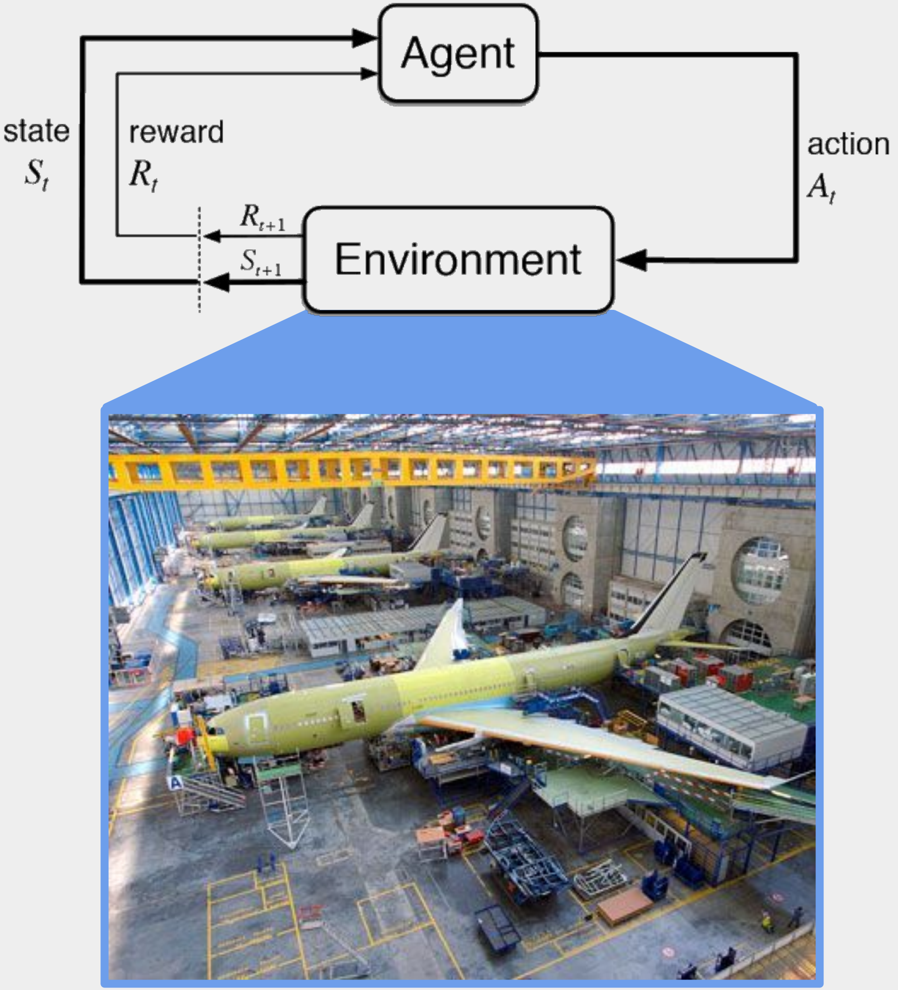
Search on the Replay Buffer

Bridging Planning & RL

Ben Eysenbach[1,2], Ruslan Salakhutdinov[1], Sergey Levine[2,3]

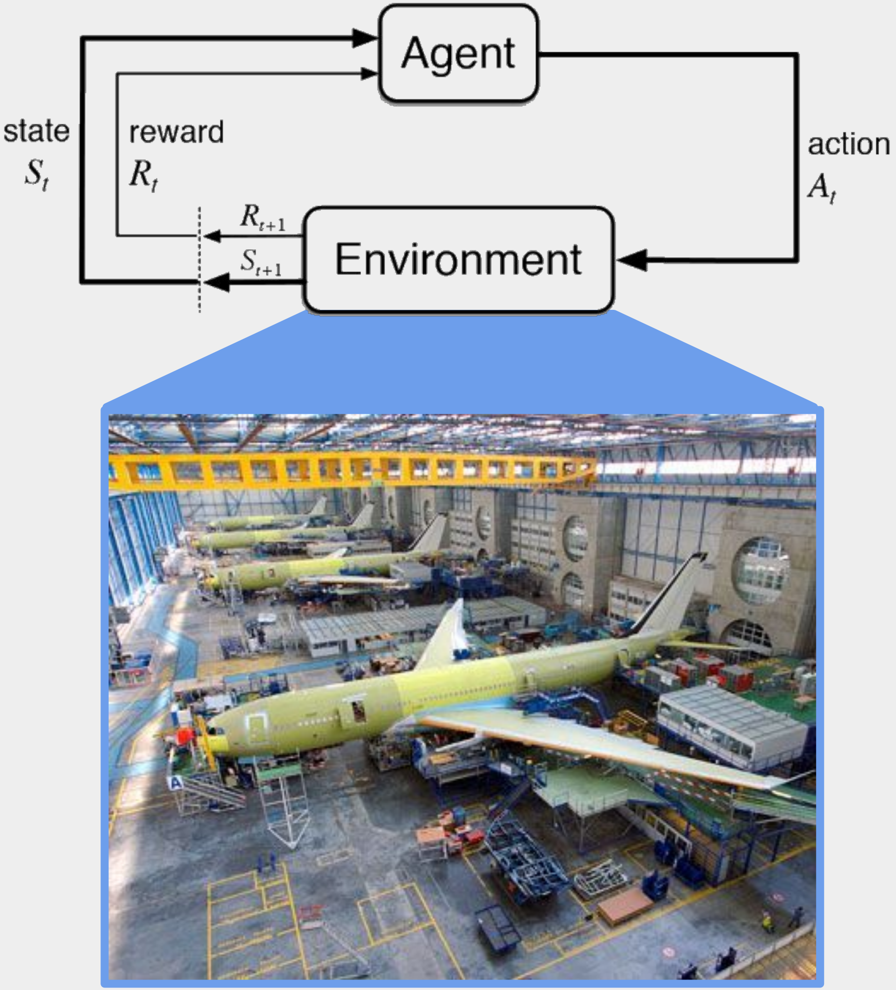
[1]CMU, [2]Google Brain, [3]UC Berkeley

Why don't we use RL to manufacture airplanes?



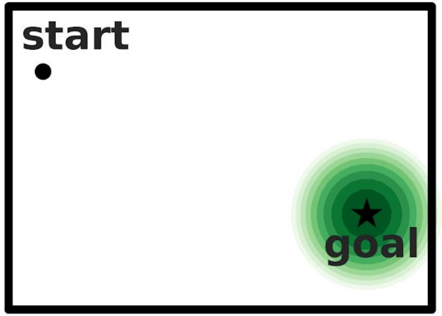
Why don't we use RL to manufacture airplanes?

- Stakes are too high to learn by trial and error.
- Instead, airplanes come together through a planning process.
- Many components might be solved with RL (e.g., painting, riveting)
- How can we meld RL with planning?



