УДК 004.85

Обучение агентов в виртуальной среде KukaDiversObjectEnv

Н.Е. Залогин
Научный руководитель: ст. преп. Д.С. Григорьев
Национальный исследовательский Томский политехнический университет,
Россия, г. Томск, пр. Ленина, 30, 634050
E-mail: nez4@tpu.ru

Training agents in a KukaDiversObjectEnv virtual environment

N.E. Zalogin
Scientific Supervisor: Senior lecturer D.S. Grigoriev
Tomsk Polytechnic University, Russia, Tomsk, Lenin str., 30, 634050
E-mail: nez4@tpu.ru

Abstract. *The present study implements and compares the DQN, PPO, Parallel PPO, and Modified PPO algorithms in the PyBullet KukaDiverseObjectEnv environment. The algorithms are tested and evaluated in a simulated test mode to assess their performance. The experiments focus on metrics such as learning speed, stability, and task completion success rate. The results provide insights into the effectiveness of each algorithm in the tested environment, aiding in the optimization of reinforcement learning algorithms for complex environments and robotics.*
Key words*: Reinforcement learning, Robotic arm, PyBullet*

Введение

Обучение с подкреплением (Reinforcement Learning, RL) стало мощным инструментом для обучения интеллектуальных агентов выполнять различные задачи в сложных средах. В последние годы все больше растет интерес к применению методов в RL робототехнике с целью обеспечения автономности и адаптивности к окружающей среде. В данной работе рассматривается процесс обучения агентов с использованием различных алгоритмов RL в виртуальной среде KukaDiversObjectEnv [1] на платформе PyBullet [2].

Экспериментальная часть

KukaDiverseObjectEnv – среда симуляции на платформе PyBullet, разработанная для обучения роботов манипулированию объектами. Входными данными служит rgb изображение, пример такого изображения приведен на рис. 1. Функция награды в этой среде является двоичной и выдается за успешное поднятие предмета. Благодаря редкой награде и сложности задачи, среда становится достаточно интересной для изучения.

Для обучения использованы алгоритмы Deep Q-Networks (DQN) [3], Proximal Policy Optimization (PPO) [4] и две модификации примененного алгоритма PPO.

Реализация DQN агента опирается на методы, описанные в учебном пособии PyTorch [5]. Алгоритм PPO агента базируется на реализации, представленной в соответствующем блог-посте [6]. Первая модификация PPO включает параллелизацию среды с использованием библиотеки multiprocessing. Вторая модификация PPO включает дополнительные улучшения такие как более глубокие слои actor-critic, включая общие слои (shared layers) и сверточные слои (convolutional layers). Кроме того, применяются определенные алгоритмические особенности, такие как нормализация преимуществ после их вычисления на основе обобщенной оценки преимуществ (GAE) [7] и уменьшение learning rate, epsilon, beta по ходу обучения.

Для тестирования и оценки алгоритмов обучения агентов был установлен критерий прекращения обучения - достижение средней награды в 50 за 100 эпизодов, что эквивалентно 50 % точности.

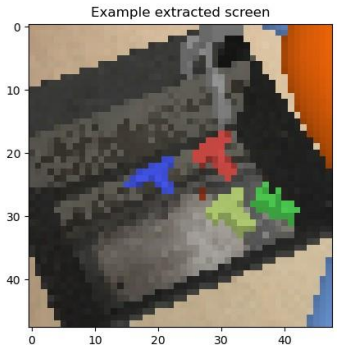

Рис. 1. Пример изображения с камеры робота

Для оценки точности обученных агентов использовалась та же среда в режиме тестирования, позволяющая проверить их способность достигать подобных результатов при отличающихся условиях.

Результаты

По ходу обучения агентов были проведены измерения средней награды за 100 эпизодов, результаты которых представлены на рисунке 2. Следует отметить, что цикл обучения агента с использованием PPO включает в себя сбор информации о 1024 шагах перед началом обучения. В то время как алгоритм DQN, обучается после каждого шага в среде. С учетом этого различия, будем считать, что каждая итерация PPO эквивалентна 1024 итерациям DQN. В дополнение к графикам средней награды, были зафиксированы показатели конечной средней награды, количество шагов обучения и время обучения, приведенные в таблице 1.

