Distributed Deep Policy Sharing for Competitive Adversarial Environment

Denis Osipychev[1]

[1]Agricultural and Biological Engineering
University of Illinois at Urbana-Champaign

Dec 11, 2018

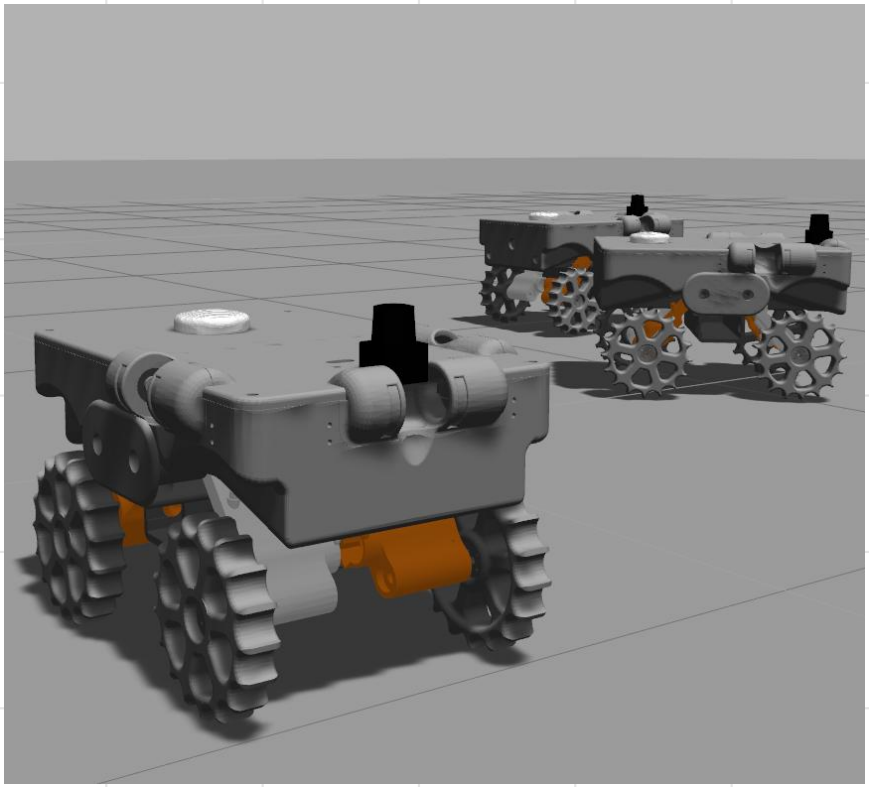
Objectives

- Establish a direct comparison of multi-agent RL training strategies in a unified framework.
- Deliver an open-source flexible multi-agent competitive environment Capture the Flag (CtF) that enables benchmarking and comparisons of potential control algorithms in a standardized manner.
- Investigate various strategies of sharing parameters of the policies and their impact on the training.

Framework

Framework specification:

- cooperative decision-making
- partially observable space
- stochastic environment
- joint reward
- part of larger robotic pipeline



Related Work

Deep Reinforcement Learning (RL) milestones related to multi-agent systems:

- Single-agent RL outperforms humans [1, 2, 3]
- MARL is very unstable [4, 5, 6]
- MARL extention of Actor-Critic method [6, 7, 8]

The results are:

- Focused on overall performance superiority
- Rarely explains the reasons behind its success
- In different domains (difficult to compare)

