Learning About Sensorimotor Data

Rich Sutton

Reinforcement Learning and Artificial Intelligence Lab

University of Alberta, Canada

with thanks to

Adam White, Joseph Modayil, Thomas Degris, Patrick Pilarski, Csaba Szepesvari, Hamid Maei, Mark Ring, Anna Koop, Leah Hackman

Outline

- The sensorimotor approach to knowing
- Robot experiments
  - the need for multi-step prediction
- The Horde-of-demons architecture
- Remarks on gradient-TD algorithms

Intelligence

- Knowing a lot
- Being able to use what you know flexibly to achieve goals (maximize reward)

Intelligence

- Knowing a lot
- Being able to use what you know flexibly to achieve goals (maximize reward)

Knowledge should be

1. *Learnable*—from low-level sensorimotor data
2. *Expressive*—able to express abstract, high-level facts as well as specific, low-level facts
3. *Useful*—for action and planning

"The problem of knowing"

Examples of stuff to know

- Twitching this muscle lifts that finger
- There is a wall behind me
- The toilet is down the hall on the left
- The shape of a teacup
- Knowing how to ride a bike
- Knowing how to call a taxi
- My keys are in my pocket
- There is an apple in the box
- There is a book on the table
- My car is red
- People usually have two feet
- The Eiffel tower is in Paris
- John has the flu

The Sensorimotor View

- In which an agent's knowledge is viewed as facts about the statistics of its sensorimotor data stream
- This point of view is interesting because
  - it is reductionist and demystifies world knowledge
  - it provides a clear way of thinking about semantics
  - it implies that knowledge can be verified and learned from data – "the knowledge is in the data"

Thus "Learning About Sensorimotor Data"

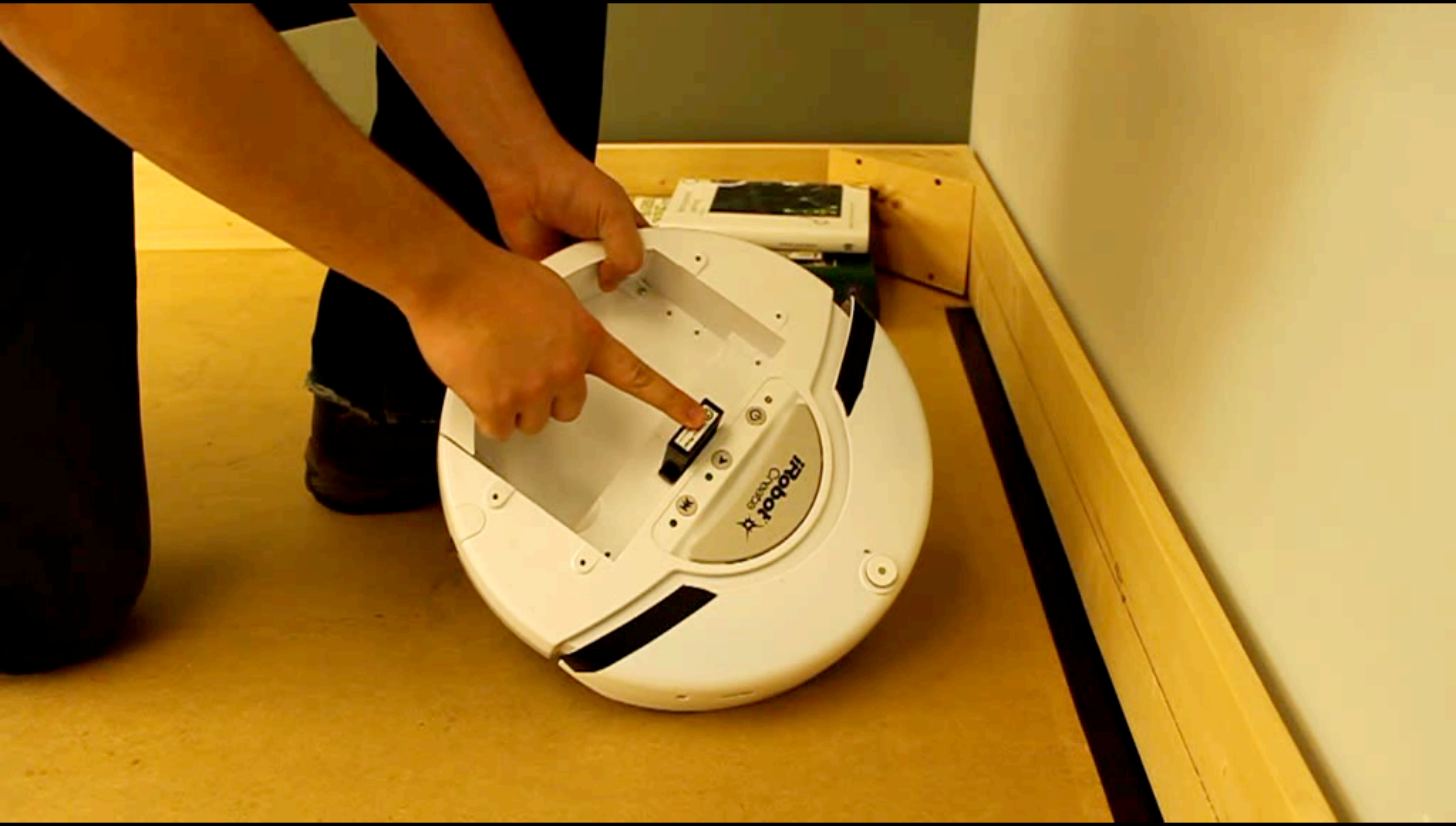
It's hard to implement the Sensorimotor View well

- Where "well" means such that it is
  - sound, stable, and efficient with function approximation
  - scalable to large numbers of predictions learned in parallel from the same experience
  - real time (online with many updates/second)
  - captures multi-step facts
- Achieving these modest goals is highly constraining

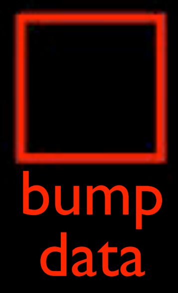
Thus a successful implementation can be informative

Robot experiments

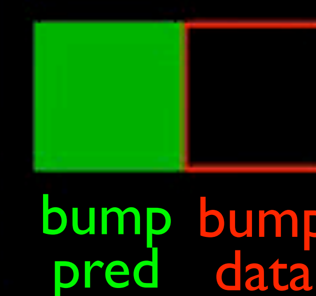
The iRobot Create



“Wall ahead” is a sensorimotor fact



Predicting: Will rolling forward soon result in a bump?



Predicting right and left bumps

pred data

left bump

both bump

right bump

Strategy

- To understand the world is to have *many predictions* about your sensorimotor data stream
- The predictions must be multi-step and policy contingent
  - because almost all interesting predictions are more-than-one-step and policy-contingent
- You must be able to learn from partial executions
  - because then you can learn about *many policies in parallel*
  - this will require *TD* and *off-policy* learning, and *FA*

Temporal-difference (TD) learning

- The core learning algorithm of online reinforcement learning
  - model-free dynamic programming
- Learning driven by TD errors (changes in prediction from one time to the next)
  - learning a guess from a guess

