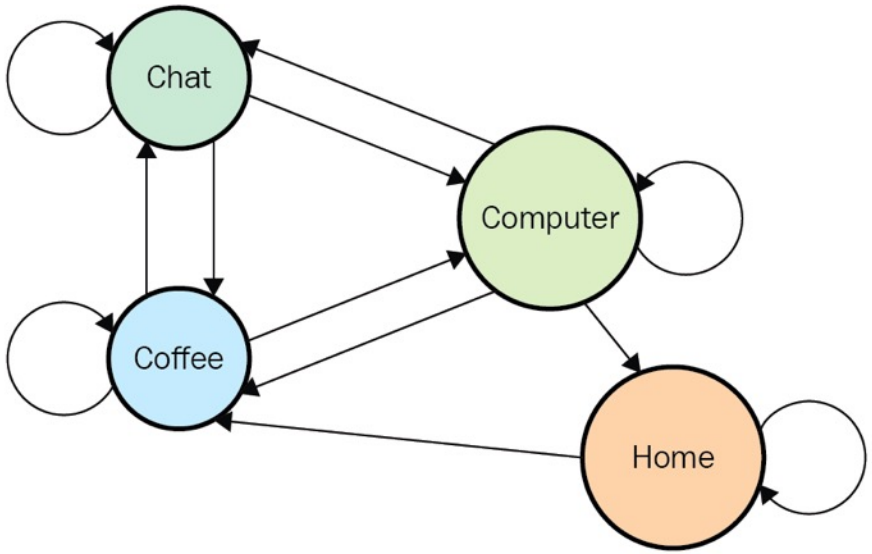
- At each step t, the agent:
  - is at state $\mathrm{S_t}$,
  - performs action $\mathrm{A_t}$,
  - receives reward $\mathrm{R_t}$.

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

Markov Decision Processes

UTN

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

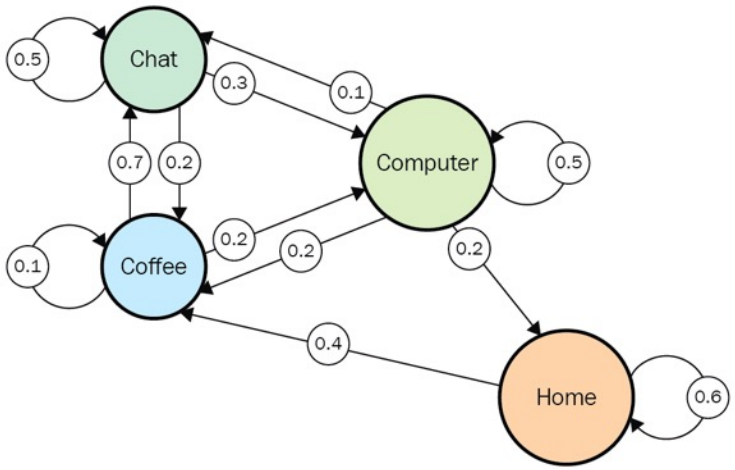


Lapan, M. (2018). Deep Reinforcement Learning Hands-On. Packt Publishing Ltd.

Markov Decision Processes

UTN

- 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%

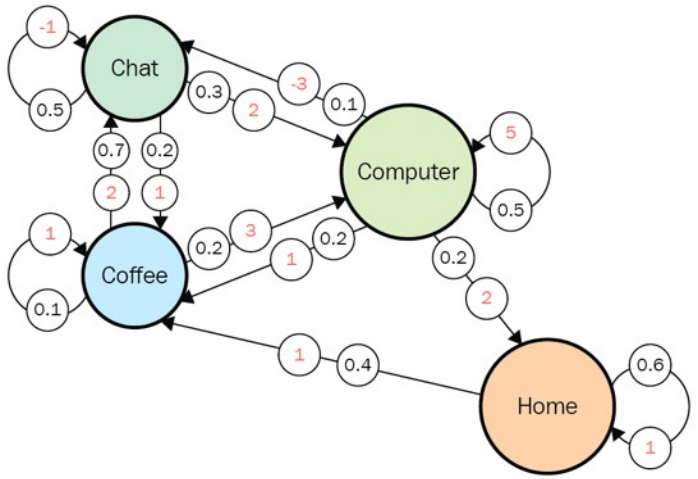
Lapan, M. (2018). Deep Reinforcement Learning Hands-On. Packt Publishing Ltd.

