МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)

На правах рукописи

Козлов Даниил Александрович

ИНТЕГРАЦИЯ ИЕРАРХИЧЕСКИХ АНСАМБЛЕЙ И ТРАНСФОРМЕРНЫХ АРХИТЕКТУР В АЛГОРИТМЫ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ

1.2.1 – Искусственный интеллект и машинное обучение

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
Мясников Владислав Валерьевич,
доктор физико-математических наук, профессор

Самара – 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования

Проблема управления роботами в сложных условиях становится все более актуальной в контексте быстрого развития технологий и увеличения сложности технических систем. Такие подходы к управлению, как обучение с подкреплением, предлагают возможности для значительного повышения эффективности и адаптивности роботов. Эти методы позволяют роботам самостоятельно изучать и оптимизировать свои стратегии поведения в реальном времени, что особенно важно для действий в условиях, где детальное предварительное моделирование среды невозможно или неэффективно.

Применение обучения с подкреплением в робототехнике привело к множеству значимых достижений в улучшении автономности, производительности и адаптивности роботов [1–3]. Одним из наиболее заметных примеров успешного применения обучения с подкреплением является разработка автономных транспортных средств [4]. Эти системы используют обучение с подкреплением для оптимизации стратегий вождения, позволяя автомобилям самостоятельно принимать решения в сложных дорожных условиях и оптимизировать потоки транспортных средств в городских сетях.

В области промышленного производства обучение с подкреплением применяется для управления роботизированными руками [5], которые выполняют задачи сборки и манипуляции с объектами. Эти роботы обучаются адаптироваться к изменениям в объектах или их расположении, что позволяет автоматизировать процессы, требующие высокой точности и гибкости.

Роботы, используемые в задачах поиска и спасения, должны работать в условиях высокой неопределенности и динамичных изменений среды [6]. Обучение с подкреплением позволяет этим роботам обучаться на основе взаимодействия с реальной средой, улучшая свои способности к самостоятельному принятию решений в критических ситуациях.

Алгоритмы и методы обучения с подкреплением также могут быть использованы для управления беспилотными летательными аппаратами [7], шагающими четырёхногими роботами в экстремальных условиях [8], манипуляционными роботами [9]. Отличительной особенностью алгоритмов обучения с подкреплением является тот факт, что для них не требуется точного моделирования среды, в которой они будут действовать. Вместо этого агент сам изучит среду и обучится принимать оптимальные решения.

Также о высоком потенциале современных методов обучения с подкреплением свидетельствуют работы таких авторов как *P.C.Саттон [10], Д.Сильвер [11], А.И.Панов, Л. Чен [12]* и другие.

Для объективного анализа актуальности выбранной темы была произведена выборка статей, содержащих заданные ключевые слова: «Reinforcement Learning», «Reinforcement Learning» одновременно с «Robot» и, «Reinforcement Learning» одновременно с «Transformer». Выборка производилась в электронном архиве с открытым доступом для научных статей и рукописей arXiv. Результат анализа представлен на рисунке 1. Нелинейный рост публикаций в данной области подтверждает актуальность выбранной темы.

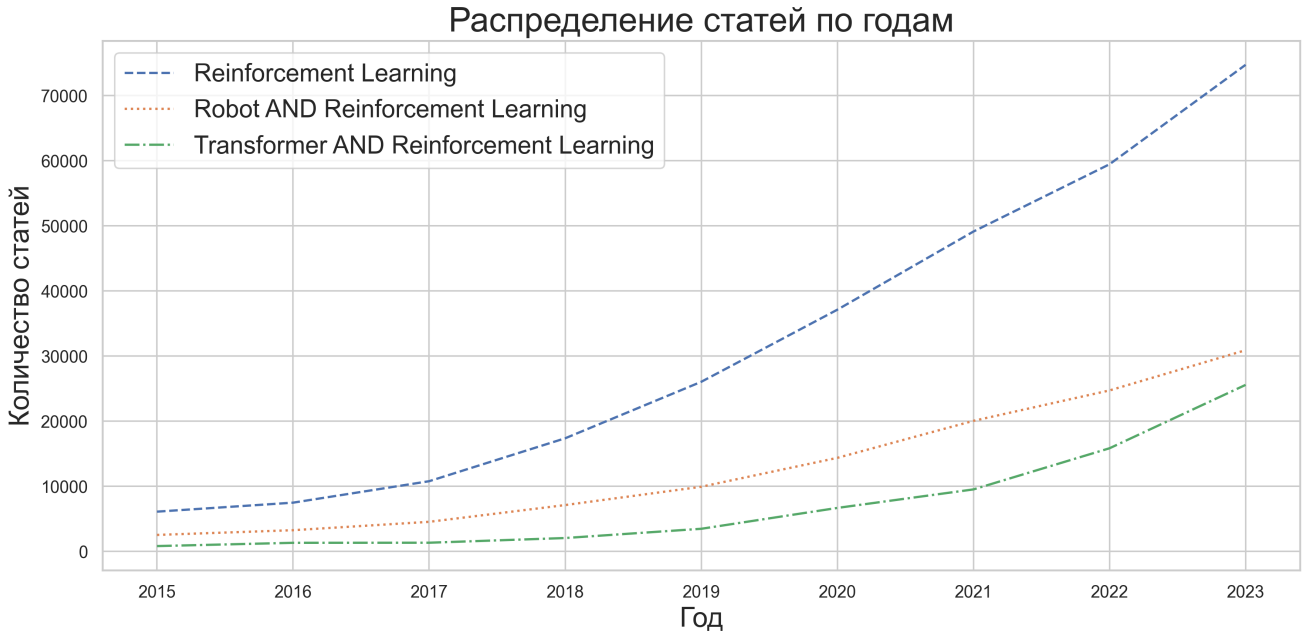

Рисунок 1 – Распределение статей с заданными ключевыми словами по годам

Цели и задачи исследования

Целью диссертационного исследования является разработка и исследование методов, алгоритмов и способов повышения качественных показателей алгоритмов обучения с подкреплением в рамках класса задач управления роботами, способными к перемещению в трехмерных средах. Для достижения указанной цели в диссертации решались следующие задачи:

1. Анализ лучших современных алгоритмов обучения с подкреплением с целью выявления их ограничений и особенностей использования в рассматриваемом классе задач.
2. Разработка модели интеграции алгоритмов обучения с подкреплением с кодировщиком трансформера, разработка и исследование нового алгоритма обучения с подкреплением, основанного на этой модели интеграции.
3. Разработка метода иерархического ансамблирования алгоритмов обучения с подкреплением, разработка и исследование нового алгоритма обучения с подкреплением, основанного на этом методе.

Методология и методы исследования

При проведении работы использовались методы машинного обучения, машинного обучения с подкреплением, разработки программного обеспечения.

Научная новизна

1. Разработана методика оценки влияния состава набора наблюдений окружающей среды на качество решений, принимаемых агентом, позволяющая упорядочить наблюдения по их полезности.
2. Предложена модель интеграции алгоритмов обучения с подкреплением и кодировщика трансформера для кодирования входных последовательностей состояний с целью повышения качества решения задачи.
3. Разработан алгоритм, интегрирующий кодировщик трансформера и алгоритм обучения с подкреплением Soft Actor-Critic.

4. Предложен метод иерархического ансамблирования алгоритмов обучения с подкреплением, который позволяет объединить несколько алгоритмов в иерархическую структуру для повышения качества обучения без дополнительных обращений к среде.
5. Разработан алгоритм обучения с подкреплением на основе предложенного метода иерархического ансамблирования с использованием алгоритма DQN в качестве управляющего и алгоритмов SAC и REDQ в качестве управляемых.

Практическая значимость

Разработанные решения улучшают качественные показатели обучения агентов, что позволяет их использовать для создания нового поколения роботов. Это расширяет области применения робототехнических систем и повышает их эксплуатационную надежность и эффективность.

На защиту выносятся

1. Метод иерархической интеграции ансамбля алгоритмов обучения с подкреплением, позволяющий объединить несколько алгоритмов в иерархическую структуру для повышения качества обучения без дополнительных обращений к среде. Доказана возможность повышения эффективности обучения за счет использования данной структуры по сравнению с отдельным использованием каждого алгоритма ансамбля.
2. Алгоритм обучения с подкреплением на основе предложенного метода иерархической интеграции, в котором алгоритм DQN используется в качестве управляющего, а алгоритмы SAC и REDQ — в качестве управляемых. Данный подход улучшает показатели качества обучения за счет распределения ролей среди алгоритмов.
3. Модель интеграции алгоритмов обучения с подкреплением и кодировщика трансформера, предназначенная для кодирования входных последовательностей состояний. Предложенная модель

улучшает качество решений задач за счет более эффективного представления информации об окружающей среде.

4. Алгоритм обучения с подкреплением, интегрирующий архитектуру трансформера в алгоритм Soft Actor-Critic для кодирования входных последовательностей состояний. Разработанный алгоритм демонстрирует улучшение результатов по сравнению с оригинальным алгоритмом Soft Actor-Critic.

Соответствие специальности

Диссертация соответствует паспорту научной специальности 1.2.1 – «Искусственный интеллект и машинное обучение» и охватывает следующие области исследования, входящие в эту специальность:

- Формализация и постановка задач управления и (поддержки) принятия решений на основе систем искусственного интеллекта и машинного обучения. Разработка систем управления с использованием систем искусственного интеллекта и методов машинного обучения в том числе – управления роботами, автомобилями, БПЛА и т.п.
- Исследования в области многослойных алгоритмических конструкций, в том числе – многослойных нейросетей.

Степень достоверности и апробация результатов

Достоверность научных результатов обеспечена применением методов статистического анализа, сравнением предложенных алгоритмов с существующими решениями и их экспериментальной проверкой на задачах управления роботами в трехмерных средах. Основные результаты научно-квалификационной работы были представлены на четырёх научных конференциях:

1. Международной конференции «Информационные технологии и нанотехнологии» (ИТНТ, Самара, Россия) - 2021 год;

2. Международной конференции «Информационные технологии и нанотехнологии» (ИТНТ, Самара, Россия) - 2022 год;

3. Международной конференции «Информационные технологии и нанотехнологии» (ИТНТ, Самара, Россия) - 2023 год;

4. Международной конференции «Информационные технологии и нанотехнологии» (ИТНТ, Самара, Россия) - 2024 год;

По теме диссертации опубликовано десять работ. Из них одна работа в изданиях, рекомендуемых ВАК, четыре работы опубликованы в изданиях, индексируемых в БД Scopus. Шесть работ выполнены без соавторов. Получено одно свидетельство Роспатента о регистрации программы для ЭВМ. [*13–22]

Результаты диссертационной работы:

1. Внедрены в рамках НИР в ООО «Давтех» в рамках договора №55/08/2023 от 01.08.2023.
2. Использованы в учебном процессе в ФГАОУ ВО «Самарский национальный исследовательский университет имени академика С. П. Королева» в курсе лекций по дисциплине «Машинное обучение и распознавание образов».
3. Использованы в рамках договора 7/2021 от 08.11.2021 (2021–2023) между АО «Самара-Информспутник» и ФГУП «ГосНИИПП».
4. Использованы в ФГАОУ ВО «Самарский национальный исследовательский университет имени академика С. П. Королева» в рамках гранта РНФ №. 21-11-00321, «Методы и алгоритмы совместного и координированного управления сигналами светофоров и подключенными автономными транспортными средствами в транспортной сети».

Структура диссертации

Диссертационная работа состоит из введения, четырех глав, заключения и списка литературы из 94 наименований. Работа содержит 98 страниц текста, включая 4 таблицы и 32 рисунка. В первой главе представлен обзор существующих технологий и их ограничений, во второй — описаны исследования проведенные с существующими методами и алгоритмами, в третьей – описан и исследован разработанная модель интеграции кодировщика трансформера в методы и алгоритмы обучения с подкреплением, в четвертой — описан и исследован разработанный ансамблевый метод обучения с подкреплением на основе иерархии.

РАЗДЕЛ 1. Обзор существующих решений

1.1 Определение и ключевые концепции обучения с подкреплением

Обучение с подкреплением (Reinforcement Learning, RL) представляет собой раздел машинного обучения, в котором агент осуществляет обучение на основе принципа максимизации совокупной награды, получаемой в результате взаимодействия с динамической средой (как показано на рисунке 2) [23]. В рамках обучения с подкреплением термин обучение интерпретируется как способность агента модифицировать своё поведение для достижения максимальной кумулятивной награды [24]. Основной задачей обучения с подкреплением является создание алгоритмов, которые обеспечивают возможность агентам самостоятельно разрабатывать оптимальные стратегии действий в сложной среде.

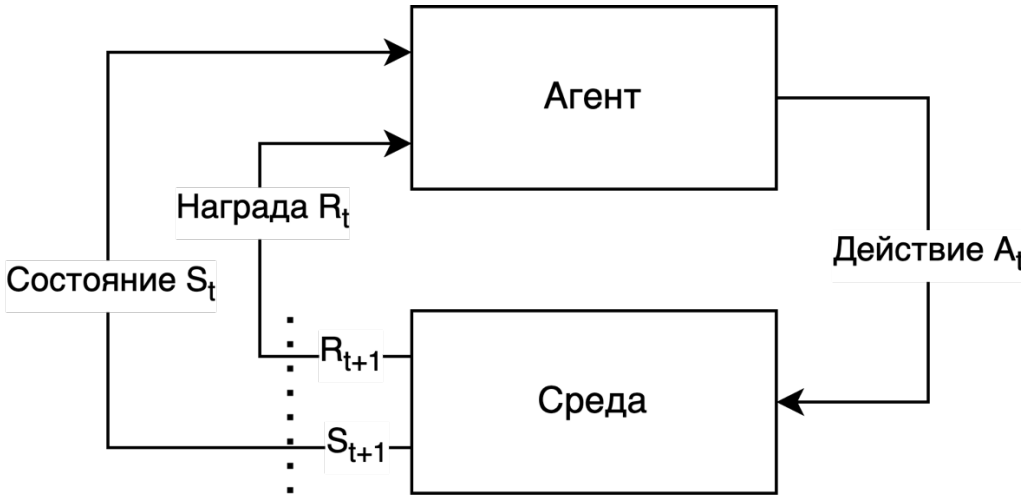

Рисунок 2 – Основная концепция обучения с подкреплением

Одним из основных понятий в обучении с подкреплением является концепция стратегии [10], определяющей план выбора действий агентом в зависимости от состояния среды. Стратегия может быть представлена в следующих формах:

– Детерминированной, когда каждому возможному состоянию среды ставится в соответствие определенное действие.

– Стохастической, когда действия выбираются согласно некоторому вероятностному распределению, что позволяет агенту исследовать среду и не застревать в локальных оптимумах.

1.2 История развития и ключевые этапы в области

История обучения с подкреплением берет свое начало в исследованиях оптимального управления и динамического программирования, которые были заложены Ричардом Беллманом в 1950-х годах. Методология Беллмана, основанная на принципе оптимальности, представляла собой попытку систематизировать процесс принятия решений в динамических системах, и легла в основу первых алгоритмов обучения с подкреплением [25]. Обучение с подкреплением уходит корнями в теорию обучения животных и используется для моделирования процессов принятия решений в нейробиологии. [26]

Прогресс в этой области стал особенно заметен с разработкой методов временных различий Ричардом Саттоном и Эндрю Барто в 1980-х годах [10]. Их работы ввели концепции, такие как Q-обучение и обучение с временными различиями, которые обеспечили основу для обучения агентов без явного моделирования среды.

В 1990-е годы исследования в области обучения с подкреплением ускорились благодаря интеграции с методами машинного обучения, в том числе с использованием нейронных сетей [27–29]. Это привело к созданию алгоритмов глубокого обучения с подкреплением, таких как Deep Q-Networks (DQN) [30-31], способных функционировать в условиях с высокой степенью неопределенности и сложности, например, таких как видеоигры, где количество возможных состояний и действий чрезвычайно велико [32].

Значимым моментом в истории развития обучения с подкреплением стала публикация исследований DeepMind в 2013 году, демонстрирующих способность алгоритма DQN справляться с играми Atari [33], исключительно на основе визуального ввода. Это подтвердило потенциальную применимость обучения с

подкреплением в широком спектре задач, от автономного вождения и робототехники до компьютерных игр с высокой степенью неопределенности [34-35].

Алгоритмы обучения с подкреплением продемонстрировали значительный прогресс, когда система AlphaGo [36], разработанная компанией DeepMind, одержала победу над чемпионом мира по игре го Ли Седолем в 2016 году. Этот успех стал важным событием в развитии искусственного интеллекта, показывая способность алгоритмов обучения с подкреплением решать задачи, требующие стратегического мышления и глубокого анализа. AlphaGo применял комбинацию глубокого обучения и методов обучения с подкреплением для анализа большого количества возможных позиций и выбора оптимальных стратегий.

Последующее развитие обучения с подкреплением включает в себя усовершенствования алгоритмов существующих алгоритмов и создание новых, таких как Proximal Policy Optimization (PPO) [37], Trust Region Policy Optimization (TRPO) [38] и Deep Deterministic Policy Gradient (DDPG) [39], которые улучшили стабильность и эффективность процесса обучения. Кроме того, последние достижения в области мультиагентного обучения с подкреплением [40] и реализация алгоритмов в реальном времени делают возможным развитие более сложных систем управления и взаимодействия.

1.3 Основные компоненты системы обучения с подкреплением

В контексте обучения с подкреплением система состоит из агента, среды, награды, стратегии и функции ценности.

Агент представляет собой сущность, которая выполняет действия в среде для достижения определённой цели. В математическом смысле агент может быть описан функцией стратегии $\pi(a|s)$, которая отображает состояния среды s в вероятности выбора действий a. Функция стратегии может быть как детерминированной, так и стохастической:

$$\pi(a|s) = P(A_t = a|S_t = s),$$

где P обозначает вероятность выбора действия a в состоянии s в момент времени t.

Среда — это динамическая система, с которой агент взаимодействует, получая информацию о текущем состоянии и наградах. Среда обычно моделируется как марковский процесс принятия решений (МППР) с заданными состояниями и вероятностями перехода между ними:

$$P(s'|s,a) = P(S_{t+1} = s'|S_t = s, A_t = a),$$

где s' — следующее состояние, s — текущее состояние, и a — действие, предпринятое агентом.

Награда — это сигнал, который агент получает после выполнения каждого действия, и который указывает на пользу или ценность результатов этого действия. Награды оцениваются функцией награды:

$$R(s,a) = E[R_{t+1}|S_t = s, A_t = a],$$

где R_{t+1} обозначает награду, полученную после перехода в новое состояние,

$E(X)$ – математическое ожидание величины X.

Стратегия — это план, согласно которому агент выбирает действия в каждом возможном состоянии. Стратегия может быть оптимизирована с целью максимизации суммарной ожидаемой награды на протяжении всего времени.

Функция ценности оценивает, насколько выгодно находиться в определенном состоянии s или паре состояние-действие (s,a). Существуют два основных типа функций ценности: функция ценности состояния $V(s)$ и функция ценности действия $Q(s,a)$, определённые как:

$$V^{\pi}(s) = E[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} |S_t = s, \pi],$$

$$Q^{\pi}(s,a) = E[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} |S_t = s, A_t = a, \pi],$$

где γ — коэффициент дисконтирования, который обеспечивает баланс между текущей и будущей наградой.

Таким образом получается, что агент использует стратегию для выбора действий в зависимости от состояния среды. Среда реагирует на действия агента, обновляя своё состояние и выдавая награды, которые агент использует для оценки и корректировки своей стратегии и функции ценности. Этот процесс происходит итеративно, позволяя агенту улучшать свою стратегию поведения с целью максимизации полной награды.

1.4 Марковские процессы принятия решений

Марковский процесс принятия решений (МППР) [41-42] представляет собой математическую модель, используемую для описания среды в контексте обучения с подкреплением. Основная особенность МППР заключается в том, что будущее состояние системы зависит только от текущего состояния и выполненного действия, что является формализацией марковского свойства, или отсутствия памяти. Это свойство упрощает анализ и понимание систем, поскольку для принятия решений не требуется информация о предыдущих состояниях, за исключением текущего.

МППР формально определяется как тройка (S, A, R), где:

- S — множество всех возможных состояний среды.
- A — множество всех возможных действий агента.
- R — функция вознаграждения $R(s, s')$, которая указывает награду, получаемую агентом при переходе из состояния s в состояние s'.

Функция перехода и функция вознаграждения вместе определяют динамику среды, в которой агент выполняет свои действия.

МППР предоставляет формализованный способ описания принятия решений в условиях неопределённости. В контексте обучения с подкреплением использование МППР позволяет рассматривать алгоритмы, которые могут систематически изучать и оптимизировать стратегию агента на основе получаемых наград и изменений состояний в среде. Каждое действие агента, основанное на

текущей стратегии, приводит к изменению состояния среды, и к получению награды, определенной функцией вознаграждения. Агенты используют эту информацию для обновления своих стратегий в направлении увеличения ожидаемой полной награды.

Компромисс между исследованием и эксплуатацией [43] является одним из ключевых аспектов в обучении с подкреплением. Он заключается в требовании нахождения баланса между изучением новых стратегий и использованием уже известных эффективных действий. Этот компромисс был наиболее тщательно изучен с помощью проблемы многорукого бандита [44-45] и марковских процессов принятия решений с конечным пространством состояний. Основной проблемой, которая возникает в связи с компромиссом между исследованием и эксплуатацией, является то, что неизвестно когда агент может применить изученное действие, а когда он должен предпринять действие, направленное на исследование среды. Если, например, речь о роботе, который учится ходить и его цель как можно быстрее достигнуть некоторой точки в пространстве, то успешным решением компромисса будет заключаться в том, что робот научится не просто ходить, а бегать, совершенствуя свои навыки путём исследования новых стратегий.

1.5 Уравнение Беллмана

Уравнение Беллмана [25] играет важную роль в теории динамического программирования и обучении с подкреплением, поскольку оно предоставляет рекурсивный метод вычисления оптимальной стратегии управления. Это уравнение было введено Ричардом Беллманом и стало основой для разработки алгоритмов обучения с подкреплением.

Уравнение Беллмана описывает связь между значением текущего состояния и значениями возможных последующих состояний. Оно может быть выражено для детерминированных и стохастических стратегий. В контексте функции ценности $V(s)$ для стратегии π, уравнение принимает следующий вид:

$$V^{\pi}(s) = \sum_{a \in A} \pi(a|s)[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a)V^{\pi}(s')]$$

где:

- $V^{\pi}(s)$ — функция ценности состояния s при следовании стратегии π,
- $\pi(a|s)$ — вероятность выбора действия a в состоянии s по стратегии π,
- $R(s,a)$ — функция вознаграждения после выполнения действия a в состоянии s,
- γ — коэффициент дисконтирования, который отражает важность будущих вознаграждений,
- $P(s'|s,a)$ — вероятность перехода в состояние s' из состояния s после выполнения действия a,
- S — множество всех возможных состояний,
- A — множество всех возможных действий.

Для определения оптимальной функции ценности $V^{*}(s)$, уравнение Беллмана модифицируется для учета максимально возможного вознаграждения:

$$V^{*}(s) = max_{a \in A}[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a)V^{*}(s')]$$

Это уравнение показывает, что оптимальное значение функции ценности для состояния s соответствует максимальному вознаграждению, которое можно получить, выбрав наилучшее действие a, учитывая текущее состояние и ожидаемое будущее значение.

Уравнения Беллмана обеспечивают теоретическую основу для большинства алгоритмов обучения с подкреплением, включая Q-обучение и методы временных различий. Они позволяют агентам оценивать и оптимизировать свои стратегии в сложных и динамически изменяющихся средах.

1.6 Классификация и обзор алгоритмов обучения с подкреплением

Обучение с подкреплением представляет собой мощный класс алгоритмов машинного обучения, позволяющих агентам автономно оптимизировать свои стратегии поведения через взаимодействие с динамической средой. В этом разделе будут классифицированы алгоритмы обучения с подкреплением на основе их подходов к моделированию и решению задач, что включает в себя разделение на модельные и безмодельные алгоритмы, а также разделение на методы, основанные на значении и основанные на стратегии.

1.6.1 Модельные и безмодельные алгоритмы

Модельные алгоритмы используют или стремятся построить модель среды, с которой взаимодействует агент. Эта модель может быть использована для симуляции и планирования, позволяя агенту предсказать последствия действий перед их выполнением. Примером модельного подхода может служить Dyna-Q [46], который комбинирует прямое обучение с обучением на основе симулированного опыта взаимодействия со средой, полученным из модели:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[R_{t+1} + \gamma max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

где $Q(s, a)$ — функция ценности действия a в состоянии s , α — скорость обучения, R_{t+1} — награда, полученная после выполнения действия, и γ — коэффициент дисконтирования.

Безмодельные алгоритмы, в отличие от модельных, не стремятся построить явное представление о динамике среды, а напрямую оптимизируют свои действия на основе полученного опыта. Примером безмодельного подхода является Q-learning, который обновляет свои оценки функции ценности действий на основе наград и переходов, наблюдаемых в процессе взаимодействия со средой:

$$Q(s_t, a_t) \leftarrow (1 - \alpha)Q(s_t, a_t) + \alpha[r_t + \gamma max_{a'} Q(s_{t+1}, a')]$$

1.6.2 Основанные на значении и основанные на стратегии алгоритмы

Основанные на значении алгоритмы фокусируются на определении функции ценности оптимального действия для каждого состояния. Примером такого подхода является уже упомянутый Q-learning, где оптимальные действия определяются через функцию $Q(s, a)$, максимизирующую ожидаемые награды.