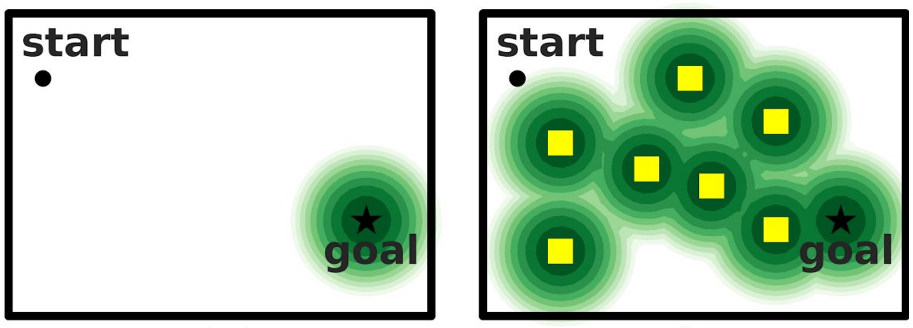
Search on the Replay Buffer

Agent wants to navigate from start to goal, but will only succeed if it starts close to the goal (green region).



(a)

Search on the Replay Buffer

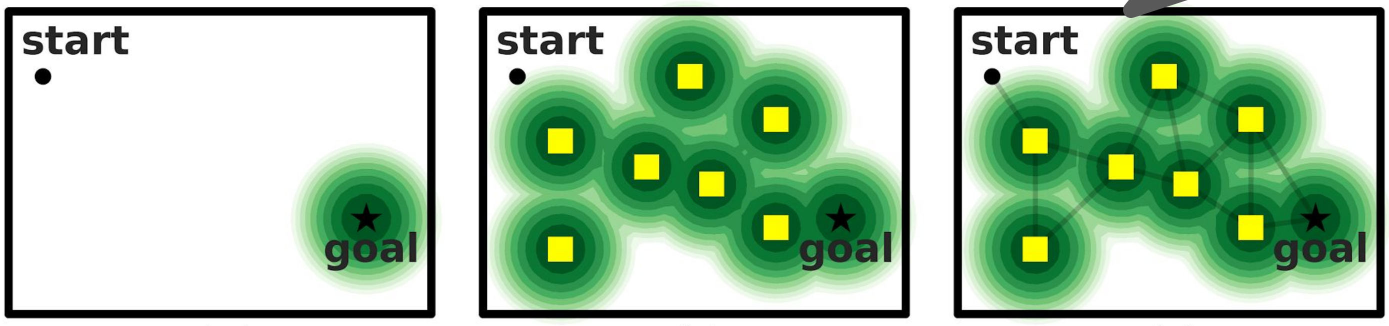


(a) (b)

Consider states that we've seen before (i.e., drawn from our replay buffer).

Search on the Replay Buffer

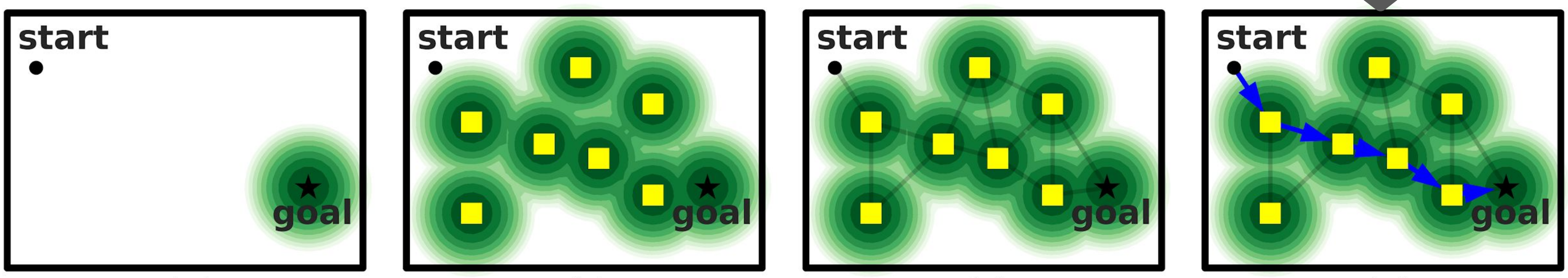
While we still cannot reach the goal state, we can reach some of the intermediate states.



(a) (b) (c)

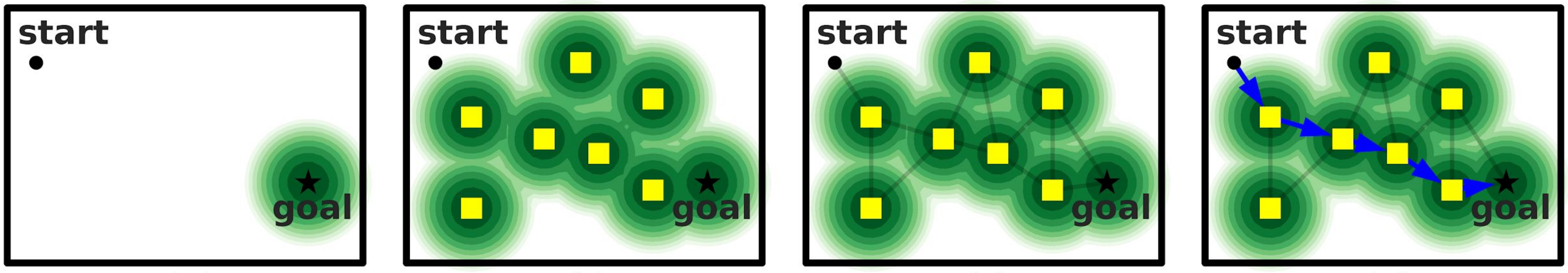
Search on the Replay Buffer

Key Idea: Build a graph on previously seen states, use shortest path algorithm to find subgoals, use policy to reach each subgoal.



(a) (b) (c) (d)

Search on the Replay Buffer



(a) (b) (c) (d)

Problems to solve:

- How to learn the policy?
- How to learn the distances?

