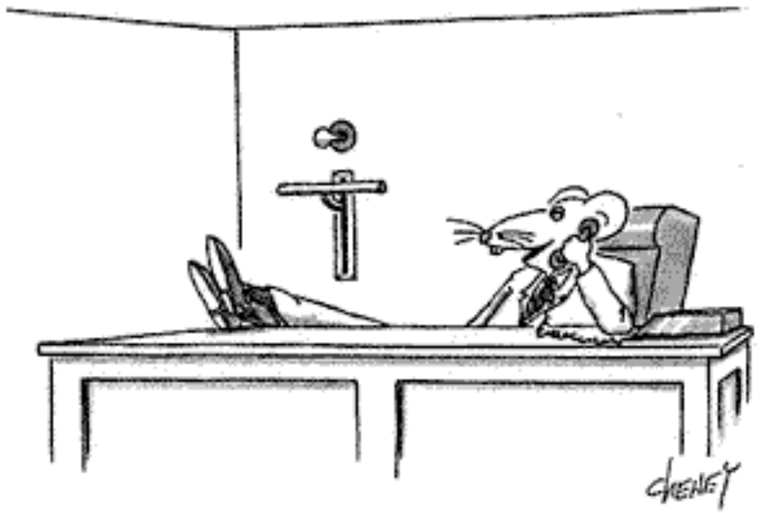
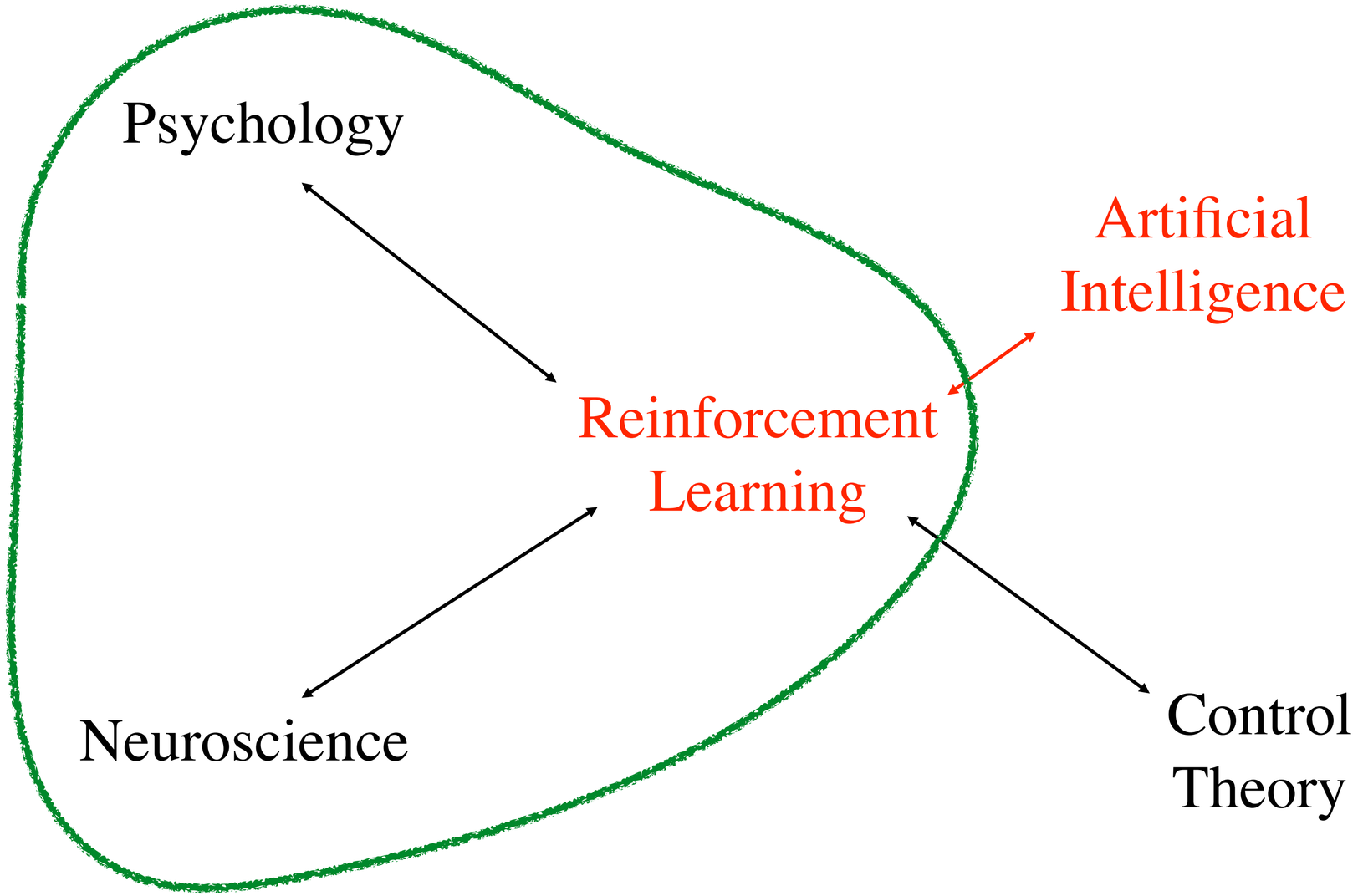
Reinforcement Learning in Psychology and Neuroscience



"Oh, not bad. The light comes on, I press the bar, they write me a check. How about you?"

with thanks to Rich Sutton &
Elliot Ludvig
University of Warwick


Psychology
Artificial
Intelligence
Reinforcement
Learning
Neuroscience
Control
Theory

Any information processing system can be understood at multiple “levels”

- The Computational Theory Level
  - *What* is being computed?
  - *Why* are these the right things to compute?
- Representation and Algorithm Level
  - *How* are these things computed?
- Implementation Level
  - How is this implemented physically?

David Marr, 1972

Psychology has identified two primitive kinds of learning

- *Classical* Conditioning
- *Operant* Conditioning (a.k.a. Instrumental learning)
- Computational theory:
  - *Classical* = Prediction
    - What is going to happen?
  - *Operant* = Control
    - What to do to maximize reward?

Classical Conditioning

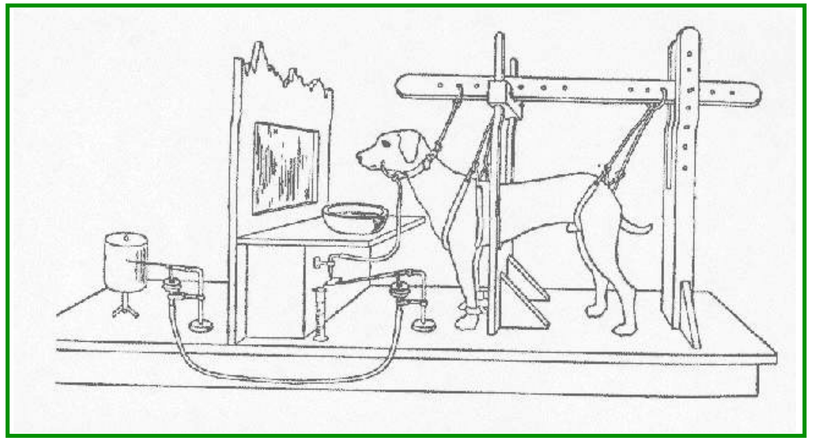
Classical Conditioning as Prediction Learning

- Classical Conditioning is the process of learning to predict the world around you

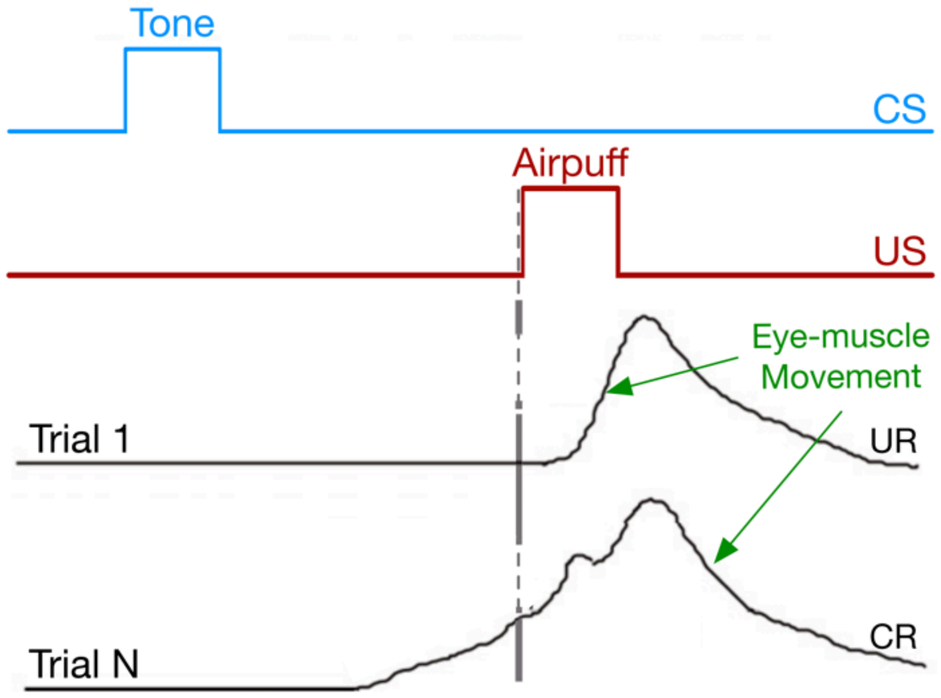
  - Classical Conditioning concerns (typically) the subset of these predictions to which there is a hard-wired response

