Deep Reinforcement Learning, Decision Making, and Control

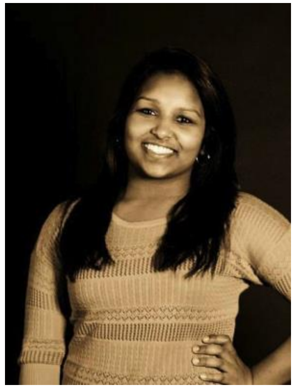
CS 285

Course logistics

Class Information & Resources

Sergey Levine
Instructor

Frederik Ebert
Head GSI



Anusha Nagabandi
GSI

Avi Singh
GSI

Kelvin Xu
GSI

- Course website: http://rail.eecs.berkeley.edu/deeprlcourse
- Piazza: UC Berkeley, CS285
- Gradescope: UC Berkeley, CS285
- Subreddit (for non-enrolled students): www.reddit.com/r/berkeleydeeprlcourse/
- Office hours: check course website (mine are after class on Wed)

Prerequisites & Enrollment

- All enrolled students must have taken CS189, CS289, CS281A, or an equivalent course at your home institution
    - Please contact Sergey Levine if you haven't
- If you are not eligible to enroll directly into the class, **fill out the enrollment application form:** http://rail.eecs.berkeley.edu/deeprlcourse/
    - We will enroll subject to availability based on responses to this form
    - We will not use the official CalCentral wait list!
    - **Fill out an application before Friday!**
- Lectures are recorded and live streamed (link on course website)

What you should know

- Assignments will require training neural networks with standard automatic differentiation packages (TensorFlow by default)
- Review Section
  - Kelvin Xu will review TensorFlow on Mon in week 3 (Sep 9)
  - You should be able to understand the overview here: https://www.tensorflow.org/guide/low_level_intro
    - If not, make sure to attend Kelvin's lecture and ask questions!

What we'll cover

- Material will be similar to previous year: http://rail.eecs.berkeley.edu/deeprlcourse-fa18/

1. From supervised learning to decision making
2. Model-free algorithms: Q-learning, policy gradients, actor-critic
3. Advanced model learning and prediction
4. Transfer and multi-task learning, meta-learning
5. Exploration
6. Open problems, research talks, invited lectures

- Lecture 1: Introduction and Course Overview
- Lecture 2: Supervised Learning and Imitation
- Lecture 3: TensorFlow and Neural Nets Review Session (notebook)
- Lecture 4: Reinforcement Learning Introduction
- Lecture 5: Policy Gradients Introduction
- Lecture 6: Actor-Critic Introduction
- Lecture 7: Value Functions and Q-Learning
- Lecture 8: Advanced Q-Learning Algorithms
- Lecture 9: Advanced Policy Gradients
- Lecture 10: Optimal Control and Planning
- Lecture 11: Model-Based Reinforcement Learning
- Lecture 12: Advanced Model Learning and Images
- Lecture 13: Learning Policies by Imitating Other Policies
- Lecture 14: Probability and Variational Inference Primer
- Lecture 15: Connection between Inference and Control
- Lecture 16: Inverse Reinforcement Learning
- Lecture 17: Exploration: Part 1
- Lecture 18: Exploration: Part 2
- Lecture 19: Transfer Learning and Multi-Task Learning
- Lecture 20: Meta-Learning
- Lecture 21: Parallelism and RL System Design
- Lecture 22: Advanced Imitation Learning and Open Problems

Assignments

1. Homework 1: Imitation learning (control via supervised learning)
2. Homework 2: Policy gradients (“REINFORCE”)
3. Homework 3: Q learning and actor-critic algorithms
4. Homework 4: Model-based reinforcement learning
5. Homework 5: Advanced model-free RL algorithms
6. Final project: Research-level project of your choice (form a group of up to 2-3 students, you’re welcome to start early!)

Grading: 50% homework (10% each), 50% project

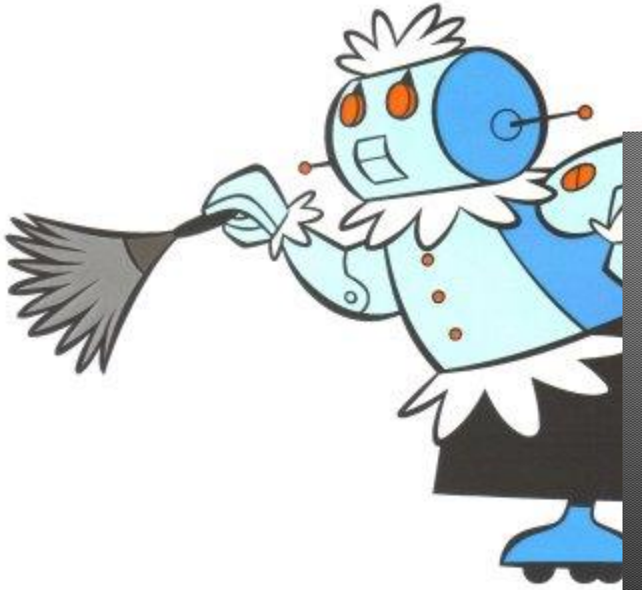
5 late days total

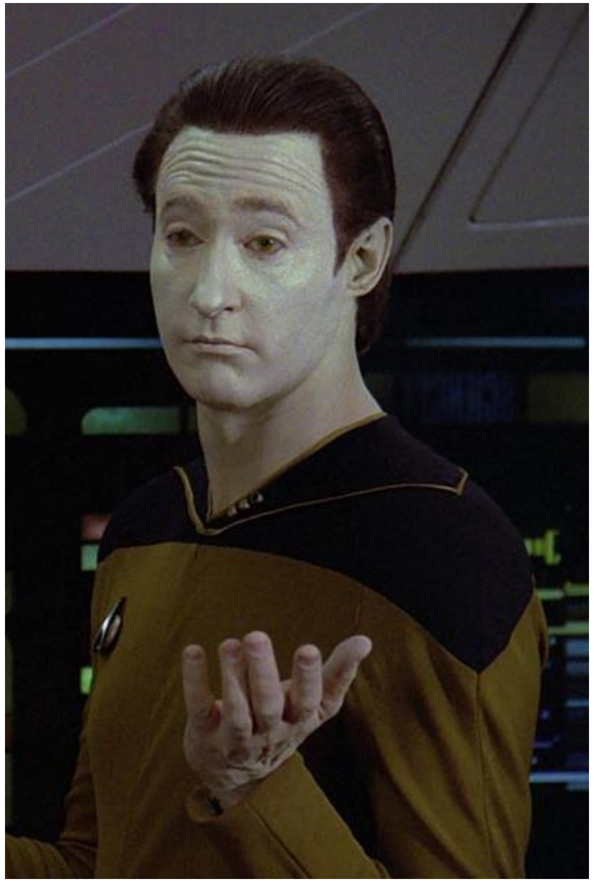
Your “Homework” Today

1. Sign up for Piazza (UC Berkeley CS285)
2. Start forming your final project groups, unless you want to work alone, which is fine
3. Review this: https://www.tensorflow.org/guide/low_level_intro

What is reinforcement learning, and why should we care?

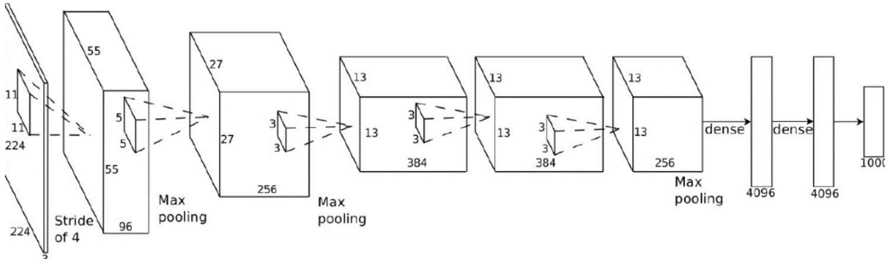
How do we build intelligent machines?

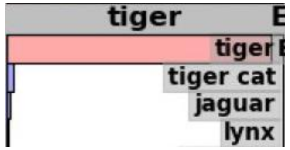


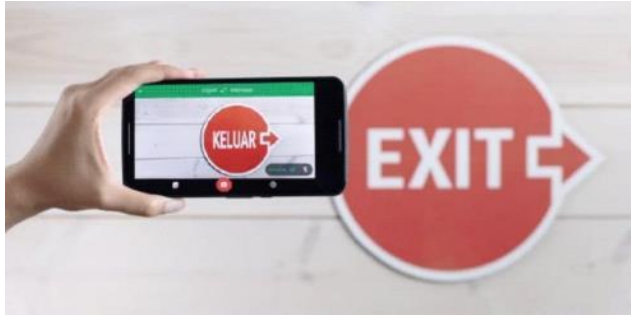


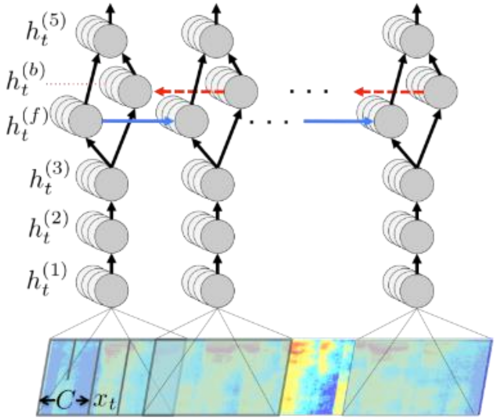
Intelligent machines must be able to adapt

Deep learning helps us handle *unstructured environments*



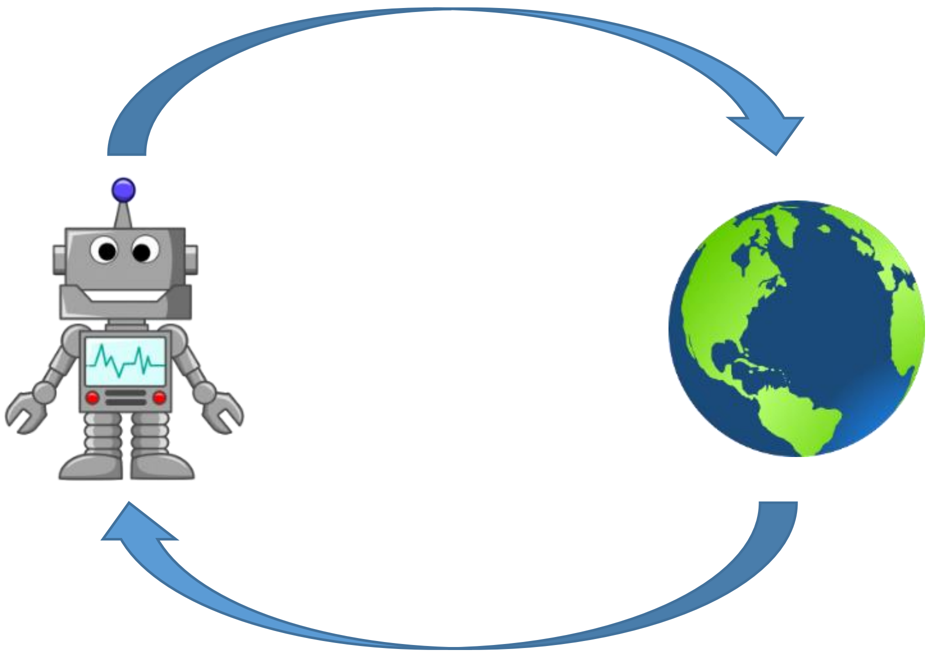






Reinforcement learning provides a formalism for *behavior*

decisions (actions)



