Multi-step Prediction

Richard Sutton

Reinforcement Learning and Artificial Intelligence Laboratory

Department of Computing Science

University of Alberta

Skilled perception and action…learned without labels

Labels are are a crutch, a cheat

- The right representations must *already be in the unlabeled data!*
- Where do you think the labelers get them?
- We will never learn how to perceive properly until we stop relying on labels
- This is a fundamental weakness

Definitions

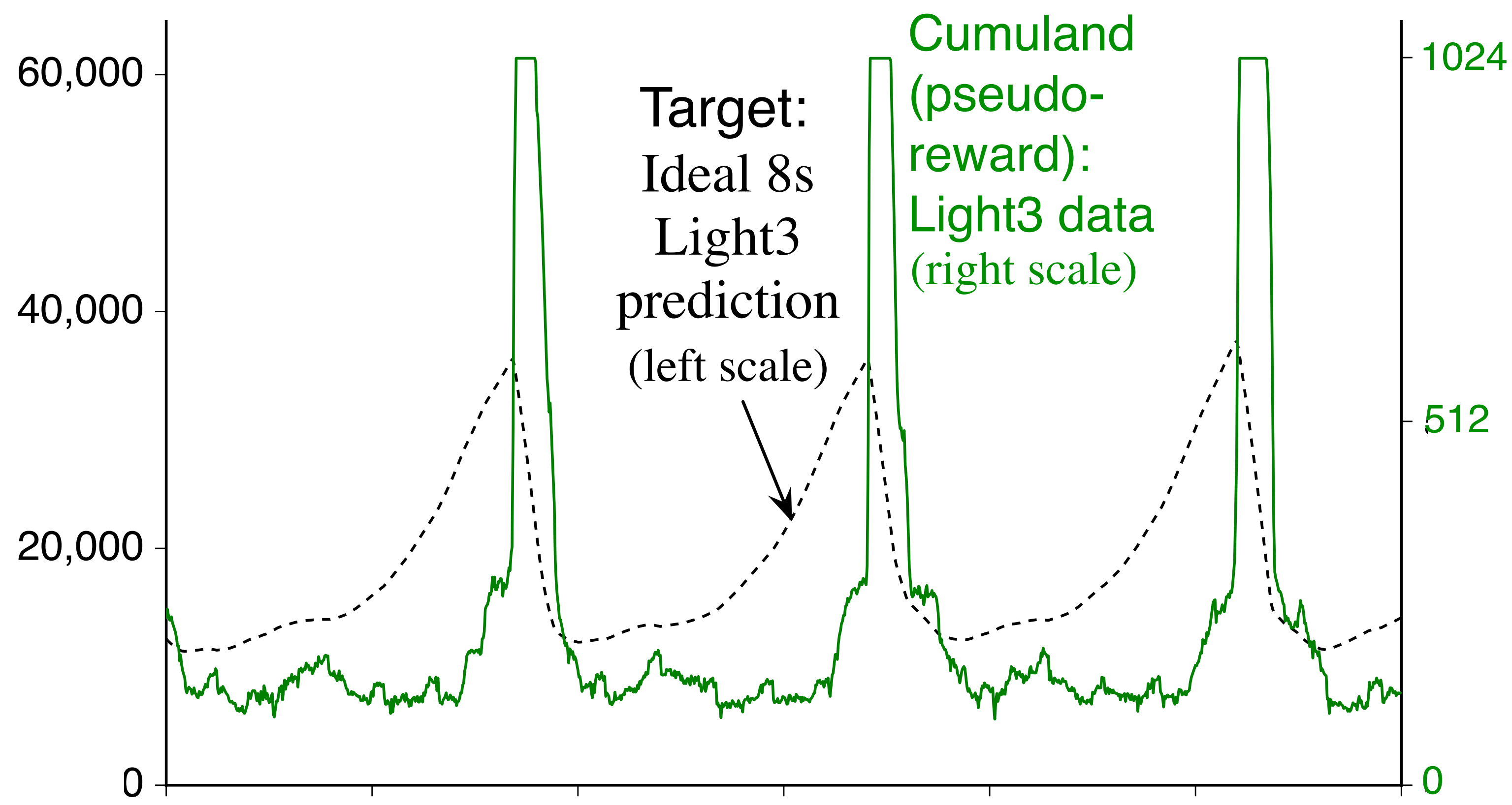
- Every 10ms or so we observe a new sensory vector and emit a new motor vector; this is the *data*
- By *prediction* I mean a statement at time *t* about what will happen at times >*t*
- I assume *repeated* prediction—at *each* time *t* we say something about the future
- If, at each *t*, we say something only about *t+1*, then that is a *one-step prediction*
- Everything else is a *multi-step prediction*

