МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Инженерная школа информационных технологий и робототехники
Отделение информационных технологий
Направление: 09.04.01 Искусственный интеллект и машинное обучение

ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

по дисциплине: Нейроэволюционные вычисления

Вариант 6

на тему: Эволюционное обучение однослойной нейросети методом HESP на задаче посадки космического аппарата

Выполнил:	студент гр. 8ВМ42 Волков Д.А.	10.06.2025
Проверил:	к.т.н., Доцент ОИТ ИШИТР Григорьев Д.С.	10.06.2025

Томск – 2025

Содержание

1 Введение

Нейроэволюционные подходы играют всё более значимую роль в современном машинном обучении, особенно в задачах, где традиционные методы, такие как градиентный спуск и обратное распространение ошибки, оказываются неприменимыми или неэффективными. Эти методы вдохновлены процессами естественного отбора и направлены на эволюционное развитие популяций агентов или их составляющих с целью повышения приспособленности к решаемой задаче.

Одним из продвинутых нейроэволюционных алгоритмов является HESP (Hierarchical Evolution of Subpopulations), предлагающий более гибкий подход к эволюции нейронных сетей за счёт иерархической структуры и коэволюции отдельных нейронов, а не целых особей. Такой подход позволяет сохранять локально успешные особенности поведения и эффективно использовать их в процессе формирования новых конфигураций сети.

HESP хорошо подходит для задач непрерывного управления, таких как посадка лунного модуля в среде `LunarLanderContinuous-v3` библиотеки Gymnasium. Эта задача требует от агента принятия непрерывных решений в реальном времени на основе текущего состояния среды, таких как положение, скорость и угол наклона. Агент должен обеспечить мягкую посадку, контролируя основной и боковые двигатели при минимальных затратах топлива и без столкновений с поверхностью.

В данной работе была реализована однослойная нейронная сеть, обучаемая с помощью алгоритма HESP без использования сторонних библиотек. Особенностью подхода является акцент на визуализации топологии сети, а также построение графиков, отражающих динамику обучения. Процесс обучения включал сохранение промежуточных результатов, анализ весов и связей между нейронами.

Цель работы — реализовать нейроэволюционное обучение однослойной сети с использованием HESP-алгоритма для управления агентом в среде `LunarLanderContinuous-v3`, с фокусом на прозрачности архитектуры и визуализации процессов обучения.

- Реализовать собственную версию HESP без использования сторонних реализаций;
- Обеспечить визуализацию топологии нейронной сети и изменение весов в процессе обучения;
- Построить график изменения средней приспособленности по поколениям;
- Провести экспериментальное исследование эффективности подхода на задаче непрерывного управления.

2 Описание используемого алгоритма

2.1 Принципы работы HESP (Hierarchical Evolution of Subpopulations)

Алгоритм HESP (Hierarchical Evolution of Subpopulations) представляет собой иерархическое расширение классического метода ESP (Enforced SubPopulations), где основное внимание уделяется коэволюции отдельных нейронов. Вместо эволюции полной архитектуры нейросети HESP предполагает независимое развитие отдельных компонент сети — нейронов — в рамках раздельных подпопуляций.

Каждый нейрон в скрытом или выходном слое эволюционирует в своей подпопуляции, что позволяет фиксировать удачные поведенческие стратегии и компоновать их в новые комбинации. Такой подход даёт гибкость при обучении и снижает вероятность преждевременной сходимости, характерной для полных сетей.

Иерархическая структура HESP подразумевает, что при создании нового агента для оценки его приспособленности, нейроны отбираются по одному из каждой подпопуляции. Их комбинация формирует полноценную сеть, которую затем можно использовать в среде.