TD Learning in Engineering and Biology

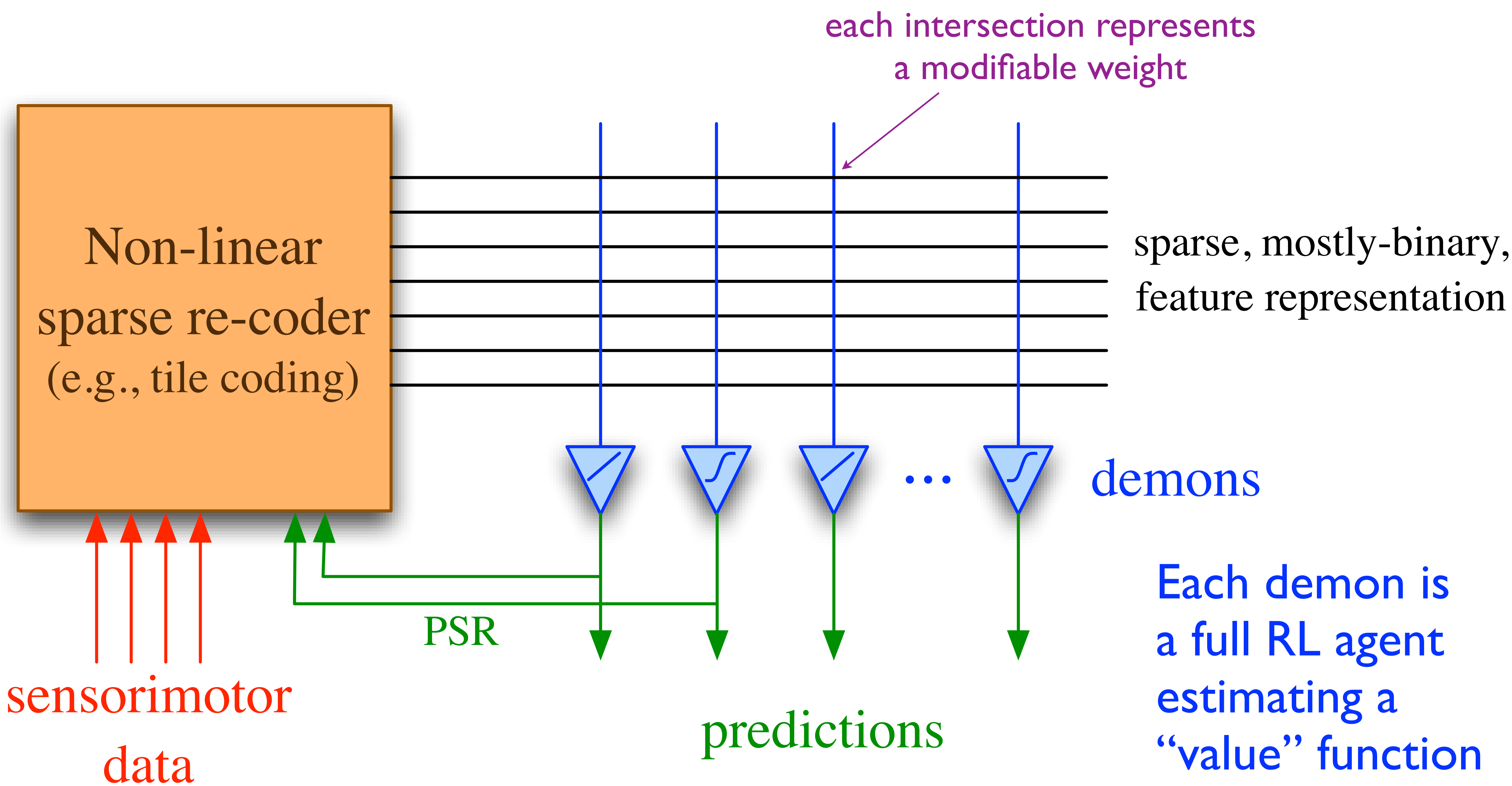
- TD algorithms are the standard model of reward-based learning in both
  - *engineering* (artificial intelligence and optimal control)
  - *biology* (neuroscience and psychology)
- TD algorithms have been *independently validated* in four distinct fields
- This is an unprecedented convergence

TD is in no way specific to reward

- TD is a real-time prediction-learning method
- suitable for predicting any signal, not just reward

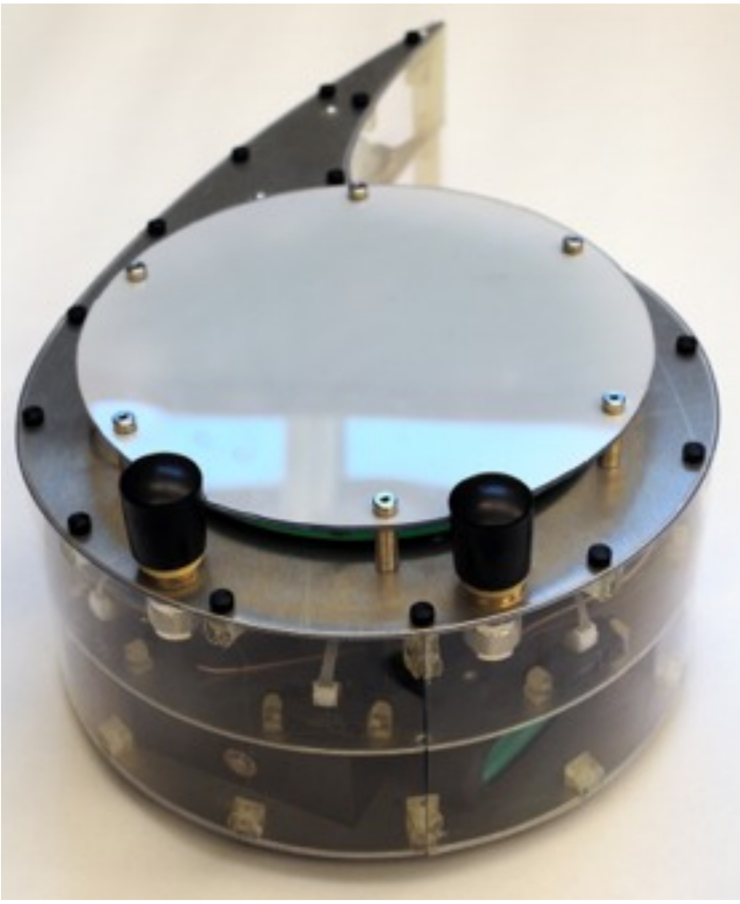
- it is a candidate for a universal prediction-learning algorithm

