NEW YORK UNIVERSITY

Facebook AI Research

Intrinsic Motivation and Automatic Curricula via Asymmetric Self-Play

Sainbayar Sukhbaatar

New York University

Joint work with:

Arthur Szlam

Facebook

AI Research

Rob Fergus

Facebook

AI Research

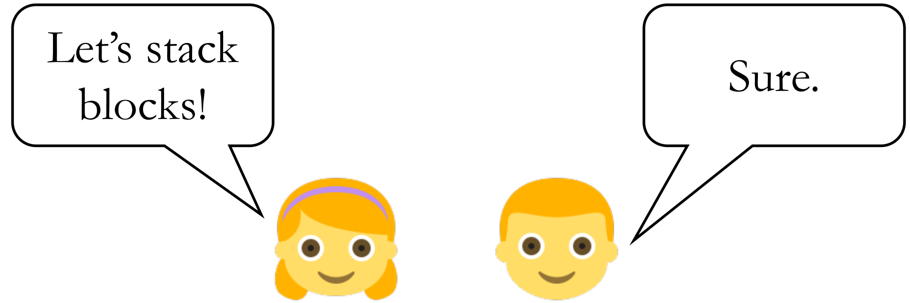
& NYU

Motivation

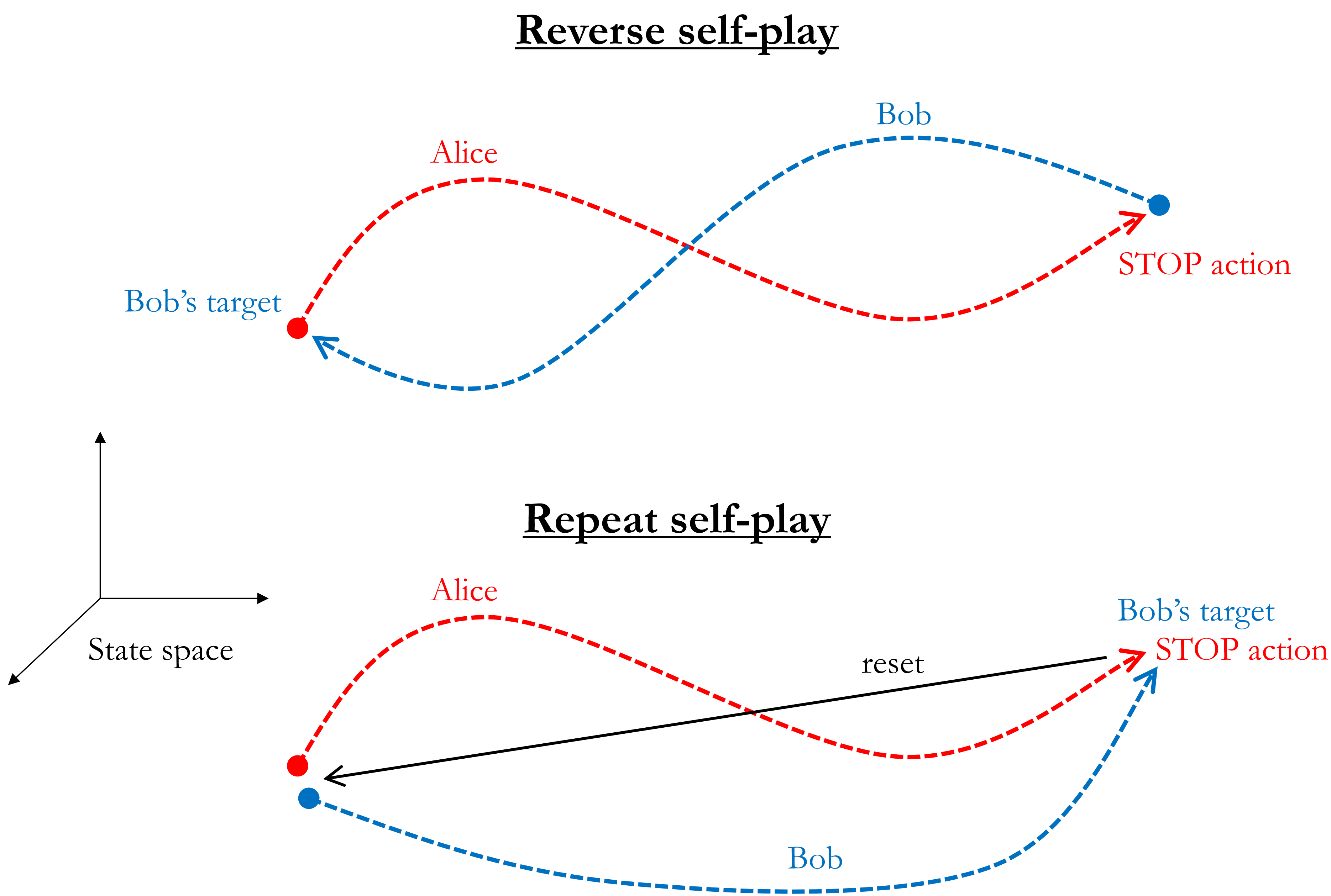
- Reinforcement Learning (RL) typically requires a **huge** number of episodes
- Often **supervision** signal (i.e. reward) is **expensive** to obtain

