Deep Reinforcement Learning
CS 294 - 112

Course logistics

Class Information & Resources

Sergey Levine
Assistant Professor
UC Berkeley

Abhishek Gupta
PhD Student
UC Berkeley

Josh Achiam
PhD Student
UC Berkeley

- Course website: rll.berkeley.edu/deeprlcourse/
- Piazza: UC Berkeley, CS294-112
- Subreddit (for non-enrolled students): www.reddit.com/r/berkeleydeeprlcourse/
- Office hours: after class each day (but not today), sign up in advance for a 10-minute slot on the course website

Prerequisites & Enrollment

- All enrolled students must have taken CS189, CS289, or CS281A
  - Please contact Sergey Levine if you haven't
- Please enroll for 3 units
- Students on the wait list will be notified as slots open up
- Lectures will be recorded
  - Since the class is full, please watch the lectures online if you are not enrolled

What you should know

- Assignments will require training neural networks with standard automatic differentiation packages (TensorFlow by default)
- Review Section
  - Josh Achiam will teach a review section in week 3
  - You should be able to at least do the TensorFlow MNIST tutorial (if not, come to the review section and ask questions!)

What we’ll cover

- Full syllabus on course website

1. From supervised learning to decision making
2. Basic reinforcement learning: Q-learning and policy gradients
3. Advanced model learning and prediction, distillation, reward learning
4. Advanced deep RL: trust region policy gradients, actor-critic methods, exploration
5. Open problems, research talks, invited lectures

Assignments

1. Homework 1: Imitation learning (control via supervised learning)
2. Homework 2: Policy gradients ("REINFORCE")
3. Homework 3: Q learning with convolutional neural networks
4. Homework 4: Model-based reinforcement learning
5. Final project: Research-level project of your choice (form a group of up to 2-3 students, you're welcome to start early!)

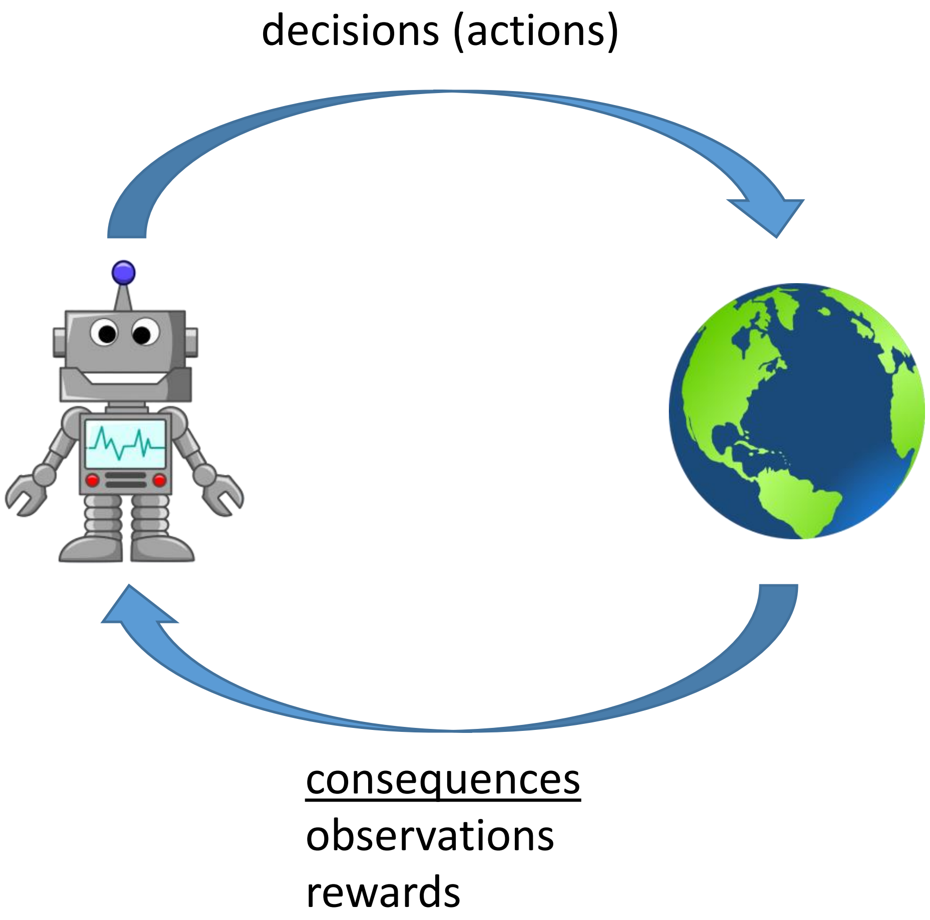
Grading: 40% homework (10% each), 60% project

Your “Homework” Today

1. Sign up for Piazza (see course website)
2. Start forming your final project groups, unless you want to work alone, which is fine
3. Fill out the enrolled student survey if you haven’t already!
4. Check out the TensorFlow MNIST tutorial, unless you’re a TensorFlow pro

What is reinforcement learning, and why should we care?

What is reinforcement learning?



Examples

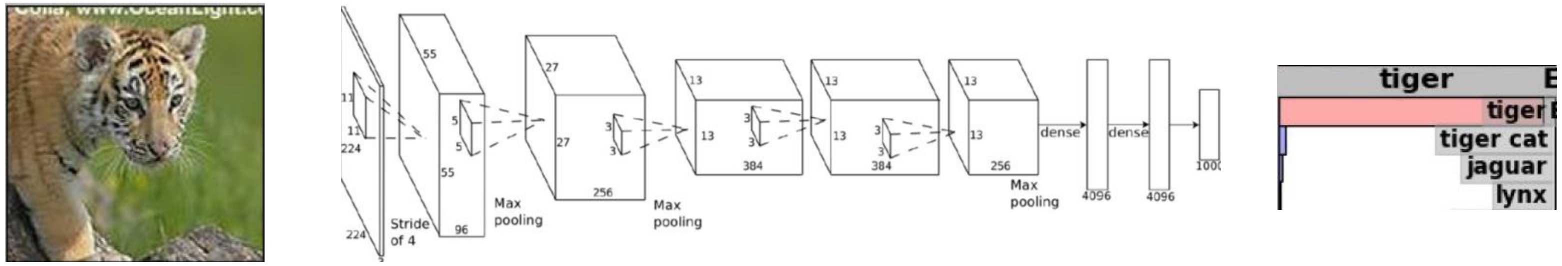
Actions: muscle contractions
Observations: sight, smell
Rewards: food

Actions: motor current or torque
Observations: camera images
Rewards: task success measure
(e.g., running speed)