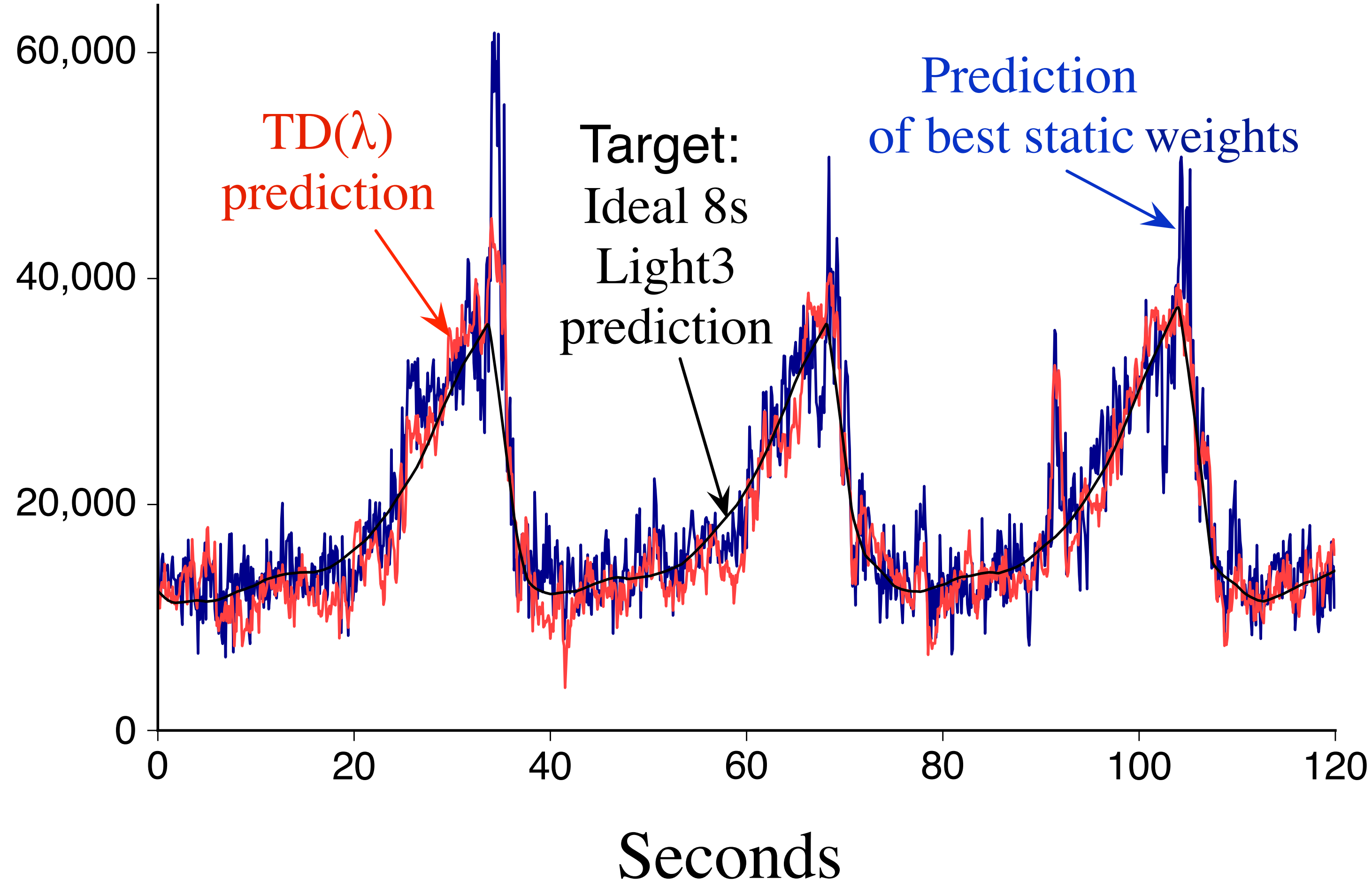
Examples of multi-step prediction

- Predicting the outcome of a game, like chess or backgammon
- Predicting what a stock-market index will be at the end of the year, or in six months
- Predicting who will be the next US president
- Predicting who the US will next go to war against
  - or how many US soldiers will be killed during a president's term
- Predicting a sensory observation, in 10 steps, in roughly 10 steps, or when something else happens
- Predicting discounted cumulative reward conditional on behavior

