Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА «ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИЙ»

Тут полям очень грустно

Лабораторная работа № 12 по дисциплине «Основы систем искусственного интеллекта»

Тема Подкрепление

Студент Куличенков А. П.

Группа ИУ7-36Б

Преподаватель Строганов Ю. В.

Москва, 2025

Содержание

ВВЕДЕНИЕ

Целью данной лабораторной работы является разработка алгоритма управления роботизированной кистью Adroit для выполнения задачи Adroit Pen с использованием методов обучения с подкреплением. Объектом исследования выступает виртуальная модель роботизированной кисти Adroit Hand в среде симуляции Gymnasium-robotics. Модель кисти включает в себя лучезапястный сустав, суставы пальцев и 24 степени свободы, управляемые вектором действий, содержащим значения углов поворота в каждом суставе. Задача Adroit Pen предполагает манипулирование ручкой для достижения заданной целевой позиции.

Работа включает реализацию алгоритма обучения с подкреплением, обучение агента взаимодействию со средой симуляции Gymnasium-robotics для управления кистью Adroit Hand, анализ эффективности разработанного алгоритма и оценку качества выполнения задачи Adroit Pen.

```
Задачи упустили:

Целью работы является....
Для достижения поставленной цели необходимо выполнить
следующие задачи:
1. ....
2. ....

....
```

ССЫЛКИ НА ИСТОЧНИКИ

1 Аналитическая часть

Нет раздела про обучение с подкреплением, лучше уж сначала про него написать, потом раскрывать детальнее существующие алгоритмы

Задача управления роботизированной рукой Adroit Hand в среде Gymnasium-robotics представляет собой сложную задачу, характеризующуюся высокой размерностью пространства действий (24 степени свободы), нелинейной динамикой модели кисти, взаимодействием с объектами и ограничениями на диапазоны движения в суставах. Успешное достижение целевого положения объекта манипуляции (ручки) требует высокой точности управления.

1.1 Теоретический анализ методов и подходов к решению

Надо объединять по темам, "теоретический анаоиз" не является объединекним по теме – у вас по сути вся аналитическая часть это аналитический анализ

Для решения задачи управления Adroit Hand применяются методы обучения с подкреплением (Reinforcement Learning, RL). Ниже рассматриваются алгоритмы, применимые к данной задаче.

Нельзя из текста ссылаться на заголовки подразделов – не заставляйте читателя скроллить ниже, чтобы понять, о чем пойдет речь, и затем вернуться читать про PPO

1.1.1 Proximal Policy Optimization (PPO)

Что такое политика в контексте обучения с подкреплением ? Вы не дали определения

Алгоритм Proximal Policy Optimization (PPO) является алгоритмом обучения с подкреплением, основанным на политике, и применяется для обучения агентов в непрерывном пространстве действий. PPO итеративно улучшает стохастическую политику $\pi_\theta(a|s)$, где θ - параметры политики, a - действие, s - состояние. Цель PPO - оптимизировать политику, максимизируя ожидаемое вознаграждение, при этом ограничивая изменения политики на каждом шаге обучения, чтобы обеспечить стабильность обучения. PPO минимизирует функционал:

А что есть E в этой формуле?

Вынесите в отдельную формулу, иначе в распечатанном виде прочитать будет невозможно

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right], \tag{1.1}$$

где $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ - отношение вероятностей действий, $\hat{A}_t$ - оценка advantage-функции, ϵ - параметр клиппирования. Векторизация среды обучения позволяет агенту взаимодействовать с n экземплярами среды параллельно, что ускоряет сбор опыта.

Это не является частью самого алгоритма же

1.1.2 Deep Deterministic Policy Gradient (DDPG)

Дефис вместо тире вышел

Deep Deterministic Policy Gradient (DDPG) - это алгоритм обучения с подкреплением, основанный на критике (actor-critic), который используется для обучения агентов, действующих в непрерывном пространстве действий. DDPG обучает детерминированную политику $\mu(s)$, которая непосредственно определяет действие в данном состоянии, и критическую функцию $Q(s,a)$, которая оценивает ожидаемое вознаграждение для данной пары состояния и действия. DDPG основан на Bellman equation:

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})), \tag{1.2}$$

Минимизируем нерасшифрованный непереведенный текст, тем более что уравнение Беллмана довольно неплохо переводится на русский язык

Разрыв между формулой и пояснением тут не очень хорошо выглядит, но это скорее на будущее

Рассогласование, деепричастный оборот тут не может быть использован

где $r(s_t, a_t)$ - вознаграждение, γ - коэффициент дисконтирования. Целью является максимизация Q функции, подбирая соответствующие действия μ. Особенностью DDPG является использование двух наборов сетей: actor-сеть, определяющая политику, и critic-сеть, оценивающая качество действий, а также наличие target сетей для стабилизации обучения.