<u>consequences</u>
observations
rewards

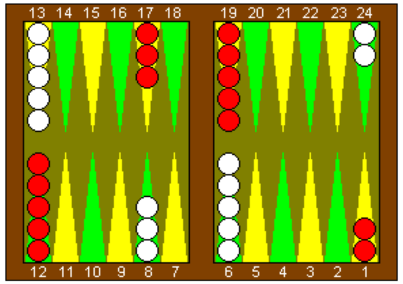
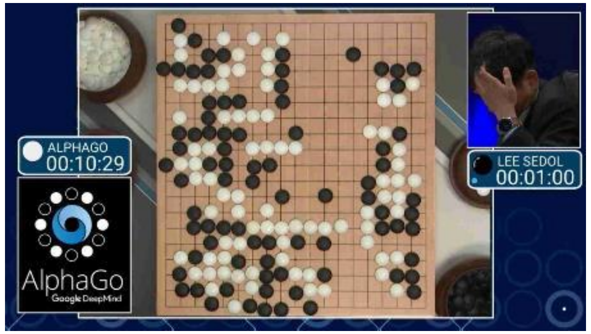
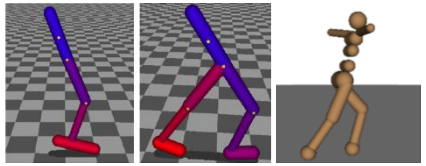
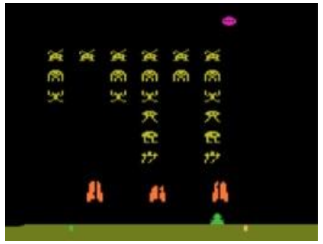


Figure 2. An illustration of the normal opening position in backgammon. TD-Gammon has sparked a near-universal conversion in the way experts play certain opening rolls. For example, with an opening roll of 4-1, most players have now switched from the traditional move of 13-9, 6-5, to TD-Gammon's preference, 13-9, 24-23. TD-Gammon's analysis is given in Table 2.

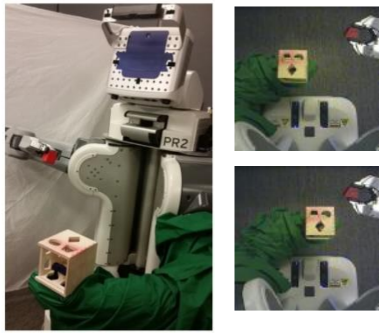




Schulman et al. '14 & '15



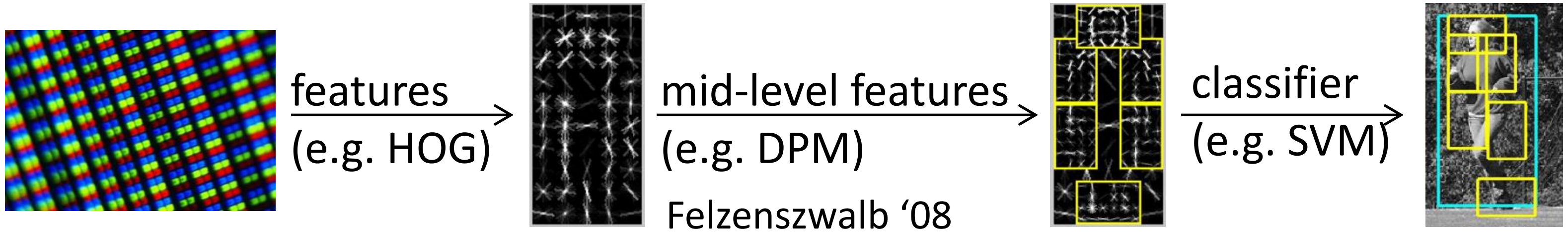
Mnih et al. '13



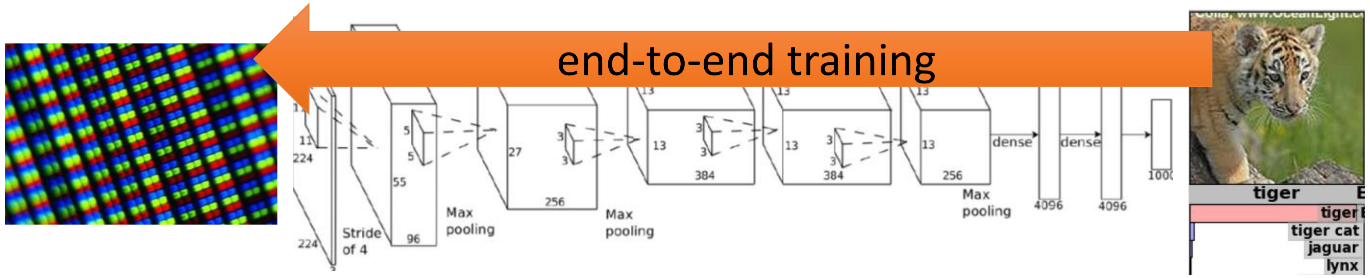
Levine*, Finn*, et al. '16

What is deep RL, and why should we care?