Goal-Conditioned RL [Kaelbling 93, Schaul 15, Pong 18, Andrychowicz 17]

Define:

$$r(s, a, s_g) \triangleq -1 \qquad \gamma = 1$$

$$\text{done}(s_t, a_t, s_g) \triangleq \delta(s_t = s_g)$$

Q values correspond to negative shortest-path distance:

$$V(s, s_g) = -d_{\text{sp}}(s, s_g)$$

Goal-Conditioned RL [Kaelbling 93, Schaul 15, Pong 18, Andrychowicz 17]

How do you find the policy? ✓
How do you find the distances? ✓

Constructing a Graph

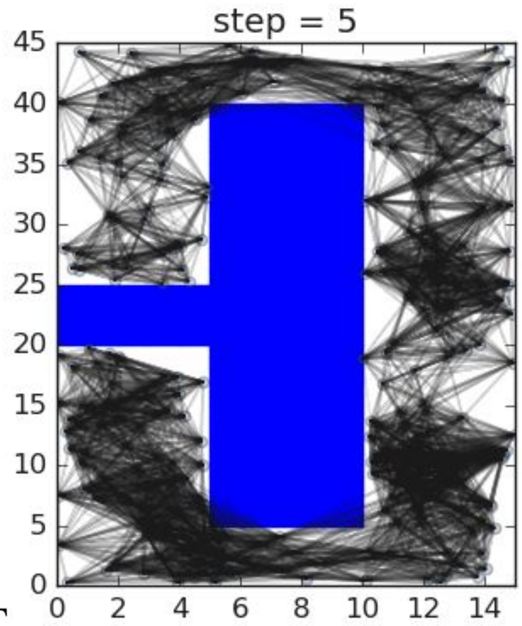
Nodes: Observations from our replay buffer.

Edges: All pairs of nodes

$$\mathcal{E} = \mathcal{B} \times \mathcal{B} = \{e_{s_1 \to s_2} \mid s_1, s_2 \in \mathcal{B}\}$$

Weights:

$$\mathcal{W}(e_{s_1 \to s_2}) = \begin{cases} d_\pi(s_1, s_2) & \text{if } d_\pi(s_1, s_2) < \textsc{MaxDist} \\ \infty & \text{otherwise} \end{cases}$$



SoRB: a policy that performs search internally.

function $\textsc{SearchPolicy}(s, s_g, \mathcal{B}, V, \pi)$

$s_{w_1}, \cdots \leftarrow \textsc{ShortestPath}(s, s_g, \mathcal{B}, V)$

$d_{s \to w_1} \leftarrow -V(s, s_{w_1})$

$d_{s \to g} \leftarrow -V(s, s_g)$

if $d_{s \to w_1} < d_{s \to g}$ or $d_{s \to g} > \textsc{MaxDist}$

$a \leftarrow \pi(a, \mid s, s_{w_1})$

else

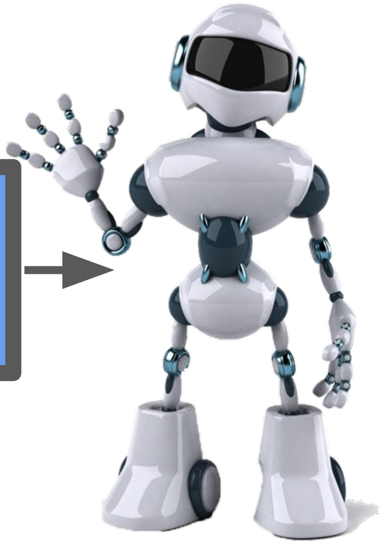
$a \leftarrow \pi(a, \mid s, s_g)$

return a

Search on the Replay Buffer

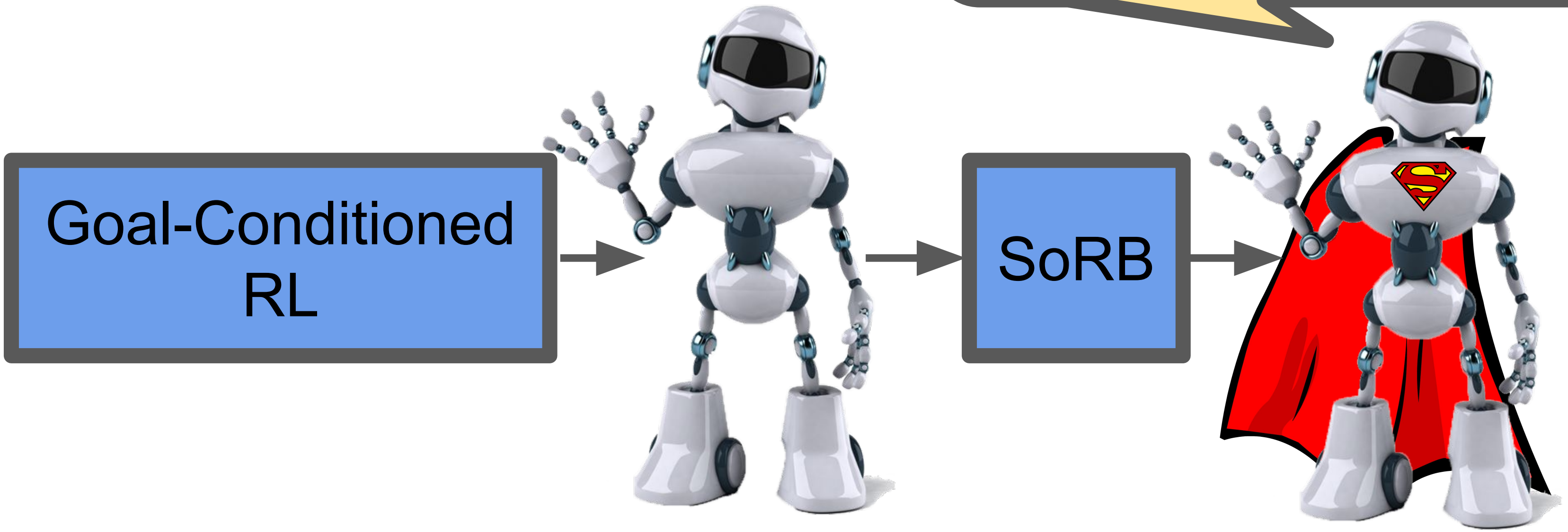
Goal-conditioned RL produces an agent that is adept at reaching some goals.

Goal-Conditioned RL

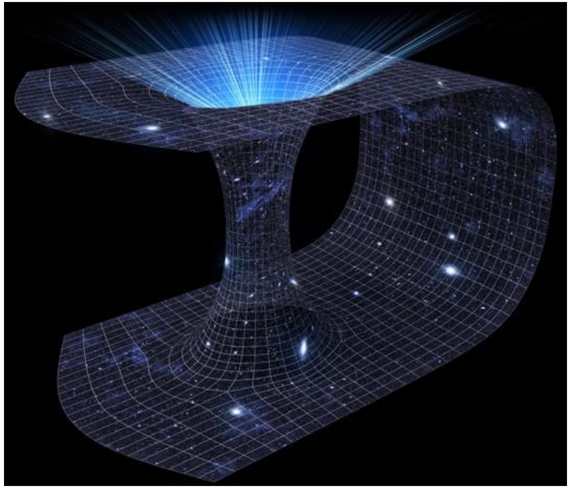


Search on the Replay Buffer

Search on the Replay Buffer is a simple trick for improving that policy, *without retraining*.