For Example: Predicting the discounted sum of future light-sensor readings

with a time constant of 8 seconds (80 steps, infinite span)





Predictive span

- The *span of a prediction* is the maximum number of time steps that might elapse between making a prediction and completely observing the outcome, or *target*
- One-step predictions have unit span
- Multi-step predictions have span > 1
- We are particularly interested in predictions that can be learned with computational complexity that is *independent of span*

Do we need to think about multi-step predictions?

- Can’t we just learn one-step predictions, and then iterate them (compose them) to produce multi-step predictions when needed?
- Can’t we just think of the multi-step as one big step, and then use one-step methods?
- No, we really can’t (and shouldn’t want to)

Can’t we learn one-step predictions, and iterate them to get multi-step predictions?

- Yes, sort of, but ultimately No, very much No
- Yes in the sense that if we have learned one-step predictions that are exactly correct, then they are informationally sufficient to make all multi-step predictions exactly correct; we would not have to learn cached answers to each one individually, we could just compute them on the fly
- However, computing the multi-step predictions involves a branching process (if the world is stochastic or we have action choices); the complexity of making the prediction is *exponential* in the span
- If there is any error in the one-step predictions, then the error will propagate and *expand exponentially* with span; the multistep predictions will be poor approximations, much poorer than if they were learned directly

Can’t we just use our familiar one-step learning methods?

- Can’t we just wait until the target is known, then use a one-step method? (reduce to input-output pairs)
    - E.g., wait until the end of the game, then regress to the outcome
- No, not really; there are significant computational costs to this
    - memory is O(span)
    - computation is poorly distributed over time
- These can be avoided with learning methods specialized for multi-step
- Also, sometimes the target is never known (off-policy)
- We should not ignore these things; they are not nuisances, they are <u>clues</u>, hints from nature

Every prediction has both a *question* and an *answer*

- Q: How much rain will fall in the next 24 hours? A: 0.5 centimeters
- Q: Will i win this chess game? A: with Probability 0.9
- Q: What will the dow jones index be at the end of the year? A: 18,000
- Q:What will be the discounted sum of rewards from here forward? A: 5.7 (or whatever the value of the state is)

- The question describes the procedure for calculating the target
- The answer is the expected value of the target (say)
- The answer process is familiar; it might be a deep/neural network
- The question process is less familiar, possibly more important

Generalized value functions (GVFs)

- A particular class of predictive questions
  - Inspired by the value functions of optimal control/RL
  - Whose answers can be learned in a computationally efficient way
    - independent of span and taking advantage of the state property (approximate Markov)
  - Linked to planning via dynamic programming and ‘option’ models

The question part of a GVF

- Given
  - a policy $\pi : \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$
  - a signal to be added up, the *cumuland* $R_t \in \Re$
  - a termination or discounting condition $\gamma : \mathcal{S} \rightarrow [0, 1]$
- The target is the sum of the cumuland signal up until termination, if the policy is followed:

$$G_t = \frac{\pi(A_t|S_t)}{\mu(A_t|S_t)} \Big(R_{t+1} + \gamma(S_{t+1}) G_{t+1} \Big)$$