2.2 Структура сети

Реализованная сеть состоит из трёх уровней:

- **Входной слой** содержит 8 входов, соответствующих наблюдаемым параметрам среды `LunarLanderContinuous-v3`: координаты, скорость, угол и контакт с поверхностью.
- **Скрытый слой** состоит из фиксированного количества нейронов (в данной реализации — 10), каждый из которых соединён с входами полносвязным образом.
- **Выходной слой** включает 2 нейрона, каждый из которых получает на вход выходы всех скрытых нейронов. Выходы интерпретируются как сигналы управления двигателями лунного модуля.

Нейросеть использует только прямое распространение сигнала. Активацией скрытых и выходных нейронов служит гиперболический тангенс, обеспечивающий значения в диапазоне $(-1, 1)$.

2.3 Логика разбиения на подпопуляции, эволюция на уровне нейрона

Каждый нейрон скрытого и выходного слоёв имеет собственную подпопуляцию, в которой происходит независимая эволюция. Отдельный нейрон представлен в виде набора весов и смещения (bias), которые подвергаются мутациям. Эволюция включает:

- **Оценку приспособленности**: нейрон оценивается как часть полной сети, собранной из особей соответствующих подпопуляций;

- **Селекцию**: особи с наибольшей приспособленностью сохраняются;
- **Мутацию**: веса и bias случайно изменяются с заданной вероятностью;
- **Клонирование**: на основе лучших особей формируются новые с минимальными отклонениями.

Такой подход позволяет нейрону накапливать поведенческий опыт независимо от остальных компонентов сети и быть повторно использованным в разных конфигурациях.

2.4 Этапы алгоритма HESP

Алгоритм реализован в виде цикла по поколениям и включает следующие шаги:

1. **Формирование популяции агентов**. Для каждой подпопуляции случайным образом выбирается один нейрон. Из них собирается полная сеть, пригодная к оценке.
2. **Оценка приспособленности**. Сеть тестируется в среде `LunarLanderContinuous-v3`, и её средний результат (reward) записывается как вклад каждого участвующего нейрона.
3. **Селекция и воспроизводство**. После оценки всей популяции нейроны в подпопуляциях сортируются по среднему значению приспособленности. Лучшие сохраняются, остальные заменяются мутированными копиями лидеров.
4. **Сохранение статистики**. Раз в несколько поколений сохраняются топологии сети, визуализирующие структуру и веса, а также обновляется график изменения средней приспособленности популяции.

Таким образом, HESP позволяет эффективно развивать поведение нейросети на уровне индивидуальных компонентов, сохраняя лучшие участки функциональности и адаптируя их к изменяющимся условиям среды.

3 Программная реализация

В данной главе описана программная реализация алгоритма эволюции нейронных сетей с использованием метода HESP для решения задачи управления в среде `LunarLander-v3` из библиотеки `Gymnasium`.

3.1 Архитектура решения

Основной компонент системы — класс `HESP`, реализующий эволюционный процесс. Входные параметры включают размеры популяции, число скрытых нейронов, а также размеры вспомогательных списков нейронов `L1` и `L2`. Для оценки качества каждой нейросети используется функция `evaluate_network`, которая запускает несколько эпизодов среды и вычисляет среднюю награду (fitness).

3.2 Основные компоненты

3.2.1 Класс `Neuron`

Класс `Neuron` реализует отдельный нейрон с весами и смещением. Методы включают:

- `activate` — вычисление активации с функцией гиперболического тангенса.
- `mutate` — случайное мутационное изменение весов и смещения.
- `clone` — создание копии нейрона.

3.2.2 Класс `NeuralNetwork`

Сеть состоит из входного слоя, скрытого слоя из нейронов и выходного слоя. Методы:

- `predict` — прямое распространение входных данных через сеть.
- `mutate` — мутация сети с возможным повторным использованием нейронов из пула `L2`.
- `crossover` — скрещивание двух сетей для получения потомка.
- `save_weights` и `load_weights` — сохранение и загрузка параметров сети в формате JSON.