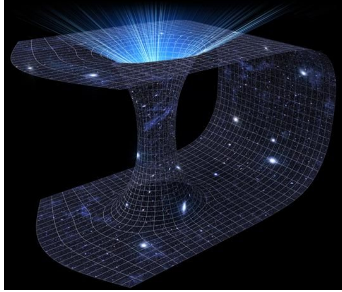
Two Problems with Distance Estimates

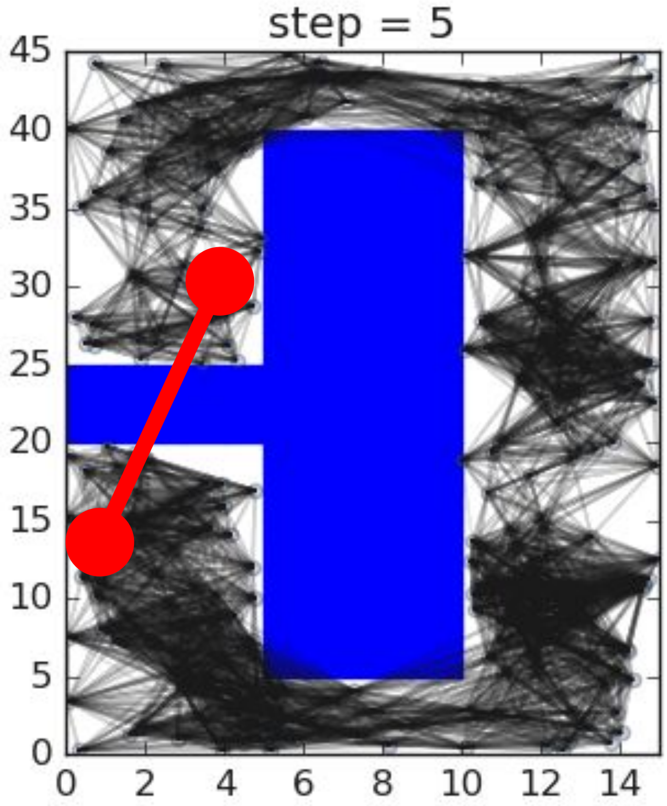


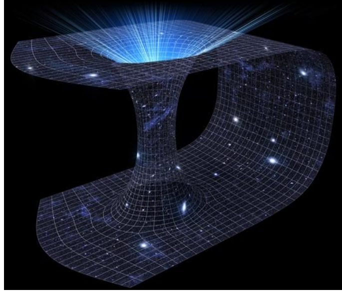
Avoiding Wormholes

Calibration

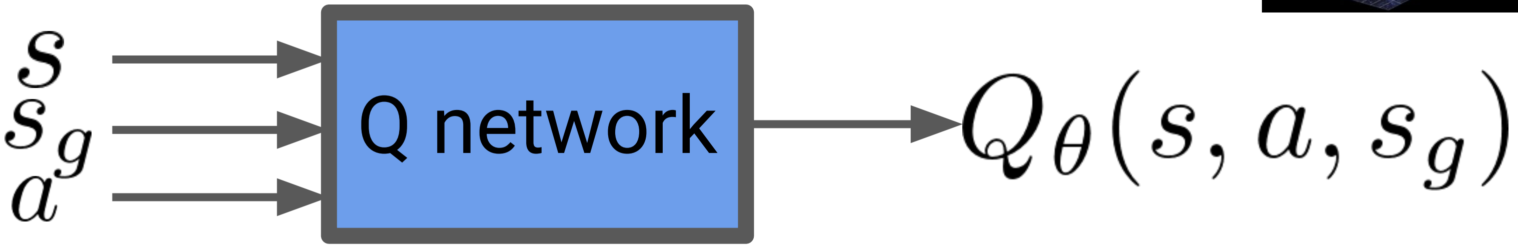
Problem 1: Avoiding Wormholes



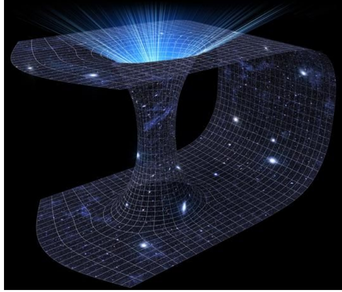


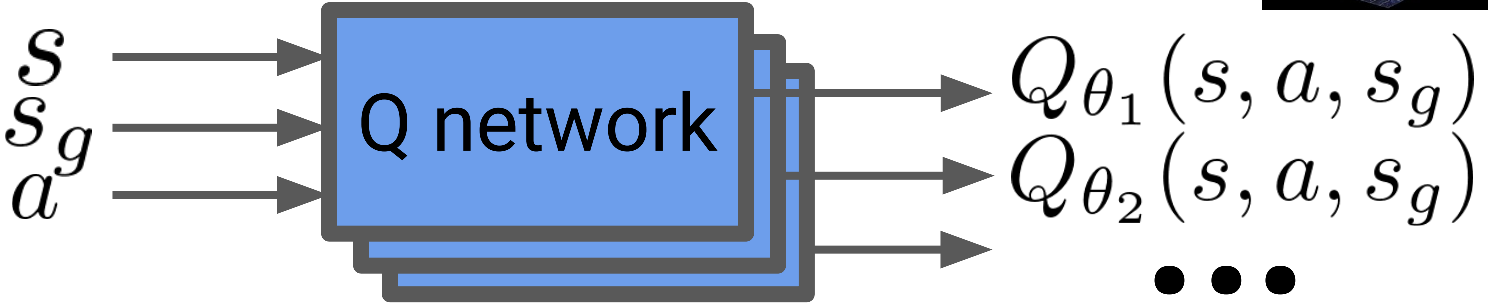


Solution to Wormholes: Ensembles



Solution to Wormholes: Ensembles





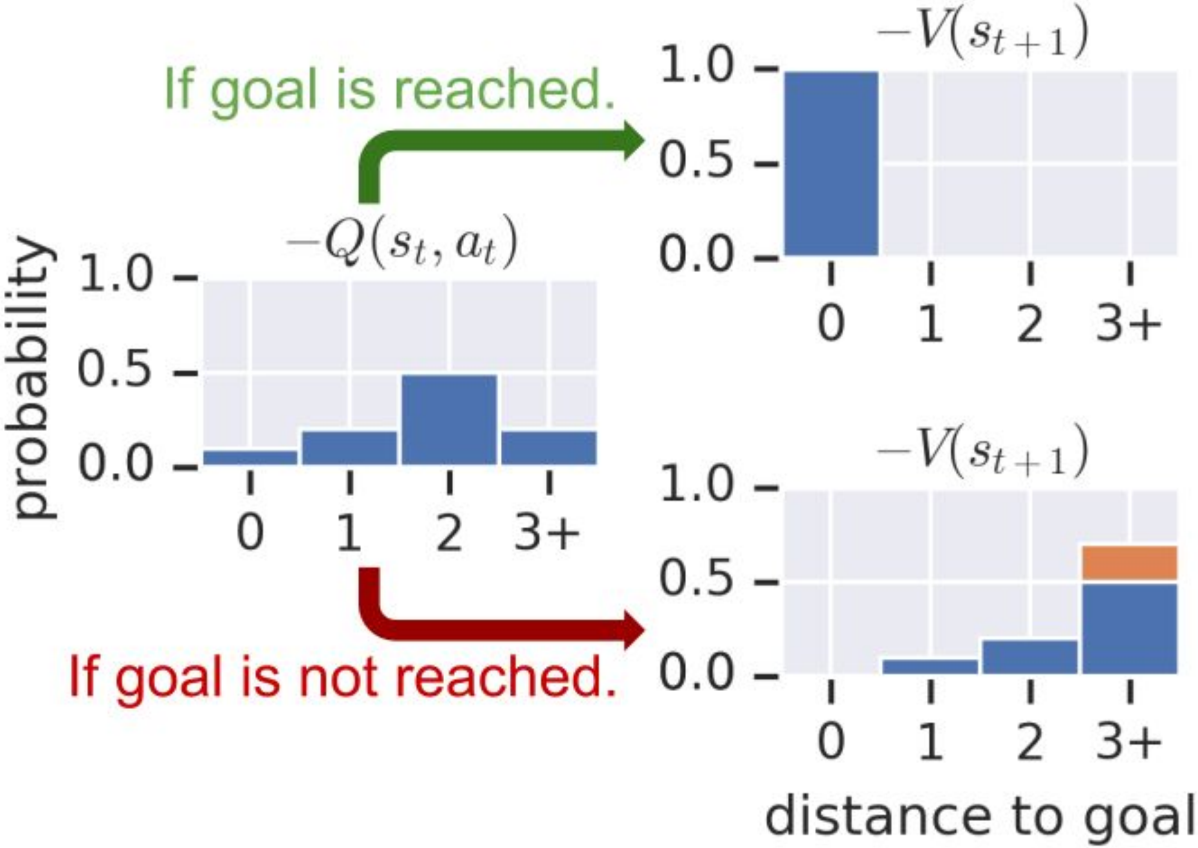