standard computer vision

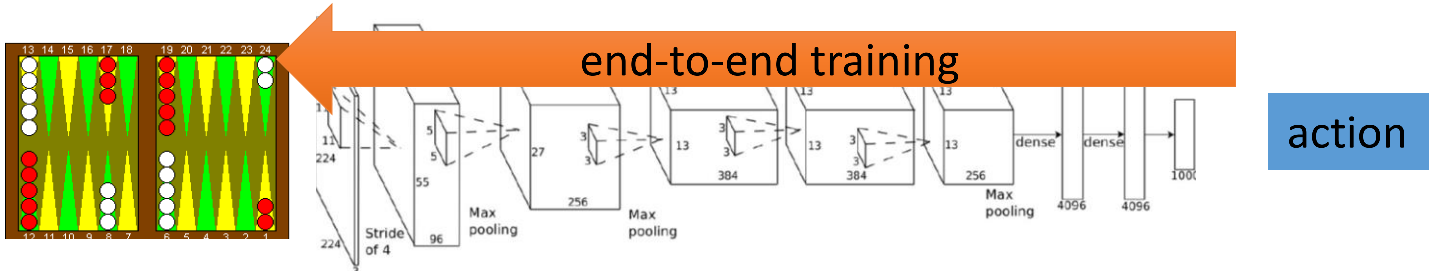


deep learning

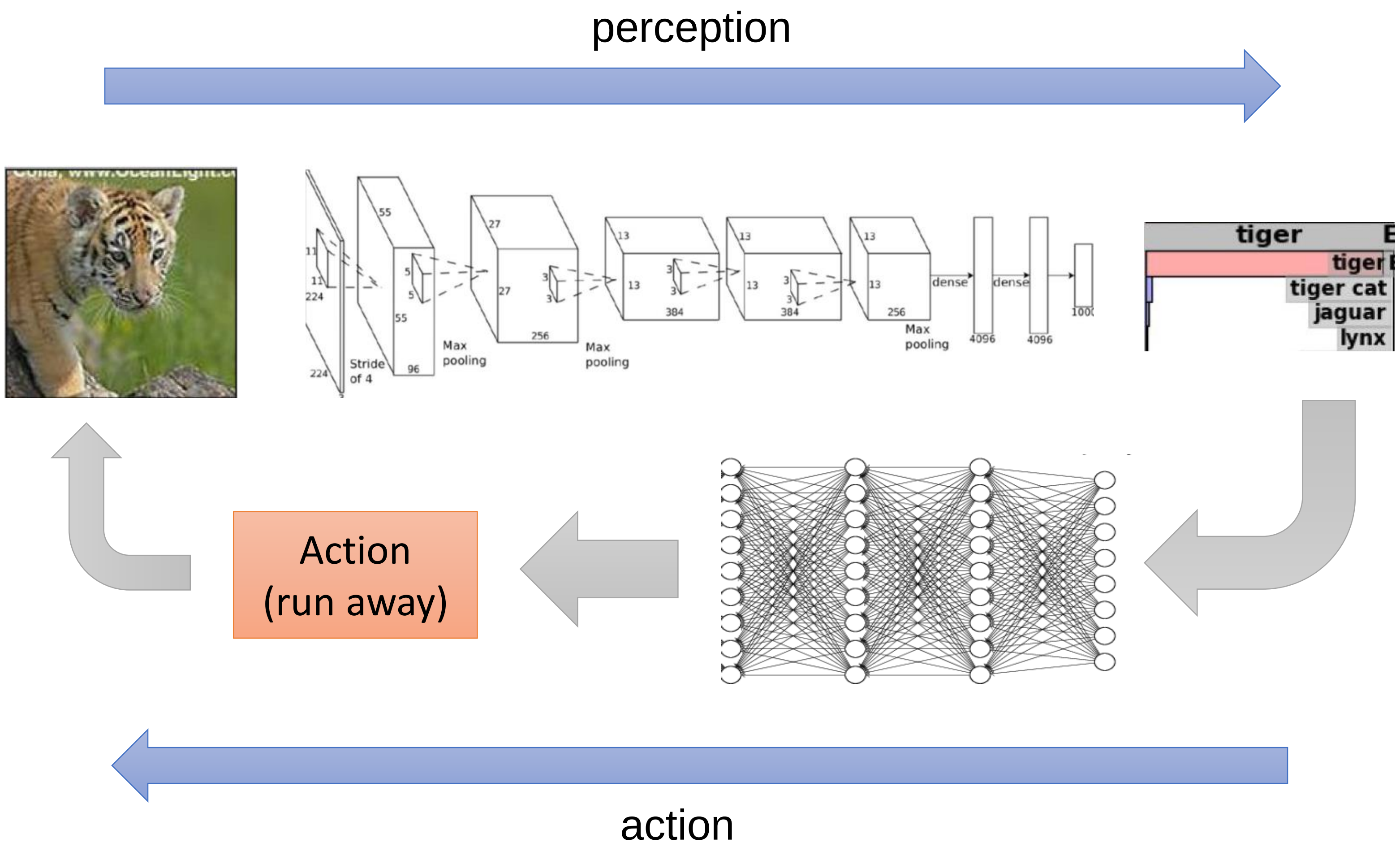


standard reinforcement learning

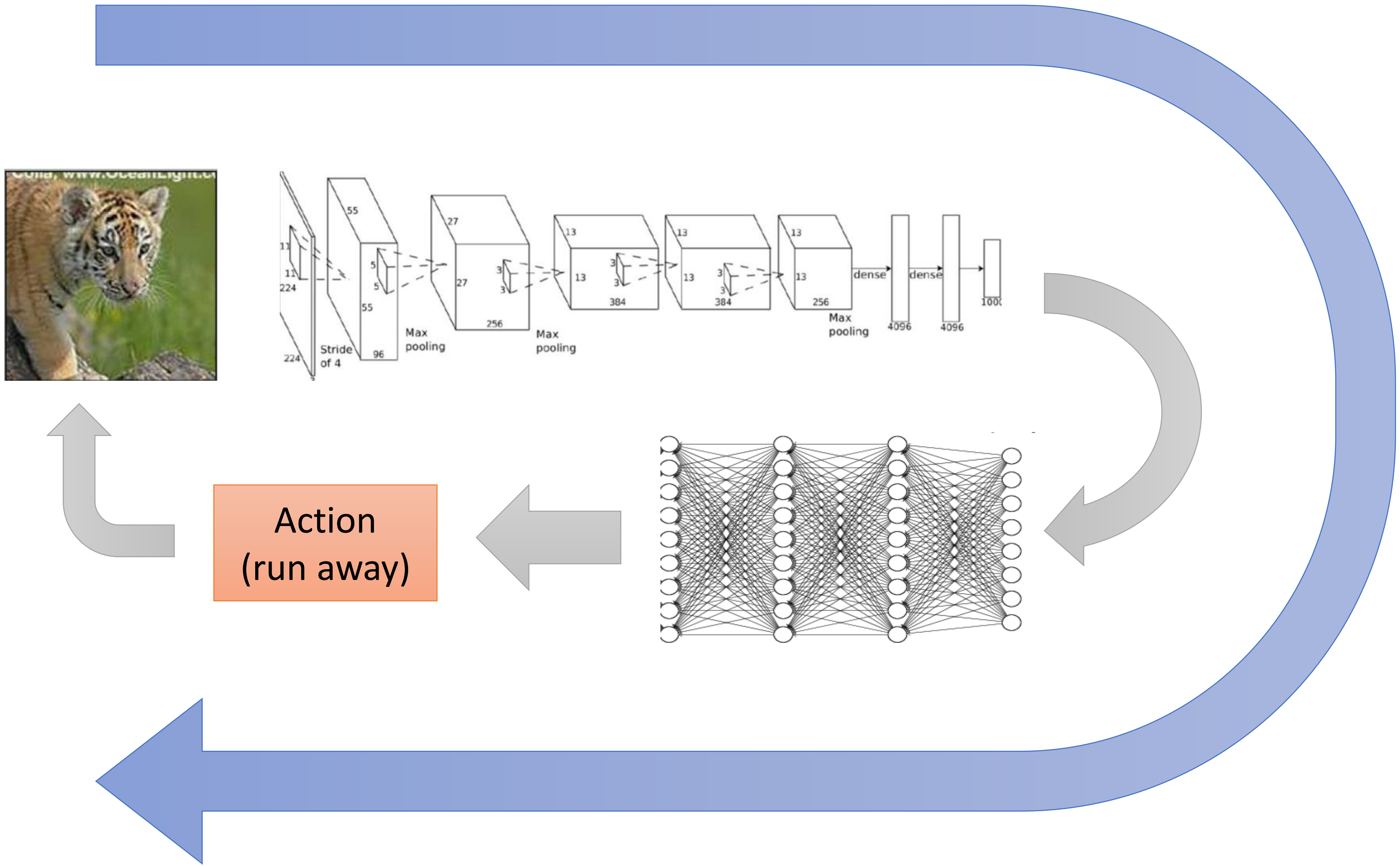
deep reinforcement learning



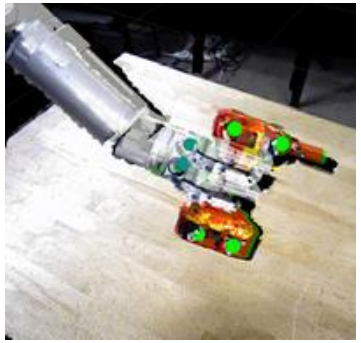
What does end-to-end learning mean for sequential decision making?

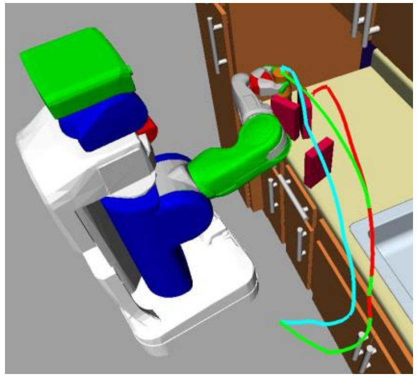

perception
Stride of 4
Max pooling
Max pooling
Max pooling
dense
dense
tiger
tiger
tiger cat
jaguar
lynx
Action
(run away)
action

sensorimotor loop



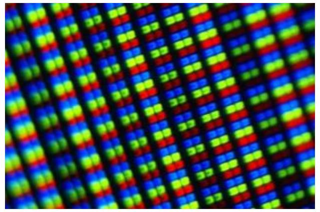
Example: robotics

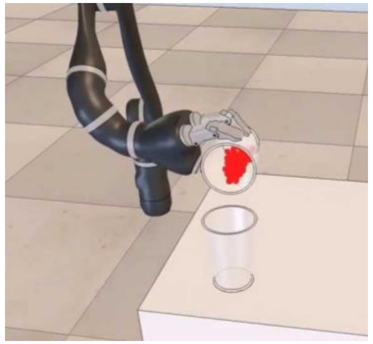


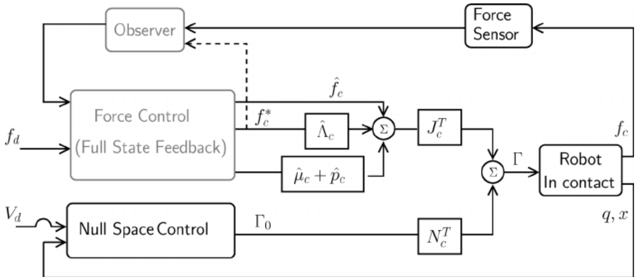


robotic control pipeline

observations → state estimation (e.g. vision) → modeling & prediction → planning → low-level control → controls