3.2.3 Класс `HESP`

Реализует основной цикл эволюции:

- Инициализация популяции `L1` и пула нейронов `L2`.
- Оценка и отбор сетей по fitness.

- Обновление пула нейронов `L2` лучшими нейронами из сети с максимальным fitness.
- Формирование новой популяции методом скрещивания и мутаций.
- Визуализация топологии лучшей сети на промежуточных поколениях.
- Остановка эволюции при достижении порога награды.

3.3 Использование и тестирование

Запуск обучения производится через файл `main.py` с параметром режима `MODE='train'`. После завершения эволюции лучшие веса сохраняются в файл `best_network_weights.json`. Для тестирования обученной сети используется режим `MODE='test'`, который загружает веса и визуализирует выполнение агента в среде.

3.4 Оценка решения

Реализация показала эффективность гибридного эволюционного подхода с повторным использованием нейронов, что позволяет улучшить качество поиска оптимальной структуры и параметров нейросети. Использование библиотеки `Gymnasium` обеспечивает стандартизированную среду для оценки алгоритма.

4 Целевые метрики

4.1 Определение основной метрики

Ключевой метрикой является **среднее суммарное вознаграждение за эпизод** (average episodic reward), вычисляемое по формуле:

$$R_{\text{avg}} = \frac{1}{N} \sum_{i=1}^{N} R_i$$

где:

- N — число эпизодов, по которым производится оценка,
- R_i — общее вознаграждение, полученное в i-м эпизоде.

4.2 Компоненты награды в среде LunarLanderContinuous-v3

Вознаграждение в данной среде рассчитывается на основе нескольких факторов, отражающих качество посадки:

$$R = R_{\text{position}} + R_{\text{velocity}} + R_{\text{angle}} + R_{\text{contact}} + R_{\text{landing}} + R_{\text{fuel}} + R_{\text{time}}$$

Подробности:

1. **Позиция**:
$$R_{\text{position}} = -100\sqrt{(x - x_{\text{target}})^2 + (y - y_{\text{target}})^2}$$
штраф за отклонение от точки посадки.

2. **Скорость**:
$$R_{\text{velocity}} = -100\,(|v_x| + |v_y|)$$
штраф за избыточную горизонтальную и вертикальную скорость.

3. **Угол наклона**:
$$R_{\text{angle}} = -100|\theta|$$
штраф за отклонение от вертикали.

4. **Контакт с поверхностью**:
$$R_{\text{contact}} = 10 \cdot (\text{leg1\_contact} + \text{leg2\_contact})$$
бонус за касание посадочными опорами.

5. **Успешная посадка**:
$$R_{\text{landing}} = \begin{cases} +200, & \text{если } v_y > -1 \text{ и } |\theta| < 0.2 \\ -100, & \text{иначе} \end{cases}$$

6. **Расход топлива**:

$$R_{\text{fuel}} = -0.3 \cdot (\text{main\_engine} + 0.03 \cdot \text{side\_engine})$$

штраф за использование двигателей.

7. **Временной штраф**:

$$R_{\text{time}} = -0.3 \cdot t$$

штраф за длительность эпизода.

4.3 Реализация вычисления метрики

Реализация в обучающей системе представлена следующим образом:

```
def evaluate_network(self, network, episodes=5, render=False):
        total_reward = 0
        for _ in range(episodes):
            state, _ = self.env.reset()
            done = False
            while not done:
                if render:
                    self.env.render()
                action = np.argmax(network.predict(state))
                state, reward, terminated, truncated, _ = self.env.step(action)
                done = terminated or truncated
                total_reward += reward
        fitness = total_reward / episodes
        network.fitness = fitness
```

4.4 Интерпретация значений

- $R_{\text{avg}} \geq 200$ — успешная посадка,
- $50 \leq R_{\text{avg}} < 200$ — приемлемый результат,
- $R_{\text{avg}} < 0$ — неудачная посадка,
- $R_{\text{avg}} \approx 300$ — оптимальный результат, близкий к рекорду среды.