- Can we learn about environment in **unsupervised** way?
- Assumption: interaction with the environment is cheap

Approach



- Agent plays a **game** where it challenges itself
- **Single** physical agent, but **two** separate minds:
  - Alice's job is to **propose** a task
  - Bob's job is to **complete** that task
- Alice propose a task by actually **doing** it
- We consider two classes of environments:
  1. Actions are reversible within same time → **reverse** self-play
  2. Reset to the initial state is allowed → **repeat** self-play
- Jointly train with **self-play** and **target task**
  - Randomly choose type of episode



Reverse self-play
Alice
Bob
Bob's target
STOP action
Repeat self-play
Alice
Bob's target
STOP action
reset
Bob
State space

Internal reward during self-play

- Bob's reward:

$$R_b = -t_b$$

Time spent

If Bob fails: $t_b = t_{\max}$

Alice's reward:

$$R_a = \max(0, t_b - t_a)$$

Intuition: make Bob fail with less effort

- Alice's optimal behavior is to find simplest tasks that Bob cannot complete.
- Makes learning for Bob easy since the new task will be only just beyond his current capabilities.
- Gives self-regulating feedback between Alice and Bob
  - Yields **automatic curriculum**

Parameterizing Policy Functions

- Self-play: $a_{\text{Alice}} = f_A(s_t, s_0)$ $a_{\text{Bob}} = f_B(s'_t, s'_0)$

  Initial state Target state

- Target task: $a_{\text{Target}} = f_B(s''_t, e)$

  task description (dummy vector)

- Self-play lets Bob build representation of environment
- Assumption: self-play tasks are close to target task
- Explore discrete / continuous settings
  - Using small NN for f(.)

Self-play equilibrium & Universal Bob

- Claim: Under some strong assumptions (tabular policies, finite state, etc.), Bob must learn all possible tasks, i.e. learn how to transition between any pair of states as efficiently as possible.
- Let's assume the self-play has converged to a Nash equilibrium (can't gain anything if other's policy is fixed)
- If Bob fails on a certain task, then Alice would propose that task to increase her reward
- Then Bob must've seen this task and learnt it to increase his reward
- Thus: Bob must have learned all possible tasks.

Related work

- **Self-play:** checkers (Samuel, 1959), backgammon (Tesauro, 1995), and Go, (Silver et al., 2016), and RoboSoccer (Riedmiller et al., 2009)
  - Uses external reward vs internal reward for ours
- **GANs** (Goodfellow et al., 2014): dialogue generation (Li et al., 2017), variational auto-encoders (Mescheder et al., 2017)
  - Alice → "generator" of hard examples; Bob → "discriminator"
- **Intrinsic motivation** (Barto, 2013; Singh et al., 2004; Klyubin et al., 2005; Schmidhuber, 1991): curiosity-driven exploration (Schmidhuber, 1991; Bellemare et al., 2016; Strehl & Littman, 2008; Lopes et al., 2012; Tang et al., 2016)
  - Reward for novelty of state
  - Ours: learning to transition between pairs of states
- **Robust Adversarial Reinforcement Learning** (Pinto et al. 2017)
  - Concurrent work; adversarial peturbations to state