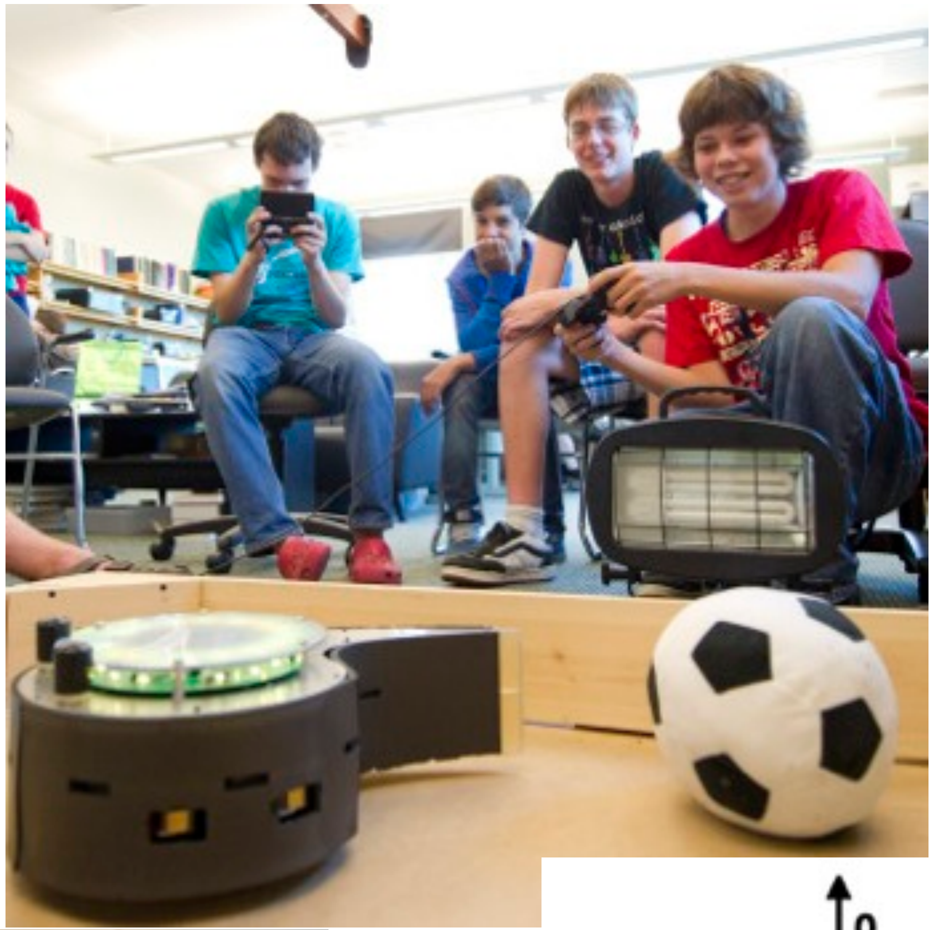
The Horde Architecture

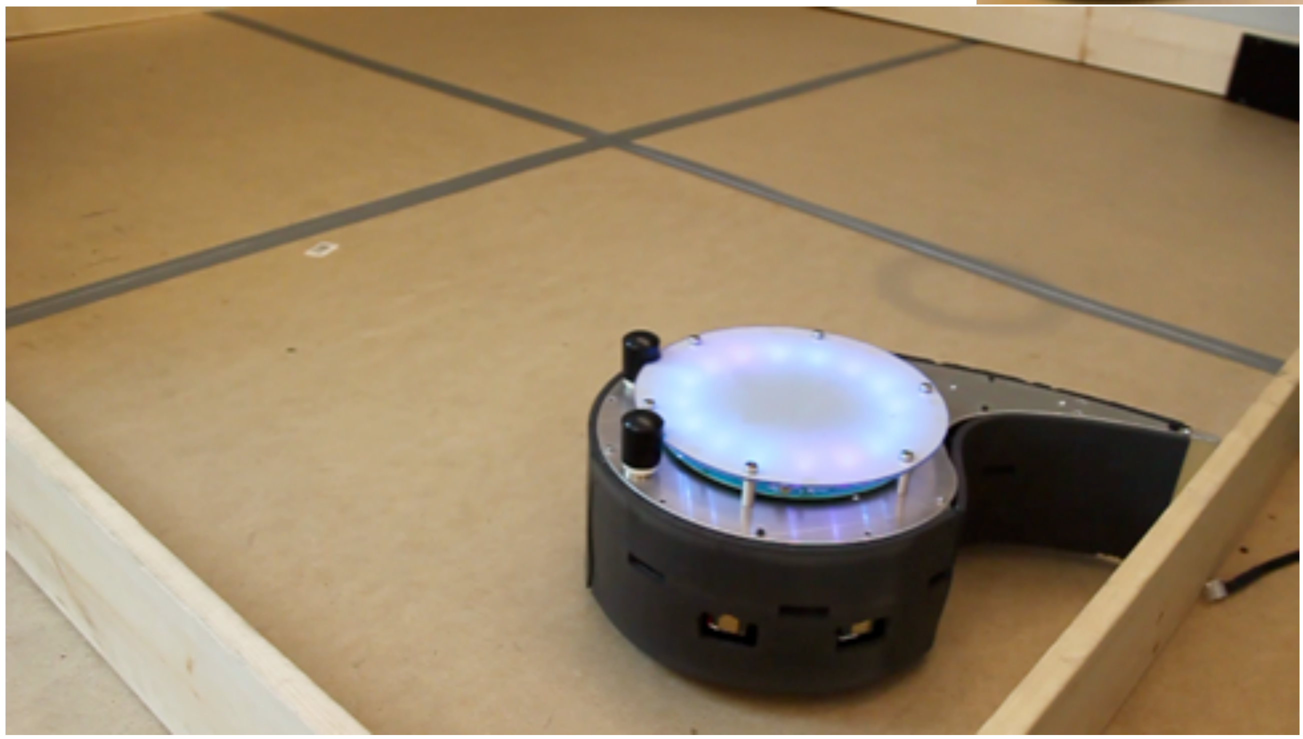


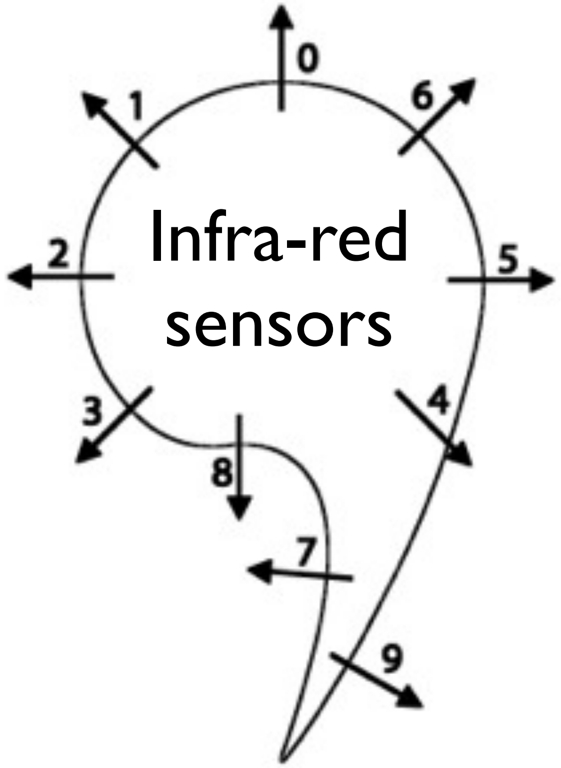
Each demon is
a full RL agent
estimating a
"value" function

