Сравнение алгоритмов обучения с подкреплением в задаче приобретения навыков передвижения в трёхмерном пространстве

Д.А. Козлов
Самарский национальный исследовательский университет им. академика С.П. Королева
Самара, Россия
djoade100@gmail.com

***Аннотация*—В работе выполняется сравнение современных методов обучения с подкреплением на примере решения задачи приобретения агентом навыков передвижения в трёхмерном пространстве. Сравнение производится в симуляторе Unity с использованием пакета ml-agents. В качестве сравниваемых алгоритмов выступают: SAC, PPO, MA-POCA. Они используются для обучения навыкам передвижения нескольких моделей агентов: 3DBall, Crawler, Walker и авторской SimplestBipedal. Результаты экспериментов говорят о преимуществах алгоритма Soft Actor Critic, что делает его более перспективным для использования в реальных средах.**

Ключевые слова— обучение с подкреплением, SAC, PPO, MA-POCA, робототехника, Unity ML-Agents, симуляция, MDP, POMDP.

1. Введение

Проблема передвижения в трёхмерном пространстве автономных устройств – одна из центральных в робототехнике. Она может рассматриваться в различных постановках, начиная с задач самостоятельной ориентации и планирования маршрута (SPLAM-системы, [1]) и заканчивая задачами обучения собственно движению сложных устройств. Настоящая работа посвящена последней проблеме применительно к антропоморфным и животно-подобным устройствам. Выбор объекта исследования обусловлен тем, что для типовых (колесных, гусеничных) беспилотных наземных транспортных средств передвижение по пересеченной местности затруднительно или невозможно. Это дает нетипичным (антропоморфным и многоногим) устройствам преимущество при решении задач в сложных ситуациях, а задачу обучения их передвижению делает актуальной.

В настоящей работе рассматриваются подобные устройства, взаимодействующие с виртуальной симуляцией реального мира в среде игрового движка Unity. Симуляция позволяет ускорить процесс их обучения и тестирования.

Целью работы является сравнение современных эффективных алгоритмов обучения с подкреплением (англ.: Reinforcement Learning, RL) в задаче обучения автономных (антропоморфных и многоногих) мехатронных объектов походке и передвижению. В качестве сравниваемых алгоритмов выступают: Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO) и MultiAgent POsthumous Credit Assignment (MA-POCA). Указанные алгоритмы используются для обучения навыкам передвижения нескольких различных по структуре моделей агентов: 3DBall, Crawler, Walker и авторской SimplestBipedal.

Наиболее похожее сравнение приводится в работе [2]. Указанная работа отличается от настоящей тем, что в [2] отсутствует сравнение с алгоритмом MA-POCA; кроме того, в настоящей работы мы расширили состав моделей – ввели в рассмотрение новую созданную нами модель SimplestBipedal. Сравнение с MA-POCA было представлено в работе [3], но оно было выполнено авторами алгоритма MA-POCA. Также в работе [3] отсутствует сравнение с алгоритмом SAC, а также для сравнения используется иной набор задач/моделей. Кроме приведенных выше двух работ, авторами не было обнаружено независимых сравнений эффективности использования RL-методов для обучения передвижению разнотипных моделей ни для среды Unity, ни в похожих задачах; а найденные сравнения выполняются для более тривиальных сред/моделей.

Структура работы следующая. В разделе 2 представлены подробности исследуемых алгоритмов, в разделе 3 даны постановка и результаты экспериментов. В конце работы приводится заключение, в котором приведены выводы на основе полученных в экспериментах результатов.

2. Метод и алгоритмы обучения

Обучение с подкреплением – способ машинного обучения, при котором *агент* (англ.: actor) обучается, взаимодействуя со средой. Для формализации RL-методов используется нотация Марковского процесса принятия решений по причине известного свойства марковости – переходы между состояниями зависят только от последнего состояния, а не от предыдущей истории. А именно: (S, A, R, P, ρ_0), здесь S — множество допустимых состояний, A — множество допустимых действий, $R: S \times A \times S \to \mathbb{R}$ — функция вознаграждения, где $r_t = R(s_t, a_t, s_{t+1})$, $P: S \times A \to \mathcal{P}(S)$ - функция вероятности перехода, то есть $P(s'|s,a)$ - вероятность перехода в состояние s', если вы находитесь в состоянии s и выполняете действие a, ρ_0 — начальное распределение состояний. Термином (*стохастическая*) *политика* обозначают стратегию поведения агента, то есть правило или способ выбора конкретного действия в конкретном состоянии: $\pi(a|s)$. Если политика явно не формируется, тогда алгоритм обозначают «off-policy».

A. Алгоритм Soft Actor-Critic

Soft Actor Critic – off-policy алгоритм глубокого RL. В нем агент стремится максимизировать ожидаемое вознаграждение, а также максимизировать энтропию. То

есть преуспеть в задаче, действуя как можно хаотичнее. Сочетая хаотичность со стабильной структурой подхода «actor-critic», метод достигает наилучшей производительности в ряде задач непрерывного управления. Кроме того, в отличие от других off-policy алгоритмов, метод очень стабилен, то есть достигает при разных случайных начальных значениях примерно одинаковой эффективности решения.