Сетей каких, рыболовных?)

1.1.3 Twin Delayed Deep Deterministic Policy Gradient (TD3)

Twin Delayed Deep Deterministic Policy Gradient (TD3) является усовершенствованием алгоритма DDPG, направленным на устранение переоценки Q-функции и повышение стабильности обучения. TD3 использует два критика (две Q-функции) и выбирает минимальное из оценок Q-функций при обновлении политики. Кроме того, TD3 откладывает обновление политики и обновляет ее только через несколько шагов обновления критика. Алгоритм TD3 вносит следующие изменения в процесс обучения:

Никак тут не переформулировать? А то обновление на обновлении

вычисляется / определяется

1) Использует два критика Q_1 и Q_2, где целевое значение получается как минимум из оценок:

$$y = r + \gamma \min(Q_1'(s', \mu(s')), Q_2'(s', \mu(s'))). \tag{1.3}$$

Мы по-русски пишем

2) Применяет policy smoothing для улучшения стабильности обучения и ограничения обновления политики:

$$a' = \text{clip}(\mu(s') + \epsilon, a_{\min}, a_{\max}) \tag{1.4}$$

где ϵ является шумом.

3) Откладывает обновление политики и выполняет его реже, чем обновление критиков, с целью уменьшения влияния ошибок в оценке Q-функции.

1.1.4 Soft Actor-Critic (SAC)

Дефис вместо тире вышел x2

Soft Actor-Critic (SAC) - это алгоритм обучения с подкреплением, основанный на энтропии. SAC использует стохастическую политику и максимизирует не только ожидаемое вознаграждение, но и энтропию политики, что поощряет исследование пространства состояний. SAC обучается одновременно с критиком и актором. Обновление критика выполняется путем минимизации ошибки, где целевым значением является:

$$y = r(s, a) + \gamma \mathbb{E}_{s' \sim p(s'|s,a)}[V(s')], \tag{1.5}$$

а функция V имеет вид:

$$V(s) = \mathbb{E}_{a \sim \pi(a|s)}[Q(s, a) - \alpha \log \pi(a|s)], \tag{1.6}$$

где α - коэффициент энтропии, который регулирует важность энтропии в обучении.

прям важность? или все-таки вес энтропии это?

1.2 Сравнительный анализ методов

Таблица 1.1 — Сравнение методов обучения с подкреплением

Характеристика	**PPO**	**DDPG**	**TD3**	**SAC**
Тип политики	Стох.	Дет.	Дет.	Стох.
Стабильность	Высокая	Средняя	Средняя	Высокая
Исследование	Умеренное	Низкое	Низкое	Высокое
Сходимость	Быстрая	Медленная	Средняя	Медленная
Сложность	Средняя	Низкая	Средняя	Высокая

Таблица 1.1 сравнивает методы обучения с подкреплением. PPO и SAC используют стохастические политики, обеспечивая более стабильное обучение, в отличие от детерминированных DDPG и TD3. SAC лучше исследует пространство состояний, но сходится медленнее. PPO сочетает стабильность и скорость обучения, что подходит для задач управления. DDPG и TD3 проще в реализации, чем PPO и SAC. Выбор PPO обусловлен балансом между стабильностью и эффективностью обучения.

Вывод

Управление Adroit Hand сложно из-за высокой размерности пространства действий и нелинейной динамики. Методы обучения с подкреплением подходят для решения этой задачи. PPO обеспечивает баланс между стабильностью и эффективностью, что делает его подходящим для рассматриваемой задачи.