Environment Overview

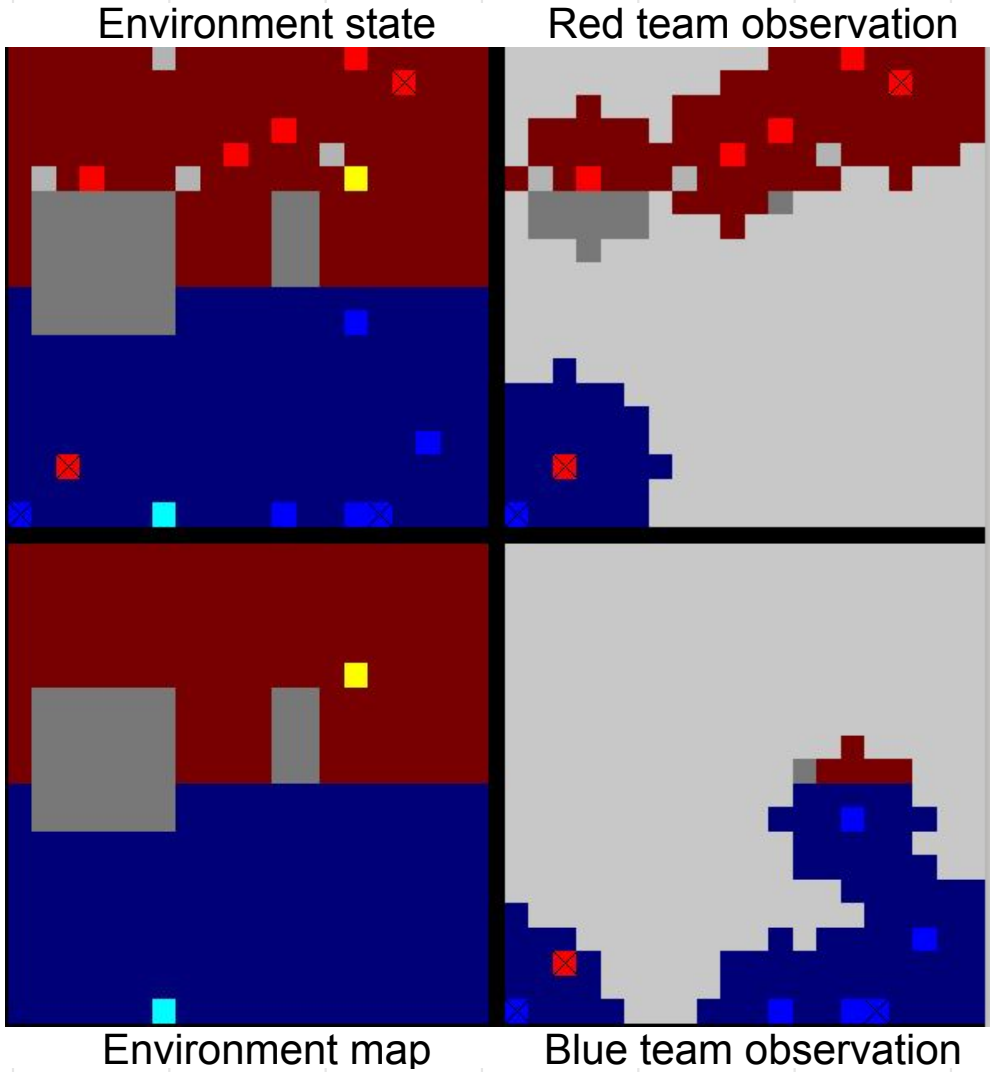


Figure: Python-based environment Capture the Flag (CtF) is designed to throw together various AI algorithms. The proposed simulation is a "gladiator pit" for algorithms. There are one neutral team (grey) and two teams that confront each other (red and blue). The goal is to capture the other team's flag or destroy the enemy units. Two types of units exist in each team and they have different abilities.

Agent Description

Type	UGV	UAV
Speed	1	3
Obs. Range	3	5
Attack Range	2	0
Qty	4	2

Table: Type of units available to the team. Default speed and range parameters shown in factual values (e.g. cell/step).

Multi-Agent Policy Gradient with Baseline

$$\nabla_\theta E_{s\sim\pi(s;\theta)}[R(s)] = \nabla_\theta \sum_s \pi(s,\theta)R(s) = \nabla_\theta \sum_s \frac{\pi(s,\theta')}{\pi(s;\theta')}\pi(s,\theta)R(s)$$

$$= \nabla_\theta E_{s\sim\pi(s;\theta')}\Big[\frac{\pi(s;\theta)}{\pi(s;\theta')}R(s)\Big] = E_{s\sim\pi(s;\theta')}[\frac{\pi(s;\theta)}{\pi(s;\theta')}R(s)\nabla_\theta \log\pi(s;\theta)] \quad (1)$$