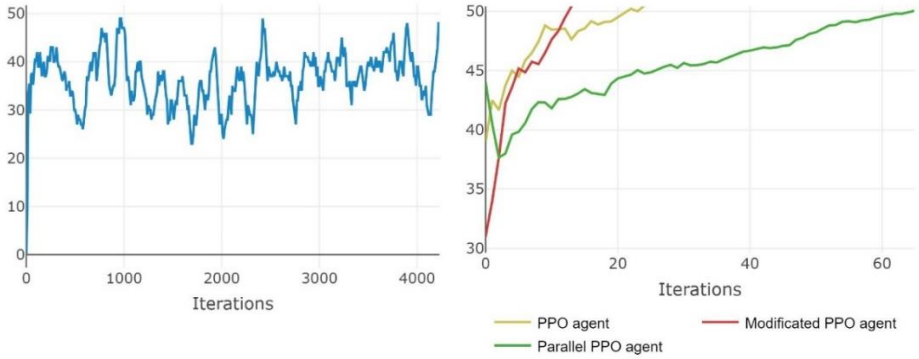

Рис. 2. Средняя награда агента DQN (а), Средняя награда агентов PPO (б)

Таблица 1

Сравнение показателей обученных агентов DQN и PPO

Агент	Средняя награда	Шаги обучения	Время обучения
DQN	51,000	4239	2:06:54,878
PPO	50,690	26 (26624)	1:52:07,842
Параллельный PPO	50,061	65 (66560)	0:35:12,836
Модифицированный PPO	50,067	13 (13312)	0:09:34,629

Из результатов, представленных на рисунках 2-3 и таблице 1, можно сделать вывод о том, что параллелизация среды ускорила процесс обучения агента. Однако, это привело к увеличению количества итераций обучения. Модификация параллельного PPO успешно решает эту проблему, позволяя сильнее сократить время обучения агента.

Для тестирования полученных агентов использовалось 1000 эпизодов среды в тестовом режиме. Результаты тестирования приведены в таблице 2.

Таблица 2

Тестирование обученных агентов DQN и PPO

Агент	DQN	PPO	Параллельный PPO	Модифицированный PPO
True episode	47,8%	50,2%	49,1%	52,4%
False episode	52,2%	49,8%	50,9%	47,6%

Из таблицы 2 можно сделать вывод, что все агенты успешно решают поставленную задачу и достигают точности в окрестности 50 процентов в 1000 тестовых эпизодов. Это показывает способность агентов эффективно приспосабливаться к среде и достигать высоких результатов.

Заключение

Благодаря параллелизации сред, процесс обучения агента был существенно ускорен, несмотря на увеличение количества итераций. Модификация параллельного PPO устраняет этот недостаток и еще более сокращает время обучения. Все реализованные агенты, включая DQN, PPO, Parallel PPO и Modified PPO, успешно решают поставленную задачу, достигая около 50 % точности за 1000 тестовых эпизодов.

В будущем предполагается дальнейшее улучшение алгоритма PPO и общего процесса обучения для достижения лучших результатов. Рассмотрение различных дополнительных стратегий, таких как HER или PCCL, может расширить возможности и повысить производительность агентов. Также замена среды обучения с KukaDiverseObjectEnv на другую или создание специальной среды может открыть новые направления исследований в области обучения с подкреплением.

Список литературы

1. Bulletphysics // GitHub: сайт. – 2024. – URL: https://github.com/bulletphysics/bullet3/blob/master/examples/pybullet/gym/pybullet_envs/bullet/kuka_diverse_object_gym_env.py (дата обращения 15.03.2024).

2. Bullet Real-Time Physics Simulation// PyBullet: сайт. – 2024. – URL: https://pybullet.org/wordpress/ (дата обращения 15.03.2024).

3. Mnih V. et al. Human-level control through deep reinforcement learning // Nature. – 2015. – Vol. 518, № 7540. – P. 529-533.

4. Schulman J. et al. Proximal policy optimization algorithms. // arXiv: сайт. – 2017. – URL: https://arxiv.org/abs/1707.06347 (дата обращения 15.03.2024).

5. Reinforcement Learning (DQN) Tutorial // PyTorch: сайт. – 2024. – URL: https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html (дата обращения: 15.03.2024).

6. Huang S. et al. The 37 implementation details of proximal policy optimization // ICLR: сайт. – 2022. – URL: https://iclr-blog-track.github.io/2022/03/25/ppo-implementation-details/ (дата обращения: 15.03.2024).

7. Schulman J. et al. High-dimensional continuous control using generalized advantage estimation// arXiv: сайт. – 2017. – URL:https://arxiv.org/abs/1707.06347 (дата обращения 15.03.2024).