https://doi.org/10.31891/2307-5732-2025-355-33
УДК 004.8’231:623.746.9:681.518

КОПИЛЕЦЬ МИХАЙЛО
Національний університет «Львівська політехніка»
https://orcid.org/0009-0004-5823-9871
e-mail: mykhailo.m.kopylets@lpnu.ua

ПОРІВНЯННЯ ЕФЕКТИВНОСТІ RL-АЛГОРИТМІВ ДЛЯ БЕЗПЕЧНОГО ОБХОДУ ПЕРЕШКОД БПЛА

У статті оцінено застосування глибинного навчання з підкріпленням для побудови автономних стратегій обходу перешкод безпілотними літальними апаратами у двовимірній симуляції з густою сіткою нерухомих перешкод. Розроблено симуляцію, де агент сприймає сектор простору попереду і виконує дискретні команди, що поєднують кут повороту та крок вперед, імітуючи реальні обмеження бортових контролерів. Дослідження зосереджено на порівнянні двох підходів Deep Q-Network (DQN) та Proximal Policy Optimization (PPO) за здатністю формувати безпечні траєкторії та запобігати зіткненням на основі ряду якісних характеристик. Додатково проаналізовано вплив різних математичних функцій активації, таких як ReLU, Leaky ReLU, Tanh, Sigmoid, на швидкість навчання та характер руху. З’ясовано, що у складних конфігураціях з багатьма перешкодами політики PPO забезпечують більш рівномірне оновлення ваг і стійкішу поведінку агента, тоді як підхід DQN демонструє високу реактивність у вузьких “коридорах” симуляції, дозволяючи обходити щільні скупчення об’єктів з мінімальними відхиленнями від запланованого маршруту.

Як критерії ефективності використовувалися здатність адаптуватися до змін розташування перешкод, плавність траєкторії, кількість різких зупинок та маневрів, стабільність навчання. Для задач, де пріоритетом виступає безпека польоту, ефективніше застосовувати конфігурації на базі PPO з функціями активації, що обмежують амплітуду градієнта. У той же час для завдань з акцентом на швидкі маневри у щільному просторі перешкод доцільніше вирішувати за допомогою DQN із застосуванням лінійних активацій без обмежень на амплітуду виходу.

Отримані результати підкреслюють необхідність комплексного підходу до вибору алгоритмічної стратегії та архітектурної конфігурації мережі з урахуванням особливостей середовища, схеми винагороди та технічних обмежень платформи. Подальші дослідження варто сконцентрувати на поширенні моделювання на тривимірні простори, інтеграції даних з реальних сенсорів і перевірці алгоритмів у польотних сценаріях з апаратною взаємодією.

Ключові слова: DQN, PPO, безпілотник, уникнення перешкод.

KOPYLETS MYKHAILO
Lviv Polytechnic National University

COMPARISON OF THE EFFECTIVENESS OF RL ALGORITHMS FOR SAFE UAV OBSTACLE AVOIDANCE

An assessment of advanced decision-making techniques based on deep reinforcement learning has been performed for unmanned aerial vehicles operating in a two-dimensional virtual arena populated by a dense array of stationary obstacles. A custom simulation platform enables each drone to observe only a limited forward sector and to issue one of several discrete commands, each of which encodes both a change in heading and a forward translation step. This design faithfully reproduces the quantized control signals typical of onboard flight computers. Attention was directed toward two leading methods. One approach is built on value-based updates through a deep Q-network structure while the other relies on policy-gradient optimization under the proximal policy optimization paradigm. Both families of agents were trained under identical conditions, after which their capacity to produce collision-free flight paths was evaluated using a collection of qualitative measures. Those measures included the ability to adapt when the arrangement of obstacles changes, the overall smoothness of the flight trajectory, the frequency of abrupt stops and sharp turns, and the consistency of learning progress over time.

Additional analysis examined how different mathematical activation functions, including the widely used rectified linear unit, its leaky variant, as well as hyperbolic tangent and logistic sigmoid, affect the speed at which each agent reaches reliable performance and the properties of the resulting motion patterns. It emerged that the policy-gradient agents generally maintain more uniform parameter updates and exhibit greater stability when navigating complex topologies. In contrast, the value-based agents excel at rapid, reactive maneuvers in narrow pathways, weaving through tightly clustered obstacles with minimal deviation from their intended course.