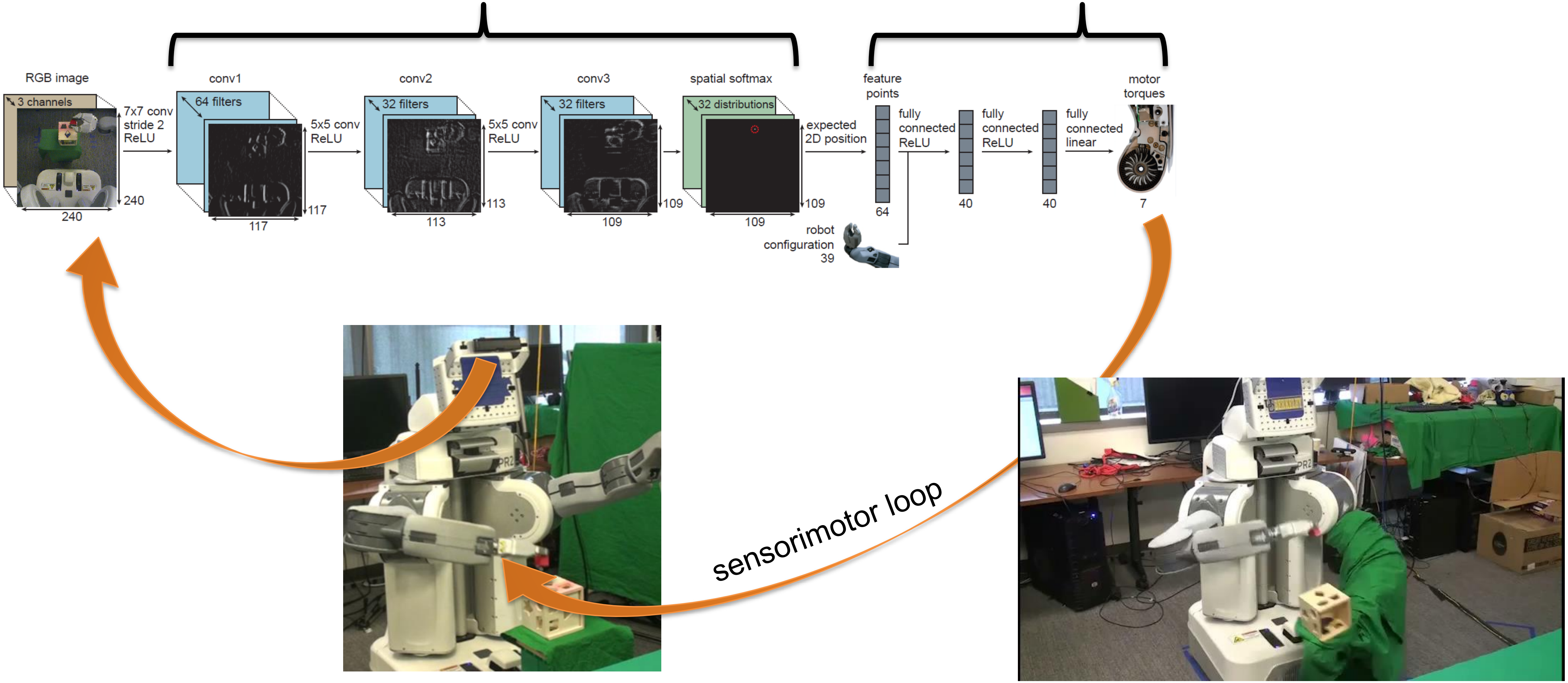


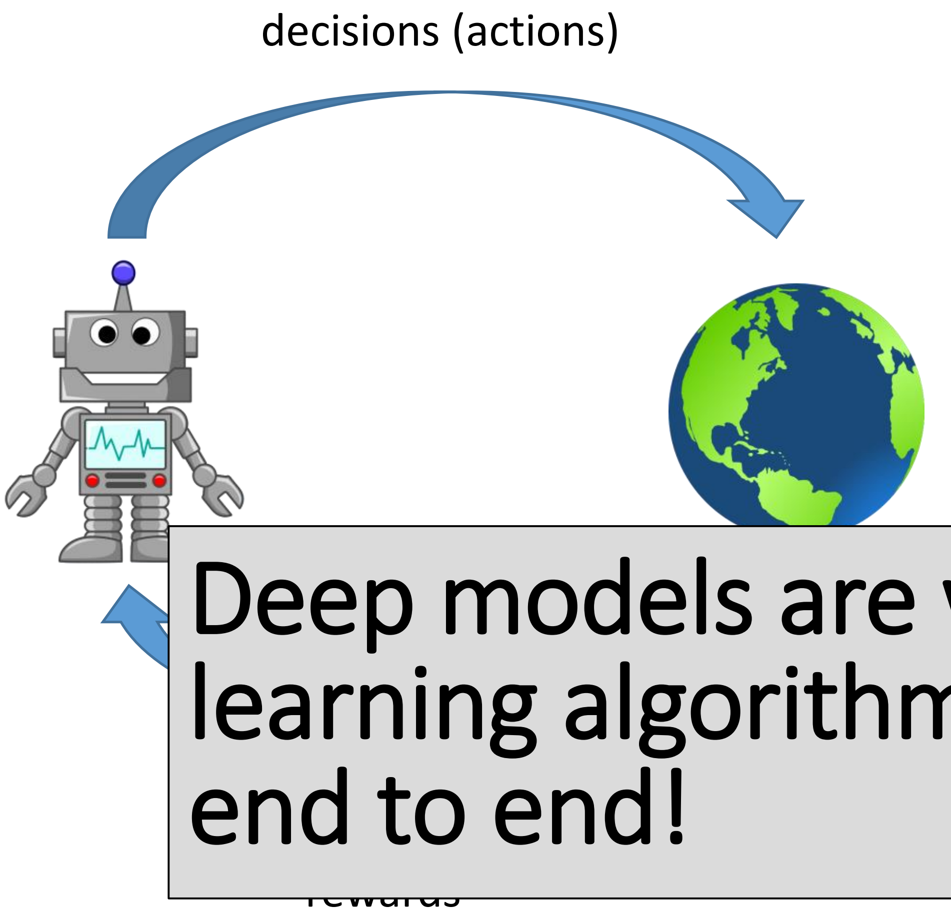




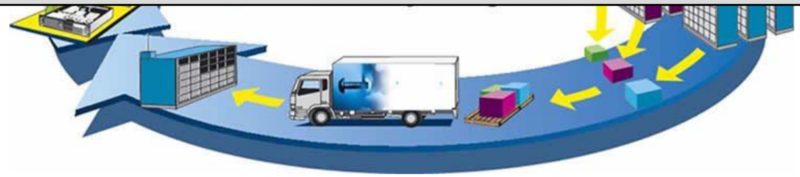
tiny, highly specialized
“visual cortex”

tiny, highly specialized
“motor cortex”





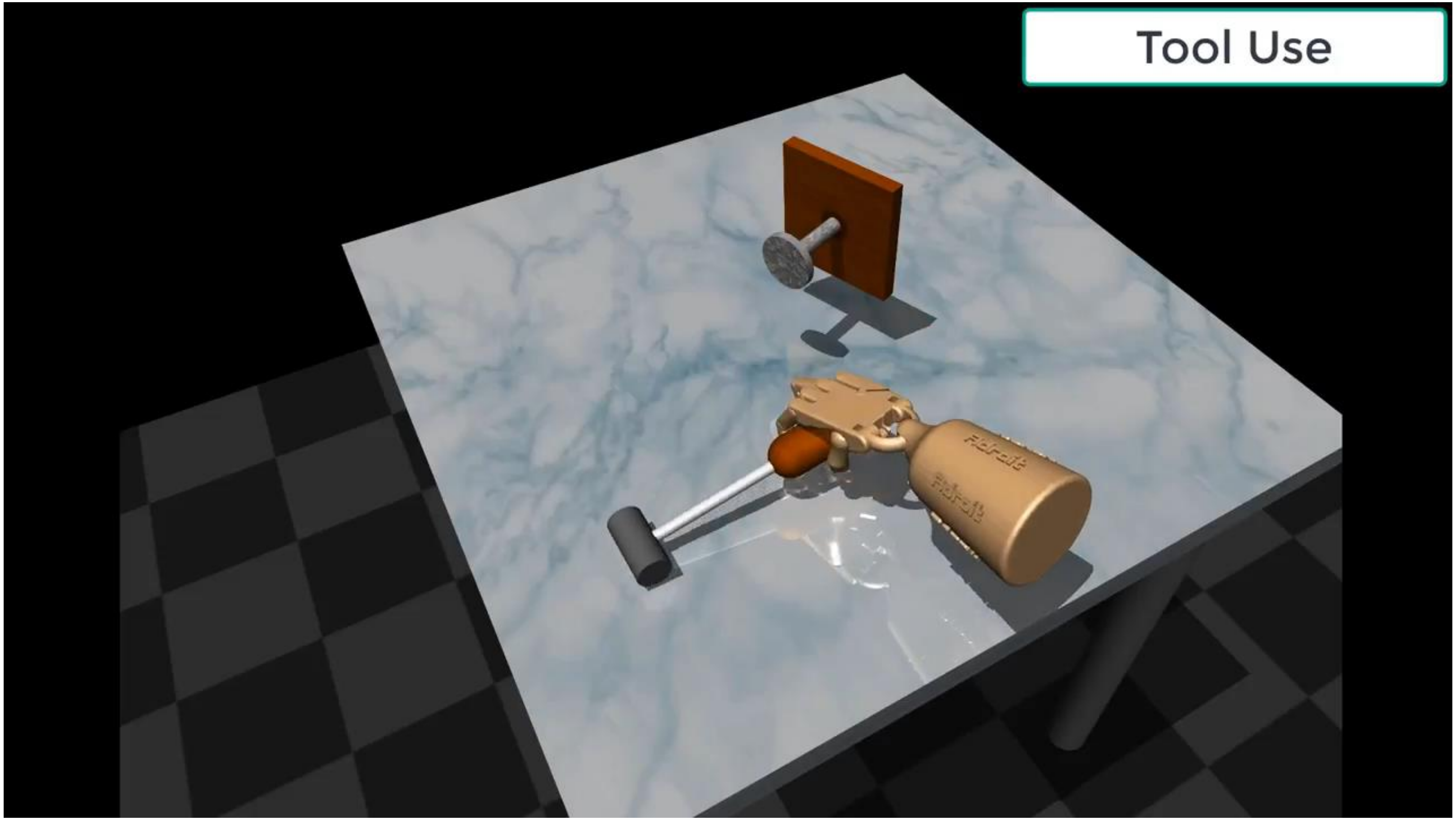
Deep models are what allow reinforcement learning algorithms to solve complex problems end to end!



Actions: what to purchase
Observations: inventory levels
Rewards: profit

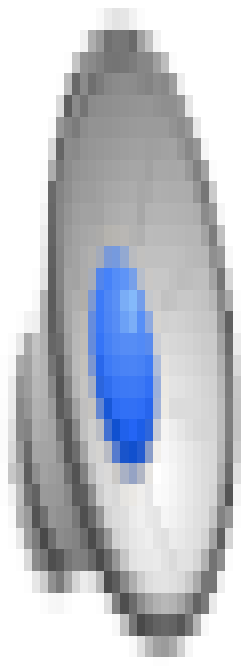
The reinforcement learning problem **is** the AI problem!

Complex physical tasks…



Rajeswaran, et al. 2018

Unexpected solutions...