Pavlov (1901)

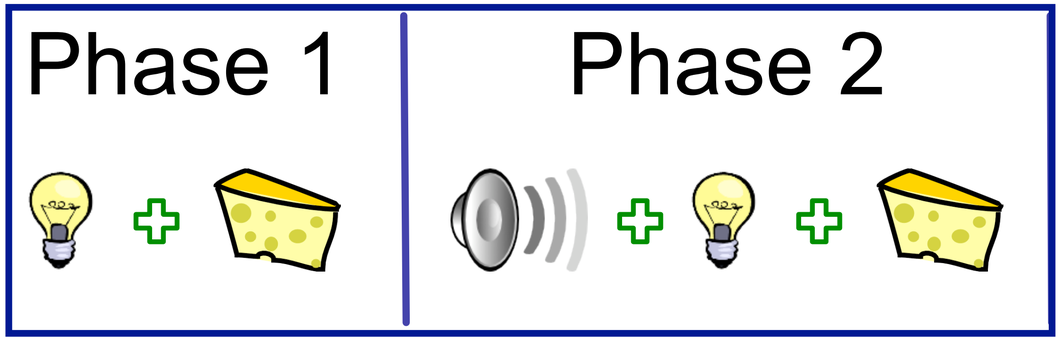
- Russian physiologist
- Interested in how learning happened in the brain
- Conditional and Unconditional Stimuli



Is it really predictions?



Blocking



Light comes to cause salivation

Will sound come to cause salivation?

Learning about the sound in Phase 2 does not occur because it is *blocked* by the association formed in Phase 1

Rescorla-Wagner Model (1972)

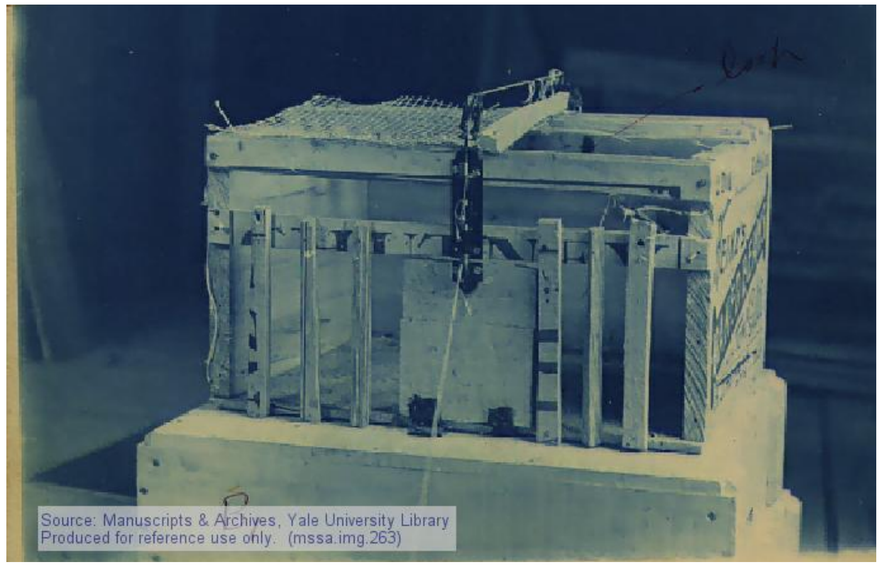
- Computational model of conditioning
  - Widely cited and used
- Learning as violation of expectations
  - As in linear supervised learning (LMS, p2)
  - TD learning is a real-time extension of this same idea

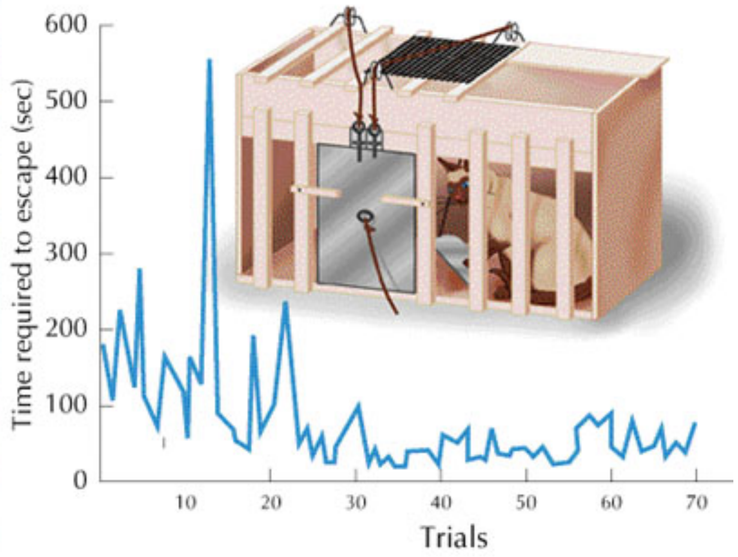
Operant Learning

- The natural learning process directly analogous to reinforcement learning
- Control! What response to make when?

Thorndike's Puzzle Box

(1910)

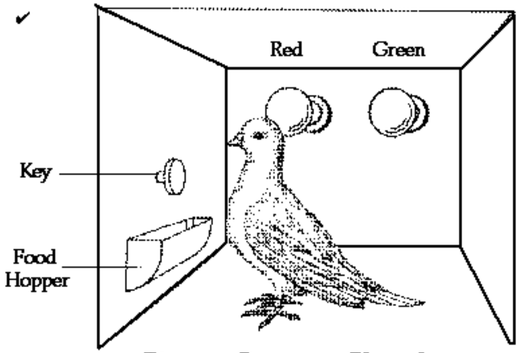




Law of Effect

- “Of several responses made to the same situation, those which are accompanied by or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur...” - Thorndike (1911), p. 244

Operant Chambers



Pigeon Operant Chamber

Complex Cognition

Any information processing system can be understood at multiple “levels”

- The Computational Theory Level
  - *What* is being computed?
  - *Why* are these the right things to compute?
- Representation and Algorithm Level
  - *How* are these things computed?
- Implementation Level
  - How is this implemented physically?