The overall findings suggest that when the primary objective is to maximize flight safety and minimize collision risk, configurations derived from proximal policy optimization combined with activation schemes that curb excessive gradient growth are most suitable. Conversely, missions that demand swift directional changes in confined environments are better served by the deep Q network model paired with activation functions that allow unrestricted linear growth. These outcomes underscore the need for a holistic selection process when choosing both the learning algorithm and neural network architecture, taking into account environmental complexity, reward structure, and hardware constraints. Future investigations should broaden the scope to three-dimensional scenarios, incorporate real-world sensor data, and validate the proposed methods through hardware-in-the-loop flight trials.

Keywords: DQN, PPO, UAV, collision avoidance.

Стаття надійшла до редакції / Received 22.05.2025
Прийнята до друку / Accepted 16.06.2025

Вступ

Безпілотні літальні апарати (БПЛА) [1] уже стали звичним інструментом у цивільних та військових операціях, від моніторингу урожаю й інфраструктури до пошуково-рятувальних місій і тактичної розвідки. Поточна безпекова ситуація в Україні тільки підкреслює стратегічне значення дронів, на полі бою вони мають працювати автономно, швидко приймати рішення й уникати зіткнень у щільно забудованих містах. Класичні методи планування маршруту базуються на детермінованих алгоритмах, що передбачають повну карту середовища й значні обчислювальні ресурси. Проте польові умови часто характеризуються неповною або

неточною інформацією, динамічними перешкодами й жорсткими обмеженнями на енергоспоживання та вагу бортової електроніки.

Підходи глибинного навчання з підкріпленням (Deep Reinforcement Learning, DRL) [2] пропонують альтернативу, оскільки агент навчається безпосередньо на взаємодії із середовищем і може адаптувати політику до нових конфігурацій перешкод без попереднього знання карти. Серед відомих методів DRL виділяють Deep Q-Network (DQN) [3] та Proximal Policy Optimization (PPO) [4]. Їх успішно застосовували у різних робототехнічних задачах, проте ефективність цих алгоритмів у щільному статичному середовищі, потребує додаткового дослідження. З цією метою створено двовимірну симуляцію, де агент бачить лише сектор простору попереду й використовує дискретні дії, що поєднують поворот і крок уперед. Таке середовище відображає обмежений огляд і квантоване управління, типові для малих дронів з невеликою ємністю акумулятора, а завдання уникнення зіткнень у такій постановці демонструє практичну проблему забезпечення безпечного польоту без громіздких карт і важких обчислень.

Аналіз досліджень та публікацій

В дослідженні [5] автори порівнюють ефективність DQN і PPO у моделюванні завдання сортування матеріалів. Результати демонструють, що PPO перевершує DQN за швидкістю навчання та завершенням завдання. Робота також підкреслює важливість правильної побудови функції винагороди, адже зміна її структури впливає на якість навчання моделей. Автори показують переваги використання передових методів підкріплювального навчання, зокрема PPO, та рекомендують досліджувати складніші підходи для підвищення продуктивності в індустріальних застосуваннях. Отримані висновки мають широку корисність. Вони надають цінну інформацію для інших галузей, зокрема для навігації та уникнення перешкод дронами, попри те, що початковим об’єктом дослідження було інше завдання..

Ще одне дослідження [6] пропонує докладну оцінку DQN, PPO та Soft Actor-Critic (SAC) [7] у контексті уникнення перешкод БПЛА, де розглядаються як статичні, так і рухомі об’єкти. Робота виконана у тривимірному симульованому середовищі і підкреслює важливість вибору належної стратегії навчання з підкріпленням для специфічних завдань дронів, особливо у складних і динамічних сценах. SAC демонструє найкращий результат, що підтверджує переваги алгоритмів без політик у ситуаціях, де потрібні гнучкі рішення. Успіх DQN, попри обмежений дискретний простір дій, показує роль зразкової ефективності та стратегічного використання буферів відтворення досвіду (experience replay buffer). Натомість труднощі PPO висвітлюють обмеження методів залежних від політик під час роботи у тривимірних середовищах із динамічними перешкодами. Це порівняння робить вагомий внесок у дискусію про ефективну навігацію БПЛА та закладає основу для подальших досліджень. Автори вказують на доцільність гібридних підходів або подальших удосконалень алгоритмів, щоб подолати виявлені обмеження, особливо для методів на зразок PPO.