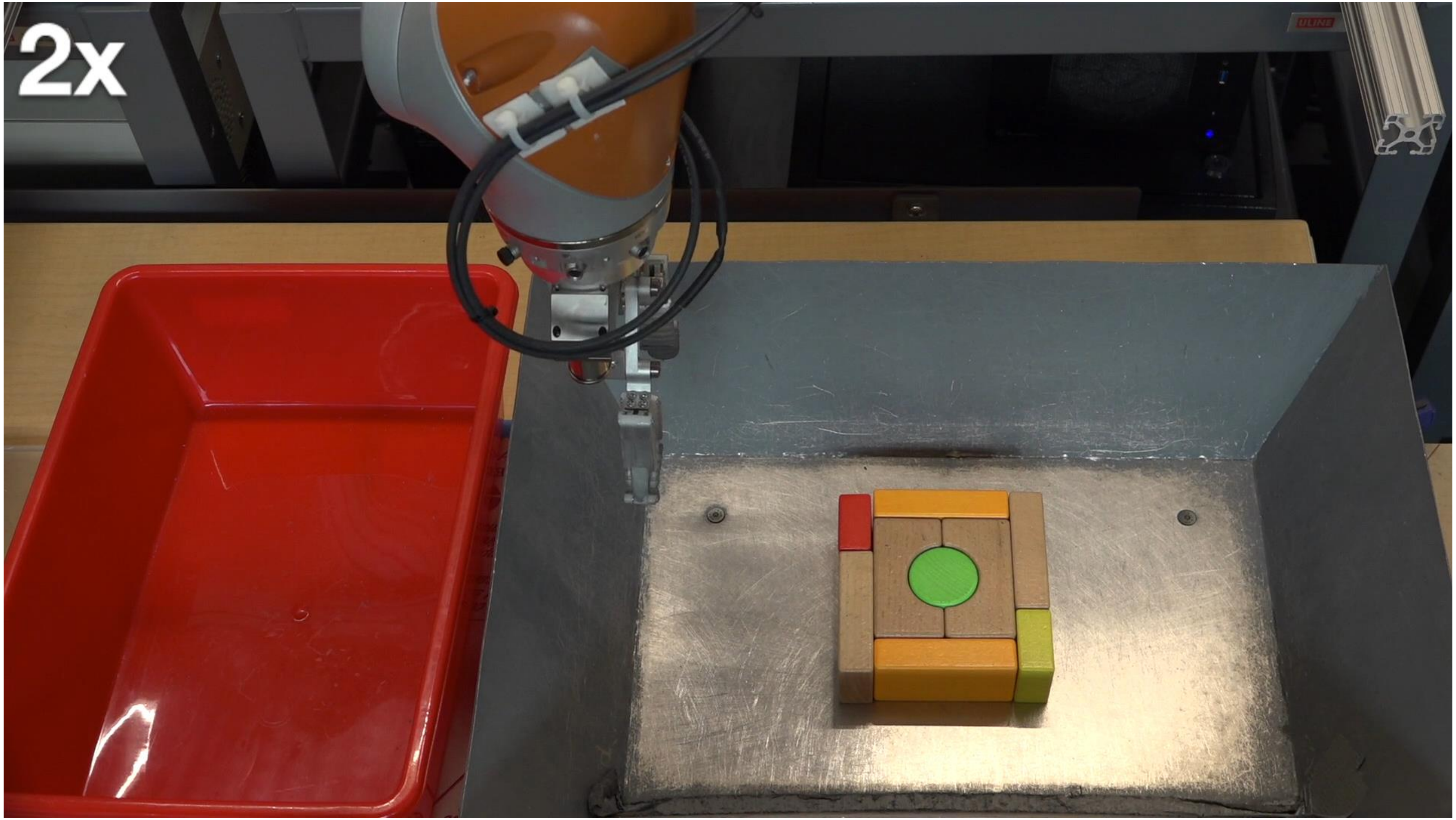


Mnih, et al. 2015

In the real world…

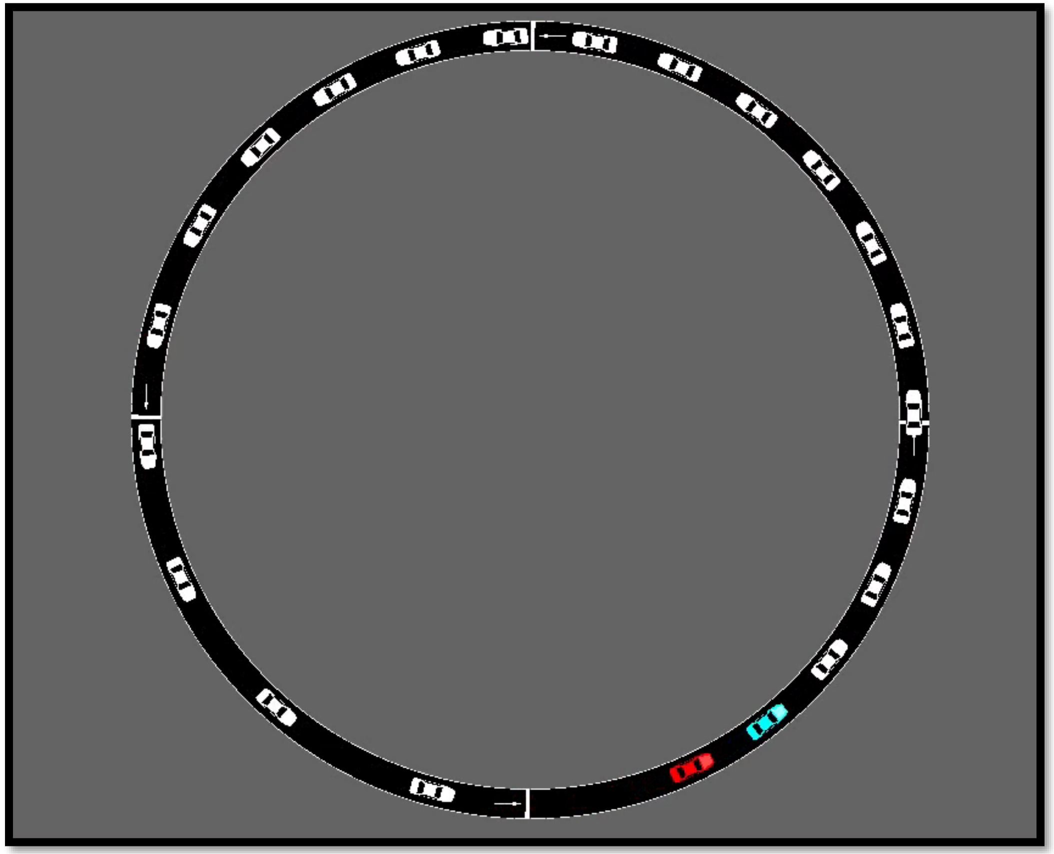
Kalashnikov et al. ‘18

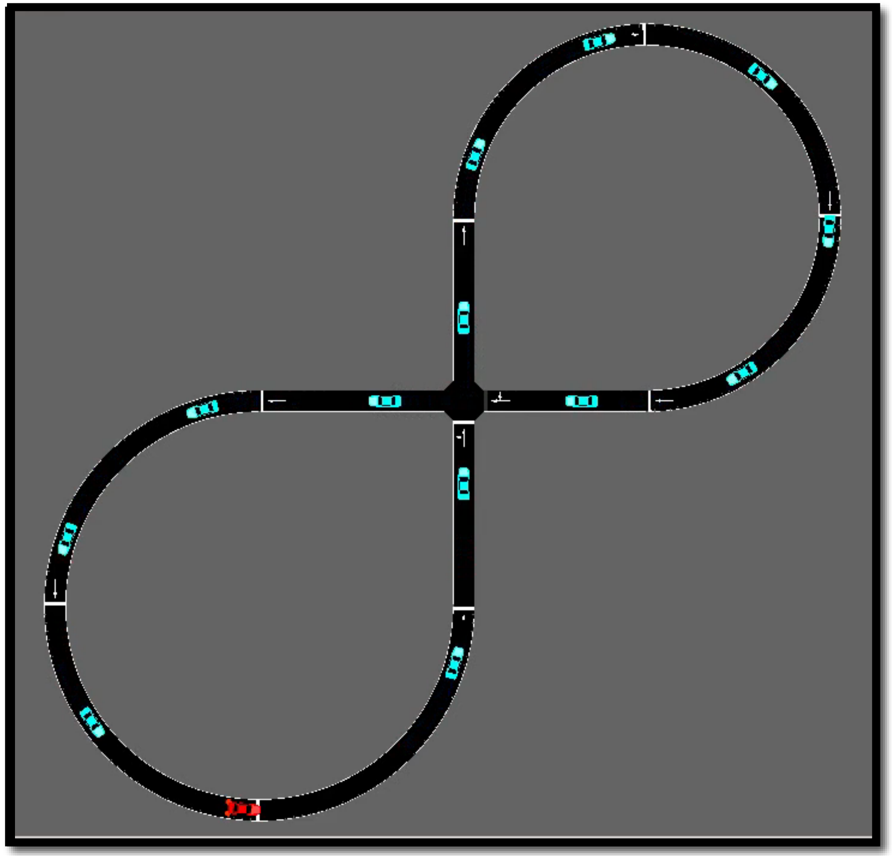
In the real world…



Kalashnikov et al. ‘18

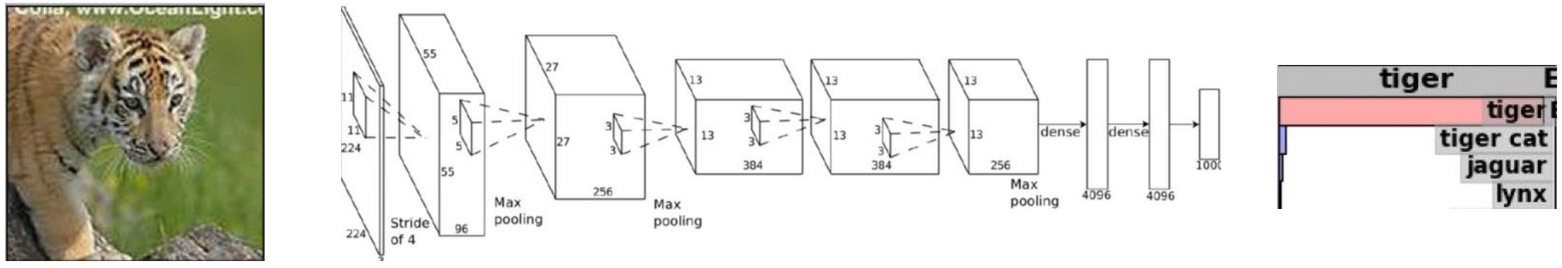
Not just games and robots!





Cathy Wu

Why should we study this **now**?



1. Advances in deep learning

2. Advances in reinforcement learning

3. Advances in computational capability

Why should we study this **now**?

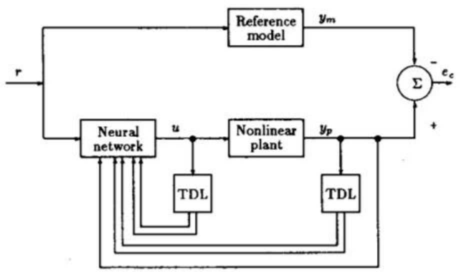
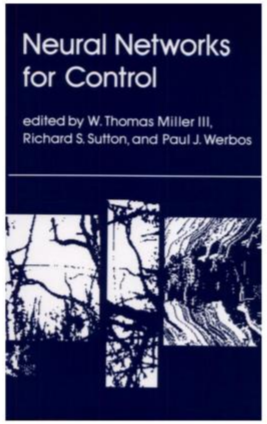
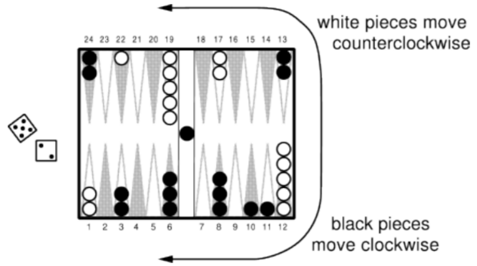


Fig. 21. Direct adaptive control of nonlinear plants using neural networks.





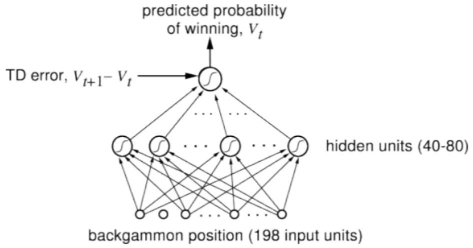


Table 11.1: Summary of TD-Gammon Results

Program	Hidden Units	Training Games	Opponents	Results
TD-Gam 0.0	40	300,000	other programs	tied for best
TD-Gam 1.0	80	300,000	Robertie, Magriel, ...	−13 pts / 51 games
TD-Gam 2.0	40	800,000	various Grandmasters	−7 pts / 38 games
TD-Gam 2.1	80	1,500,000	Robertie	−1 pt / 40 games
TD-Gam 3.0	80	1,500,000	Kazaros	+6 pts / 20 games

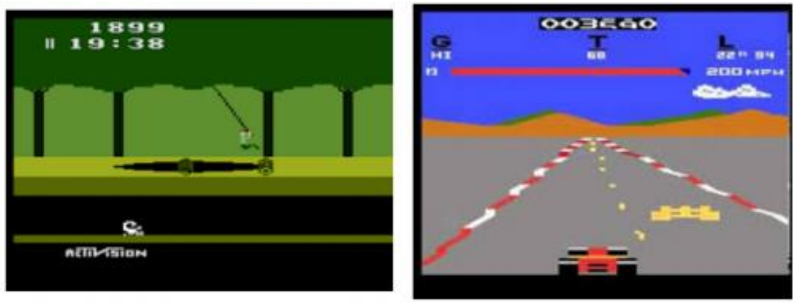
Tesauro, 1995

This dissertation demonstrates how we can possibly overcome the slow learning problem and tackle non-Markovian environments, making reinforcement learning more practical for realistic robot tasks:

- Reinforcement learning can be naturally integrated with artificial neural networks to obtain high-quality generalization, resulting in a significant learning speedup. Neural networks are used in this dissertation, and they generalize effectively even in the presence of noise and a large number of binary and real-valued inputs.
- Reinforcement learning agents can save many learning trials by using an action model, which can be learned on-line. With a model, an agent can mentally experience the effects of its actions without actually executing them. Experience replay is a simple technique that implements this idea, and is shown to be effective in reducing the number of action executions required.
- Reinforcement learning agents can take advantage of instructive training instances provided by human teachers, resulting in a significant learning speedup. Teaching can also help learning agents avoid local optima during the search for optimal control. Simulation experiments indicate that even a small amount of teaching can save agents many learning trials.
- Reinforcement learning agents can significantly reduce learning time by hierarchical learning— they first solve elementary learning problems and then combine solutions to the elementary problems to solve a complex problem. Simulation experiments indicate that a robot with hierarchical learning can solve a complex problem, which otherwise is hardly solvable within a reasonable time.
- Reinforcement learning agents can deal with a wide range of non-Markovian environments by having a memory of their past. Three memory architectures are discussed. They work reasonably well for a variety of simple problems. One of them is also successfully applied to a nontrivial non-Markovian robot task.