David Marr, 1972

The Basic TD Model

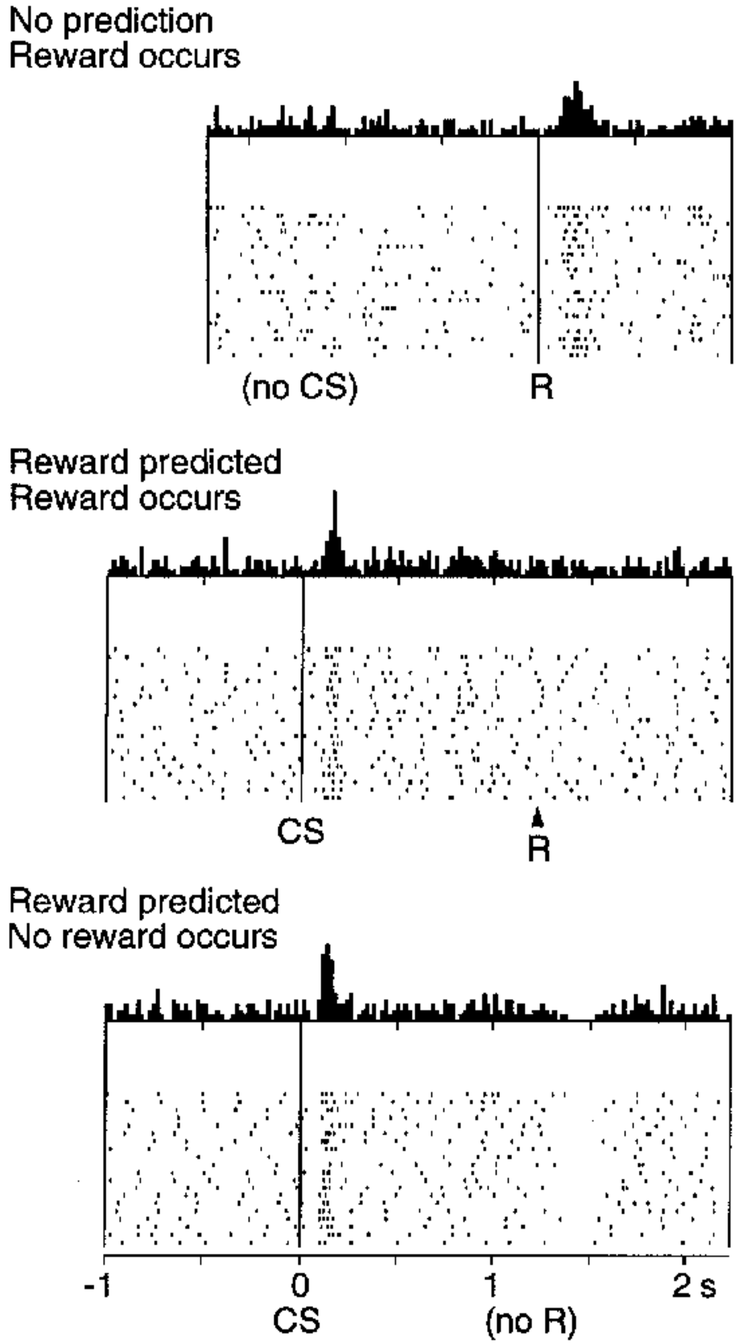
- Learn to predict discounted sum of upcoming reward through TD with linear function approximation

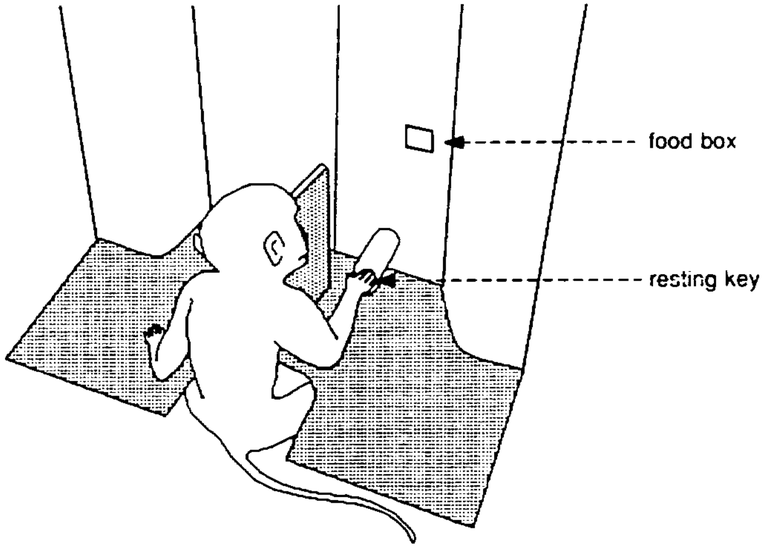
- The TD error is calculated as:

$$\delta_t \doteq R_{t+1} + \gamma\, \hat{v}(S_{t+1}, \boldsymbol{\theta}) - \hat{v}(S_t, \boldsymbol{\theta})$$

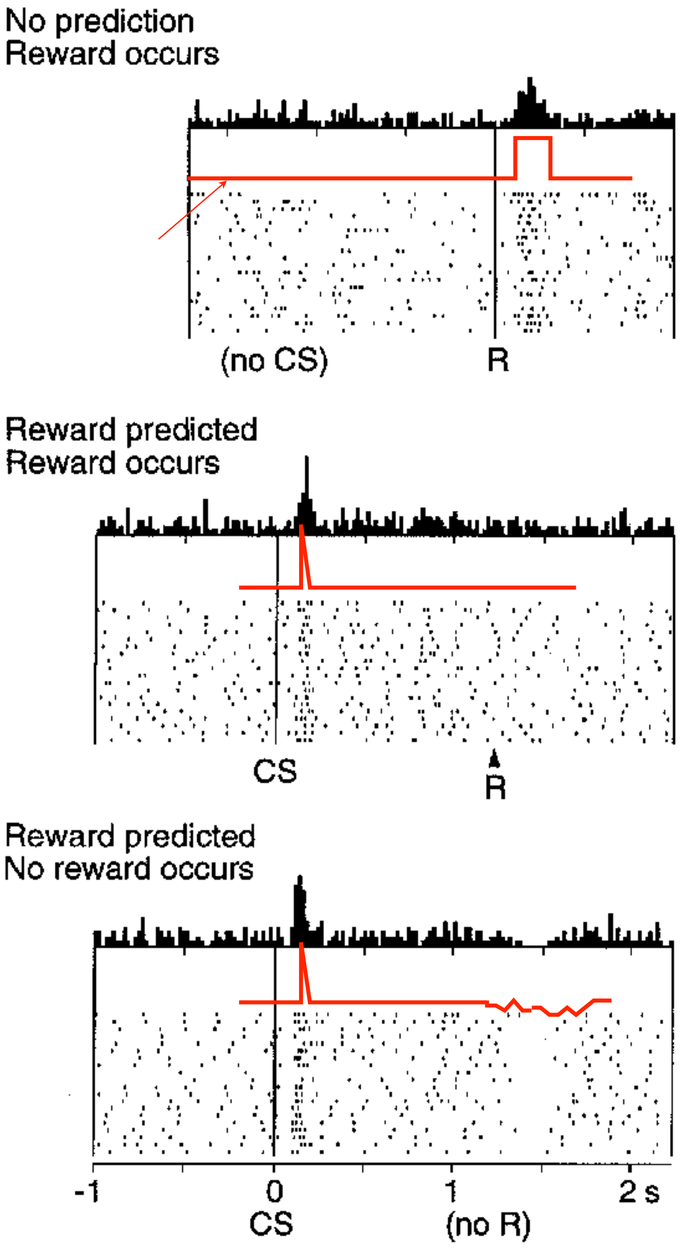
- Change in baseline dopamine responding = reward prediction or TD error

Dopamine neurons signal the error/change in prediction of reward



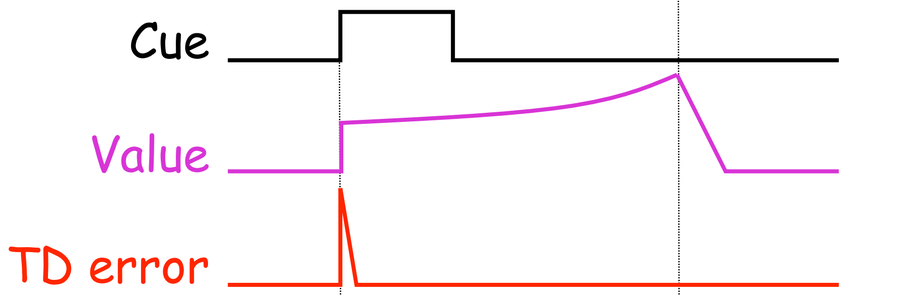


Wolfram Schultz, et al.

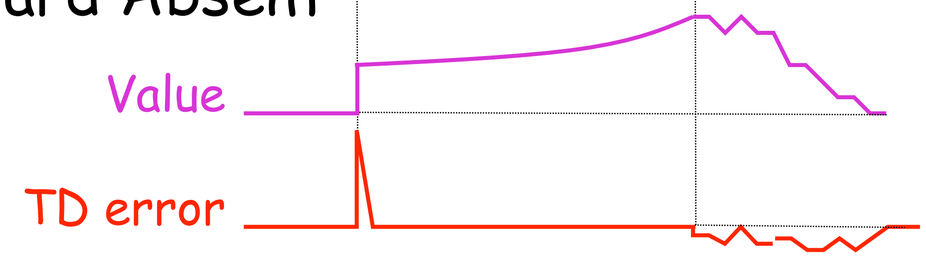


Reward Unexpected

Reward Expected



Reward Absent



$$\delta_t = R_{t+1} + \gamma \hat{v}_{t+1} - \hat{v}_t$$

The theory that *Dopamine = TD error*
is the *most important interaction ever*
between AI and neuroscience

What have you learned about in this course (without buzzwords)?