2 Конструкторская часть

В данном разделе представлено описание процесса разработки алгоритма обучения с подкреплением для управления роботизированной рукой Adroit в задаче Adroit Pen.

2.1 Схема процесса обучения

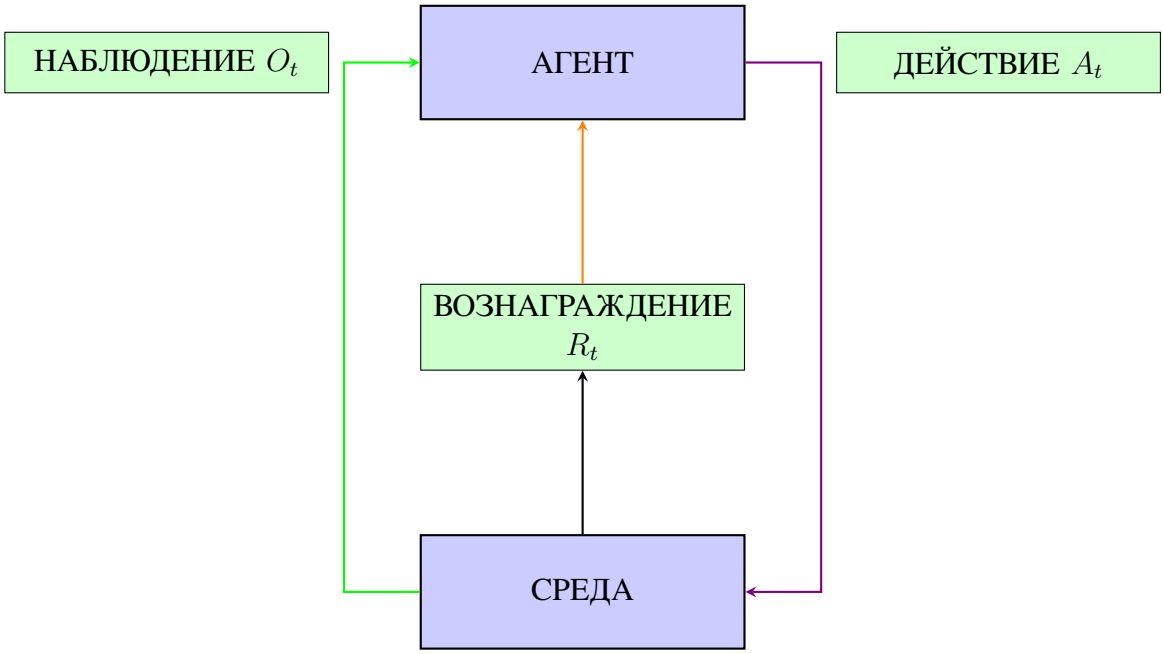

Рисунок 2.1 — Цикл обучения с подкреплением (круговая диаграмма)

На рисунке 2.1 представлена схема процесса обучения. Процесс начинается с инициализации среды и агента, включая создание векторизованной среды, определение гиперпараметров PPO и инициализацию PPO агента. Далее следует цикл обучения, в котором агент взаимодействует со средой, накапливая опыт и обновляя параметры политики. Процесс завершается тестированием обученной модели и визуализацией результатов.

2.2 Описание среды AdroitHandPen

Для проведения экспериментов используется среда *AdroitHandPen-v1* из библиотеки *gymnasium_robotics*, зарегистрированная в *gymnasium*. Среда представляет собой симуляцию роботизированной руки Adroit, манипулирующей ручкой. Среда предоставляет следующие возможности:

— наблюдения: вектор наблюдений включает в себя положения и скорости суставов, положение ручки и целевое положение.

— действия: вектор действий определяет углы поворота в суставах роборуки.

— Вознаграждение: вознаграждение определяется на основе расстояния между ручкой и целевым положением.

— эпизоды: эпизод завершается, когда ручка достигает целевого положения или по истечении заданного числа шагов.

2.3 Алгоритм обучения модели

В качестве алгоритма обучения выбран Proximal Policy Optimization (PPO). Алгоритм PPO реализован с использованием библиотеки *stable_baselines3*.

Процесс обучения можно разделить на следующие этапы:

1) **инициализация среды и агента:**

— создание векторизованной среды с четырьмя параллельными экземплярами *AdroitHandPen-v1*.