Основанные на стратегии алгоритмы напрямую оптимизируют стратегию агента без явного определения функции ценности. Один из популярных методов в этой категории — Policy Gradient, где стратегия $\pi_\theta(a|s)$ параметризирована и оптимизируется для максимизации функции награды $J(\theta)$:

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\nabla_\theta log \pi_\theta(a|s) Q^\pi(s, a)]$$

где ∇_θ означает градиент по параметрам стратегии θ.

1.6.3 Q-learning

Q-learning является классическим примером безмодельного алгоритма обучения с подкреплением, основанного на значении. Алгоритм стремится научить агента оценивать, насколько полезно выполнение каждого возможного действия в каждом возможном состоянии. Центральным элементом Q-learning является функция ценности действия, $Q(s, a)$, которая представляет собой ожидаемую сумму дисконтированных наград, получаемых после выбора действия a в состоянии s.

Формула обновления в Q-learning:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma max_{a'} Q(s', a') - Q(s, a)]$$

где:

α — скорость обучения (англ. learning rate),

r — награда, полученная за выполнение действия a в состоянии s,

γ — коэффициент дисконтирования будущих наград,

$max_{a'} Q(s', a')$ — максимальная оценка ценности действия в новом состоянии s'.

Данное соотношение позволяет агенту итеративно улучшать оценки ценности действий на основе непосредственного опыта взаимодействия со средой, способствуя развитию стратегии, которая может максимизировать кумулятивные награды.

1.6.4 Deep Q-Networks (DQN)

Deep Q-Networks (DQN) расширяют идею Q-learning за счет интеграции глубоких нейронных сетей, что позволяет обрабатывать более сложные среды с высокой размерностью состояний. В DQN, функция $Q(s, a)$ аппроксимируется сетью, что позволяет обобщать оценку ценности действий на новые, ранее не встречаемые состояния.

Формула обновления для DQN включает минимизацию функции потерь (loss function), вычисляемой как среднеквадратичная ошибка между текущими оценками сети и целевыми значениями, которые рассчитываются с использованием принципа оптимальности Беллмана:

$$L(\theta) = E[\left(r + \gamma max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)\right)^2]$$

где θ — параметры текущей сети, а θ^- — параметры целевой сети, которая регулярно обновляется значениями из θ, обеспечивая стабильность обучения.

1.6.5 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) [37] — это один из наиболее популярных алгоритмов обучения с подкреплением, основанных на стратегии. PPO был разработан для улучшения стабильности и надежности обучения по сравнению с другими алгоритмами градиентов стратегии, такими как TRPO (Trust Region Policy Optimization). Основное преимущество PPO заключается в его способности достигать хороших результатов с относительно простой реализацией и меньшими вычислительными затратами.

Формула обновления для PPO использует функцию потерь, которая ограничивает изменения в стратегии для предотвращения слишком крупных обновлений, которые могут нарушить процесс обучения:

$$L(\theta_i) = E_t[\min(r_t(\theta_i)\hat{A}_t, clip(r_t(\theta_i), 1-\epsilon, 1+\epsilon)\widehat{A_t})],$$

где $r_t(\theta)$ — отношение новой стратегии к старой, $\widehat{A_t}$ — преимущество действия, а ϵ — гиперпараметр, задающий степень обрезки.

1.6.6 Deep Deterministic Policy Gradient (DDPG)

Deep Deterministic Policy Gradient (DDPG) представляет собой алгоритм, сочетающий идеи Q-learning и градиентов стратегии. DDPG особенно полезен в средах с непрерывным пространством действий, что делает его подходящим для задач, таких как управление роботами. DDPG использует две основные сети: сеть актора для определения действия и сеть критика для оценки этого действия.

Формула обновления для DDPG включает следующий механизм обучения:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\big[r + \gamma Q\big(s', \mu(s';\theta^{\mu})\big) - Q(s,a)\big],$$

где $\mu(s;\theta^{\mu})$ — детерминированная стратегия актера, оптимизируемая для максимизации оценки критика.

1.6.7 Policy Gradients

Policy Gradient [47] (методы градиента стратегии) методы оптимизируют параметризированную стратегию напрямую посредством градиентного спуска на ожидаемой сумме дисконтированных наград. Эти методы обычно используют стохастические стратегии, что позволяет проводить эффективный поиск в пространстве возможных действий и избегать локальных минимумов.

Формула обновления для градиентов стратегии:

$$\nabla_{\theta} J(\theta) = E\big[\nabla_{\theta} \log \pi_{\theta}(a|s) Q^{\pi(s|a)}\big],$$

где $\nabla_{\theta} J(\theta)$ — градиент меры качества стратегии по параметрам стратегии θ , $\pi_{\theta}(a|s)$ — стохастическая стратегия, зависящая от параметров θ.

1.6.8 Actor-Critic

Actor-Critic методы сочетают идеи, заложенные в методах градиента стратегии и методах, основанных на значении, для обучения с подкреплением. Эти алгоритмы используют две основные компоненты: *актор*, который определяет выбор действия, и *критик*, который оценивает действие, предложенное *актор* посредством оценки функции ценности.

Структура Actor-Critic:

- Актор: определяет стратегию действий $\pi(a|s;\theta)$, которая указывает, какое действие следует предпринять в данном состоянии. Стратегия моделируется с помощью нейронной сети с параметрами θ.
- Критик: оценивает, насколько хорошо было выбранное действие с помощью функции ценности V(s; ω) или Q(s, a; ω) , где ω — параметры сети оценщика.

Обновление параметров в Actor-Critic методах происходит через следующие шаги:

1. Критик обновляет оценку функции ценности, используя среднеквадратичную ошибку между оцененным и фактическим вознаграждениями:

$$L(\omega) = \big(r + \gamma V(s';\omega) - V(s;\omega)\big)^2$$

где r — вознаграждение за переход из состояния s в s' при действии a.

2. Актор обновляет стратегию, используя градиентный спуск для максимизации меры качества:

$$\nabla_\theta J(\theta) = \nabla_\theta \log \pi(a|s;\theta) \cdot Q(s,a;\omega)$$

или с использованием функции преимущества $A(s,a;\omega,\theta) = Q(s,a;\omega) - V(s;\omega)$:

$$\nabla_\theta J(\theta) = \nabla_\theta \log \pi(a|s;\theta) \cdot A(s,a;\omega,\theta)$$

Это сочетание позволяет Actor-Critic методам учиться более стабильно, по сравнению с методами, основанными только на стратегии, поскольку оценка критика помогает уменьшить дисперсию обновлений, которые получает актор.

1.6.9 Soft Actor-Critic (SAC)

Soft Actor-Critic (SAC) [48] — это относительно новый и продвинутый вариант Actor-Critic алгоритмов, который внедряет дополнительные понятия энтропийной регуляризации для улучшения исследовательского поведения агента. Этот подход не только улучшает производительность алгоритма в условиях неопределенности, но и способствует более широкому исследованию пространства действий, что важно в сложных средах.

Основная формула обновления для SAC включает максимизацию следующей целевой функции:

$$J(\pi) = E_{s_t,a_t\sim\pi}[r_t + \gamma(V(s_{t+1}) - \alpha \log \pi(a_t|s_t))],$$

где α — коэффициент температуры, который контролирует компромисс между эксплуатацией полученных знаний и исследованием новых действий.

SAC демонстрирует высокую производительность в ряде задач, от игр до реальных робототехнических применений, благодаря своей способности эффективно балансировать между исследованием и эксплуатацией.

1.6.10Randomized Ensembled Double Q-Learning (REDQ)

Метод Randomized Ensembled Double Q-Learning (REDQ) [49] предназначен для улучшения стабильности и эффективности обучения с подкреплением посредством использования ансамбля Q-функций. Основная идея заключается в применении случайных подмножеств ансамбля Q-функций на каждом шаге обновления, что снижает вычислительную сложность и улучшает устойчивость обучения.

Для реализации REDQ создается ансамбль из N Q-функций, обозначенных как $(Q_{\theta_1}, Q_{\theta_2}, \dots, Q_{\theta_N})$, где θ_i — параметры Q-функции. На каждом шаге обучения выбирается случайное подмножество из M Q-функций, которые используются для обновления.

Алгоритм REDQ включает следующие шаги:

1. Для каждого шага обучения случайным образом выбирается подмножество M Q-функций из ансамбля.

2. Для каждой пары (s, a) состояния и действия вычисляется минимальное значение среди выбранных Q-функций:

$$Q_{min}(s, a) = \min_{i \in M} Q_{\theta_i}(s, a),$$

где M — индексное множество выбранных Q-функций.

3. Обновление Q-функций выполняется с использованием двойного Q-обучения:

$$y = r + \gamma\, Q_{\theta'_{min}}(s', \pi_\phi(s')),$$

где θ'_{min} — параметры Q-функции с минимальным значением, π_ϕ — стратегия агента, γ — коэффициент дисконтирования, s' — следующее состояние, а r — полученная награда.

4. Обновление параметров Q-функций выполняется путем минимизации среднеквадратичной ошибки:

$$\theta_i \leftarrow \theta_i - \alpha \nabla_{\theta_i} \big(Q_{\theta_i}(s, a) - y\big)^2,$$

где α — скорость обучения.

REDQ демонстрирует высокую эффективность на различных задачах и достигает более быстрой и стабильной сходимости по сравнению с традиционными методами Q-обучения благодаря ансамблевому подходу и технике двойного Q-обучения.

1.6.11Выводы на основе представленного обзора существующих методов

Наиболее актуальными методами обучения с подкреплением, хорошо себя проявляющими в задачах с непрерывными пространствами состояний и действий высокой размерностью считаются такие методы как PPO, SAC, REDQ.

PPO отличается простотой реализации, а также эффективностью в различных задачах, но может страдать от проблем с выбором подходящих гиперпараметров для обрезки. DDPG обеспечивает эффективное решение для сред с непрерывным

пространством действий, но требует тщательного подбора параметров и может страдать от нестабильности в некоторых задачах.

Soft Actor-Critic (SAC), с другой стороны, предлагает баланс между стабильностью и эффективностью за счет внедрения энтропийной регуляризации, что делает его одним из наиболее робастных алгоритмов для обучения с подкреплением в сложных средах. SAC обеспечивает хорошую производительность при обучении и хорошую сходимость во многих задачах, что объясняет его популярность в современных исследованиях и приложениях.

В данном подразделе был проведен обзор и классификация алгоритмов обучения с подкреплением, которые используются для автоматизации и оптимизации процесса принятия решений в сложных и динамичных средах. Была приведена классификация алгоритмов на модельные и безмодельные алгоритмы, и на алгоритмы, основанные на значении и алгоритмы, основанные на стратегии. Были рассмотрены ключевые алгоритмы, такие как Q-learning, Deep Q-Networks (DQN), Policy Gradients, Actor-Critic, и Soft Actor-Critic (SAC), Randomized Ensembled Double Q-Learning (REDQ) а также другие значимые алгоритмы, включая Proximal Policy Optimization (PPO) и Deep Deterministic Policy Gradient (DDPG). В таблице 1 отображена краткая информация об актуальных алгоритмах обучения с подкреплением.

Таблица 1 – Современные актуальные методы

Метод/ Алгоритм	Тип RL	Основная идея/характеристика	Применение	Год
Deep Q-Network (DQN)	Оффлайн Q-обучение	Использует глубокие нейросети для аппроксимации Q-функции ценности.	Игровые среды, такие как Atari	2015
Trust Region Policy Optimization (TRPO)	Градиент стратегии	Ограничивает шаги обновления политики с помощью метода доверительных регионов	Робототехника, системы с непрерывными действиями	2015

Метод/ Алгоритм	Тип RL	Основная идея/характеристика	Применение	Год
Deep Deterministic Policy Gradient (DDPG)	Оффлайн, Градиент стратегии	Адаптирует Actor-Critic для задач с непрерывными пространствами действий	Управление роботами, автомобильные симуляции	2016
AlphaZero	Моделирование динамики среды	Комбинирует глубокое обучение и алгоритм Монте-Карло для поиска и обучения	Настольные игры, шахматы, го	2017
Proximal Policy Optimization (PPO)	Градиент стратегии	Ограничивает обновления политики с помощью KL-дивергенции для повышения стабильности обучения	Робототехника, управление сложными системами	2017
Soft Actor-Critic (SAC)	Оффлайн, энтропийная регуляризация	Максимизирует энтропию политики для улучшения стабильности и исследовательской способности	Робототехника, симуляции с непрерывными действиями	2018
MuZero	Моделирование динамики среды	Изучает оптимальную политику без знания динамики среды, комбинирует обучение модели и планирование	Игровые среды (Go, Chess, Atari)	2019
Offline Reinforcement Learning (CQL)	Оффлайн	Цель — улучшение эффективности offline RL с помощью консервативного Q-learning	Управление роботами, рекомендательные системы	2020
Behavioral Cloning Transformer (BC-Transformer)	Имитационное обучение	Применяет архитектуру трансформера для имитационного обучения	Игровые среды, задачи с большим объемом данных	2021

Метод/ Алгоритм	Тип RL	Основная идея/характеристика	Применение	Год
Randomized Ensembled Double Q-learning (REDQ)	Off-policy Q-обучение	Использует несколько Q-функций для улучшения стабильности и сходимости	Робототехника, задачи с высокой стохастичностью	2021
Decision Transformer	Моделирование последователь-ностей	Применяет архитектуру трансформера для обучения с подкреплением, формулируя задачу как последовательное моделирование	Игровые среды, управление роботами	2021

Было выявлено, что каждый алгоритм имеет свои сильные и слабые стороны в зависимости от конкретных требований задачи и условий ее реализации:

- Q-learning и DQN отлично справляются с задачами, где пространство состояний и действий дискретно, но могут страдать от большого количества степеней свободы в более сложных средах.
- Policy Gradients и Actor-Critic методы предоставляют мощные инструменты для работы с непрерывными действиями и стратегиями, но требуют тщательного подбора параметров и могут быть подвержены высокой дисперсии в оценках.
- SAC обеспечивает высокую производительность и стабильность за счет введения энтропийной регуляризации, делая его предпочтительным выбором для многих современных приложений, особенно в робототехнике.

1.7 Применение обучения с подкреплением в робототехнике

Обучение с подкреплением представляет значительный интерес для области робототехники, поскольку предлагает мощные методы для разработки автономных

систем, способных адаптироваться к динамическим и часто непредсказуемым средам. В последние годы обучение с подкреплением применяется для решения ряда задач в робототехнике [50–53], включая навигацию, манипуляцию и взаимодействие с окружающей средой, что подтверждает его потенциал в создании интеллектуальных и адаптивных робототехнических систем. Помимо робототехники обучение с подкреплением находит применение в области здравоохранения, рекомендательных систем, многоагентных транспортных систем [54–57].

В контексте робототехники обучение с подкреплением часто используется для решения задачи навигации. Навигация включает в себя способность робота самостоятельно перемещаться в пространстве, избегая препятствий и оптимизируя свой маршрут. Использование обучения с подкреплением в навигации роботов [58-59] основывается на формировании стратеги, которые могут адаптироваться к новым и изменяющимся условиям без прямого программирования каждого возможного сценария. Это достигается за счет обучения оптимальной стратегии поведения, основанной на накопленном опыте взаимодействия с окружающей средой.

Обучение с подкреплением также находит применение в роботизированной манипуляции [54], позволяя роботам обучаться выполнению задач, связанных с захватом, перемещением и сборкой объектов. В таких задачах алгоритмы обучения с подкреплением могут быть использованы для разработки стратегий, которые динамически адаптируются к изменениям в свойствах объектов или окружающей среде.

Обучение с подкреплением хорошо подходит для задач, где роботы должны взаимодействовать с динамически изменяющимися и непредсказуемыми средами. Это может включать сценарии, где роботы работают совместно с людьми или другими роботами, и требуется высокий уровень адаптации и безопасности [6].

1.7.1 Специфические сложности и требования

Применение алгоритмов обучения с подкреплением в робототехнике накладывает определенные требования, которые важно учитывать при разработке и внедрении роботизированных систем. Эти требования касаются как технических аспектов, так и высоких стандартов надежности и безопасности, особенно когда речь идет о взаимодействии с операторами людьми или об использовании в публичных местах.

Роботы часто оперируют в динамических, быстро меняющихся средах, где решения нужно принимать и исполнять в реальном времени. Алгоритмы обучения с подкреплением должны быть способны не только быстро адаптироваться к изменениям в среде, но и гарантировать стабильное и предсказуемое поведение в любой ситуации.

Одним из подходов к удовлетворению этих требований является использование алгоритмов с низкой вычислительной сложностью во время исполнения или разработка методов, которые могут эффективно обрабатывать входные данные в параллельных вычислительных системах. Например, методы, основанные на нейросетевых архитектурах, таких как сверточные нейронные сети (CNN), могут обрабатывать многомерные данные быстро и эффективно.

В контексте робототехники, особенно когда речь заходит о медицинских приложениях или роботах-компаньонах, надежность и безопасность являются критически важными. Роботы должны быть способны не только правильно интерпретировать ситуацию и принимать адекватные решения, но и гарантировать безопасность людей вокруг.

Решение проблемы безопасности часто включает внедрение механизмов контроля, таких как безопасные стратегии или процедуры аварийного отключения, которые могут активироваться при обнаружении потенциально опасных ошибок в работе алгоритма. Также важно рассматривать возможности формальной верификации алгоритмов, чтобы убедиться в их корректности по отношению к поставленным требованиям безопасности.

Перенос решений, разработанных и проверенных в контролируемых или симулированных средах, в реальный мир является сложной задачей. Реальные условия часто намного сложнее и непредсказуемее, чем лабораторные условия.

Один из способов решения проблемы масштабирования заключается в использовании обучения с подкреплением в сочетании с передачей знаний (transfer learning), где предварительно обученные модели адаптируются к новым условиям с минимальными дополнительными затратами на обучение. Предварительно обученную модель можно получить, например, в симуляторе [60]. Это может существенно ускорить процесс внедрения и обеспечить лучшую адаптацию к разнообразным условиям эксплуатации.

1.8 Разработка сред и формулировка функции наград в обучении с подкреплением для решения реальных задач

В обучении с подкреплением создание эффективной среды и точная формулировка функции награды являются ключевыми аспектами для успешного обучения агентов. Они играют решающую роль в определении того, как агенты учатся взаимодействовать с окружающей средой и достигать желаемых результатов.

Среда в обучении с подкреплением предоставляет контекст, в котором агенты исследуют последствия своих действий и получают обратную связь в виде наград.

Разработка реалистичной среды требует учета следующих факторов:

1. Реалистичность. Среда должна адекватно моделировать реальные условия задачи, с которыми агент будет сталкиваться в практических сценариях. Это включает в себя физические законы, динамические ограничения и случайные факторы, влияющие на исходы действий агента.
2. Наблюдаемость. Среда должна обеспечивать агенту доступ ко всей необходимой информации о текущем состоянии. Это может включать

состояния датчиков, изображения или другие формы данных, которые агент может использовать для принятия решений.

3. Взаимодействие. Механизмы взаимодействия агента со средой должны быть четко определены, включая возможные действия и их влияние на состояние среды.

Функция награды сильно влияет на процесс обучения с подкреплением [61], поскольку она управляет процессом обучения агента, указывая, какие действия считаются предпочтительными. Разработка эффективной функции награды требует рассмотрения следующих аспектов [62]:

1. Выражение целей. Награда должна точно отражать цели задачи. Например, в задаче навигации робота награда может быть выражена через минимальное время достижения цели или максимальное расстояние, пройденное без столкновений.
2. Балансировка. Избегание чрезмерных или недостаточных наград за действия, которые могут привести к нежелательному поведению, такому как застревание в локальных оптимумах или избыточное исследование.
3. Спецификация. Функция награды должна быть достаточно специфичной, чтобы обеспечить четкое различие между полезными и вредными действиями агента в среде.

Примеры формулировки функции наград:

– Робототехника (манипуляция):

$$R(s,a) = \begin{cases} +100, & \text{если объект собран} \\ -1, & \text{за каждое движение} \\ -10, & \text{если происходит столкновение} \end{cases}$$

Эта функция награды поощряет быстрое выполнение задачи с минимальным количеством движений и штрафует за столкновения.

– Автономные транспортные средства:

$$R(s,a) = v_t - \lambda \sum \text{штрафы за нарушения ПДД},$$

где v_t — скорость движения транспортного средства, а λ — коэффициент, регулирующий важность соблюдения правил дорожного движения.

Разработка среды и формулировка функции награды являются фундаментальными аспектами создания эффективных решений в области обучения с подкреплением. Поэтому важно также рассмотрение методов, способных выполнять процесс обучения с подкреплением без учителя, то есть взаимодействия с миром без сигнала награды, для более быстрой адаптации к будущим задачам [63]. Понимание и правильное применение этих компонентов в реальных задачах позволяет значительно повысить производительность и адаптивность робототехнических систем и других приложений, где используется обучение с подкреплением.

1.9 Выводы и результаты первого раздела

Применение обучения с подкреплением в робототехнике предоставляет значительные возможности для развития более интеллектуальных, адаптивных и эффективных робототехнических систем. Несмотря на различные трудности, связанные с надежностью, безопасностью и требованиями к реальному времени, продолжающееся развитие и усовершенствование алгоритмов обучения с подкреплением позволяет успешно преодолевать эти препятствия.

Использование обучения с подкреплением в робототехнике продемонстрировало его эффективность в решении сложных задач, включая навигацию, манипуляцию и взаимодействие с динамическими средами. Особенно значимым стало внедрение таких алгоритмов, как SAC, REDQ и PPO, которые обеспечивают высокую степень адаптации и оптимизации поведения роботов в условиях, где традиционные программные подходы оказываются неэффективными.

В дальнейшем развитии обучения с подкреплением в робототехнике важное значение будет иметь интеграция с другими технологиями искусственного интеллекта, такими как машинное зрение и естественный язык, что позволит создавать мультифункциональные роботизированные системы, способные

взаимодействовать с людьми и другими роботами в еще более сложных сценариях. Кроме того, значительное внимание следует уделить улучшению алгоритмов в контексте их способности к быстрому обучению в реальных условиях без значительных предварительных затрат на симуляции.

Важными направлениями для будущих исследований остаются:

- Улучшение алгоритмов обучения с подкреплением для повышения их эффективности, скорости и адаптивности к изменяющимся условиям.
- Разработка безопасных и надежных систем управления для роботов, обеспечивающих безопасность использования в общественных и частных пространствах.
- Исследование взаимодействия человека и робота с целью создания интуитивно понятных и дружелюбных интерфейсов для широкого круга пользователей.

Обучение с подкреплением остается одной из перспективных областей исследований в робототехнике, способной радикально изменить подходы к разработке и использованию роботизированных систем. Продолжающиеся исследования и разработки в этом направлении не только улучшают технические характеристики роботов, но и способствуют более глубокому пониманию процессов автоматизации и взаимодействия машин с окружающим миром.

РАЗДЕЛ 2. Исследование существующих методов обучения с подкреплением

2.1 Введение в экспериментальную конфигурацию

В данном подразделе описывается экспериментальная конфигурация, применяемая для исследования процессов обучения с подкреплением в задачах управления движением автономных агентов в трехмерном пространстве. Основной акцент сделан на использование передовых технологий машинного обучения и компьютерного моделирования для разработки, тестирования и сравнения различных алгоритмов, в виртуальных средах.

Эксперименты в контролируемых виртуальных средах позволяют изучать основные принципы обучения и взаимодействия агентов с окружением, тестировать гипотезы о влиянии различных переменных на эффективность обучения, и тем самым обеспечивают глубокое понимание механизмов обучения. Это ключевое знание необходимо для разработки надежных и эффективных систем автономного управления.

В следующих разделах будет представлен подробный обзор инструментов и методик, применяемых в исследовании, включая обсуждение программного обеспечения, языков программирования, спецификаций симуляционных сред и параметров эксперимента.

2.1.1 Программное обеспечение и языки программирования

В данном исследовании был выбран Python в качестве основного языка программирования на всех этапах разработки. Выбор Python обоснован его широкой поддержкой в академическом сообществе и наличием многочисленных библиотек, специализированных для применения в машинном обучении и обучении с подкреплением.

Python отличается высокой читаемостью и модифицируемостью, что обеспечивает быстрое прототипирование. Для обеспечения совместимости в исследовании использовалась версия Python 3.10.

2.1.2 Библиотека PyTorch

В процессе проведения экспериментов по обучению с подкреплением широко применяется библиотека машинного обучения PyTorch [64]. Эта библиотека обеспечивает необходимые инструменты для конструирования и обучения нейронных сетей, что имеет ключевое значение для эффективности указанных алгоритмов.

Преимуществом PyTorch является его способность к динамическому построению графа, позволяющая оперативно адаптировать процесс обучения нейронных сетей, включая возможность корректировки и оптимизации в процессе выполнения. Эта функциональность оказывается особенно значимой при анализе новых концепций в области алгоритмов управления и обучения.

2.1.3 Библиотека TorchRL

Кроме PyTorch, значительное применение в обучении с подкреплением находит библиотека TorchRL [65]. Она представляет собой модуль, ориентированный на реализацию алгоритмов обучения с подкреплением, и является расширением функциональности PyTorch.

В TorchRL включен обширный набор стандартных алгоритмов обучения с подкреплением, что облегчает процесс разработки и проведения экспериментов. Библиотека поддерживает как традиционные, так и современные методы, ускоряя научно-исследовательскую деятельность.

Преимущество TorchRL заключается в её полной интеграции с PyTorch, что позволяет использовать все функции основной библиотеки, включая автоматическое дифференцирование и эффективную работу с тензорами. Это

обеспечивает удобство в построении и оптимизации моделей обучения с подкреплением.

2.1.4 Интеграция с Gymnasium

Библиотека Gymnasium [66] широко используется в исследованиях обучения с подкреплением за счёт своего универсального набора сред, предназначенных для тестирования алгоритмов. Особенное внимание заслуживает интеграция с MuJoCo — физическим движком, предназначенным для моделирования и точной симуляции динамических систем с множеством степеней свободы. Эта библиотека создана на основе OpenAI Gym [67].