Experiments

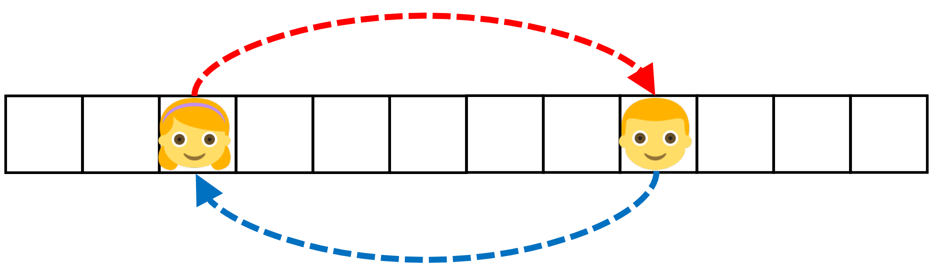
- Use **Reinforce** algorithm with learnt baseline and entropy regularization
- 2-layer NN model for Alice and Bob (separate)
- Train on 20% target task + 80% self-play episodes
- Discrete and continuous environments
- Measure target task reward vs # target task episodes
  - Self-play episodes are "free"

- Baselines:
  - No self-play: just target task episodes
  - Random Alice: Alice takes random actions. Bob learns policy
  - Exploration approaches: count-based & variants

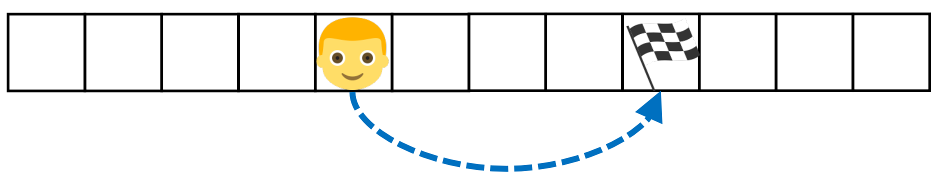
Toy example: Long hallway

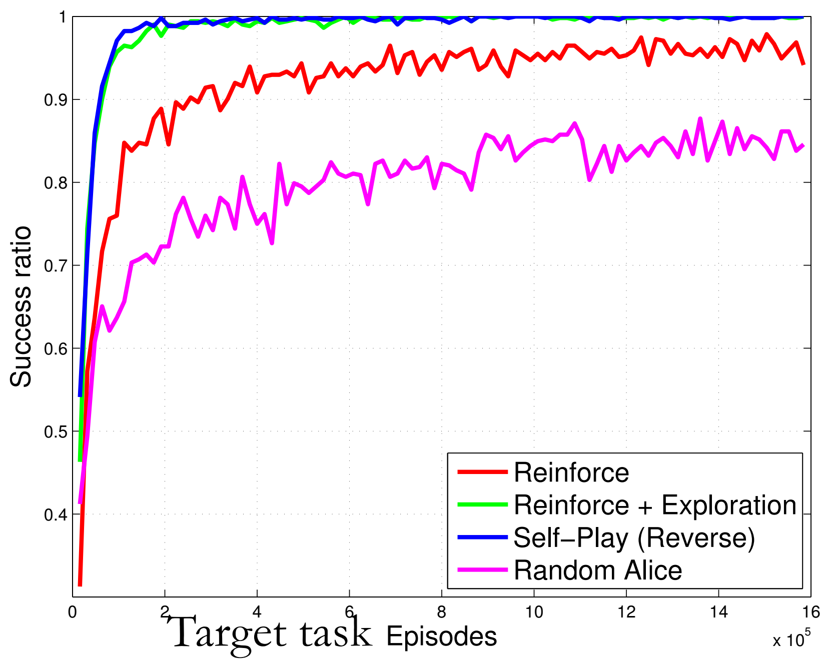
- Learn to navigate in a long corridor
- Reverse self-play
- Simple tabular policies

