**Министерство науки и высшего образования Российской Федерации**

**федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет ИТМО»**

**(Университет ИТМО)**

Факультет СУ и Р

**Образовательная программа** Робототехника и искусственный интеллект

О Т Ч Е Т

о производственной проектно-конструкторской практике

Тема задания: Подготовка данных для обучения нейронных сетей (и обучение)

Обучающийся Овчинников Павел Алексеевич, гр. R3242

Руководитель практики от университета: Ведяков Алексей Алексеевич, доцент

Санкт-Петербург

2024

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc1" \o "#_Toc1)

[1 Выбор среды 4](#_Toc2" \o "#_Toc2)

[2 Поиск агента 4](#_Toc3" \o "#_Toc3)

[3 Сбор датасетов 5](#_Toc4" \o "#_Toc4)

[4 Оформляем всё как контейнер 6](#_Toc5" \o "#_Toc5)

[ВЫВОДЫ 6](#_Toc6" \o "#_Toc6)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 7](#_Toc7" \o "#_Toc7)

# ВВЕДЕНИЕ

К выполнению предлагается задание: подготовка данных для обучения нейронных сетей (и обучение). В частности, задача заключается в сборе датасетов для имитационного обучения (imitation learning) моделей управления физическими системами на основе существующих агентов.

Коротко об имитационном обучении — это тип социального обучения, при котором новое правильное поведение приобретается через подражание. В робототехнике такое обучение используется как альтернатива традиционному обучению с подкреплением, позволяя роботам учиться на основе успешных примеров, продемонстрированных человеком, вместо того чтобы начинать со случайных действий и самостоятельно искать правильное решение проблемы. Это помогает быстрее и эффективнее достигать целей, особенно в сложных задачах с редкими наградами.

В нашем случае мы имеем уже заранее предписанную среду исследования (observation), для которой требуется решить проблему. Эту проблему решает агент (agent), который на основе наблюдений среды выполняет действия (actions). Среда вознаграждает агента (reward) за правильные действия, которые приближают к решению проблемы.

Обычно в роли агента выступает уже обученная с подкреплением нейросеть, которая способна выполнять задачу достаточно хорошо, либо человек, который хорошо понимает, какие успешные примеры необходимо продемонстрировать, чтобы по ним можно было проводить имитационное обучение. Обычно последний вариант более затратный, но в то же время мы получаем более качественный датасеты.

Здесь и появляются те самые датасеты, собранные на основе полученной информации о действиях «профессионального» агента. Мы имеем действия агента и то, как на них реагирует среда — с такой информацией можно обучать «новеньких» агентов методом имитационного обучения.

Изначальный стек:

* WSL 2 → Ubuntu 22.04.1 LTS
* Docker version 26.1.4
* VS Code 1.91.0-insider (system setup)
* git version 2.37.3.windows.1

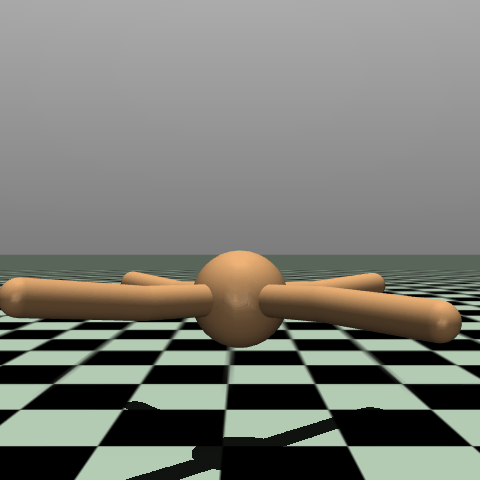
В дальнейшем из-за планового переезда летом на Linux стек поменялся:

* KDE Neon 6.0, основанный на Ubuntu
* Docker version 27.0.2
* VS Code 1.91.0-insider
* git version 2.34.1

Весь код, написанный во время прохождения практики, находится в репозитории на GitHub: [https://github.com/snowlue/ITMO-Practice1](https://github.com/snowlue/ITMO-Practice1" \o "https://github.com/snowlue/ITMO-Practice1).