Markov Decision Processes

UTN

- 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

UTN

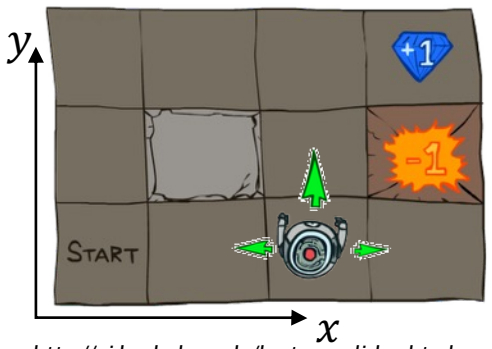
- Markov Decision Process (MDP)
  - Description of an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$:



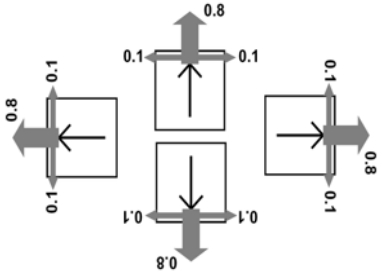
http://ai.berkeley.edu/lecture_slides.html

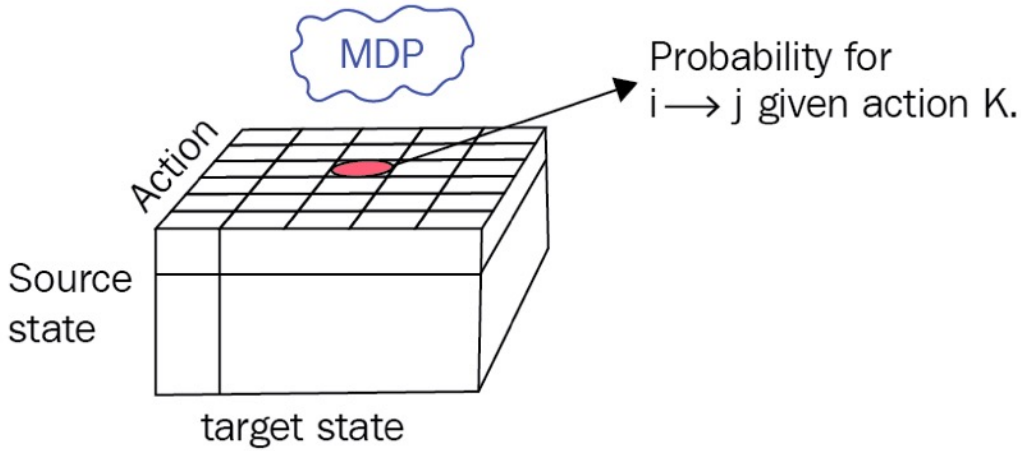
Markov Decision Processes

- 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

UTN

- 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:

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

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

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 (environment's private representation)**
    - Includes all the data that the environment uses to select next observation and reward
    - Private to the environment, not visible, maybe irrelevant information
  - **Agent state S_t^a (agent's private representation; actually used)**
    - Private to the agent, history of observations, rewards, and actions
    - The agent constructs a state representation using a function of history $S_t^a = f(H_t)$ to decide on the next action
  - **Information state (useful information from the history)**
    - Basically, S_t^a with special constraints in $f(H_t)$

Markov Decision Processes

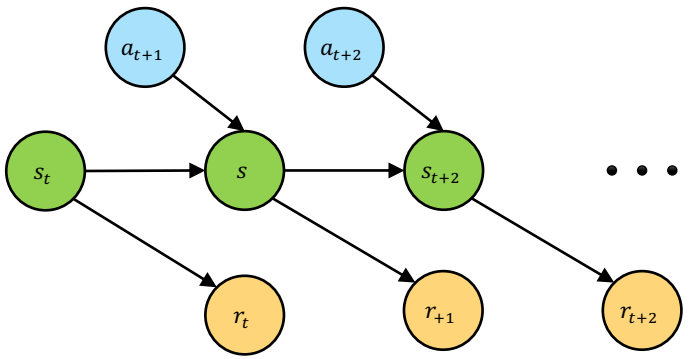
about the state space $\mathcal{S}$

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

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

  - Past states $S_1, \dots, S_{t-1}$ 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"**