The Critterbot

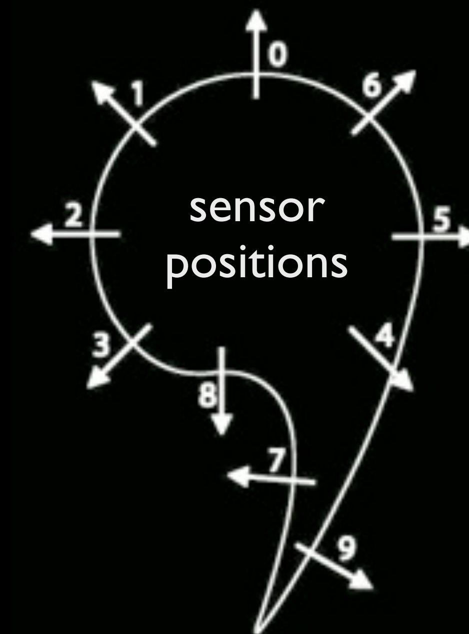


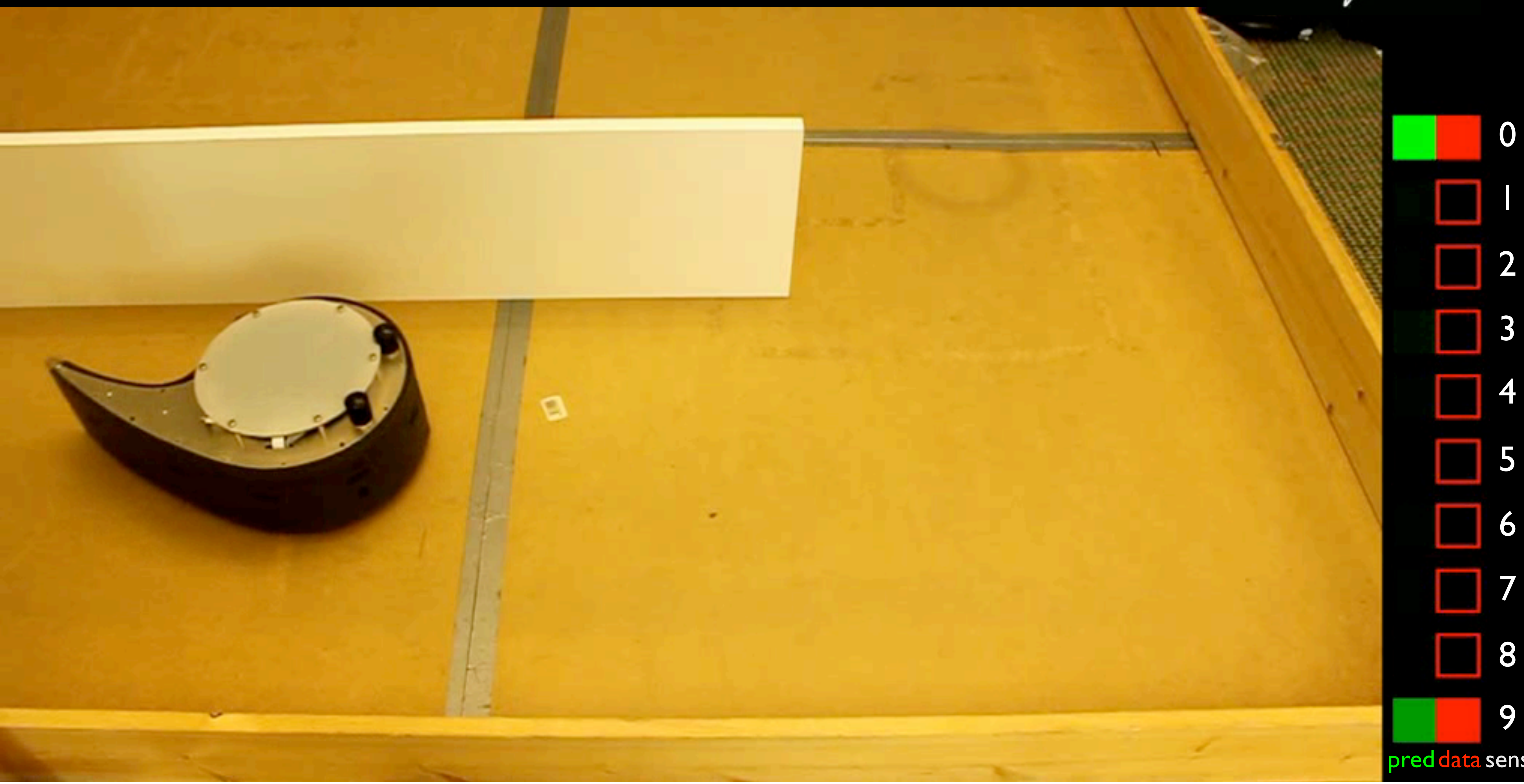




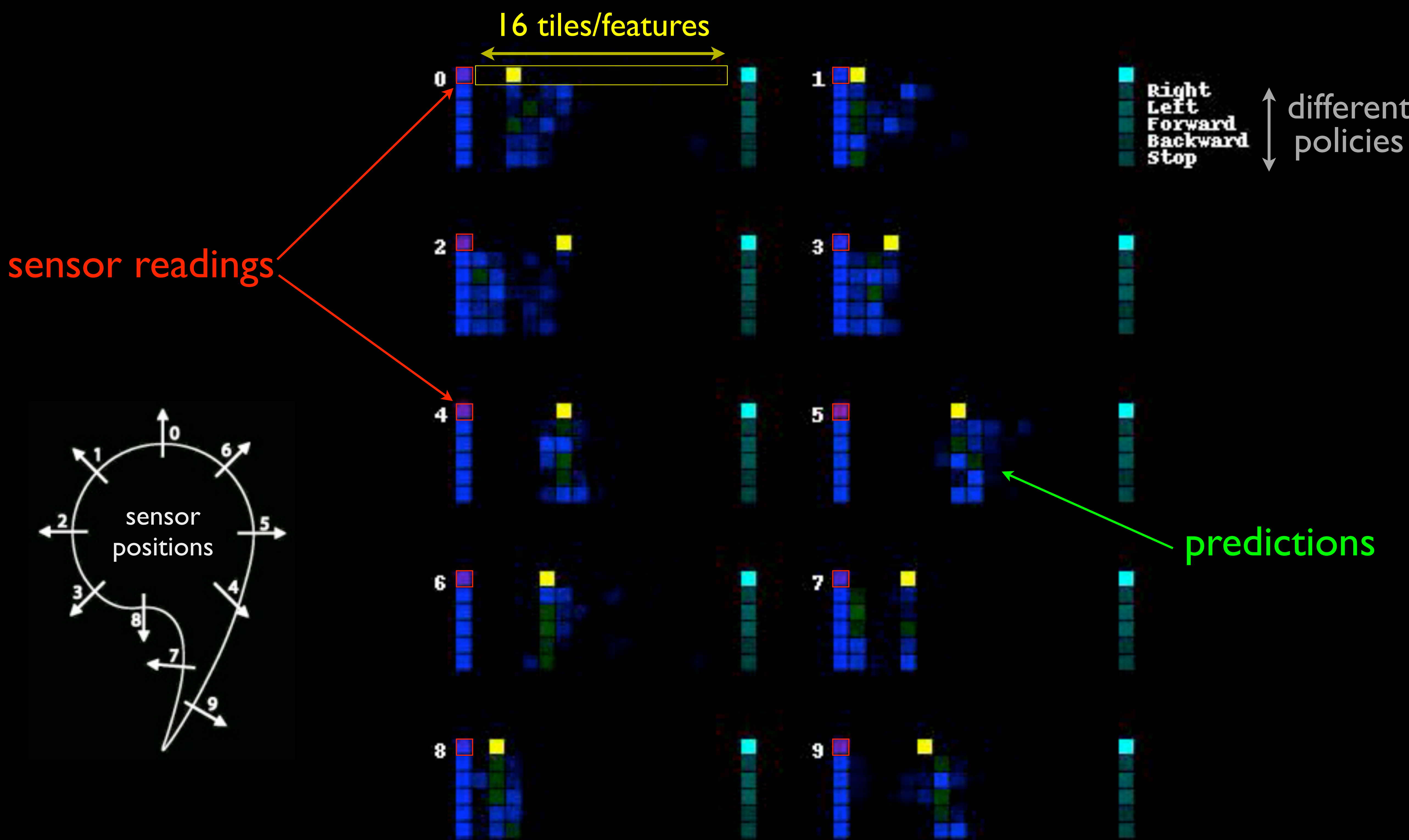


Infrared-sensor data and predictions



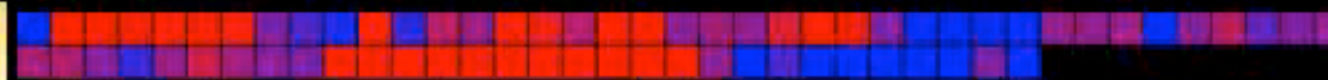