MuJoCo [68], входящий в состав сред Gymnasium, обеспечивает высокую точность симуляций, что критически важно для обучения алгоритмов, требующих детализированного воспроизведения физических взаимодействий и механик. Это особенно значимо при работе с робототехникой и другими комплексными динамическими системами.

Преимущество использования MuJoCo в контексте Gymnasium заключается в его способности к быстрой и эффективной симуляции, что позволяет значительно ускорить процесс обучения моделей. MuJoCo поддерживает расширенные возможности для обработки коллизий, мягкой динамики и точного расчёта сил, что делает его незаменимым инструментом в экспериментальной физике и инженерии.

Также важно отметить, что Gymnasium с MuJoCo предлагает гибкость в настройке сред и параметров симуляции, облегчая адаптацию экспериментов под специфические научные задачи и исследовательские интересы. Это способствует более глубокому пониманию основных принципов управления и взаимодействия в сложных системах.

2.1.5 Интеграция с ML-Agents

ML-Agents [69] представляет собой пакет для Unity, разработанный компанией Unity Technologies, который позволяет разработчикам легко внедрять и

тестировать алгоритмы машинного обучения в виртуальных средах. Этот инструмент особенно полезен для обучения с подкреплением, так как он предоставляет гибкую и мощную среду для быстрой итерации и тестирования различных подходов.

ML-Agents позволяет создавать сложные сценарии, в которых агенты могут взаимодействовать с окружающим миром и друг с другом. Этот пакет включает в себя симуляцию физических взаимодействий, таких как столкновения и трение, что делает его идеальным инструментом для экспериментов в робототехнике и других областях, где может быть полезна симуляция физических процессов. Также ML-Agents тесно интегрирован с такими библиотеками, как PyTorch и TensorFlow, что позволяет использовать продвинутые алгоритмы обучения прямо в Unity без необходимости переноса моделей между различными платформами. Такая интеграция упрощает процесс обучения, позволяя разработчикам сосредоточиться на архитектуре модели и экспериментальных настройках, а не на технических аспектах интеграции.

2.1.6 Интеграция с ROS

Для построения гибкой системы управления роботом зачастую используется ROS (Robot Operation System) [70]. ROS – представляет собой набор программных решений, библиотек и инструментов разработчика с открытым исходным кодом для создания робототехнических приложений. ROS предоставляет разработчикам такие возможности, как низкоуровневое управление устройствами, высокоуровневые абстракции и межпроцессное взаимодействие.

Основной особенностью ROS является модульность и масштабируемость, это достигается за счёт реализации всей системы в виде набора узлов, каждый из которых решает отдельную задачу. При этом узлы могут обмениваться информацией посредством публикации сообщений определенного формата. Другие узлы могут подписываться на эти публикации.

Также можно выделить такой важный аспект ROS как большое количество различных готовых к использованию инструментов и библиотек для различных задач.

ROS 2 [71] является эволюцией оригинального ROS. ROS 2 предлагает улучшенную архитектуру, повышенную надежность и более широкие возможности для применения в промышленных и критически важных системах.

Одним из важнейших инструментов ROS и ROS 2 является симулятор робототехнических систем Gazebo [72]. Он позволяет моделировать физическую среду, поведение роботов и их взаимодействие с окружающим миром.

Gazebo обладает следующими особенностями:

- Симуляция физики. Gazebo использует современные физические движки, такие как ODE [73], Bullet [74] и DART [75], для симуляции движений, столкновений и других физических взаимодействий. Это позволяет симулировать реальное поведение роботов в сложных средах.
- В Gazebo можно моделировать различные сенсоры, такие как камеры, лидары, GPS, IMU и другие, что позволяет тестировать алгоритмы обработки данных в симуляции до их применения на реальном роботе.
- Gazebo позволяет создавать и настраивать сложные симуляционные среды, включая ландшафты, здания, объекты и т. д. Эти среды могут быть использованы для тестирования роботов в условиях, которые сложно воспроизвести в реальной жизни.
- Gazebo тесно интегрирован с ROS, что делает его мощным инструментом для разработки и тестирования программного обеспечения для роботов. Через ROS можно управлять роботами в симуляции, получать данные с их сенсоров и отлаживать поведение системы.

Таким образом, возможно создать модель робота, повторяющую реального как с физической точки зрения, так и с программной, и поместить в виртуальную среду. Важно отметить, что использование такого способа симуляции полезно не

только для тестирования робототехнических систем, но и для выполнения предварительного обучения алгоритмов обучения с подкреплением в моделируемой среде. Такое обучение возможно выполнить с использованием программного обеспечения gym-gazebo [76]. Подобный подход был рассмотрен в статье [18].

2.2 Сравнение реализаций метода DQN в среде симулятора Gazebo

В данном подразделе диссертационного исследования описывается экспериментальное исследование, которое заключается в тестирования среды симуляции Gazebo и пригодности алгоритмов Deep Q-Networks для решения задач, связанных с передвижением в трехмерном пространстве.

2.2.1 Методология экспериментального исследования

Одной из классических задач обучения с подкреплением является «шест на тележке» (англ. cartpole). Впервые задача описана в [77]. Шест прикреплен к тележке при помощи шарнира без трения, сама тележка движется по дорожке, также без трения.

Задача состоит в том, чтобы, управляя движением тележки удерживать шест вертикально как можно дольше. Таким образом, пространство действий в этой задаче является дискретным и возможные действия могут принимать два значения: 0 – выполняем движение тележки влево и 1 – выполняем движение тележки вправо. В качестве наблюдений среды передаются положение тележки на дорожке, её скорость, угол шеста и его угловая скорость. Так как задача состоит в том, чтобы как можно дольше удерживать шест вертикально, то награда равна 1 за каждый выполненный шаг, до тех пор, пока эпизод не закончен. Эпизод считается законченным, когда шест отклоняется от вертикального положения больше чем на 12 градусов, когда тележка достигает края экрана и когда значение награды больше 200. На рисунке 3 приведена реализация этой среды из пакета gymnasium [78].

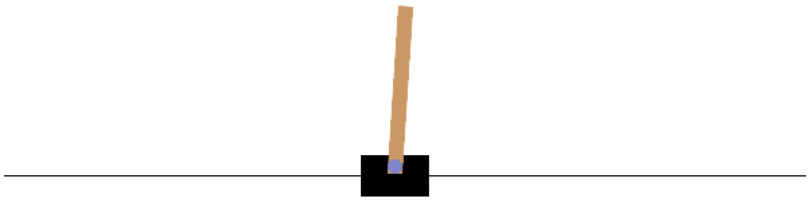

Рисунок 3 – Реализация среды cartpole из пакета gymnasium

Учитывая дискретное пространство действий в этой среде, то для решения задачи хорошо подойдёт алгоритм обучения с подкреплением Q-Learning, а также его нейросетевая версия Deep Q-Networks. Для исследования были выбраны три реализации [79–81] алгоритма DQN для задачи cartpole из открытого списка лидеров. Обозначим эти алгоритмы как Алгоритм 1, Алгоритм 2 и Алгоритм 3 соответственно.

В таблице лидеров указано, что для решения задачи этим алгоритмам требуется 933, 85 и 306 эпизодов соответственно. Задача считается решенной, когда будет достигнут средний уровень вознаграждения 100 за 100 последовательных попыток.

Все три алгоритма реализуют DQN с буффером воспроизведения опыта. Отличия сводятся к архитектуре используемого многослойного перцептрона, функциям активаций и значением гиперпараметров.

Для проверки этих алгоритмов в более реалистичных условиях с лучшей симуляцией физики и непрерывным пространством действий было решено использовать среды CartPole и SimplestBipedal, реализованные в среде Gazebo. Среда CartPole, изображенная на рисунке 4, повторяет оригинальную задачу за исключением способа симуляции физики процесса.

Среда SimplestBipedal, изображенная на рисунке 5, реализована специально в рамках этого исследования и представляет собой двуногого робота кубической формы, ноги которого, прикреплены при помощи моторизированных шарниров, и выполнены в виду двух шестов длиной в два раза больше основной части робота. В этой среде награда назначается пропорционально положению по высоте основной части робота, таким образом задача сводится к удержанию роботом равновесия и сохранения вертикального положения. Важно заметить, что пространство действий здесь является непрерывным, так как управление ногами робота симулируется как управление сервоприводами.

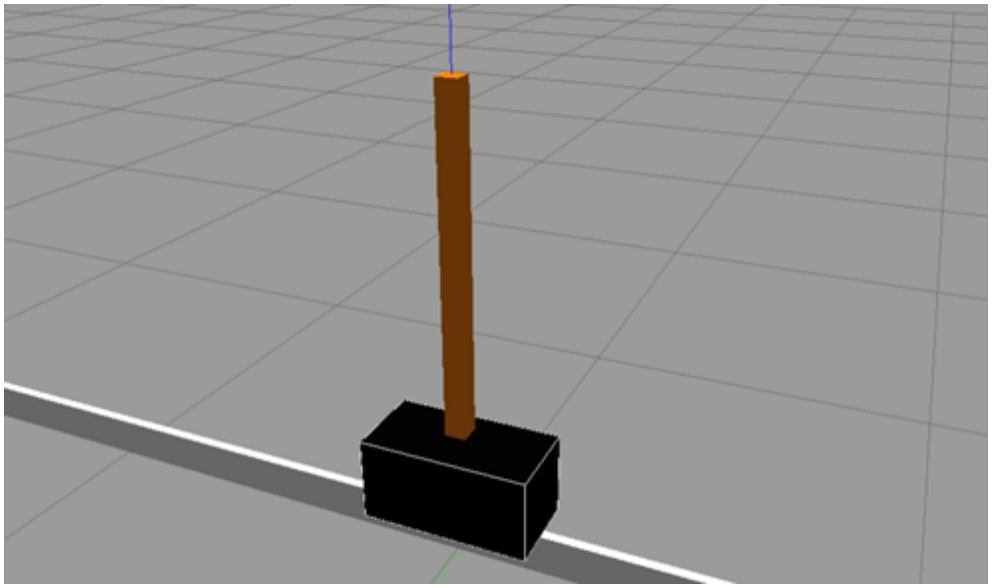

Рисунок 4 – Реализация среды CartPole в симуляторе Gazebo

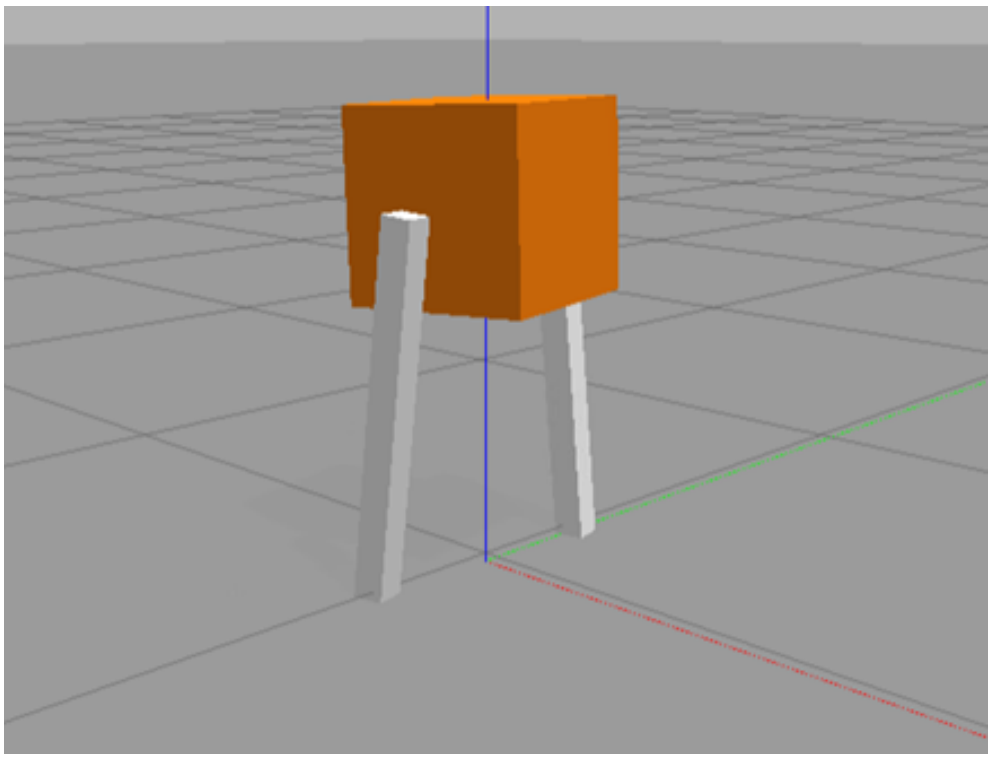

Рисунок 5 – Реализация среды SimplestBipedal в Gazebo

Для создания управляющего воздействия в непрерывном пространстве действий при помощи алгоритма DQN, пространство действий разделяется на несколько отрезков и каждый отрезок назначается одному из значений из дискретного пространства действий, с которым работает алгоритм DQN.

2.2.2 Результаты экспериментального исследования

Таким образом, все три алгоритма были протестированы в средах Cartpole и SimplestBipedal, реализованных в симуляторе Gazebo.

Результаты обучения Алгоритма 1 в среде Cartpole представлены на рисунке 6. Результаты в этой среде для Алгоритма 2 и Алгоритма 3 представлены на рисунках 7 и 8 соответственно.

Сравнение результатов обучения алгоритмов в среде SimplestBipedal приведены на рисунке 9.

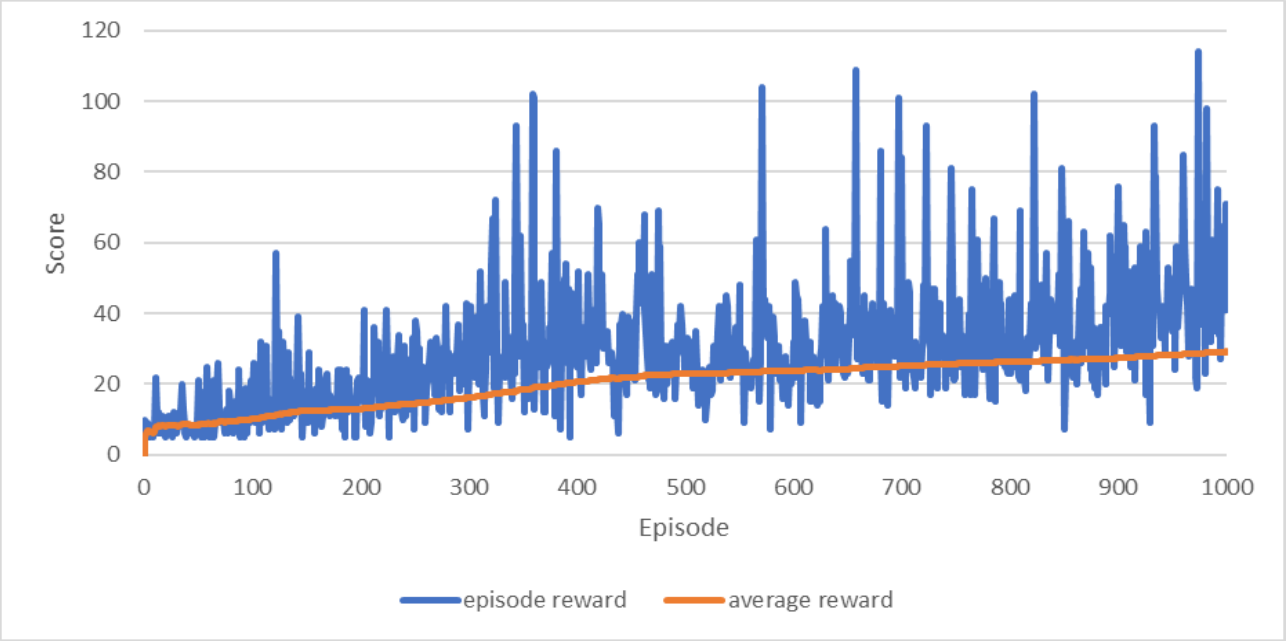

Рисунок 6 – Результаты обучения Алгоритма 1 в среде Cartpole реализованной в симуляторе Gazebo. Вертикальная ось графика показывает значение награды достигнутое в эпизоде, обозначенном по горизонтальной оси

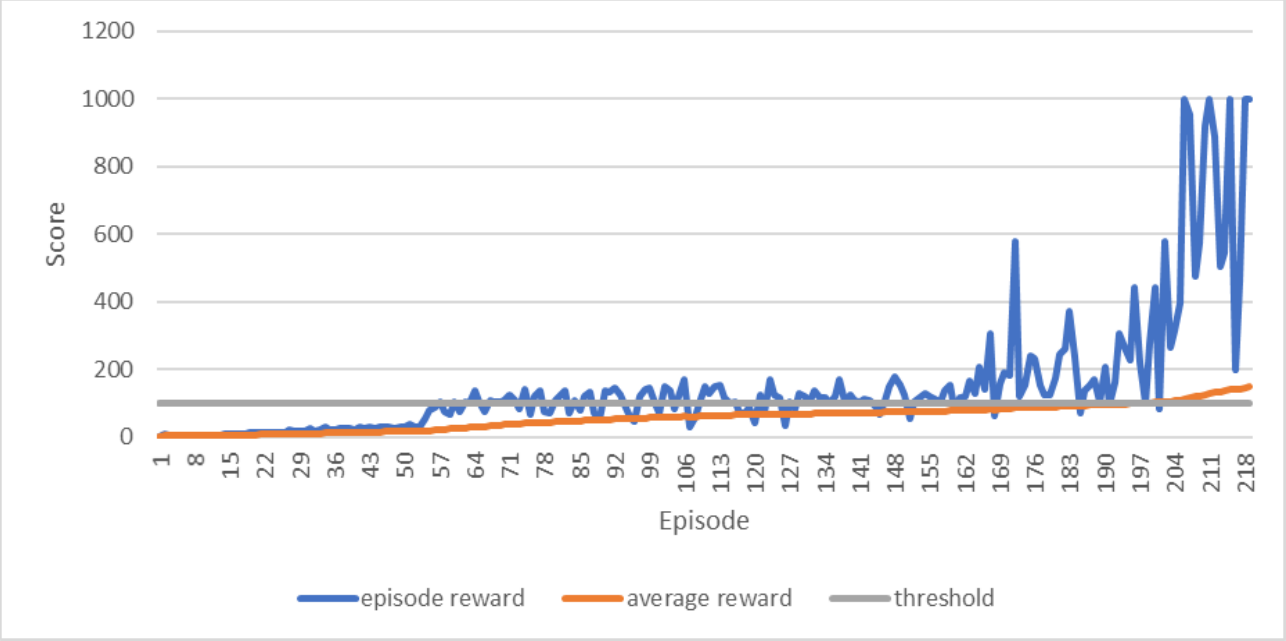

Рисунок 7 – Результаты обучения Алгоритма 1 в среде Cartpole реализованной в симуляторе Gazebo. Вертикальная ось графика показывает значение награды достигнутое в эпизоде, обозначенном по горизонтальной оси

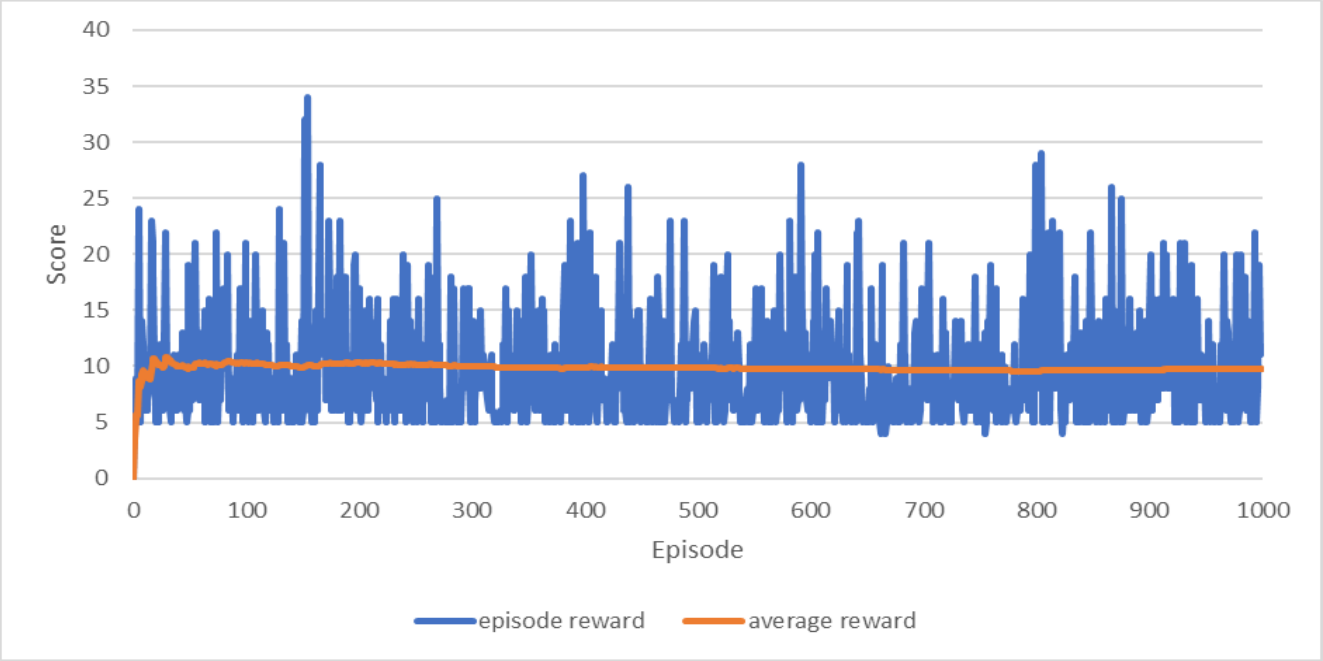

Рисунок 8 – Результаты обучения Алгоритма 1 в среде Cartpole реализованной в симуляторе Gazebo. Вертикальная ось графика показывает значение награды достигнутое в эпизоде, обозначенном по горизонтальной оси

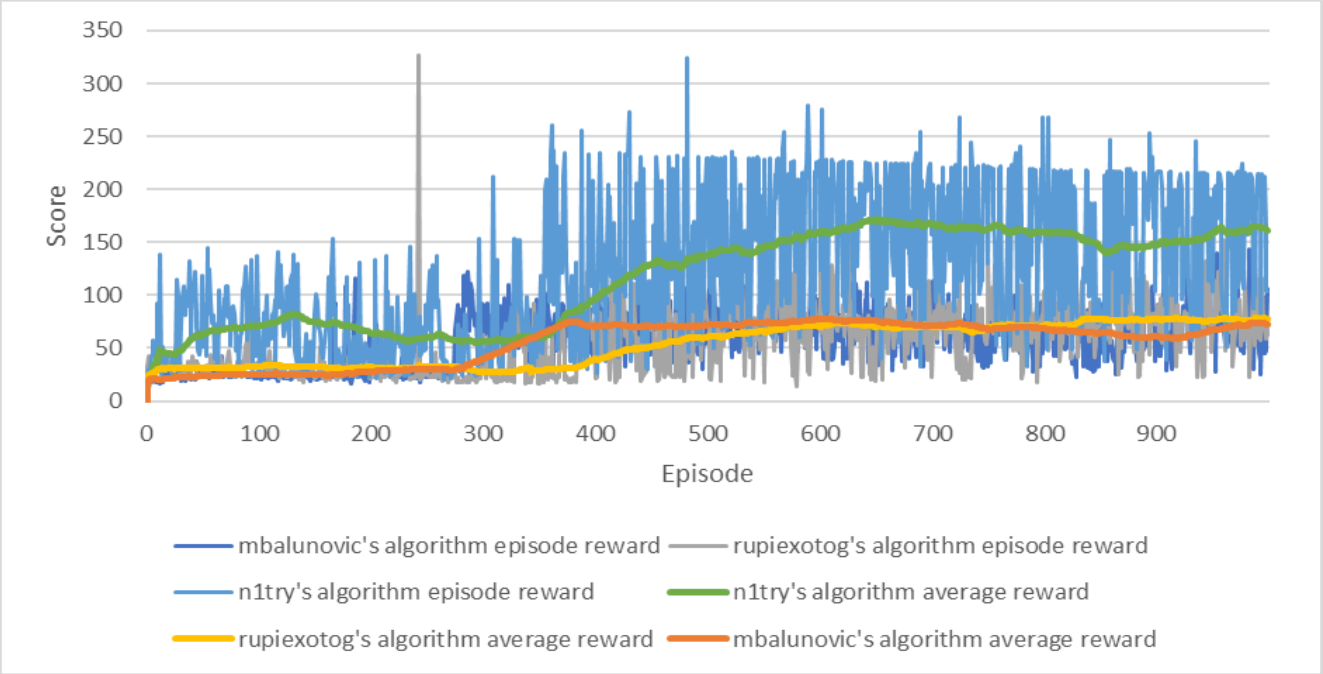

Рисунок 9 – Результаты обучения всех трёх алгоритмов в среде SimplestBipedal реализованной в симуляторе Gazebo. Вертикальная ось графика показывает значение награды достигнутое в эпизоде, обозначенном по горизонтальной оси

Можно заметить, что только Алгоритм 1 смог решить задачу, поставленную в среде Cartpole. Также видно, что наилучший результат в среде SimplestBipedal достигнут при помощи Алгоритма 2. При этом в общем значения наград для всех алгоритмов не принимают высоких значений, а алгоритмы склонны попадать в локальные оптимумы из-за большого количества степеней свободы в решаемой задаче.