# Выбор среды

Передо мной встал вопрос выбора среды, с которой я буду работать. Сначала выбор пал на Reacher — двухшарнирного робота, задача которого переместить свой конец как можно ближе к точке, которая появляется в случайном месте, но затем эта среда мне показалось слишком простой в сравнении с Ant. Ant —- это 3D-робот, состоящий из одного свободно вращающегося туловища с прикрепленными к нему четырьмя ногами, каждая из которых имеет две части, соединённые коленом. Цель состоит в том, чтобы скоординировать движение четырех ног в прямом направлении путём приложения крутящих моментов к восьми шарнирам, соединяющим две части каждой ноги и туловище.



Эта среда кажется интереснее с точки зрения исследования, так как перед нами пространство из 8 действий, предлагаемых агенту, а пространство наблюдений состоит из 27 величин.

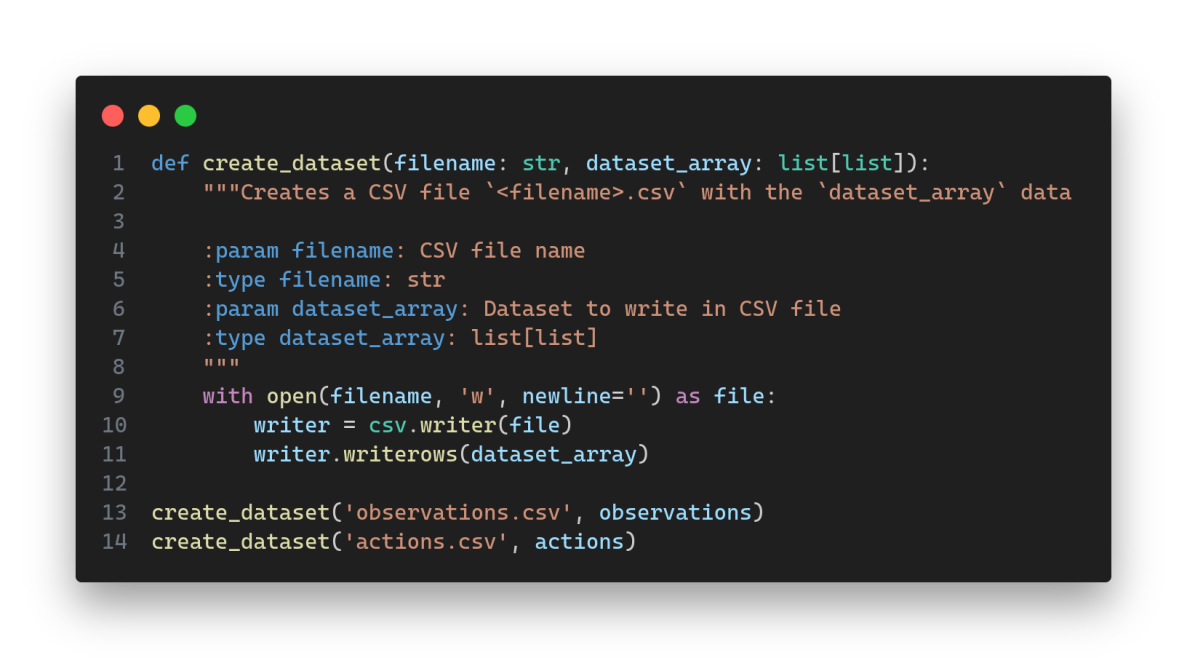
# Поиск агента

Поиск агента привёл меня к официальному гайду в документации gymnasium, который объясняет, [как быстро сделать агента с обучением с подкреплением для обратного маятника](https://gymnasium.farama.org/tutorials/training_agents/reinforce_invpend_gym_v26/" \o "https://gymnasium.farama.org/tutorials/training_agents/reinforce_invpend_gym_v26/). По сути это готовый макет для создания своего агента, в котором мы можем регулировать типы нейронов, их количество в каждом слое, количество слоёв и т.д. Так как мы не занимаемся обучением агента в рамках практики, я оставляю большинство параметров как есть и двигаюсь дальше. Класс Policy\_Network в коде описывает структуру нейронной сети, а класс REINFORCE реализует обучение с подкреплением.