Scaling up: IR predictions for multiple tiles and policies



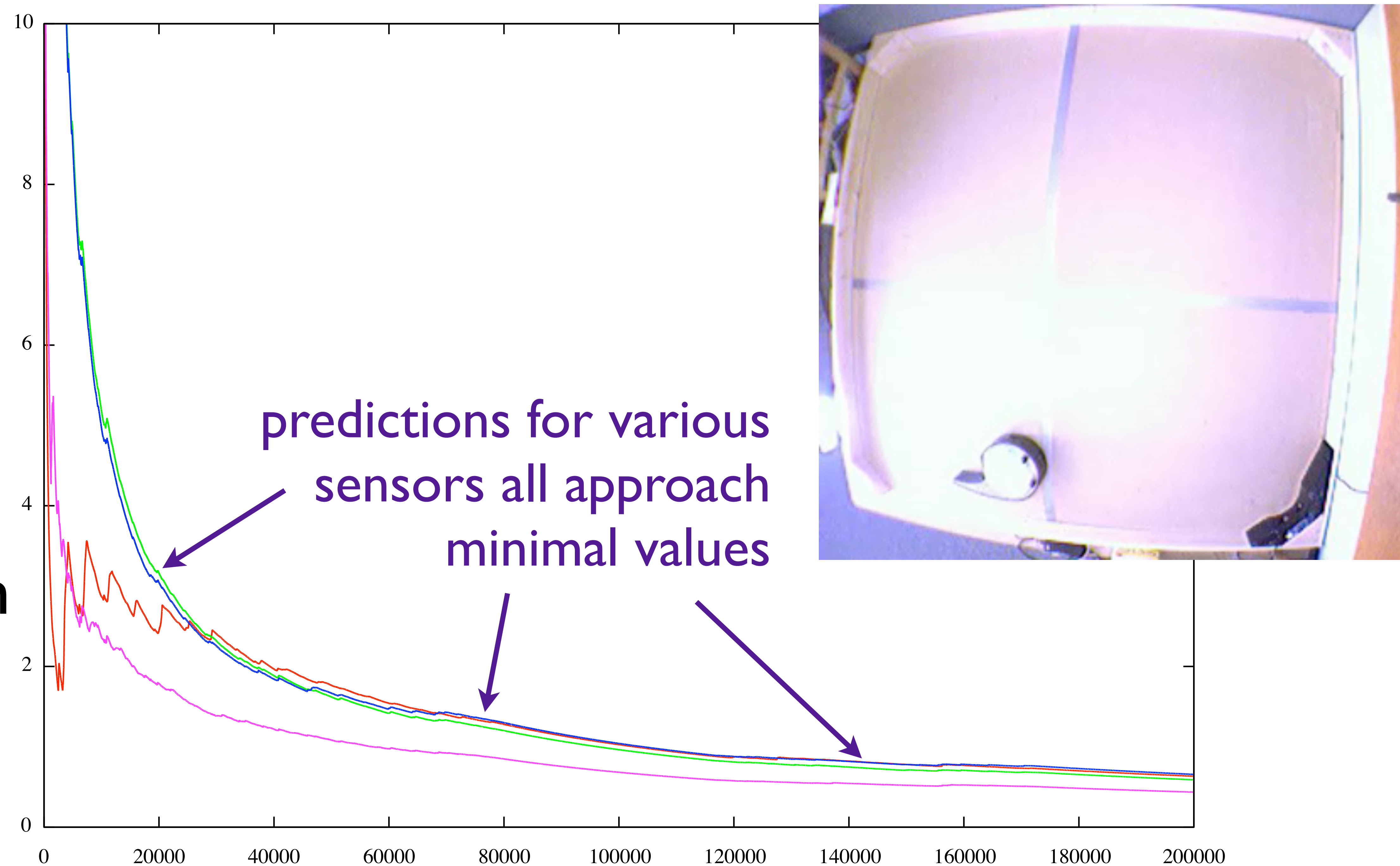
Scaling Up

continuous observation data x 69



Learning is fast enough

Mean-square error in prediction



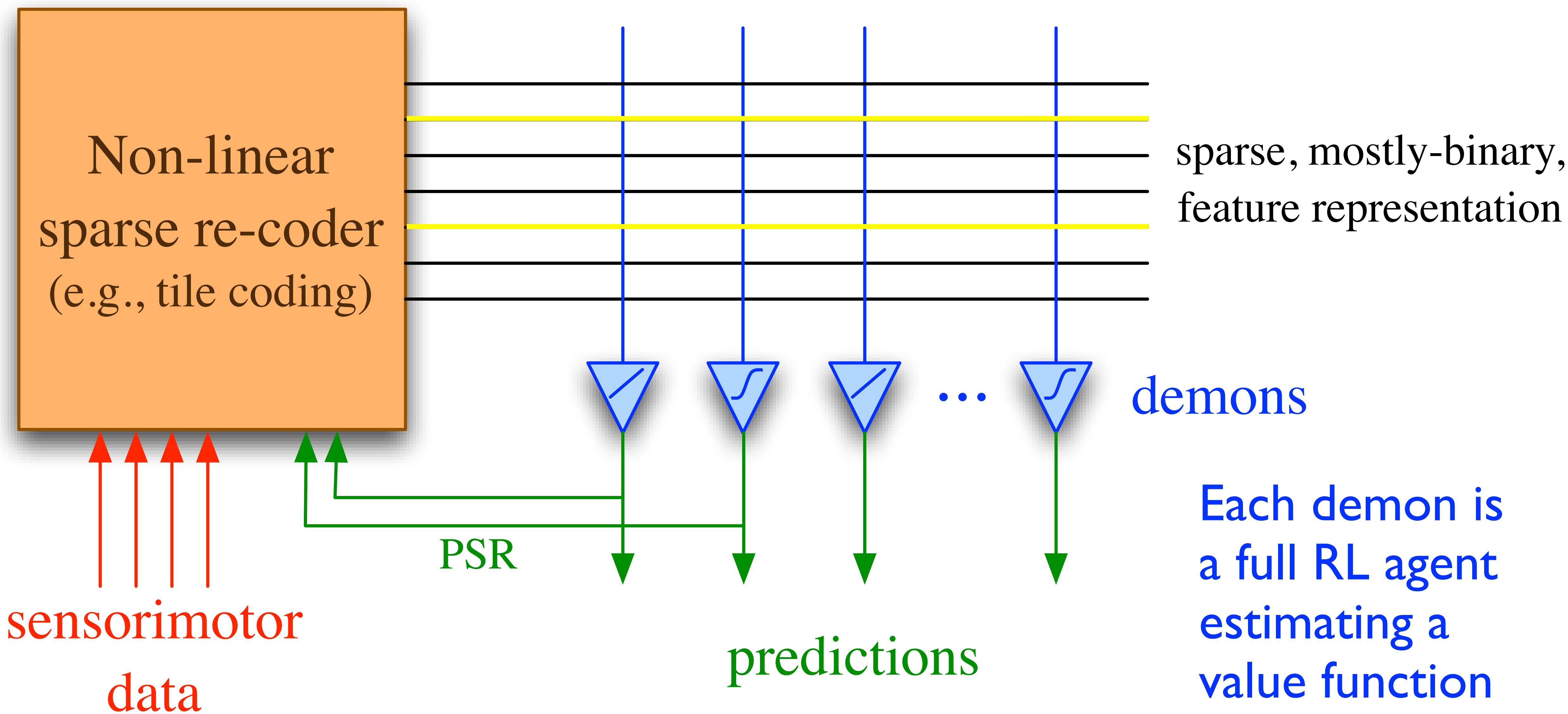
5 hours of training (100ms time steps)