$$d(s, s_g) = \max\left(-V_{\theta_1}(s, s_g), -V_{\theta_2}(s, s_g), \cdots\right)$$

Plan using largest predicted distance (pessimistic).
We tried to use weight sharing, but predictions were too correlated.

Problem 2: Uncalibrated Distances

- Correct Q-value for an unreachable goal is $-\infty$.
- Causes Q-learning to diverge.

Solution to Calibration: Distributional RL[1]

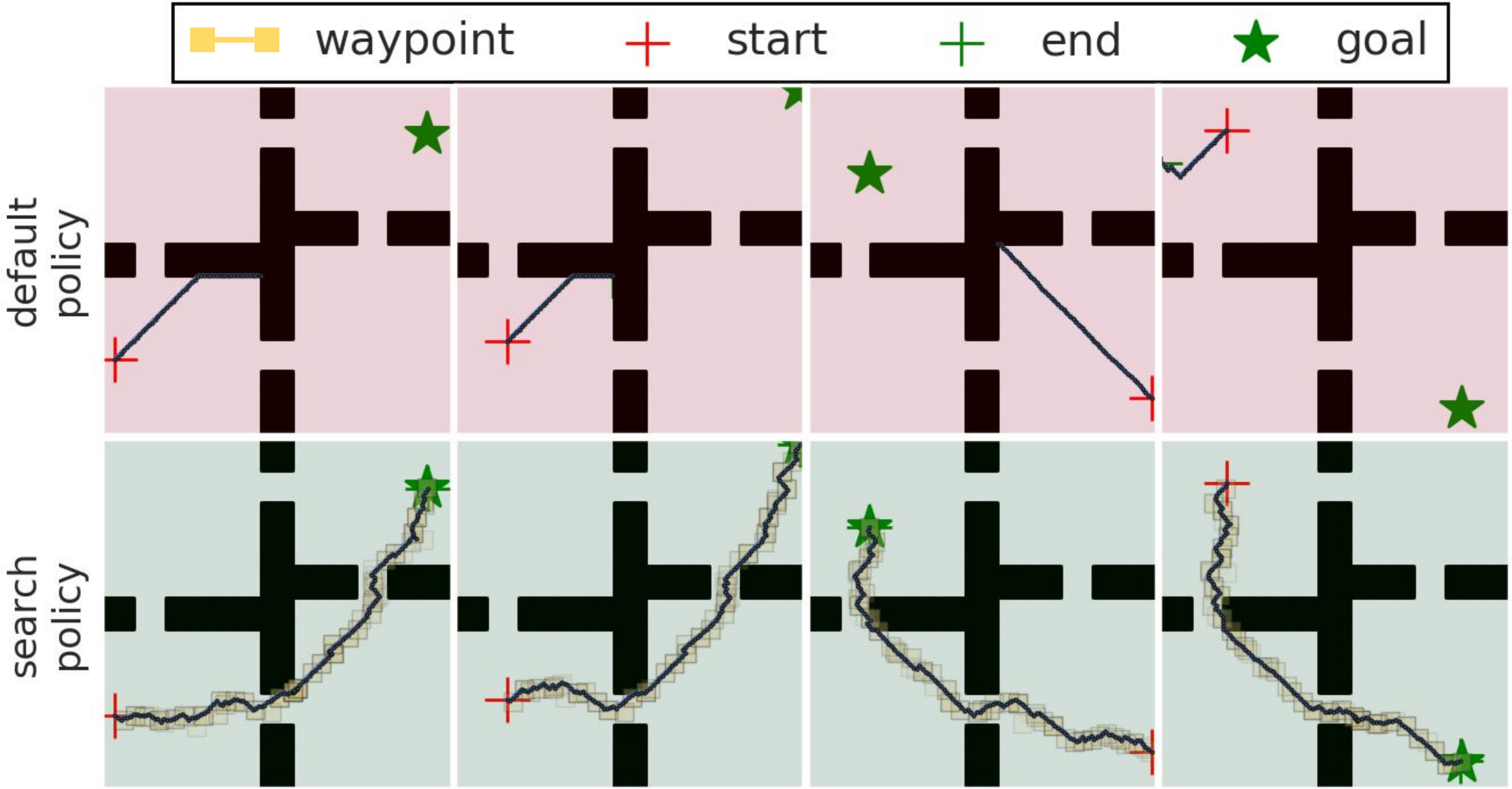


Distributional RL is simple when using sparse rewards and no discounting.