Формулювання цілей статті

Мета роботи полягає у зіставленні можливостей DQN і PPO щодо формування безпечних траєкторій у середовищі з щільною сіткою перешкод і виробленні рекомендацій для практичної інтеграції в системи управління БПЛА. Для досягнення цієї мети поставлено завдання:

- створити наближене 2D-середовище, яке відтворює характерні виклики навігації дронів;
- адаптувати DQN і PPO до задачі уникнення зіткнень із урахуванням обмеженого поля зору й дискретного простору дій;
- дослідити вплив різних функцій активації (ReLU, Leaky ReLU, Tanh, Sigmoid) на процес навчання та стиль маневрування;
- порівняти траєкторії агентів отримані в результаті навчання.

Виклад основного матеріалу

Технологічна основа симуляційного середовища побудована на мові Python, яка відзначається динамічністю та гнучкістю. Екосистема Python з широким набором бібліотек і фреймворків підтримує швидку розробку та прототипування, що робить її ідеальним вибором для створення детальних симуляцій. Графічну візуалізацію та інтерактивні елементи середовища забезпечує бібліотека Pygame [12] - кросплатформений набір модулів для Python, призначений для розробки відеоігор. Її потужний функціонал і зрозумілий синтаксис створюють зручний інтерфейс для відображення сцени та обробки взаємодій користувача чи штучного інтелекту. Штучний інтелект симуляції реалізовано з використанням бібліотеки stable_baselines3 [13, 14], яка містить надійні алгоритми підкріплювального навчання для Python. Ця бібліотека пропонує готові нейронні політики, такі як DQN та PPO, що є ключовими для розробки моделей навігації дронів. Використання stable_baselines3 спрощує впровадження складних алгоритмів і забезпечує єдність та відтворюваність процесу навчання.

Симуляційне середовище зображено на рис. 1. Це інтерактивний двовимірний простір, де дрон взаємодіє з різноманітними перешкодами. Межі арени оформлені у вигляді квадратів, що задають чіткі кордони й створюють навігаційні виклики для дрона. Розташування й розміри цих кордонів безпосередньо впливають на планування маршруту та стратегії маневрування. Всередині арени перешкоди подано круги з різними радіусами. Їхнє розташування формує складний ландшафт для навігації. Конкретні координати й розміри кругів значною мірою визначають рівень складності симуляції, забезпечуючи реалістичну апроксимацію руху в небезпечних просторах Дрон має форму круга з обладнаною сенсорною системою. Сенсор розташовано в центрі дрона, він охоплює 180° передньої зони з певним невеликим радіусом.

Інформація, отримана через сенсор, відіграє ключову роль у прийнятті рішень, дозволяючи дрону реагувати та адаптуватися до оточення. Інтерпретація даних сенсора виконується через перетворення абсолютних координат перешкод у систему відліку дрона. Це перетворення {15] включає зміщення початку координат у положення дрона та обертання системи відповідно до його напрямку руху (рис. 2):

- Зсув системи координат так, щоб початок нової системи співпадав із поточним положенням дрона

$$x' = x - x_{drone},$$
$$y' = y - y_{drone}, \quad (1)$$

тут (x', y') - координати перешкоди в системі відліку дрона, (x, y) - початкові абсолютні координати, а (x_{drone}, y_{drone}) - координати дрона.

- Обертання системи координат, щоб вона відповідала напряму польоту дрона

$$\begin{cases} x_{\text{obstacle}} = x' \cos(a) - y' \sin(a), \\ y_{\text{obstacle}} = x' \cos(a) + y' \sin(a), \end{cases} \quad (2)$$

де a - поточний кут курсу дрона, а $(x_{\text{obstacle}}, y_{\text{obstacle}})$ - координати перешкоди в його власній системі відліку.

