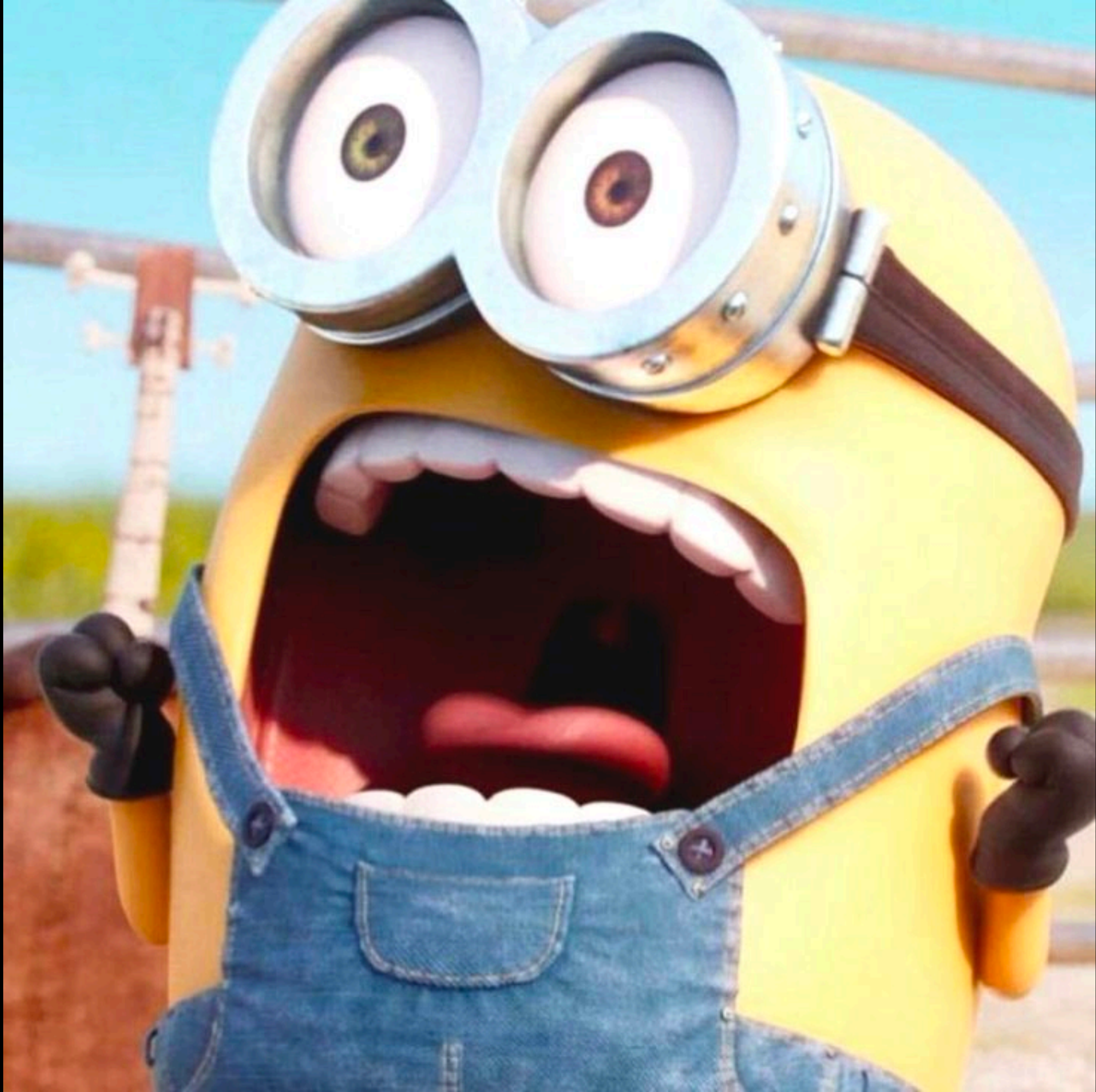
Policy Search and Black-Box Policy Optimization

Sanjiban Choudhury

Cornell Bowers CIS
Computer Science

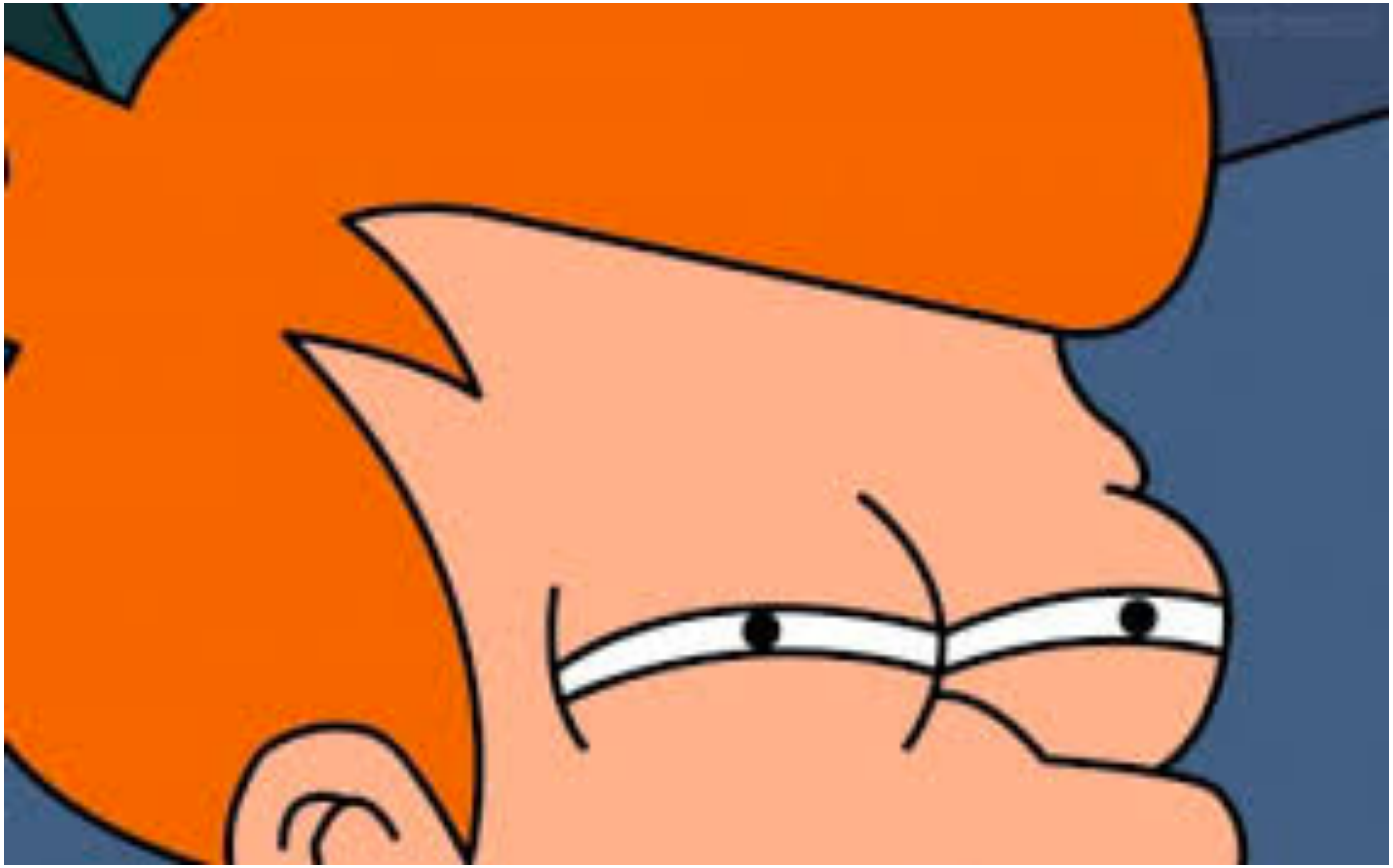
CRISIS !!!

Errors in neural network
get amplified by
dynamic programming
(Bootstrapping)



QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation

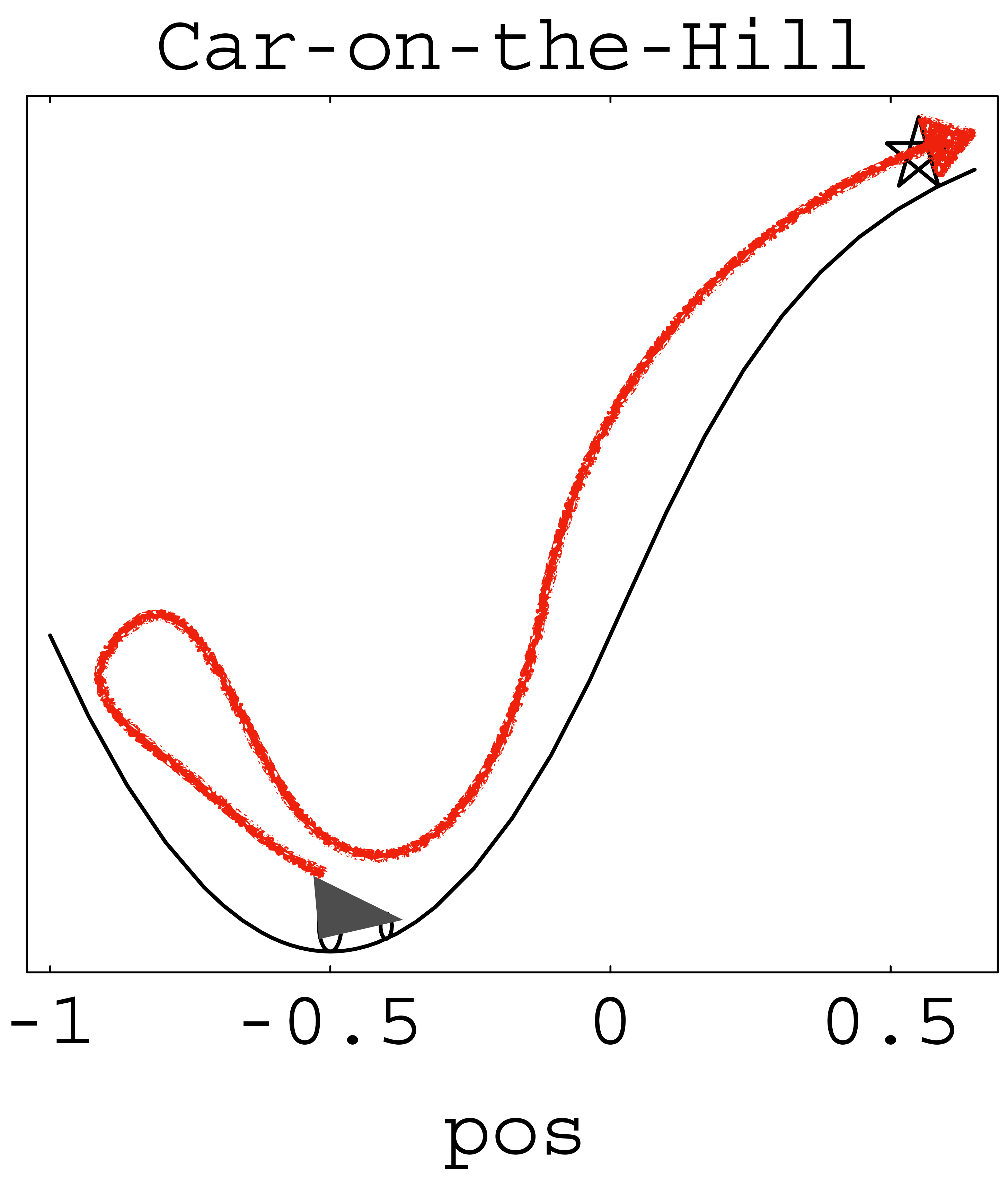
To hell with Value Estimates!