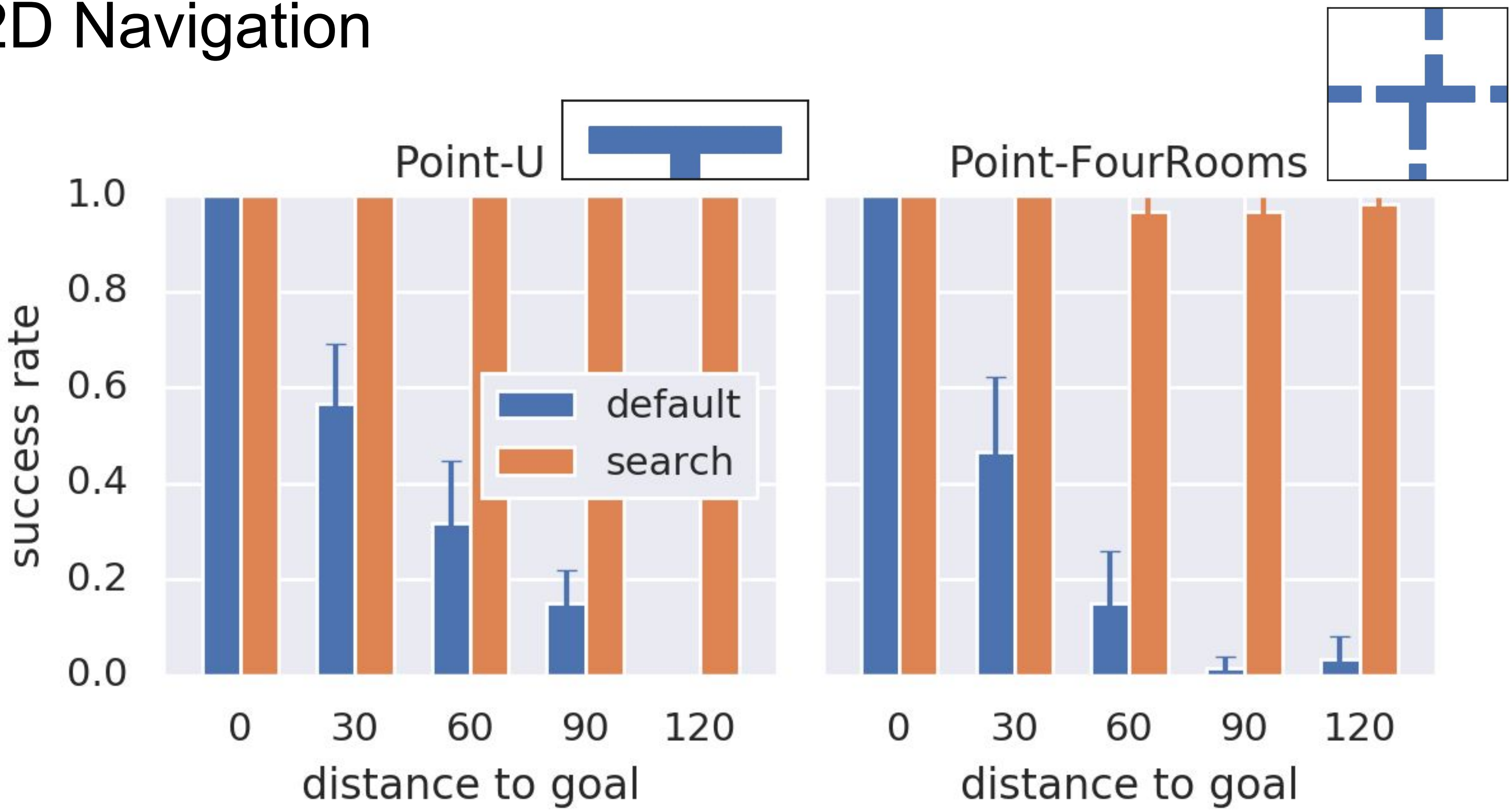
[1] Bellemare, Marc G., Will Dabney, and Rémi Munos. "A distributional perspective on reinforcement learning." ICML 2017.

Experiments

2D Navigation



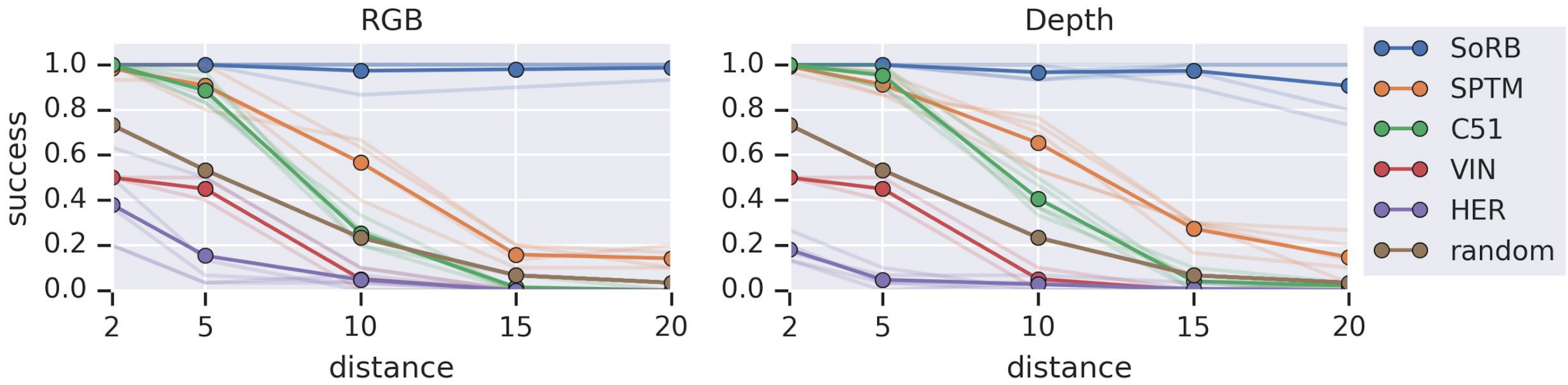
2D Navigation



Planning in Image-Space

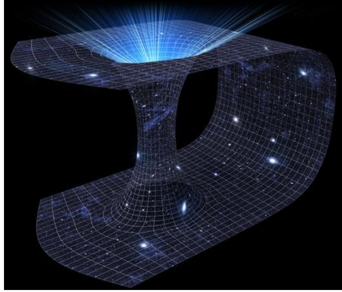
Environment from [Shah 18]