Self-play



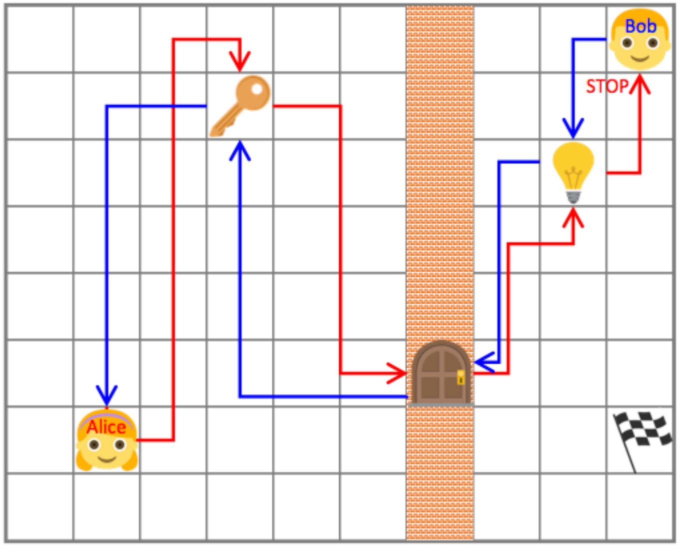
Target task





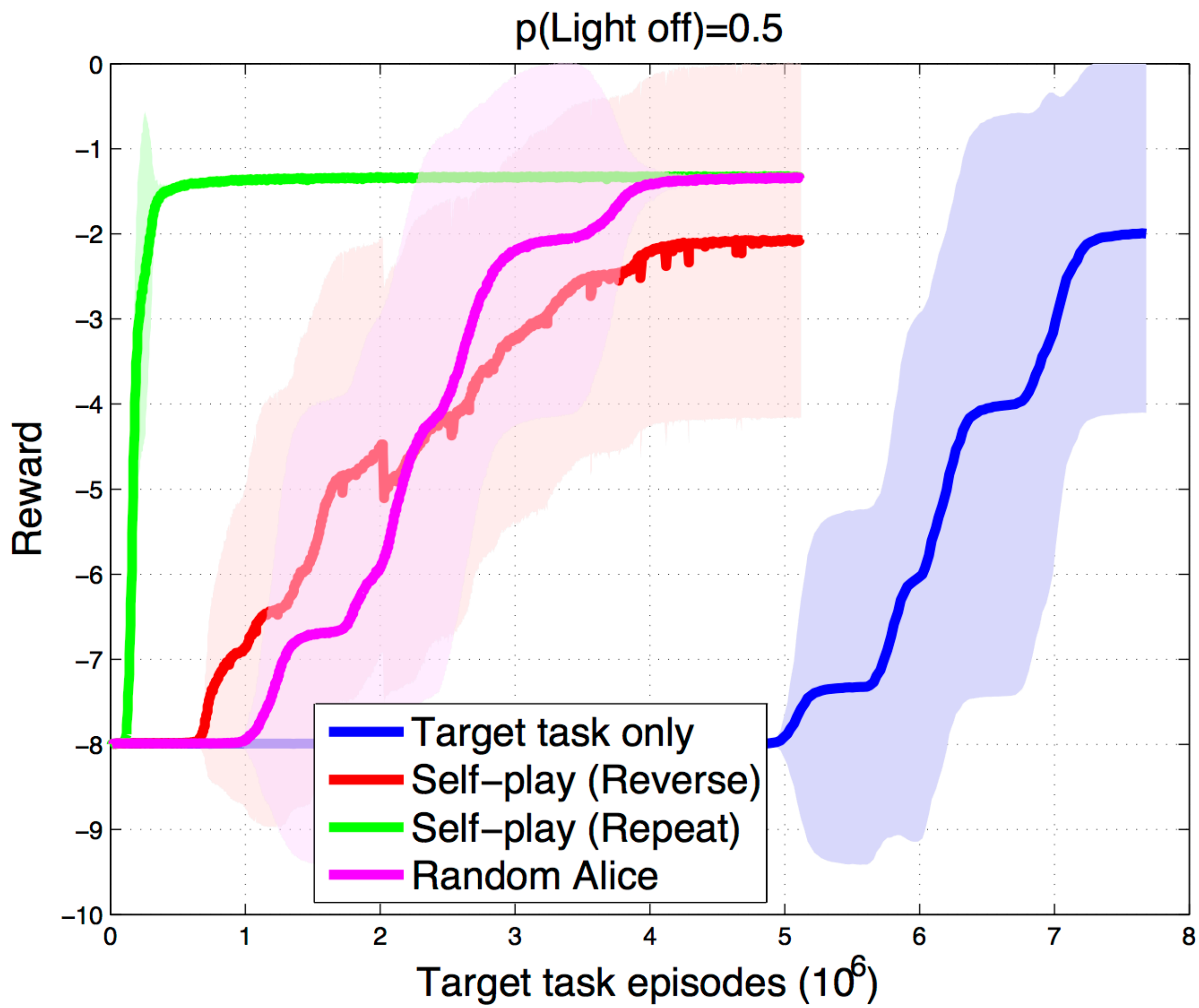
MazeBase: LightKey task

- Small 2D grid separated into two rooms by a wall
- The grid is procedurally generated
  - Object/agent locations randomized for each episode
- Toggle the key to lock/unlock door
  - Can't go through a locked door
- Toggle the light on/off
  - Only the switch is visible in dark
- Target task is to reach the goal flag in the opposite room when light is off and door is locked.