Conclusions from robot experiments

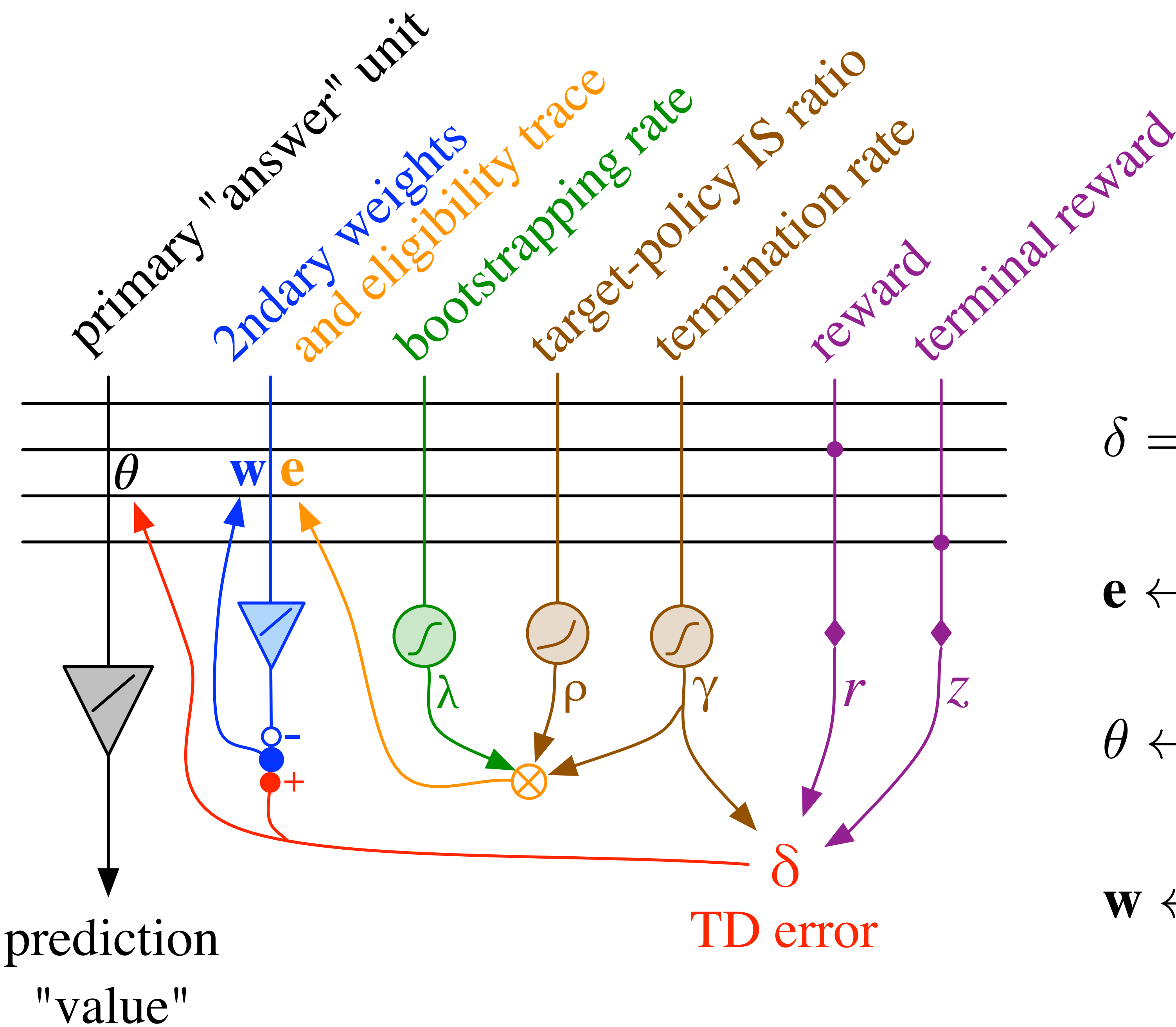
- Thousands of accurate multi-step predictions can be made and learned in real time at 10/second by linear TD algorithms
  - This could not have been done in any other way
- Model-free algorithms can learn fast enough to be useful
- *Real-time learning of sensorimotor knowledge is practical and scalable*

The Horde-of-demons architecture

The Horde Architecture



Inside a GTD(λ) Demon



$$\phi \xrightarrow[\rho,\gamma,\lambda]{r,z} \phi'$$

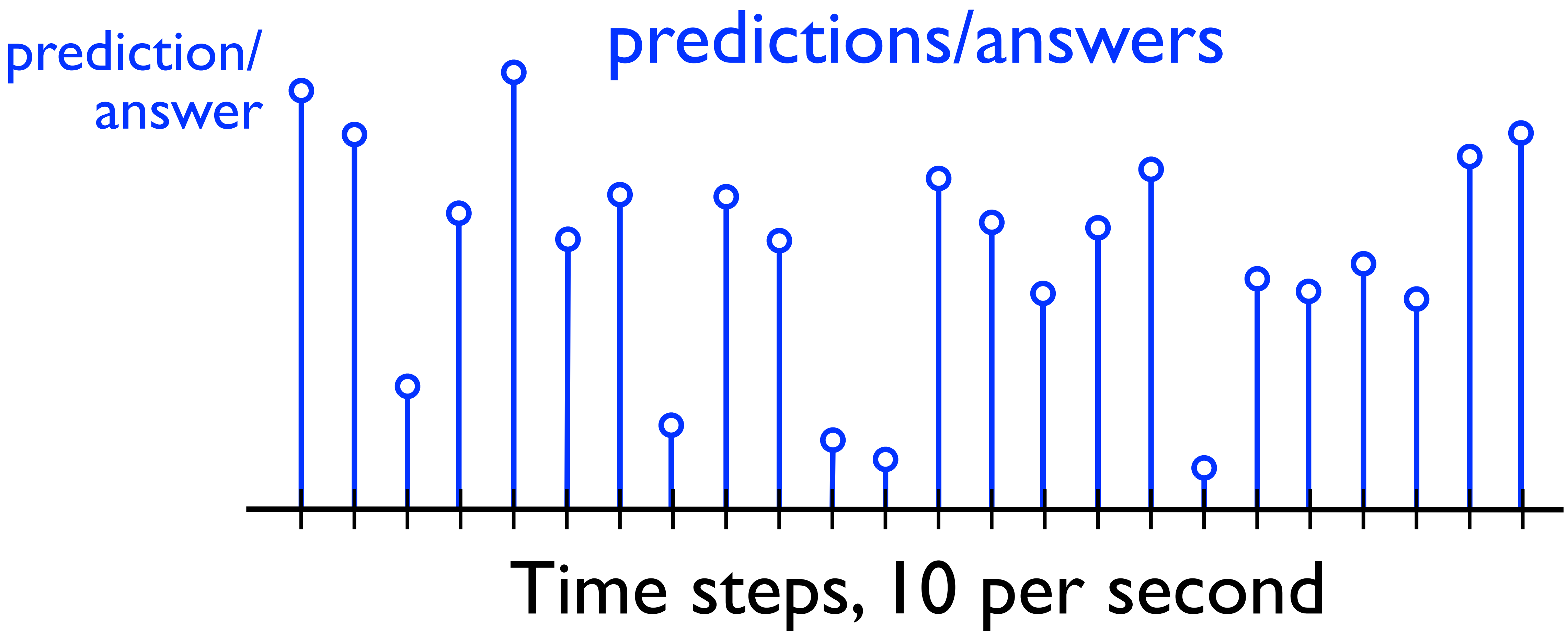
$$\delta = r + (1-\gamma)z + \gamma\theta^\top\phi' - \theta^\top\phi$$

$$\mathbf{e} \leftarrow \rho\left(\phi + \gamma\lambda\,\mathbf{e}\right)$$

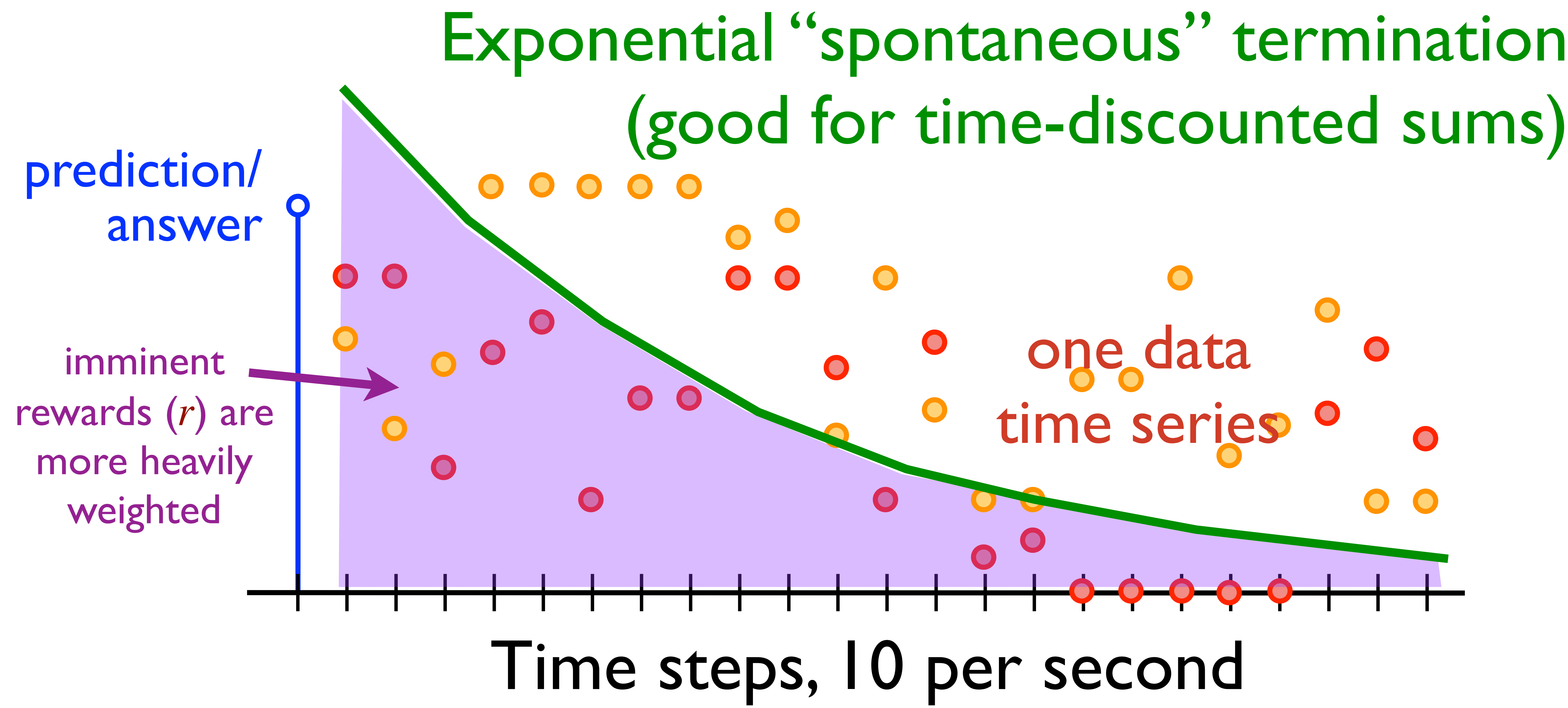
$$\theta \leftarrow \theta + \alpha\left[\delta\mathbf{e} - \gamma\left(1-\lambda\right)\left(\mathbf{w}^\top\mathbf{e}\right)\phi'\right]$$