# Сбор датасетов

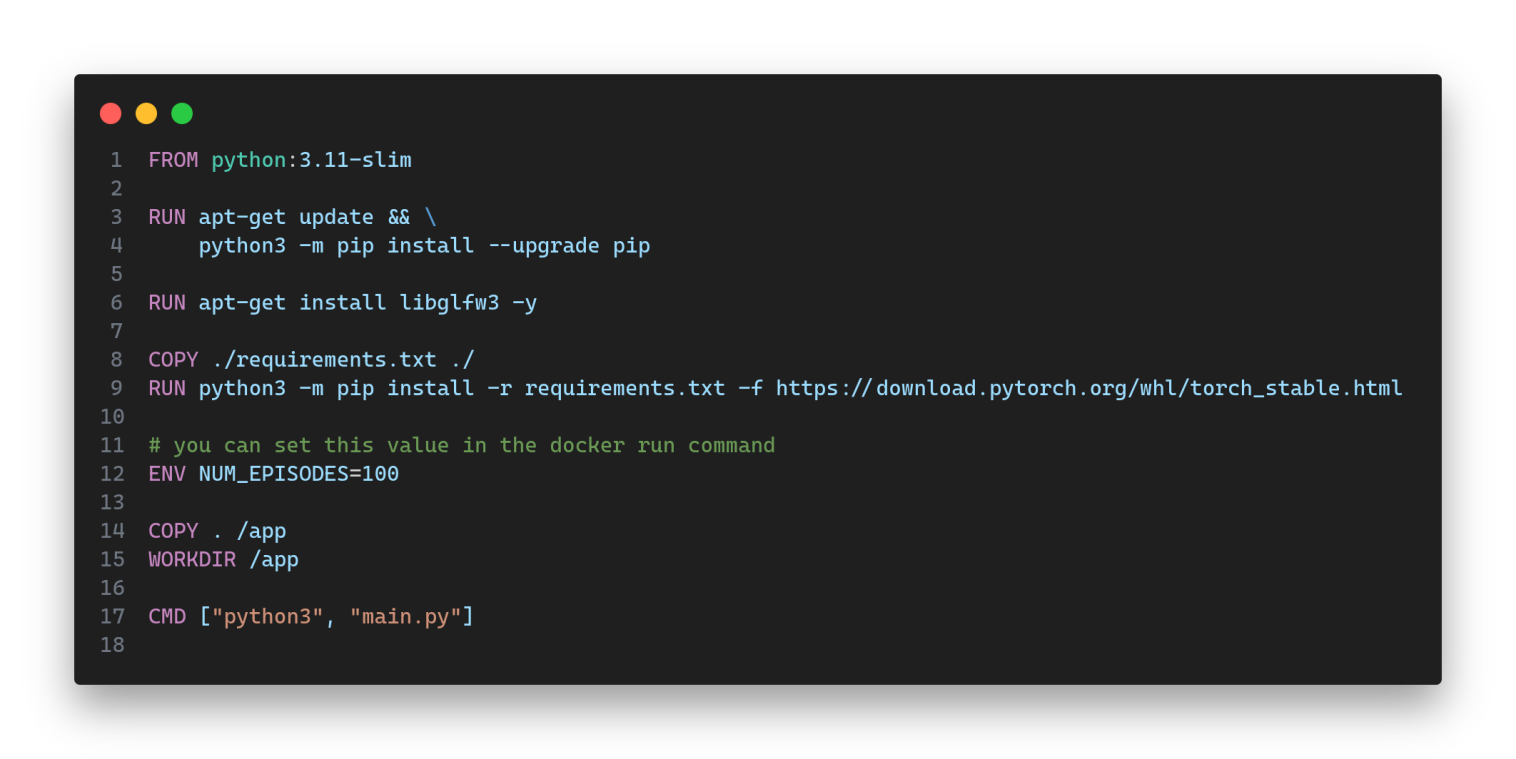
Теперь необходимо написать код, который будет запускать N эпизодов среды и собирать данные наблюдений и данные действий агента. Для этого напишем следующий код.