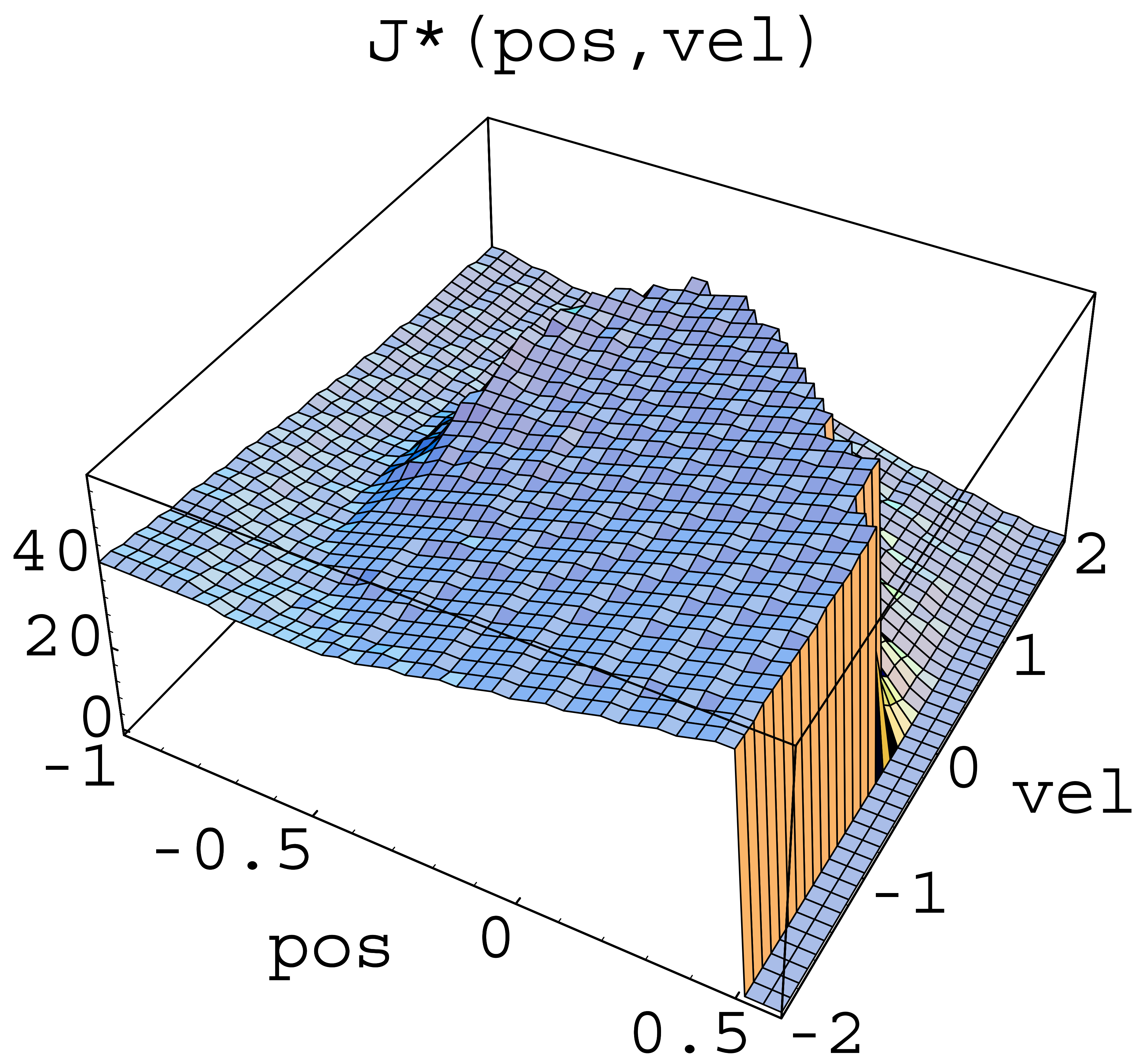
Trust ONLY actual Returns

What if we focused on finding good policies ... ?

Sometimes a policy is waaaaay simpler than the value



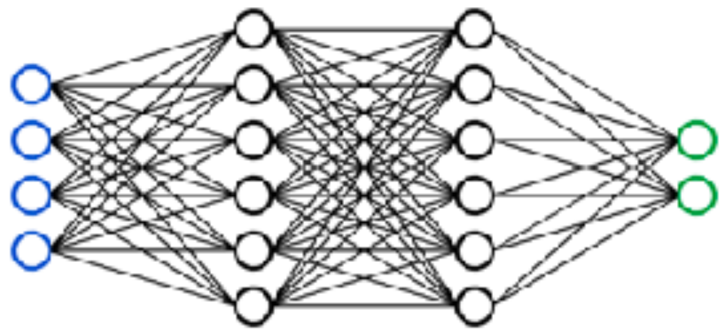
The Policy!



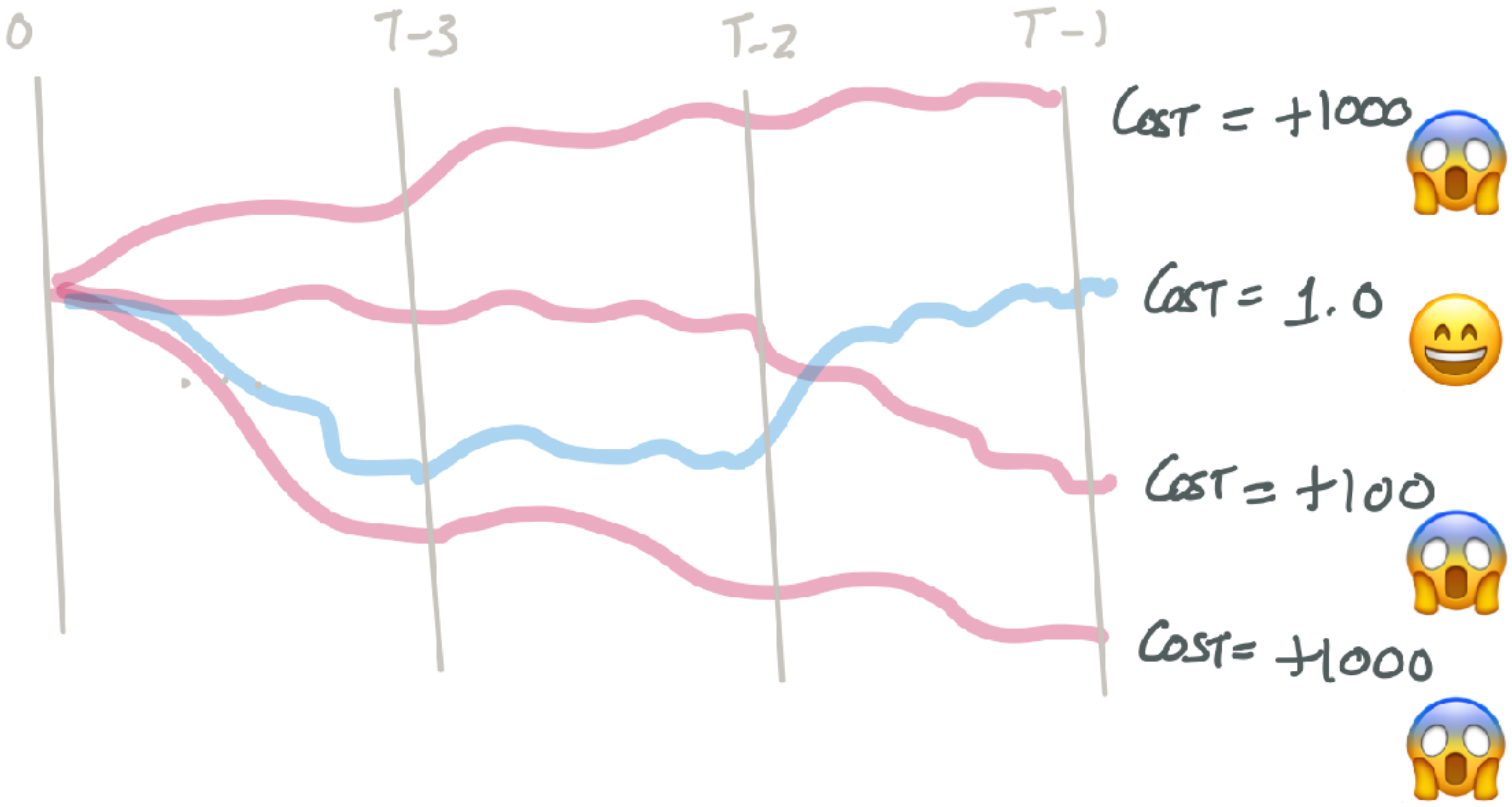
The Value!

Can we just focus on finding a good policy?