$$\nabla_\theta E_{s\sim\pi(s;\theta)}[R(s)] = E_{s\sim\pi(s;\theta')}[\frac{\pi(s;\theta)}{\pi(s;\theta')}(R(s) - \hat{V}(s;\omega))\nabla_\theta \log\pi(s;\theta)]$$

$$= E_{s\sim\pi(s;\theta')}[\frac{\pi(s;\theta)}{\pi(s;\theta')}(r(s) + \gamma\hat{V}(s';\omega) - \hat{V}(s;\omega))\nabla_\theta \log\pi(s;\theta)] \quad (2)$$

$$\nabla_\theta E_{z\sim\pi(\{z_i(t)\};\theta)}[R(s)] = \nabla_\theta \sum_s \pi(\{z_i(t)\};\theta)R(s)$$

$$= E_{s\sim\pi(\{z_i(t)\};\theta')}[\frac{\pi(\{z_i(t)\};\theta)}{\pi(\{z_i(t)\};\theta')}\delta_i(t)\nabla_\theta \log\pi(\{z_i(t)\},\theta)]$$

$$= E_{s\sim\pi(\{z_i(t)\};\theta')}[\frac{\pi(\{z_i(t)\};\theta)}{\pi(\{z_i(t)\};\theta')}\sum_{i\in I}\delta_i(t)\nabla_\theta \log\pi(z_i(t);\theta)] \quad (3)$$

, where $\delta_i(t) = r(s(t)) + \gamma\hat{V}(z_i(t+1);\omega) - \hat{V}(z_i(t);\omega)$

Deep Policy Sharing Algorithm

- Proposed Multi-agent distributed deep policy sharing (DDPS) algorithm is compared with state-of-the-art algorithms:
  - MADDPG
  - Population based A3C
- The DDPS algorithm is based on a Policy Gradient (PG) method.
- In contrast to the methods with a shared critic [6], it shares both policy and critic components between the agents.
- In addition, it adopts the following techniques to improve convergence:
  - first-person view observation
  - experience replay
  - policy transfer
  - self-play

Deep Policy Sharing Algorithm

Data: CtF simulation
Result: Optimal policy model θ^*
Initialize $\theta, \omega \in \mathbb{R}^{\mathbb{D}}$;
while $k < N_{max\ ep}$ **do**
  Generate trajectory τ_n for each agent using current policy $\pi(a|s, \theta)$:
  for $t = 0..T_{max}$ **do**
    $Z_n \leftarrow \text{Preproc}(s)$;
    $A_n, P_n \sim \pi(A_n|Z_n, \theta)$;
    $s', r \leftarrow \text{CtF}(A_n)$;
    $\tau_n \leftarrow (Z_n, A_n, Z'_n, r, P_n)$;
  end
  Populate buffer B with τ_n;
  if $k = K_{update}$ **then**
    Sample buffer $z, r, t \sim B(U(0, N_{\text{buffer}}))$;
    $\delta \leftarrow r(t) + \gamma \hat{V}(z_i(t+1); \omega) - \hat{V}(z_i(t); \omega)$;
    $\theta \leftarrow \theta + \alpha_\theta \gamma^t \frac{p(a|z,\theta)}{p(a|z,\theta')} \delta \nabla \log(\pi(a_t|z_t; \theta))$;
    $\omega \leftarrow \omega + \alpha_\omega \gamma^t \delta \nabla \hat{V}(z; \omega)$;
  end
end