behavior policy

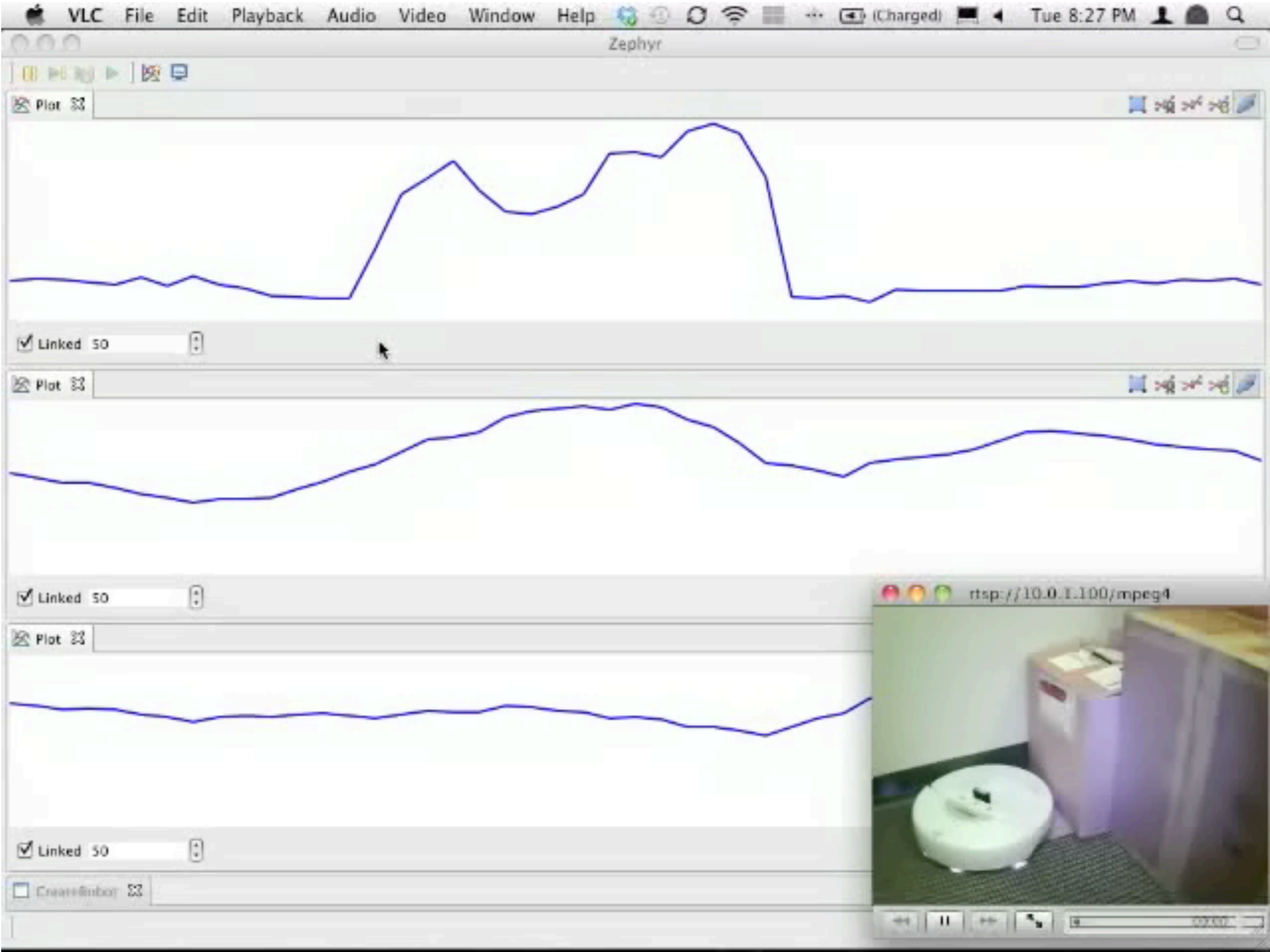
The answer part of a GVF

(determines the nature of the approximation)