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

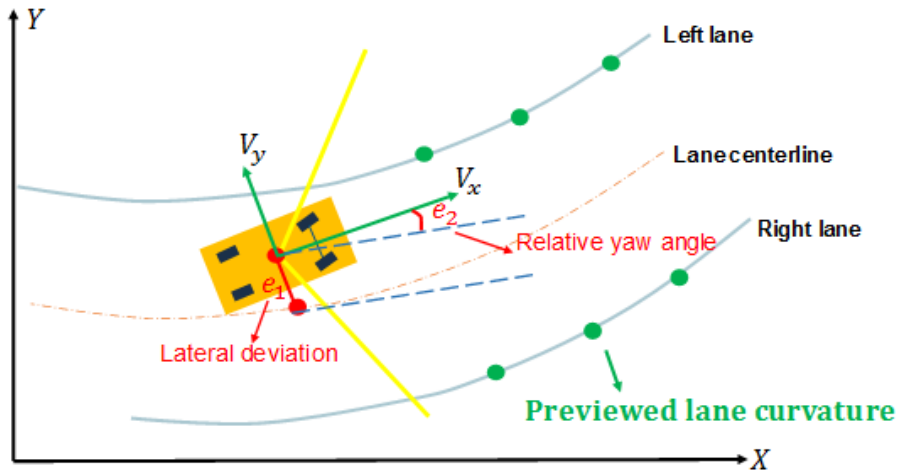


- State is the information used to determine what happens next
  - Direct (fully observable): $O_t = S_t^e$
  - Indirect (partially observable): $O_t = f(S_t^e)$

Markov Decision Processes

about the state space $\mathcal{S}$

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



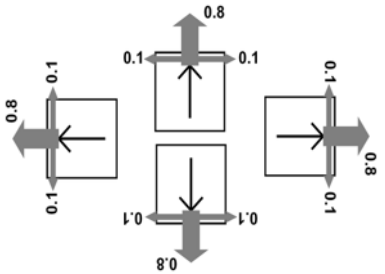
Sensor Measurements:
- Speed, Angle

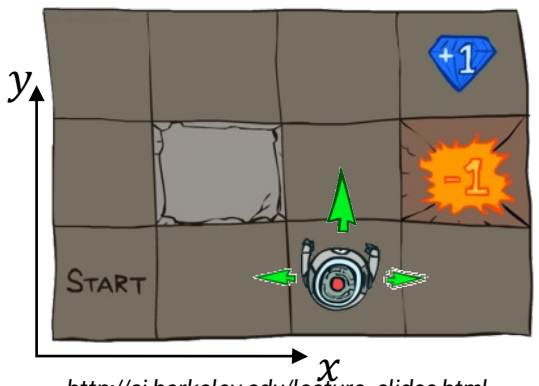
Requirements:
- Lateral acceleration
- Angular velocity

Markov Decision Processes

about the action space $\mathcal{A}$

- MDP example: Gridworld, episodic task





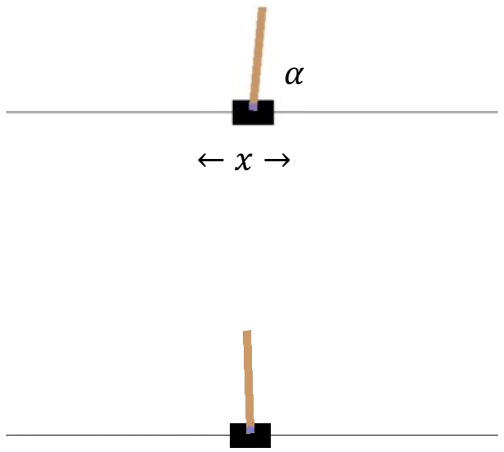
http://ai.berkeley.edu/lecture_slides.html