Different Strategies of Parameter Sharing

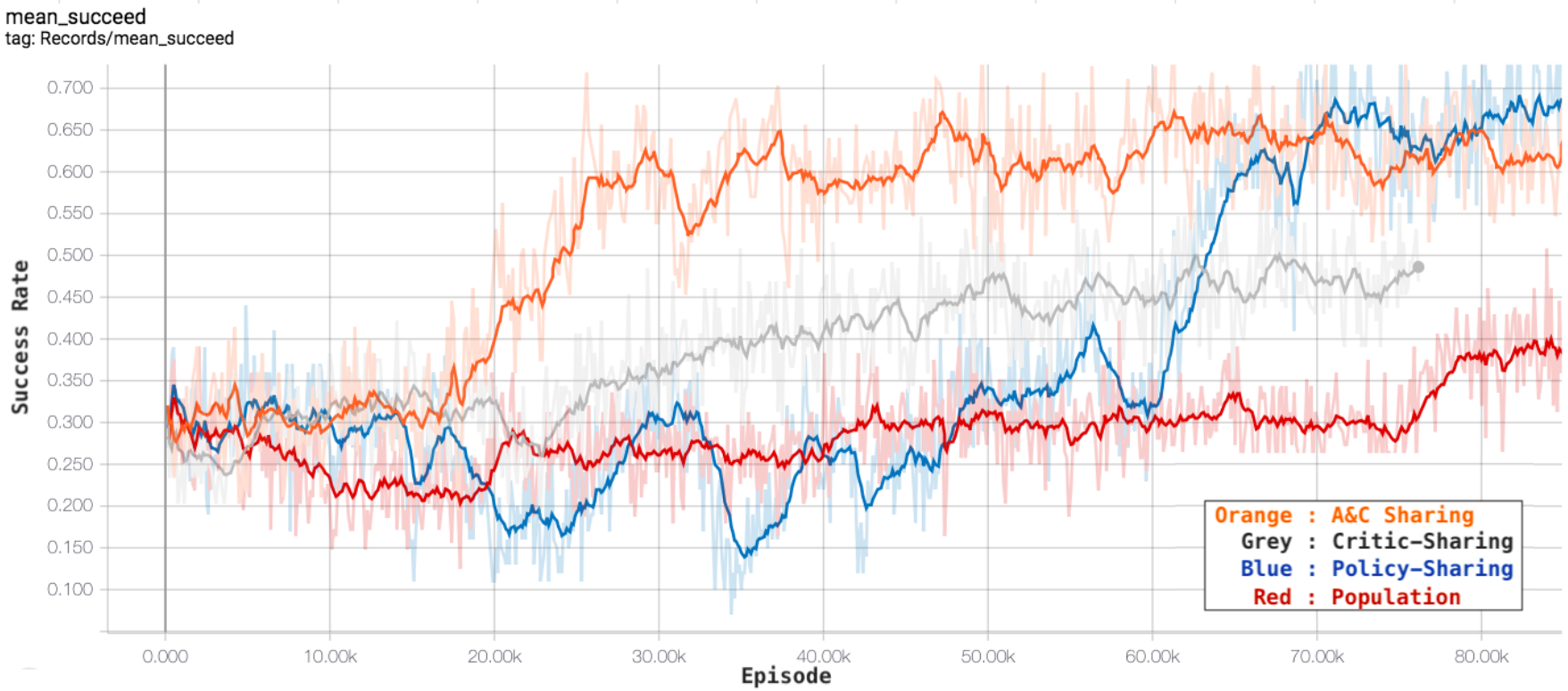


Figure: Performance comparison during the training. The proposed DDPS algorithm (denoted as A&C Sharing) is compared with different policy sharing strategies.