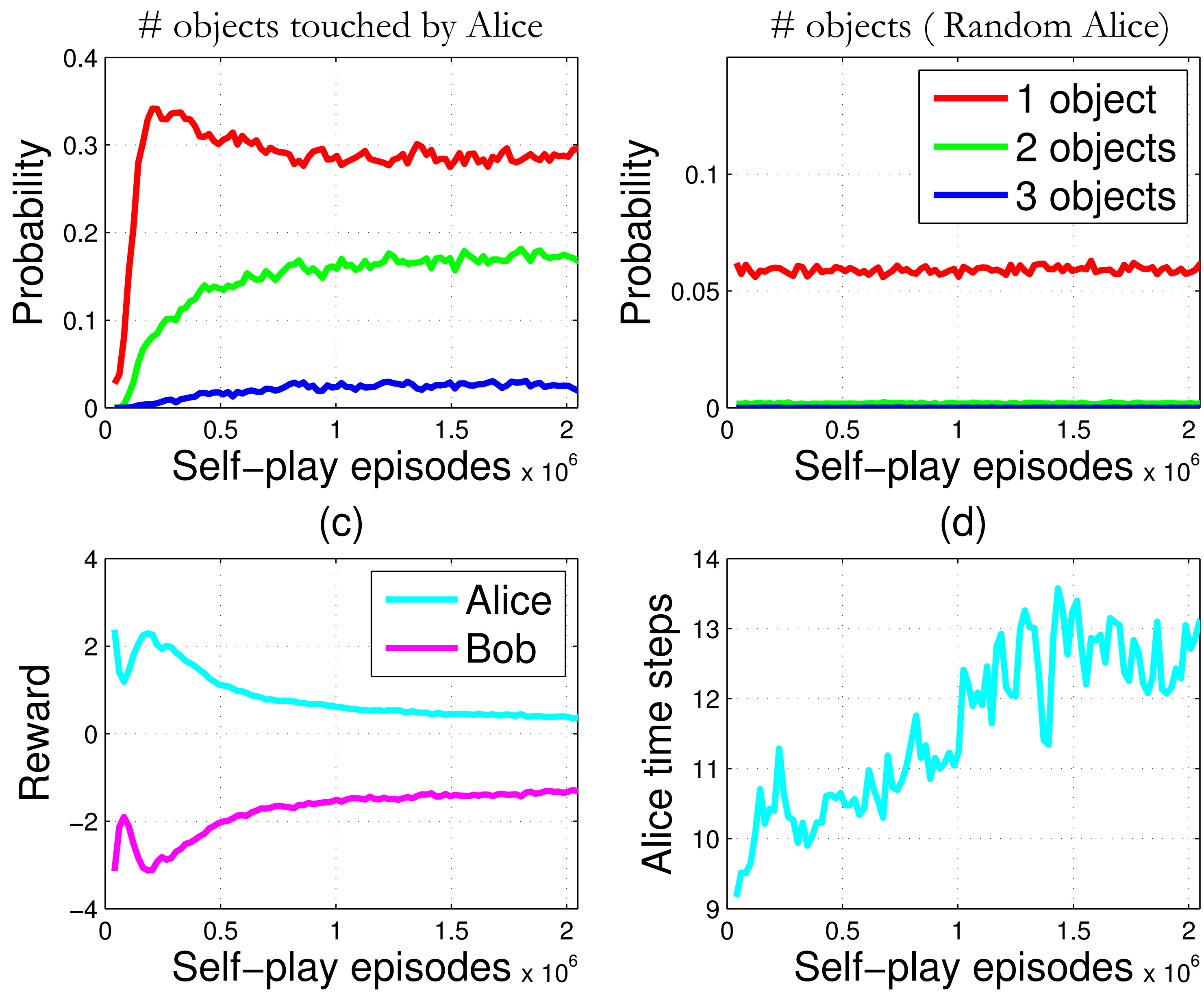


MazeBase: LightKey task

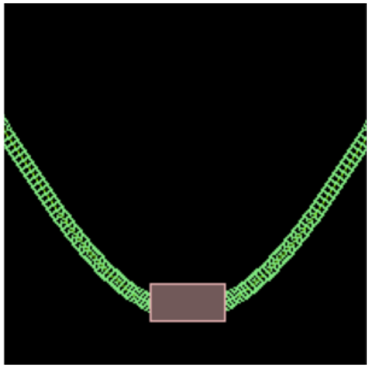
- Learn to navigate in a long corridor



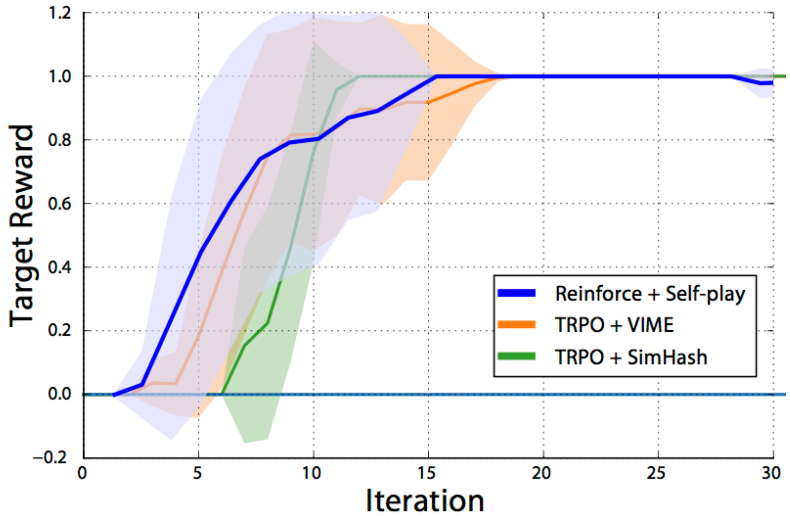
MazeBase: LightKey task



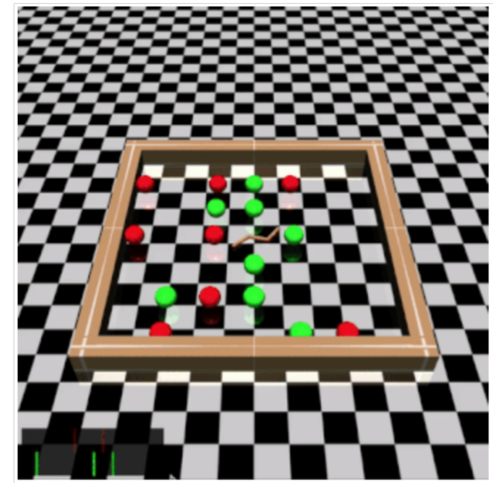
RL-Lab: Mountain Car



- Control a car stuck in 1D valley
  - Need to build momentum by reversing
- Sparse reward
  - +1 reward only if it reaches the left hill top
- Hard task because random exploration fails
- Asymmetric environment
  → repeat self-play
- As good as other exploration methods