Рис. 1. Симуляційне середовище

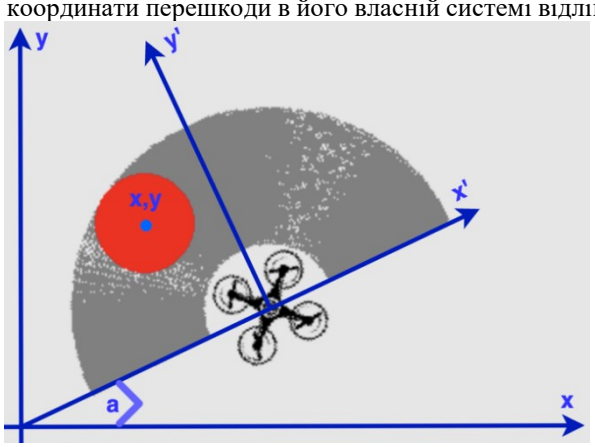

Рис. 2. Перетворення координат

Наступним етапом обробки сенсорних даних є нормалізація. Дрон порівнює відстані до перешкод із радіусом свого поля зору та враховує власний розмір для точного визначення близькості до об'єктів. Напрямні вектори, що показують положення перешкод відносно дрона, також піддаються масштабуванню для уніфікованого подання. Після нормалізації значення x-координат потрапляють до інтервалу від -1 до 1, а y-координат від 0 до 1. Щоб підвищити швидкодію й зосередитися на найважливіших об'єктах, дрон обробляє лише перші десять перешкод у полі зору. Такий вибірковий підхід сприяє оптимальному плануванню траєкторії та оперативному прийняттю рішень.

Перед початком симуляцій використовується заздалегідь підготовлений файл із координатами та радіусами перешкод. Це гарантує, що всі моделі тренуються в однакових умовах і з тими самими наборами об'єктів. Симуляційне середовище розгортається так, щоб дрон за замовчуванням опинявся в центрі. Для безпечного старту тренування навколо початкової позиції дрона виділяється безпечна зона певного радіусу у межах якої перешкоди не генеруються. Рух дрона в середовищі називається кроком. Епізод охоплює послідовність кроків від старту до моменту зіткнення з перешкодою. Після колізії дрона повертають у початковий центр, а з файлу завантажують новий набір перешкод і візуалізують його в арені. Цикл зіткнень і перезапуску складає основу процесу навчання, даючи змогу дрону відпрацьовувати різноманітні ситуації. Для експериментів моделі тренуються протягом одного мільйона кроків, що приблизно відповідає десяти годинам безперервної симуляції. Дослідження охоплює дві основні архітектури DQN і PPO. Кожну з них запускають чотири рази з різними функціями активації, щоб оцінити вплив нелінійних перетворень на динаміку навчання та кінцеву продуктивність моделей.

Простір дій для моделей нейронних мереж є дискретним і містить цілі значення від 0 до 99. Кожне значення дії складається з двох компонентів: кута повороту та відстані руху. Кут повороту обчислюється як ціла частина від ділення значення дії на 10 і задає напрям обертання дрона. Якщо цей кут не перевищує 5, дрон повертається вліво на вказану кількість градусів. Якщо кут більший за 5, дрон повертається вправо на різницю між 10 і значенням кута в градусах. Кут повороту визначається за формулою:

$$\begin{cases} angle = \begin{cases} action \div 10, & if\ action \div 10 \leq 5 \\ 10 - action \div 10, & if\ action \div 10 > 5 \end{cases}, \quad action \in [0; 99] \end{cases} \quad (3)$$

Відстань, на яку дрон переміщується вперед у пікселях, визначається залишком від ділення значення дії на 10, що дозволяє рухатися на відстань від 0 до 9 пікселів за крок. Відстань задається формулою:

$$distance = action \bmod 10, \quad action \in [0; 99] \quad (4)$$

Простір спостереження визначається масивом трійок, кожна з яких містить x, y та l. Тут x і y позначають нормалізований вектор напрямку від дрона до перешкоди, а l вказує нормалізовану відстань до цієї перешкоди. Така організація надає нейронній мережі повне уявлення про середовище, детально описуючи відносні положення та відстані до десяти найближчих перешкод біля дрона. Простір спостереження визначається формулою:

$$space = x_1, y_1, l_1, \dots x_i, y_i, l_i, \dots x_n, y_n, l_n, \quad x_i \in [-1; 1], y_i \in [0; 1], l_i \in [0; 1], n = 10 \tag{5}$$