2.2.3 Выводы по результатам экспериментального исследования

В данном подразделе приведены экспериментальные исследования, направленные на то, чтобы выяснить потенциальное качество решения задачи приобретения навыков передвижения в трехмерном пространстве посредством алгоритмов Deep Q Networks, а также исследовать возможность применения симулятора Gazebo с целью моделирования различных реальных сред.

Исходя из проведенного исследования можно сделать следующие выводы:

- Алгоритмы семейства DQN плохо подходят для сложных задач с непрерывными пространствами действий и большими размерностями.
- Использование симулятора Gazebo позволяет избежать прототипирования реального робота и длительного процесса обучения его в реальном мире, однако этот симулятор не реализует изменения масштаба времени симуляции и параллельного запуска нескольких эпизодов для ускорения процесса обучения. В результате эксперименты занимают куда больше времени, чем, например, в средах Unity, реализованных при помощи пакета ML-Agents.

2.3 Сравнение эффективности современных алгоритмов обучения с подкреплением в задаче управления движением агентов в трехмерном пространстве

В данном подразделе диссертационного исследования проводится анализ и сравнение эффективности трех современных алгоритмов обучения с

подкреплением: Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO) и Randomized Ensembled Double Q-Learning (REDQ). Основное внимание уделяется их применению для управления движением агентов в симулированных трехмерных пространствах. Реализация экспериментов осуществлена в средах пакета gymnasium, что обеспечивает создание многообразных условий для тестирования алгоритмов и оценки их производительности в задачах навигации и манипуляции.

Основной целью данного исследования является оценка способности исследованных алгоритмов обучения с подкреплением адаптироваться к сложным динамическим задачам в трехмерном пространстве и сравнение их производительности с точки зрения скорости обучения, устойчивости достигнутых результатов и качества полученного поведения.

2.3.1 Методология экспериментального исследования

Для проведения экспериментального исследования были выбраны пять трехмерных сред, каждая из которых моделирует задачи, требующие от агентов развития навыков перемещения в трехмерном пространстве. Все среды взяты из пакета gymnasium.

Для проведения экспериментальных исследований были выбраны следующие симулированные среды из пакета gymnasium: Ant, Humanoid, Half-Cheetah, Hopper и Walker. Эти среды изображены на рисунке 10. Причина такого выбора заключается в наличии разнообразия и способности к симуляции реальных физических взаимодействий и движений. Оценка производительности агентов проводилась по критериям скорости обучения, которая проявляется в виде выборочной эффективности. Выборочная эффективность (sample efficiency) [82] в контексте обучения с подкреплением — это показатель, отражающий способность алгоритма достигать высокого уровня производительности с использованием ограниченного числа обучающих примеров или взаимодействий с окружающей средой.

Симулированная среда Ant моделирует трёхмерного робота-муравья с телом и четырьмя ногами, каждая из которых состоит из двух сегментов. Задача включает в себя координацию движений всех четырёх ног для перемещения вперёд, применяя моменты к восьми сочленениям, которые соединяют сегменты ног с туловищем. Среда изображена на рисунке 10 слева.

Half-Cheetah [83] представляет собой двумерного робота из девяти сегментов и восьми сочленений. Задача заключается в приложении моментов к сочленениям для максимизации скорости движения вперёд, при этом за движение вперёд начисляются положительные награды, а за движение назад — отрицательные. Среда изображена на рисунке 10 посередине.

Среда Humanoid [84] представляет трёхмерного двуногого робота, моделирующего человека. Он состоит из туловища с двумя ногами и руками, где каждая нога разделена на три сегмента, а руки — на два. Основная задача агента заключается в движении вперёд как можно быстрее без падения. Среда изображена на рисунке 10 справа.

Hopper [85] — это одноногий двумерный робот, состоящий из четырёх основных частей: торса, бедра, голени и ступни. Задача агента — выполнение прыжков вперёд, применяя силу к трем сочленениям. Среда изображена на рисунке 11 слева.

Walker расширяет среду Hopper за счёт добавления второго набора ног, позволяя роботу передвигаться вперёд. Задача заключается в координации движений всех семи частей тела, чтобы продвигаться вперёд, применяя моменты к шести сочленениям. Среда изображена на рисунке 11 справа.

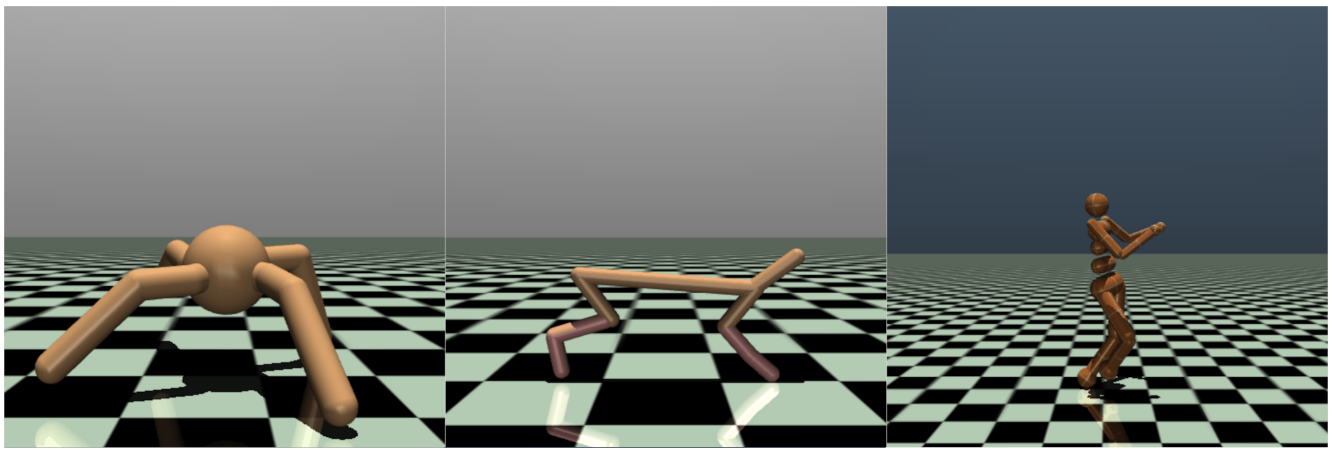

Рисунок 10 – Среды из пакета gymnasium. Ant, Half-Cheetah, Humanoid

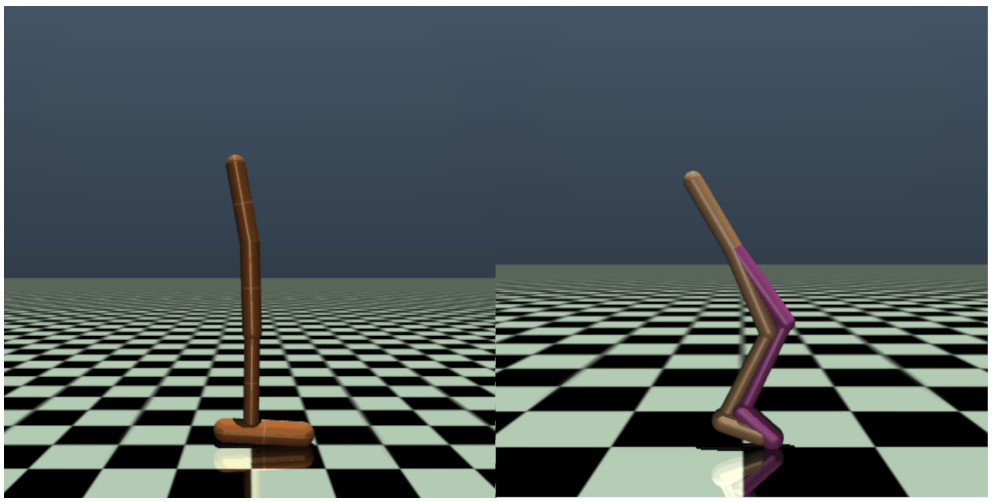

Рисунок 11 – Среды из пакета gymnasium. Hopper, Walker

Во всех средах задача алгоритма обучения с подкреплением достигнуть того, чтобы агент уверенно перемещался в пространстве. Награды во всех средах назначаются пропорционально скорости. Таким образом, цель алгоритма не только овладеть способностью передвижения, но и делать это максимально эффективно.

Обучение для всех методов выполнялось в течение 10^6 эпизодов.

2.3.2 Результаты экспериментального исследования

В рамках проведенных экспериментов для среды Ant были получены кривые обучения, результаты которых представлены на рисунке 12. В этом эксперименте наилучшего среднего результата достигает алгоритм SAC, но при этом дисперсия

награды выше, чем у остальных. Алгоритм PPO показывает наименьшую эффективность.

На рисунке 13 представлен график для эксперимента со средой HafCheetah. В этой среде лидирует алгоритм REDQ. Несмотря на это результат SAC отстаёт незначительно и анализируя форму кривых можно предположить о том, что на некотором шаге обучения эффективность алгоритмов сравняется. PPO значительно отстаёт от других алгоритмов.

На рисунке 14 приведен график обучения сравниваемых алгоритмов в среде Hopper. Здесь к концу обучения наивысшего результата достигает SAC, при этом SAC и REDQ в середине процесса обучения показывают внезапное ухудшение результата. Это может свидетельствовать о попадании алгоритма в некоторый локальный минимум. PPO ведёт себя стабильно, но при этом показывает наихудшую эффективность.

На рисунке 15 приведены результаты для среды Humanoid. В этой среде SAC значительно опережает другие методы.

На рисунке 16 представлен график обучения алгоритмов в среде Walker2d. В этом случае алгоритмы REDQ и SAC показывают схожую эффективность на последних шагах эксперимента, однако в середине эксперимента у SAC сложности с попаданием в локальный минимум. Также для SAC видна большая дисперсия награды. PPO снова оказывается худшим, а лучший в этом эксперименте себя показывает REDQ.

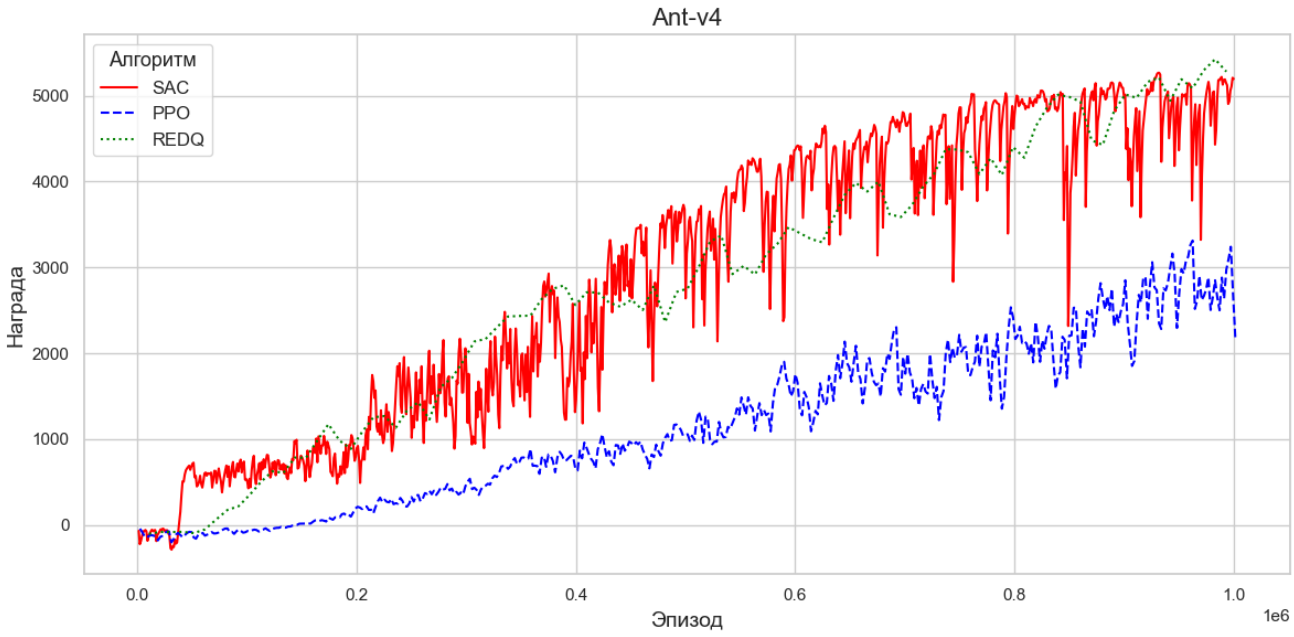

Рисунок 12 – Результат сравнения выборочной эффективности алгоритмов в среде Ant-v4

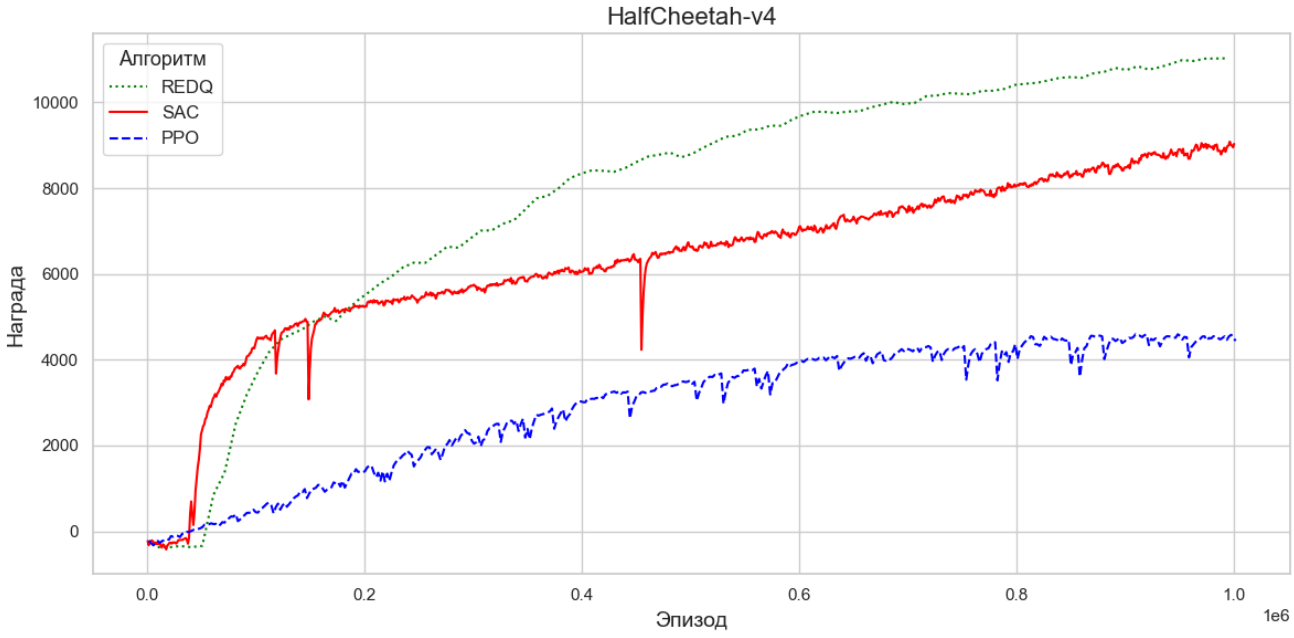

Рисунок 13 – Результат сравнения выборочной эффективности алгоритмов в среде HalfCheetah-v4

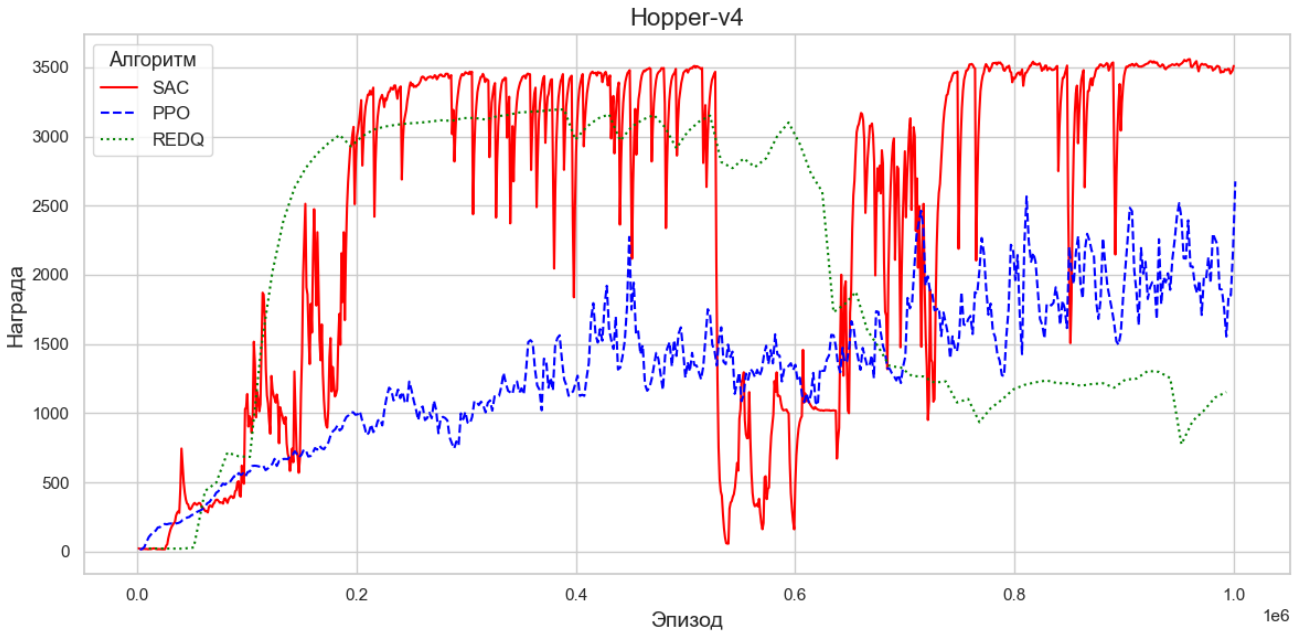

Рисунок 14 – Результат сравнения выборочной эффективности алгоритмов в среде Hopper-v4

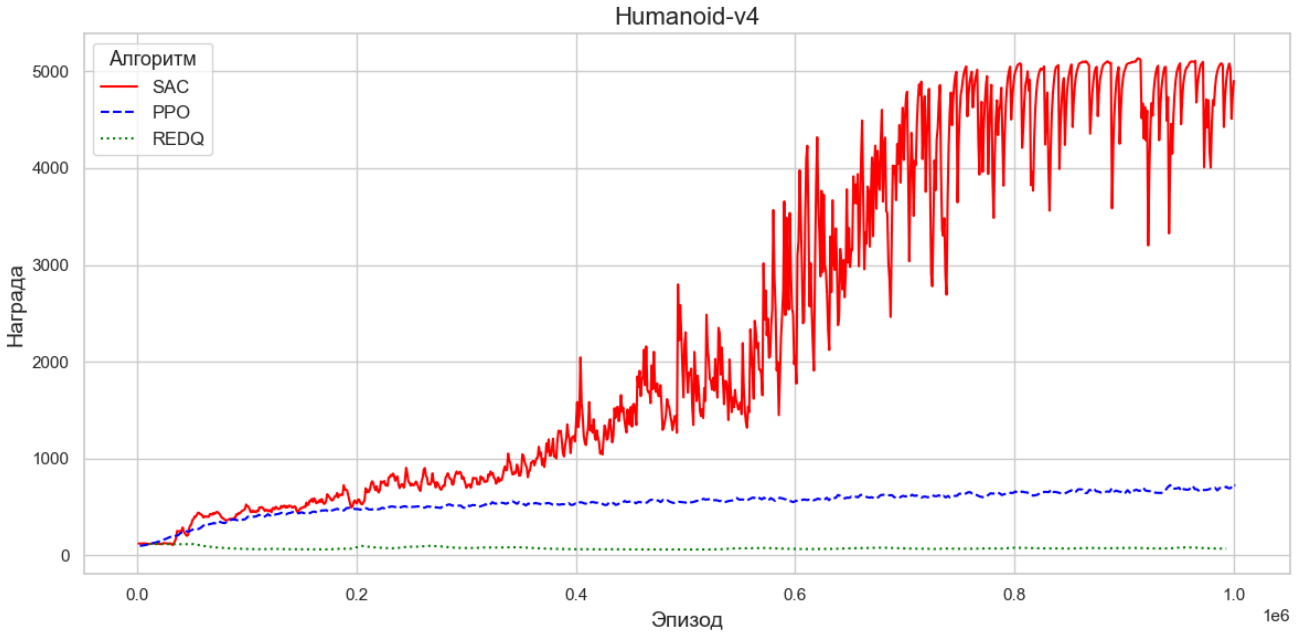

Рисунок 15 – Результат сравнения выборочной эффективности алгоритмов в среде Humanoid-v4

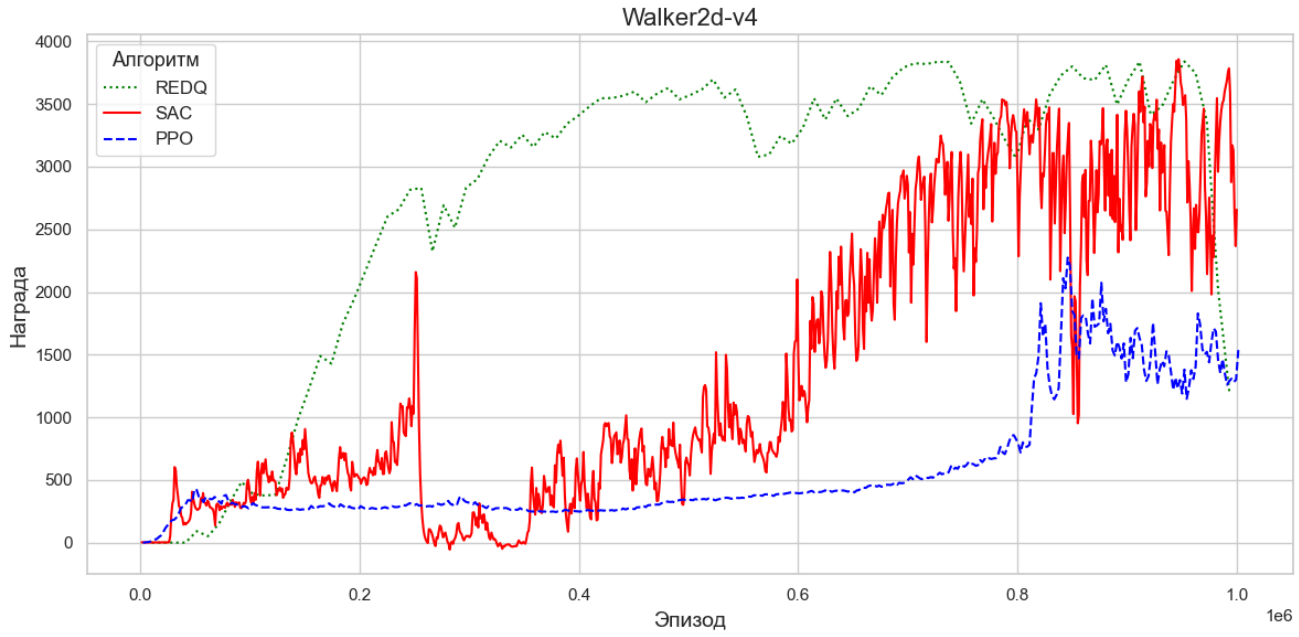

Рисунок 16 – Результат сравнения выборочной эффективности алгоритмов в среде Walker2d-v4

2.3.3 Выводы по результатам экспериментальных исследований

В данном подразделе приведены итоги проведенных экспериментальных исследований эффективности трех алгоритмов обучения с подкреплением: Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO) и Randomized Ensembled Double Q-Learning (REDQ) в различных трехмерных средах.

SAC продемонстрировал наилучшую производительность во всех исследуемых средах, особенно в задачах, требующих сложной координации и точности движений. Благодаря механизму энтропийной регуляризации SAC эффективно уравновешивал исследование и эксплуатацию, достигая быстрой сходимости и обеспечивая стабильность процесса обучения.

PPO и REDQ показали хорошие результаты в стабильных и предсказуемых условиях, однако столкнулись с трудностями в динамичных средах, требующих быстрой адаптации к изменениям. Несмотря на это, благодаря способности поддерживать стабильность при значительных изменениях в стратегии обучения, PPO оставался эффективным в долгосрочной перспективе.

По результатам проведенных экспериментальных исследований было выявлено, что выбор алгоритма обучения с подкреплением должен основываться на специфике задачи и условиях окружающей среды. SAC оказался наиболее универсальным и показал лучшие результаты в большинстве сценариев, особенно когда требовалась быстрая адаптация к новым условиям, также SAC обладает наименьшей вычислительной сложностью. Второе предпочтение отдаётся алгоритму REDQ.

2.4 Влияние состава набора окружающих наблюдений на процесс приобретения агентом навыков движения в трехмерном пространстве

В рамках диссертационного исследования было изучено воздействие качества и объёма наблюдений об окружающей среде на эффективность процесса обучения агентов с подкреплением в трехмерном пространстве. Выявлено, что наличие избыточных или нерелевантных данных может не только замедлять обучение, но и влиять на способность агента успешно решать поставленные задачи. Экспериментальные исследования проводились с использованием игрового движка Unity и пакета ML-Agents, что позволило детально анализировать динамику обучения в контролируемой виртуальной среде.

Алгоритм Soft Actor-Critic был выбран для изучения за его наилучшую, как было продемонстрировано в разделе 2.2, способность эффективно обучать агентов выполнению сложных двигательных навыков в трехмерных средах. SAC особенно ценен в условиях, где требуется баланс между исследованием среды и оптимизацией действий, что делает его идеальным выбором для демонстрации влияния качества наблюдений на процесс обучения.

Цель данного раздела — продемонстрировать, как полнота и точность информации о среде, предоставляемая агенту, определяют его способность эффективно осваивать двигательные навыки в трехмерном пространстве, используя алгоритм Soft Actor-Critic для достижения наилучших результатов. Результаты исследования также приводятся в работе [15-20].

