Comparison of Reinforcement Learning Algorithms in Problems of Acquiring Locomotion Skills in 3D Space

D. A. Kozlov[1]
djoade100@gmail.com

Introduction

In this work, we compare the effectiveness of modern reinforcement learning methods in solving the problem of acquiring movement skills by an agent in three-dimensional space. Algorithms SAC, PPO, MA-POCA are considered as compared algorithms. The comparison is made in the Unity simulator using the ML-Agents package. The result of the experiments confirms the effectiveness of modern reinforcement learning algorithms in the problem of acquiring the skills of movement in space. At the same time, the Soft Actor-Critic algorithm has the highest sample efficiency, which makes it possible to use it to complete training in real environments with a model previously trained in the simulator.

Reinforcement Learning

Reinforcement learning is a type of machine learning in which a system learns by interacting with its environment. The agent, interacting with the environment, receives a reward depending on his actions. The task of the agent is to maximize this reward. In this case, labeled datasets are not needed as supervised learning methods, but a well-formulated reward function is required.

Investigated algorithms

The following methods were compared in the work:

- Soft Actor Critic (SAC)
- Proximal Policy Optimization(PPO)
- MultiAgent POsthumous Credit Assignment (MA-POCA)

Simulation environment

The Unity simulation environment was chosen together with the ML-Agents package to compare the algorithms described above. This environment is a game engine that provides simulation of physics and graphics. Unity Machine Learning Agents Toolkit (ML-Agents) is an open-source project that allows you to use the capabilities of the Unity game engine in conjunction with implementations of high-performance machine learning algorithms written in Python using Pytorch and Tensorflow. In our experiment, the media described below were chosen. All environments aim to simulate the task of acquiring movement skills in three-dimensional space.