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

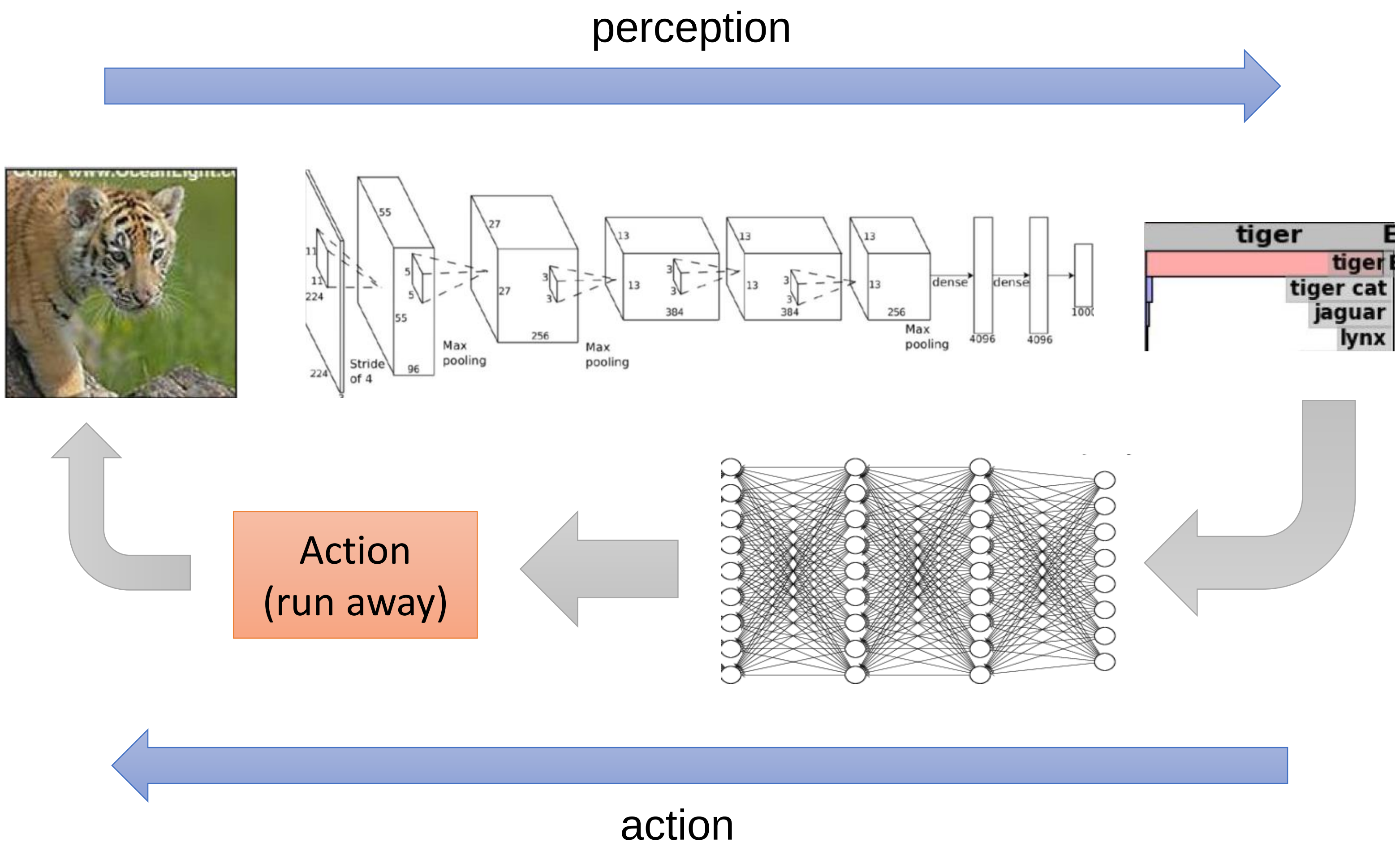
What is deep RL, and why should we care?

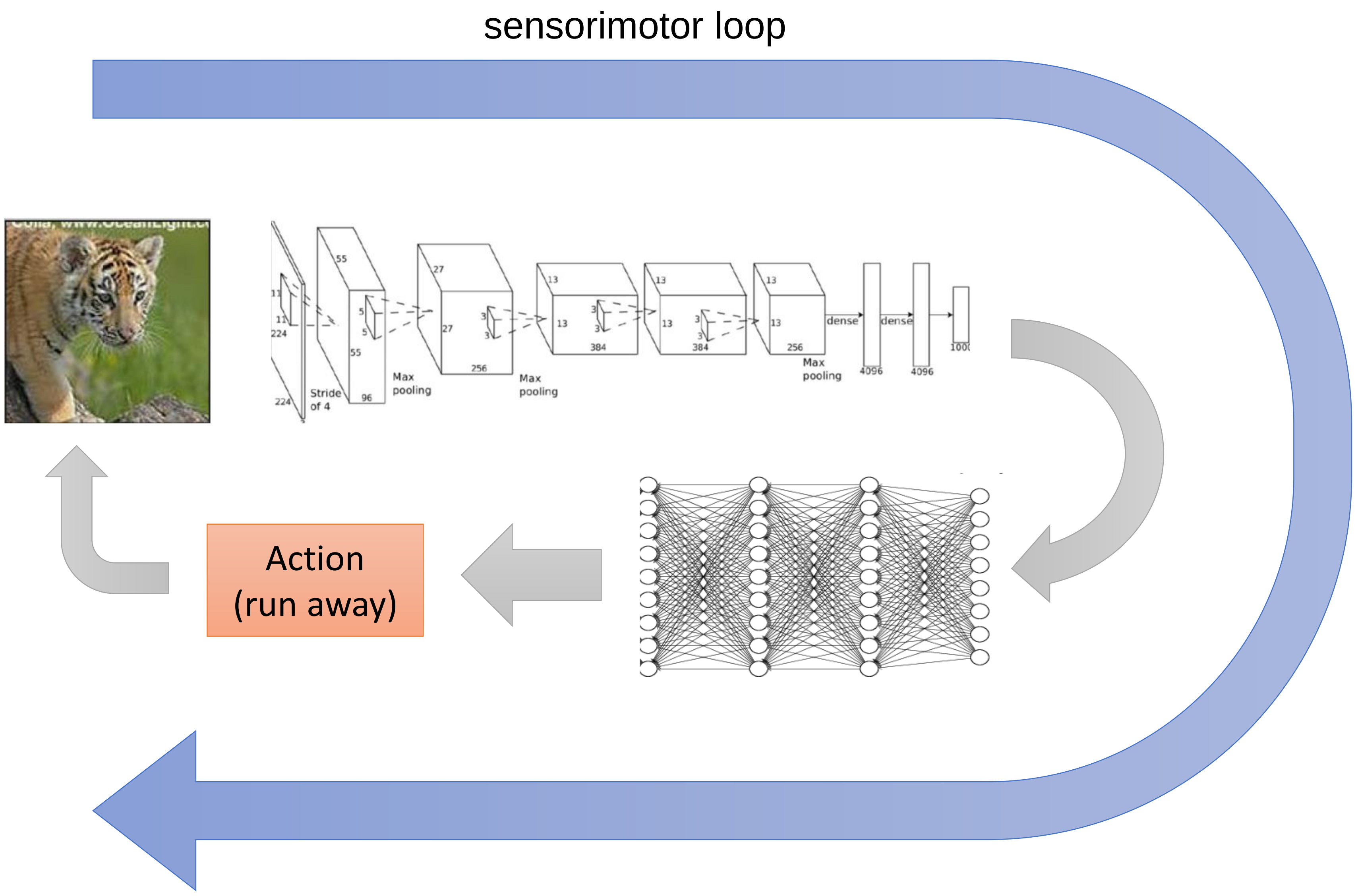
Deep learning: end-to-end training of expressive, multi-layer models



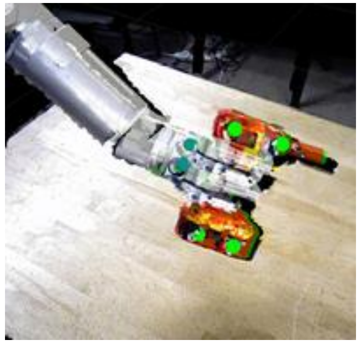
Deep models are what allow reinforcement learning algorithms to solve complex problems end to end!