Функція винагороди відіграє ключову роль у спрямуванні процесу навчання нейронних мереж. Спочатку передбачалося нараховувати штраф −10 за зіткнення і невелику позитивну винагороду +0,01 помножену на l за кожен крок без аварії, де l позначає відстань, яку дрон пройшов у пікселях. Проте експериментальні випробування виявили дві проблеми: дрон іноді залишався нерухомим і проявляв «тремтіння» - швидкі коливальні повороти поблизу перешкод. Ці нюанси вказували на необхідність удосконалення структури винагород, щоб заохотити більш бажані стратегії, як-от плавну навігацію та уникнення зайвих рухів або зволікань біля об'єктів. Щоб усунути проблему стояння на місці та надмірних поворотів, до системи винагород додано невеликі штрафи. Дрон отримує −0,01, якщо не рухається вперед, і додатково −0,01 за кожен виконаний поворот. Таке регулювання стимулює безперервний рух із мінімальною кількістю поворотів, змінюючи курс лише тоді, коли це необхідно для уникнення зіткнень. Остаточна функція винагороди визначається як:

$$reward = \begin{cases} -10, & if\ \exists i: l_i \le 0\ (collision) \\ -0.01, & if\ distance = 0\ or\ angle \neq 0 \\ 0.01 * distance, & otherwise. \end{cases} \tag{6}$$

В результати запуску симуляцій збирається комплексний набір показників, що дозволяє оцінити продуктивність моделей нейронних мереж і динаміку взаємодії дрона із середовищем. Розглянемо ці показники докладніше:

- *Середня кількість кроків в одному епізоді.* Відображає середнє число дій, виконаних протягом епізоду, і підкреслює ефективність дрона під час навігації. Ця метрика показує, наскільки раціонально дрон використовує свої дії, щоб виживати у середовищі.
- *Частка повних зупинок*. Вимірює співвідношення кроків без руху до загальної кількості кроків в епізоді. Нижчий відсоток свідчить про більш безперервну та динамічну навігацію, усуваючи проблему «застигання» на місці.
- *Частка поворотів.* Кількісно визначає частоту дій повороту (вліво чи вправо) відносно загальної кількості кроків. Показник дає уявлення про адаптивність дрона й його схильність змінювати напрям у відповідь на перешкоди, водночас допомагає уникати надмірного «тремтіння» чи непотрібних обертів.
- *Середня пройдена відстань за епізод*. Відображає сумарну дистанцію (у пікселях), яку дрон долає протягом епізоду. Ця метрика напряму корелює зі здатністю дрона просуватися середовищем і слугує індикатором ефективного уникнення перешкод.
- *Середня швидкість руху за крок*. Обчислюється як середня відстань, подолана за одну дію. Дає розуміння балансу між швидкістю й обережністю, вищі значення засвідчують швидший прогрес до цілей виживання.
- *Середня нагорода за епізод*. Середнє значення накопиченої за епізод винагороди, що об'єднує всі аспекти поведінки дрона - від уникнення перешкод до підтримання руху та мінімізації зайвих дій.

Для DQN найкраще себе показала функція Sigmoid, що забезпечила найменший відсоток повних зупинок 0,626%, порівняно низький відсоток поворотів 59,983% і найвищу середню швидкість 4,984. Також рівень середньої винагороди є найбільшим -5,666. Для PPO найкращі результати належать Tanh із середньою винагородою 88,527, а також удвічі вищими середніми кроками в епізоді та середньою пройденою дистанцією в порівнянні з іншими функціями. ReLU та Leaky ReLU демонструють найменший відсоток повних зупинок, що вказує на безперервніший рух, тоді як Sigmoid поступається за більшістю метрик, крім середньої кількості кроків за епізод. Детальні значення всіх оціночних показників для DQN і PPO наведено у таблицях 1 та 2 відповідно.