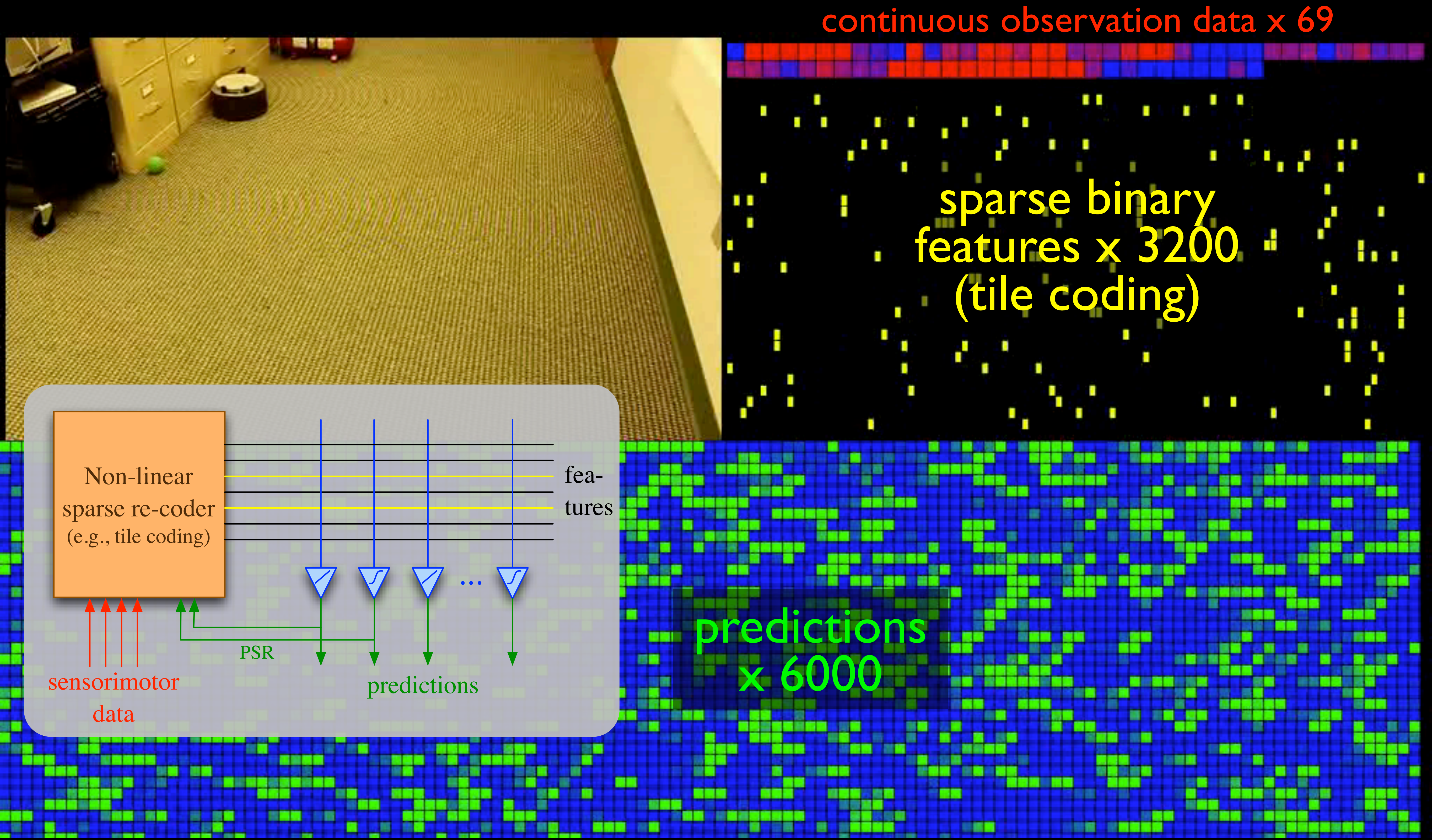
- Given:
  - a parameterization (e.g., the features of a linear approximation, the structure of a deep net) $\hat{v} : \mathcal{S} \times \Re^n \rightarrow \Re$
  - a bootstrapping function $\lambda : \mathcal{S} \rightarrow [0,1]$
  - an interest function $i : \mathcal{S} \rightarrow \Re_+$
  - a source of data (e.g., behavior policy) $\mu : \mathcal{A} \times \mathcal{S} \rightarrow [0,1]$
- Find $\boldsymbol{\theta}$ to minimize: $\sum_{s \in \mathcal{S}} d_\mu(s) i(s) \Big(\hat{v}(s, \boldsymbol{\theta}) - \mathbf{E}_\pi \big[G_t^\lambda | S_t = s \big] \Big)^2$

where:

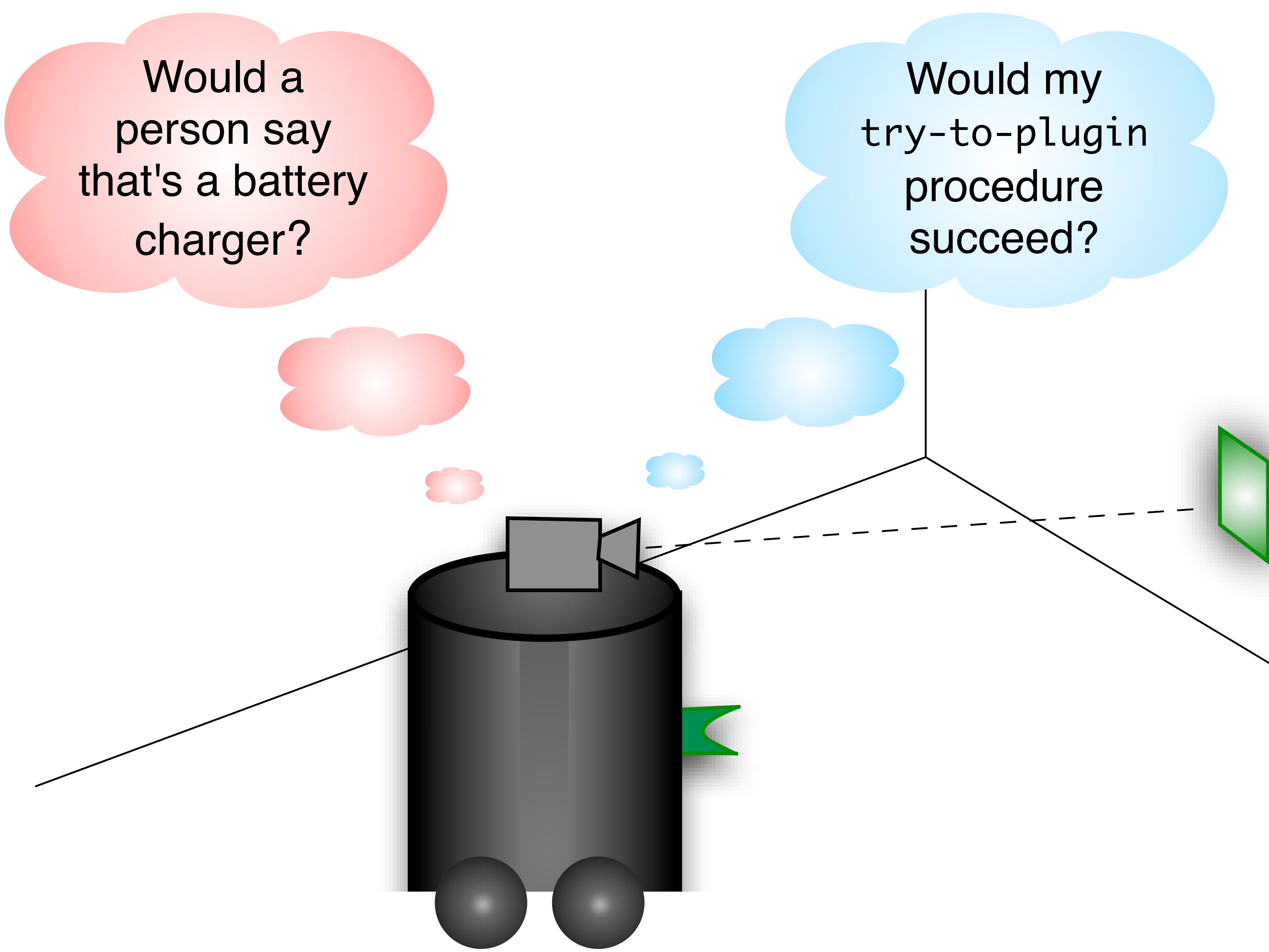
$$G_t^\lambda = R_{t+1} + \gamma(S_{t+1}) \Big((1 - \lambda(S_{t+1})) \hat{v}(S_{t+1}, \boldsymbol{\theta}) + \lambda(S_{t+1}) G_{t+1}^\lambda \Big)$$


VLC File Edit Playback Audio Video Window Help
(Charged)
Tue 8:27 PM
Zephyr
Plot
Linked 50
Plot
Linked 50
Plot
Linked 50
rtsp://10.0.1.100/mpeg4