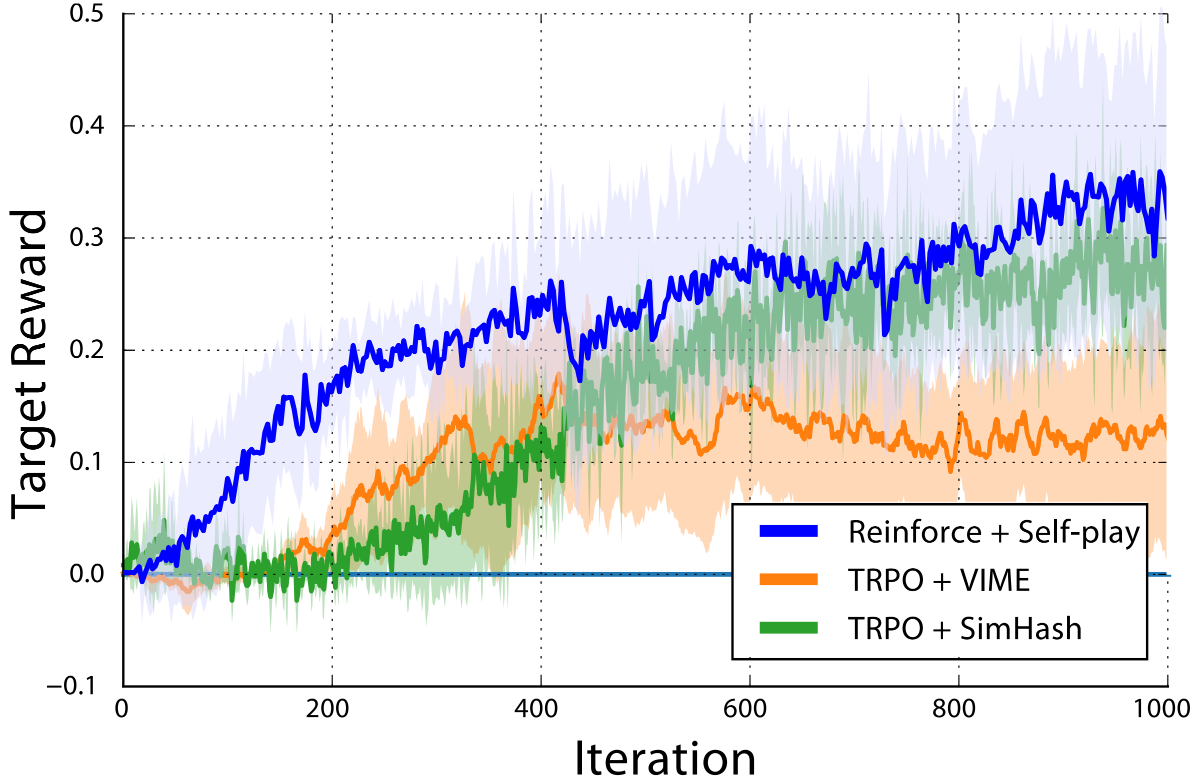
RLLab: Swimmer Gather



- Control a worm with two flexible joints, swimming in a 2D viscous fluid
- Reward +1 for eating green apples and -1 for touching red bombs
- Reverse self-play even though the environment is not strictly symmetric
- No apples or bombs during self-play
- Use only location (not full state) when deciding Bob's success during self-play