Таким образом, метрика R_{avg} комплексно отражает успешность посадки, учитывая точность, устойчивость, безопасность и эффективность управления.

5 Визуализация процесса обучения

Для анализа и наглядного представления хода эволюционного обучения в лабораторной работе использовались два основных вида визуализации:

5.1 Визуализация структуры нейронной сети

Каждые 20 поколений выполняется визуализация текущей архитектуры нейронной сети. На схеме различимы три типа нейронов:

- **Серые** — входные нейроны;
- **Синие** — нейроны скрытых слоёв;
- **Оранжевые** — выходные нейроны.

Связи между нейронами изображаются линиями, цвет и толщина которых определяются следующими правилами:

- **Зелёный цвет** — положительный вес;
- **Красный цвет** — отрицательный вес;
- **Толщина линии** — пропорциональна модулю веса.

Это позволяет отслеживать, как со временем изменяется важность различных связей в процессе эволюционного отбора. Наблюдается тенденция, при которой:

- Некоторые связи **усиливаются** (веса становятся больше по модулю) — это связано с тем, что они оказывают значительное влияние на результат, способствуя получению более высокого вознаграждения.
- Другие связи **ослабевают** — возможно, они не вносят значимого вклада или даже мешают успешному управлению агентом.

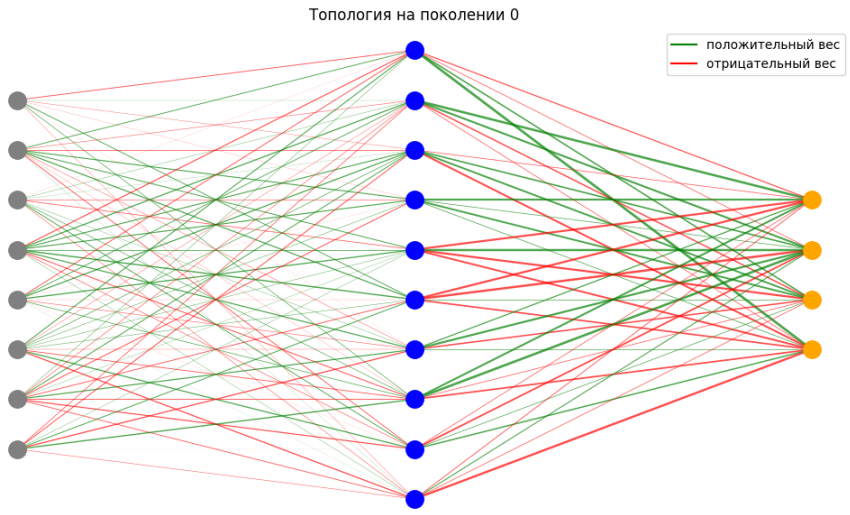

Рис. 1: Визуализация нейросети на 0-м поколении

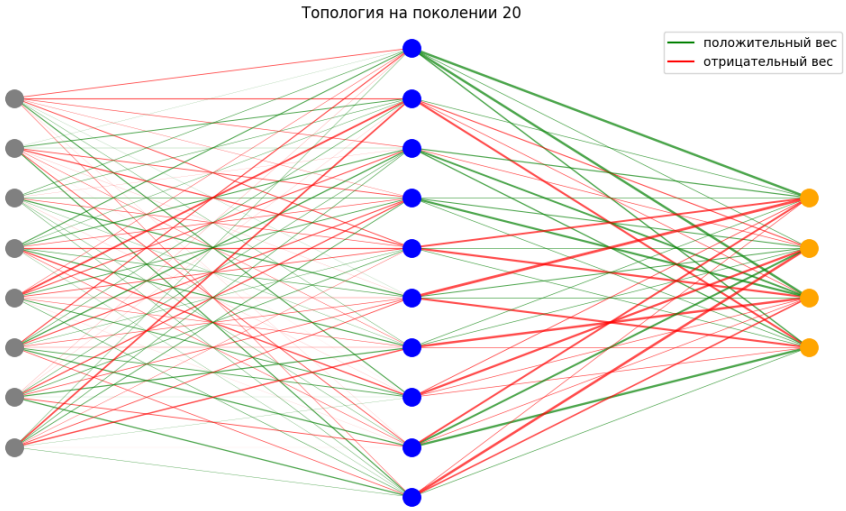