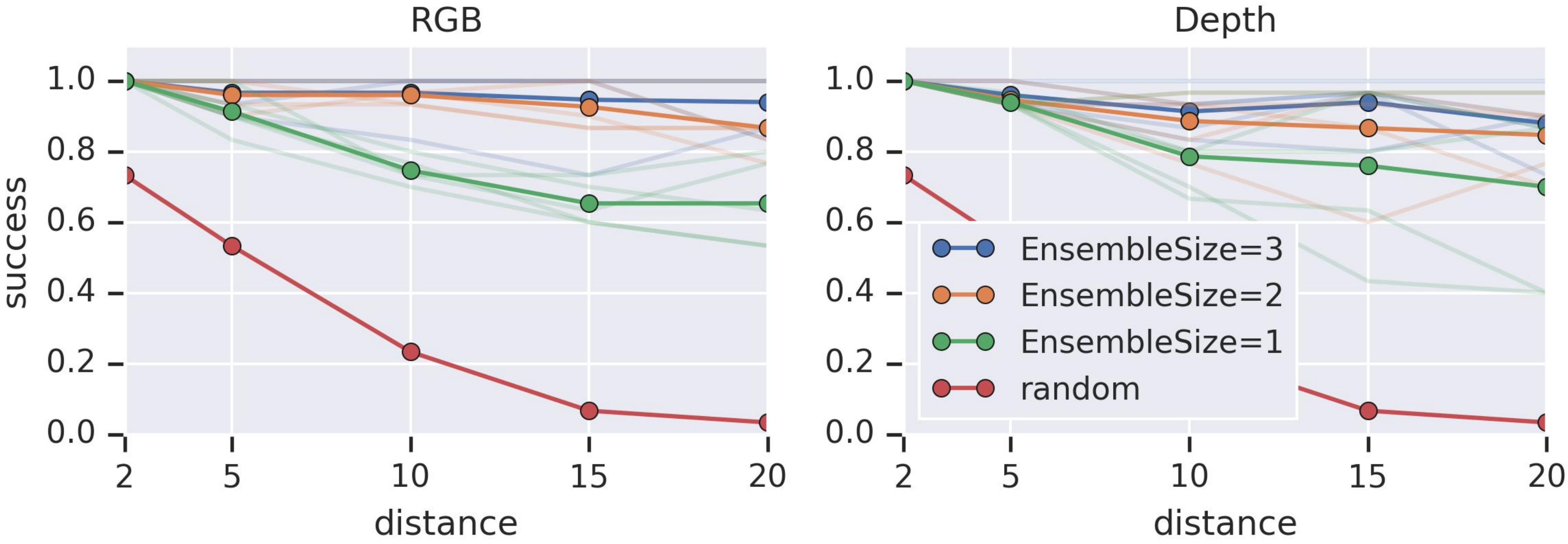
Planning in Image-Space



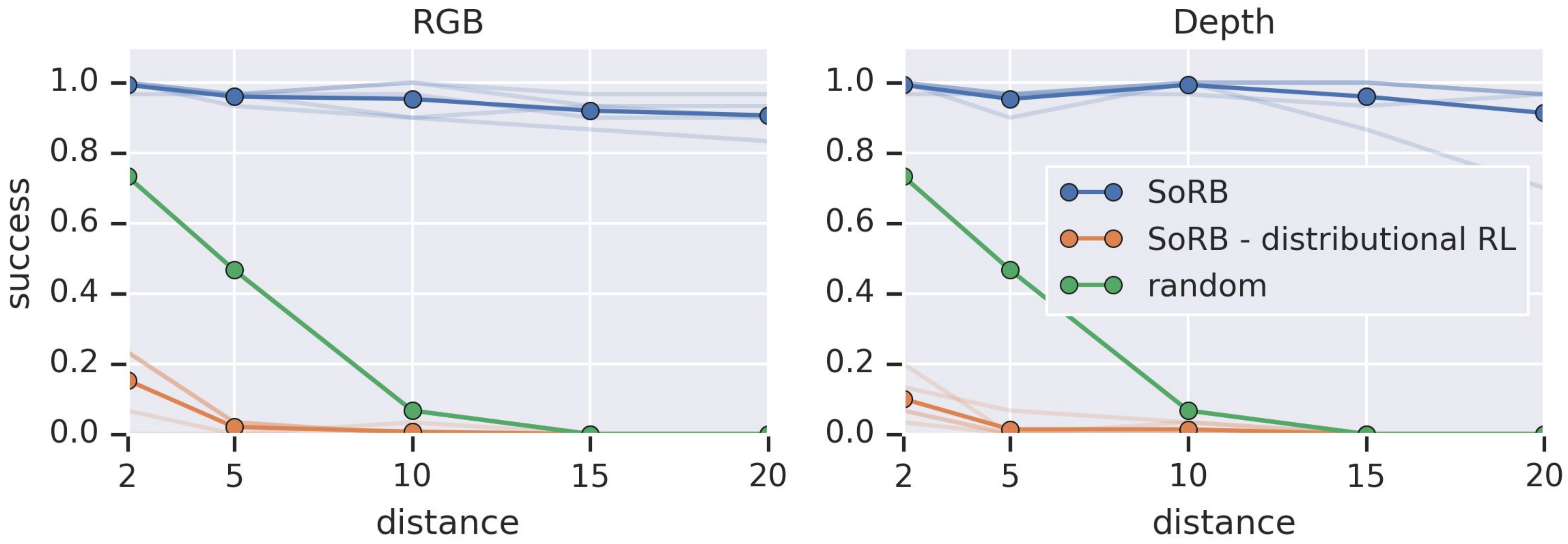
- SoRB = Search on the Replay Buffer (our method)
- SPTM = Semi-Parametric Topological Memory [Savinov 18]
- C51 = Distributional RL [Bellemare 18]
- VIN = Value Iteration Networks [Tamar 16]
- HER = Hindsight Experience Replay [Andrychowicz 17]
- Random = Random agent