- "Decision-making over time to achieve a long-term goal"
  - includes learning and planning
  - makes plain why value functions are so important
  - makes plain why so many fields care about these algorithms
    - AI
    - Control theory
    - Psychology and Neuroscience
    - Operations Research
    - Economics
  - all involve decision, goals, and time…
    - the essence of… mind

AI & knowledge

- One objective of an AI agent is to know a lot
  - people know that the sun will rise, how to walk, how the desk will feel if I touch it, …
- My objective is to build learning systems that can know a lot about their world
  - learning for the sake of acquiring knowledge
- Determining how to represent and acquire knowledge is a classic problem in AI research

Predictive knowledge

- Knowledge represented as a question about the future & its answer
- A question about an outcome, conditioned on some way of behaving
- Predictive knowledge is personal & related to an agent's specific abilities, environment, experiences

Examples of predictive knowledge

- How likely am I to bump into the wall over the next few seconds, if I were to walk forward?
- How long will it take me to get to the store? (declarative knowledge)
- How do I get to the printer? (procedural knowledge)

Benefits of the predictive approach

- The knowledge can be updated from interaction with the world:
  - make a prediction, act, observe the outcome & update
- Knowledge can be maintained independent of people
- Learning can be scaled with *computation* and *data*
- Less developed than the public approach

Challenges for the predictive approach

- Building up from (potentially) low-level predictions to abstract knowledge
- Loss of human understanding
- Demonstrating how the knowledge can be used

outline

- Summary of prior developments in acquiring knowledge in the form of predictions
- General value functions
- Multi-scale nexting on a robot

Predictive knowledge and AI

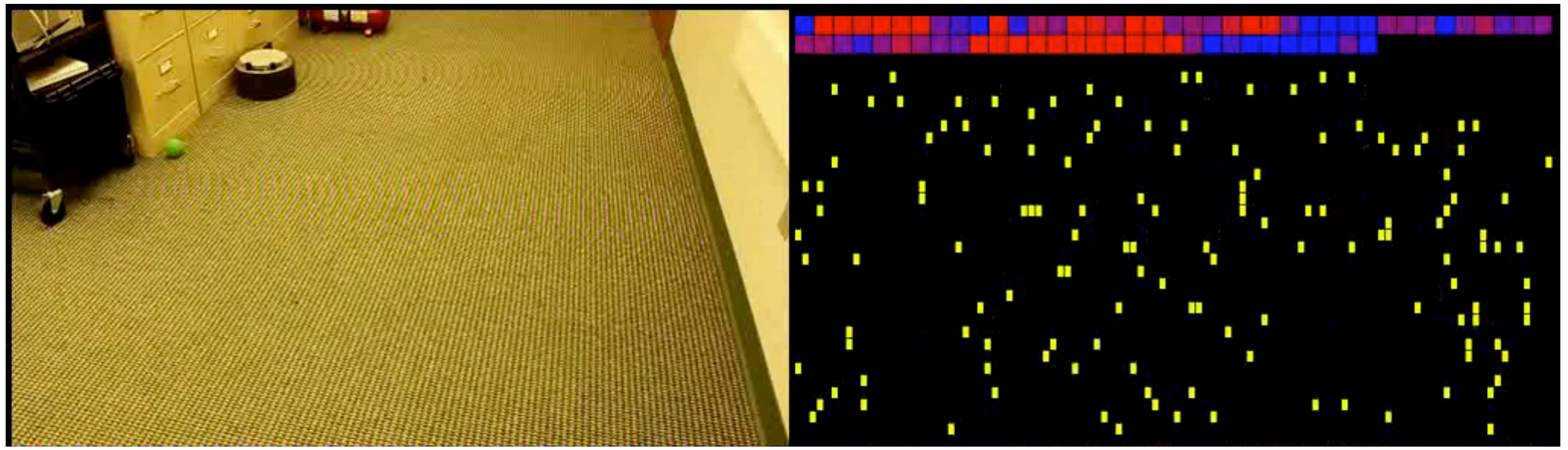
- The ability to predict and anticipate has been proposed as a key part of intelligence
- Several early efforts were inspired by the developmental theories of Piaget: stages of sensorimotor development
  - Drescher (1991): context—action—>result schemas. Schemas make action-conditional predictions. Small program

Our setting

behavior policy:

$\mu : \mathcal{S} \times \mathcal{A} \to [0, 1], \quad \mu(s, a) = \mathbf{Pr}(a|s)$

feature vector: $\mathbf{x} \in \mathbb{R}^n$



Conventional value functions

$$v(s) = \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... | S_t = s, A_{t:\infty} \sim \mu]$$

- A value function predicts future reward
- **If** we can represent predictive knowledge with a value function,
- **Then** we can learn them with value function learning algorithms
  - can learn online, are computationally frugal (potentially highly **scalable**)
  - can use parametric function approximation, have non-linear variants, and train off-policy (potentially broadly **applicable**)

A GVF is a prediction about the future

- A GVF predicts something about the future of the data stream
- $v(s;\pi,\gamma,z) = \mathbb{E}_\pi[G_t|S_t = s],\ G_t$ is called the **target**
- G_t is computed from future **cumulant** $Z_{t:\infty} \in \mathbb{R}$and **termination** signals $\gamma_{t:\infty} \in [0,1]$
- a conventional value function predicts future reward, & GVFs predicts future cumulant. The distinction is in how we define the Z , γ, and π

To define a GVF we need to define

- a cumulant signal Z
- a termination signal γ
- a target policy π

Cumulant: Z_t

- Like reward in conventional value function learning
  - not necessarily something to maximise or minimise
- Z_t is any signal of interest that can be observed by the agent
  - e.g., battery level, a constant (e.g. 1), current humidity
- We can have many cumulants, one for each GVF
- Can be useful to think of $Z_t = z(S_t)$

Termination signal: γ_t

- We can think of $1\text{-}\gamma_t$ as the prob. of terminating upon entry into S_t:
  - full or hard termination: $\gamma = 0$,
  - no possibility of termination: $\gamma = 1$
  - stochastic or soft termination: $\gamma = 0.9$

$$G_t = \mathbb{E}[Z_{t+1} + \gamma_{t+1} Z_{t+2} + \gamma_{t+1}\gamma_{t+2} Z_{t+3} + ...]$$

- We can have many γ, each representing a hypothetical termination
- Unlike conventional termination, hypothetical termination does not cause interruption of transition dynamics (future $\boldsymbol{x}_t$ and A_t)

Example terminations

- While driving home: define the following hypothetical terminations
  - at the half-way point home: $\gamma_t=0$
  - a constant ($\gamma_t=0.9$); γ_t determines the time scale of the prediction
  - mixtures of these

Target policy: π

- Each GVF and its question about future cumulant is conditioned on future actions being selected according to the target policy π
- π might be equal to the policy that is generating the actions (on-policy)
- or it might be different (off-policy)
- E.g., μ = driving home & π = taking a detour to the park. An hypothetical course of action

A state GVF

- Is a prediction from the current state (e.g., now or t) of future cumulants, conditioned on actions selected according to target policy, and terminations occurring according to γ

$$v(s;\pi,\gamma,z) \stackrel{\text{def}}{=} \mathbb{E}_\pi\left[G_t | S_t = s\right]$$

$$G_t \stackrel{\text{def}}{=} \sum_{k=0}^{\infty} \left(\Pi_{j=1}^{k} \gamma_{t+j}\right) Z_{t+k+1}.$$

A state-action GVF

- Is a prediction from the current state of future Z, if we first select action a, then select future actions according to π, and terminations occurring according to γ:

$$q(s, a; \pi, \gamma, z) \stackrel{\text{def}}{=} \mathbb{E}_\pi \left[G_t | S_t = s, A_t = a \right]$$

Examples of GVFs

- How long will it take me to get to the store?
  - $v(s; \pi,\gamma,z)$: $Z_t = 1$, π = `goto_store`, $\gamma_t = 0$ on observation of the distinct sensory pattern corresponding to “the store”, else **1.0**
- How imminent is the onset of bumping into the wall, over the next ~20 steps, if I were to walk forward?
  - $v(s; \pi,\gamma,z)$: $Z_t = \text{bump}_t$, $\gamma_t = 0$ on bump, 0.95 otherwise, π = `walk_forward`
- What will the front distance reading be on bump if I drive forward?
  - $v(s; \pi,\gamma,z)$: $Z_t = (1 - \gamma_t)\ \text{IR\_dist}_t$, $\gamma_t = 0$ on bump, 1.0 otherwise, π = `drive_forward`

Predictive knowledge =
{predictive question specified by a GVF} +
{the learned approximate answer}

Approximate GVFs

- As is common in RL we consider approximations of the value function
- Assume the state of the world is summarised by a finite length feature vector, $|\boldsymbol{x}| << |S|$
- The prediction on the current step is the inner product between the features and a modifiable weight vector:

$$v(s;\pi,\gamma,x) \approx \hat{v}(s_t) = \mathbf{w_t}^\top \mathbf{x_t}$$

- $\boldsymbol{w}$ can be learned from samples of $< \mathbf{x}_t, A_t, Z_{t+1}, \gamma_{t+1}, \mathbf{x}_{t+1} >$ with value function learning methods from RL

