

Лабораторна робота №2.

Перша політика навчання з підкріпленням для висіння

10 вересня 2026 р.

Мета

Сформулювати задачу висіння нанодрона Crazyflie 2.1 як задачу навчання з підкріпленням: спостереження, дія, винагорода. Навчити політику алгоритмом PPO у середовищі `gym-pybullet-drones`, оцінити її на 20 епізодах за середньою віддачею й часткою успішних епізодів і порівняти з ПІД-регулятором із роботи №1.

Склад бригади і розподіл ролей

Робота виконується бригадою з трьох осіб. Кожен учасник відповідає за свою підсистему й захищає її окремо.

- **Учасник А — середовище навчання.** Реалізує середовище, сумісне з Gymnasium: простір спостережень, простір дій, умови дострокового завершення, обчислення нормованої віддачі.
- **Учасник В — навчання політики.** Запускає PPO, веде журнал у TensorBoard, фіксує криву навчання й характерні фази, повторює навчання щонайменше на п'яти зернах.
- **Учасник С — оцінювання і порівняння.** Пише критерій успіху, оцінює політику на 20 епізодах, порівнює її з ПІД-регулятором із роботи №1 і готує відео польоту.

Інтеграційний крок виконують разом: середовище учасника А приймає політику учасника В, після чого учасник С проганяє обидві стратегії на однакових початкових станах.

Теоретичні відомості

Політику реалізує нейромережа з двома шарами по 64 нейрони. Вона 50 разів на секунду отримує спостереження й видає дію. Спостереження складається з дев'яти чисел бортового фільтра Калмана: похибка висоти $z - z^*$ (м), швидкості v_z, v_x, v_y (м/с), крен і тангаж (рад), кутові швидкості p, q, r (рад/с). Дія *одна*: нормована тяга $u \in [0; 1]$. Кутове положення утримує бортовий ПІД із прошивки.

Фізична тяга дорівнює $u \cdot F_{\max}$, де $F_{\max} = 0,596$ Н (60,8 г-сили) при масі $m = 27$ г — це модель CF2X, та сама, що в роботі №1. Отже, при $u = 0,44$ апарат висить, при $u = 1,0$ набирає ≈ 12 м/с², при $u = 0$ вільно падає; навчена політика має тримати $u \approx 0,44$.

Винагороду нараховують на кожному кроці, тобто 50 разів на секунду:

$$r_t = -w_z |z_t - z^*| - w_v |v_{z,t}| - w_u |u_t - u_{t-1}| + b \quad (1)$$

де z_t, z^* — поточна й задана висота, м (перший член тримає висоту); $v_{z,t}$ — вертикальна швидкість, м/с (не дає «пружинити»); u_t, u_{t-1} — тяга на поточному й попередньому кроці (не дає смикати мотори); $w_z = 1,00$, $w_v = 0,20$, $w_u = 0,05$ — ваги штрафів; $b = 0,10$ — бонус за життя за

кожен крок, на якому епізод не обірвався. Без нього всі складові від’ємні й апарату вигідно швидко впасти.

Один крок циклу триває 0,02 с модельного часу. Епізод завершується після вичерпання часу ($8\text{ с} = 400$ кроків) або достроково, якщо апарат упав ($z < 0,05\text{ м}$) чи вилетів за межі зони ($|z - z^*| > 1,0\text{ м}$). Епізоди мають різну довжину, тому всюди беремо *нормовану* віддачу — суму винагород, поділену на тривалість епізоду в секундах (ідеальному висінню відповідає +5,0).

Крива навчання не монотонна й має характерні фази. Одна крива нічого не доводить: за бюджету 500 тис. кроків запуски з різними seed розходяться на 1–2 одиниці.

Кроки навчання	Що робить апарат	Нормована віддача
0–50 тис.	падає одразу, епізод 0,5 с	≈ -8
50–200 тис.	тримається $\approx 1\text{ с}$	до -3
200–500 тис.	«пружинить» на $\pm 25\text{ см}$	$-1,0 \dots -0,4$
0,5–1 млн	тримає $\pm 8\text{ см}$	$+2 \dots +4$
1–2 млн	тримає $\pm 2 \dots 3\text{ см}$	плато $+4,5$

Корисні посилання для ознайомлення:

- [Stable-Baselines3 documentation](#) — реалізація PPO, яку використовуємо;
- [Schulman та ін. Proximal Policy Optimization Algorithms](#) — оригінальна стаття про PPO;
- [Gymnasium](#) — інтерфейс середовища;
- [Henderson та ін. Deep RL that Matters](#) — чому одна крива нічого не доводить.