$$\pi_\theta : s_t \rightarrow a_t$$

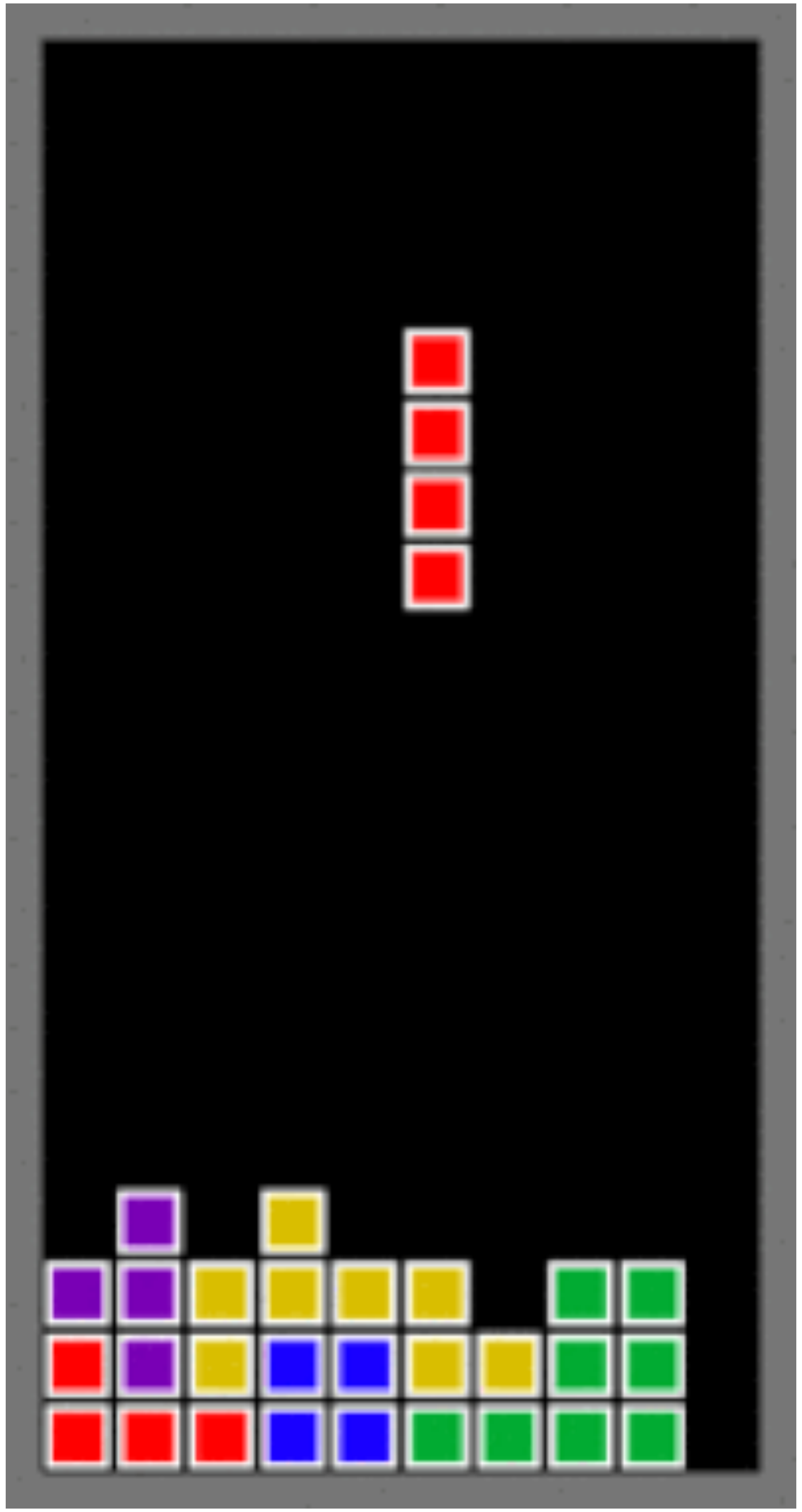


Learn a mapping from states to actions



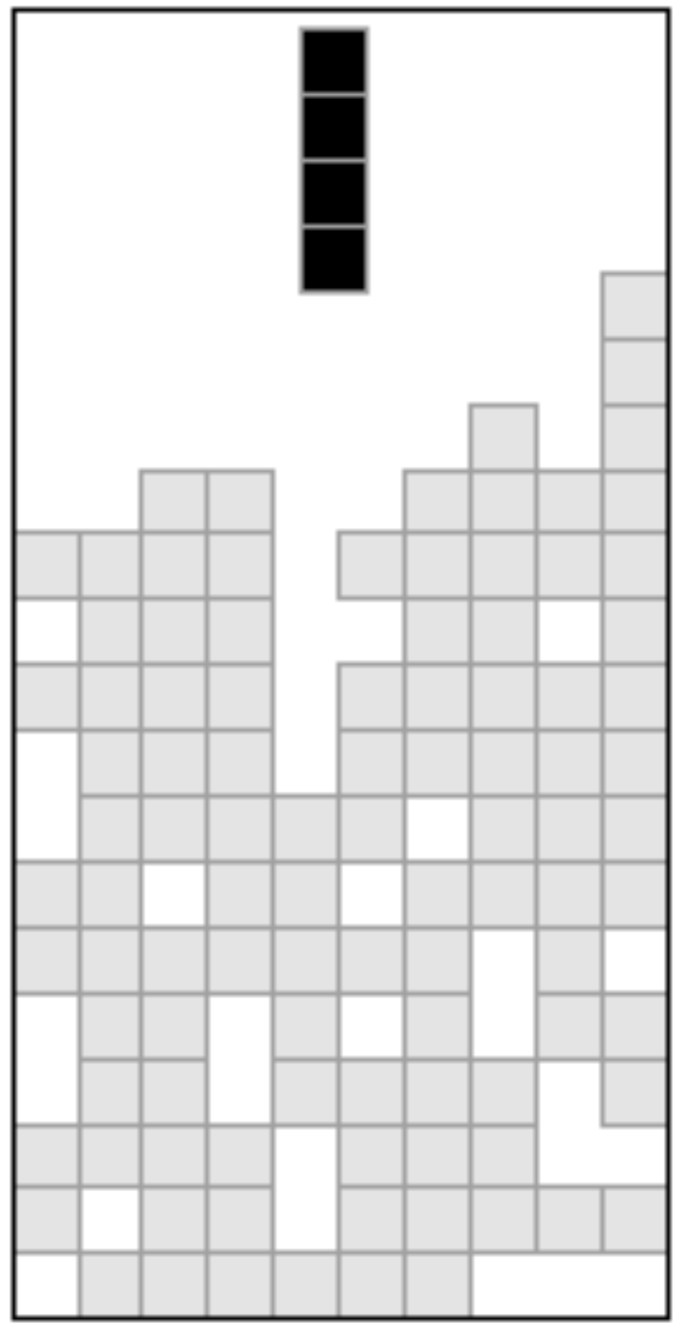
Roll-out policies in the real-world to estimate value

The Game of Tetris



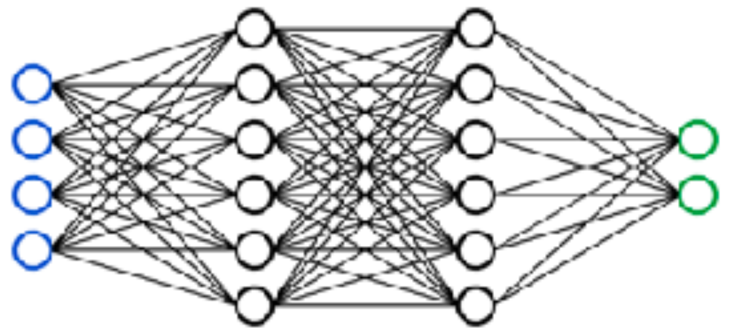
What's a good policy representation for Tetris?

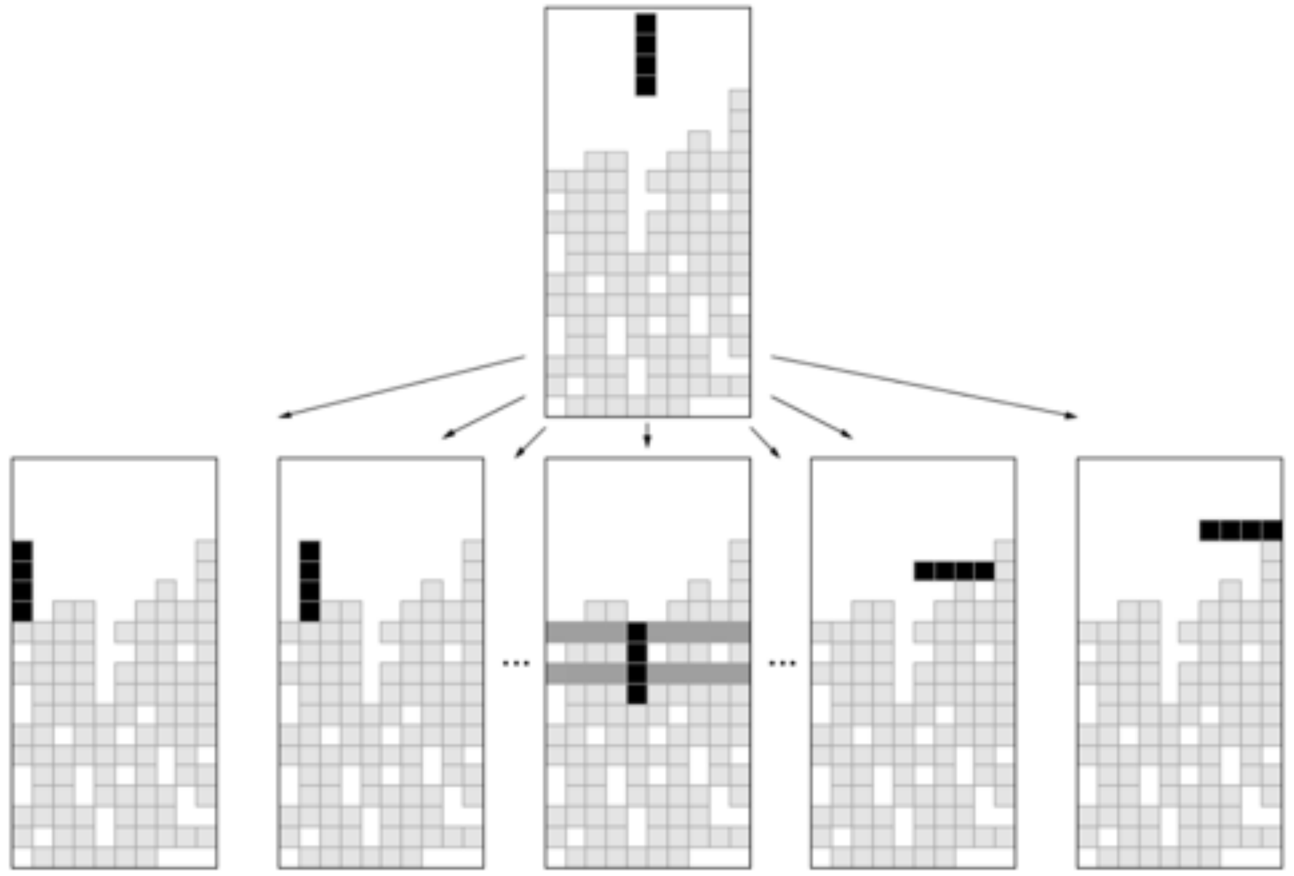
(4 rotations)*(10 slots)
- (6 impossible poses) = 34



State (s_t)

$$\pi_\theta : s_t \rightarrow a_t$$





Action (a_t)

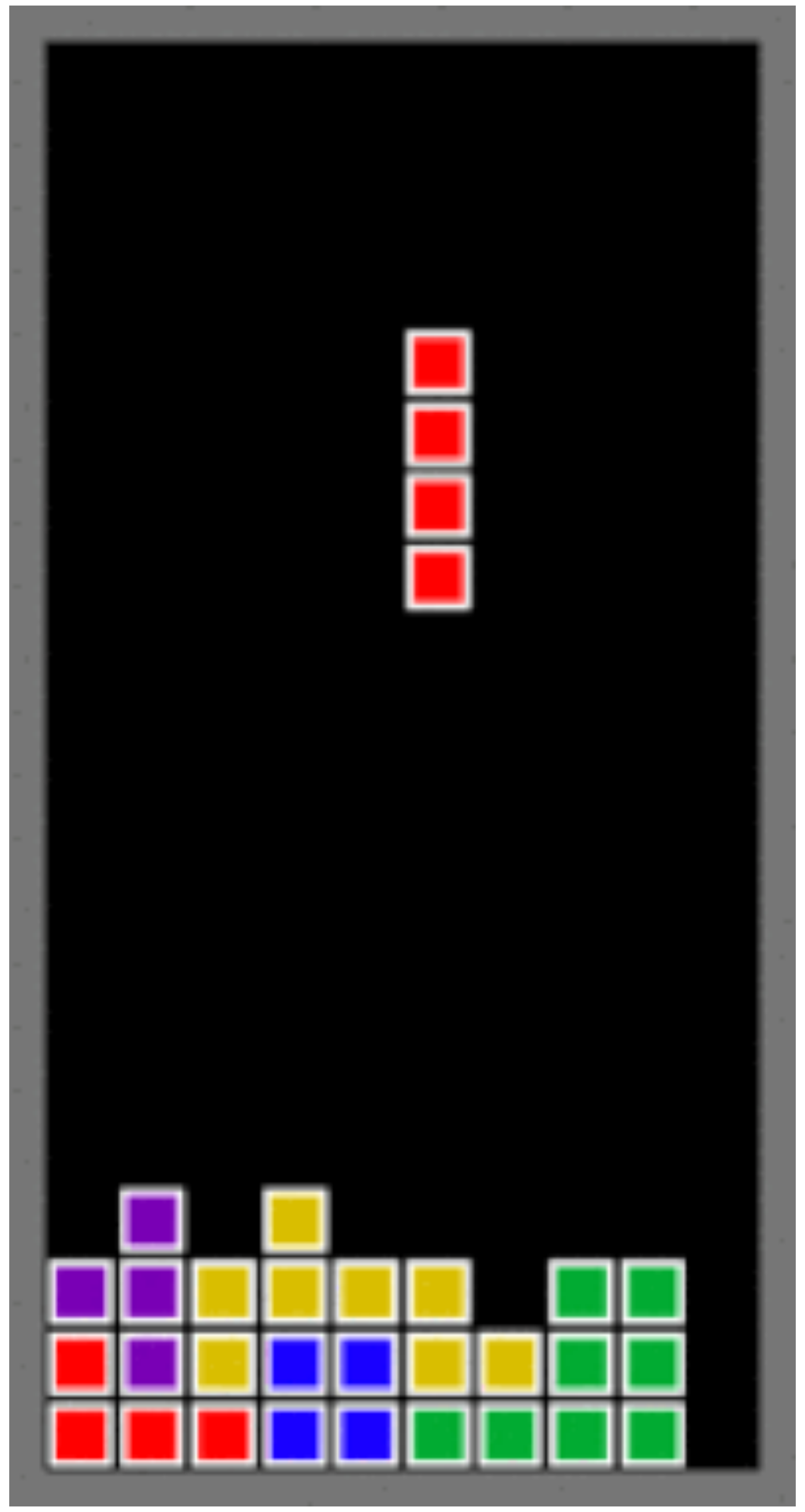
Activity!

Think-Pair-Share

Think (30 sec): Ideas for how to represent policy for tetris?

Pair: Find a partner

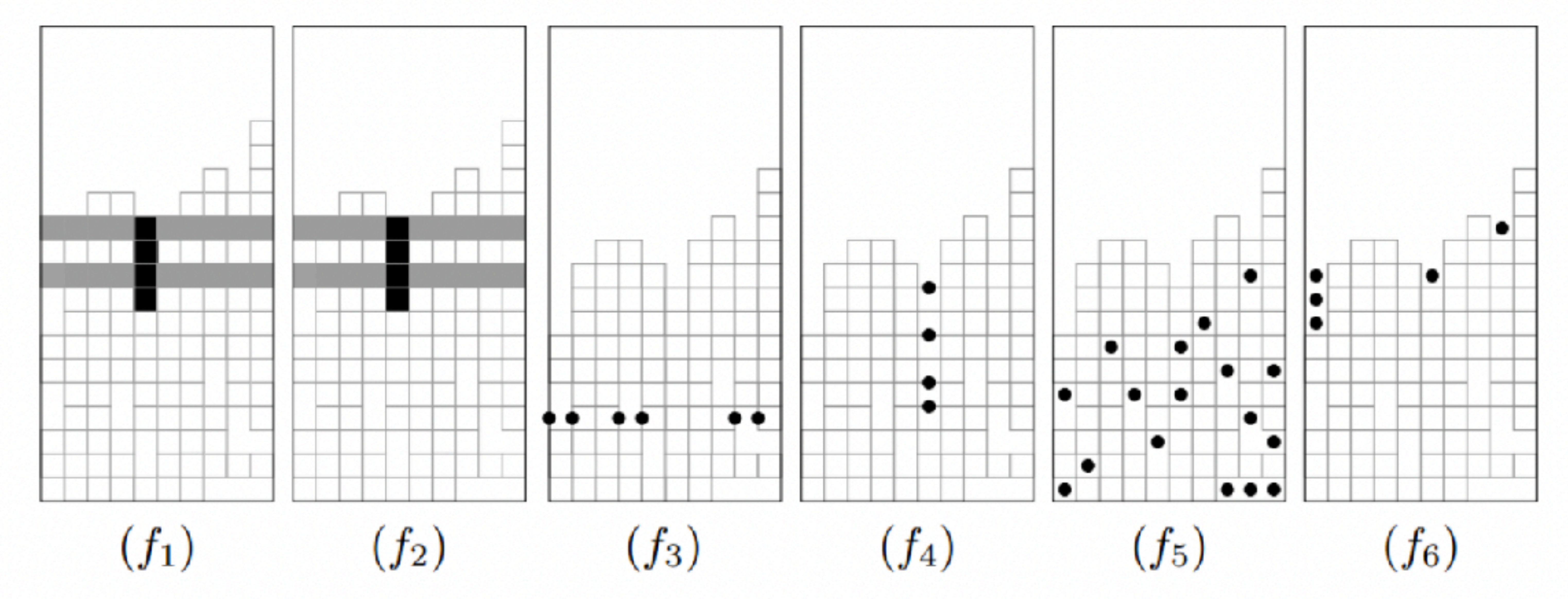
Share (45 sec): Partners exchange ideas



Some inspiration for Tetris policy

Until 2008, the best artificial Tetris player was handcrafted, as reported by Fahey (2003). Pierre Dellacherie, a self declared average Tetris player, identified six simple features and tuned the weights by trial and error.