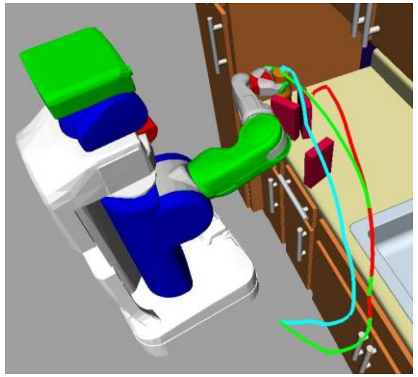
What does end-to-end learning mean for sequential decision making?


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


sensorimotor loop
Action
(run away)
Max
pooling
Max
pooling
Max
pooling
Stride
of 4
dense
dense

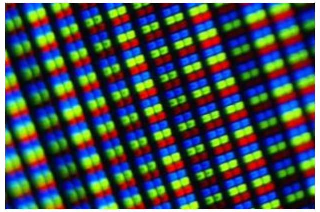
Example: robotics

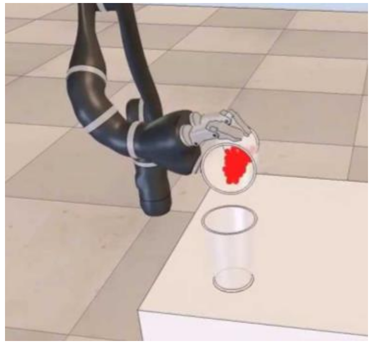


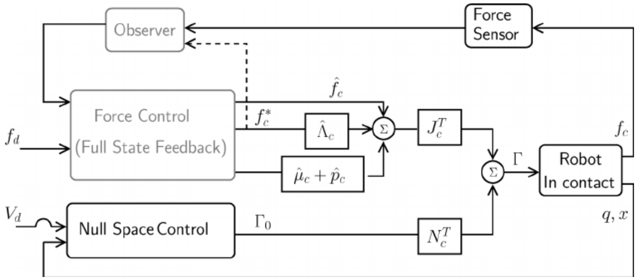


robotic control pipeline

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



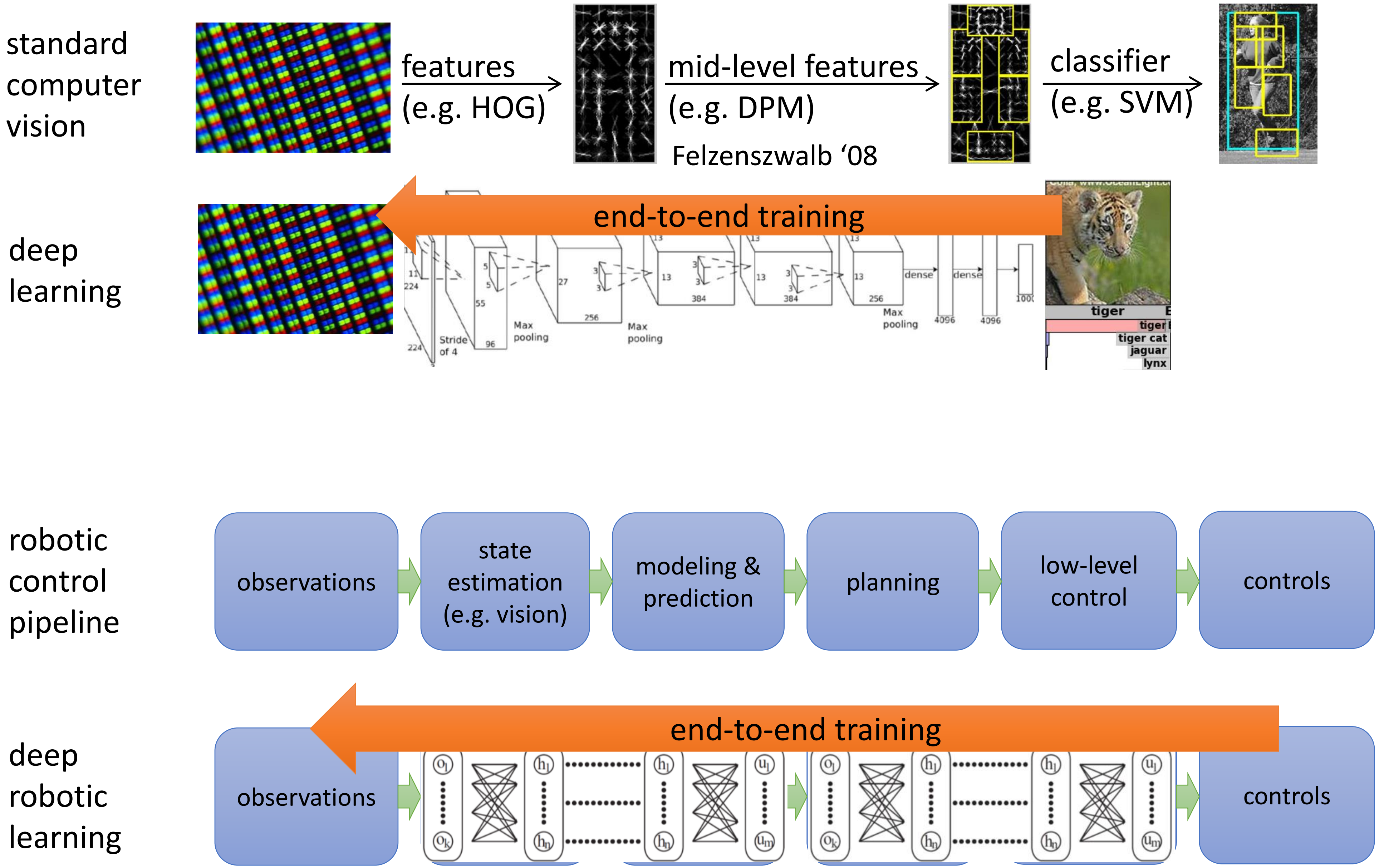




Example: playing video games

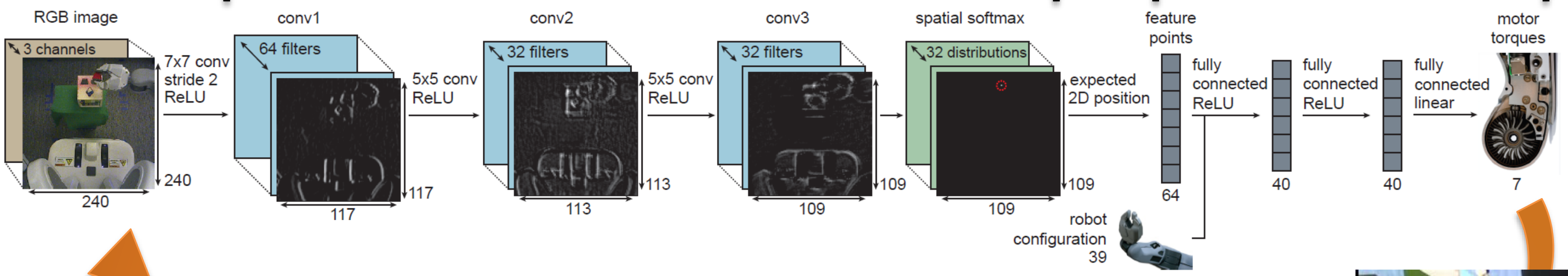
video game AI pipeline

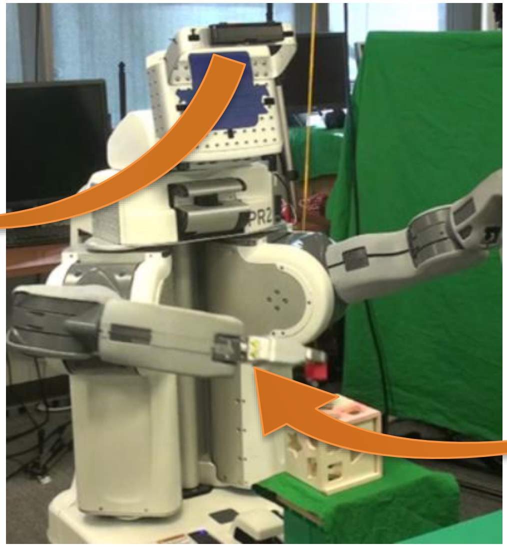
game API → extract relevant features → state machine for behavior → planner → low-level bot control → controls

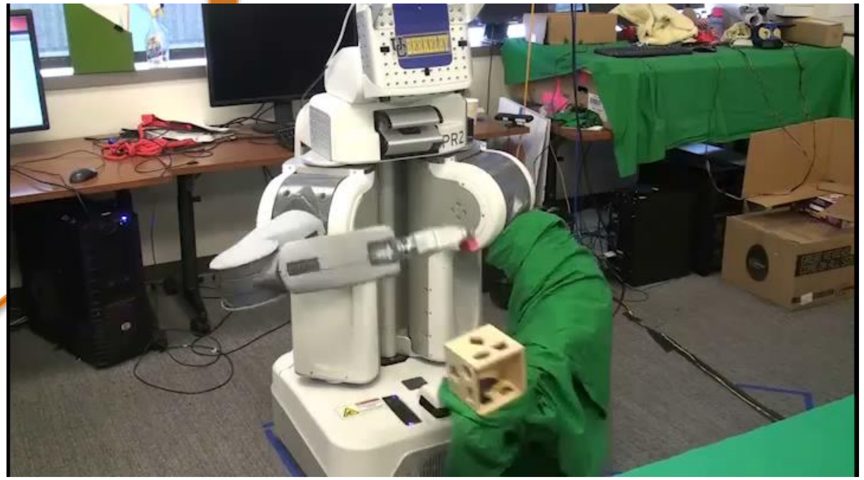

standard computer vision
features (e.g. HOG)
mid-level features (e.g. DPM)
Felzenszwalb ‘08
classifier (e.g. SVM)
deep learning
end-to-end training
Stride of 4
Max pooling
Max pooling
Max pooling
dense
dense
tiger
tiger cat
jaguar
lynx
robotic control pipeline
observations
state estimation (e.g. vision)
modeling & prediction
planning
low-level control
controls
deep robotic learning
end-to-end training
observations
controls

tiny, highly specialized
“visual cortex”