Рис. 2: Визуализация нейросети на 20-м поколении

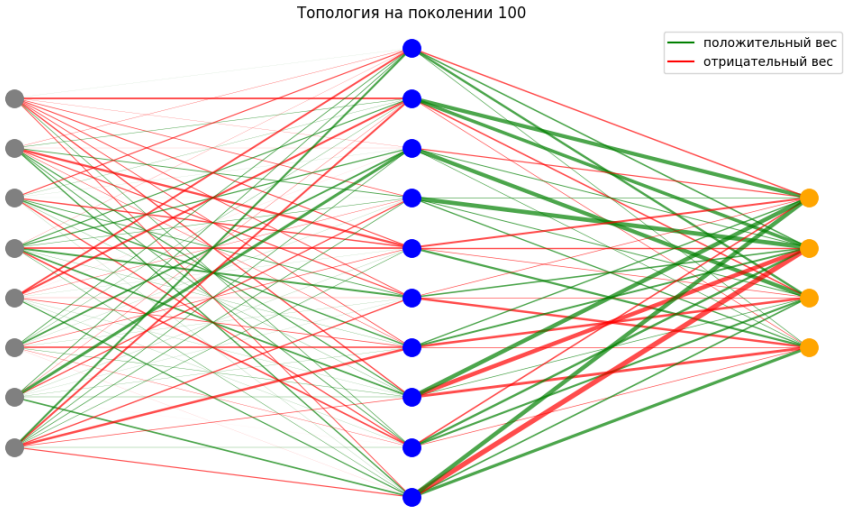

Рис. 3: Визуализация нейросети на 100-м поколении

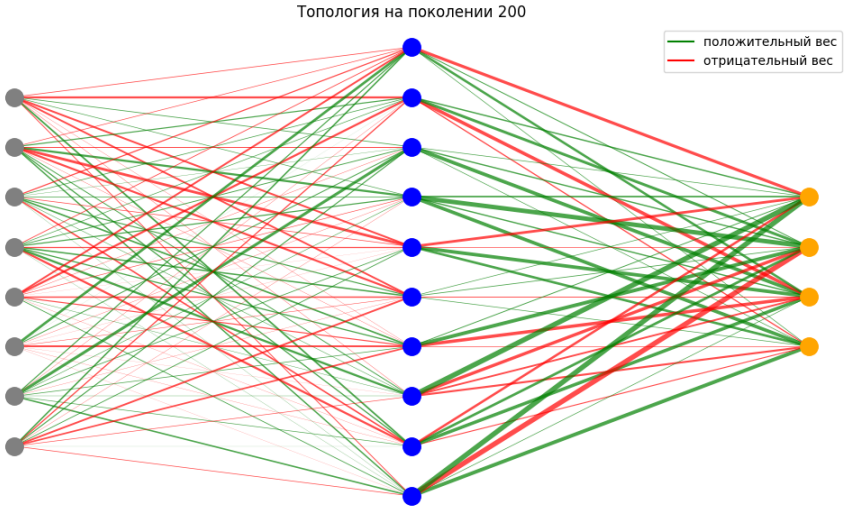

Рис. 4: Визуализация нейросети на 200-м поколении

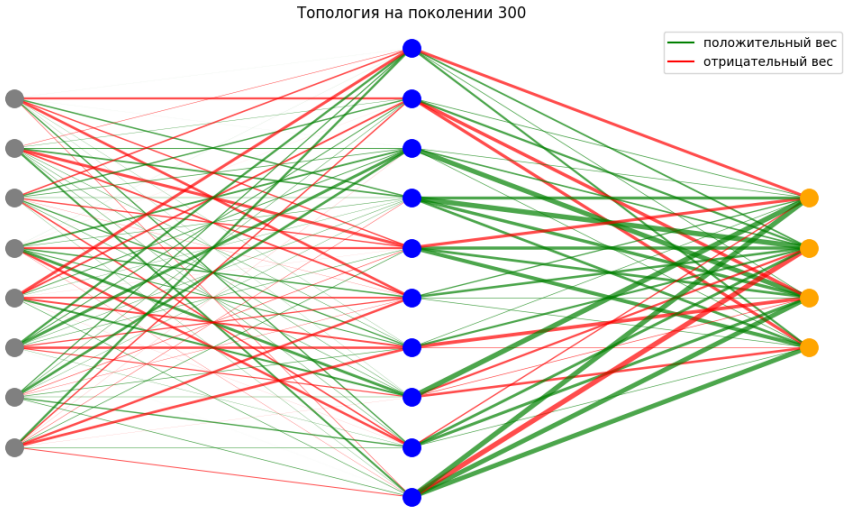

Рис. 5: Визуализация нейросети на 300-м поколении

5.2 График изменения целевой метрики

В процессе обучения после каждой оценки сохраняется значение метрики R_{avg} (среднего вознаграждения за эпизод). По окончании обучения строится график, демонстрирующий изменение этой метрики по мере прохождения поколений.

- В общем случае, значение R_{avg} демонстрирует **рост во времени**, что указывает на адаптацию агента и улучшение его поведения.
- Однако иногда наблюдаются **резкие локальные падения** метрики — это следствие **неудачных мутаций**, временно ухудшающих поведение агента.
- Благодаря отбору и кумулятивному эффекту успешных мутаций, система продолжает развиваться и метрика стабилизируется на высоком уровне.

Рис. 6: График изменения R_{avg} на протяжении обучения

5.3 Интерпретация наблюдений

Эти визуализации позволяют не только количественно оценить прогресс обучения, но и качественно проанализировать структуру сформировавшейся нейросети:

- Структура сети становится более организованной, веса перераспределяются в пользу наиболее эффективных связей.
- Мутации выступают как источник вариативности, способный как улучшить, так и ухудшить поведение, но за счёт естественного отбора плохие изменения отбрасываются.