Dellacherie Features



Landing Heights

Eroded Cells

Row Transitions

Column Transitions

Holes

Cumulative Wells

The contribution of the last piece to the cleared lines time the number of cleared lines.

The number of filled cells adjacent to the empty cells summed over all rows

A well is a succession of empty cells and the cells to the left and right are occupied

A *magic* formula ?!?

$$-4 \times holes - cumulative\,wells$$
$$- row\,transitions - column\,transitions$$
$$- landing\,height + eroded\,cells$$

A *magic* formula ?!?

$$-4 \times holes - cumulative\,wells$$
$$- row\,transitions - column\,transitions$$
$$- landing\,height + eroded\,cells$$

This linear evaluation function cleared an average of 660,000 lines on the full grid …
… In the simplified implementation used by the approaches discussed earlier, the games would have continued further, until every placement would overflow the grid. Therefore, this report underrates this simple linear rule compared to other algorithms.

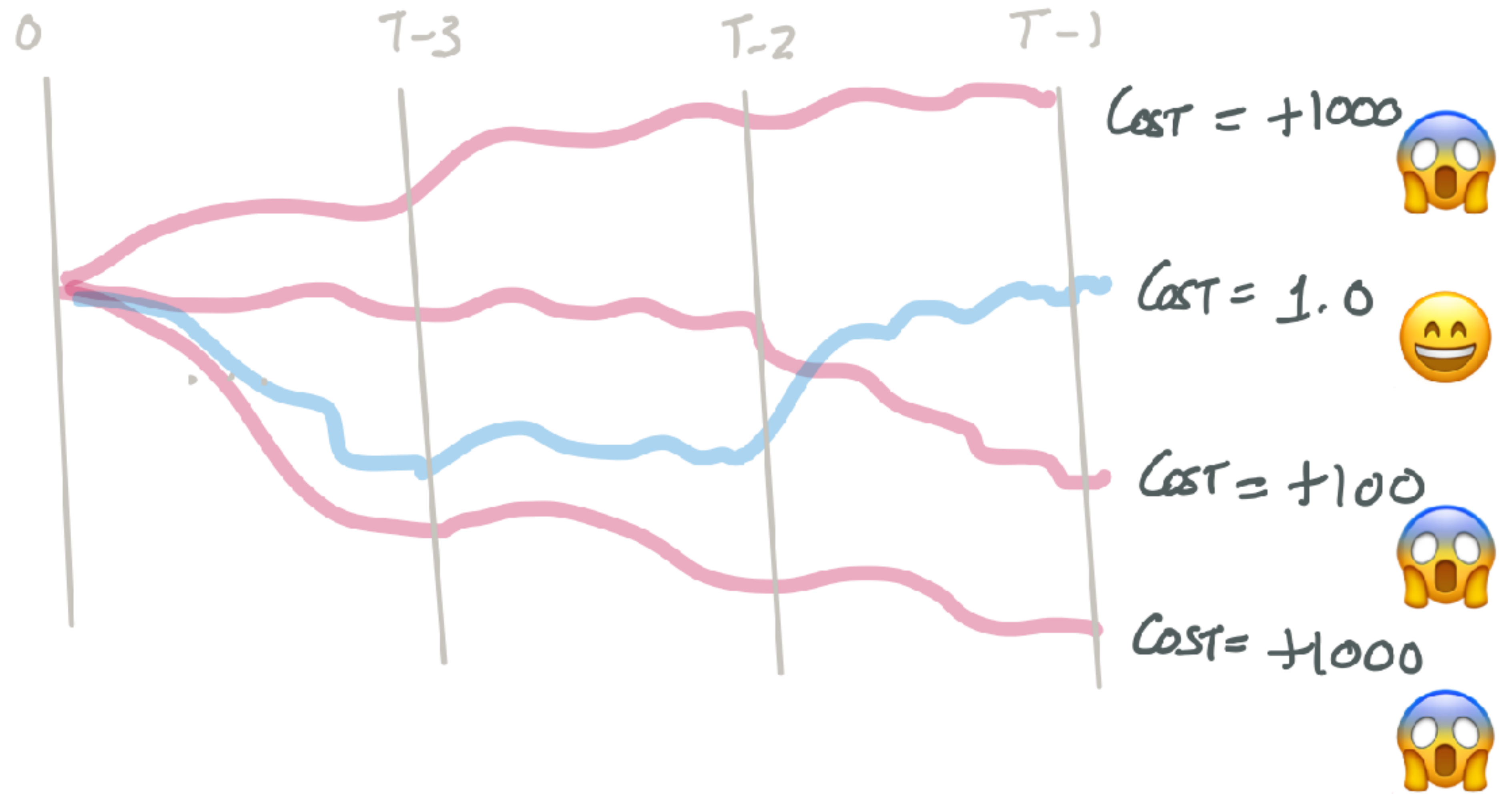
Can YOU do better than Dellacherie?

The Goal of Policy Optimization

$$\pi_\theta(s) = \arg\min_a \theta^T f(s, a)$$

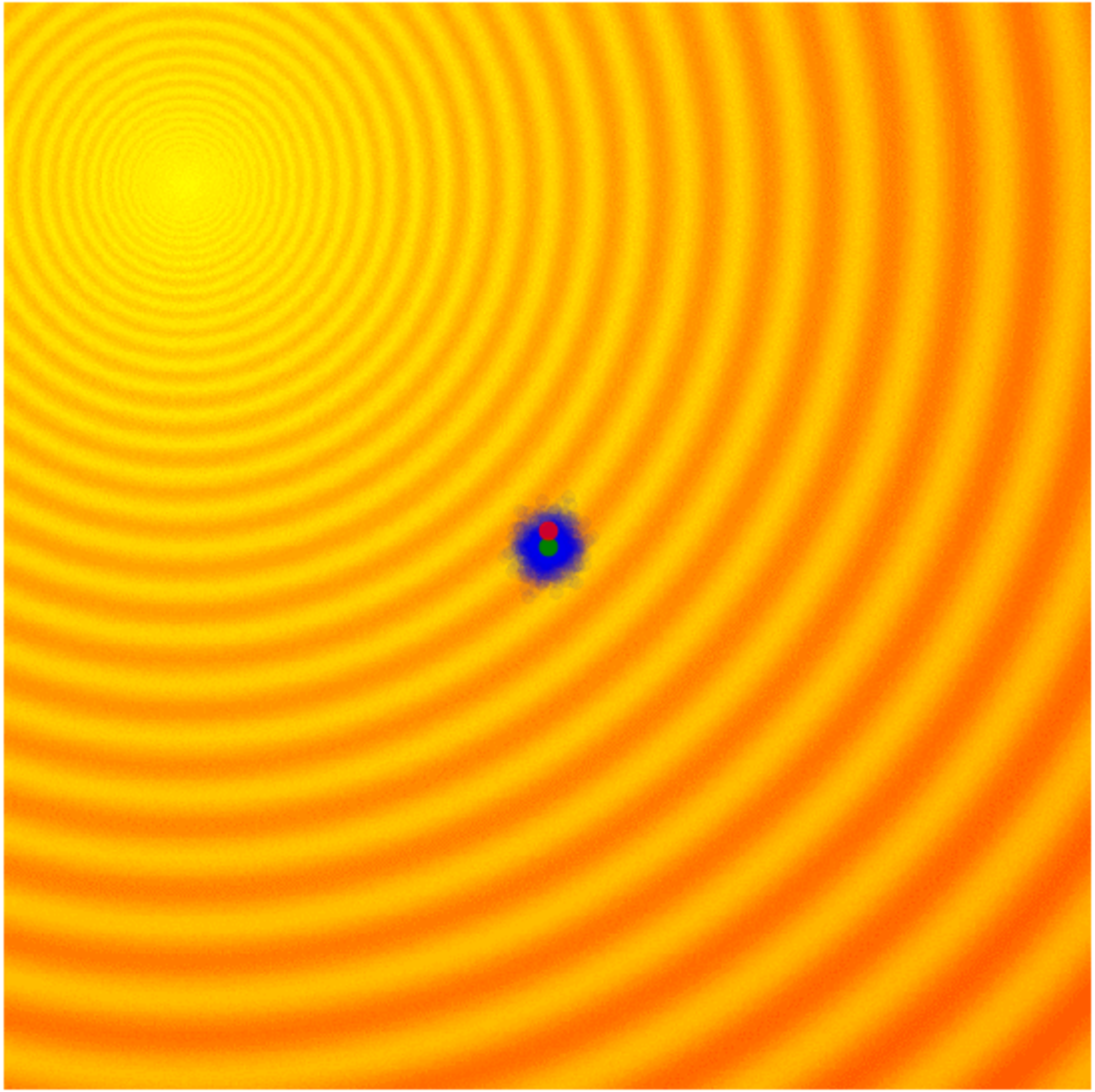
#Think of f(s,a) being dellacherie features

$$\min_\theta J(\theta) = \sum_{t=0}^{T-1} \mathbb{E}_{\pi_\theta} c(s_t, a_t)$$



#Think of c(s,a) as
-num_rows_cleared


Cross Entropy
If you were ever stranded on an island ...



Credit: https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

Green: Mean of distribution

Blue: Samples from distribution

Red: Best solution found so far

Let's formalize!