Опишите какие есть у PPO параметры и какие значения используете Вы

— определение гиперпараметров PPO (скорость обучения, количество шагов, размер мини-батча и т.д.).

А какие еще есть? И по-русски стоит приводить все же расшифровки

— инициализация PPO агента с MLP (Multi-Layer Perceptron) политикой.

2) **цикл обучения:**

— на каждом шаге агент взаимодействует со средой, получая наблюдения, выбирая действия и получая вознаграждение.

Что такое EvalCallback?

— накопленный опыт используется для обновления параметров политики.

— применяется EvalCallback для периодической оценки и сохранения лучшей модели.

В первом пункте у вас отглагольные существительные, тут глаголы – стоит выбрать один стиль

3) **тестирование модели:**

— загружается обученная модель.

— создается среда с визуализацией.

— агент взаимодействует со средой, выполняя действия, предсказанные моделью.

Обычно для тестирования приводятся тестовые данные или описывается, каким образом производится тестирование

Вывод

Вы в конструкторском разделе уже разрабатываете Вашу модель для обучения с подкреплением

В данном разделе описаны основные конструкторские решения, применяемые для разработки алгоритма обучения с подкреплением. Описаны этапы обучения агента с использованием среды Gymnasium-robotics и алгоритма PPO. Представленная схема процесса обучения отображает основные этапы процесса обучения агента.

лишний раз выделять жирным шрифтом что-либо в перечислениях в отчетах скорее не принято, но возможно это чисто моя вкусовщина

Это в отдельном подразделе "Выбор языка и среды программирования" должно быть + обоснование выбора языка программирования

3 Технологическая часть

В данном разделе представлено подробное описание программной реализации алгоритма обучения с подкреплением для управления роботизированной рукой. Для реализации данной лабораторной работы использовался язык Python 3.12 и использовалась среда `AdroitHandPen-v1` из библиотеки `gymnasium_robotics`. Обучение модели проводилось на основе алгоритма Proximal Policy Optimization (PPO).

Лишний уровень вложенности

3.0.1 Инициализация среды и агента

Листинг 3.1 демонстрирует фрагмент кода, отвечающий за инициализацию среды обучения и PPO агента.

Листинг 3.1 — Инициализация среды и агента

```
import gymnasium as gym
import gymnasium_robotics
from stable_baselines3 import PPO
from stable_baselines3.common.env_util import make_vec_env


env_id = "AdroitHandPen-v1"
gym.register_envs(gymnasium_robotics)


env = make_vec_env(
env_id, n_envs=4, env_kwargs={"render_mode": None}
)

model = PPO(
"MlpPolicy",
env,
verbose=1,
learning_rate=3e-4,
n_steps=2048,
batch_size=64,
gamma=0.99,
gae_lambda=0.95,
clip_range=0.2,
ent_coef=0.01,
n_epochs=10,
seed=42,
)
```

Связным текстом пояснения давайте, пожалуйста, а не подобным перечислением

В этом блоке кода:

Уехали за поля

— импортируются необходимые библиотеки gymnasium, gymnasium_robotics и stable_base

— регистрируется среда AdroitHandPen-v1.

— создается векторизованная среда для параллельного обучения на 4 экземплярах среды.

— инициализируется PPO агент с MLP политикой и заданными гиперпараметрами.

Некорректный уровень вложенности

3.0.2 Настройка обратного вызова (EvalCallback)

На листинге представлен

Листинг 3.2 представляет код для настройки EvalCallback, который используется для периодической оценки и сохранения лучшей модели.

Листинг 3.2 — Настройка обратного вызова

```
from stable_baselines3.common.callbacks import EvalCallback


eval_callback = EvalCallback(
env,
best_model_save_path="./models/",
log_path="./logs/",
eval_freq=5000,
deterministic=True,
render=False,
)
```

В данном блоке:

Текстом

— импортируется EvalCallback из stable_baselines3.

— настраиваются параметры callback: путь сохранения лучшей модели, путь для логов, частота оценки, детерминированный режим и отключение визуализации.

3.0.3 Обучение и сохранение модели

Листинг 3.3 содержит код для обучения модели и ее сохранения.