tiny, highly specialized
“motor cortex”





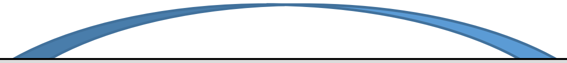


no direct supervision

actions have consequences

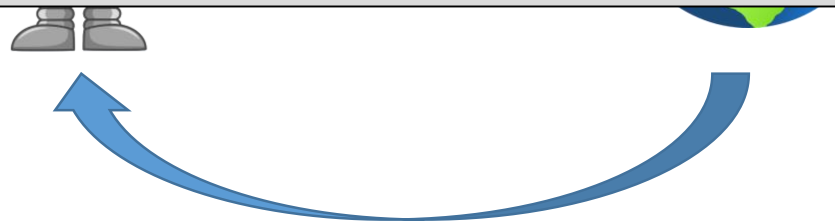
The reinforcement learning problem

decisions (actions)



Actions: motor current or torque
Observations: camera images
Rewards: task success measure (e.g., running speed)

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



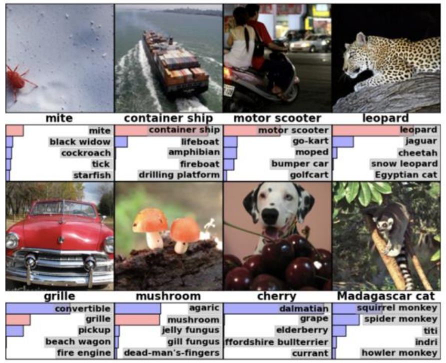
Observations: words in English
Rewards: BLEU score

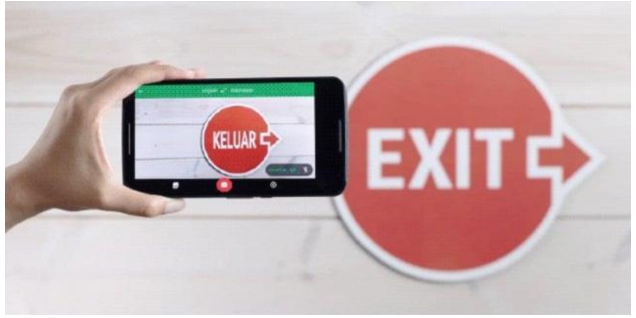
<u>consequences</u>
observations
rewards

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

When do we **not** need to worry about sequential decision making?

When your system is making single isolated decision, e.g. classification, regression
When that decision does not affect future decisions





When **should** we worry about sequential decision making?

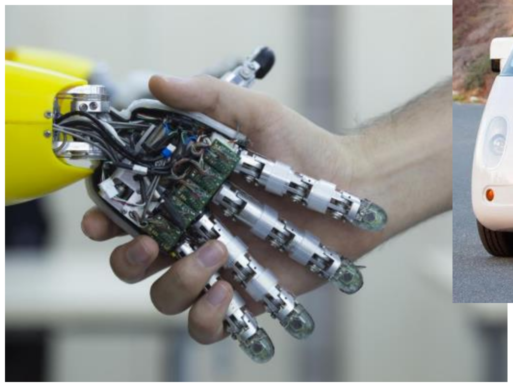
Limited supervision: you know **what** you want, but not **how** to get it

Actions have consequences

Common Applications

autonomous driving

business operations

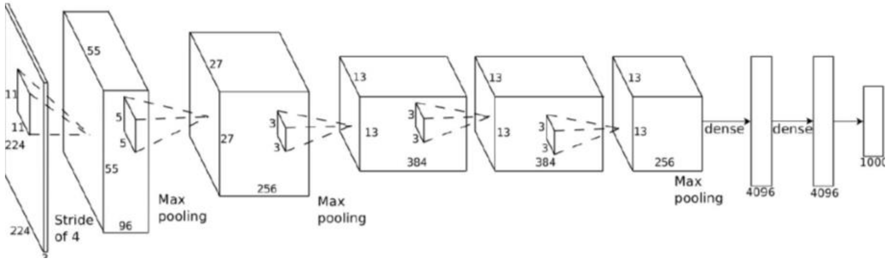


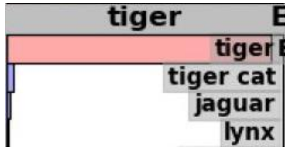
robotics

language & dialogue
(structured prediction)

finance

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**?