Таблиця 1

Порівняння характеристик DQN

Функція активації	**Середня к-сть кроків за епізод**	**Частка повних зупинок**	**Частка поворотів**	**Середня дистанція за епізод**	**Середня швидкість**	**Середня нагорода за епізод**
Tanh	159.434	1.497	62.558	649.456	4.318	-6.982
Sigmoid	**211.367**	**0.626**	**59.983**	**797.7136**	**4.984**	**-5.666**
ReLU	205.576	3.881	67.722	761.555	3.993	-7.616
Leaky ReLU	209.794	2.216	67.46	772.307	4.103	-7.406

Таблиця 2

Порівняння характеристик PPO

Функція активації	**Середня к-сть кроків за епізод**	**Частка повних зупинок**	**Частка поворотів**	**Середня дистанція за епізод**	**Середня швидкість**	**Середня нагорода за епізод**
Tanh	**2727.577**	0.13	**50.839**	**12781.262**	4.2	**88.527**
Sigmoid	1538.434	0.149	60.723	5535.878	3.428	31.117
ReLU	1730.587	**0.074**	51.748	8023.028	4.12	43.598
Leaky ReLU	1444.961	0.088	51.174	6546.359	**4.269**	33.8

Порівняння моделей показало, що PPO беззаперечно перевершує DQN за всіма показниками, навіть найслабші результати PPO випереджають найкращі показники DQN, що свідчить про кращу здатність до навігації й уникнення перешкод. Аналіз також виявив, що обмежені функції активації (Sigmoid і Tanh) загалом дають кращі результати, ніж необмежені (ReLU та Leaky ReLU). Водночас у PPO Sigmoid поступилася ReLU та Leaky ReLU, що вказує на те, що вибір оптимальної функції залежить від конкретного алгоритму навчання та характеру завдання - універсального рішення немає.

Подальші спостереження навігаційних стратегій у середовищах із великою кількістю перешкод виявили суттєві відмінності у підходах. DQN стабільно намагалася проходити вузькими коридорами між об'єктами, підвищуючи ризик зіткнень. PPO, навпаки, віддає перевагу обминанню перешкод і руху до відкритих ділянок. Така поведінка проявлялася як у середовищах із перешкодами, так і без них, DQN маневрувала надто близько до об'єктів, що призводило до нижчої середньої винагороди, тоді як PPO обирала безпечніші маршрути й демонструвала вищу ефективність.

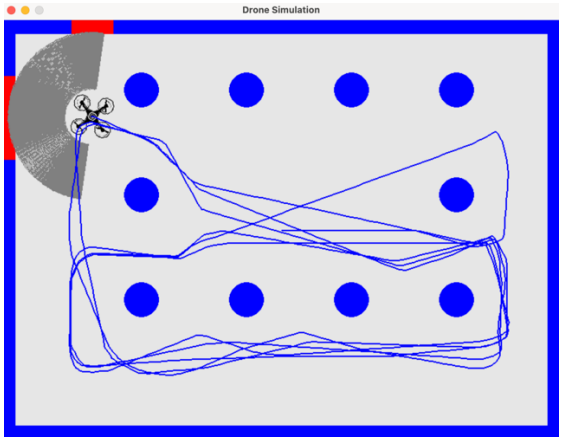

Рис. 3. Стратегія навігації DQN

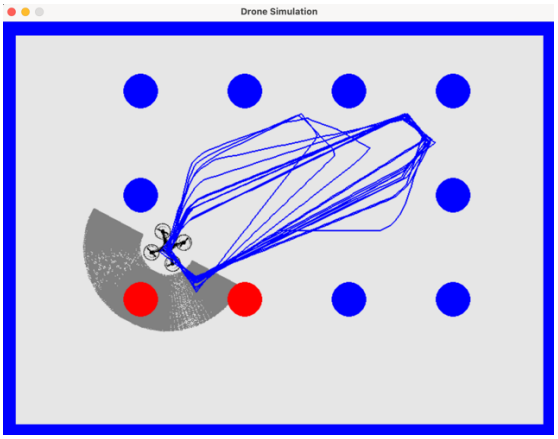

Рис. 4. Стратегія навігації PPO

Траєкторії навігації моделей DQN і PPO наведено на рис. 3 та 4 відповідно.

Отже, результати підкреслюють різні уподобання у навігації та керуванні ризиками в моделях DQN і PPO для обходу перешкод БПЛА, причому PPO демонструє перевагу за всіма основними показниками. Вибір функції активації суттєво впливає на поведінку та продуктивність моделі, але жодна функція не є однозначно найкращою.