2.4.1 Методология экспериментального исследования

Для создания моделирующей среды были использованы игровой движок Unity и пакет ML-Agents, что позволило проводить детальные симуляции физических процессов с возможностью их гибкой настройки. Преимуществом использования симулятора является отсутствие необходимости в физическом перезапуске экспериментов. В отличие от реальных роботов, которым требуется время для возвращения в исходное положение, в симуляции перемещение происходит мгновенно. Кроме того, симулятор позволяет ускорить течение времени, что способствует выполнению большего количества обучающих итераций в тот же период реального времени. Ещё одно значительное преимущество — возможность одновременного запуска множества копий агентов, что ускоряет процесс обучения. В данном исследовании одновременно обучалось шестнадцать клонов агента.

Агент, названный SimplestBipedal, оснащён парой ног, каждая из которых имеет два сустава. Верхний сустав обеспечивает подвижность в двух плоскостях, в то время как нижний сустав — только в одной. Этот робот размещается на платформе, размер которой в сто раз превышает размер самого агента. На начало каждого эпизода агент устанавливается в центр платформы, а цель —помещается в случайной точке платформы. При достижении цели агент получает вознаграждение, после чего цель исчезает и появляется в новом случайном месте, требуя от агента корректировки направления движения для продолжения задания.

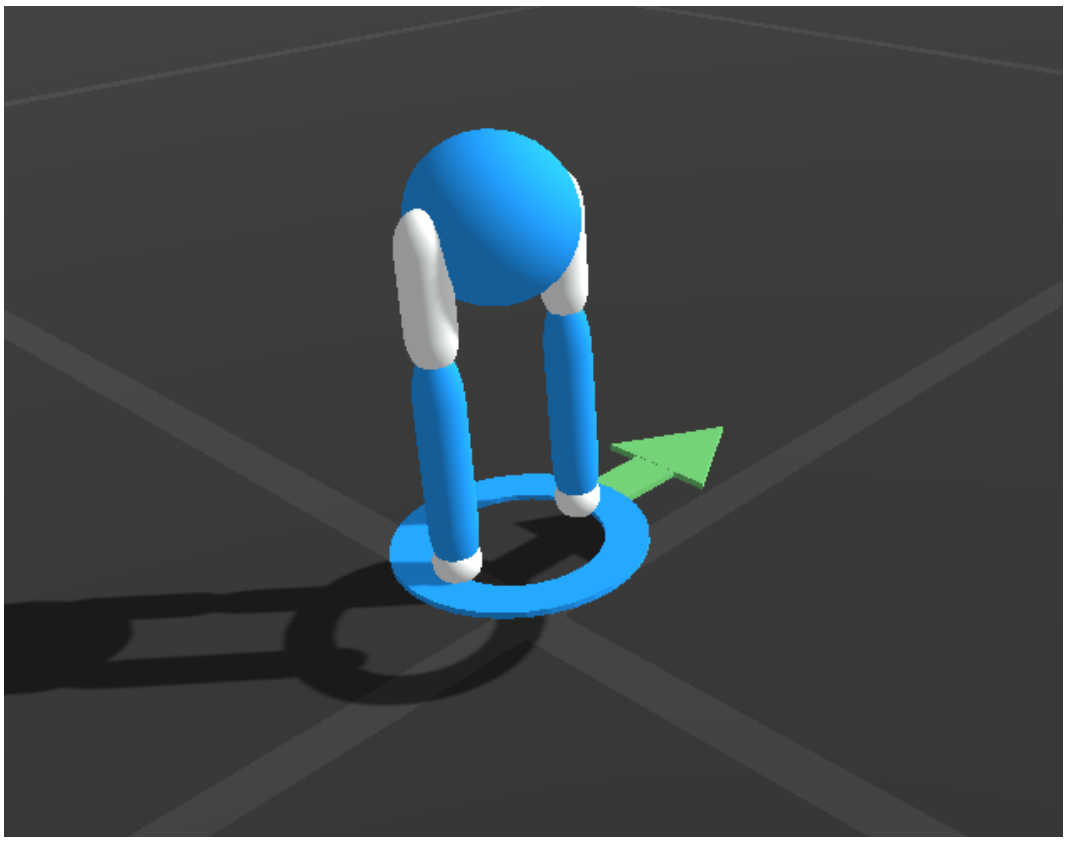

Рисунок 17 – Внешний вид агента SimplestBipedal

Разработанный специально для экспериментальных целей, этот агент представляет собой простую физическую модель, решающую сложную задачу с множеством степеней свободы, но при этом характеризующуюся низким уровнем сложности. В ходе всех тестирований агент получает информацию о своём окружении, включая данные о контакте конечностей с поверхностью, расстоянии до цели, направлении к цели и высоте тела над поверхностью. Дополнительные данные варьируются в зависимости от конкретного эксперимента.

Всего было проведено девять экспериментов, в которых наблюдения были организованы различными способами для проверки гипотезы о том, что более полная информация, предоставляемая агенту, способствует более быстрому решению задачи. Выбор данных для наблюдений в каждом эксперименте был осуществлён в соответствии с параметрами, как указано в таблице 2.

Таблица 2 – Конфигурация проведенных экспериментов

Информация передаваемая агенту	Номер эксперимента								
	1	2	3	4	5	6	7	8	9
Перемещение конечностей (глобальные координаты)	+	+	+	+	+	+		+	+
Перемещение конечностей (локальные координаты)			+	+	+	+	+		
Поворот конечностей (глобальные координаты)	+	+	+	+	+		+	+	+
Поворот конечностей (локальные координаты)	+		+	+	+		+	+	
Скорость конечностей (глобальные координаты)	+			+	+	+	+		
Скорость конечностей (локальные координаты)				+	+			+	
Сила прилагаемая к суставам					+				+

2.4.2 Результаты экспериментального исследования

На рисунке 18 представлены результаты обучения, выполненные согласно методологии, описанной в предыдущем разделе. Для улучшения визуального восприятия данных на графиках использовалось экспоненциальное взвешивание с коэффициентом сглаживания 0,1.

Особенностью данной задачи является взаимосвязь между продолжительностью эпизода и размером получаемого вознаграждения: агент получает вознаграждение пропорционально времени, в течение которого он успешно выполняет задание. Если агент терпит неудачу в начале эпизода, и его продолжительность оказывается краткой, вознаграждение не начисляется.

Результаты экспериментальных исследований показали, что увеличение объема информации, предоставляемой агенту, не всегда способствует улучшению его производительности. Кроме того, было выявлено, что предоставление данных о силе, приложенной к суставам, может снижать эффективность тренировок. Важно отметить, что шестой эксперимент показал лучшие результаты, несмотря на то что агенту предоставлялось минимальное количество информации.

Такие результаты подчеркивают, что некоторые данные, несмотря на их прямую связь с задачей, могут увеличивать время на её решение и влиять на качество исхода. Это подчеркивает необходимость тщательного выбора данных, которые агент использует в процессе обучения, и важность оптимизации набора наблюдений для максимизации вознаграждения.

Кроме того, необходимо учитывать, что в других сценариях, например таких как поиск выхода из лабиринта, выводы, сделанные на основе пятого эксперимента, могут не подтвердиться. Это подчеркивает, что более обширная информация об окружающей среде не гарантирует более успешного решения задачи и подчеркивает важность адаптации подхода к конкретным условиям задачи.

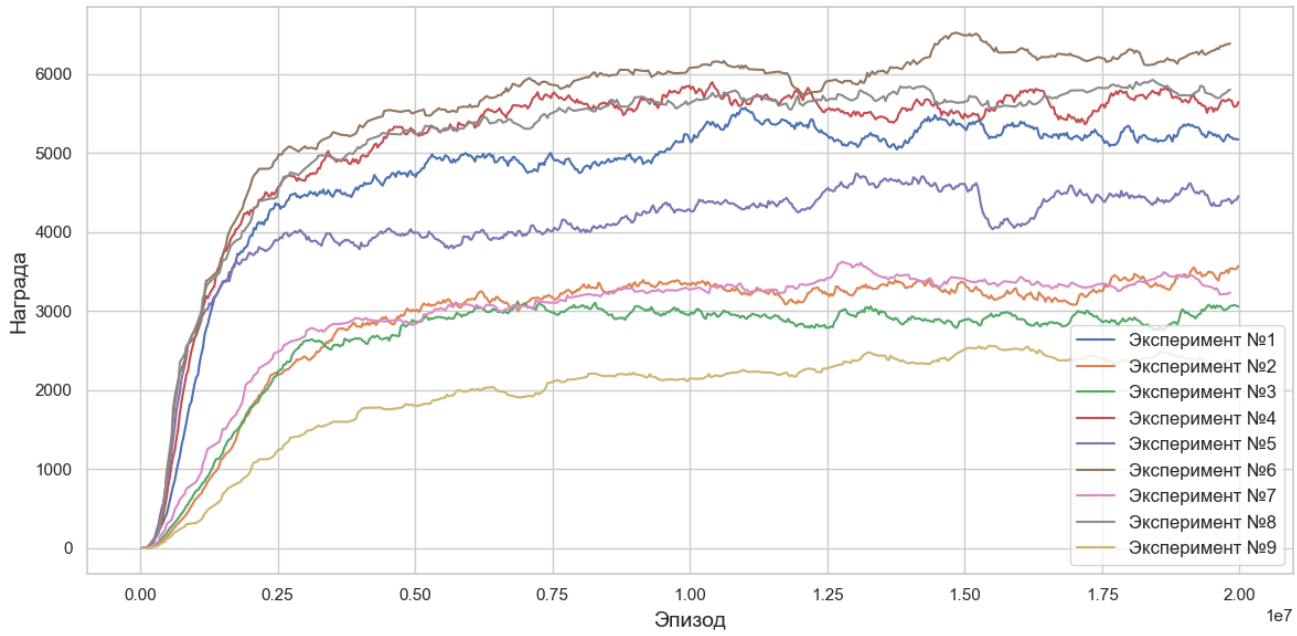

Рисунок 18 – Значение суммарной награды агента в зависимости от шага обучения для каждого эксперимента

2.4.3 Выводы по результатам экспериментальных исследований

В ходе проведенного исследования было выявлено, что значение каждого компонента в наборе передаваемых агенту наблюдений не всегда является ясным на начальных этапах обучения. Это подчеркивает важность предварительного изучения влияния конкретных данных на исходы обучения до его начала.

Хотя основная задача остаётся неизменной, результаты обучения с подкреплением могут значительно различаться в зависимости от выбранных подходов и особенно от разнообразия наборов передаваемых наблюдений. Изначально наборы данных могут казаться случайными без очевидных закономерностей в их структуре.

На основе результатов исследования предлагается методика оценки влияния состава набора наблюдений окружающей среды на качество решений, принимаемых агентом.

Рассмотрим множество N доступных признаков состояния среды, где каждый признак представляет собой компонент наблюдения, доступного агенту при принятии решений. Пусть проведено J экспериментов, и каждый эксперимент $e_j (j \in J)$ характеризуется итоговой (суммарной) наградой R_j, полученной агентом, и набором использованных признаков $N_j \subseteq N$. Участие признака $n \in N$ в эксперименте e_j будем обозначать как $n \in e_j$, то есть признак n был использован в эксперименте e_j.

Для каждого признака n определим множество экспериментов, в которых этот признак участвовал, как:

$$E_n = \{e_j | j \in J \text{ и } n \in N_j\}.$$

Для оценки полезности признака n, вычислим его вес w_n, который отражает вклад признака в итоговую награду. Вес рассчитывается по следующей формуле:

$$w_n = \frac{1}{|E_n|} \sum_{e_j \in E_n} \frac{R_{e_j}}{|N_{e_j}|}$$

где R_{e_j} — суммарная награда, полученная в эксперименте e_j, а $|N_{e_j}|$ — количество признаков, использованных в эксперименте e_j.

Таким образом, полезность каждого признака оценивается как среднее значение вкладов признака в итоговую награду во всех экспериментах, в которых он использовался, нормированное на количество признаков в каждом эксперименте. Высокое значение w_n указывает на значительный вклад признака n в успешность обучения агента.

В результате, предлагается возможность улучшения алгоритмов обучения с подкреплением, особенно в части выбора и оптимизации конкретных наблюдений для передачи в систему обучения. Эффективным шагом может быть проведение статистического анализа, в соответствии с предложенной методикой, чтобы оценить вклад различных характеристик в обучающий процесс и сделать выводы на основе полученных данных. Реализация серии обучающих эпизодов с разными наборами наблюдений позволит не только оптимизировать их состав, но и глубже понять связь между успешным решением задачи и вкладом каждого параметра.

2.5 Выводы и результаты второго раздела

В данном разделе были рассмотрены существующие методы обучения с подкреплением для решения задачи приобретения навыков передвижения роботами в трехмерном пространстве. Были рассмотрены как уже ставшие классическими методы Deep Q-Networks (DQN), так и современные методы Soft-Actor Critic (SAC), Proximal Policy Optimization (PPO) и Randomized Ensembled Double Q-Learning (REDQ).

Были рассмотрены различные методы моделирования реальных сред. Было установлено, что среды, реализованные при помощи Gazebo, могут полностью симулировать реальных роботов, как с физической точки зрения, так и программной. Этот факт делает ROS и Gazebo незаменимыми инструментами при прототипировании роботов.

Также результаты экспериментальных исследований говорят о непригодности методов DQN для решения поставленных задач. Вместе с этим результаты исследований говорят о том, что выполнение предобучения алгоритмов обучения с подкреплением в среде Gazebo может быть слишком длительным. По этой причине предпочтительным способом выполнения предварительного обучения роботов в виртуальных средах можно назвать способ, при котором первым симулятором будет выступать Unity, способный симулировать физику с ускорением и в многопоточном режиме, что позволяет многократно ускорить процесс получения параметров метода. Затем продолжается обучение метода с использованием более точного прототипа робота в среде Gazebo и только на последнем шаге необходимо выполнять обучение робота в реальных условиях. Однако в рамках данного исследования было решено сфокусироваться на средах из пакетов gymnasium и dm_control, которые являются общепризнанными бенчмарками для методов обучения с подкреплением.

Также в данном разделе был проведен анализ эффективности трех алгоритмов обучения с подкреплением: Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO) и Randomized Ensembled Double Q-Learning (REDQ). Исследование было направлено на изучение их эффективности в управлении движением агентов в трехмерных симулированных пространствах.

Результаты экспериментальных исследований показали, что SAC обладает наибольшей стабильностью среди протестированных алгоритмов, особенно в задачах, требующих быстрой адаптации к изменяющимся условиям. SAC демонстрировал устойчивость и высокую скорость сходимости. В то время как PPO и REDQ показали положительные результаты, однако они оказались менее устойчивыми в динамичных условиях.

Было проведено исследование того, как изменение набора наблюдений влияет на процесс обучения. Была продемонстрирована необходимость предварительного исследования среды на предмет полезности тех или иных наблюдений. Результаты экспериментальных исследований указывают на то, что правильный выбор и настройка параметров обучения существенно влияют на результаты. Предложена

методика оценки влияния состава набора наблюдений окружающей среды на качество решений, принимаемых агентом.

По теме раздела опубликованы работы [*10–12, *14–16].

РАЗДЕЛ 3. Модель интеграции алгоритмов обучения с подкреплением с кодировщиком трансформера

В четвёртом разделе диссертационного исследования рассматривается использование архитектуры кодировщика трансформера для кодирования состояний в алгоритмах обучения с подкреплением. Представлена модель интеграции алгоритмов обучения с подкреплением с кодировщиком трансформера, объединяющий трансформеры с уже известными алгоритмами, такими как Soft Actor-Critic, с целью улучшения производительности и обобщающей способности. Результаты проведенных экспериментальных исследований демонстрируют, что такой подход может улучшить процесс обучения в задачах приобретения сложных двигательных навыков в трехмерном пространстве.

Несмотря на успехи, достигнутые алгоритмом SAC, был выявлен потенциал для его дальнейшего усовершенствования, особенно в контексте обработки последовательностей и пространственных данных. Архитектура трансформера, разработанная изначально для задач обработки естественного языка, демонстрирует отличные способности к анализу последовательностей и контекстного понимания, что делает её перспективной для использования в глубоком обучении с подкреплением.

В данном исследовании на основе предложенной модели интеграции алгоритмов обучения с подкреплением с кодировщиком трансформера, разработан алгоритм, который подразумевает интеграцию трансформера в структуру SAC для улучшенной обработки и интерпретации последовательностей состояний. Предполагается, что такой подход улучшит не только производительность SAC в сложных условиях, но и обеспечит более глубокое понимание динамики сред и поведения агентов. Цель текущего исследования — продемонстрировать, как трансформеры могут быть эффективно использованы для улучшения эффективности обучения с подкреплением.

3.1 Архитектура трансформер

Трансформер — это архитектура нейронной сети, разработанная для обработки последовательных данных, таких как текст, с использованием механизма внимания. Трансформеры были впервые представлены в статье [86]. Основная идея трансформера заключается в замене рекуррентных слоев и свёрточных слоев на механизм самовнимания (self-attention), что позволяет эффективно моделировать долгосрочные зависимости в последовательностях данных.

Трансформер разделяется на две части: кодировщик и декодировщик. Кодировщик состоит из нескольких идентичных слоёв, каждый из которых содержит механизм мультиголового внимания и полносвязный нейронный слой. Каждый из этих подуровней оснащён остаточной связью и нормализацией.

Декодировщик также включает несколько слоёв с механизмами внимания, которые, кроме стандартного внимания, используют маскированное внимание для предотвращения утечки информации из будущих элементов. Это ключевое отличие, позволяющее декодировщику генерировать вывод по одному элементу за раз, что делает его идеальным для генеративных задач, таких как перевод текста. Комбинация этих структур позволяет трансформеру эффективно обрабатывать и генерировать последовательности данных, обеспечивая при этом высокую параллелизацию обработки и глубокое понимание контекста входных данных.

В обучении с подкреплением часто используются марковские процессы принятия решений как математическая модель, предполагающая, что будущее состояние зависит только от текущего состояния и совершённого действия. Однако на практике многие процессы являются немарковскими, поскольку события и состояния могут зависеть не только от текущего момента, но и от всей предшествующей последовательности событий.

Важно отметить, что хотя марковские процессы представляют удобную математическую абстракцию, реальные задачи часто включают более сложные временные и контекстные зависимости, что затрудняет прямое использование классических марковских моделей. При этом отмечается, что обучение с

подкреплением можно использовать в шумной и немарковской среде, однако такое допущение может значительно повлиять на эффективность методов [87–89]. Такое наблюдение подчеркивает значимость поиска альтернативных подходов в моделировании поведения и принятии решений.

Применение архитектуры трансформера в обучении с подкреплением является мощным подходом [12, 90, 91] к адаптации к реальным и сложным средам, которые не всегда подчиняются марковским свойствам. Разработанные первоначально для обработки последовательных данных, трансформеры обладают способностью моделировать сложные зависимости на различных временных и пространственных шкалах благодаря своей архитектуре, базирующейся на механизмах внимания.

Трансформеры могут учитывать долгосрочные зависимости и исторический контекст, что делает их особенно ценными в таких областях, как автономное вождение, финансовое моделирование и робототехника. Эти области требуют учёта множества предшествующих событий и сложной динамики окружающей среды при принятии решений.

Также трансформеры способствуют обобщению и адаптации моделей к изменяющимся условиям, что критически важно в динамичных и непредсказуемых средах. Их способность интегрировать разнообразные типы информации — от текстовой до визуальной — позволяет создавать гибкие модели, способные принимать обоснованные решения в сложных и немарковских условиях.

3.2 Описание предложенной модели и разработанного алгоритма

Модель интеграции алгоритмов обучения с подкреплением и кодировщика трансформера, представляющая собой концептуальный подход к обработке последовательностей состояний. Основная идея модели заключается в том, что вместо использования марковского предположения, как в классических алгоритмах обучения с подкреплением, где текущее состояние полностью определяет будущее поведение системы, предлагается учитывать последовательность предыдущих

состояний. Для этого используется кодировщик трансформера, который преобразует последовательности состояний в более репрезентативные латентные представления, содержащие информацию о динамике среды и истории взаимодействий агента.

В рамках данного исследования, на основе предложенной выше модели, разработан алгоритм интегрирующий кодировщик трансформера с алгоритмами Soft Actor-Critic. Предложенный подход включает использование трансформера в роли кодировщика для обработки последовательностей состояний перед их передачей в сети актора и критика, характерные для SAC. Этот подход не только способствует более глубокому представлению состояний, но и позволяет учитывать контекстную информацию, что является критически важным в динамически изменяющихся и сложных условиях.

Разработанный алгоритм начинается с кодировщика трансформера, который преобразует входные последовательности состояний в латентное пространство. Полученные латентные представления далее используются в сетях актора и критика в контексте стандартного SAC. Схематическое представление разработанного алгоритма обучения подкрепления, использующего архитектуру трансформер представлено на рисунке 19.

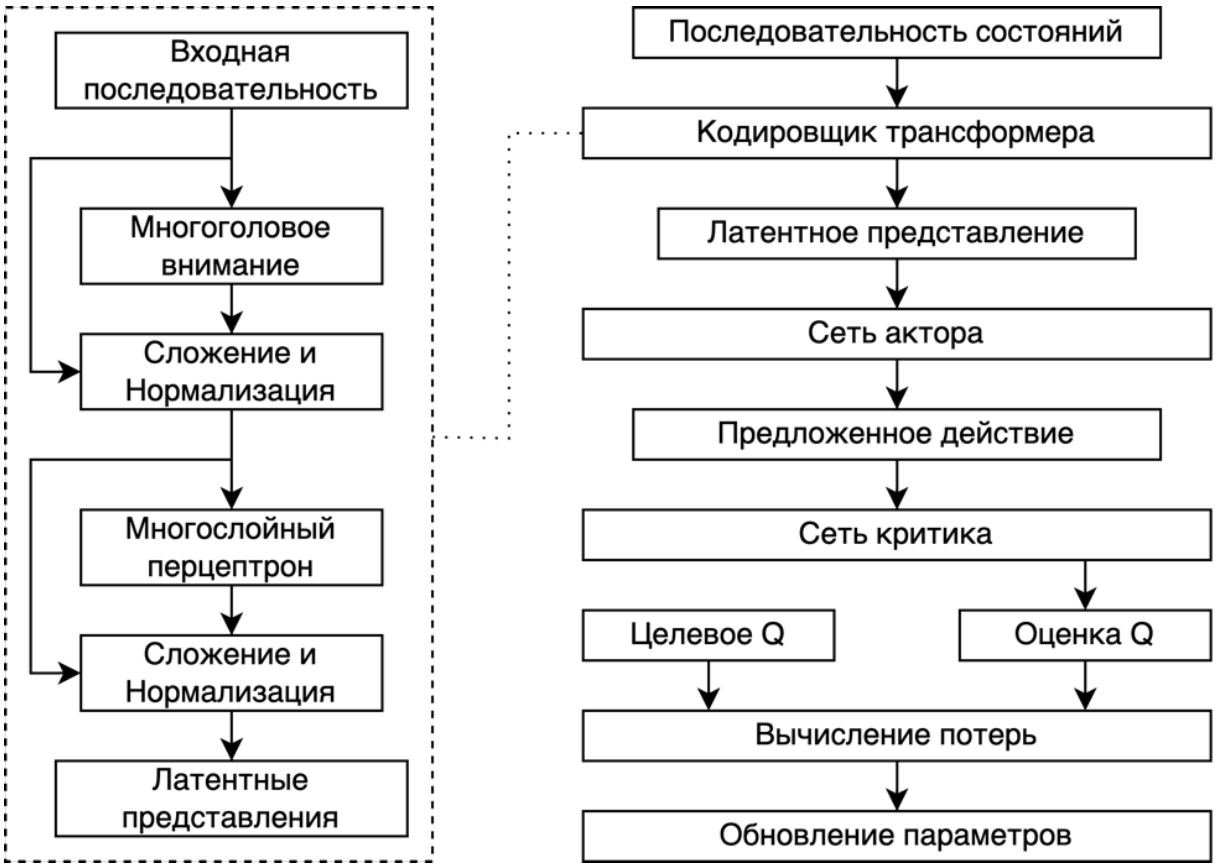

Рисунок 19 – Схема предложенного алгоритма обучения с подкреплением

Для эффективного использования трансформера в SAC, буфер воспроизведения был адаптирован для хранения и выборки последовательностей переходов, а не отдельных событий. Это изменение позволяет использовать контекстуальную информацию о предыдущих состояниях при принятии решений о будущих действиях, улучшая тем самым обучение агента.

Буфер воспроизведения представляет собой центральный элемент алгоритмов обучения с подкреплением, особенно для алгоритмов, основанных на различиях во времени, таких как SAC. В буфере сохраняются предыдущие переходы, включающие состояния, действия, вознаграждения, последующие состояния и информацию о завершении эпизода. Главная задача буфера воспроизведения заключается в повторном использовании накопленного опыта для повышения эффективности обучения и обеспечения стабильности, минимизируя влияние коррелированных данных.

В связи с интеграцией архитектуры трансформера в SAC появилась потребность адаптировать буфер воспроизведения для поддержки выборки последовательностей данных, а не отдельных событий. Трансформеры требуют обработки данных в форме последовательностей для корректного контекстного анализа и обработки временных зависимостей. Следовательно, буфер воспроизведения был модифицирован для эффективного хранения и предоставления последовательностей переходов.

Структура данных в буфере была реорганизована таким образом, чтобы данные хранились в виде последовательностей, каждая из которых содержит ряд последовательных переходов. Это изменение позволяет модели извлекать и использовать контекстную информацию из данных последовательностей.