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

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

about the action space $\mathcal{A}$

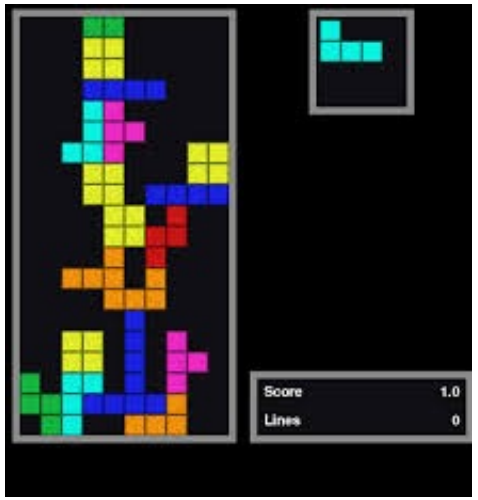
- MDP example: Cartpole, episodic or continuing task



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

Markov Decision Processes

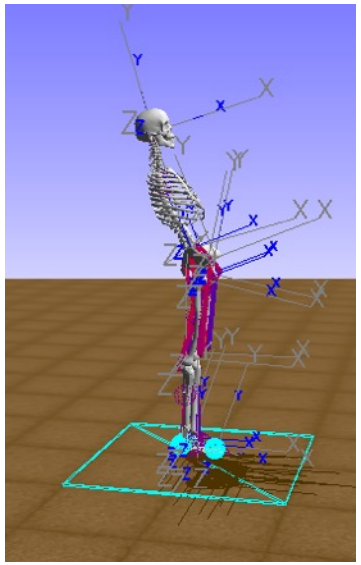
MDP example: Tetris, *episodic task*

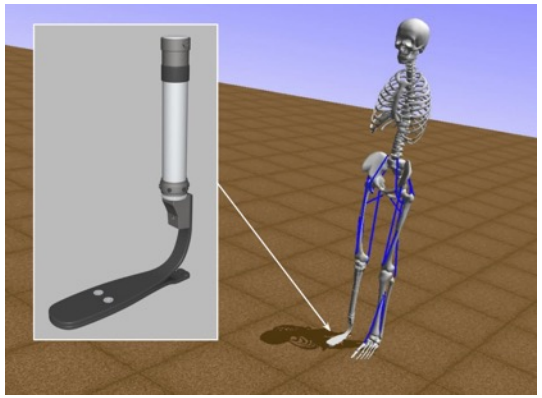




Markov Decision Processes

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





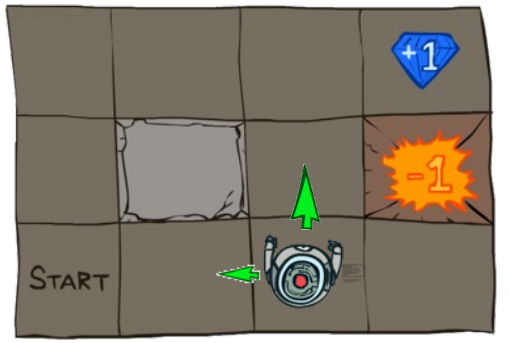
# of muscles	19
# degrees of freedom	14
reward	negative distance from requested velocity

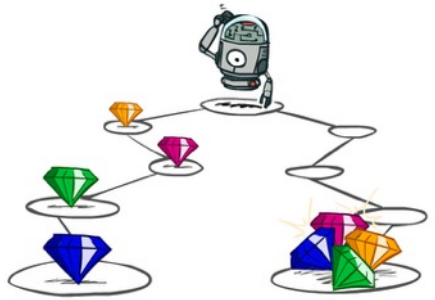
Markov Decision Processes

UTN

- 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

Markov Decision Processes

UTN

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

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

  - **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

UTN

- 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 $\mathbb{E}(\mathbf{G})$.**

- 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!**

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



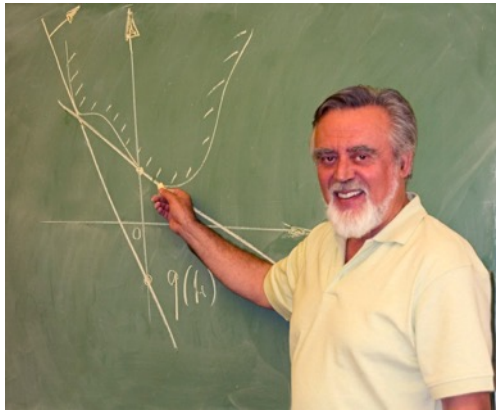
Lev Pontryagin

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

Markov Decision Processes

UTN

- 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.