The Cross Entropy Algorithm

D_θ

INIT D_θ

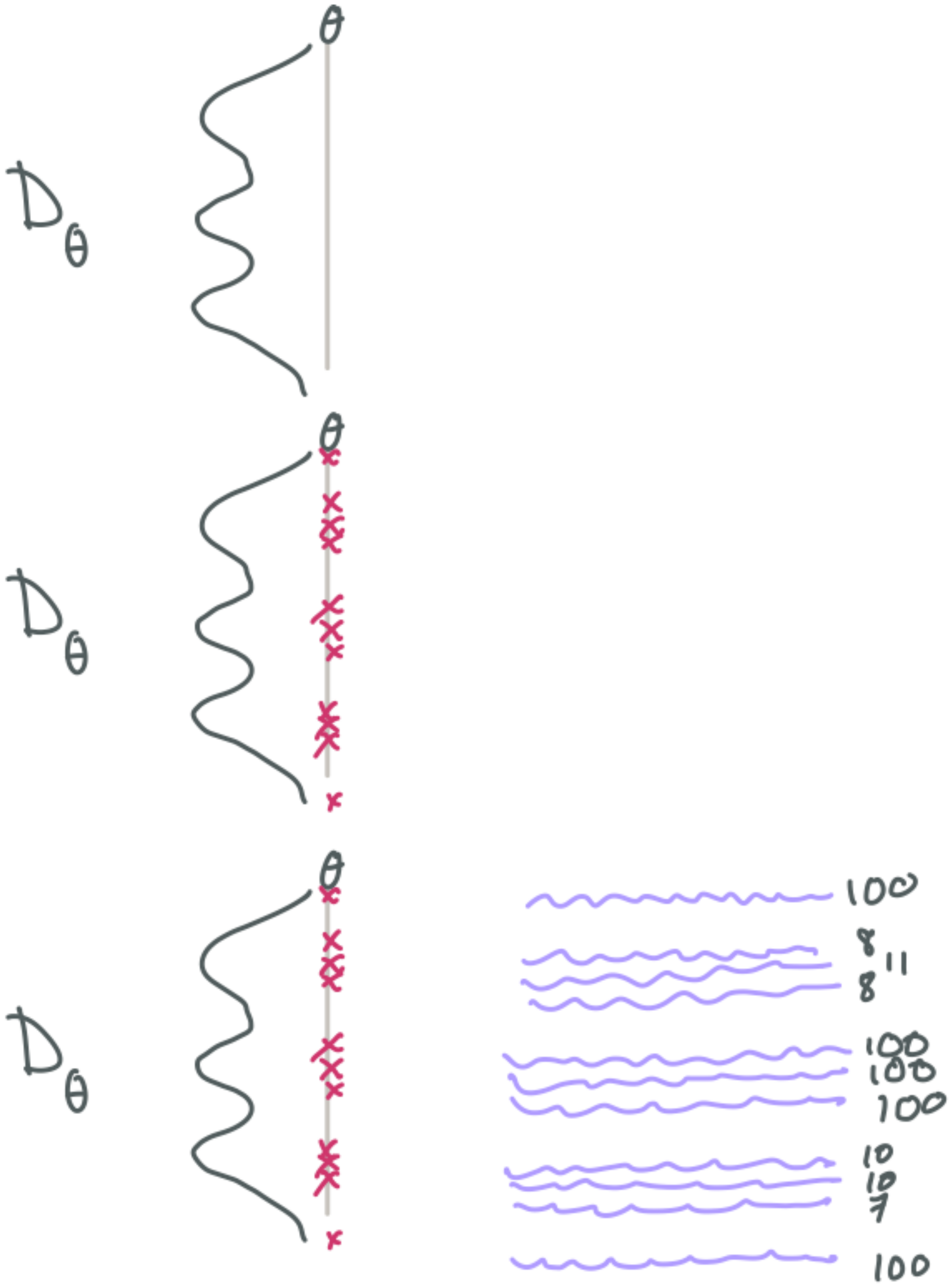
The Cross Entropy Algorithm

INIT D_θ

SAMPLE k TIMES

to get $\{\theta_i\}_{i=1}^{k}$

The Cross Entropy Algorithm



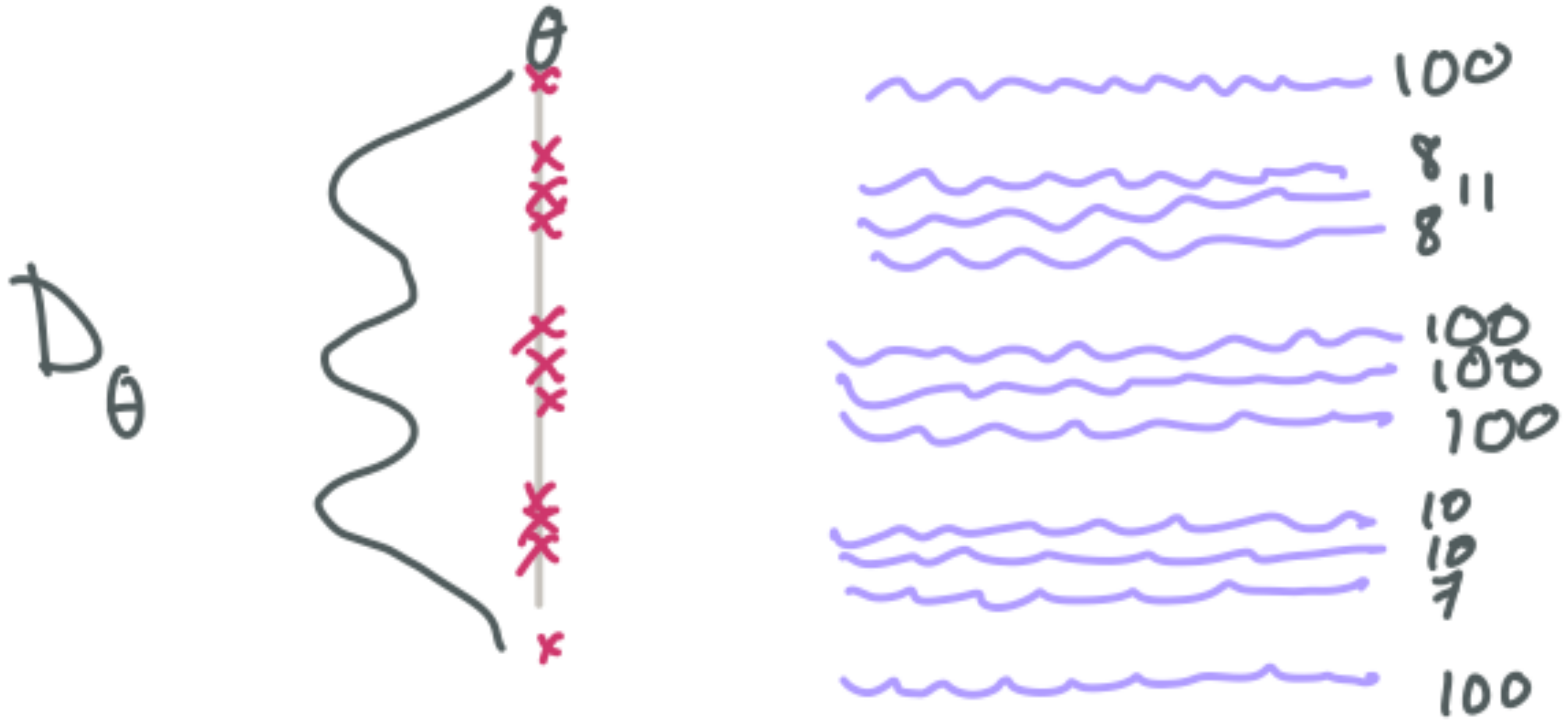
INIT D_θ

SAMPLE k TIMES to get $\{\theta_i\}_{i=1}^{k}$

EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES

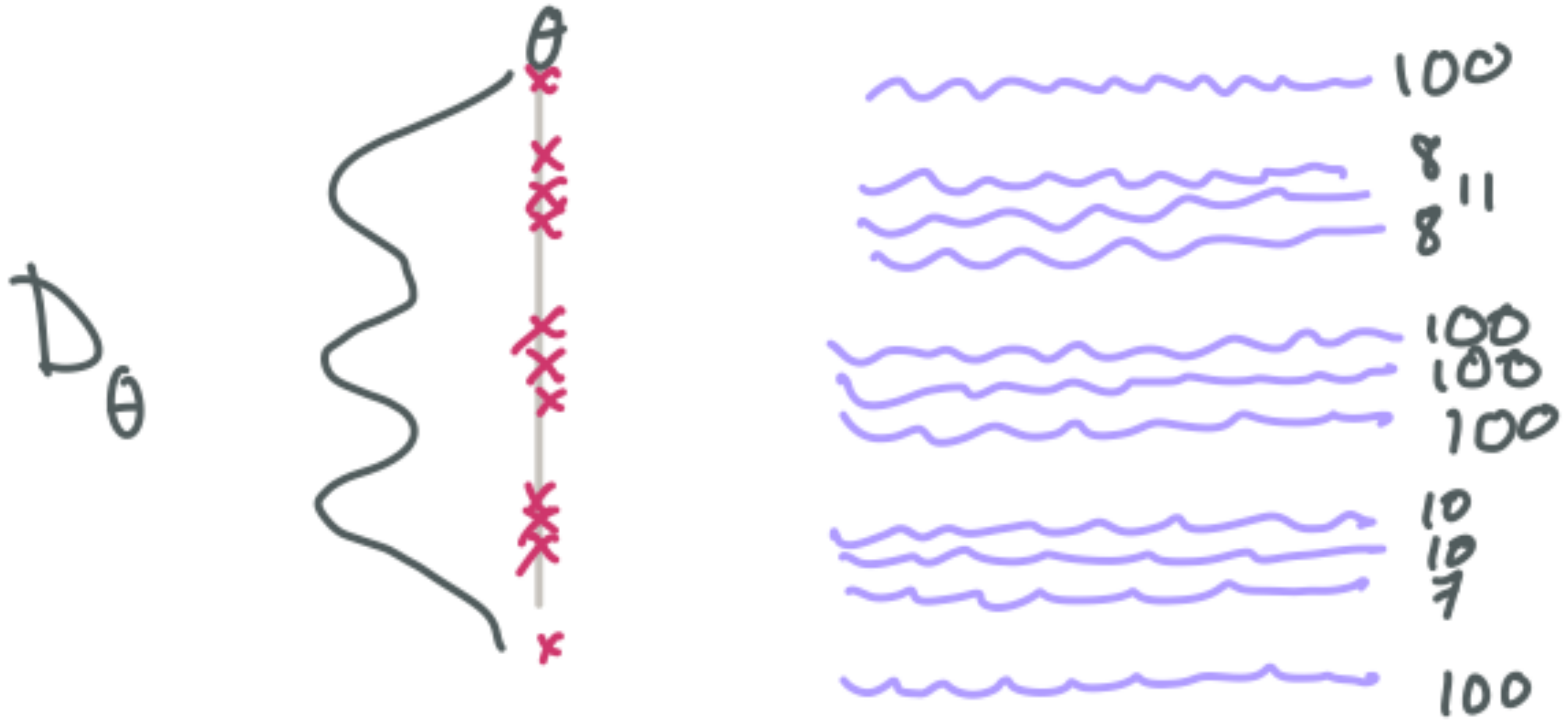
The Cross Entropy Algorithm



EVALUATE EACH θ_i:

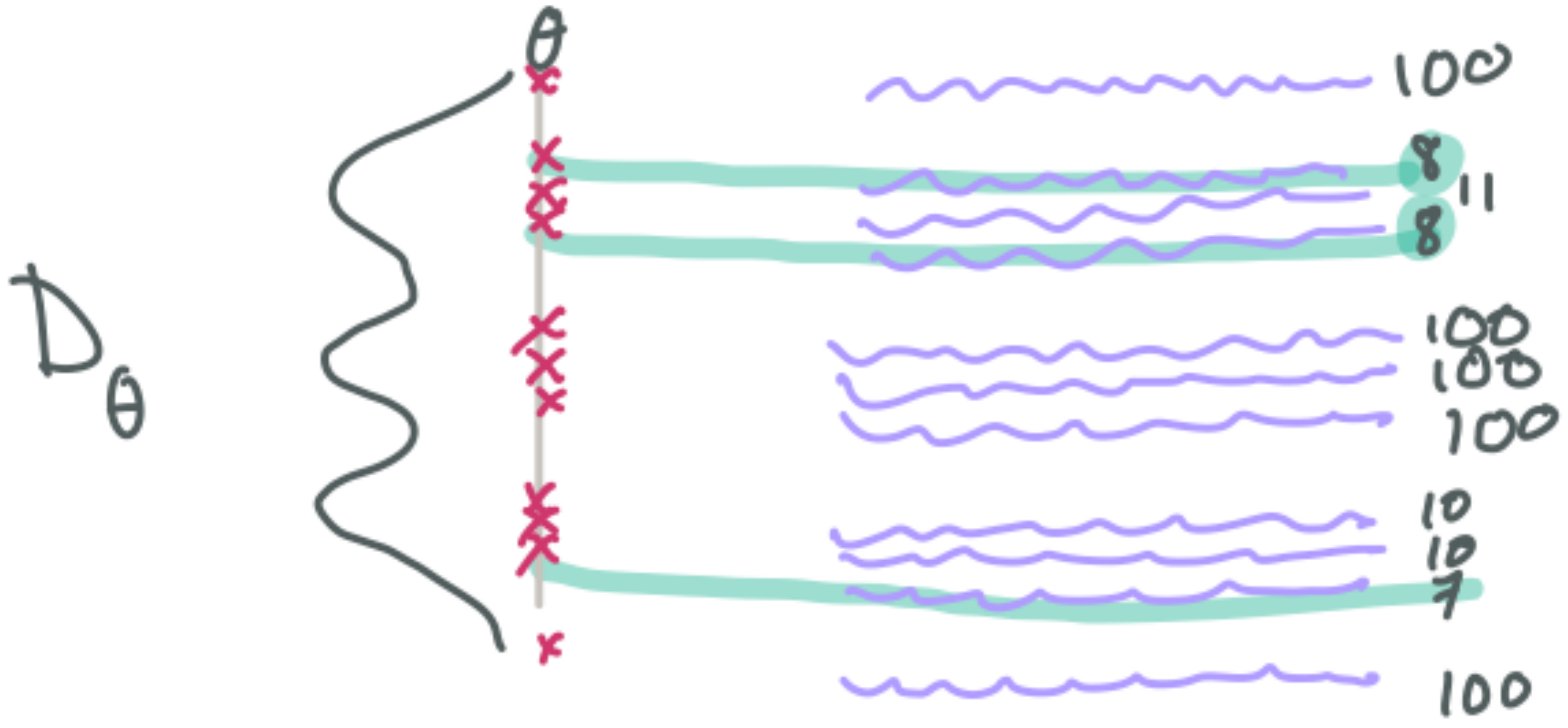
- EXECUTE POLICY MULTIPLE TIMES

The Cross Entropy Algorithm



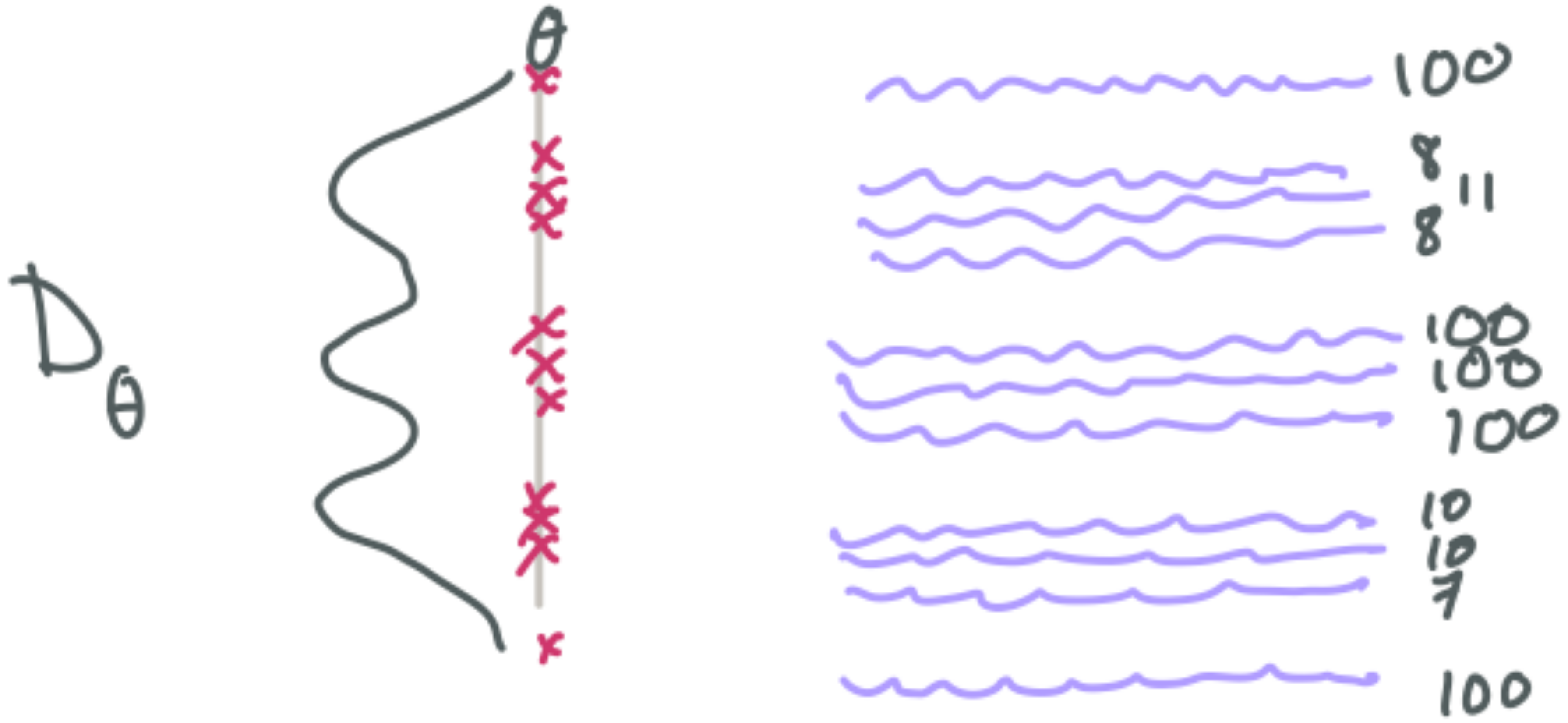
EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES



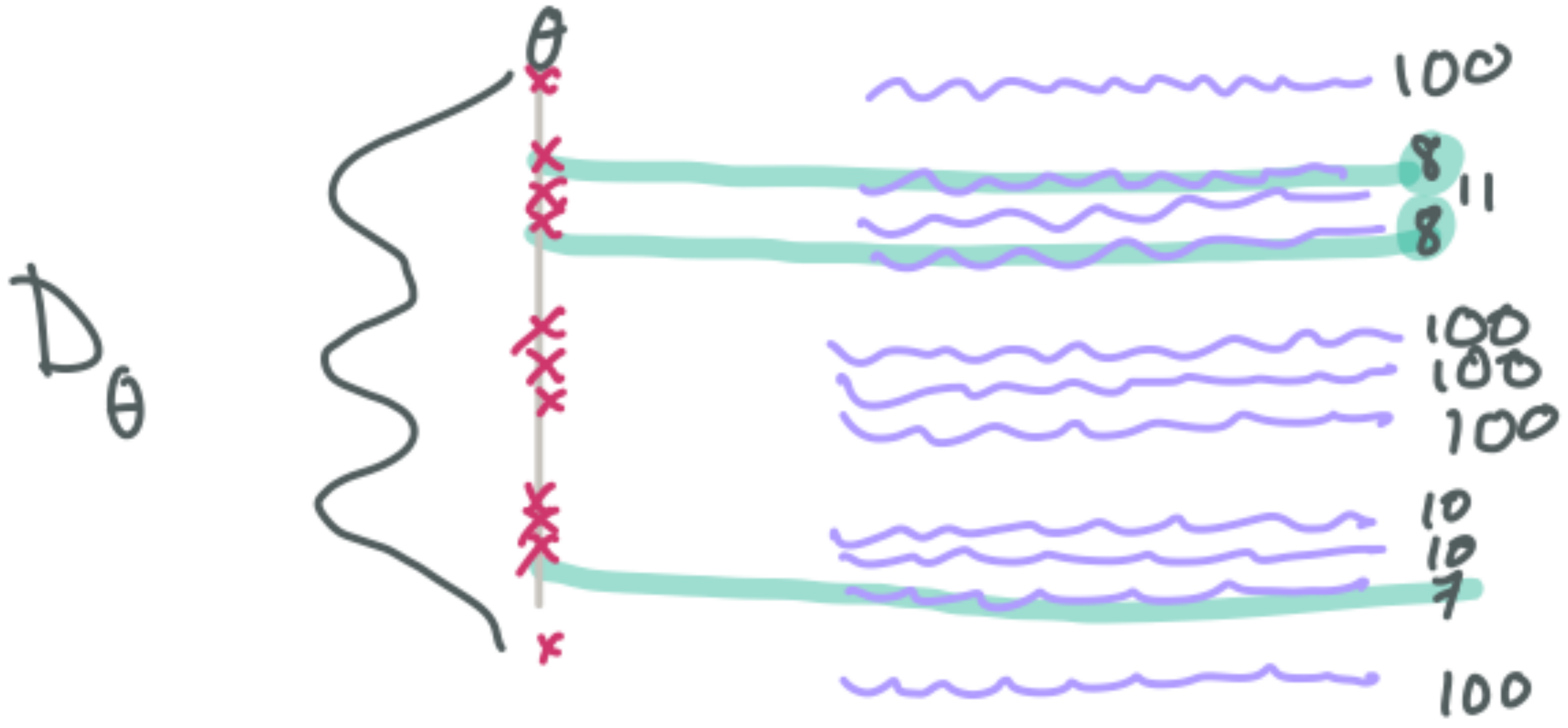
FIND TOP 'E' ELITES (e.g. 25%)

The Cross Entropy Algorithm

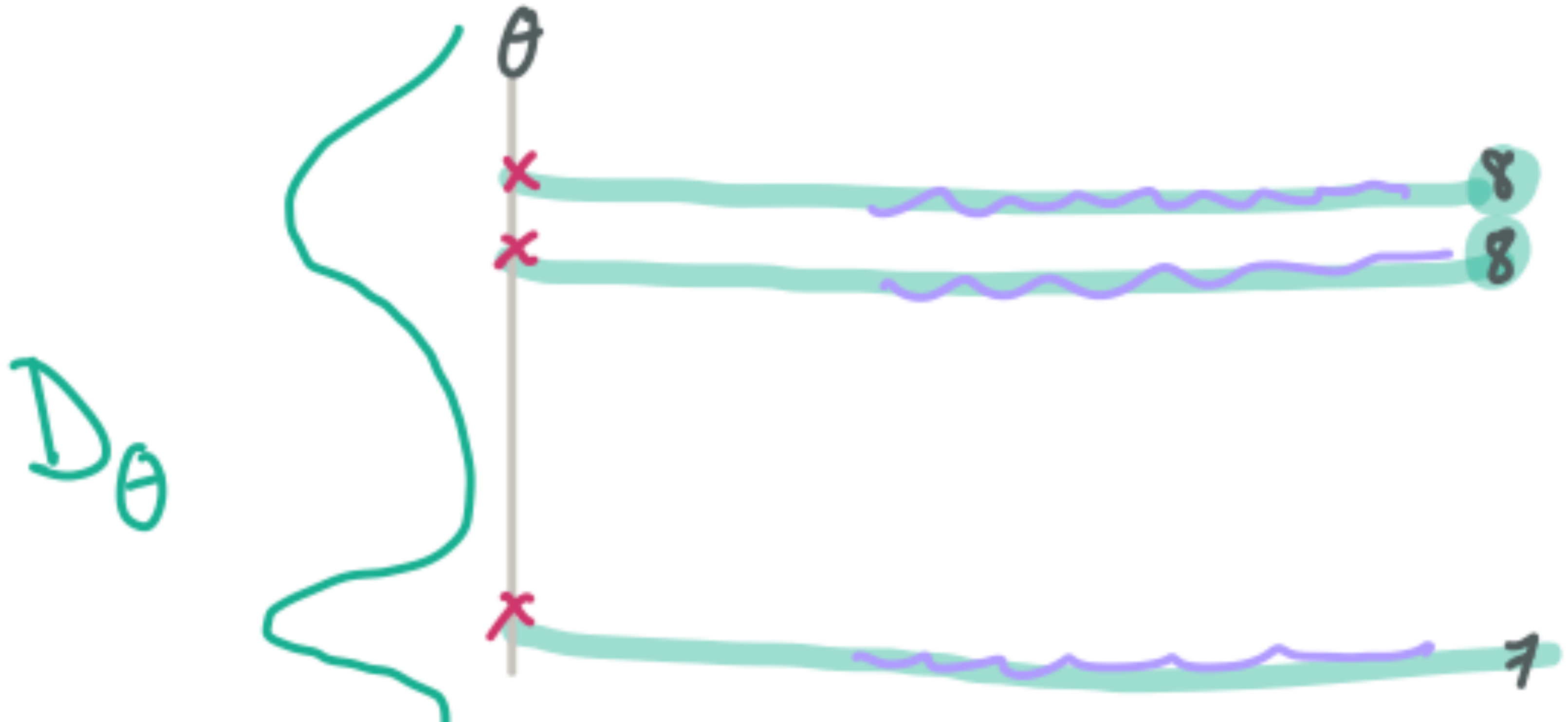


Evaluate each θ_i:

- Execute policy multiple times



Find top 'E' elites (e.g. 25%)



Fit a new distribution

D_θ

Cross Entropy for Gaussian

Gaussian Distribution $D_\theta := \mathcal{N}(\mu, \Sigma)$

Mean $$\mu^t = \frac{1}{e}\sum_{i=1}^{e}\theta_i$$

Variance $$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2$$

Does it work?

Learning Tetris Using the Noisy Cross-Entropy Method

István Szita
szityu@eotvos.elte.hu
András Lőrincz
andras.lorincz@elte.hu
Department of Information Systems, Eötvös Loránd University, Budapest, Hungary H-1117

The cross-entropy method is an efficient and general optimization algorithm. However, its applicability in reinforcement learning (RL) seems to be limited because it often converges to suboptimal policies. We apply noise for preventing early convergence of the cross-entropy method, using Tetris, a computer game, for demonstration. The resulting policy outperforms previous RL algorithms by almost two orders of magnitude.

Does it work?