Была разработана система выборки, которая извлекает последовательности данных из буфера для использования в процессе обучения. Данная система обеспечивает сохранение временной непрерывности данных в последовательностях и целостность временных переходов. Кроме того, были введены специальные меры для обработки границ эпизодов внутри последовательностей, чтобы предотвратить смешивание данных из различных эпизодов.

В реализованном алгоритме в качестве основы для кодирования используется кодировщик трансформера, состоящий из двух слоев и трех «голов» внимания. Рассмотрим пример применения этого кодировщика в среде Humanoid.

В этой среде входное состояние $s \in \mathbb{R}^{376}$ расширяется до размера, кратного числу голов внимания $n_{head} = 3$, путем дополнения нулями. Таким образом, размерность векторного пространства модели d_{model}устанавливается равной 378:

- Размерность головы внимания d_{head} вычисляется как $\frac{d_{model}}{n_{head}} = 126$.

Ключевые параметры трансформера включают следующие матрицы весов:

$$W_K, W_Q, W_V \in \mathbb{R}^{378\times126}$$

Далее применяется полносвязная сеть для проекций:

- Первая проекция: преобразование $\mathbb{R}^{378} \rightarrow \mathbb{R}^{2048}$.

– Вторая проекция: преобразование $\mathbb{R}^{2048} \to \mathbb{R}^{378}$.

3.3 Методология экспериментального исследования

Для проведения экспериментальных исследований были выбраны симулированные среды из пакета gymnasium: Ant, Humanoid, Half-Cheetah, Hopper и Walker. Эти среды обеспечивают разнообразие и моделируют реальные физические взаимодействия и движения. Оценка производительности агентов проводилась по скорости обучения, эффективности действий и способности адаптироваться к изменяющимся условиям.

Среда Ant моделирует трёхмерного робота-муравья с четырьмя ногами, задача которого — координировать их движение для продвижения вперёд. В среде Humanoid [84] агент должен двигаться вперёд, избегая падений, управляя двуногим роботом. В Half-Cheetah [83] задача состоит в максимизации скорости движения вперёд с помощью робота с восьмью сочленениями. В среде Hopper [85] агент управляет одноногим роботом, выполняя прыжки для продвижения. В Walker агент решает задачу координации движений робота с двумя ногами для передвижения вперёд.

Для оценки изменений в качестве процесса обучения введём величину, характеризующую изменение суммарной награды, полученной алгоритмом A, по сравнению с алгоритмом B:

$$Improvement\ (\%) = \frac{\sum_{i=1}^{10} R_{A,\, argsort(R_A)_i} - \sum_{i=1}^{10} R_{B,\, argsort(R_B)_i}}{\sum_{i=1}^{10} R_{B,\, argsort(R_B)_i}} \times 100,$$

где $R_A = \{R_{A,1}, R_{A,2},\ \ldots R_{A,N}\}$ – массив наград для метода A,

$R_B = \{R_{B,1}, R_{B,2},\ \ldots R_{B,N}\}$ – массив наград для метода B,

$argsort(x)$ – функция, которая возвращает индексы элементов вектора, отсортированных по убыванию, таким образом $\sum_{i=1}^{10} R_{A,\, argsort(R_A)_i}$ – сумма значений 10 наибольших наград алгоритма A.

3.4 Результаты экспериментального исследования

В рамках проведенных экспериментальных исследований было выявлено, что интеграция архитектуры трансформера с методом Soft Actor-Critic приводит к значительному улучшению производительности агента в сложных трехмерных средах. В частности, было отмечено увеличение выборочной эффективности решения задач, что проявлялось в повышении значения награды агентов на 5–10% по сравнению с классическим SAC при аналогичном количестве шагов обучения, как видно из графиков на рисунках 20–24. В таблице 3 приведены значения изменения суммарной награды разработанного алгоритма относительно оригинального SAC для каждой среды.

Таблица 3 – Изменение суммарной награды при использовании разработанного алгоритма

Среда	Разница
BipedalWalker	-5,34%
Walker	+103,86%
Ant	+45,07%
Hopper	+38,90%
Humanoid	-90,04%

Тем не менее, как показали результаты экспериментальных исследований, улучшение производительности наблюдалось не во всех тестовых средах. В частности, в среде Half-Cheetah интеграция трансформеров не всегда приводила к повышению результатов, иногда даже отмечалось их ухудшение.

Выборочная эффективность, ключевой показатель в обучении с подкреплением, отражает способность алгоритма достигать высокого уровня производительности, используя ограниченное количество обучающих примеров. В условиях, где сбор данных обходится дорого или временно ограничен, высокая выборочная эффективность позволяет агентам быстрее адаптироваться к среде и минимизировать количество требуемых итераций обучения.

В представленном исследовании было выполнено тестирование различных конфигураций кодировщика трансформера в симуляциях HalfCheetah и Humanoid, что позволило изучить, как изменения в архитектуре кодировщика влияют на обработку и интерпретацию входных данных. Полученные результаты показали, что архитектурные вариации трансформера значительно влияют на эффективность решения задач в разных симулированных средах.

Интеграция трансформера с SAC предлагает преимущества в виде улучшенной способности агентов к интерпретации состояний и адаптации к динамическим средам. Однако это сопряжено с повышенной вычислительной сложностью и необходимостью тщательной настройки параметров.

Этот подход может найти применение в робототехнике для улучшения производительности роботов, в играх для разработки адаптивного искусственного интеллекта и в автоматизированном управлении, например, в автономном вождении.

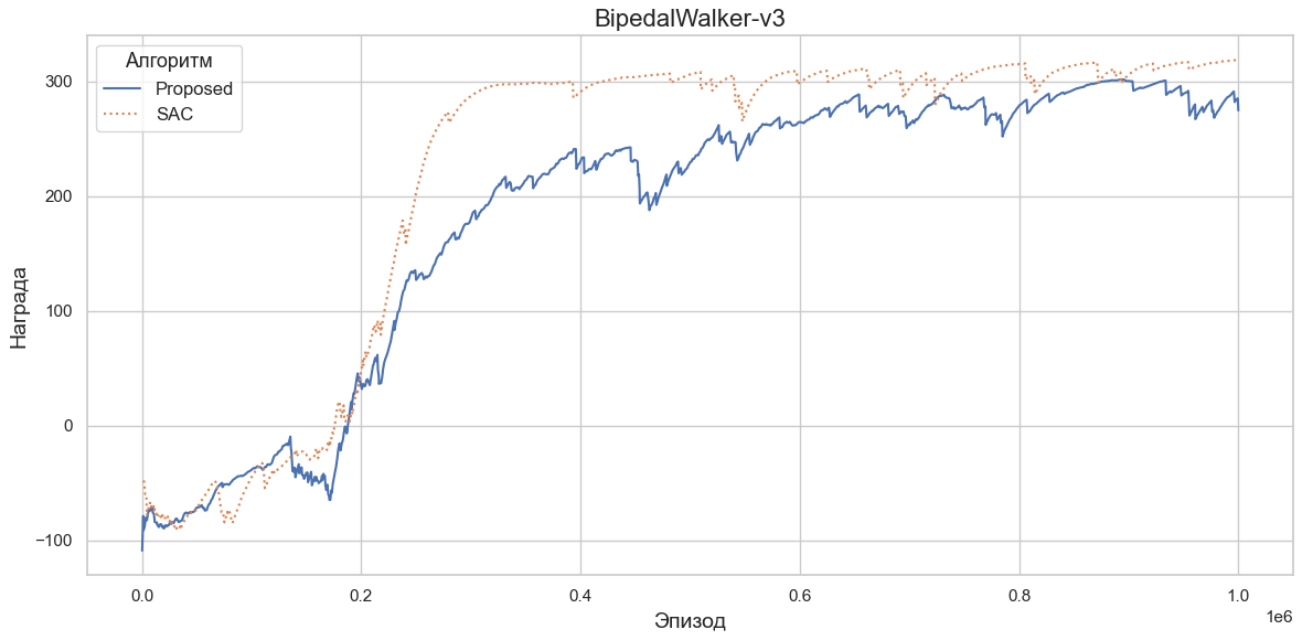

Рисунок 20 – График зависимости значения награды от шага обучения для среды HalfCheetah-v4

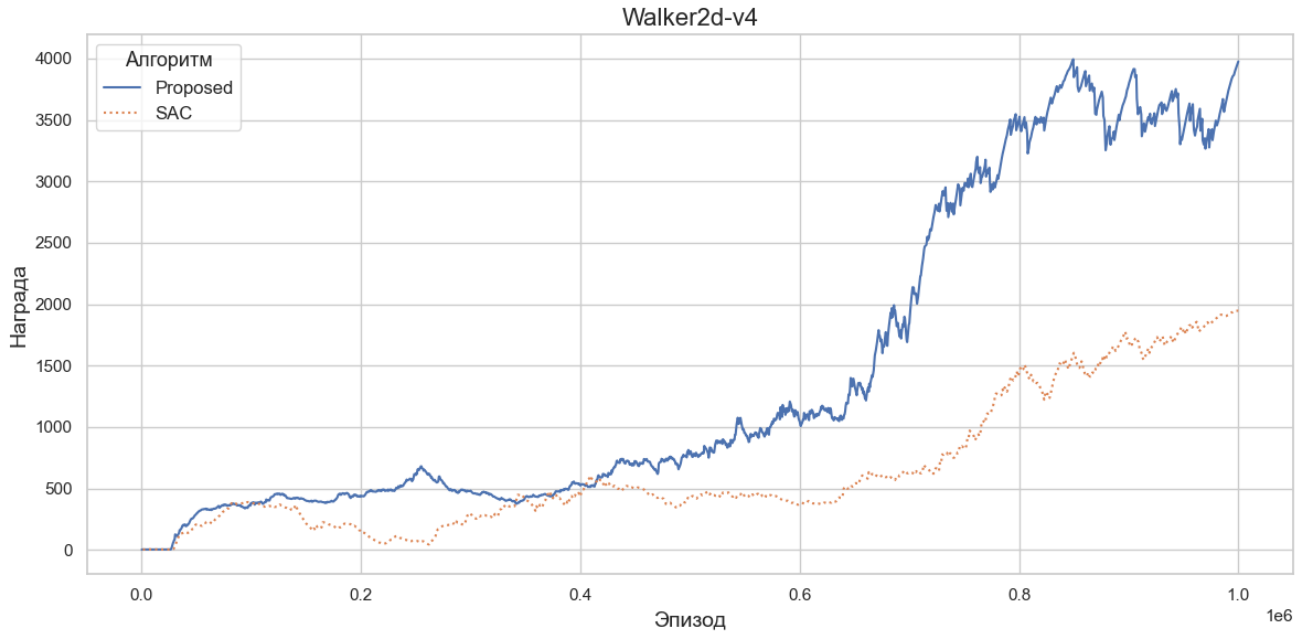

Рисунок 21 – График зависимости значения награды от шага обучения для среды Walker2d-v4

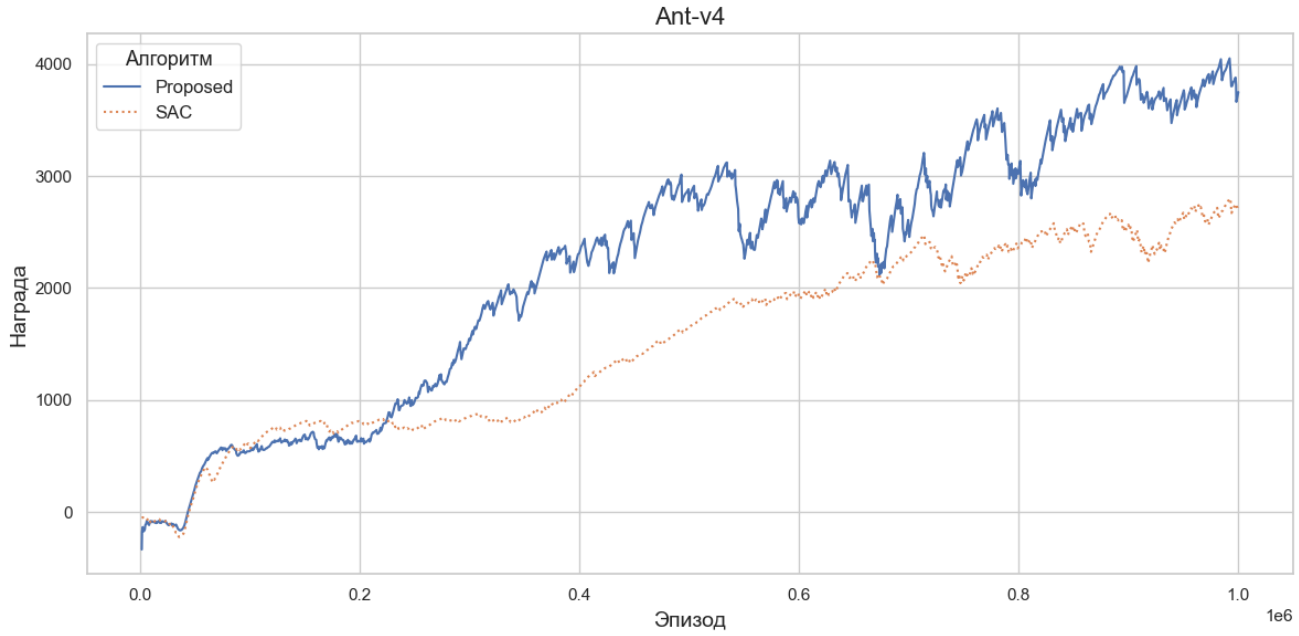

Рисунок 22 – График зависимости значения награды от шага обучения для среды Ant-v4

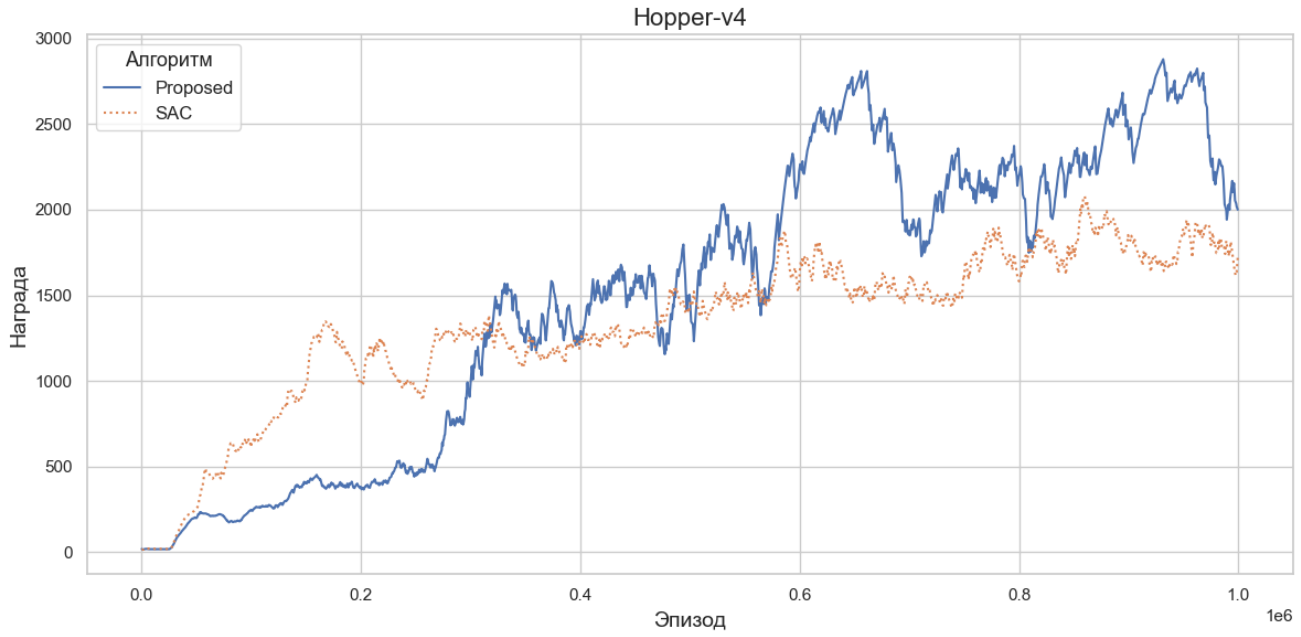

Рисунок 23 – График зависимости значения награды от шага обучения для среды Hopper-v4

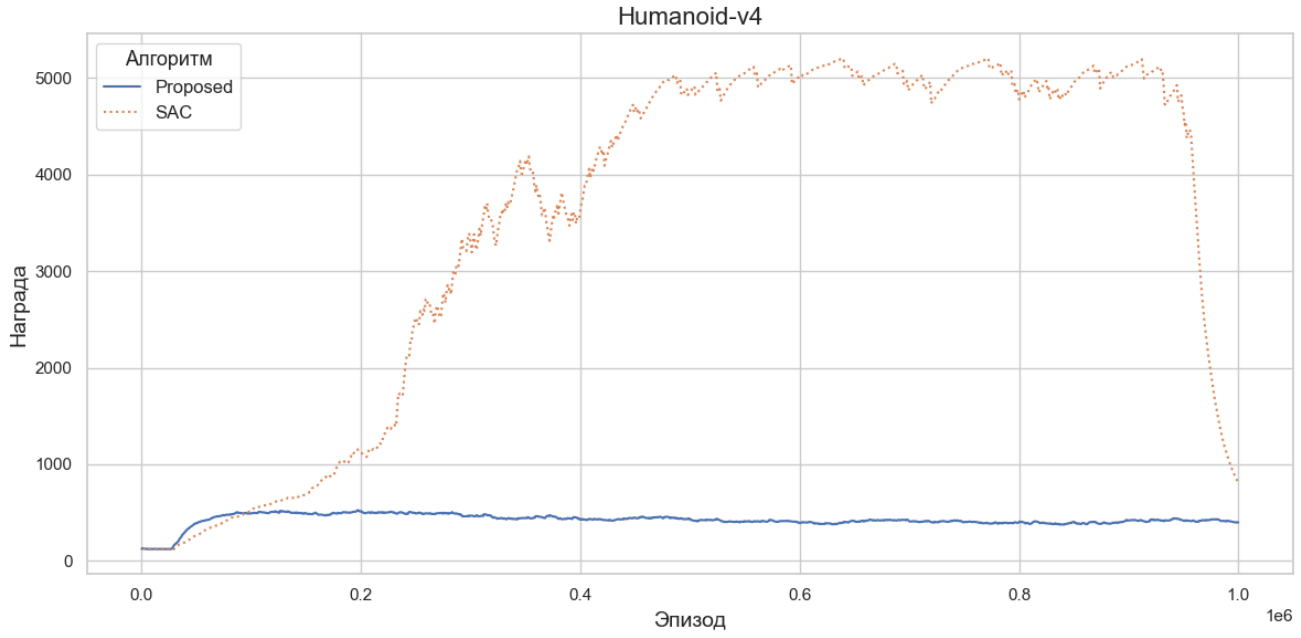

Рисунок 24 – График зависимости значения награды от шага обучения для среды Humanoid-v4

3.5 Выводы и результаты третьего раздела

В третьем разделе диссертации было проведено исследование разработанной модели интеграции алгоритмов обучения с подкреплением с кодировщиком трансформера, а также разработанного на основе модели алгоритма, интегрирующий кодировщик трансформера с алгоритмом Soft Actor-Critic (SAC).

Было продемонстрировано, что использование кодировщика трансформера может существенно улучшить обработку последовательностей состояний, что приводит к повышению эффективности обучения и улучшению обобщающей способности моделей. Экспериментальные исследования показали, что данный подход улучшает производительность в различных сложных задачах. Разработанный алгоритм обеспечивает улучшение среднего суммарного значения награды на 18,5% (преимущество). Частота случаев, когда результаты обучения по сравнению с оригинальным алгоритмом SAC улучшаются или остаются на прежнем уровне, составляет 80%.

В практическом плане предложенный подход предоставляет широкие перспективы для применения. Он может быть использован для создания более эффективных робототехнических систем, улучшения существующих систем искусственного интеллекта в играх и симуляциях, а также для разработки надежных систем автоматизированного управления.

Ключевой особенностью разработанного алгоритма обучения с подкреплением с использованием архитектуры трансформер является тот факт, что предложенный алгоритм способен более эффективно обрабатывать случаи, не являющиеся марковскими процессами. Несмотря на то, что все процессы в обучении с подкреплением абстрактно представляются как марковский процесс в реальности редко можно найти задачу, которую полностью можно описать при помощи марковского процесса. Либо количество переменных в таком процессе может превысить все разумные пределы.

По теме раздела опубликованы работы [*18-19].

РАЗДЕЛ 4. Метод иерархического ансамблирования алгоритмов обучения с подкреплением

В рамках данного диссертационного исследования был разработан ансамблевый метод для алгоритмов обучения с подкреплением, цель которого — повышение общей эффективности по сравнению с применением отдельных алгоритмов. В конкретно рассмотренном случае ансамбль включает в себя алгоритмы REDQ и SAC. Выбор результата осуществляется на основе вывода алгоритма DQN, который выполняет функцию контрольного алгоритма. Предложенная методика позволяет интегрировать дополнительные алгоритмы и управлять их количеством в ансамбле.

Обучение с подкреплением является перспективной областью машинного обучения, где ключевой задачей остаётся решение сложных задач с использованием мета-алгоритмов. Предложенный подход может найти применение в решении комплексных задач, состоящих из множества подзадач, для которых эффективные решения предлагаются различными алгоритмами в рамках ансамбля.

Одной из ключевых проблем в обучение с подкреплением является обобщение обучения на новые, ранее не встреченные среды или задачи. Мета-алгоритмы в обучении с подкреплением направлены на разработку стратегий, которые могут адаптироваться к новым условиям без переобучения с нуля. В данном диссертационном исследовании предложен ансамблевый подход как метод обобщения, позволяющий интегрировать несколько алгоритмов для обработки широкого спектра задач в рамках единой системы.

В процессе исследования была сформулирована гипотеза, что использование ансамбля нескольких алгоритмов обучения с подкреплением, управляемого разработанным методом, может улучшить выборочную эффективность. Выборочная эффективность (sample efficiency) [82] в контексте обучения с подкреплением — это показатель, отражающий способность алгоритма достигать высокого уровня производительности с использованием ограниченного числа обучающих примеров или взаимодействий с окружающей средой. Она оценивает,

насколько эффективно алгоритм использует собранные данные для улучшения своей стратегии действий и достижения максимального вознаграждения. Контрольный алгоритм ансамбля имеет дискретный выход и при переходе в новое состояние все алгоритмы выполняют оптимизационный шаг. Выбор конкретного действия осуществляется в соответствии с выводом контрольного алгоритма, что позволяет наиболее эффективно использовать алгоритм в данном состоянии. Дополнительно предлагается использование иерархии алгоритмов, что способствует агрегации вывода ансамбля.

4.1 Описание предложенного метода

На рисунке 25 представлена блок-схема общего предложенного метода. Ансамбль включает в себя несколько алгоритмов, причем действия каждого алгоритма сохраняются в буфере, из которого затем производится выбор с помощью контрольного алгоритма. Тот же самый ансамбль алгоритмов может представлять каждый алгоритм ансамбля, что позволяет создавать сложные иерархические структуры ансамблей алгоритмов, способных решать задачи, состоящие из большого числа подзадач.

Каждый алгоритм в ансамбле получает информацию о задаче: вознаграждение, текущее состояние, следующее состояние и последнее действие. Контрольный алгоритм также получает эту информацию, однако вместо данных о действии он получает сведения о последнем управляющем действии или мета-действии. После выполнения действия необходимо обновить настройки стратегии всего ансамбля. Для этого собираются все значения функций потерь, после чего выполняется шаг оптимизации.

Блок выбора действия предполагает, что для мета-действия, закодированного с помощью унитарного кодирования [92] длиной N, будет сравниваться набор из N действий, и из них будет выбрано действие, для которого в мета-действии будет указана единица. Эта система позволяет точно и эффективно управлять выбором

действий в зависимости от текущих условий и задач, стоящих перед ансамблем алгоритмов.