Shared Actor with/without Critic

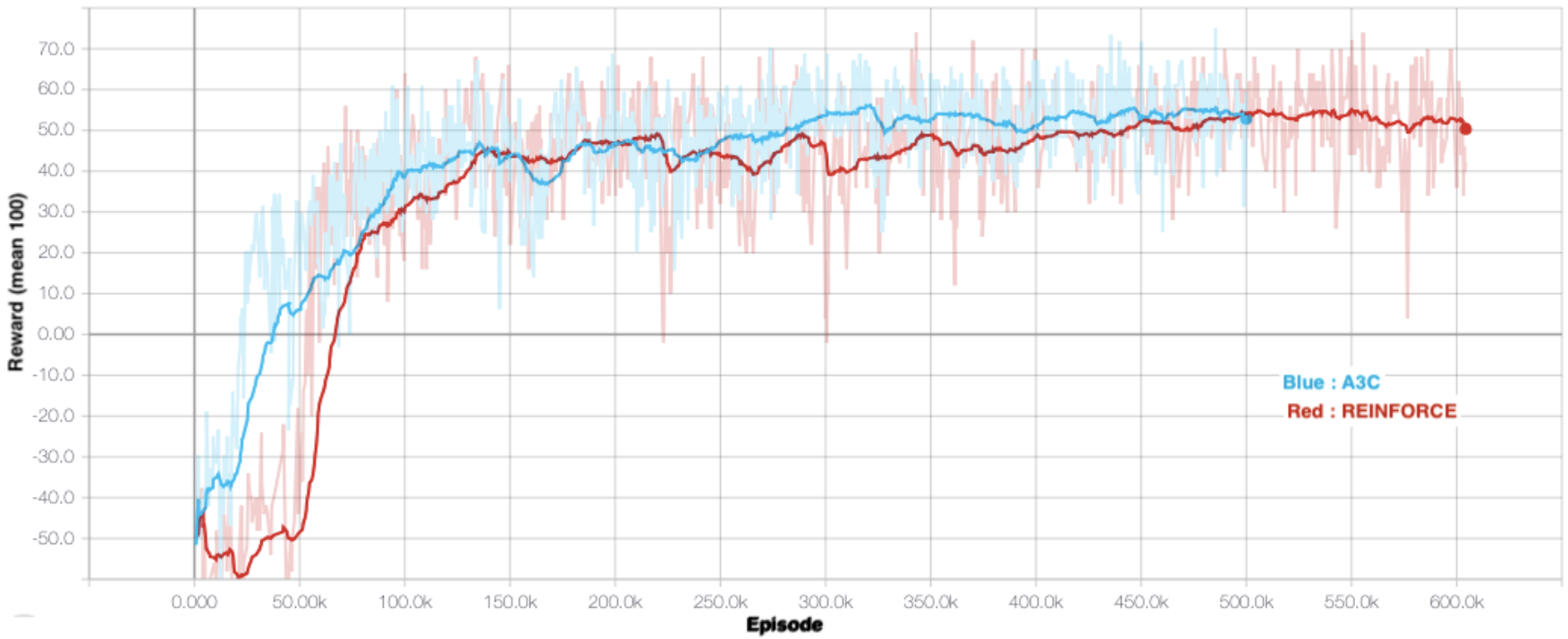


Figure: Performance comparison during the training. The proposed DDPS algorithm (denoted as A3C on the plot) is compared with REINFORCE-based algorithm with shared actor only.

One-on-One Final Policy Comparison

Team A	Team B	Win A	Win B	Ratio A	Ratio B	Draw
DDPS	REINFORCE	442	438	44.2	43.8	12
DDPS	POPULATION	465	464	46.5	46.4	7.1
REINFORCE	CRITIC	459	466	45.9	46.6	7.5
DDPS	Heuristic	571	97	57.0	9.7	33.2
REINFORCE	Heuristic	680	198	67.9	19.8	12.2

Table: Results of one-on-one comparison from 1000 games.