	Algorithm	Grid Size	Lines Cleared	Feature Set Used
Tsitsiklis & Van Roy (1996)	Approximate value iteration	16×10	30	Holes and pile height
Bertsekas & Tsitsiklis (1996)	λ - PI	19×10	2,800	Bertsekas
Lagoudakis et al. (2002)	Least-squares PI	20×10	$\approx 2{,}000$	Lagoudakis
Kakade (2002)	Natural policy gradient	20×10	$\approx 5{,}000$	Bertsekas
Dellacherie [Reported by Fahey (2003)]	Hand tuned	20×10	660,000	Dellacherie
Ramon & Driessens (2004)	Relational RL	20×10	≈ 50	
Böhm et al. (2005)	Genetic algorithm	20×10	480,000,000 (Two Piece)	Böhm
Farias & Van Roy (2006)	Linear programming	20×10	4,274	Bertsekas
Szita & Lörincz (2006)	Cross entropy	20×10	348,895	Dellacherie
Romdhane & Lamontagne (2008)	Case-based reasoning and RL	20×10	≈ 50	
Boumaza (2009)	CMA-ES	20×10	35,000,000	BCTS
Thiery & Scherrer (2009a;b)	Cross entropy	20×10	35,000,000	DT
Gabillon et al. (2013)	Classification-based policy iteration	20×10	51,000,000	DT for policy DT + RBF for value

Practical Issues and Fixes

Problem 1: What happens to the variance?

$$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2$$

Collapses too quickly!

Simple fix: Add a bit of noise to the variance

$$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2 + \Sigma_{noise}$$

Problem 2: What if we have a bad batch of samples?

$$\mu^t = \frac{1}{e}\sum_{i=1}^{e}\theta_i$$

The elites can be bad, and the mean can slingshot into a bad value

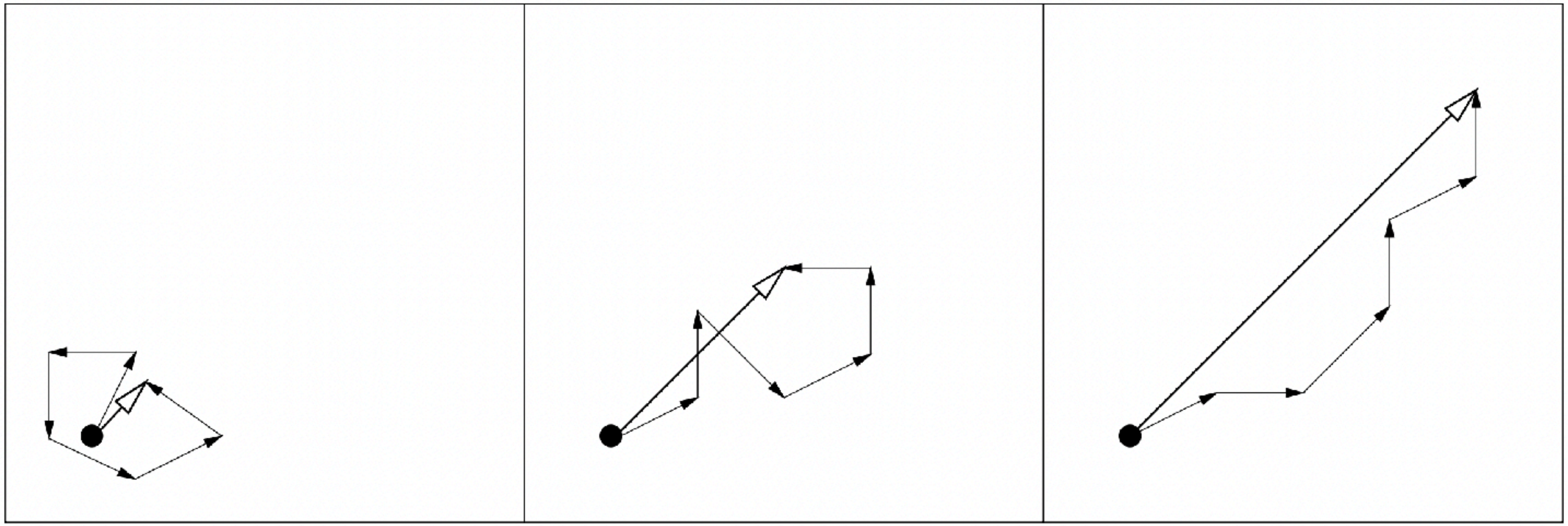
Simple fix: Slowly update mean

$$\mu^t = \mu^{t-1} + \eta\frac{1}{e}\sum_{i=1}^{e}\theta_i$$

Problem 3: What if we never converge and do random walks?

Single-steps cancel out
Use small Σ

Progress correlated
Use large Σ



A very fancy version of Cross Entropy: CMA-ES

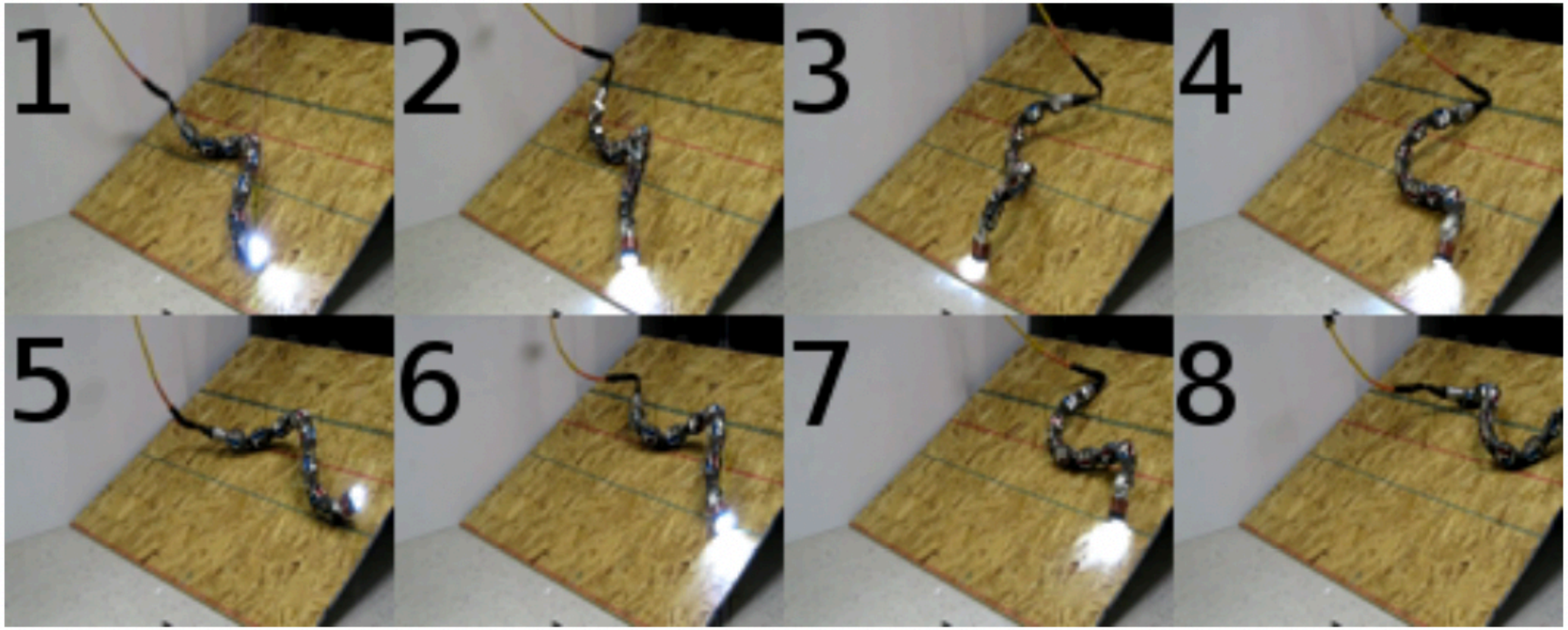
Tetris is cute...

But what about *real* robots?

Cross Entropy for Snake Robot Gaits

Using Response Surfaces and Expected Improvement to Optimize Snake Robot Gait Parameters

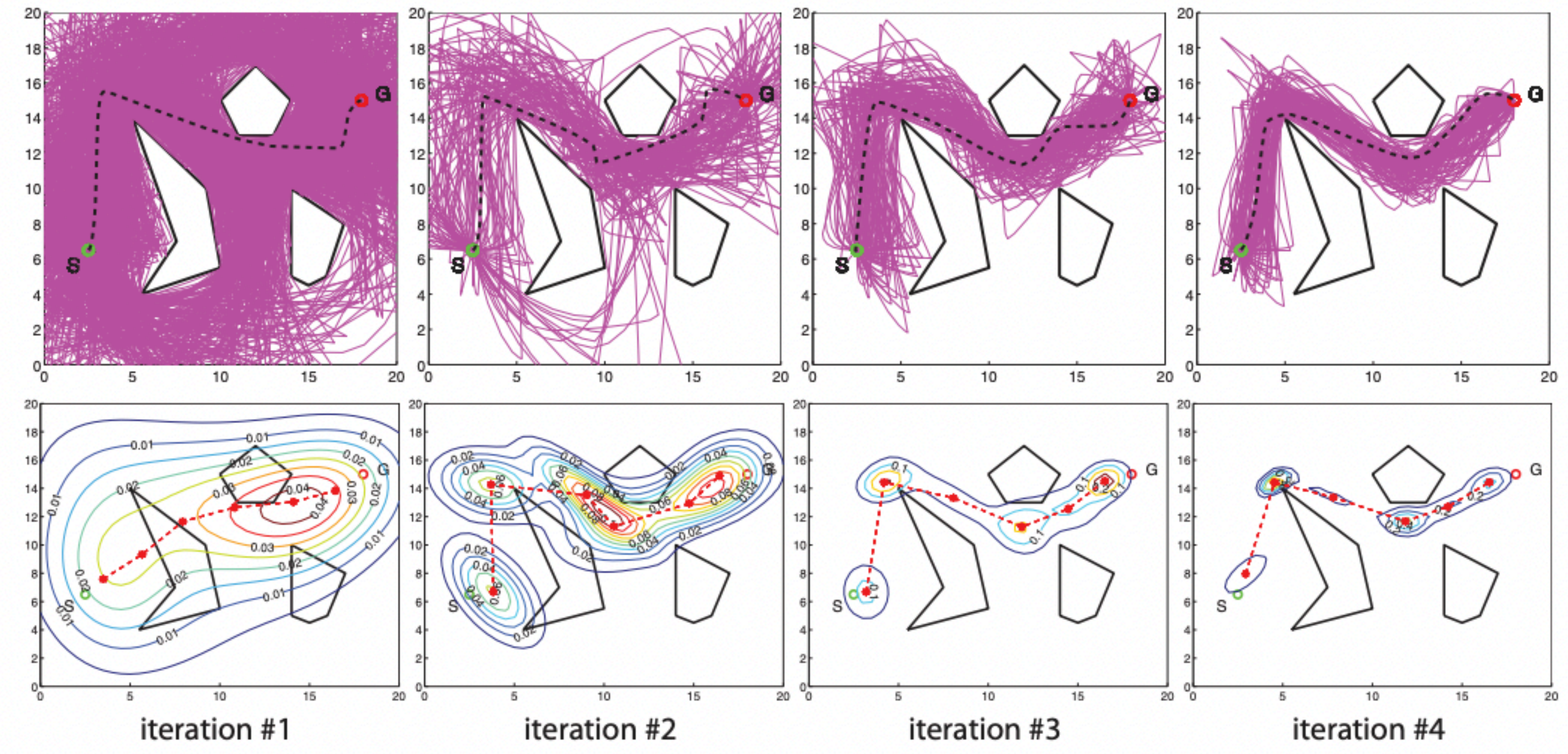
Matthew Tesch, Jeff Schneider, and Howie Choset



Uses a Gaussian Process to fit a distribution

Prove it can find the optimal gait with *minimal samples*

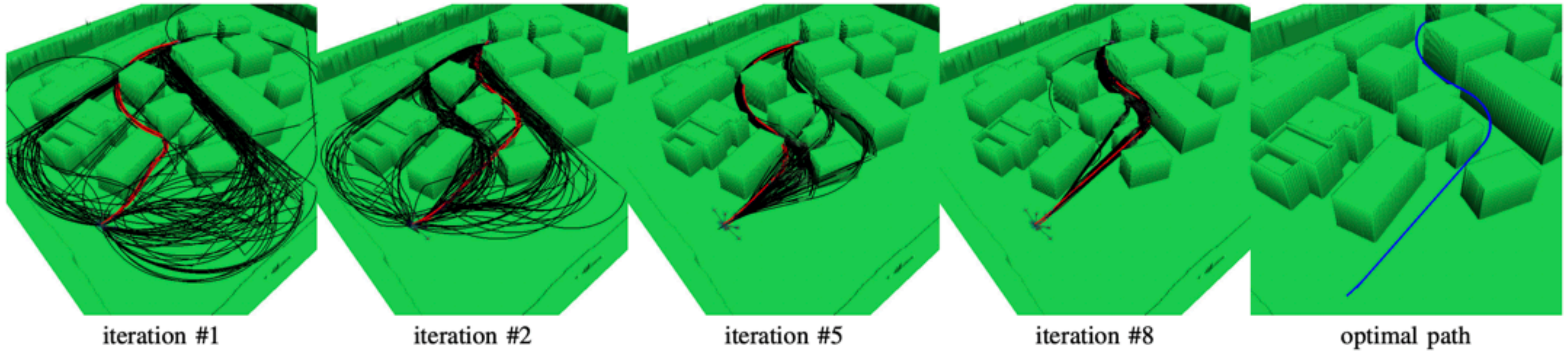
Cross Entropy Search for Motion Planning