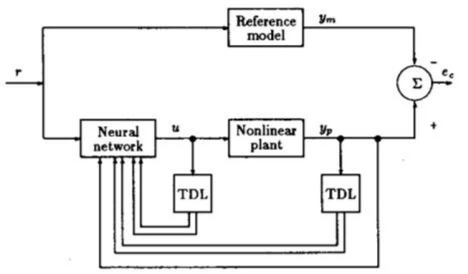
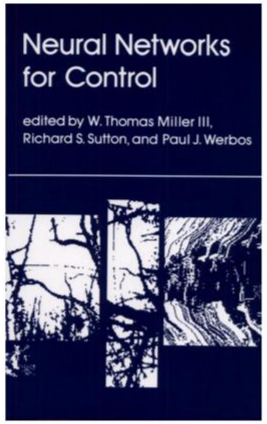
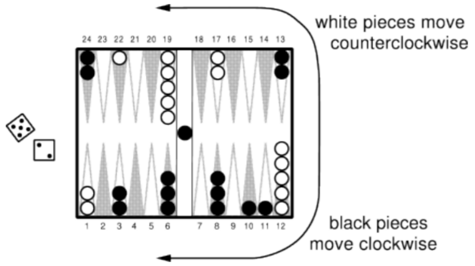


Fig. 21. Direct adaptive control of nonlinear plants using neural networks.





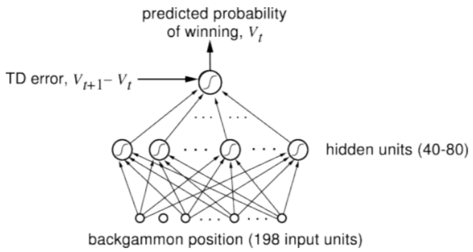


Table 11.1: Summary of TD-Gammon Results

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

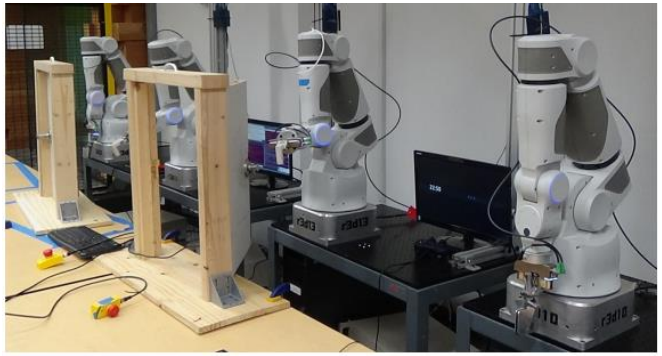
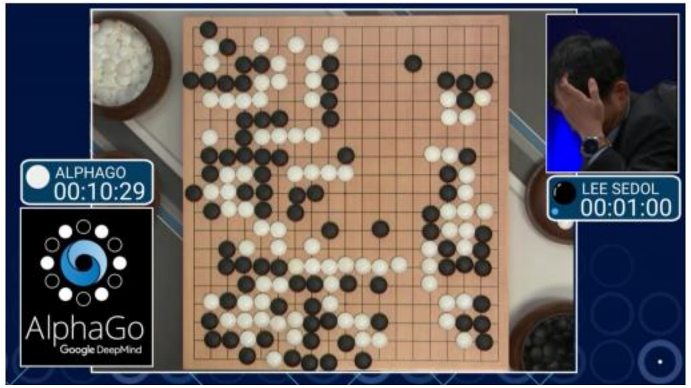
Tesauro, 1995

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:

S. Gu*, E. Holly*, T. Lillicrap, S. Levine. "Deep Reinforcement Learning for Robotic Manipulation with Asynchronous Off-Policy Updates". (2016).

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

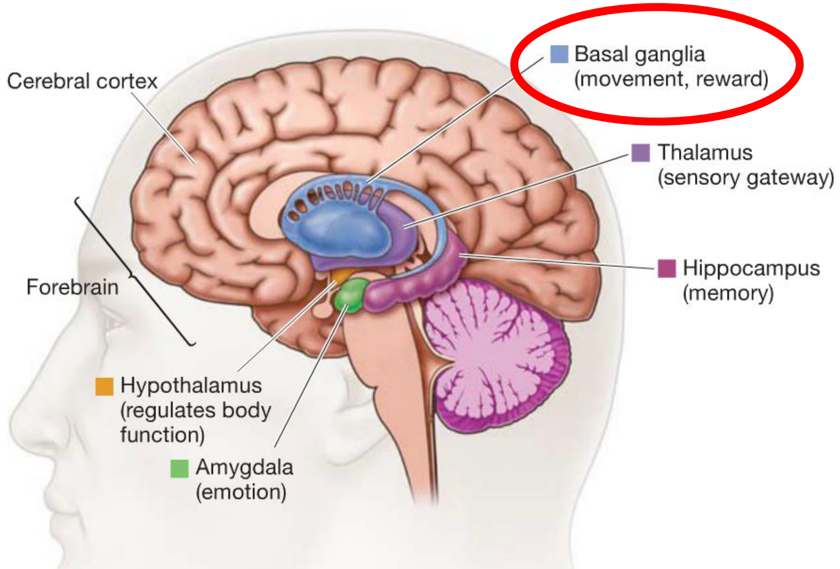
- Basic reinforcement learning deals with maximizing rewards
- This is not the only problem that matters for sequential decision making!
- We will cover more advanced topics
  - Learning reward functions from example (inverse reinforcement learning)
  - Transferring skills between domains
  - Learning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

Where do rewards come from?