$$\mathbf{w} \leftarrow \mathbf{w} + \beta\left[\delta\mathbf{e} - \left(\mathbf{w}^\top\phi\right)\phi\right]$$

General value functions as a language for multi-step predictive *questions*

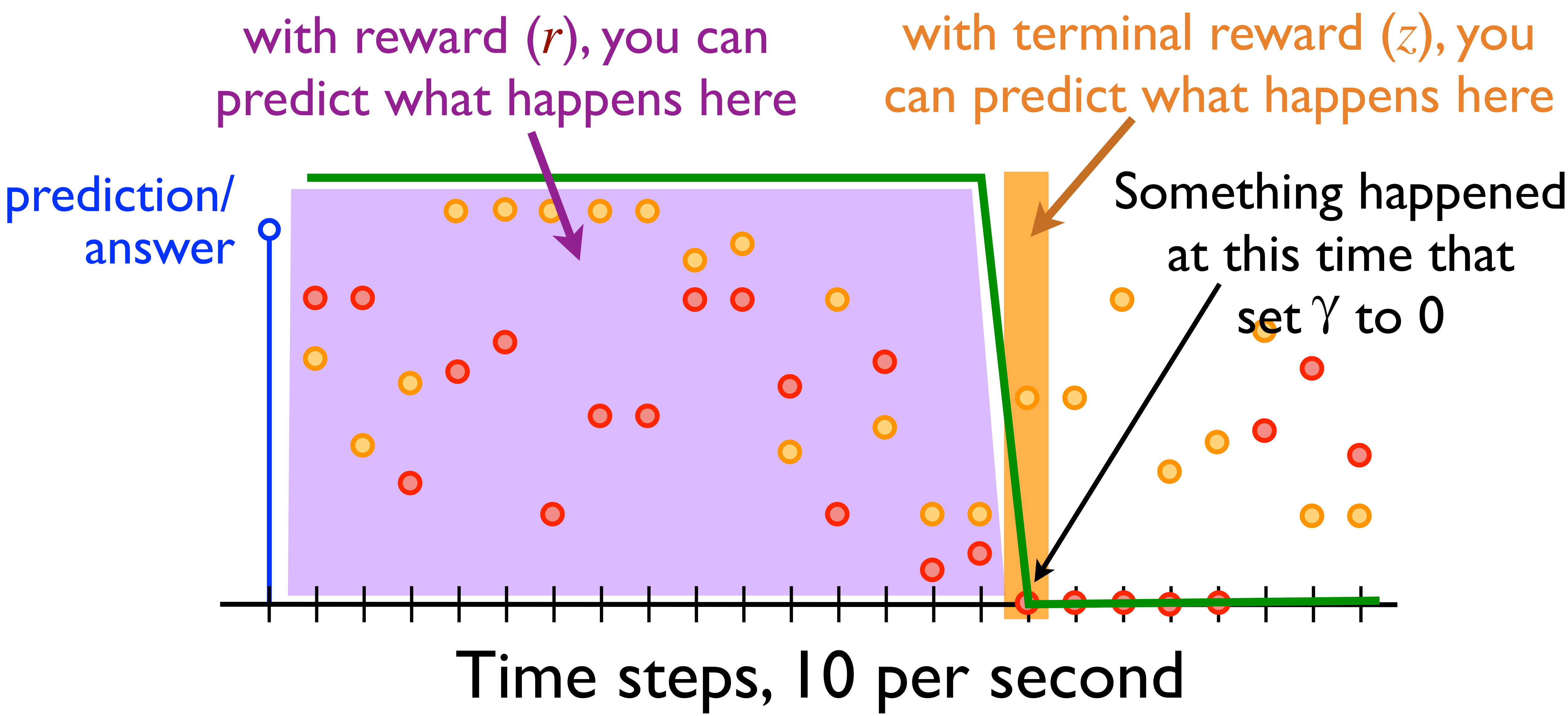


General value functions as a language for multi-step predictive <u>*questions*</u>



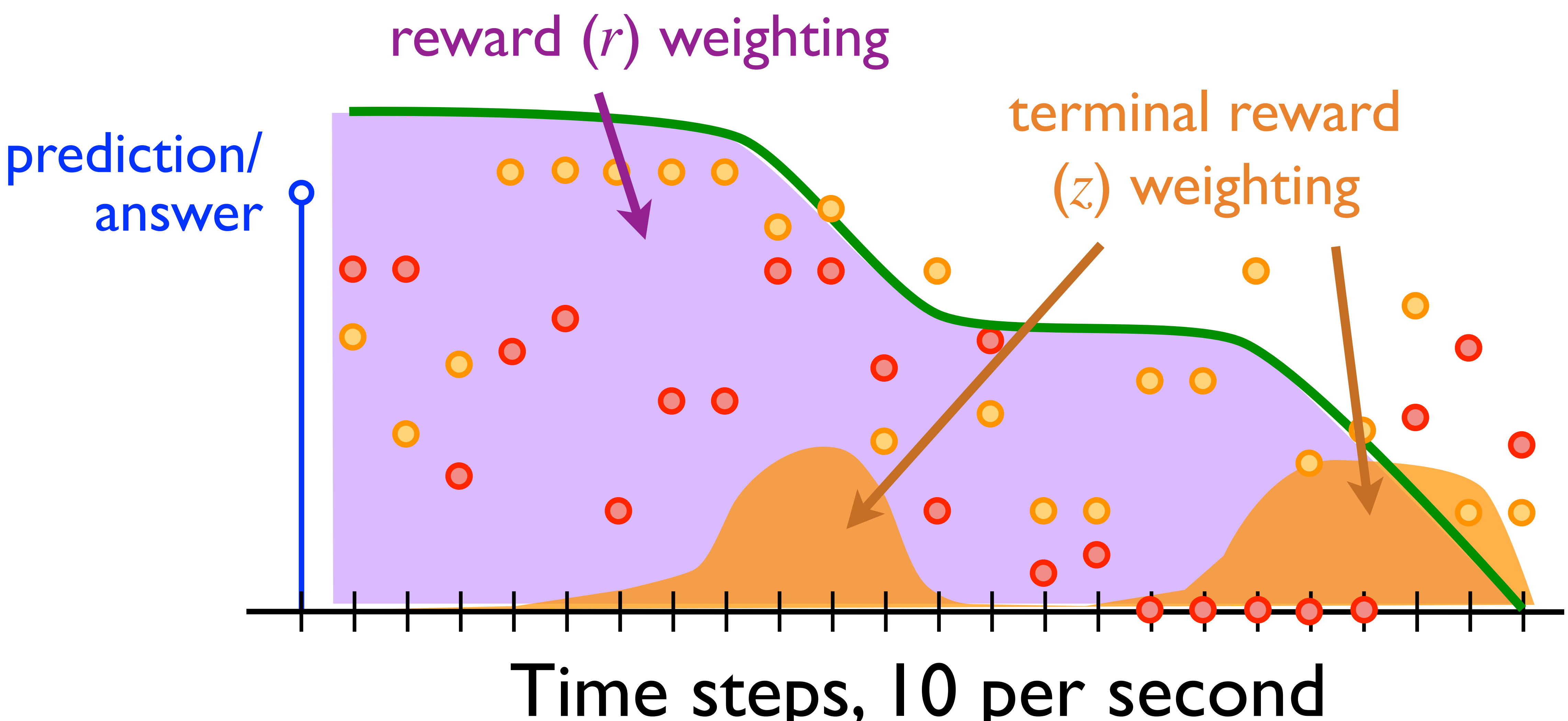
$$\theta^\top \phi(s) \;\approx\; V^{\pi,\gamma,r,z}(s) \;=\; \mathbb{E}[r(S_1) + \cdots + r(S_T) + z(S_T) \mid S_0 = s, T \sim \gamma, A_{0:T-1} \sim \pi]$$