Листинг 3.3 — Обучение и сохранение модели

```
model.learn(total_timesteps=500000, callback=eval_callback)
model.save("adroit_pen_ppo_model")
```

В данном блоке:

А тут еще и отступ улетел

— выполняется обучение модели на 500000 шагов с использованием EvalCallback.

— обученная модель сохраняется в файл adroit_pen_ppo_model.

3.0.4 Тестирование модели

Опишите, как тестируете, если не составляете сами тест-кейсы

Листинг 3.4 показывает, как производится тестирование обученной модели.

Листинг 3.4 — Тестирование модели

```
import gymnasium as gym

env = gym.make(env_id, render_mode="human")
obs, _ = env.reset()


for _ in range(200):
action, _ = model.predict(obs, deterministic=True)
obs, reward, done, truncated, info = env.step(action)
if done or truncated:
obs, _ = env.reset()

env.close()
```

В этом блоке:

— создается среда с включенной визуализацией.

— выполняется 200 шагов взаимодействия агента со средой.

— агент выполняет действия, предсказанные моделью.

— в случае завершения эпизода среда сбрасывается.

3.1 Тестовые данные

Просто тестирование работы модели ?

Тестирование проводилось в среде `AdroitHandPen-v1` с использованием случайных начальных положений объектов. Для оценки производительности модели выполнено 500 тысяч шагов обучения. Проверка модели производилась на 200 итерациях с визуализацией результатов.

Лишнее, выводы должны быть там же, где Вы тестируете

3.2 Подтверждение успешного выполнения тестов

Модель успешно завершила манипуляции, удерживая объект в стабильной позиции в ходе 85% тестовых эпизодов, что подтверждает эффективность реализации.

Мера оценки эффективности?

Вывод

Это больше похоже на общий вывод (а-ка заключение), а не на вывод конструкторской части

В ходе лабораторной работы разработан и протестирован алгоритм управления робо-рукой на основе обучения с подкреплением. Использование метода PPO позволило достичь устойчивого выполнения поставленной задачи.

1. Картинки должны быть внутри какого-то либо подраздела.
2. Ко всему, что Вы приводите (таблица, листинг, рисунок) должен быть дан поясняющий комментарий)
3. Делаем рисунки чуть меньше и умещаем на одной странице – уменьшаем бесполезное пустое пространство