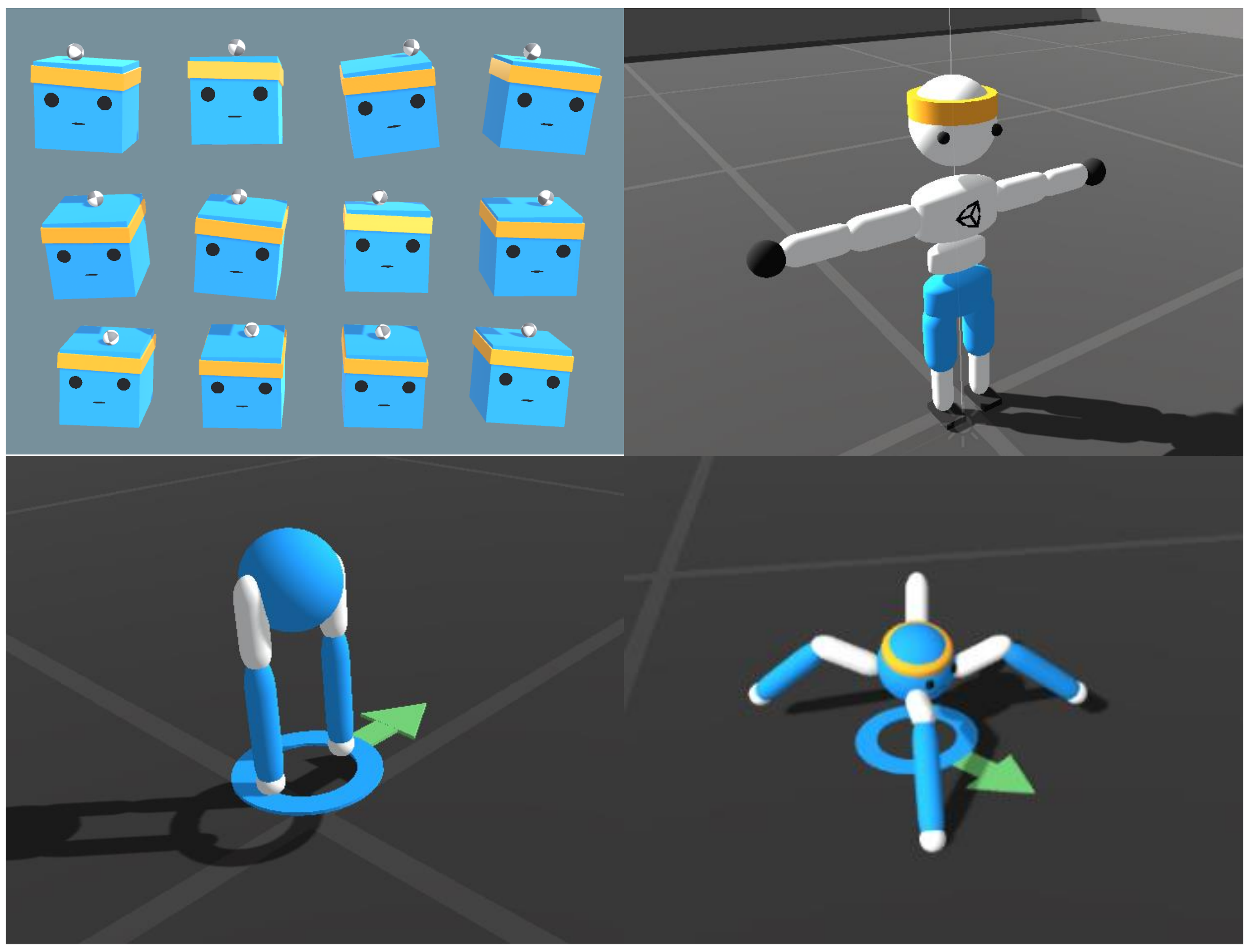

Fig 1. Appearance of Agents 3DBall, Walker, SimplestBipedal and Crawler

Environments

Each agent sits on a large square platform. The size of the platform exceeds the size of the agent by more than 100 times. At the beginning of the episode, the agent is placed in the middle of this square area, and the object that is the goal that the agent needs to reach is placed in a random point of this area. If the agent reaches the target, then he receives a reward, and the target disappears and appears at another random point on the site. Now the agent needs to turn to the target and continue moving.

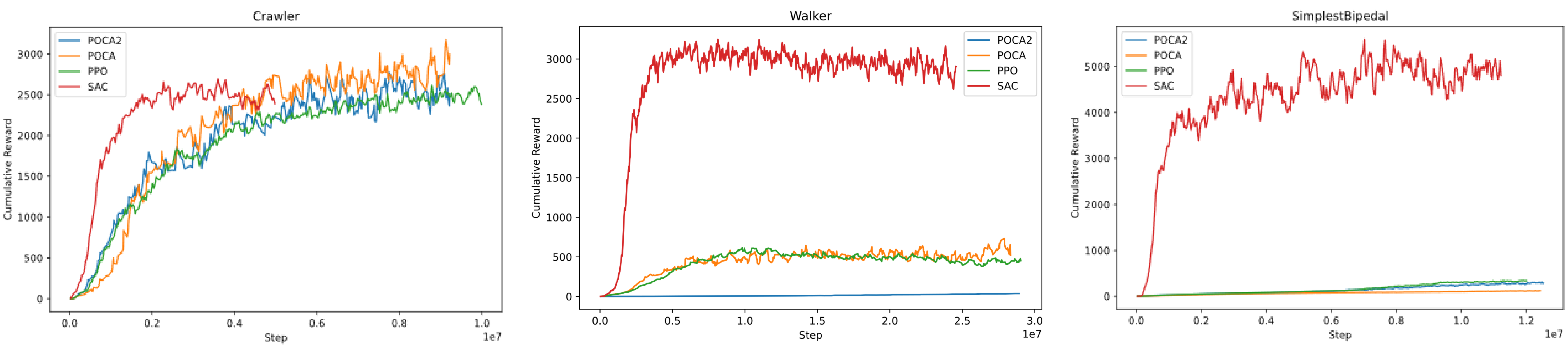

Fig 2. The result of comparing the work of algorithms in the Crawler, Walker, SimplestBipedal environments

Conclusion

The experiments make it clear that the SAC algorithm is best suited for this task. It is also worth noting that these graphs are constructed so that the steps of the algorithm are plotted along the horizontal axis. That is, these graphs reflect the sample efficiency of the algorithm. If we consider the actual time of execution of specific learning processes, then in the case of, for example, the Crawler environment, the results of all algorithms are almost identical. However, it is worth remembering that such a scenario is only possible when working with a simulator. If you need to teach a real robot something, it will spend much time returning the robot to its original position to repeat the learning process and perform the following steps. Therefore, for real robots, algorithms that are more computationally complex in the learning process but at the same time have greater sample efficiency will be much more valuable.

Acknowledgement

The work was supported by the Russian Science Foundation grant No. 21-11-00321, https://rscf.ru/en/project/21-11-00321.

[1]SAMARA NATIONAL RESEARCH UNIVERSITY
34, Moskovskoye shosse, Samara, 443086, Russia
www.ssau.ru

SAMARA UNIVERSITY