reward

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

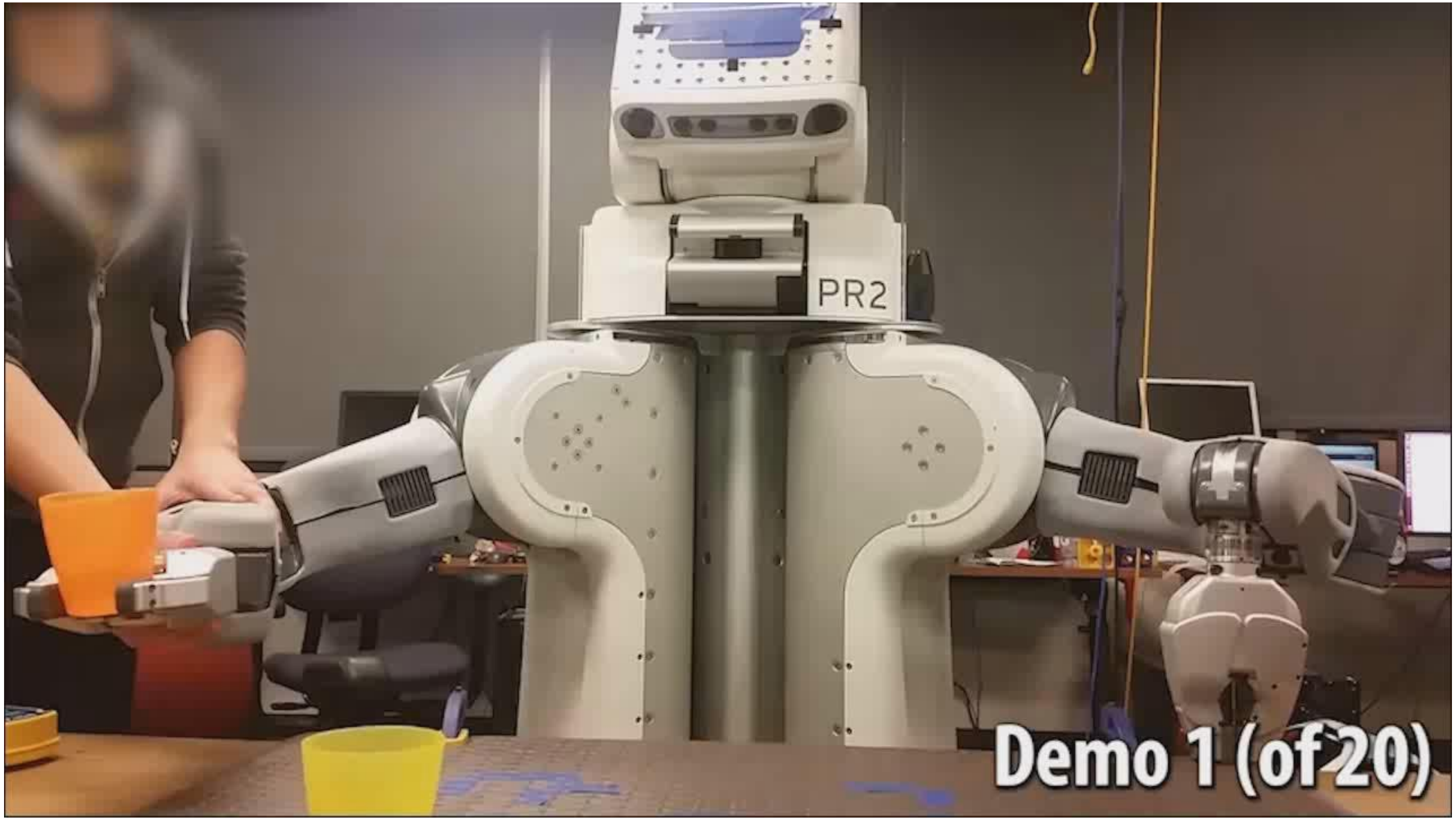
Imitation learning

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

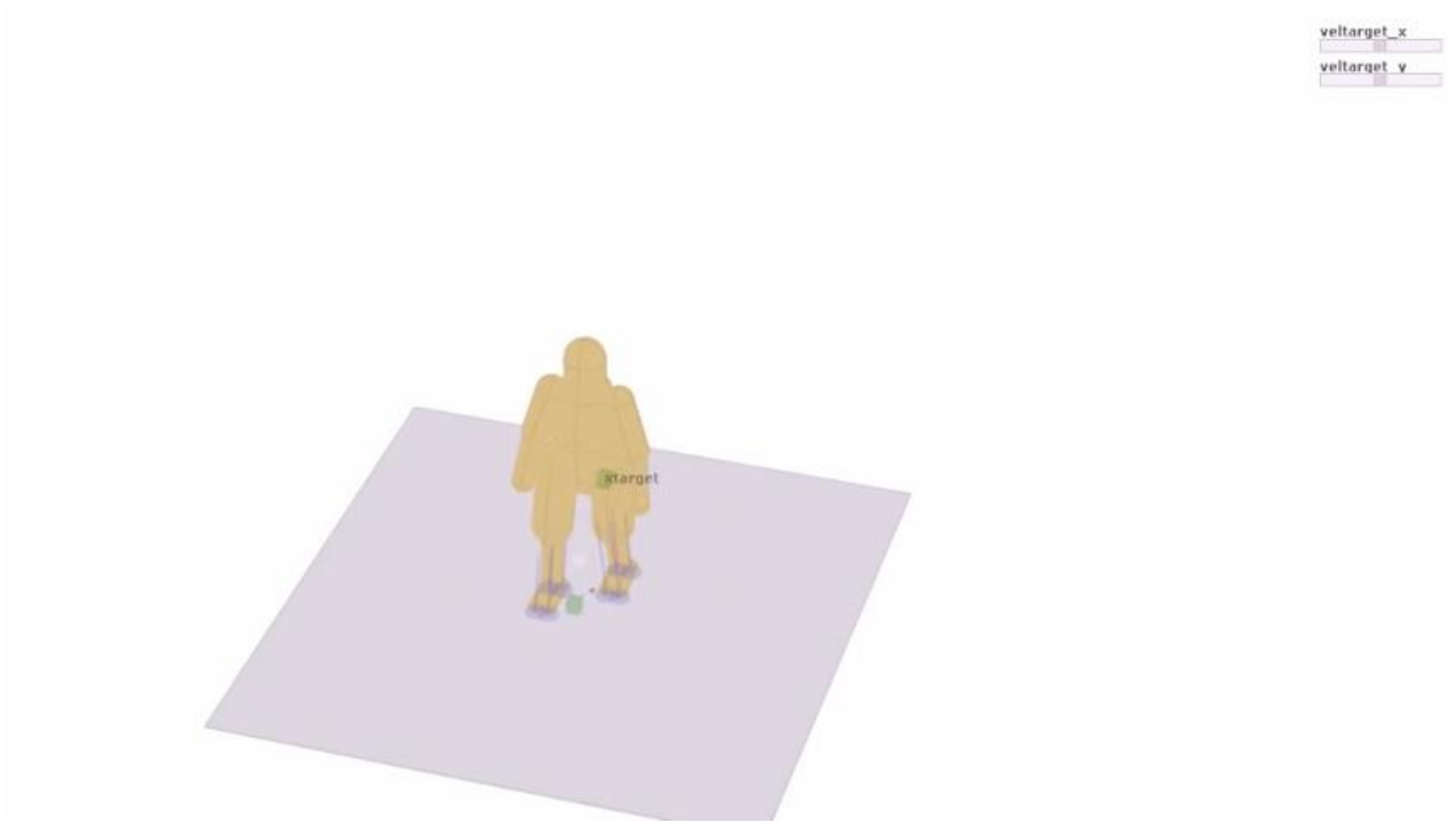
Sarah J. Blakemore, Susan J. Goodbody, and Daniel M. Wolpert