Attributes of our GVF approach

- Can represent a large class of predictive questions: all knowledge representable as a prediction about the outcome of a procedure/experiment

- GVFs like value functions can be learned by temporal difference learning methods:

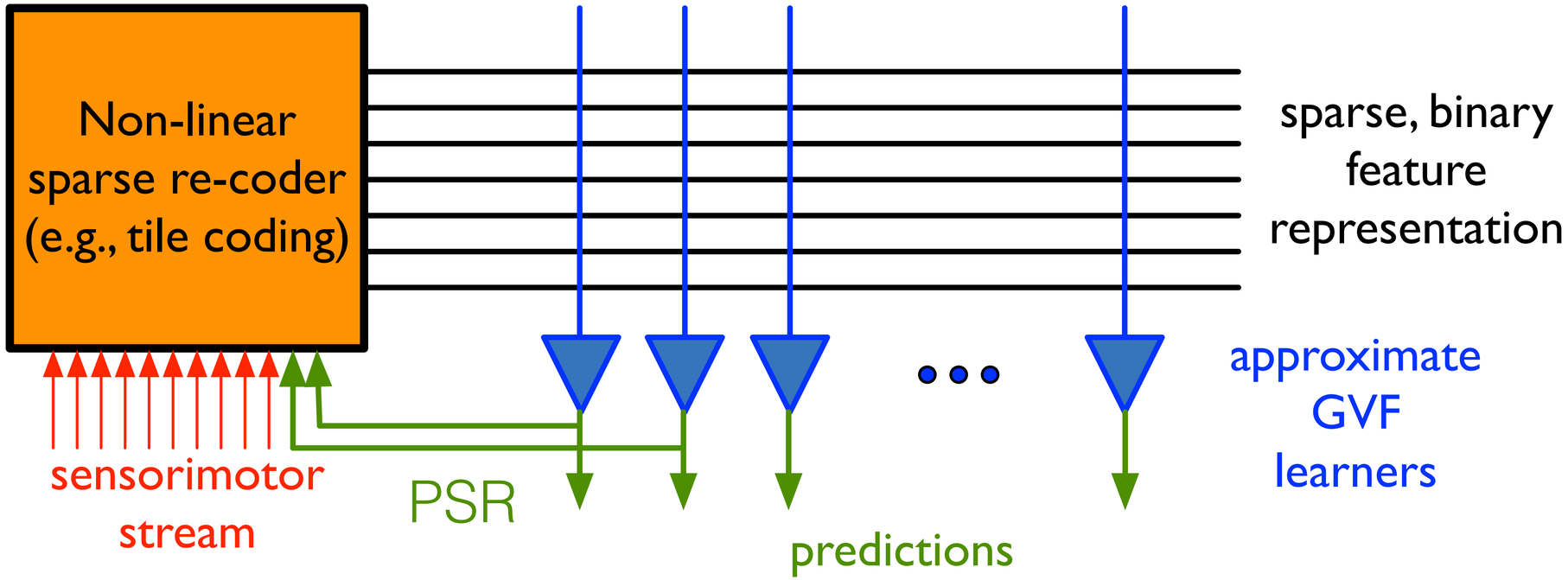
  - learnable online, with function approximation (continuous inputs), with linear computation and storage (in the size of $\boldsymbol{x}$)

- GVFs can be learned off-policy with off-policy TD methods like GTD, TO-GTD, GQ, PTD, PTD, emphatic TD

Attributes of our GVF approach (2)

- GVFs support predictive features (like a PSR)
- GVFs facilitate temporal abstraction, predicting the consequences of policies (like TD-nets with options)
- GVFs support compositional prediction, using the learned answer from one approximate GVF to specify a different GVF (like TD-nets)
- Each GVF forms a partial & incomplete predictive model of the world (like a prediction-profile model)
- GVFs can be learned massively in parallel …

A Horde of Demons



What kinds of predictions can be represented as GVFs, and can they be learned efficiently and accurately?

Continual prediction

- Psychologists conjecture that humans and many animals continually make large numbers of short-term predictions about their sensory input
  - when you here a melody you might predict the next note or beat
  - as you track a object flying though the air, or handle an object you continually make and confirm multiple predictions
  - As you read you predict at letter, word, and sentence
- Animals anticipate upcoming shock : heightened breathing & then paw retraction.
- Animals learn predictive relationships between stimuli

Nexting

- Gilbert (2006) called this “nexting”
- Nexting predictions are simple, short-term (but multi-step) predictions that are specific to the agent’s environment, capabilities, and experiences

A nexting robot

- Humans and animals seem to predict regularities in their experience & we want our robot to do the same
- We want our robot to notice little changes in its environment
- We want our robot to be is notice when unusual things happen, to be aware of its environment in a meaningful way
- We want our robot to next

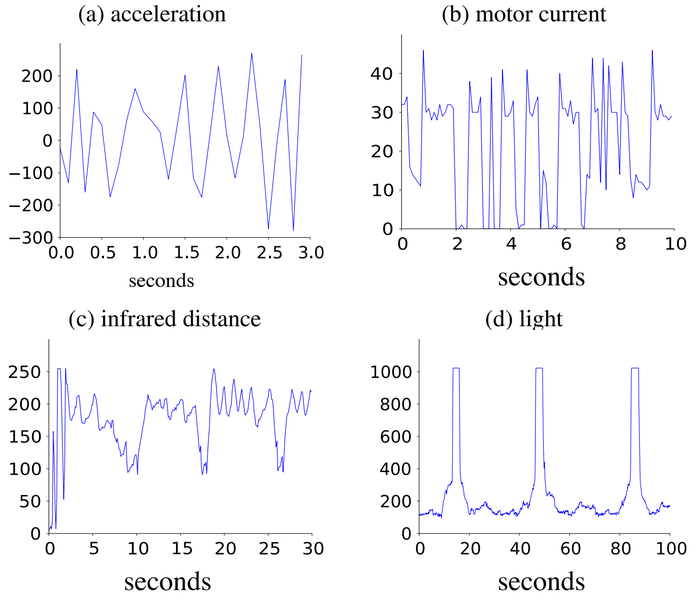
- Can we make and update **many** nexting predictions, in real-time and on a robot?

- Is a single fixed feature representation adequate to learn **accurate** predictions?

- Is learning with model free TD methods fast enough?

The Critterbot

- ~50 dimen. sensor vector, updated ~100 ms
- Similar to the regularities in a person's experience, our robot has predictable regularities at time scales ranging from tenths of seconds to tens of seconds