L.-J. Lin, "Reinforcement learning for robots using neural networks." 1993

Why should we study this **now**?



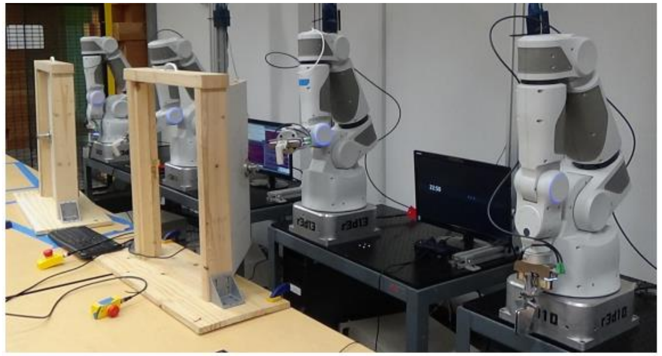
Atari games:

Q-learning:

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, et al. “Playing Atari with Deep Reinforcement Learning”. (2013).

Policy gradients:

J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. “Trust Region Policy Optimization”. (2015).
V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, et al. “Asynchronous methods for deep reinforcement learning”. (2016).



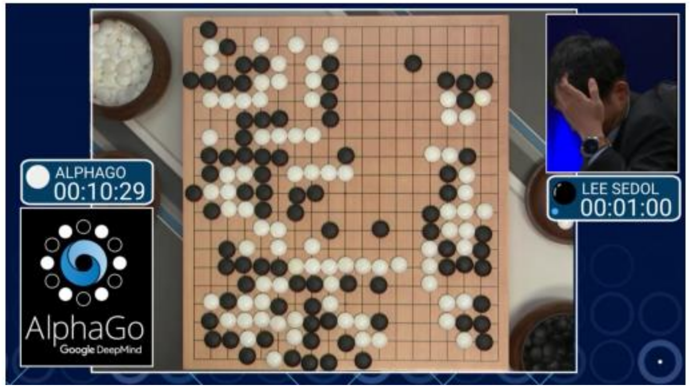
Real-world robots:

Guided policy search:

S. Levine*, C. Finn*, T. Darrell, P. Abbeel. “End-to-end training of deep visuomotor policies”. (2015).

Q-learning:

D. Kalashnikov et al. “QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation”. (2018).



Beating Go champions:

Supervised learning + policy gradients + value functions + Monte Carlo tree search:

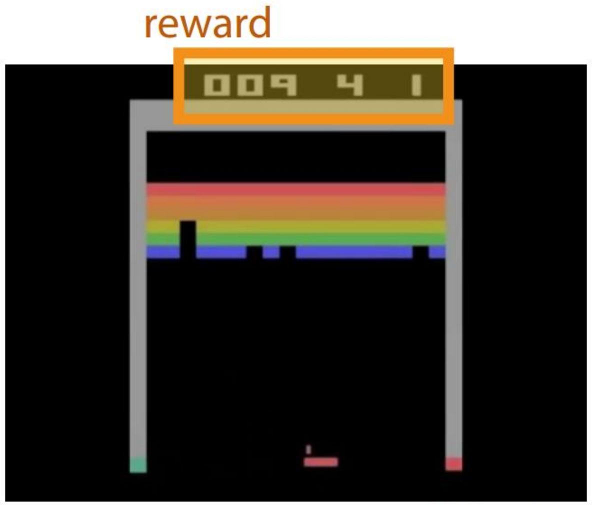
D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, et al. “Mastering the game of Go with deep neural networks and tree search”. Nature (2016).

What other problems do we need to solve to enable real-world sequential decision making?

Beyond learning from reward

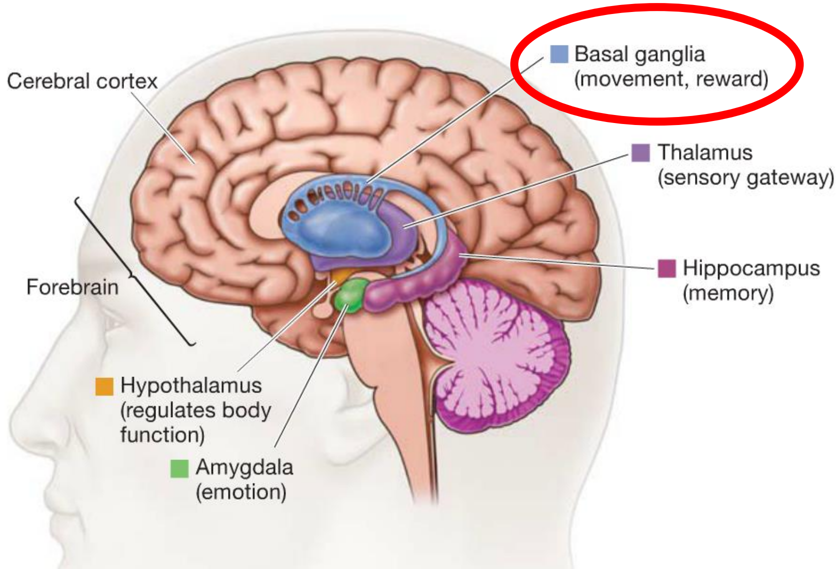
- Basic reinforcement learning deals with maximizing rewards
- This is not the only problem that matters for sequential decision making!
- We will cover more advanced topics
    - Learning reward functions from example (inverse reinforcement learning)
    - Transferring knowledge between domains (transfer learning, meta-learning)
    - Learning to predict and using prediction to act

Where do rewards come from?



Mnih et al. '15

reinforcement learning agent



[–] LazyOptimist 32 points 5 days ago

As human agents, we are accustomed to operating with rewards that are so sparse that we only experience them once or twice in a lifetime, if at all.

Are there other forms of supervision?

- Learning from demonstrations
  - Directly copying observed behavior
  - Inferring rewards from observed behavior (inverse reinforcement learning)
- Learning from observing the world
  - Learning to predict
  - Unsupervised learning
- Learning from other tasks
  - Transfer learning
  - Meta-learning: learning to learn

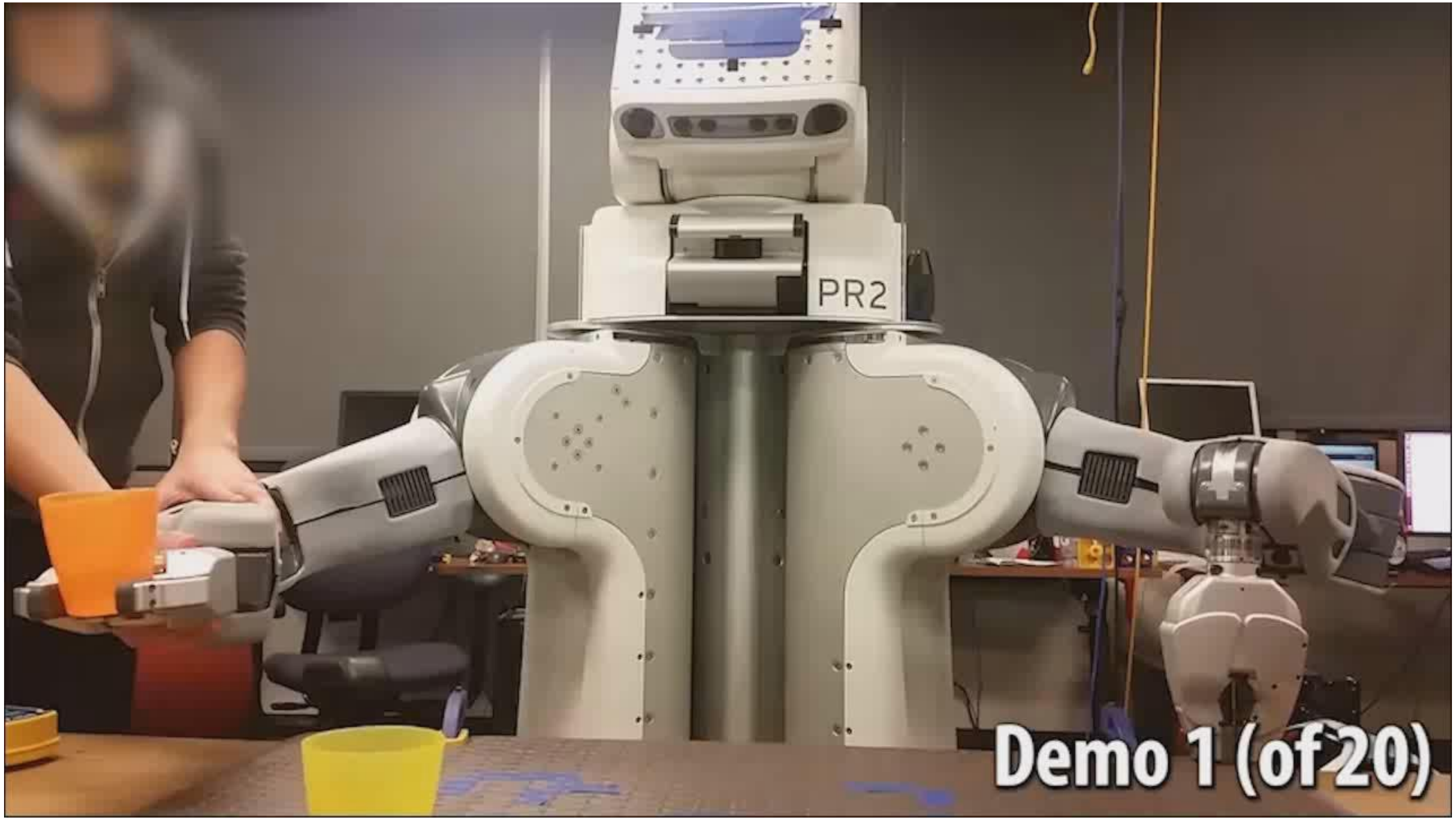
Imitation learning

Bojarski et al. 2016

More than imitation: inferring intentions

Warneken & Tomasello

Inverse RL examples



Finn et al. 2016

Prediction

"the idea that we **predict the consequences of our motor commands** has emerged as an important theoretical concept in all aspects of sensorimotor control"

Prediction Precedes Control in Motor Learning

J. Randall Flanagan,[1,*] Philipp Vetter,[2] Roland S. Johansson,[3] and Daniel M. Wolpert[2]

Procedures for details). Figure 1 shows, for a single subject, the hand path (top trace) and the grip (middle)

Predicting the Consequences of Our Own Actions: The Role of Sensorimotor Context Estimation