One-on-One Final Policy Comparison

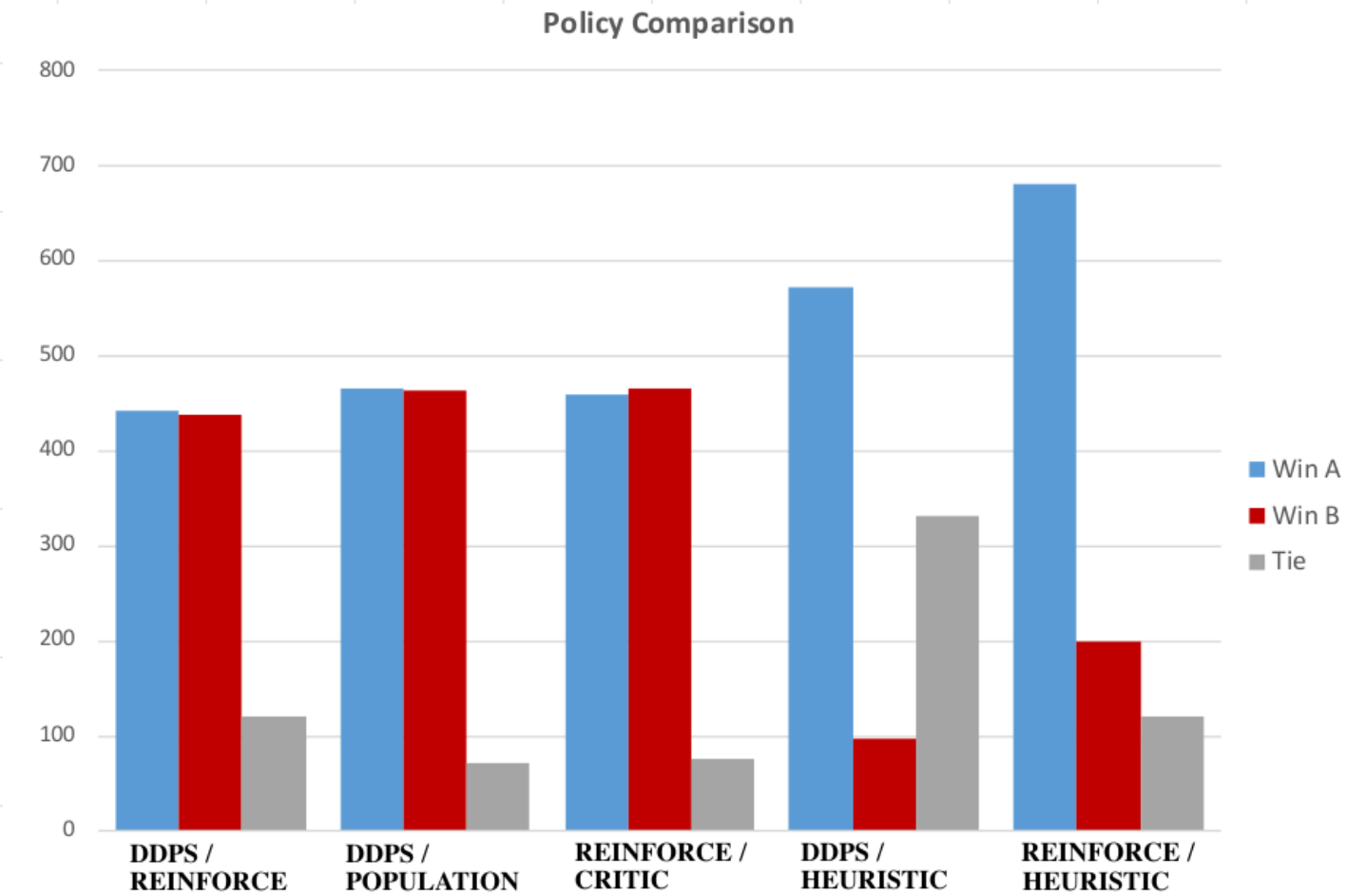


Figure: Performance of the final policy in a direct comparison.

Conclusion

Pros:

- Policy sharing outperforms other sharing strategies
- Allows collaborative behavior without expressing an intent
- Stabilizes training even without a critic network
- Sample efficient

Cons:

- Expensive agent communication

Future Work

- Adaptive hyper-parameters
- Meta-Learning for highly generalized solution
- Recurrent Networks to address POMDP


V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.


X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang, "Deep learning for real-time atari game play using offline monte-carlo tree search planning," in *Advances in neural information processing systems*, pp. 3338–3346, 2014.


D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.


A. Nowé, P. Vrancx, Y. De Hauwere, M. Wiering, and M. van Otterlo, "Reinforcement learning: state-of-the-art," *Game Theory and Multi-agent Reinforcement Learning*, pp. 441–470, 2012.


S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," *arXiv preprint arXiv:1703.06182*, 2017.


R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, pp. 6379–6390, 2017.


M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castaneda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, *et al.*, "Human-level performance in first-person multiplayer games with population-based deep reinforcement learning," *arXiv preprint arXiv:1807.01281*, 2018.


J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems*, pp. 66–83, Springer, 2017.