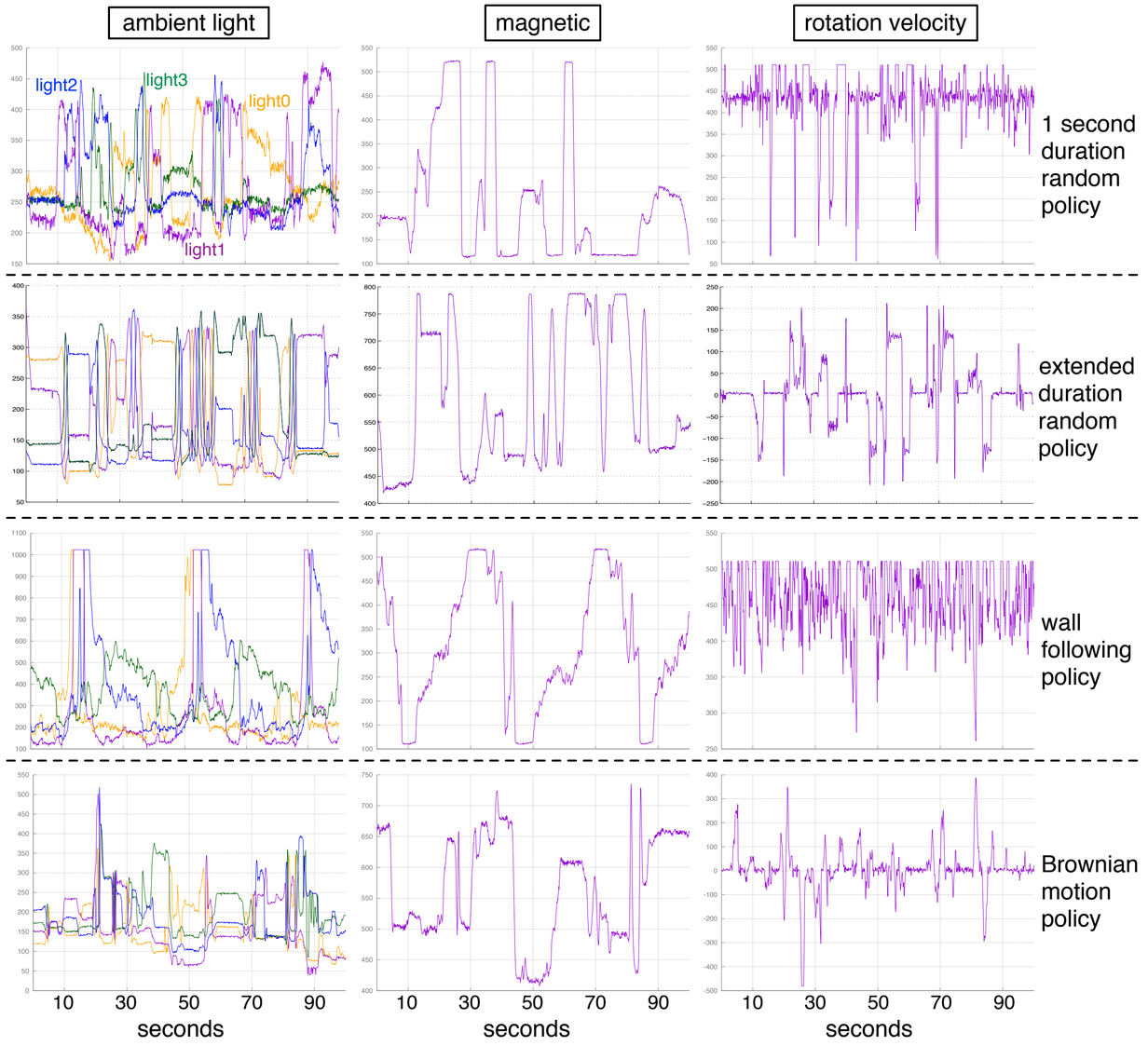
Objective: implement a form of nexting on a robot

- Demonstrate the scalability & applicability of our GVF approach
- Simulate an important psychological phenomenon

Robot Nexting

- our robot has many sensors, lets predict each sensor at several different (short) timescales, learning online while the robot interacts with the world
- & predict the many feature components ($\boldsymbol{x}$) as well
- => learn a collection of GVFs

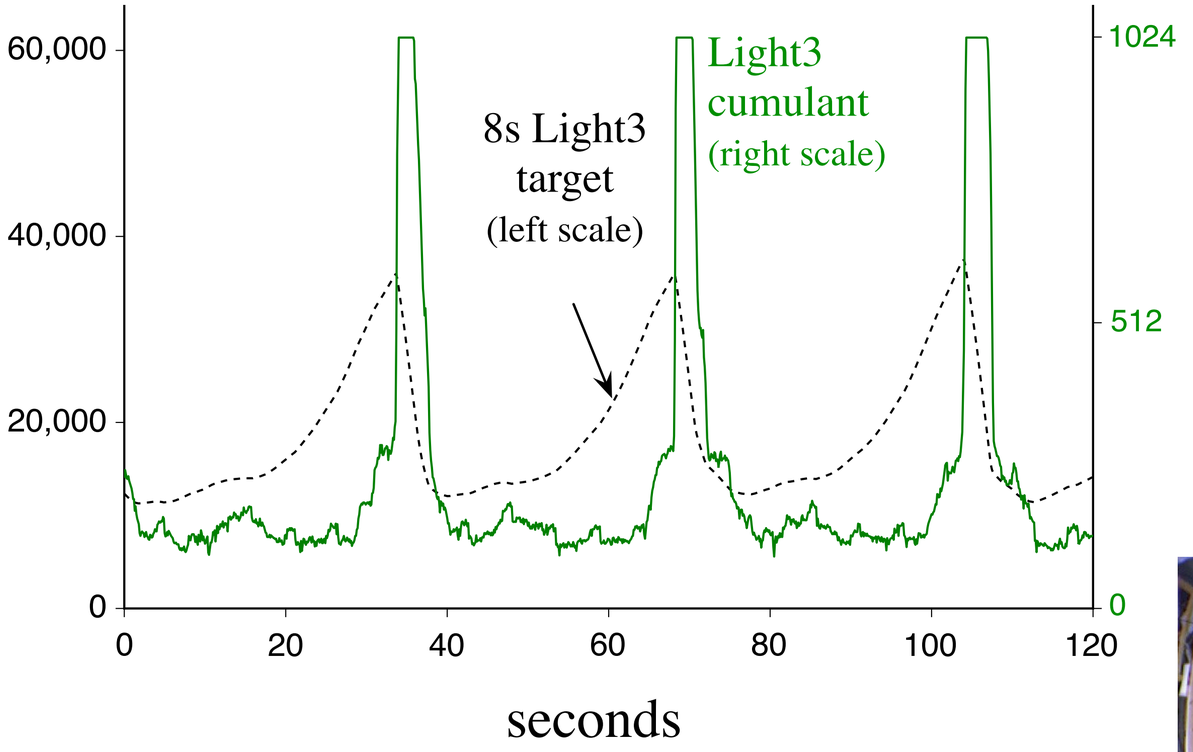
What happens next depends on the policy

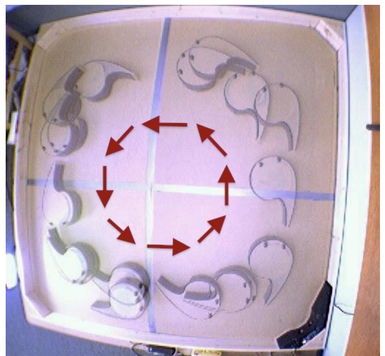


Nexting state GVFs

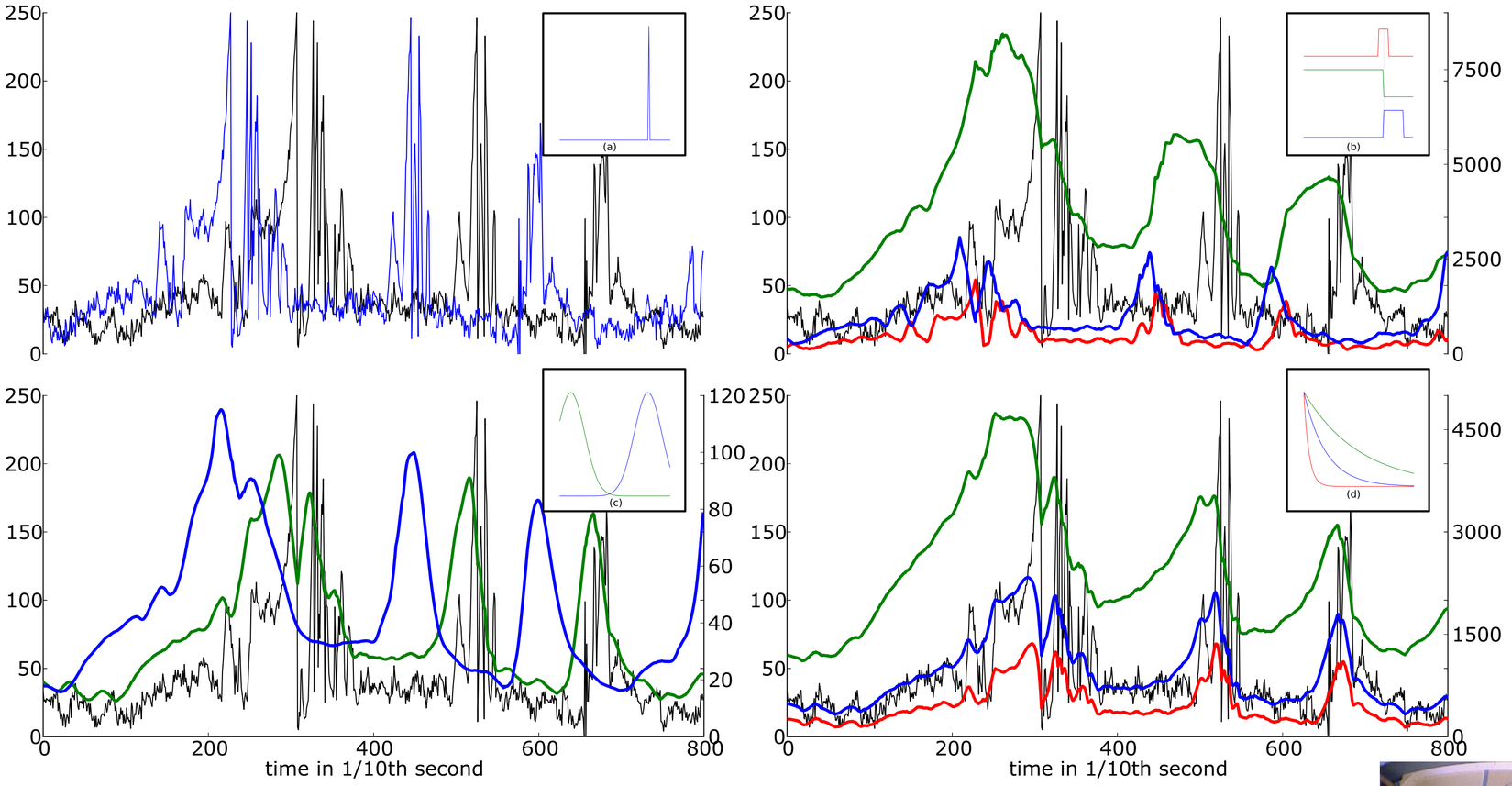
- Z = one of the robot's sensors (e.g. $Z_t = \text{light3}_t$) or feature component (e.g., $Z_t = \mathbf{x}_t[2445]$)
- γ = one of $\{0.0, 0.8, 0.95, 0.9875\}$ corresponding to $\{.1, .5, 2, \& \ 8\}$ second prediction time-scales
- μ = hand-coded **wall following** policy, $\pi = \mu$