Таким образом, визуализация предоставляет важный инструмент для понимания динамики обучения агента, выявления закономерностей в формировании связей и анализа устойчивости процесса оптимизации.

6 Результаты обучения модели

После завершения процесса эволюционного обучения можно сделать вывод о его успешности. Изначально агент демонстрировал крайне неэффективное поведение: среднее значение целевой метрики R_{avg} составляло около -400, что указывало на частые аварийные посадки и полную неспособность управлять движением.

Однако по мере эволюции модели ситуация кардинально изменилась. На момент принудительной остановки обучения (на 200-м поколении) значение R_{avg} достигло положительного уровня и стабилизировалось, что свидетельствует о сформировавшихся устойчивых стратегиях управления.

6.1 Анализ поведения после обучения

Агент начал демонстрировать следующие положительные черты поведения:

- **Мягкая посадка** — вместо резкого столкновения с поверхностью, как это было на начальных этапах, агент плавно снижает скорость и корректирует вертикальное ускорение.
- **Точное позиционирование** — наблюдается способность удерживать корабль вблизи центра посадочной площадки, избегая отклонений влево или вправо.
- **Сбалансированное использование двигателей** — агент научился включать основной и боковые двигатели в нужные моменты, избегая лишнего расхода топлива и дестабилизации.