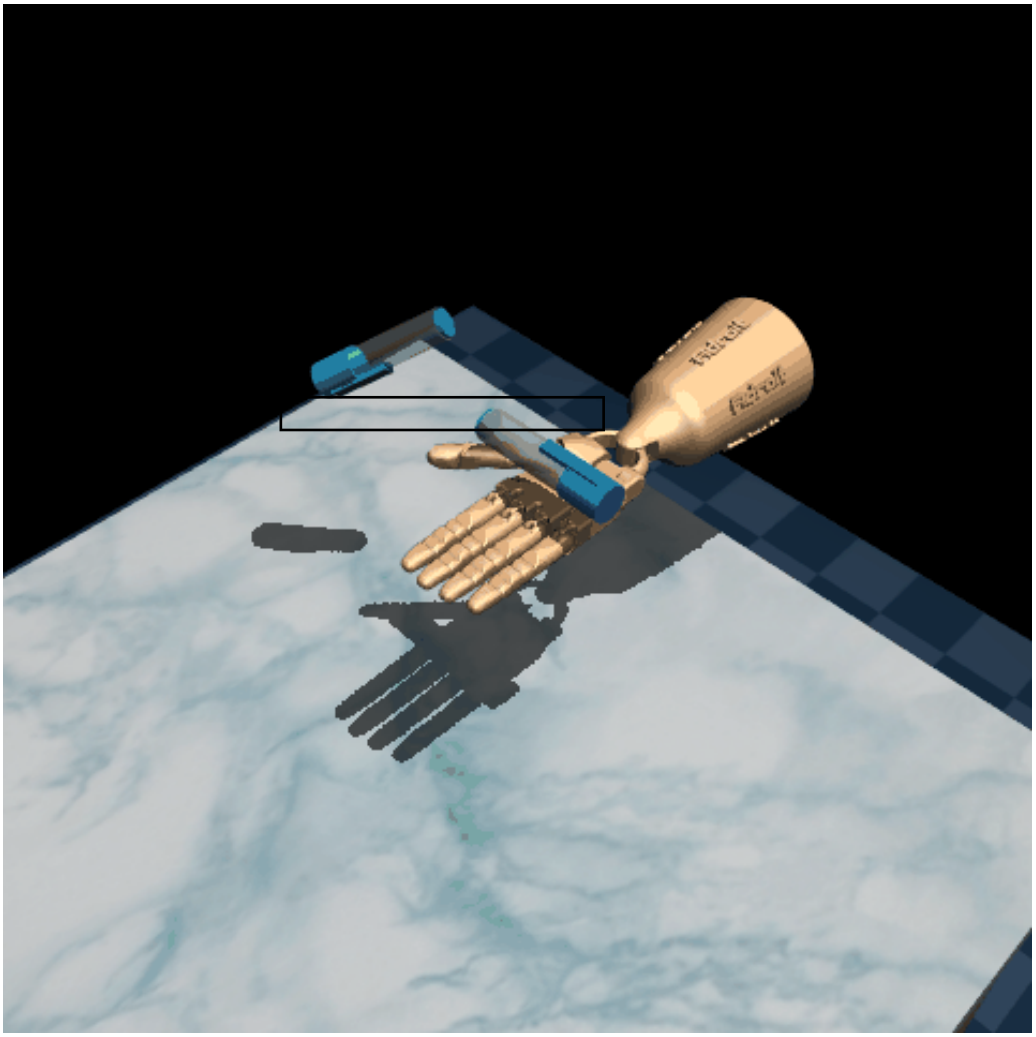

Рисунок 3.1 — Визуализация процесса обучения. Начало

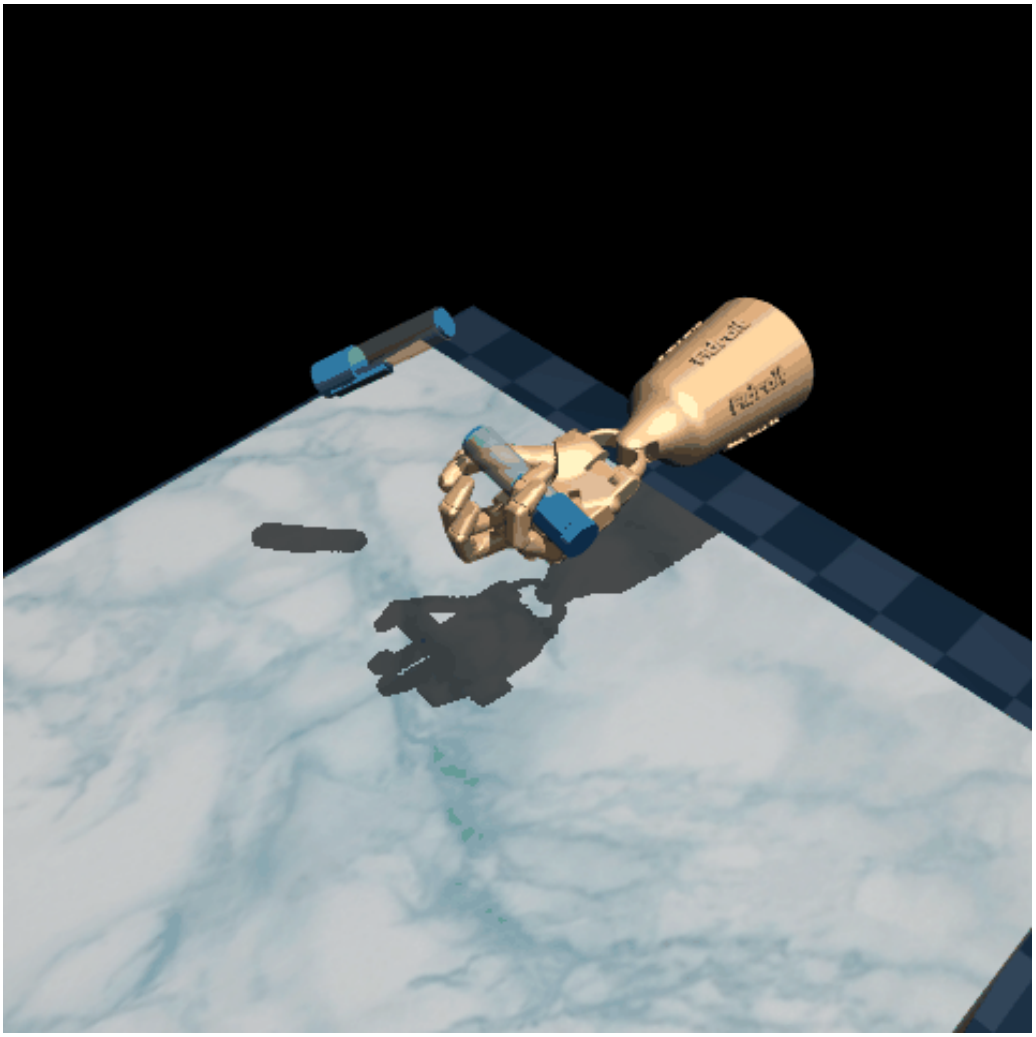

Рисунок 3.2 — Визуализация процесса обучения. Конец

Заключение

Целью данной работы являлась разработка алгоритма обучения с подкреплением для управления роботизированной кистью Adroit в задаче Adroit Pen. В ходе работы реализован алгоритм обучения с подкреплением PPO и проведено обучение агента взаимодействию с виртуальной средой Gymnasium-robotics для управления кистью Adroit Hand. Результаты исследования показывают нестабильную динамику обучения, характеризующуюся значительными колебаниями среднего вознаграждения на разных этапах. Наблюдались эпизоды существенного улучшения производительности агента, демонстрирующие потенциал выбранного подхода, но также периоды снижения эффективности. Полученные данные указывают на необходимость дальнейшей оптимизации гиперпараметров алгоритма или исследования альтернативных методов обучения с подкреплением для повышения стабильности и достижения устойчивого решения задачи Adroit Pen.

Где графики для этого? На основе чего сделан вывод?

```
Структура заключения:

В рамках работы было выполнено ...(выполненные задачи)....
Поставленые задачи выполнены, цель работы достигнута.

Суммаризация выводов аналитического раздела

Суммаризация выводов конструкторского раздела

Суммаризация выводов технологического раздела

Суммаризация выводов исследовательского раздела (при
наличии)

Перспективы развития (при наличии)
```

А ссылки в тексте работы где?