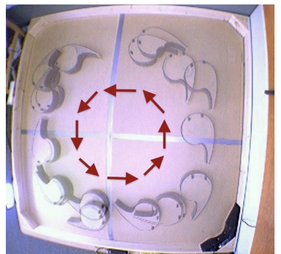
A light Nexting prediction





Other possible IR-light prediction targets





Learning and making many Nexting predictions in parallel with TD(λ)

- Each demon i, uses TD(λ) to update the weights for each approximate GVF:

$$\mathbf{w}_{t+1}^{(i)} = \mathbf{w}_t^{(i)} + \alpha \left(Z_{t+1}^{(i)} + \gamma^{(i)} \mathbf{x}_{t+1}^\top \mathbf{w}_t^{(i)} - \mathbf{x}_t^\top \mathbf{w}_t^{(i)} \right) \mathbf{e}_t,$$

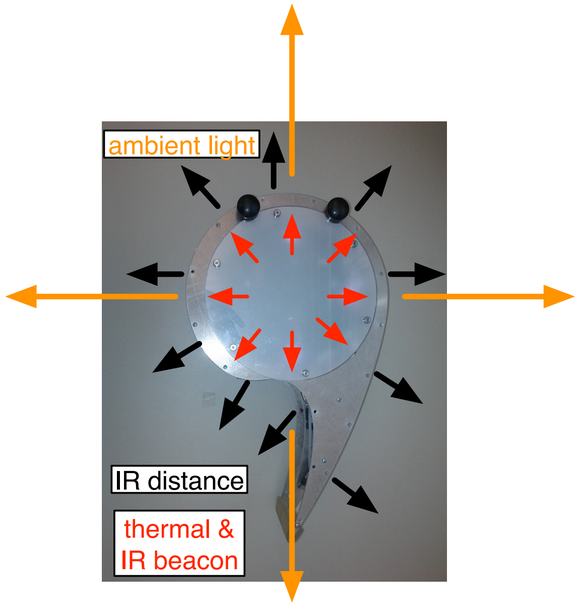
$$\mathbf{e}_t^{(i)} = \gamma^{(i)} \lambda \mathbf{e}_{t-1}^{(i)} + \mathbf{x}_t,$$

- Each demon forms a new prediction on each step:

$$V_t^{(i)} = \mathbf{x}_t^\top \mathbf{w}_t^{(i)}$$

Feature generation

- Used a variant of tile-coding, a common technique for mapping continuous inputs to high-dimensional binary feature vectors.



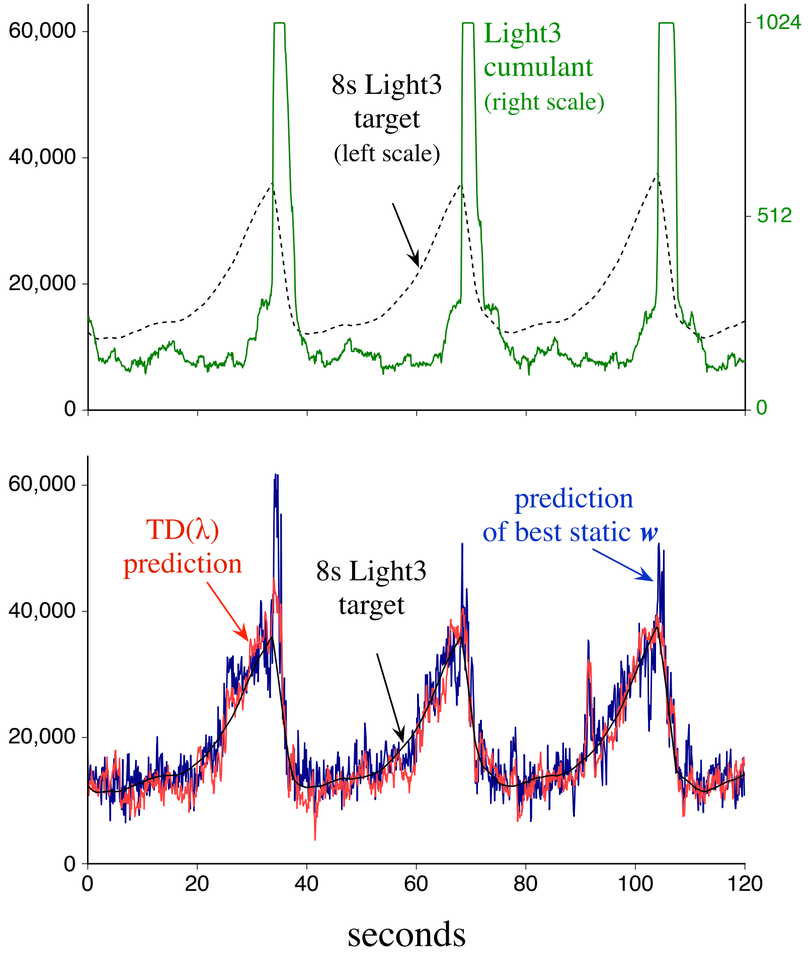
Sensor type	Num of sensors	tiling type	Num of intervals	Num of tilings
IRdistance	10	1D	8	8
		1D	2	4
		2D	4	4
		2D+1	4	4
Light	4	1D	4	8
		2D	4	1
IRlight	8	1D	8	6
		1D	4	1
		2D	8	1
		2D+1	8	1
Thermal	4(8)	1D	8	4
RotationalVelocity	1	1D	8	8
Magnetic	3	1D	8	8
Acceleration	3	1D	8	8
MotorSpeed	3	1D	8	4
		2D	8	8
MotorVoltage	3	1D	8	2
MotorCurrent	3	1D	8	2
MotorTemperature	3	1D	4	4
LastMotorRequest	3	1D	6	4
OverheatingFlag	1	1D	2	4

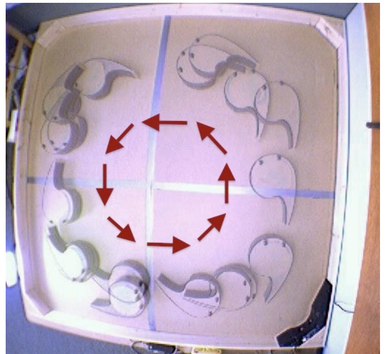
$$|\boldsymbol{x}| = 6065$$

Can we make and update many nexting predictions in real-time on a robot?