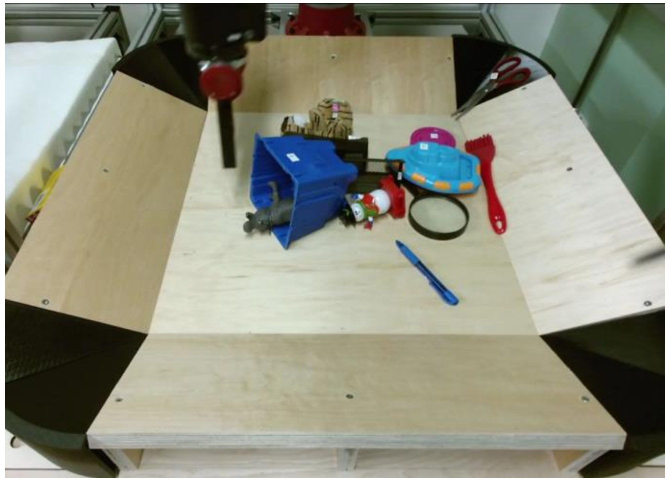
Sarah J. Blakemore, Susan J. Goodbody, and Daniel M. Wolpert

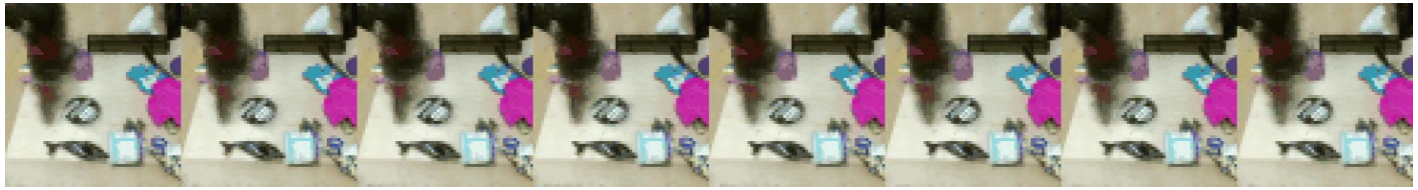
Sobell Department of Neurophysiology, Institute of Neurology, University College London, London WC1N 3BG,

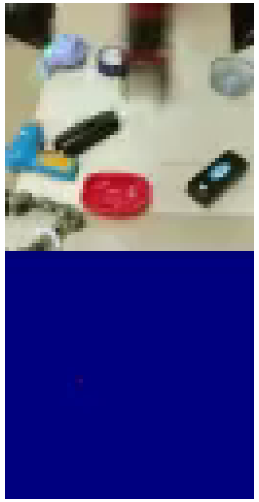
Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects

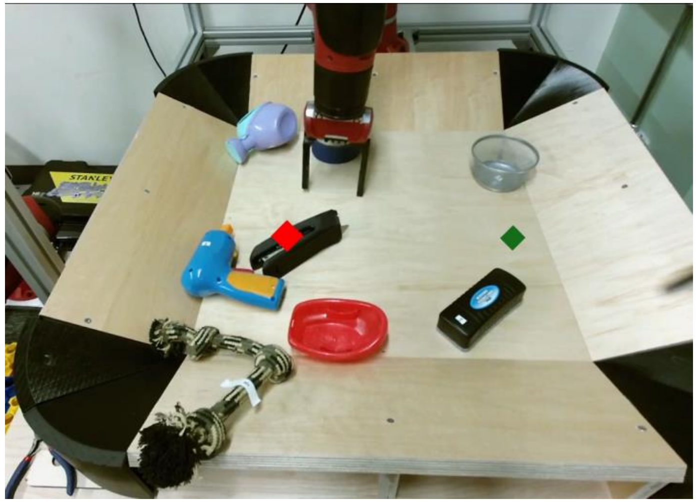
Rajesh P. N. Rao[1] and Dana H. Ballard[2]