Здесь мы запускаем среду, получаем количество эпизодов из переменной окружения, которая в дальнейшем будет задана внутри Docker-контейнера, создаём агента и запускаем цикл эпизодов. Каждый эпизод мы будем проигрывать до того момента, пока не сработают триггеры завершения эпизода внутри среды.

Да-да, наш агент будет обучаться параллельно с наполнением датасета. Это связано с тем, что каждый раз контейнер будет запускаться с нуля, а процесс обучения требует нескольких тысяч эпизодов, что займёт некоторое время. В рамках обучения процессу сбора данных такой случай подойдёт, но в реальных задачах необходимо использовать предобученного агента.

Нам остаётся только лишь записать полученные данные в CSV-файлы. Для этого создадим функцию create\_dataset.

# Оформляем всё как контейнер

Последние штрихи — оформляем всё как контейнер. Для этого используем образ python:3.11-slim, в котором нет ничего лишнего и который весит совсем немного. Нам понадобится библиотека libglfw3 для OpenGL и несколько модулей из requirements.txt, которые я туда добавил: gymnasium[mujoco] и torch==2.3.1+cpu (с приоритетной выгрузкой из стабильных версий pytorch). Далее задаём переменную NUM\_EPISODES, через которую можно будет задавать количество эпизодов при запуске контейнера.

Остаётся только прописать в README.md, как нужно работать с кодом, чтобы датасет собирался и выгружался в ту же папку, в которой был запущен образ. Посмотреть результат можно на моём репозитории в GitHub — [https://github.com/snowlue/ITMO-Practice1](https://github.com/snowlue/ITMO-Practice1" \o "https://github.com/snowlue/ITMO-Practice1).

# ВЫВОДЫ

Имитационное обучение — дело непростое, потому как требует от себя наличия как минимум одного «профессионального» агента, который сможет выполнить задачу так, чтобы мы получили максимально полезный датасет. В противном случае новый агент будет перенимать те же ошибки, что были у того, на ком он учился.

Я научился собирать датасет, который, с некоторыми оговорками, пригоден для имитационного обучения — входами в нейронную сеть обучаемого агента будут данные из observations.csv и её задачей будет выдавать что-то максимальное похоже на действия прошлого агента, хранящиеся в actions.csv.

Также я узнал, чем имитационное обучение отличается от обучения с подкреплением, потому как раньше часто путал их друг с другом: задача в обучении с подкреплением максимизировать выгоду, а в имитационном обучении —- как можно более точное воспроизведение действий эксперта.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Gymnasium Documentation — Basic Usage [Электронный ресурс]. URL: [https://gymnasium.farama.org/content/basic\_usage/](https://gymnasium.farama.org/content/basic_usage/" \o "https://gymnasium.farama.org/content/basic_usage/)
2. Репозиторий ITMO-Practice1 на GitHub [Электронный ресурс]. URL: [https://github.com/snowlue/ITMO-Practice1](https://github.com/snowlue/ITMO-Practice1" \o "https://github.com/snowlue/ITMO-Practice1)
3. Gymnasium Documentation — Ant [Электронный ресурс]. URL: [https://gymnasium.farama.org/environments/mujoco/ant](https://gymnasium.farama.org/environments/mujoco/ant" \o "https://gymnasium.farama.org/environments/mujoco/ant)
4. Gymnasium Documentation — Training using REINFORCE for Mujoco [Электронный ресурс]. URL: [https://gymnasium.farama.org/tutorials/training\_agents/reinforce\_invpend\_gym\_v26](https://gymnasium.farama.org/tutorials/training_agents/reinforce_invpend_gym_v26" \o "https://gymnasium.farama.org/tutorials/training_agents/reinforce_invpend_gym_v26)
5. Wikipedia — Imitative Learning [Электронный ресурс]. URL: [https://en.wikipedia.org/wiki/Imitative\_learning](https://en.wikipedia.org/wiki/Imitative_learning" \o "https://en.wikipedia.org/wiki/Imitative_learning)