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

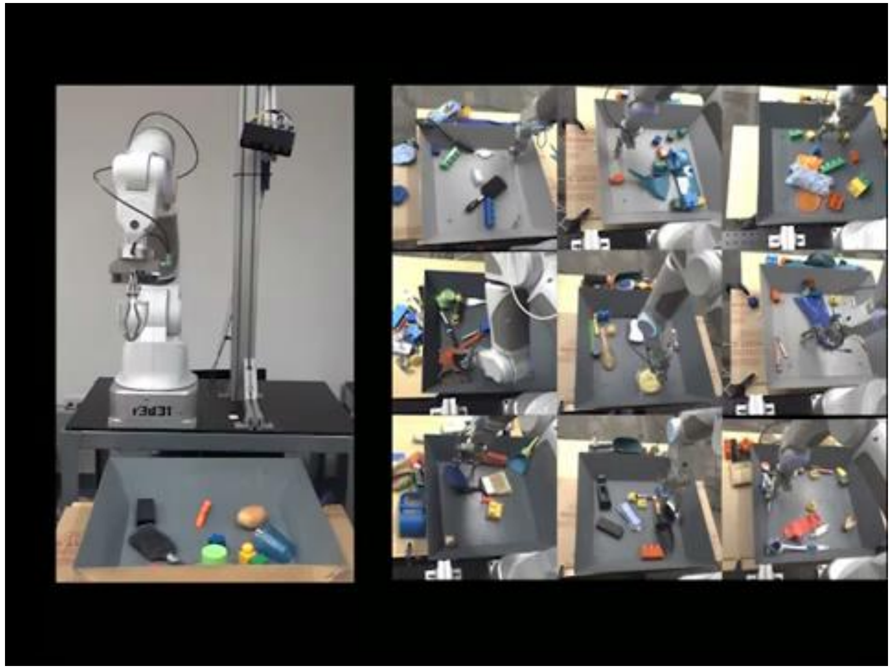
Rajesh P. N. Rao[1] and Dana H. Ballard[2]

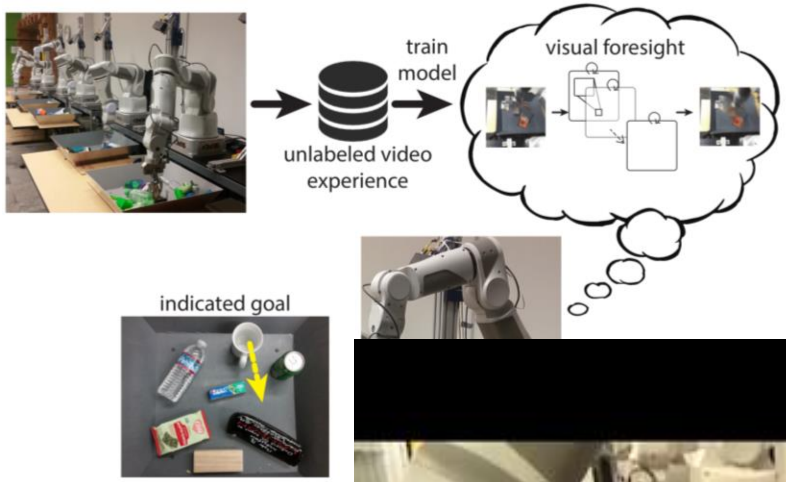
What can we do with a perfect model?



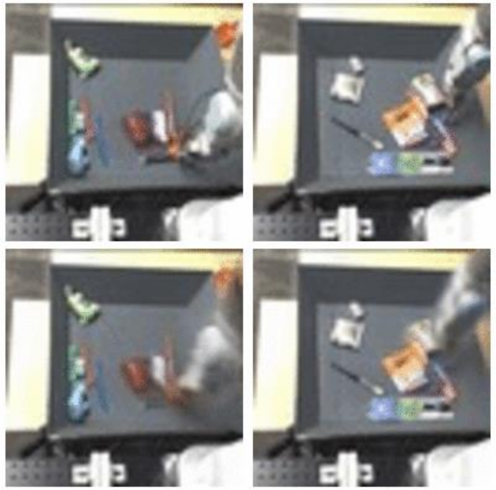
Mordatch et al. 2015

Prediction for real-world control





original
video



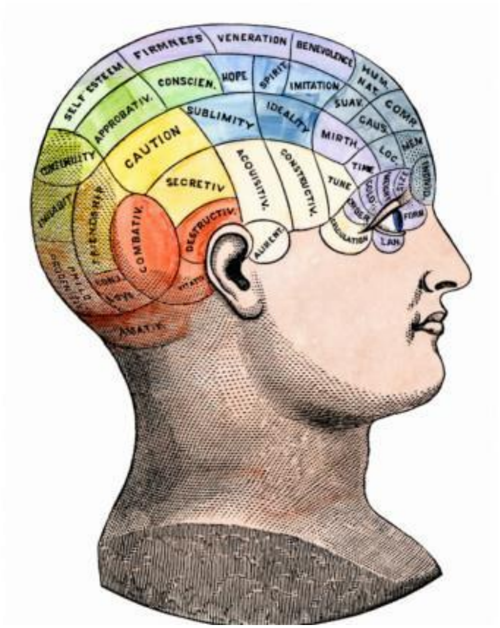
predictions

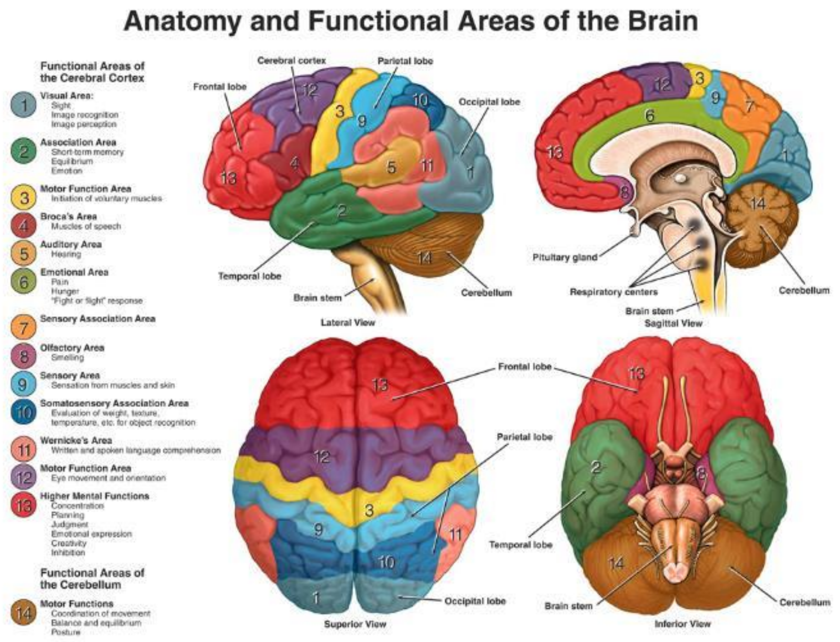
Finn et al. 2017

How do we build intelligent machines?

How do we build intelligent machines?