Prediction for real-world control



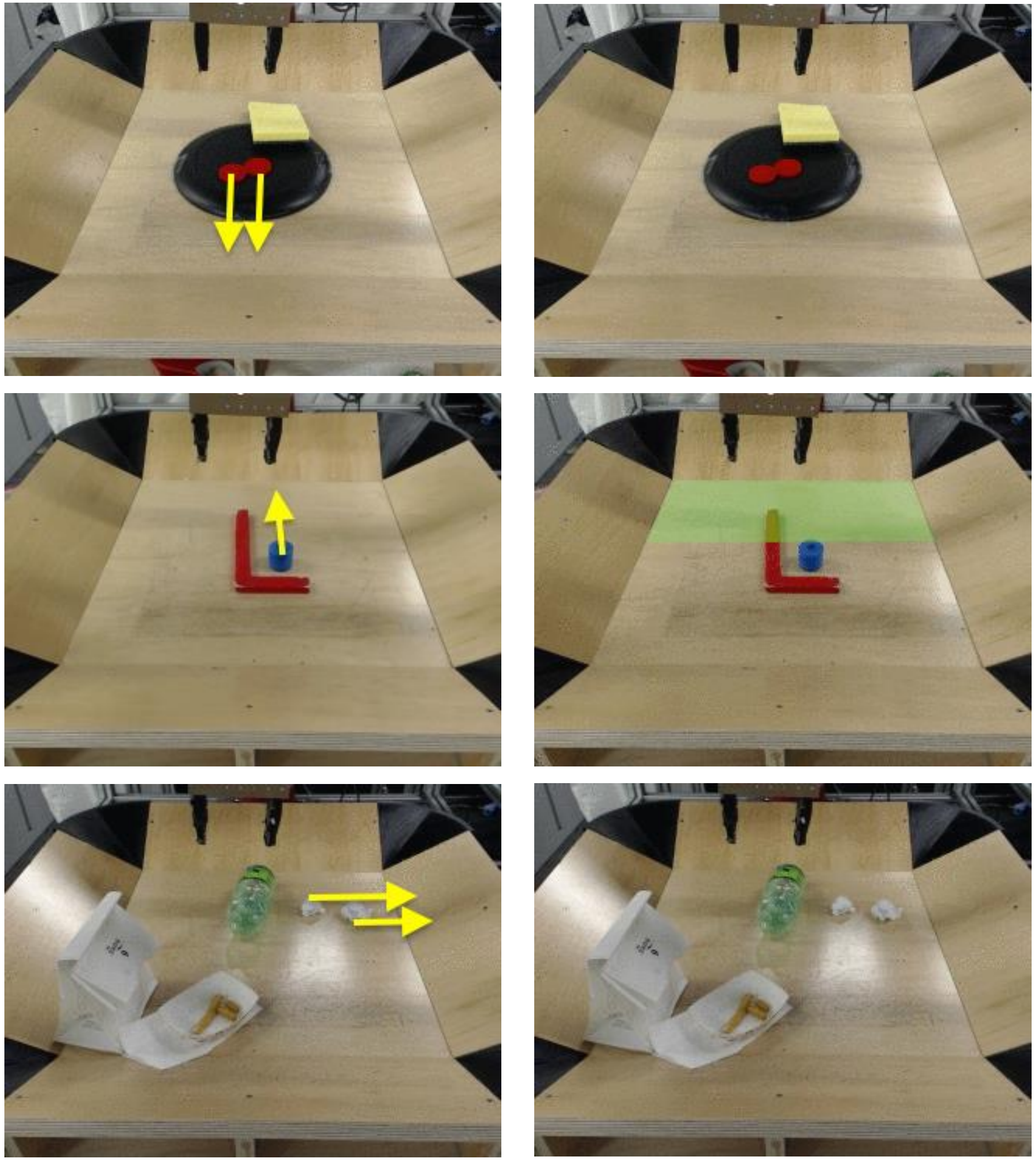






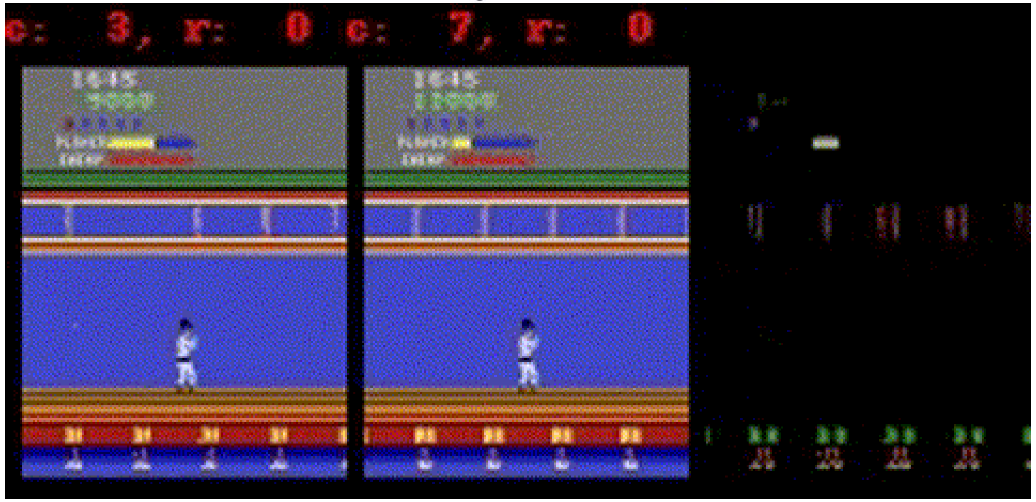
Ebert et al. 2017

Using tools with predictive models



Xie et al. 2019

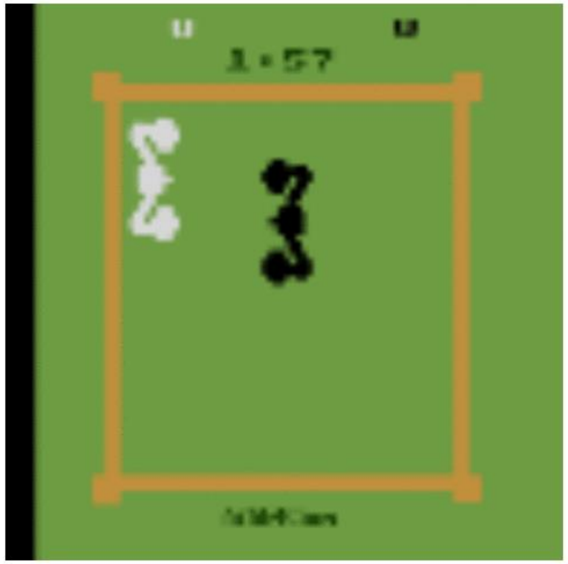
Playing games with predictive models



predicted

real

But sometimes there are issues…

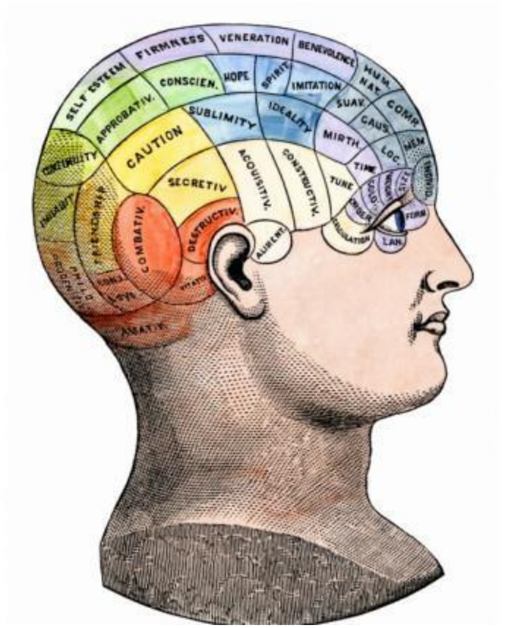


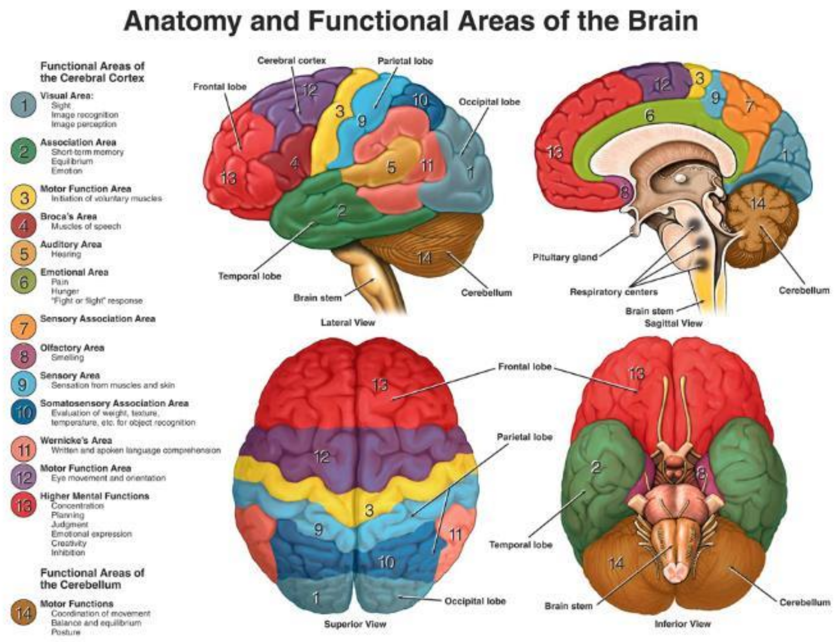
Kaiser et al. 2019

How do we build intelligent machines?

How do we build intelligent machines?

- Imagine you have to build an intelligent machine, where do you start?



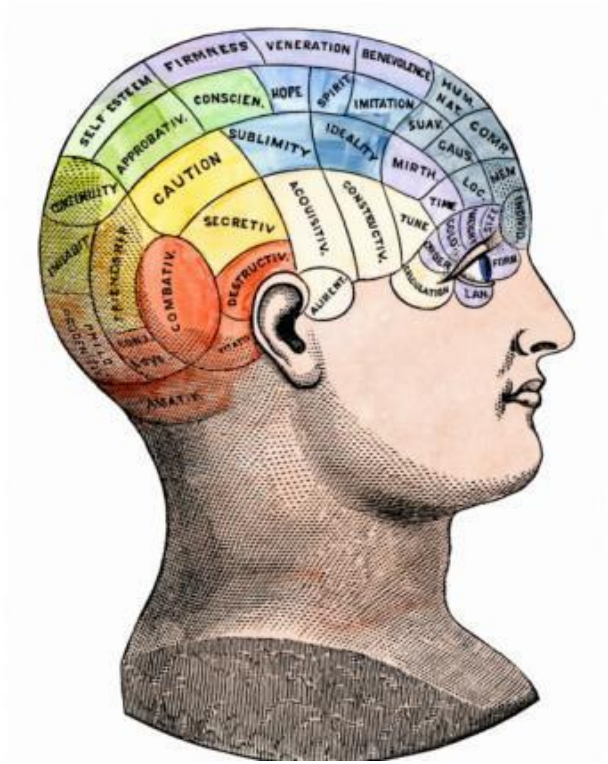


Learning as the basis of intelligence

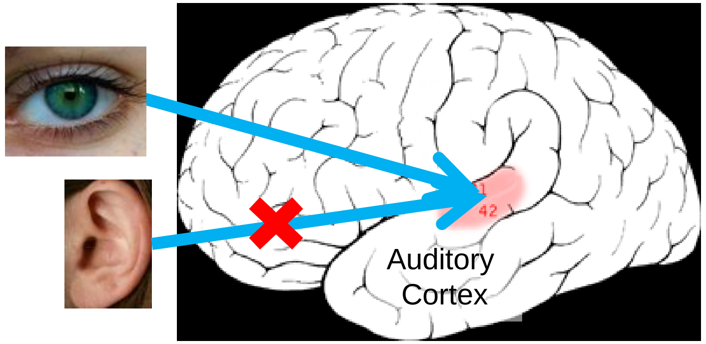
- Some things we can all do (e.g. walking)
- Some things we can only learn (e.g. driving a car)
- We can learn a huge variety of things, including very difficult things
- Therefore our learning mechanism(s) are likely powerful enough to do everything we associate with intelligence
  - But it may still be very convenient to “hard-code” a few really important bits

A single algorithm?

- An algorithm for each “module”?
- Or a single flexible algorithm?



Seeing with your tongue

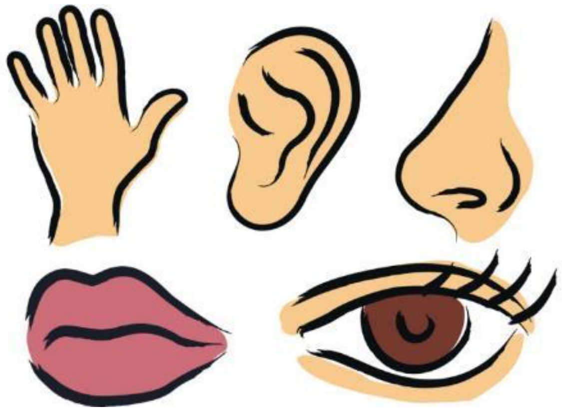


[BrainPort; Martinez et al; Roe et al.]

adapted from A. Ng

What must that single algorithm do?

- Interpret rich sensory inputs



- Choose complex actions

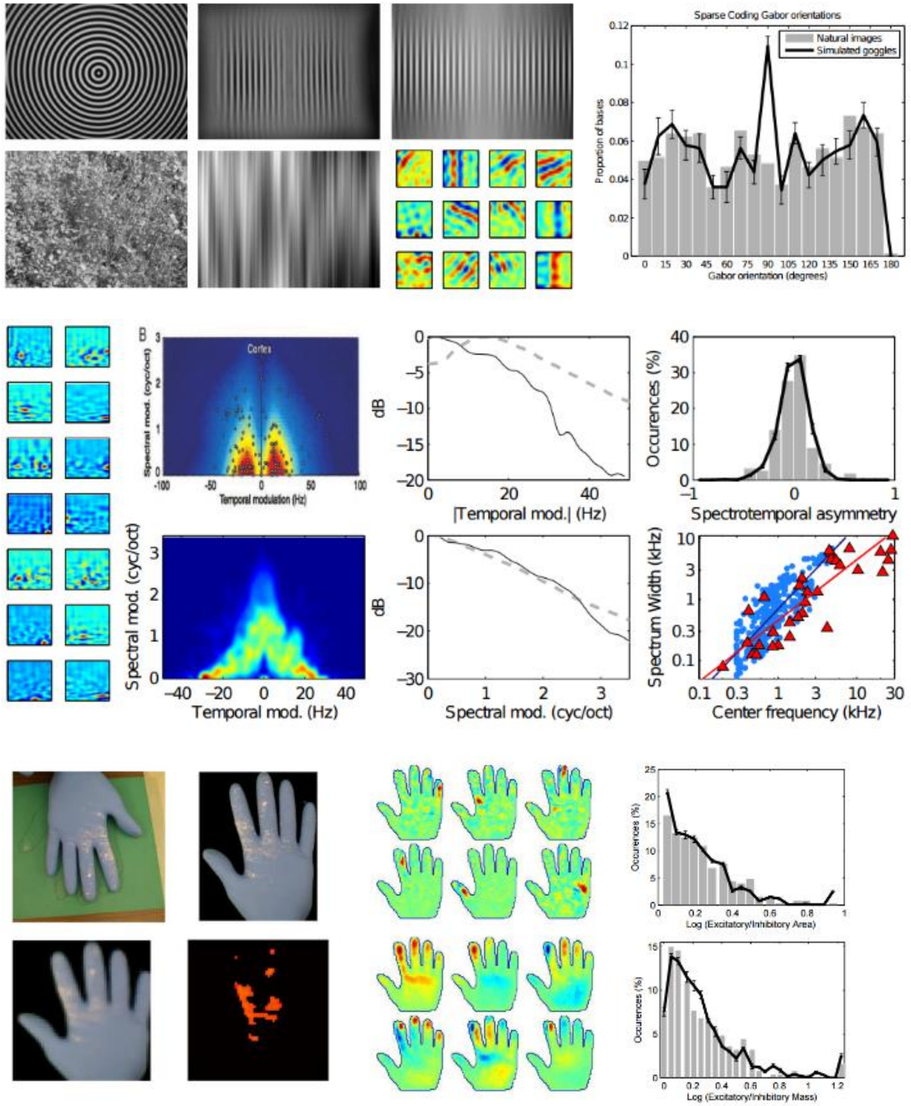
Why deep reinforcement learning?

- Deep = can process complex sensory input
  - …and also compute really complex functions
- Reinforcement learning = can choose complex actions

Some evidence in favor of deep learning

Unsupervised learning models of primary cortical receptive fields and receptive field plasticity

Andrew Saxe, Maneesh Bhand, Ritvik Mudur, Bipin Suresh, Andrew Y. Ng
Department of Computer Science
Stanford University
{asaxe, mbhand, rmudur, bipins, ang}@cs.stanford.edu



Some evidence for reinforcement learning

- Percepts that anticipate reward become associated with similar firing patterns as the reward itself
- Basal ganglia appears to be related to reward system
- Model-free RL-like adaptation is often a good fit for experimental data of animal adaptation
  - But not always…

Reinforcement learning in the brain

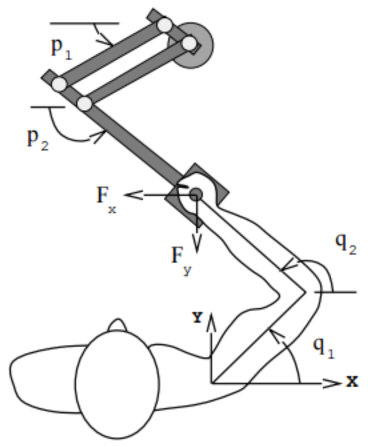
Yael Niv
Psychology Department & Princeton Neuroscience Institute, Princeton University

What can deep learning & RL do well now?

- Acquire high degree of proficiency in domains governed by simple, known rules
- Learn simple skills with raw sensory inputs, given enough experience
- Learn from imitating enough human-provided expert behavior

What has proven challenging so far?

- Humans can learn incredibly quickly
  - Deep RL methods are usually slow
- Humans can reuse past knowledge
  - Transfer learning in deep RL is an open problem
- Not clear what the reward function should be
- Not clear what the role of prediction should be



Instead of trying to produce a program to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain.

- Alan Turing