Середовище

Python 3.12, gym-pybullet-drones (MIT) з усіма залежностями: numpy, matplotlib, gymnasium, stable-baselines3, pybullet. Усе встановлюється з PyPI і GitHub.

Одна тонкість встановлення: pybullet не має готових збірок під Python 3.12, тому компілюється з джерел і потребує компілятора C++. На Ubuntu це `sudo apt install build-essential`, на macOS — інструменти командного рядка Xcode. Якщо збірка не проходить, робіть роботу в хмарному ноутбукі з попередньо встановленим компілятором — умови завдання від цього не змінюються.

Порядок виконання

Обчислювальний бюджет. Відеокарта не потрібна: усі запуски розраховані на процесор звичайного ноутбука. Тримайтеся межі **30 хвилин на один запуск навчання** (п’ять зерен послідовно — близько двох з половиною годин, їх зручно лишити рахуватися без нагляду); кількість кроків добирайте під цю межу й фіксуйте її в ноутбукі однаковою для всіх порівнюваних запусків. Криву, що не встигла вийти на полицю, теж можна аналізувати — висновок робиться про *однакову* недонавчені політики.

1. **[разом]** Сформулювати задачу висіння як задачу навчання з підкріпленням: простір спостережень, простір дій і функцію винагороди за формулою (1). Обґрунтувати склад кожної з трьох частин.
2. **[A]** Реалізувати середовище, сумісне з інтерфейсом Gymnasium, з умовами дострокового завершення й обчисленням нормованої віддачі. Перевірити його коротким пробним навчанням.

3. **[В]** Навчити політику алгоритмом РРО. Вести журнал так, щоб хід навчання було видно за середньою віддачею й довжиною епізоду; зафіксувати моменти, коли епізод уперше дожив до кінця і коли віддача стала додатною.
4. **[С]** Ввести власний критерій успішного епізоду й оцінити навчену політику на 20 епізодах детермінованими діями. Звести показники в таблицю.
5. **[В]** Повторити навчання **щонайменше на п'яти зернах**, не змінюючи жодного іншого параметра, і подати всі криві на одному графіку із середнім і смугою розкиду. Обов'язково підпишіть, що саме показує смуга — стандартне відхилення чи похибку середнього. П'ять зерен — це не примха: за меншого числа висновків може перевернутися від запуску до запуску, і чек-лист відтворюваності вимагає саме п'яти.
6. **[С]** Оцінити ПД-регулятор із роботи №1 на тих самих епізодах і початкових станах. Порівняти його з навченою політикою за точністю, швидкодією та гладкістю керування.
7. **[С]** Записати відео одного епізоду й долучити кадри старту, перехідного процесу та усталеного висіння.
8. **[разом]** Проаналізувати форму кривої навчання, розкид між зернами й те, у чому політика програє ПД-регулятору.
9. **[разом]** **Оформити ноутбук.** Зібрати роботу в один Jupyter-ноутбук: код кожного учасника у своїх комітках із короткими коментарями там, де це доречно. Обов'язково всередині: постановку задачі, криві навчання, показники оцінювання, порівняння з ПД-регулятором і кадри з відео, а також висновки — три-п'ять перевірюваних тверджень із числами. Ноутбук має виконуватися від початку до кінця без ручних правок; у першій комітці вкажіть, хто з учасників за які комірки відповідав.

Контрольні запитання

1. Які дев'ять чисел подаються в політику? Чому похибка $z - z^*$, а не z ?
2. Чому політика видає одне число, а не чотири сигнали на мотори?
3. Що фізично означають $u = 0$; $u = 0,44$; $u = 1,0$ для апарата масою 27 г?
4. Поясніть чотири члени формули (1). Що станеться без бонусу b ?
5. Скільки кроків містить епізод 8 с за частоти 50 Гц і яка максимальна віддача?
6. Опишіть фази кривої навчання. Як видно, що навчання застрягло на першій фазі?
7. Що саме змінює seed і чому результат не можна подавати однією кривою?
8. Політика має високу віддачу, але низьку частку успішних епізодів. Чому?
9. За якими трьома показниками порівнюють політику з ПД?