General value functions as a language for multi-step predictive *questions*



$$\theta^\top \phi(s) \;\approx\; V^{\pi,\gamma,r,z}(s) \;=\; \mathbb{E}[r(S_1) + \cdots + r(S_T) + z(S_T) \mid S_0 = s, T \sim \gamma, A_{0:T-1} \sim \pi]$$

General value functions as a language for multi-step predictive *questions*



$$\theta^\top \phi(s) \approx V^{\pi,\gamma,r,z}(s) = \mathbb{E}[r(S_1) + \cdots + r(S_T) + z(S_T) \mid S_0 = s, T \sim \gamma, A_{0:T-1} \sim \pi]$$

General value functions—Fundamental or idiosyncratic?

- GVFs are a powerful rep'n language for the semantics of sensorimotor knowledge
  - GVFs seem powerful enough to encode all scientific knowledge (knowledge with experimentally testable predictions)
  - But we don't yet have extensive experience; some changes will probably be needed
- Crafted for efficient recursive computations
- Proven utility in control, planning, neuroscience

Remarks on gradient-TD algorithms

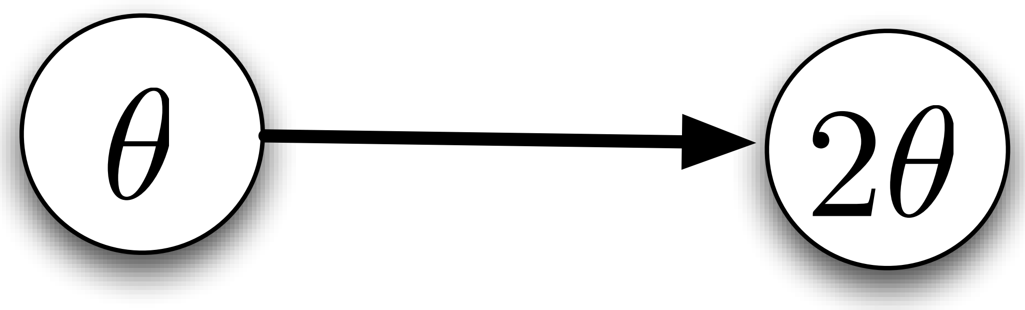
TD with FA

- TD with function approximation (FA) has historically been problematic:
  - for linear FA, there has been no TD algorithm with linear complexity that is sound under off-policy training
    - Q-learning diverges with linear FA
  - for non-linear FA, there has been no sound algorithm with constant per-step comp.
- The root problem is that there have been no true *gradient-descent* TD algorithms

TD and GD: Headlines

- Convention gradient-based TD algorithms are not true GD (because they ignore the effect on the new guess)
  - guaranteed convergent on-policy but not off-policy
- Baird's Residual Gradient and VAPS methods are GD in the wrong objective
  - converge to the wrong thing even in tabular case
- Precup's Importance Sampling methods too slow
  - too slow to benefit from parallel off-policy learning
- New true-GD methods (Maei, Szepesvari, Sutton et al.)

TD(0) can diverge: A simple example



$$
\begin{aligned}
\delta &= r + \gamma\theta^\top\phi' - \theta^\top\phi \\
&= 0 + 2\theta - \theta \\
&= \theta
\end{aligned}
$$

TD update: $\Delta\theta = \alpha\delta\phi$

$= \alpha\theta$ Diverges!

TD fixpoint: $\theta^* = 0$

TD with FA: Non-GD solutions?

- Linear least-squares methods: LSTD, LSPI
  - complexity is $O(n^2)$/step
- Gordon's averagers, Gaussian Processes
  - require storing examples—not scalable FA
- Policy-Gradient methods
  - RL not TD; don't learn multi-step facts
- Model-based methods
  - non-starter for the sensorimotor approach

The Gradient-TD Family

- GTD(λ) and GQ(λ), for learning GVF V and Q
- Developed by Maei, Szepesvari, Sutton, Precup, Bhatnagar, Silver, Wiewiora 2008-11
- Solve two open problems:
  - convergent linear-complexity off-policy TD learning
  - convergent non-linear TD
- True gradient-descent algorithms

Gradient-TD convergence theorem

The weights of Gradient TD methods follow the gradient of a projected-Bellman-error objective function in expected value:

step size

Bellman operator

vector of estimated values, one per state

$$E_D[\Delta\theta] = -\alpha \nabla_\theta \| V_\theta - \Pi T V_\theta \|_D^2$$

expectation under data distribution

gradient vector of partial derivatives

projection back into the space of representable functions

2-norm under the data distribution

which guarantees convergence to the TD fixpoint (under step-size conditions)

TD vs Gradient-TD

- TD error:

$$\delta_t = r_{t+1} + \gamma\theta_t^\top\phi_{t+1} - \theta_t^\top\phi_t$$

- Linear TD(0):

$$\theta_{t+1} = \theta_t + \alpha\delta_t\phi_t$$

- Importance sampling ratio:

$$\rho_t = \frac{\pi_{\text{target}}(s_t, a_t)}{\pi_{\text{behavior}}(s_t, a_t)}$$

- Off-policy linear GTD(0)

$$\theta_{t+1} = \theta_t + \alpha\rho_t\left[\delta_t\phi_t - \gamma\left(\mathbf{w}_t^\top\phi_t\right)\phi_{t+1}\right]$$

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \beta\left(\rho_t\delta_t - \mathbf{w}_t^\top\phi_t\right)\phi_t$$ 2nd weight vector

My message in one sentence

If it's important for your AI agent to know a lot,

and you take the sensorimotor approach,

then you are forced to multi-step predictions,

and to policy-contingent predictions,

which require TD (a new reason for TD!),

and, in fact, a new kind of gradient-TD,

if you want to proceed in a practical and scalable way (linear-complexity function approximation).

Further frontiers

- Learning directing action: Curiosity, intrinsic motivation
- Discovering features and questions
- Better gradient-TD algorithms
- Parallel learning by policy-gradient (actor-critic) methods?
- Models and planning
- It will be interesting just to keep scaling

Questions?

- What is the latest on Gradient-TD algs?
- Where do the questions come from?
- How do you know off-policy predictions are accurate?
- How can you be abstract when predictions are about low-level data?
- Can you give a simple example/intuition of why conventional TD methods diverge?
- Can you show us the simplest gradient-TD algorithm
- How can the predictions be used for action?
- How can GVFs form a world model for planning?

Thank you for your attention

- And thanks again to Adam White, Joseph Modayil, Thomas Degris, and the RLAI group