Ensembles are Important





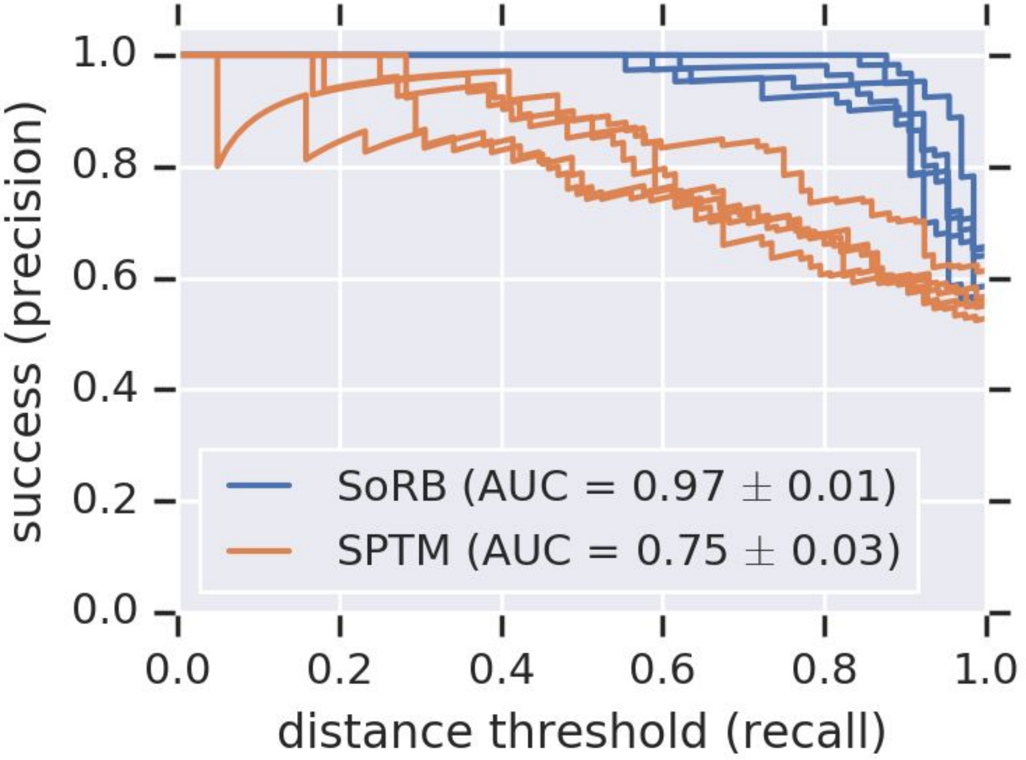
Distributional RL is Important



Just Learn Distances with Supervised Learning?

SPTM [Savinov 18]: Collect data from random policy, learn distances via supervised learning

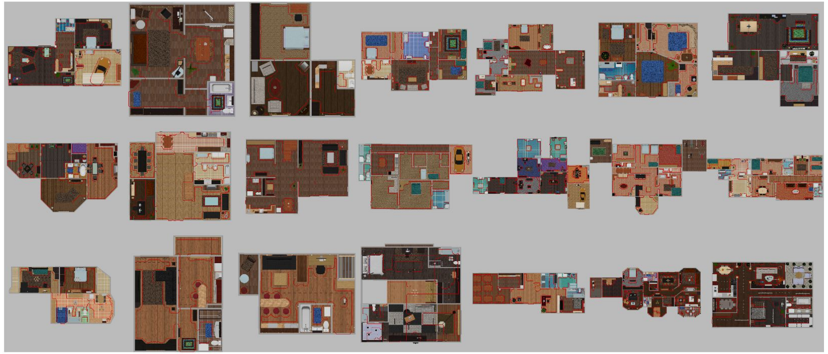
Problem: Learns distances w.r.t. random policy, but used to predict performance of non-random policy.



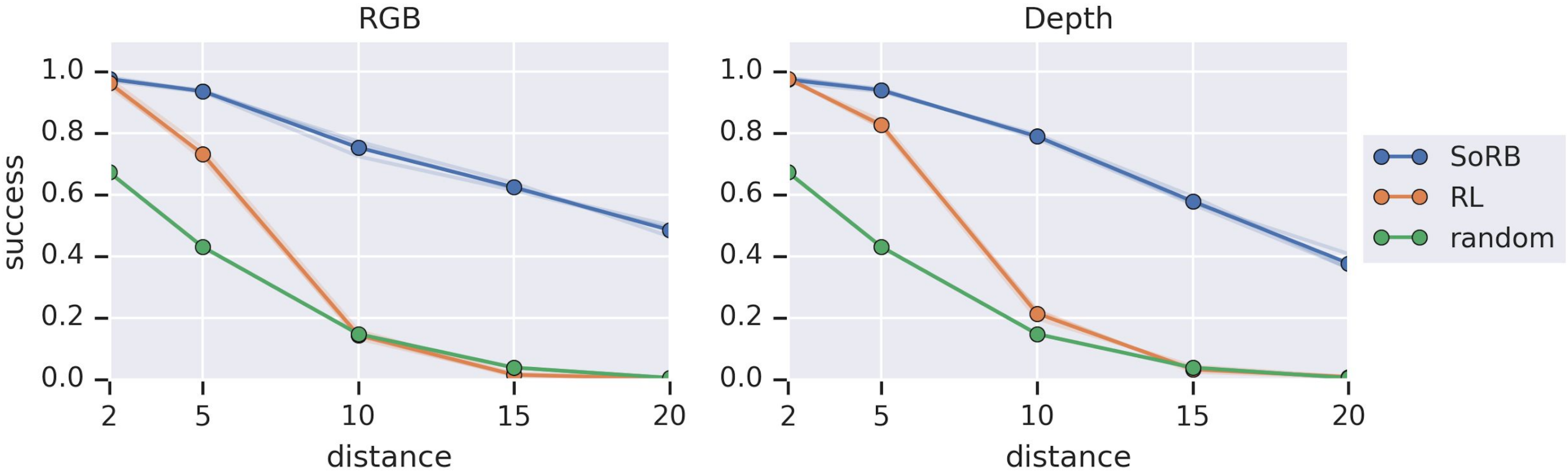
Does SoRB Generalize?

Train on many houses

Test on new houses



Does SoRB Generalize?



Relation to Prior Work

SoRB as a Planning Algorithm [Choset 05, LaValle 06, Kavraki 96, Lau 05]

- Combining RL + Planning [Chiang 19, Faust 18, Savinov 18]
- Representations for Planning [Florensa 19, Savinov 18, Wu 18, Lenz 15, Watter 15]
- Differentiable Planners [Amos 18, Lee 18, Srinivas 18, Tamar 16]
- Hierarchical RL [Bacon 17, Frans 17, Kaelbling 93, Kulkarni 16, Nachum 18, Parr 98, Precup 2000, Sutton 99, Vezhnevets 17, Drummond 02, Fox 17, Simsek 05]
- Model-Based RL [Agrawal 17, Chua 18, Finn 17, Kurutach 18, Nagabandi 18, Oh 15, Sutton 90]

Relation to Prior Work

model	real states	multi-step	prediction dimension
state-space	✓	✓	1000s+
latent-space	✗	✓	10s
inverse	✓	✗	10s
SoRB	✓	✓	1

SoRB as a Planning Algorithm [Choset 05

- Combining RL + Planning [Chiang 19
- Representations for Planning [Florensa 19, Savinov 18, Wu 18, Lenz 15, Watter 15]
- Differentiable Planners [Amos 18, Lee 18, Srinivas 18, Tamar 16]
- Hierarchical RL [Bacon 17, Frans 17, Kaelbling 93, Kulkarni 16, Nachum 18, Parr 98, Precup 2000, Sutton 99, Vezhnevets 17, Drummond 02, Fox 17, Simsek 05]
- Model-Based RL [Agrawal 17, Chua 18, Finn 17, Kurutach 18, Nagabandi 18, Oh 15, Sutton 90]

Takeaways & Open Problems

Takeaways

- Planning is useful for many real-world problems.
- Goal-Conditioned RL works well locally.
- Graph search is a tool to boost performance of goal-conditioned agent.
- Distributional RL + Ensembles provide robust distance estimates.

Open Problems

- How to incorporate planning into policy search?
- SoRB as an inference procedure?
- Better generalization?
- What states to use for planning?

Run SoRB in your browser!
http://bit.ly/rl_search