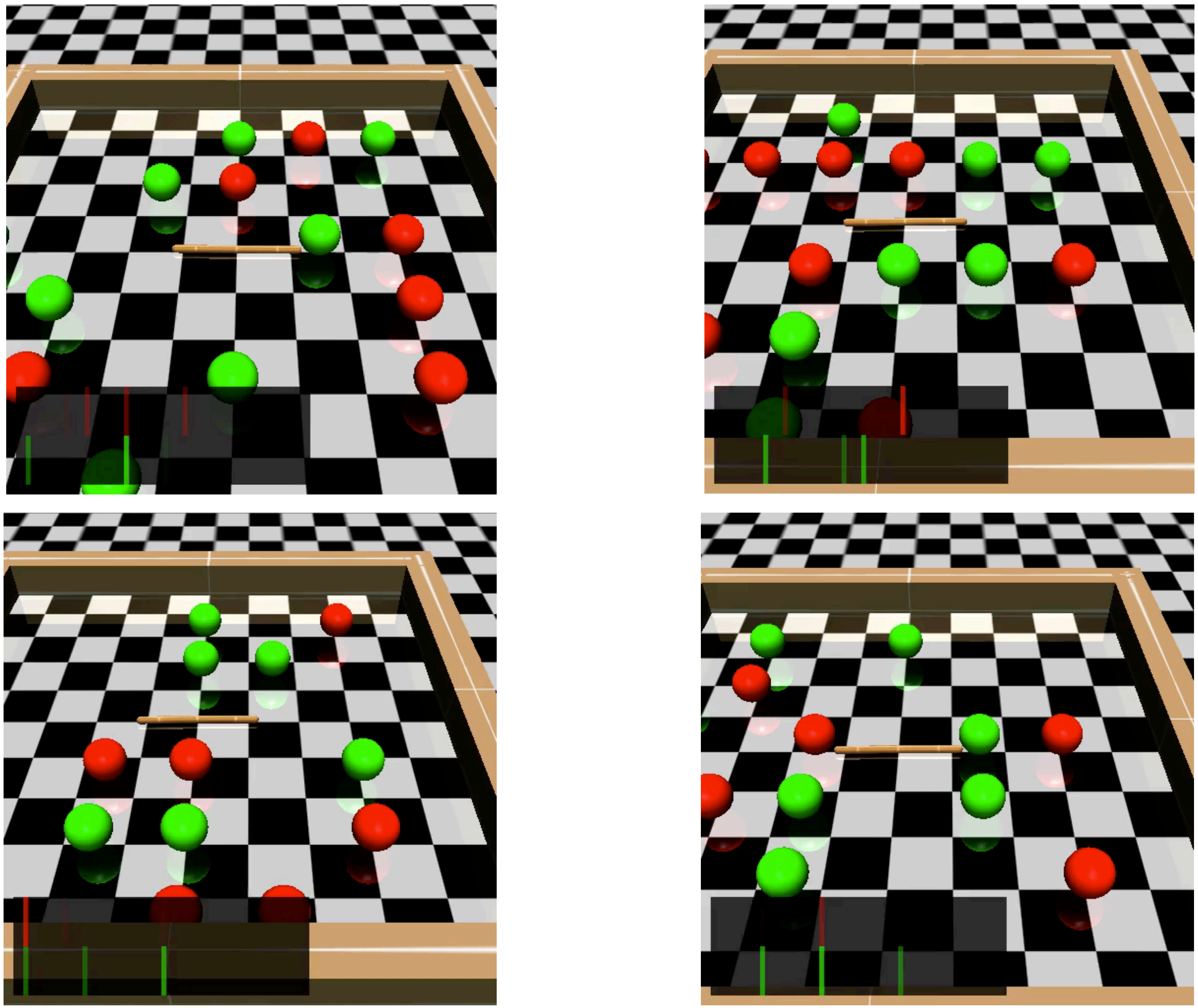
RLLab: Swimmer Gather

- Mean & S.D. over 10 runs
- Reinforce on target task alone gets zero reward



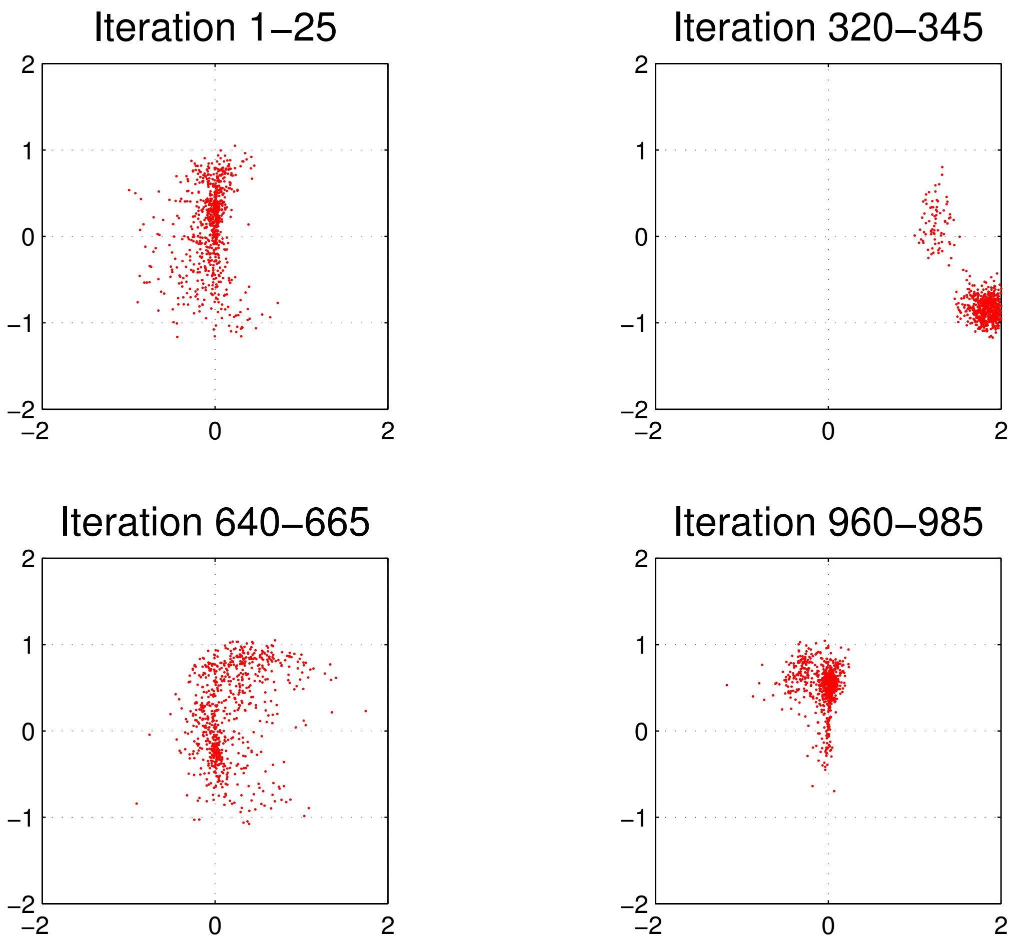
RLLab: Swimmer Gather

- Policy trained with Reinforce + self-play



RLLab: Swimmer Gather

- Distribution of locations where Alice hands over to Bob



Discussion

Paper: https://arxiv.org/abs/1703.05407

- Simple methods that works with discrete and continuous environments
- Meta–exploration for Alice
  - We want Alice to propose diverse set of tasks
  - But Alice focuses on the single best task
  - Multiple Alices?
- Future works:
  - Alice explicitly mark the target state
  - Alice propose task by communication without doing it
  - Alice propose a hypothesis and Bob test it