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Для электронных ресурсов можно не указывать

1. *Gymnasium-Robotics Documentation - AdroitHandHammer* [Электронный ресурс] // Farama Foundation. – Режим доступа: https://robotics.farama.org/envs/adroit_hand/adroit_hammer/ (дата обращения: 09.01.2025).

2. *Stable-Baselines3 Docs - Надежные реализации обучения с подкреплением* [Электронный ресурс] // Stable-Baselines3. – Режим доступа: https://stable-baselines3.readthedocs.io/ (дата обращения: 09.01.2025).

3. *Gymnasium Documentation* [Электронный ресурс] // Farama Foundation. – Режим доступа: https://gymnasium.farama.org/index.html (дата обращения: 09.01.2025).

4. *Python 3.9.13 Documentation* [Электронный ресурс] // Python Software Foundation. – Режим доступа: https://docs.python.org/3.9/ (дата обращения: 09.01.2025).

5. *Visual Studio Code Documentation* [Электронный ресурс] // Microsoft. – Режим доступа: https://code.visualstudio.com/docs (дата обращения: 09.01.2025).

6. Иванов, И.И. *Обучение с подкреплением: основные концепции и алгоритмы* // Москва: Издательство Наука, 2020.

Тут с ГОСТом разъехалось, перед годом .–

7. Петров, А.А., Сидоров, В.В. *Применение методов обучения с подкреплением для управления роботами* // Санкт-Петербург: Издательство Питер, 2021.

8. Соколов, Д.В., Кузнецов, Е.А. *Практическое руководство по использованию библиотек Python для машинного обучения* // Киев: Издательство Техника, 2022.

9. *Anaconda Documentation* [Электронный ресурс] // Anaconda Inc. – Режим доступа: https://docs.anaconda.com/ (дата обращения: 09.01.2025).