Б. Алгоритм Proximal Policy Optimization

Proximal Policy Optimization (PPO)[4] - семейство RL-методов градиента политики, которые чередуют наблюдения (для определения состояния) посредством взаимодействия с окружающей средой и оптимизацию «суррогатной» целевой функции с использованием стохастического градиентного подъема. В отличие от стандартных методов градиента политики, которые выполняют одно обновление градиента для каждого наблюдения, здесь обеспечивается несколько эпох мини-пакетных обновлений.

В. Алгоритм MA-POCA

MA-POCA [3] (MultiAgent POsthumous Credit Assignment), представляет собой многоагентный алгоритм. Он использует искусственную нейронную сеть (ИНС), выступающую в роли критика/оценщика решения, которая действует как «тренер» для целой группы агентов (которые также могут быть реализованы как ИНС).

3. Эксперимент

А. Среда симуляции

Unity Machine Learning Agents Toolkit [2] (ML-Agents) – проект с открытым исходным кодом, позволяющий использовать возможности игрового движка Unity совместно с реализациями высокопроизводительных алгоритмов машинного обучения, написанных на Python с использованием Pytorch и Tensorflow [5-6]. В качестве задач обучения и агентов выступали: 3DBall, Crawler, Walker и оригинальный SimplestBipedal. На рисунке 1 слева представлен агент 3DBall. Задача этого агента - балансировать шар на платформе. Наблюдение состояния включает координаты шара, скорости вдоль каждой оси и наклон платформы. Награда в этой задаче определяется как время, в течение которого шар находится на платформе. Далее на рисунке 1 представлены агенты Crawler, Walker и SimplestBipedal. Задача этих агентов – переместиться как можно ближе и быстрее к заданной цели в 3D пространстве. Для них награда даётся пропорционально скорости движения агента и направлению движения к цели. Следует отметить, что SimplestBipedal – это самостоятельно созданная задача. Она подразумевает высокую сложность задачи обучения из-за структурных особенностей агента - в верхнем суставе присутствует подвижность в двух плоскостях, а в нижнем – лишь в одной.

Б. Результат экспериментов

В результате проведенных экспериментов были получены усредненные значения суммарной награды, показанные в таблице 1. Из таблицы видно, что SAC превосходит остальные исследуемые алгоритмы.

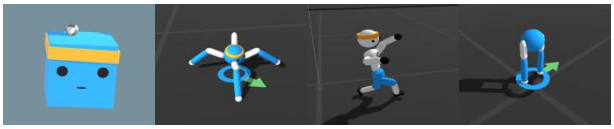

Рис. 1. Агенты 3DBall, Crawler, Walker, SimplestBipedal

Следует также отметить, что в ходе нашего сравнения были получены результаты, отличающиеся от представленных в работе [2]. Это может быть связано с тем, что указанное исследование проводилось 2 года назад, за которые вносились изменения как в реализации алгоритмов, так и в движок Unity.

Таблица I. Результаты экспериментов

Агент	**3D Ball**	**Crawler**		**Walker**		**Simplest Bipedal**	
Шаг (млн.)	***0.5***	***2***	***5***	***5***	***10***	***2***	***10***
SAC	100	**2500**	2500	**3000**	**3000**	**3700**	**4700**
PPO	100	1500	2500	250	500	0	100
MA-POCA	100	1500	2500	250	500	0	100

4. Заключение

В результате проведенного исследования было выявлено, что наиболее эффективным методом обучения с подкреплением для решения задачи приобретения навыков передвижения в трёхмерном пространстве является алгоритм Soft Actor Critic. Видно, что SAC в среде Crawler достигает того же уровня награды, что и другие алгоритмы, но за меньшее количество шагов, а в экспериментах Walker и SimplestBipedal SAC значительно превосходит PPO и MA-POCA по значению достигнутой средней суммарной награды. Эти результаты говорят о том, что SAC является наиболее перспективным вариантом для использования в различных робототехнических приложениях. В том числе алгоритм может использоваться для обучения нетипичных устройств в реальной среде.

Благодарности

Работа выполнена при поддержке Российского научного фонда (проект № 21-11-00321, https://rscf.ru/en/project/21-11-00321/).

Литература

[1] Kozlov, D. Development of an Autonomous Robotic System Using the Graph-based SPLAM Algorithm / D. Kozlov, V. Myasnikov // International Conference on Information Technology and Nanotechnology (ITNT). – 2021. – P. 1-5. DOI: 10.1109/ITNT52450.2021.9649028.

[2] Juliani, A. Unity: A General Platform for Intelligent Agents / A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, D. Lange // ArXiv: 1809.02627, 2020.

[3] Cohen, A. On the Use and Misuse of Absorbing States in Multi-agent Reinforcement Learning / A. Cohen, E. Teng, V.-P. Berges, R.-P. Dong, H. Henry, M. Mattar, A. Zook, S. Ganguly // ArXiv: 2111.05992, 2021.

[4] Schulman, J. Proximal Policy Optimization Algorithms / J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov // ArXiv: 1707.06347, 2017.

[5] PyTorch [Electronic resource]. – Mode of access: https://www.pytorch.org (19.04.2022).

[6] TensorFlow [Electronic resource]. – Mode of access: https://www.tensorflow.org/?hl=ru (19.04.2022).