Cross-Entropy Randomized Motion Planning

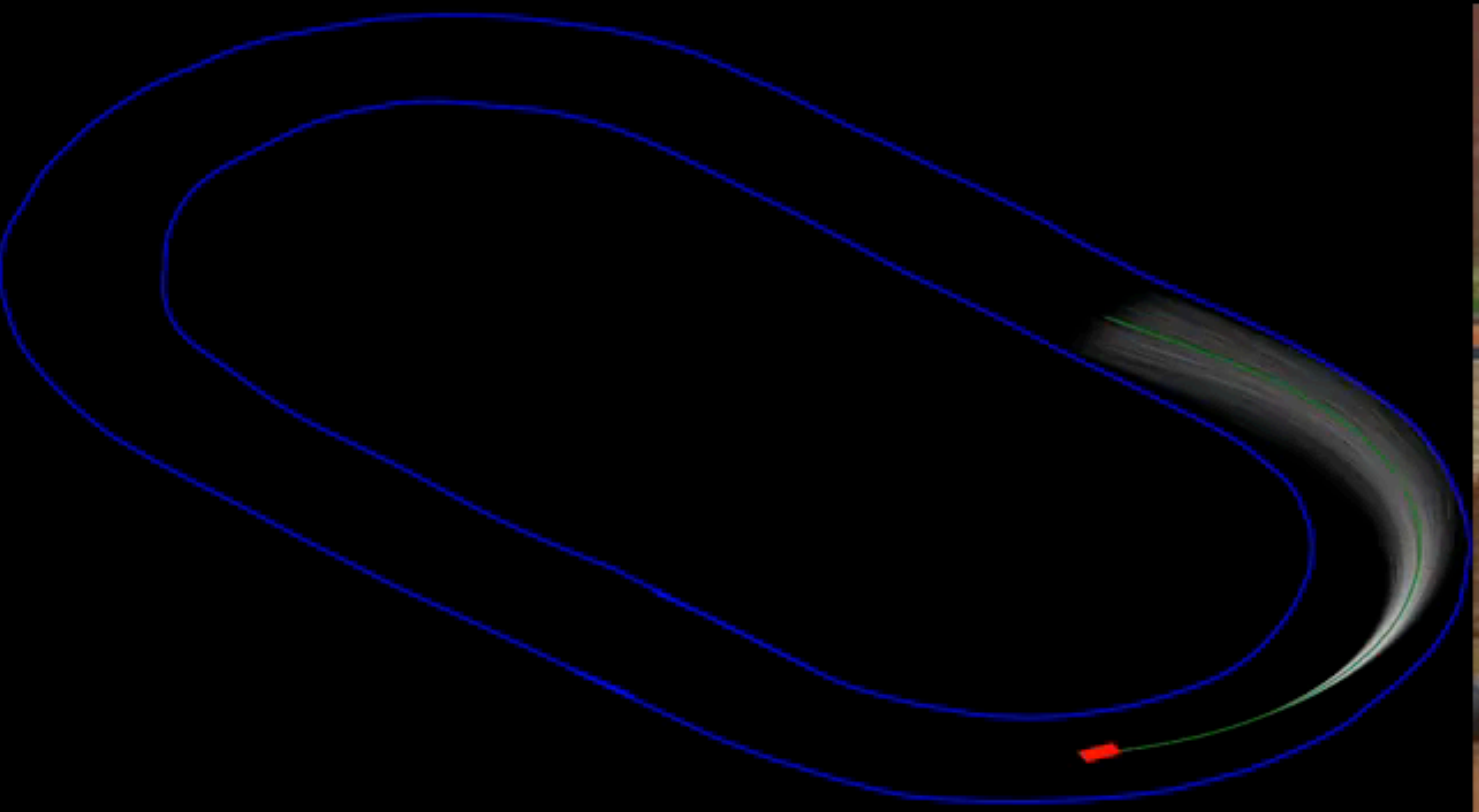
Marin Kobilarov

Distribution over control trajectories



Cross Entropy for Control

2560, 2.5 second trajectories sampled with cost-weighted average @ 60 Hz

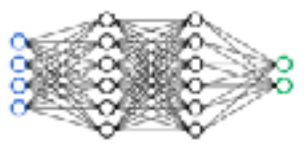


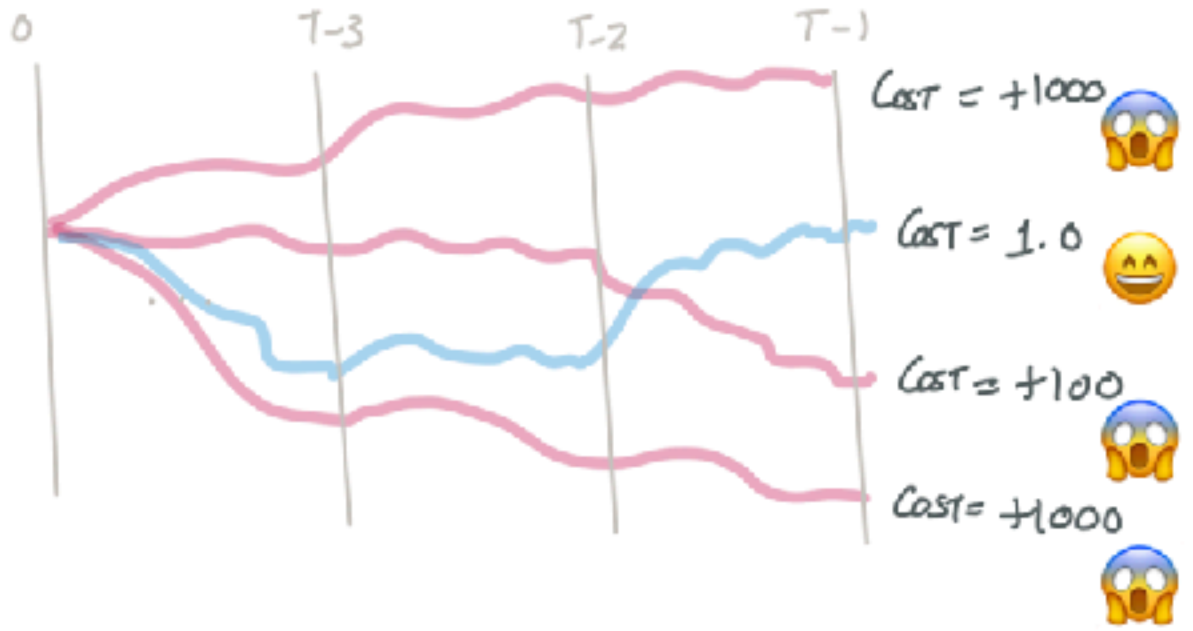
Georgia Tech Auto Rally (Byron Boots lab)

tl;dr

Can we just focus on finding a good policy?

$$\pi_\theta : s_t \rightarrow a_t$$





Learn a mapping from states to actions

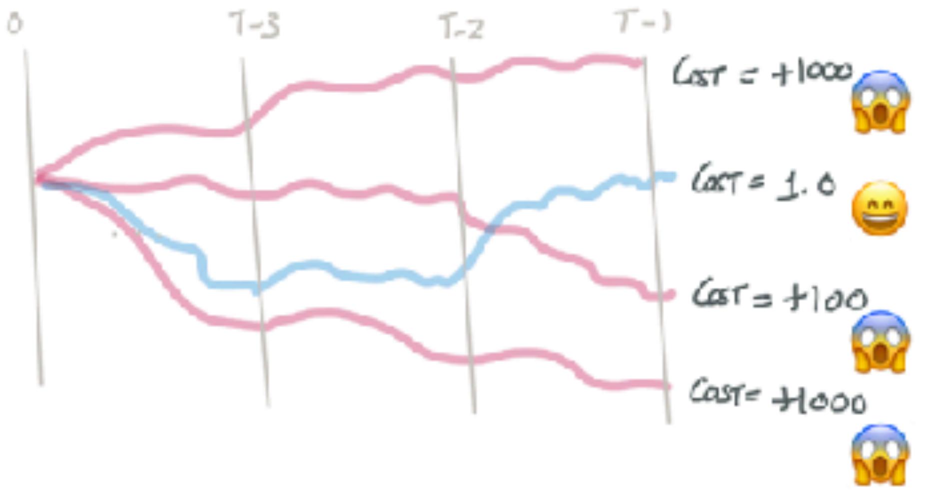
Roll-out policies in the real-world to estimate value

7

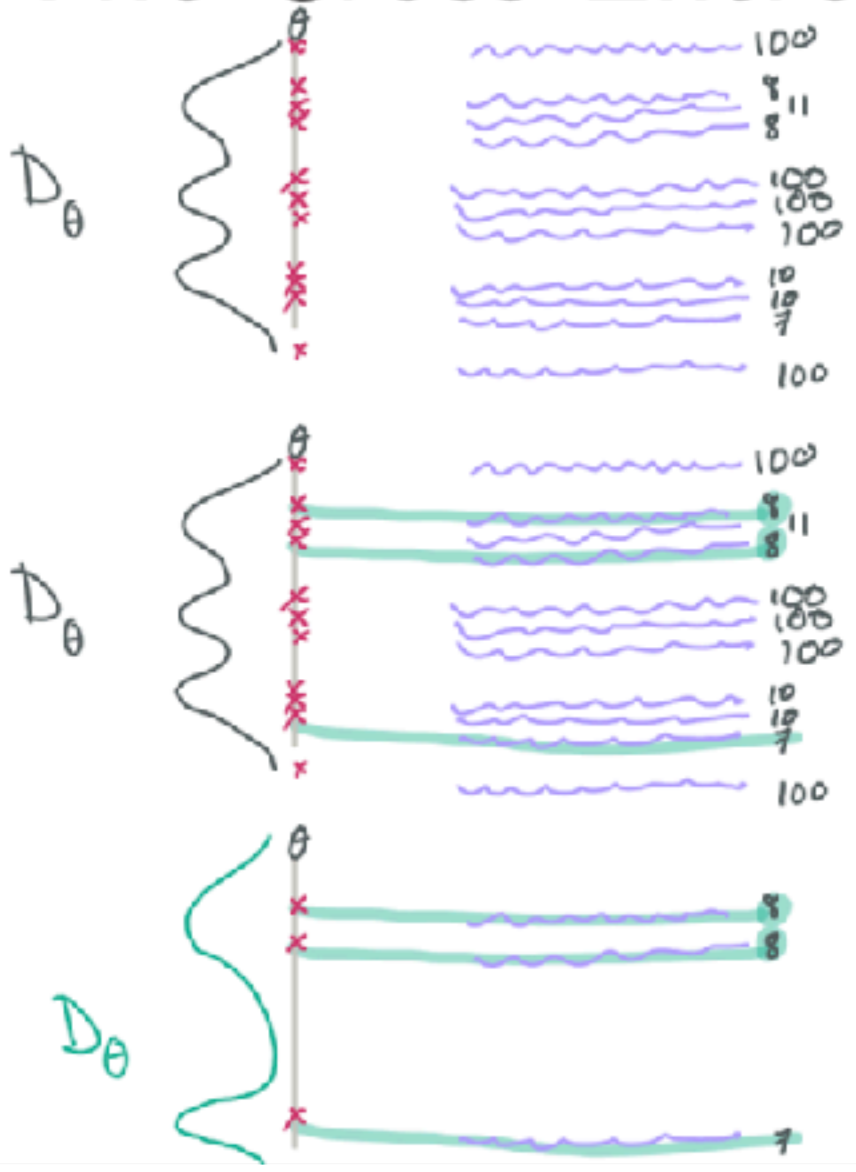
The Goal of Policy Optimization

$$\pi_\theta(s) = \arg\min_a \theta^T f(s, a)$$

$$\min_\theta J(\theta) = \sum_{t=0}^{T-1} \mathbb{E}_{\pi_\theta} c(s_t, a_t)$$



The Cross Entropy Algorithm



Evaluate each θ_i:

- Execute policy multiple times

Find top 'E' elites (e.g. 25%)

Fit a new distribution

D_θ

30