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



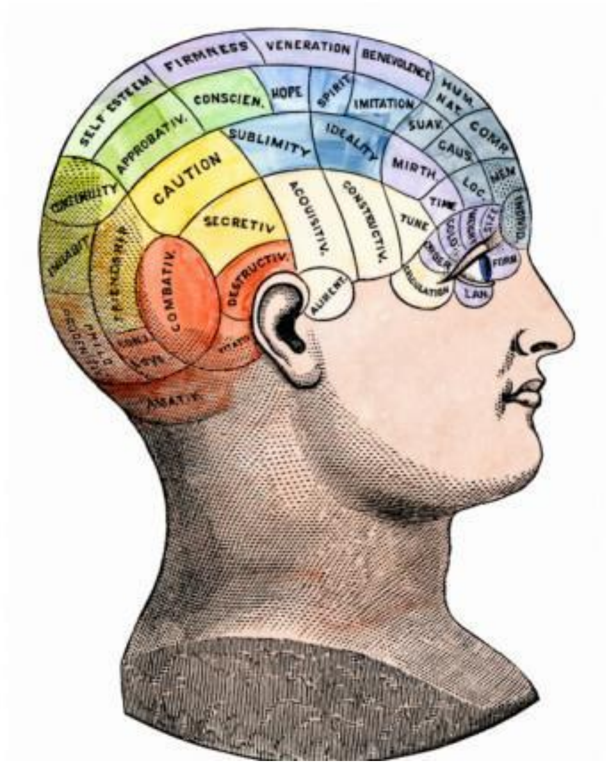


Learning as the basis of intelligence

- 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

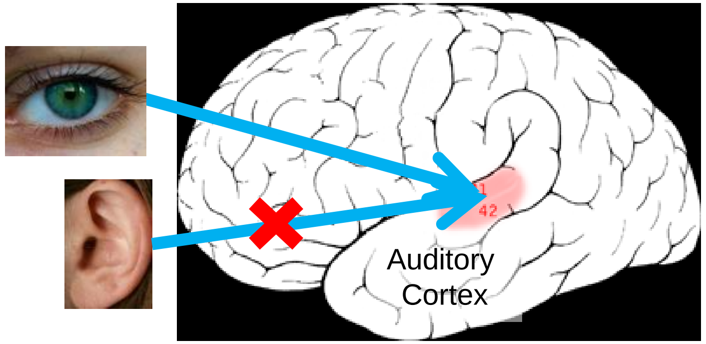
A single algorithm?

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



Seeing with your tongue

Human echolocation (sonar)

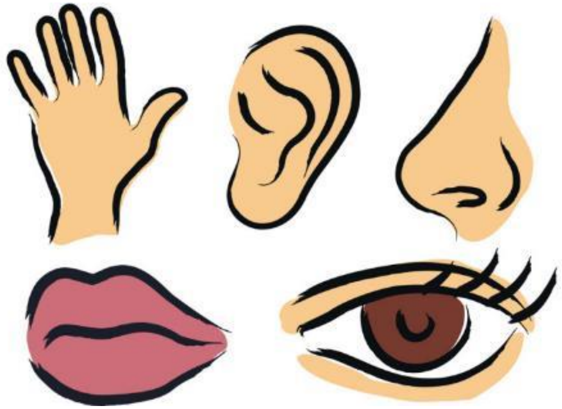


[BrainPort; Martinez et al; Roe et al.]

adapted from A. Ng

What must that single algorithm do?

- Interpret rich sensory inputs



- Choose complex actions

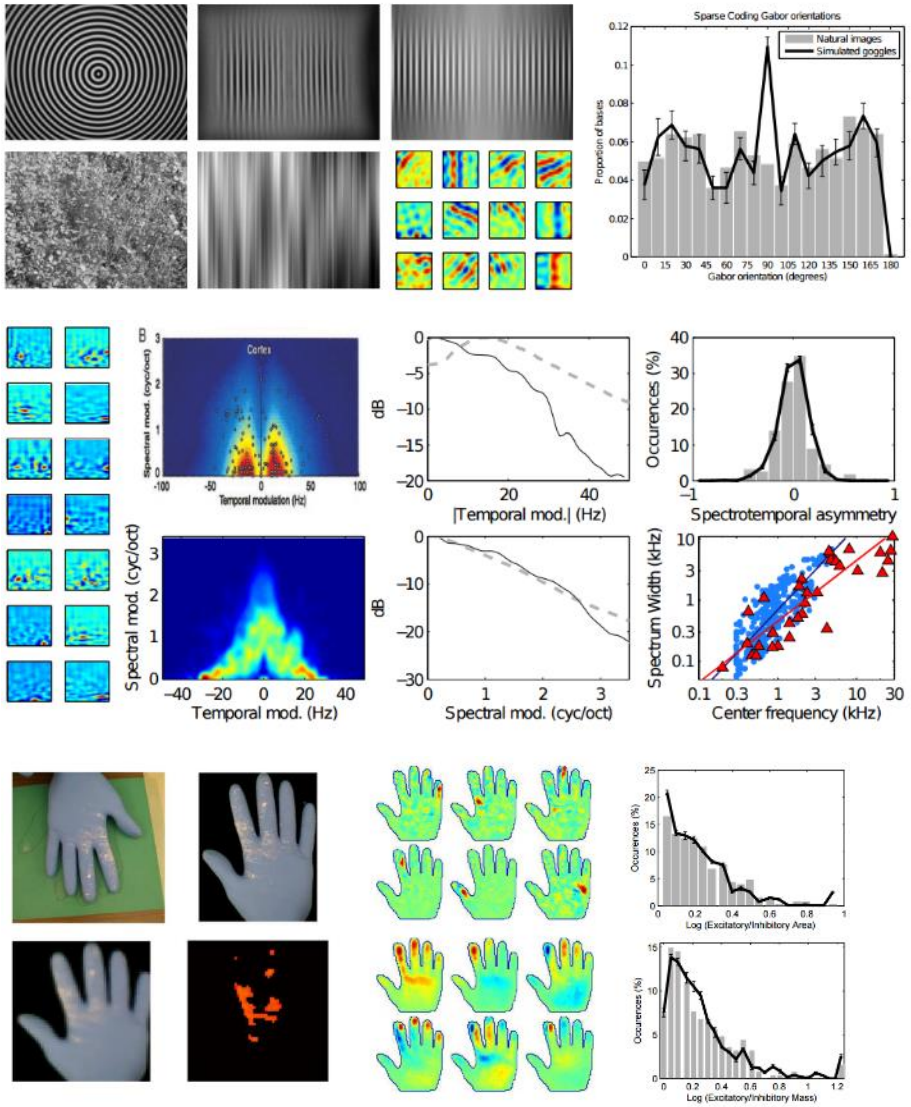
Why deep reinforcement learning?

- 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

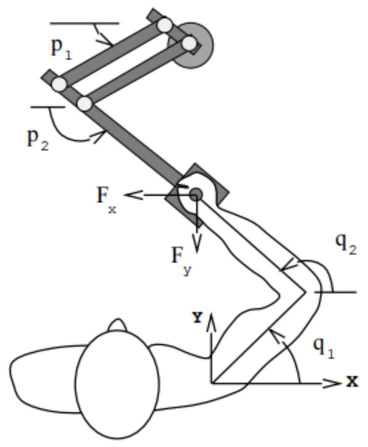
Yael Niv
Psychology Department & Princeton Neuroscience Institute, Princeton University

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