Massive real-time prediction learning
Up to one billion weight updates/second



The power of policy conditioning



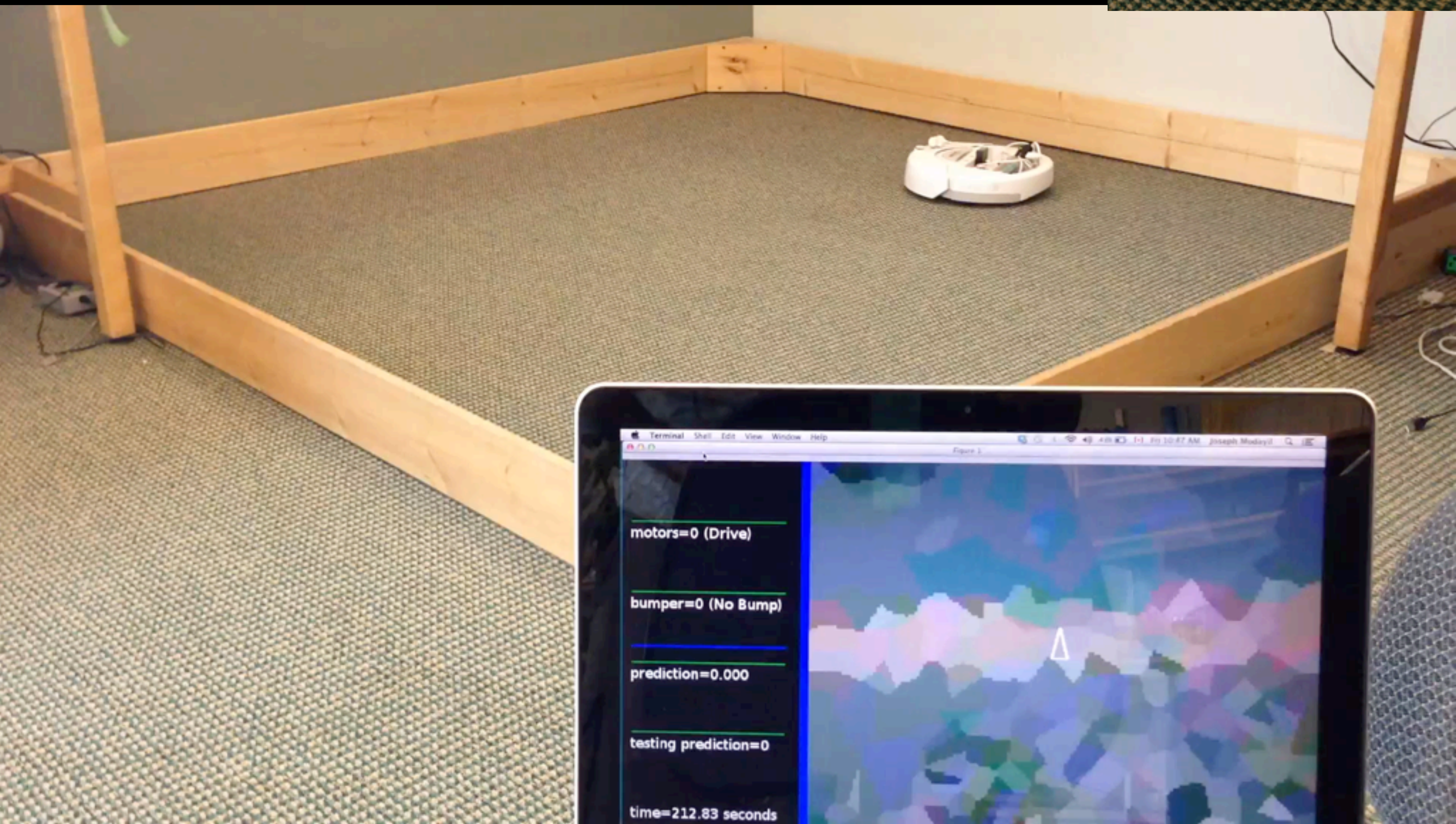
Multi-step predictions represent more interesting perceptual concepts

- They include physical things, like distance and weight
- But also higher-level, more abstract and cognitive things, like functions and opportunities, possibilities
  - a pitch I can hit, a girl I can kiss, a thing I can sit on, a way to get my email…
- cf. correlations between simultaneous signals
- cf. information theory, compression
- cf. invariances

Algorithmic issues in multi-step prediction

- eligibility traces
- temporal-difference learning
- off-policy learning and importance sampling
    - a very challenging technical problem with new methods still being proposed
- supporting composition and planning

An example of using the predictions for control



In conclusion, why do multi-step predictions matter?

- They are another source of data for perceptual learning
  - from sensory or sensorimotor (robot) streams
  - these are potentially *huge and scalable*
- They yield *higher-level concepts* than do one-step predictions
- Their questions can be represented *in the machine*
- Different algorithms are needed to learn them efficiently in both data and computation

Thank you for your attention

and thanks to

Rupam Mahmood, Adam White, Joseph Modayil, Harm van Seijen, Doina Precup, Hado van Hasselt