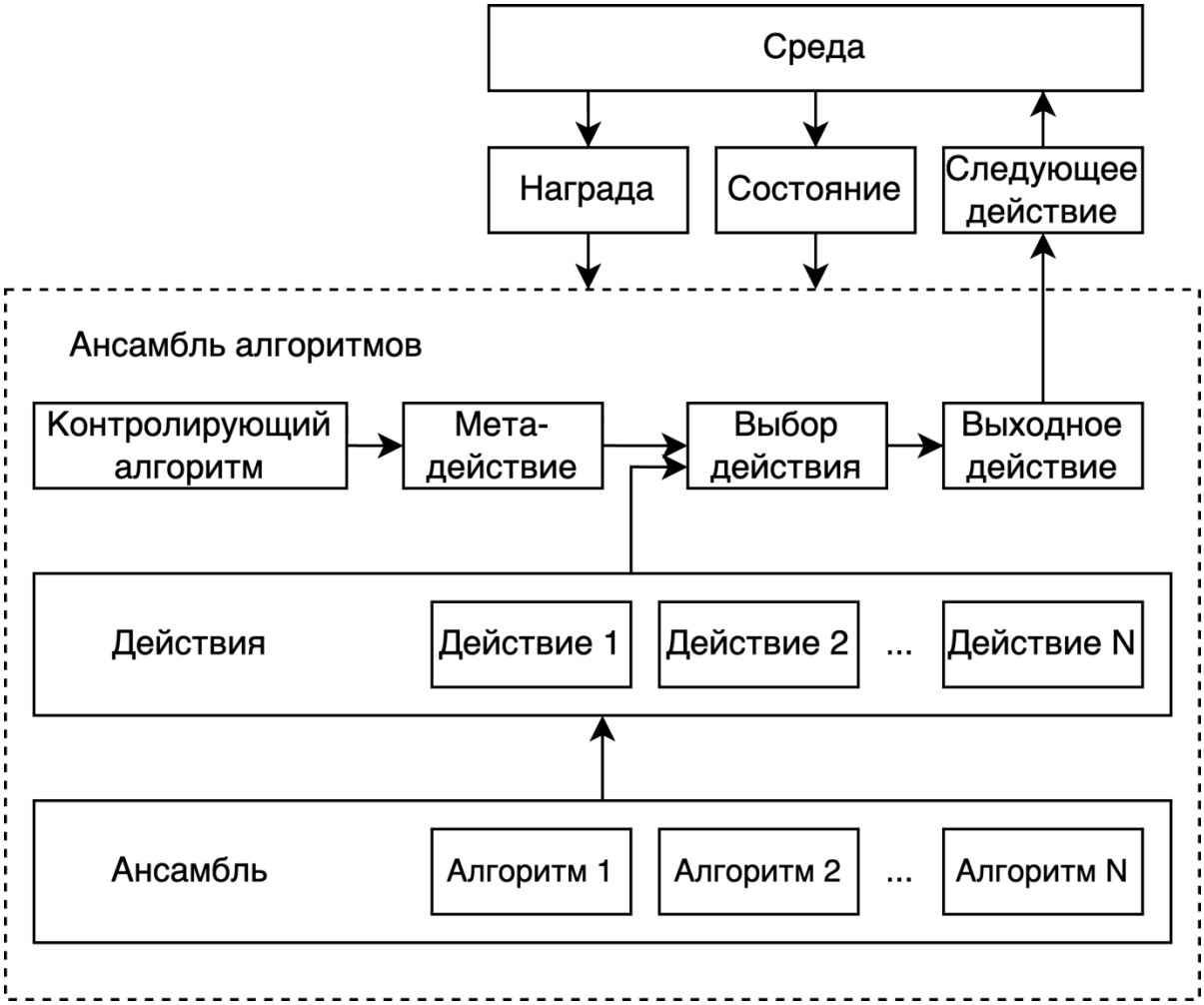

Рисунок 25 – Блок схема предложенного метода

4.2 Реализация предложенного метода

В рамках данного исследования рассматривается модификация предложенного метода, которая включает алгоритм DQN в качестве верхнего алгоритма в иерархии или контрольного алгоритма. Действия, предложенные этим алгоритмом, будут называться мета-действиями. В этой задаче к четырем данным, полученным от окружающей среды, добавляется мета-действие.

Блок-схема разработанной структуры представлена на рисунке 26. Алгоритм включает следующие шаги:

1. Получение состояния, вознаграждения, следующего состояния от среды и получение от агента действия и мета-действия, предпринятого на последнем шаге.

2. Передача информации о состоянии и вознаграждении всем алгоритмам, информации о действии алгоритмам ансамбля и информации о мета-действии управляющему алгоритму.

3. Расчет значений функции потерь для каждого алгоритма.

4. Выполнение шага оптимизации параметров каждого алгоритма с использованием алгоритма стохастического градиентного спуска Adam [93]. Каждое обновление происходит на основе градиентов, полученных в результате применения алгоритма обратного распространения ошибок [94] для следующего значения функции потерь:

$$Loss = Loss_{REDQ} + Loss_{DQN} + Loss_{SAC}$$

5. Получение действий от каждого алгоритма и мета-действия. На основе дискретного мета-действия выбирается одно из действий алгоритмов ансамбля в качестве выходного и передается вместе с мета-действием в среду. Мета-действие передается для обобщения задачи и получения его на следующем шаге.

Также была изучена реализация алгоритма, в которой шаг номер 4 отличается. Вместо совместной оптимизации параметров нейронных сетей в каждом алгоритме, оптимизация происходила для каждого алгоритма поочередно. Но такая реализация показала худший результат, поэтому в экспериментальных исследованиях не участвует.

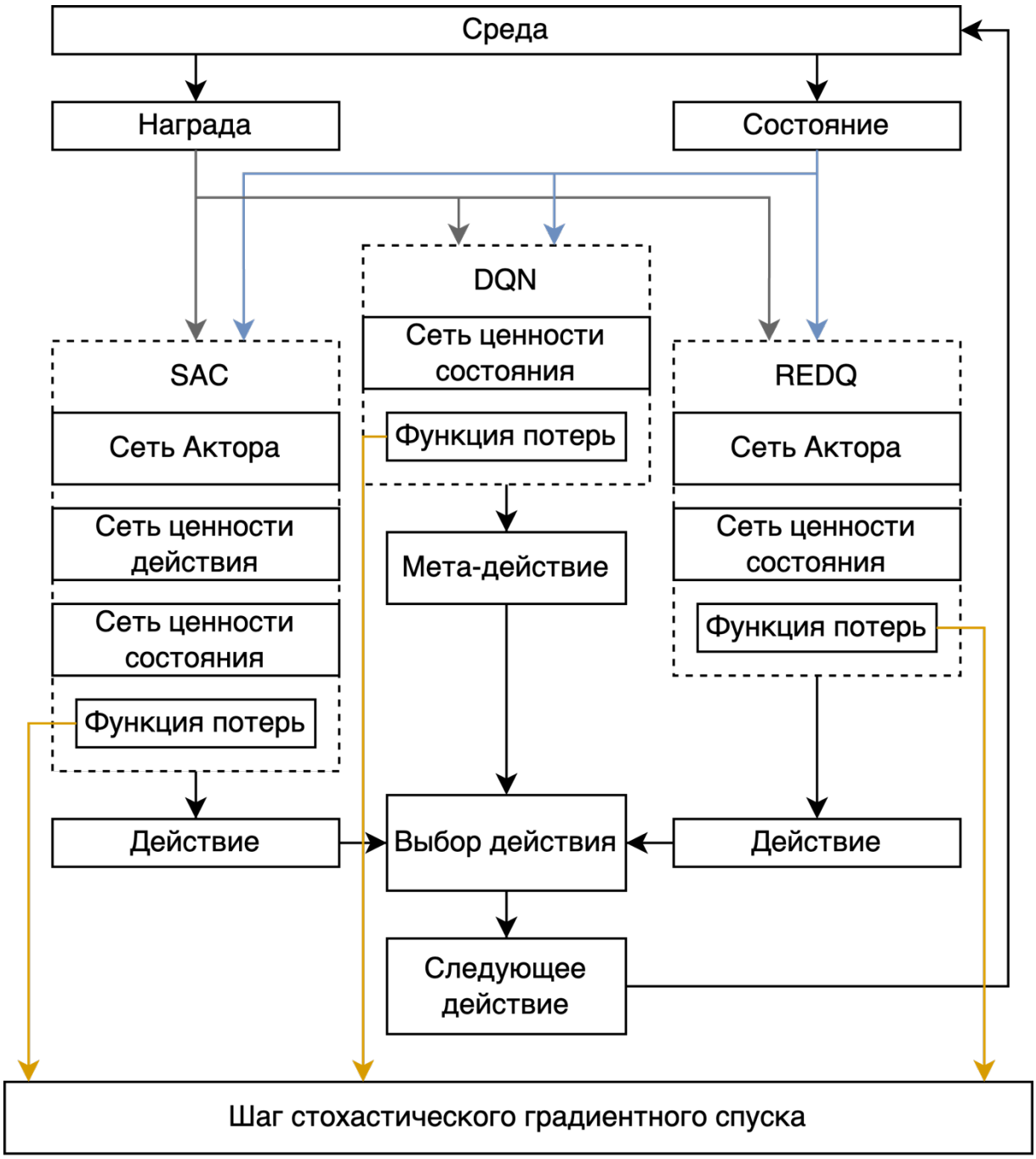

Рисунок 26 – Реализованный алгоритм, версия с совместной оптимизацией параметров сетей

Ключевой особенностью рассматриваемого алгоритма является его способность к распространению опыта между различными алгоритмами в ансамбле. Эта особенность заключается в использовании механизмов обучения с подкреплением вне стратегии, что позволяет эффективно обучать все алгоритмы ансамбля, даже если они не были активированы на текущем этапе.

Каждый управляемый алгоритм в ансамбле может функционировать в двух режимах:

- Режим действия. В данном режиме алгоритм выбирается управляющим алгоритмом и выполняет свои функции в обычном режиме, непосредственно взаимодействуя с окружающей средой.
- Режим наблюдения. В этом режиме алгоритм обучается на основе наблюдаемого опыта, что позволяет ему накапливать знания и адаптироваться к изменениям в среде.

4.3 Методология экспериментального исследования

Предложенный алгоритм реализован с использованием фреймворка TorchRL. В рамках экспериментальных исследований были протестированы реализации алгоритмов REDQ, SAC и предложенного алгоритма в среде dm-control, которая повторяет среды gymnasium, использующие MuJoCo, в задачах «cheetah run», человекообразный робот в задачах «stand» и «run pure state», а также в среде «walker» в задаче ходьба «walk». Каждый тест включал 10^6 шагов обучения (взаимодействий с окружающей средой). Все среды направлены на решение задачи приобретения навыков передвижения в пространстве, что представляет интерес с точки зрения прикладной робототехники. Все среды обладают большим числом степеней свободы. Каждый 3D-сустав требует получения 3D-вектора для установки его положения в пространстве. Например, если суставов пять, задача будет иметь 15 степеней свободы и т. д.

В среде Cheetah агент представляет собой двумерное двуногое существо, напоминающее гепарда. У агента шесть суставов. Вознаграждение агента прямо пропорционально его скорости движения вперед до максимальной скорости в 10 метров в секунду. Скорости свыше 10 метров в секунду вознаграждаются аналогично скоростям свыше 10 метров в секунду. Изображено на рисунке 27 слева.

Агент Humanoid представляет собой упрощенную модель человеческого тела с 21 суставом. Задачи различаются целевой горизонтальной скоростью в 0 метров

в секунду для задачи «stand». Задача агента заключается в том, чтобы оставаться в вертикальном положении. Целевая скорость для задачи «run» составляет 10 метров в секунду. В этой задаче поощряется любая горизонтальная скорость, то есть агент может двигаться вбок или назад. Это создает необходимые условия для исследования возможных локальных минимумов. Изображено на рисунке 27 в середине.

В среде Walker агент представляет собой трехмерное существо с двумя ногами. Общее количество суставов составляет шесть. Награда сочетает в себе компоненты, поощряющие вертикальную осанку, минимальную высоту торса над землей и горизонтальную скорость движения вперед. Изображено на рисунке 27 справа.

Рисунок 27 – Среды из пакета dm-control. Cheetah (слева), Humanoid (посередине), Walker (справа)

4.4 Результаты экспериментального исследования

На графиках, представленных на рисунках 28–32, отображается вознаграждение агента в каждой задаче и среде, описанных выше, в зависимости от шага обучения. Предложенный подход в среднем обеспечивает более высокий уровень вознаграждения, за исключением задачи ходьбы. В каждом эксперименте проводилось три независимых запуска алгоритма в данной среде. Для каждого шага вознаграждения из трех запусков были усреднены, что гарантирует систематический, а не случайный характер преимущества. Вознаграждения

усреднялись с использованием экспоненциально взвешенного сглаживания с коэффициентом, равным 0,25, что позволяет лучше проследить тенденцию на представленных графиках. В таблице 4 приведены значения изменения суммарной награды разработанного алгоритма относительно оригинального SAC для каждой среды.

Таблица 4 – Изменение суммарной награды при использовании разработанного алгоритма

Среда	Разница
Walker walk	-0,39%
Cheetah run	+4,32%
Humanoid run pure state	+4,41%
Humanoid run	+1,40%
Humanoid stand	+3,52%

Также было проведено исследование выборочной и временной эффективности предложенного алгоритма. Графики на рисунках 28–32 показывают, что в среднем алгоритм демонстрирует несколько более высокую выборочную эффективность по сравнению с SAC или REDQ. Выборочная эффективность измеряется количеством обучающих шагов, необходимых для достижения определённого уровня вознаграждения, что позволяет разработанному алгоритму достигать аналогичного уровня вознаграждения, что и SAC или REDQ, за меньшее количество взаимодействий с окружающей средой.

Что касается временной эффективности, то есть обратной величины чистого времени, требуемого для обучения алгоритма, она уменьшается пропорционально числу алгоритмов в ансамбле. Если для SAC или REDQ требовалось около 5-7 часов для выполнения 10^6 шагов на процессоре i7-9700K и графическом процессоре RTX 2080Ti, то разработанный алгоритм требует примерно 10-11 часов для изучения того же количества шагов. Однако стоит учесть, что эксперименты проводились в виртуальной среде моделирования, что позволяет ускорить различные физические процессы. В реальной системе разница во времени, которое требуется затратить на взаимодействие с окружающей средой, могла бы не проявляться, поскольку время, необходимое для оптимизации, было бы меньше по

сравнению со временем, требуемым на взаимодействие с реальной средой. В таком случае миллион взаимодействий с реальной средой мог бы занять до 12 дней непрерывного обучения, при условии, что один шаг взаимодействия с окружающей средой занимает приблизительно одну секунду.

В контексте исследования различные временные затраты алгоритмов указывают на их разную производительность при масштабировании и адаптации к реальным условиям. Это подчеркивает необходимость дальнейшего анализа не только выборочной, но и временной эффективности алгоритмов, чтобы обеспечить их практическую применимость и оптимизацию.

Таким образом, результаты проведенных экспериментальных исследований демонстрируют, что предложенный алгоритм обеспечивает улучшение в выборочной эффективности, что позволяет достигать аналогичного уровня вознаграждения с меньшим количеством взаимодействий с окружающей средой. Это открывает перспективы для разработки более экономичных и эффективных систем обучения, способных работать в условиях ограниченных ресурсов или в реальном времени. Однако приложение этих методов в реальном мире потребует дополнительной работы по адаптации алгоритмов к ограничениям, связанным с физическими аспектами взаимодействия сред и длительностью обучения.

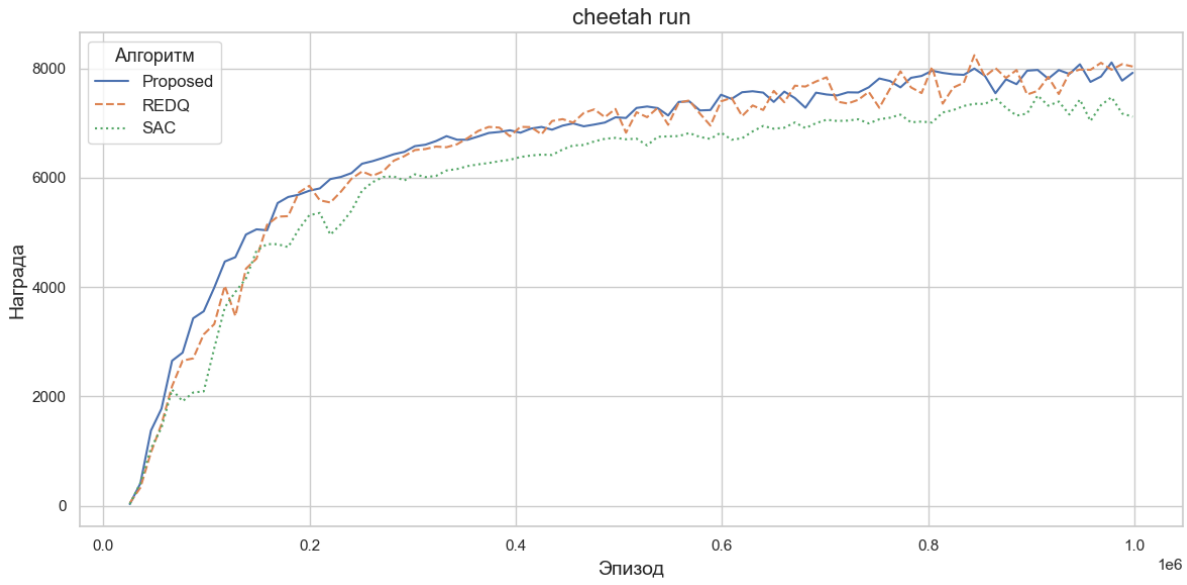

Рисунок 28 – Сравнение предложенного метода с SAC и REDQ в среде «cheetah run»

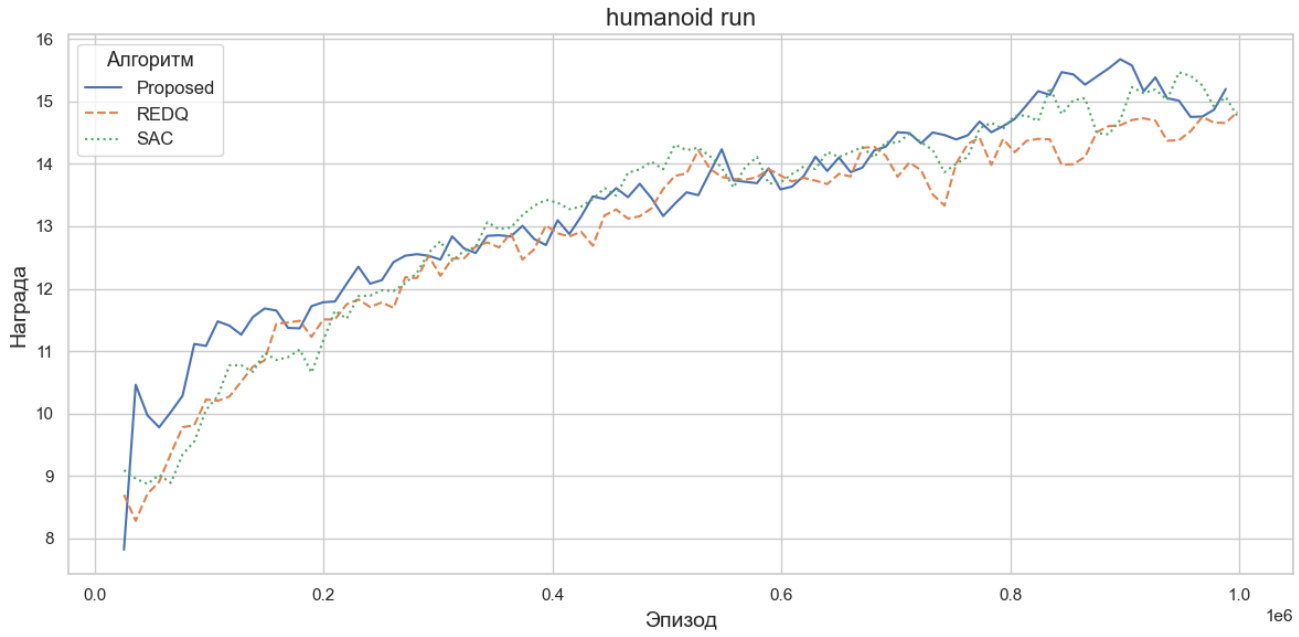

Рисунок 29 – Сравнение предложенного метода с SAC и REDQ в среде «humanoid run»

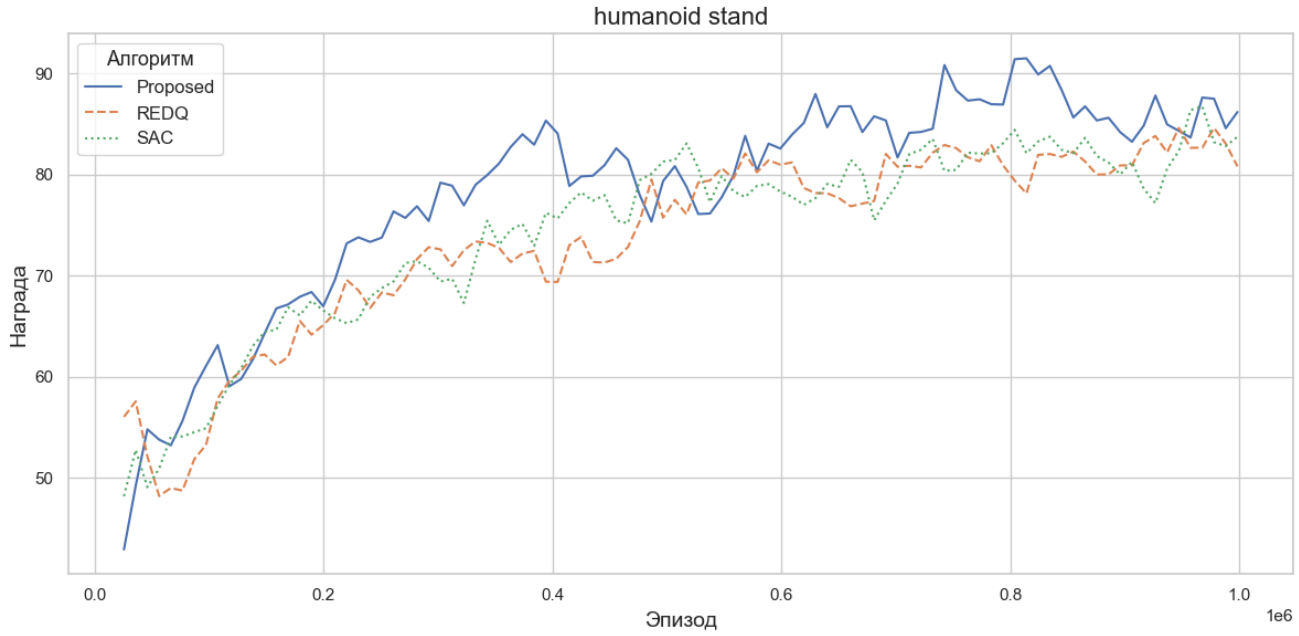

Рисунок 30 – Сравнение предложенного метода с SAC и REDQ в среде «humanoid stand»

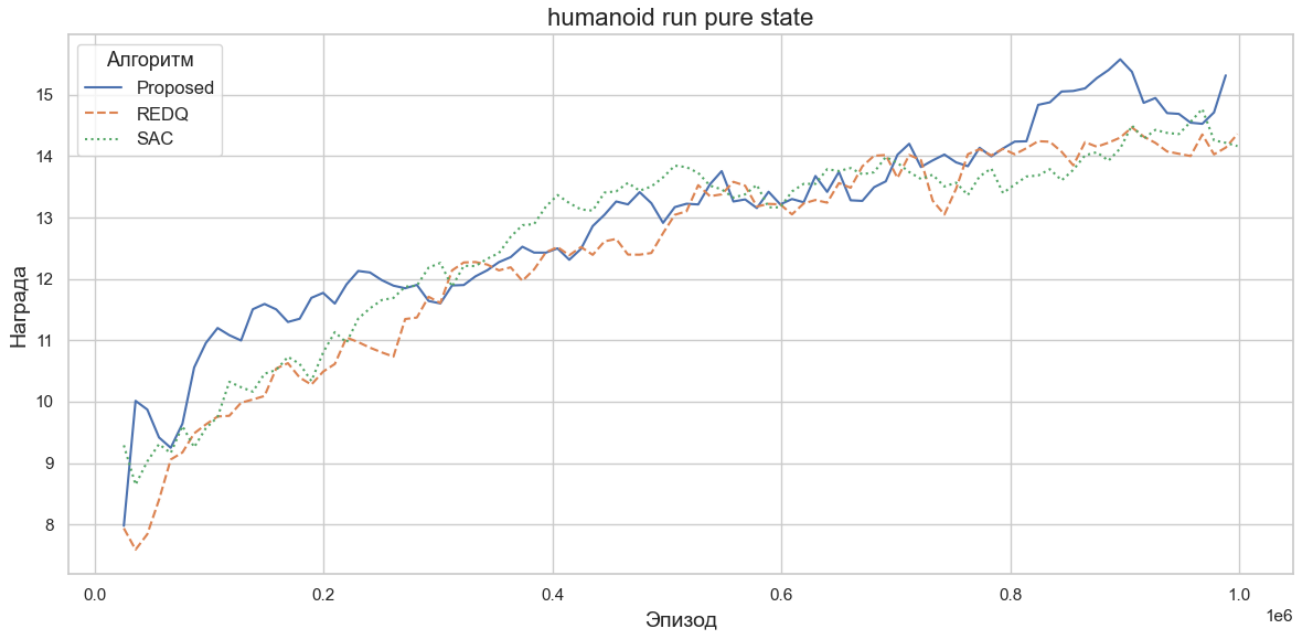

Рисунок 31 – Сравнение предложенного метода с SAC и REDQ в среде «humanoid run pure state»

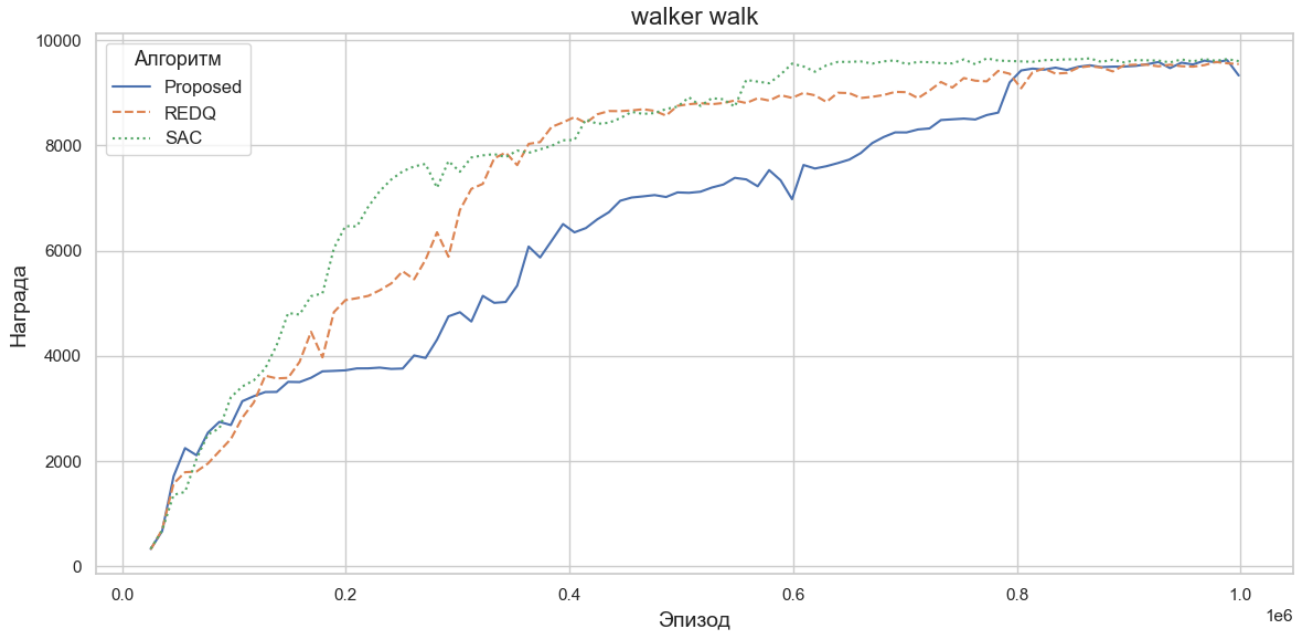

Рисунок 32 – Сравнение предложенного метода с SAC и REDQ в среде «walker walk»

4.5 Выводы и результаты четвертого раздела

В третьем разделе диссертационного исследования был рассмотрен разработанный ансамблевый метод обучения с подкреплением и проведена его оценка эффективности.