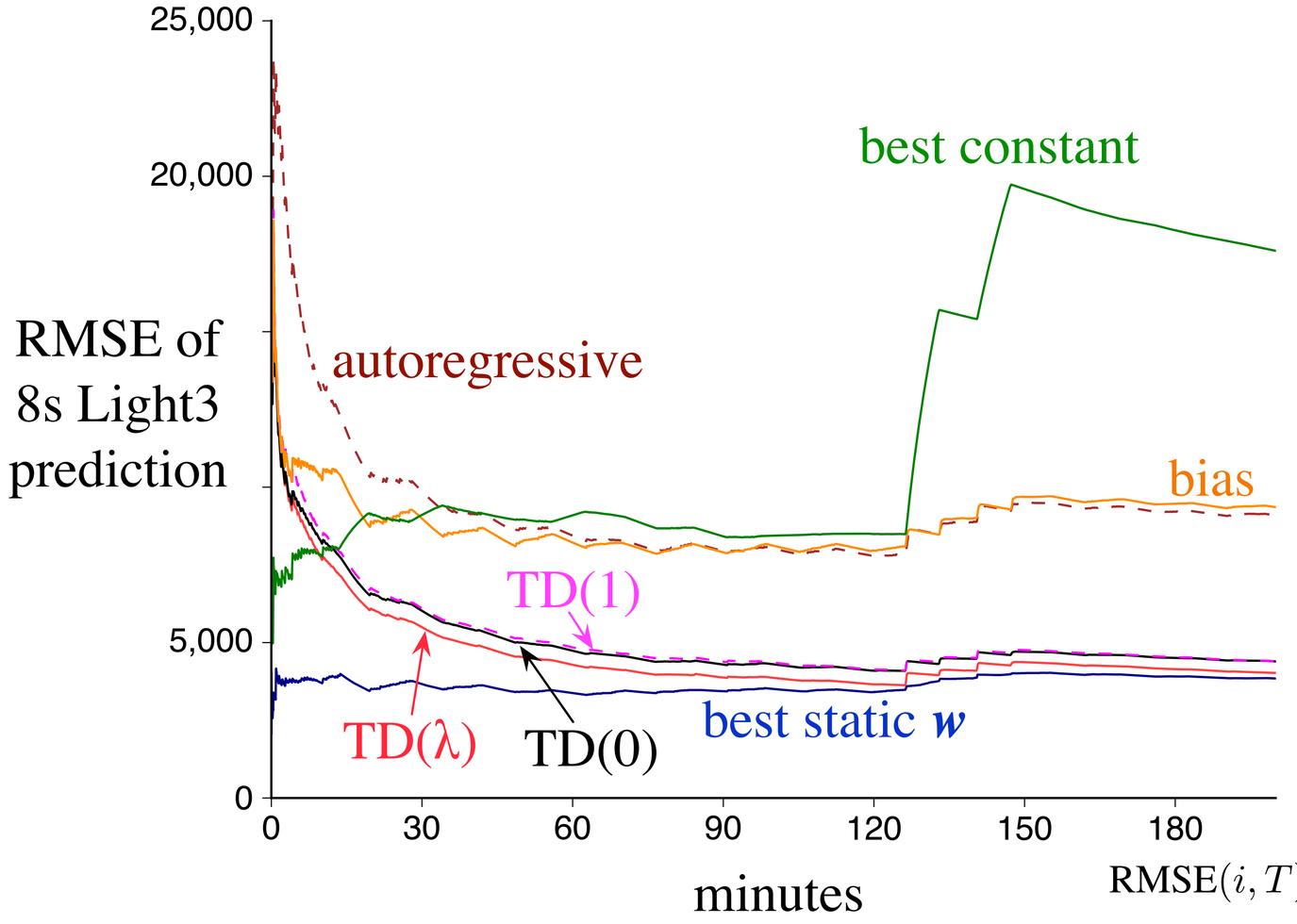
- **yes!**
- We specified 2160 nexting GVFs and ran the robot for 3h 20m (120k time steps)
- The wall-following policy, tile coding, and TD(λ) were implemented in Java and run on a laptop connected via wireless link to the Critterbot.
- The basic update loop took 55ms well within our 100ms budget
- Later on a newer laptop we repeated the experiment with 10000k GVFs in 85ms

A visual assessment of light prediction accuracy



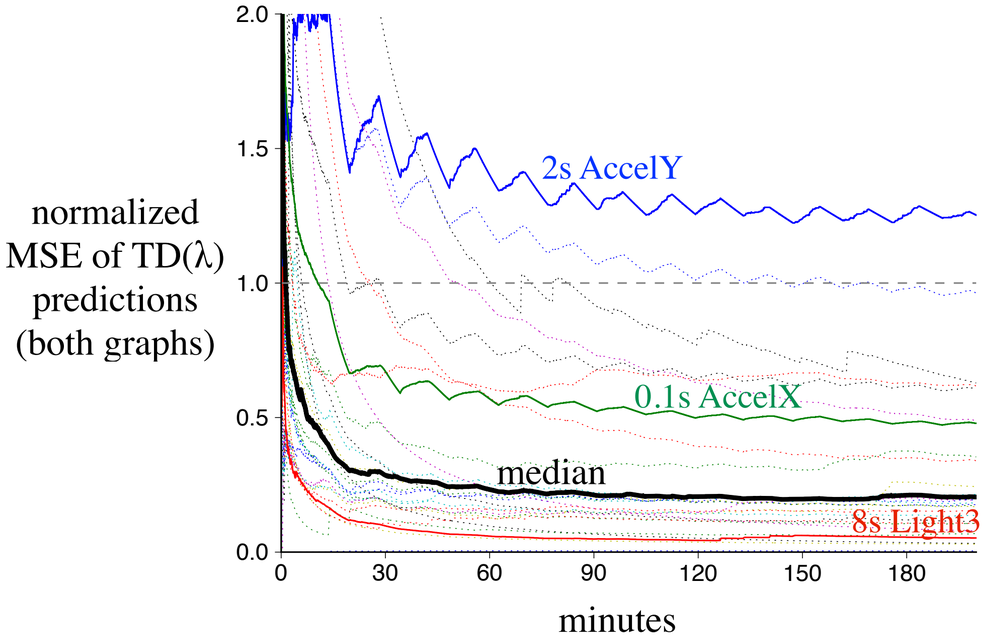


Accessing light prediction accuracy



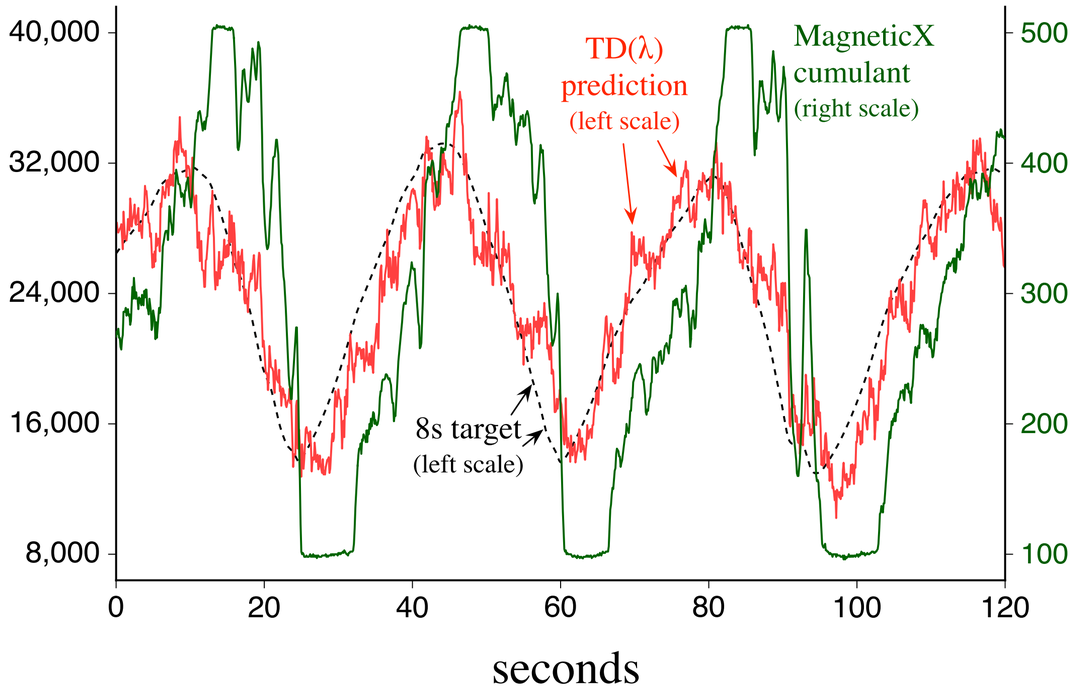
$$\mathrm{RMSE}(i,T) = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(V_t^{(i)} - G_t^{(i)})^2}.$$

Accessing accuracy of many Nexting predictions



- of the predictions whose cumulant was a sensor, the predictions explained 78% of the variance in the data (median).

Learning about an unknown world



Conclusions from the Nexting experiment

- We have implemented a robot version of the psychological phenomenon of Nexting
- Nexting is practical: scaling to thousands of predictions, based on thousands of features on a small computer
- Analysis of a subset of the predictions found them to be substantially accurate with 30 minutes of training
- Evidence that our predictive approach to knowledge based on GVFs and RL algorithms enables practical and large-scale online learning

- Should the agent-state include predictions?
  - Predictive state vs Free state
- Where do the GVFs come from?
- How should the agent behave?
- How does this all relate to planning and models?