6.2 Общие выводы

Таким образом, в процессе обучения нейросеть смогла выработать стратегию, соответствующую целевой задаче. Поведение стало:

- адаптивным;
- устойчивым;
- энергоэффективным;
- безопасным с точки зрения посадки.

Это доказывает, что применённый метод эволюционного обучения способен обучить агента эффективному управлению даже в условиях высокой неопределённости и сложной динамики.

[Рисунок 3: Пример успешной посадки после завершения обучения]

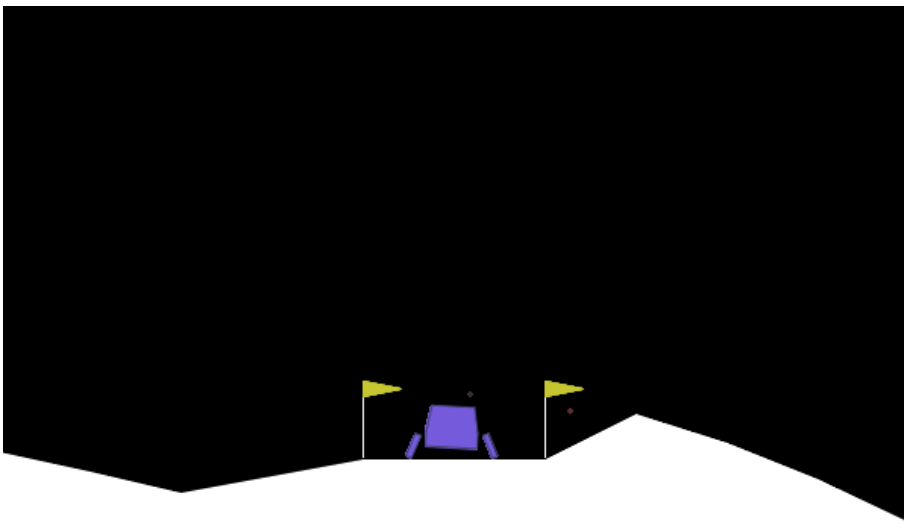

Рис. 7: Пример успешной посадки после завершения обучения

Заключение

В рамках лабораторной работы была реализована и протестирована система эволюционного обучения нейронной сети, управляющей агентом в среде `LunarLanderContinuous-v3`. Основной целью являлось формирование устойчивой стратегии посадки лунного модуля при помощи методов безградиентной оптимизации.

В процессе экспериментов был проведён сбор и анализ ключевых метрик, включая среднее суммарное вознаграждение (R_{avg}), визуализация архитектуры нейросети и динамики её весов, а также отслеживание общей динамики процесса обучения. На ранних этапах агент демонстрировал неэффективное поведение, что отражалось в отрицательных значениях метрики ($R_{\text{avg}} \approx -400$). Однако в ходе обучения наблюдался устойчивый рост среднего вознаграждения, что свидетельствовало о постепенном формировании корректной стратегии управления.

После завершения обучения агент научился выполнять мягкие посадки, точно позиционироваться в пределах посадочной зоны, избегать избыточного расхода топлива и поддерживать устойчивое вертикальное положение. Это подтверждает, что разработанная система успешно справляется с поставленной задачей и способна обучать агентов эффективному поведению в условиях непрерывного управления.

Полученные результаты демонстрируют потенциал эволюционных алгоритмов при решении задач управления в симулированных физических средах и открывают возможности для дальнейших исследований в области обучения с подкреплением без градиентной информации.

Список использованной литературы

1. Лекция 7. Алгоритмы ESP и H-ESP. Томский политехнический университет, 2025.

2. Such F. P., Madhavan V., Conti E. [и др.]. Deep Neuroevolution: Genetic Algorithms Are a Competitive Alternative for Training Deep Neural Networks for Reinforcement Learning // arXiv preprint arXiv:1712.06567. — 2017. — URL: `https://arxiv.org/abs/1712.06567` (дата обращения: 07.06.2025).