Так, используя алгоритмы REDQ и SAC в качестве основных компонентов и DQN для контроля ансамбля, был разработан алгоритм на основе метода. Разработанный алгоритм продемонстрировал значительное улучшение производительности в задачах управления динамическими системами. Разработанный алгоритм обеспечивает улучшение среднего суммарного значения награды на 2,65%. При этом результаты обучения по сравнению с лучшим алгоритмом ансамбля улучшаются или остаются на прежнем уровне во всех случаях.

В заключение, разработанный ансамблевый метод обучения с подкреплением демонстрирует значительный потенциал для улучшения решения сложных задач в области искусственного интеллекта и робототехники, предлагая значительные преимущества в адаптивности и эффективности по сравнению с одиночными алгоритмами.

Ключевой отличительной особенностью разработанного иерархического ансамблевого метода обучения с подкреплением является тот факт, что такие алгоритмы можно использовать для сложных сред, в которых возможно выделение некоторых более простых подзадач, в результате создавая самоадаптирующиеся иерархии алгоритмов, в которых отдельные алгоритмы обучаются отдельным подзадачам. По теме раздела опубликованы работы [*13-17].

ЗАКЛЮЧЕНИЕ

В данном диссертационном исследовании разработаны и исследованы методы, алгоритмы и способы повышения качественных показателей алгоритмов обучения с подкреплением в рамках класса задач управления роботами, способными к перемещению в трехмерных средах.

Основными результатами работы являются:

1. Методика оценки влияния состава набора наблюдений окружающей среды на качество решений, принимаемых агентом, позволяющая упорядочить наблюдения по их полезности. Проведенное исследование показало, что избыточная информация о состоянии среды может ухудшать качество решений, принимаемых агентом.
2. Модель интеграции алгоритмов обучения с подкреплением и кодировщика трансформера, которая позволяет учитывать сложную динамику системы и справляться с зашумленными и немарковскими средами. На основе данной модели был разработан алгоритм, интегрирующий кодировщик трансформера с алгоритмом Soft Actor-Critic (SAC). Эксперименты показали, что предложенная интеграция улучшает среднее суммарное значение награды на 18,5%, а также в 80% случаев результаты превосходят или остаются на уровне оригинального SAC.
3. Метод иерархического ансамблирования алгоритмов обучения с подкреплением, который объединяет несколько алгоритмов в иерархическую структуру, что позволяет повысить качество обучения без дополнительных обращений к среде. Исследование продемонстрировало, что предложенный метод организует взаимодействие между управляющими и управляемыми алгоритмами, улучшая качественные показатели конечного решения.
4. Алгоритм обучения с подкреплением на основе метода иерархического ансамблирования, использующий алгоритм DQN в качестве

управляющего и алгоритмы SAC и REDQ в качестве управляемых. Экспериментальные данные показали улучшение среднего суммарного значения награды на 2,65% по сравнению с лучшим из отдельных алгоритмов, а также превосходство или паритет по качеству во всех экспериментах.

СПИСОК ЛИТЕРАТУРЫ

1. Dulac-Arnold G. Challenges of Real-World Reinforcement Learning / G. Dulac-Arnold, D. Mankowitz, T. Hester — 2019. — DOI: 10.48550/arXiv.1904.12901.

2. Yu, L.A Review of Deep Reinforcement Learning for Smart Building Energy Management / L. Yu, S. Qin, M. Zhang, C. Shen, T. Jiang, X. Guan // IEEE Internet of Things Journal. — 2021. — T. 8(15). — C. 12046–12063. — DOI: 10.1109/JIOT.2021.3078462.

3. Zhou, S.K. Deep reinforcement learning in medical imaging: A literature review / S.K. Zhou, H.N. Le, K. Luu, H. V Nguyen, N. Ayache // Medical Image Analysis. — 2021. — T. 73. — C. 102193. — DOI: 10.1016/j.media.2021.102193.

4. Kiran, B.R. Deep Reinforcement Learning for Autonomous Driving: A Survey / B.R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A.A.A. Sallab, S. Yogamani, P. Pérez // IEEE Transactions on Intelligent Transportation Systems. — 2022. — T. 23(6). — C. 4909–4926. — DOI: 10.1109/TITS.2021.3054625.

5. Li, C. Deep reinforcement learning in smart manufacturing: A review and prospects / C. Li, P. Zheng, Y. Yin, B. Wang, L. Wang // CIRP Journal of Manufacturing Science and Technology. — 2023. — T. 40. — C. 75–101. — DOI: 10.1016/j.cirpj.2022.11.003.

6. Niroui, F. Deep Reinforcement Learning Robot for Search and Rescue Applications: Exploration in Unknown Cluttered Environments / F. Niroui, K. Zhang, Z. Kashino, G. Nejat // IEEE Robotics and Automation Letters. — 2019. — T. 4(2). — C. 610–617. — DOI: 10.1109/LRA.2019.2891991.

7. Song, Y. Autonomous Drone Racing with Deep Reinforcement Learning / Y. Song, M. Steinweg, E. Kaufmann, D. Scaramuzza // 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) — 2021. — P. 1205–1212. — DOI: 10.1109/IROS51168.2021.9636053.

8. Lee, J. Learning quadrupedal locomotion over challenging terrain / J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, M. Hutter // Science Robotics. — 2020. — T. 5(47). — C. eabc5986. — DOI: 10.1126/scirobotics.abc5986.

9. Levine, S. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection / S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, D. Quillen // The International Journal of Robotics Research. — 2018. — Vol. 37(4–5). — P. 421–436. — DOI: 10.1177/0278364917710318.

10. Sutton, R.S. Reinforcement Learning, second edition: An Introduction / R.S. Sutton, A.G. Barto. — Cambridge, Massachusetts London, England: Bradford Books, 2018. — 552 p.

11. Silver, D. Reward is enough / D. Silver, S. Singh, D. Precup, R.S. Sutton // Artificial Intelligence. — 2021. — Vol. 299. — P. 103535. — DOI: 10.1016/j.artint.2021.103535.

12. Chen, L. Decision Transformer: Reinforcement Learning via Sequence Modeling / L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, I. Mordatch — 2021.

13. Козлов, Д.А. Сравнение алгоритмов обучения с подкреплением для управления движением автономного робота в симуляторе Gazebo / Д.А. Козлов // Информационные технологии и нанотехнологии. Сборник трудов по материалам VII Международной конференции и молодежной школы. Изд-во Самарского Университета — Самара, 2021. — P. 21442.

14. Козлов, Д.А. Сравнение алгоритмов обучения с подкреплением в задаче приобретения навыков передвижения в трёхмерном пространстве / Д.А. Козлов // Информационные технологии и нанотехнологии. Сборник трудов по материалам VIII Международной конференции и молодежной школы. Изд-во Самарского Университета — Самара, 2022. — P. 41482.

15. Козлов, Д.А. Влияние состава наблюдений окружающей среды в задаче приобретения навыков передвижения в трёхмерном пространстве при использовании алгоритмов обучения с подкреплением / Д.А. Козлов, В.В. Мясников // Информационные технологии и нанотехнологии. Сборник трудов по материалам VIII Международной конференции и молодежной школы. Изд-во Самарского Университета — Самара, 2022. — P. 41502.

16. Козлов, Д.А. Метод ансамблирования алгоритмов обучения с подкреплением на основе иерархичности / Д.А. Козлов, В.В. Мясников // Информационные технологии и нанотехнологии. Сборник трудов по материалам IX Международной конференции и молодежной школы. Изд-во Самарского Университета — Самара, 2023. — P. 40602.

17. Козлов, Д.А. Применение трансформера для кодирования состояний в обучении с подкреплением / Д.А. Козлов // Информационные технологии и нанотехнологии. Сборник трудов по материалам X Международной конференции и молодежной школы. Изд-во Самарского Университета — Самара, 2024.

18. Kozlov, D. Comparison of Reinforcement Learning Algorithms for Motion Control of an Autonomous Robot in Gazebo Simulator / D. Kozlov // IEEE Xplore 2021 VI International Conference on Information Technology and Nanotechnology — 2021. — C. 1–5. — DOI: 10.1109/ITNT52450.2021.9649145.

19. Kozlov, D. Comparison of Reinforcement Learning Algorithms in Problems of Acquiring Locomotion Skills in 3D Space / D. Kozlov // IEEE Xplore 2022 VIII International Conference on Information Technology and Nanotechnology — 2022. — C. 1–5. — DOI: 10.1109/ITNT55410.2022.9848647.

20. Kozlov, D. The impact of a set of environmental observations in the problem of acquiring movement skills in three-dimensional space using reinforcement learning algorithms / D. Kozlov, V. Myasnikov // IEEE Xplore 2022 VIII International Conference on Information Technology and Nanotechnology — 2022. — C. 1–5. — DOI: 10.1109/ITNT55410.2022.9848598.

21. Kozlov, D. Ensemble Method for Reinforcement Learning Algorithms Based on Hierarchy / D. Kozlov, V. Myasnikov // IEEE Xplore 2023 IX International Conference on Information Technology and Nanotechnology — 2023. — C. 1–5. — DOI: 10.1109/ITNT57377.2023.10139122.

22. Kozlov, D. Application of Transformer for Encoding States in Reinforcement Learning / D. Kozlov // Автометрия — 2024. — №5 — C. 60–68. — DOI: 10.15372/AUT20240500

23. Wörgötter, F. Reinforcement learning / F. Wörgötter, B. Porr // Scholarpedia. — 2008. — T. 3. — C. 1448. — DOI: 10.4249/scholarpedia.1448.

24. Szepesvári, C. Algorithms for Reinforcement Learning: T. 4 / C. Szepesvári2010.

25. Bellman, R. On the Theory of Dynamic Programming / R. Bellman // Proceedings of the National Academy of Sciences of the United States of America. — 1952. — T. 38(8). — C. 716–719.

26. Watkins, C. Technical Note: Q-Learning / C. Watkins, P. Dayan // Machine Learning. — 1992. — T. 8. — C. 279–292. — DOI: 10.1007/BF00992698.

27. McCulloch, W.S. A Logical Calculus of the Ideas Immanent in Nervous Activity / W.S. McCulloch, W. Pitts // The Bulletin of Mathematical Biophysics. — 1943. — T. 5(4). — C. 115–133. — DOI: 10.1007/bf02478259.

28. Krizhevsky, A. ImageNet Classification with Deep Convolutional Neural Networks / A. Krizhevsky, I. Sutskever, G.E. Hinton // Advances in Neural Information Processing Systems, T. 25 — 2012.

29. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain / F. Rosenblatt // Psychological Review. — 1958. — T. 65(6). — C. 386–408. — DOI: 10.1037/h0042519.

30. Wang, Z. Dueling network architectures for deep reinforcement learning / Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, N. De Freitas // Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48: ICML'16. — 2016. — C. 1995–2003.

31. Mnih, V. Human-level control through deep reinforcement learning / V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis // Nature. — 2015. — Vol. 518(7540). — P. 529–533. — DOI: 10.1038/nature14236.

32. End-to-End Training of Deep Visuomotor Policies / S. Levine, C. Finn, T. Darrell, P. Abbeel — 2016. — DOI: 10.48550/arXiv.1504.00702.

33. Playing Atari with Deep Reinforcement Learning / V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller — 2013. — DOI: 10.48550/arXiv.1312.5602.

34. Dota 2 with Large Scale Deep Reinforcement Learning / OpenAI, C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H.P. d O. Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, S. Zhang — 2019. — DOI: 10.48550/arXiv.1912.06680.

35. Vinyals, O. Grandmaster level in StarCraft II using multi-agent reinforcement learning / O. Vinyals, I. Babuschkin, W.M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D.H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J.P. Agapiou, M. Jaderberg, A.S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T.L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, D. Silver // Nature. — 2019. — Vol. 575(7782). — P. 350–354. — DOI: 10.1038/s41586-019-1724-z.

36. Silver, D. Mastering the game of Go with deep neural networks and tree search / D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis // Nature. — 2016. — Vol. 529(7587). — P. 484–489. — DOI: 10.1038/nature16961.

37. Proximal Policy Optimization Algorithms / J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov — 2017. — DOI: 10.48550/arXiv.1707.06347.

38. Schulman, J. Trust Region Policy Optimization / J. Schulman, S. Levine, P. Abbeel, M. Jordan, P. Moritz // Proceedings of the 32nd International Conference on Machine Learning — 2015. — P. 1889–1897.

39. Lillicrap, T. Continuous control with deep reinforcement learning / T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra // CoRR. — 2015.

40. 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 // RL in Games Workshop AAAI 2022. — 2022.

41. Howard, R.A. Dynamic Programming and Markov Processes / R.A. Howard. — Cambridge, Mass: Mit Pr, 1960. — 136 p.

42. Bellman, R. A Markovian Decision Process / R. Bellman // Journal of Mathematics and Mechanics. — 1957. — T. 6(5). — C. 679–684.

43. Reinforcement Learning: A Survey / L.P. Kaelbling, M.L. Littman, A.W. Moore — 1996. — DOI: 10.48550/arXiv.cs/9605103.

44. Burnetas, A.N. Optimal Adaptive Policies for Markov Decision Processes / A.N. Burnetas, M.N. Katehakis // Mathematics of Operations Research. — 1997. — T. 22(1). — C. 222–255.

45. Auer, P. Finite-time Analysis of the Multiarmed Bandit Problem / P. Auer, N. Cesa-Bianchi, P. Fischer // Machine Learning. — 2002. — Vol. 47(2). — P. 235–256. — DOI: 10.1023/A:1013689704352.

46. Sutton, R.S. Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming / R.S. Sutton // Machine Learning Proceedings 1990 — 1990. — C. 216–224. — DOI: 10.1016/B978-1-55860-141-3.50030-4.

47. Sutton, R.S. Policy Gradient Methods for Reinforcement Learning with Function Approximation / R.S. Sutton, D. McAllester, S. Singh, Y. Mansour // Advances in Neural Information Processing Systems, T. 12 — 1999.

48. Haarnoja, T. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor / T. Haarnoja, A. Zhou, P. Abbeel, S. Levine // Proceedings of the 35th International Conference on Machine Learning — 2018. — P. 1861–1870.

49. Chen, X. Randomized Ensembled Double Q-Learning: Learning Fast Without a Model / X. Chen, C. Wang, Z. Zhou, K.W. Ross — 2020.

50. Arulkumaran, K. A Brief Survey of Deep Reinforcement Learning / K. Arulkumaran, M.P. Deisenroth, M. Brundage, A.A. Bharath // IEEE Signal Processing Magazine. — 2017. — T. 34(6). — C. 26–38. — DOI: 10.1109/MSP.2017.2743240.

51. Guliyev, Z. Reinforcement Learning Based Robot Control / Z. Guliyev, A. Parsayan // 2022 IEEE 16th International Conference on Application of Information and Communication Technologies (AICT) — 2022. — C. 1–6. — DOI: 10.1109/AICT55583.2022.10013595.

52. Kormushev, P. Reinforcement Learning in Robotics: Applications and Real-World Challenges / P. Kormushev, S. Calinon, D.G. Caldwell // Robotics. — 2013. — Vol. 2(3). — P. 122–148. — DOI: 10.3390/robotics2030122.

53. Benchmarking Reinforcement Learning Algorithms on Real-World Robots / A.R. Mahmood, D. Korenkevych, G. Vasan, W. Ma, J. Bergstra — 2018. — DOI: 10.48550/arXiv.1809.07731.

54. Nguyen, H. Review of Deep Reinforcement Learning for Robot Manipulation / H. Nguyen, H. La // 2019 Third IEEE International Conference on Robotic Computing (IRC) — 2019. — C. 590–595. — DOI: 10.1109/IRC.2019.00120.

55. Littman, M.L. Reinforcement learning improves behaviour from evaluative feedback / M.L. Littman // Nature. — 2015. — Vol. 521(7553). — P. 445–451. — DOI: 10.1038/nature14540.

56. Martín-Guerrero, J.D. Use of Reinforcement Learning in Two Real Applications / J.D. Martín-Guerrero, E. Soria-Olivas, M. Martínez-Sober, A.J. Serrrano-López, R. Magdalena-Benedito, J. Gómez-Sanchis // Recent Advances in Reinforcement Learning — 2008. — P. 191–204. — DOI: 10.1007/978-3-540-89722-4_15.

57. Reinforcement Learning Applications / Y. Li — 2019. — DOI: 10.48550/arXiv.1908.06973.

58. Lee, M.-F.R. Mobile Robot Navigation Using Deep Reinforcement Learning / M.-F.R. Lee, S.H. Yusuf // Processes. — 2022. — Vol. 10(12). — P. 2748. — DOI: 10.3390/pr10122748.

59. Singh, R. A Review of Deep Reinforcement Learning Algorithms for Mobile Robot Path Planning / R. Singh, J. Ren, X. Lin // Vehicles. — 2023. — Vol. 5(4). — P. 1423–1451. — DOI: 10.3390/vehicles5040078.

60. Osiński, B. Simulation-Based Reinforcement Learning for Real-World Autonomous Driving / B. Osiński, A. Jakubowski, P. Zięcina, P. Miłoś, C. Galias, S. Homoceanu, H. Michalewski // 2020 IEEE International Conference on Robotics and Automation (ICRA) — 2020. — C. 6411–6418. — DOI: 10.1109/ICRA40945.2020.9196730.

61. Silver, D. Reward is enough / D. Silver, S. Singh, D. Precup, R.S. Sutton // Artificial Intelligence. — 2021. — T. 299. — C. 103535. — DOI: 10.1016/j.artint.2021.103535.

62. Back, P. Real-World Reinforcement Learning: Observations from Two Successful Cases / P. Back // BLED 2021 Proceedings. — 2021.

63. Sharma, A. Emergent Real-World Robotic Skills via Unsupervised Off-Policy Reinforcement Learning / A. Sharma, M. Ahn, S. Levine, V. Kumar, K. Hausman, S. Gu — 2020. — DOI: 10.15607/RSS.2020.XVI.053.

64. Paszke, A. Automatic differentiation in PyTorch / A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer — 2017.

65. TorchRL: A data-driven decision-making library for PyTorch / A. Bou, M. Bettini, S. Dittert, V. Kumar, S. Sodhani, X. Yang, G.D. Fabritiis, V. Moens — 2023.

66. Gymnasium / M. Towers, J.K. Terry, A. Kwiatkowski, J.U. Balis, G. de Cola, T. Deleu, M. Goulão, A. Kallinteris, A. KG, M. Krimmel, R. Perez-Vicente, A. Pierré, S. Schulhoff, J.J. Tai, A.T.J. Shen, O.G. Younis — 2023. — DOI: 10.5281/zenodo.8127026.

67. OpenAI Gym / G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, W. Zaremba — 2016. — DOI: 10.48550/arXiv.1606.01540.

68. Todorov, E. MuJoCo: A physics engine for model-based control / E. Todorov, T. Erez, Y. Tassa — 2012. — C. 5026–5033. — DOI: 10.1109/IROS.2012.6386109.

69. 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 preprint arXiv:1809.02627. — 2020.

70. Robotic Operating System / Stanford Artificial Intelligence Laboratory et al.

71. Macenski, S. Impact of ROS 2 Node Composition in Robotic Systems / S. Macenski, A. Soragna, M. Carroll, Z. Ge // IEEE Robotics and Autonomous Letters (RA-L). — 2023.

72. Koenig, N. Design and use paradigms for Gazebo, an open-source multi-robot simulator / N. Koenig, A. Howard // 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), T. 3 — 2004. — C. 2149–2154 т.3. — DOI: 10.1109/IROS.2004.1389727.

73. Open Dynamics Engine [Электронный ресурс] — Режим доступа: https://www.ode.org/ (дата обращения: 28.08.2024).

74. Coumans, E. Bullet physics simulation / E. Coumans // ACM SIGGRAPH 2015 Courses. — 2015. — P. 1. — DOI: 10.1145/2776880.2792704.

75. Lee, J. DART: Dynamic Animation and Robotics Toolkit / J. Lee, M.X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S.S. Srinivasa, M. Stilman, C.K. Liu // The Journal of Open Source Software. — 2018. — T. 3(22). — C. 500. — DOI: 10.21105/joss.00500.

76. Zamora, I. Extending the OpenAI Gym for robotics: a toolkit for reinforcement learning using ROS and Gazebo / I. Zamora, N.G. Lopez, V. Vilches, A. Cordero // ArXiv. — 2016.

77. Barto, A.G. Neuronlike adaptive elements that can solve difficult learning control problems / A.G. Barto, R.S. Sutton, C.W. Anderson // IEEE Transactions on Systems, Man, and Cybernetics. — 1983. — T. SMC-13(5). — C. 834–846. — DOI: 10.1109/TSMC.1983.6313077.

78. Towers, M. Gymnasium: A Standard Interface for Reinforcement Learning Environments / M. Towers, A. Kwiatkowski, J. Terry, J.U. Balis, G. De Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, others // arXiv preprint arXiv:2407.17032. — 2024.

79. The openai gym CartPole-v0 problem [Electronic resource] // Gist. — Режим доступа: https://gist.github.com/onimaru/ea2f88c2156a77ce7262fb5e2f112fe0 (дата обращения: 28.08.2024).

80. DQN, Cartpole-v0 [Electronic resource] // Gist. — Режим доступа: https://gist.github.com/n1try/2a6722407117e4d668921fce53845432#file-dqn_cartpole-py (дата обращения: 28.08.2024).

81. Q-Network, CartPole [Electronic resource] / 262588213843476 // Gist. — Режим доступа: https://gist.github.com/mbalunovic/fb7392e2c09b2c3895a354c3ad36497e (дата обращения: 28.08.2024).

82. Lonza, A. Reinforcement Learning Algorithms with Python / A. LonzaPackt Publishing, 2019. — 366 p.

83. Wawrzyński, P. A Cat-Like Robot Real-Time Learning to Run / P. Wawrzyński // Adaptive and Natural Computing Algorithms: Lecture Notes in Computer Science. — 2009. — P. 380–390. — DOI: 10.1007/978-3-642-04921-7_39.

84. Tassa, Y. Synthesis and stabilization of complex behaviors through online trajectory optimization / Y. Tassa, T. Erez, E. Todorov // 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems — 2012. — C. 4906–4913. — DOI: 10.1109/IROS.2012.6386025.

85. Durrant-Whyte, H.Infinite-Horizon Model Predictive Control for Periodic Tasks with Contacts / H. Durrant-Whyte, N. Roy, P. Abbeel // Robotics: Science and Systems VII — 2012. — C. 73–80.

86. Vaswani, A. Attention is All you Need / A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. ukasz Kaiser, I. Polosukhin // Advances in Neural Information Processing Systems, T. 30 — 2017.

87. Pendrith, M.Reinforcement learning for real-world control applications / M. Pendrith, M. Ryan // Advances in Artifical Intelligence — 1996. — P. 257–270. — DOI: 10.1007/3-540-61291-2_57.

88. Dulac-Arnold, G. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis / G. Dulac-Arnold, N. Levine, D.J. Mankowitz, J.

Li, C. Paduraru, S. Gowal, T. Hester // Machine Learning. — 2021. — Vol. 110(9). — P. 2419–2468. — DOI: 10.1007/s10994-021-05961-4.

89. An empirical investigation of the challenges of real-world reinforcement learning / G. Dulac-Arnold, N. Levine, D.J. Mankowitz, J. Li, C. Paduraru, S. Gowal, T. Hester — 2021. — DOI: 10.48550/arXiv.2003.11881.

90. Chebotar, Y. Q-Transformer: Scalable Offline Reinforcement Learning via Autoregressive Q-Functions / Y. Chebotar, Q. Vuong, K. Hausman, F. Xia, Y. Lu, A. Irpan, A. Kumar, T. Yu, A. Herzog, K. Pertsch, K. Gopalakrishnan, J. Ibarz, O. Nachum, S.A. Sontakke, G. Salazar, H.T. Tran, J. Peralta, C. Tan, D. Manjunath, J. Singh, B. Zitkovich, T. Jackson, K. Rao, C. Finn, S. Levine — 2023.

91. Transformer Based Reinforcement Learning For Games / U. Upadhyay, N. Shah, S. Ravikanti, M. Medhe — 2019. — DOI: 10.48550/arXiv.1912.03918.

92. Сидельников, В. Теория кодирования / В. Сидельников

93. Adam: A Method for Stochastic Optimization / D.P. Kingma, J. Ba — 2017.

94. LeCun, Y. Deep learning / Y. LeCun, Y. Bengio, G. Hinton // Nature. — 2015. — Vol. 521(7553). — P. 436–444. — DOI: 10.1038/nature14539.