Висновки

У цьому дослідженні було порівняно алгоритми DQN та PPO для оцінки їхньої ефективності в задачі уникнення перешкод БПЛА у двовимірній симуляції. Результати показали, що PPO перевершує DQN у складних середовищах завдяки ефективнішим оновленням політики. Крім того, обмежені функції активації, такі як Sigmoid і Tanh, виявилися результативнішими за необмежені, що підкреслює важливість правильного вибору функції активації для підвищення продуктивності моделі. Аналіз продемонстрував, що дрони під керуванням DQN ефективно маневрують у вузьких просторах, використовуючи стратегії, орієнтовані на миттєву винагороду, тоді як дрони PPO віддають перевагу відкритим ділянкам, обираючи безпечніші траєкторії завдяки обережнішим оновленням політики. Ця відмінність підкреслює необхідність узгодження алгоритму з операційними потребами БПЛА та слугує орієнтиром для проєктування систем під конкретні навігаційні виклики.

З урахуванням отриманих результатів, попри загальну перевагу PPO у складних середовищах, здатність DQN маневрувати у тісних просторах робить його переконливим кандидатом для задач із вузькими коридорами. Такий результати спонукають до збалансованого вибору алгоритмів і функцій активації відповідно до профілю місії БПЛА. Подальші дослідження мають розширити застосування алгоритмів, інтегрувати різноманітні типи вхідних даних і перевірити ці висновки у реалістичних умовах, щоб удосконалити навігаційні можливості безпілотників.

Література

1. Mohsan S. A. H.; Othman N. Q. H.; Li Y.; Alsharif M. H.; Khan M. A. Unmanned aerial vehicles (UAVs): practical aspects, applications, open challenges, security issues, and future trends // Intelligent Service Robotics. – 2023. – Vol. 16. – P. 109–137. – DOI: 10.1007/s11370-022-00452-4.

2. Arulkumaran K.; Deisenroth M. P.; Brundage M.; Bharath A. A. Deep reinforcement learning: a brief survey // IEEE Signal Processing Magazine. – 2017. – Vol. 34. – P. 26–38. – DOI: 10.1109/MSP.2017.2743240.

3. Mnih V.; Kavukcuoglu K.; Silver D. та ін. Playing Atari with deep reinforcement learning [Електронний ресурс] // arXiv preprint, arXiv:1312.5602. – 2013. – URL: http://arxiv.org/abs/1312.5602 (дата звернення: 11.05.2025).

4. Schulman J.; Wolski F.; Dhariwal P.; Radford A.; Klimov O. Proximal policy optimization algorithms

[Електронний ресурс]. – 2017. – URL: http://arxiv.org/abs/1707.06347 (дата звернення: 11.05.2025).

5. Reuf K.; Stefan W.; Simon H. Deep Q-learning versus proximal policy optimization: performance comparison in a material sorting task // Proc. 2023 IEEE Int. Symp. on Industrial Electronics (ISIE). – 2023. – P. 1–6. – DOI: 10.1109/ISIE51358.2023.10228056.

6. Kalidas A. P.; Joshua C. J.; Md A. Q.; Basheer S.; Mohan S.; Sakri S. Deep reinforcement learning for vision-based navigation of UAVs in avoiding stationary and mobile obstacles // Drones. – 2023. – Vol. 7, № 4. – Art. 245. – DOI: 10.3390/drones7040245.

7. Haarnoja T.; Zhou A.; Hartikainen K. та ін. Soft actor-critic algorithms and applications [Електронний ресурс]. – 2019. – arXiv:1812.05905. – URL: https://arxiv.org/abs/1812.05905 (дата звернення: 11.05.2025).

8. Jang B.; Kim M.; Harerimana G.; Kim J. W. Q-learning algorithms: a comprehensive classification and applications // IEEE Access. – 2019. – Vol. 7. – P. 133653–133667. – DOI: 10.1109/ACCESS.2019.2941229.

9. Zhao D.; Wang H.; Shao K.; Zhu Y. Deep reinforcement learning with experience replay based on SARSA // Proc. 2016 IEEE Symposium Series on Computational Intelligence (SSCI). – 2016. – P. 1–6. – DOI: 10.1109/SSCI.2016.7849837.

10. Mnih V.; Kavukcuoglu K.; Silver D. та ін. Human-level control through deep reinforcement learning // Nature. – 2015. – Vol. 518. – P. 529–533. – DOI: 10.1038/nature14236.

11. Szandała T. Review and comparison of commonly used activation functions for deep neural networks // Lecture Notes in Networks and Systems. – 2021. – P. 203–224. – DOI: 10.1007/978-981-15-5495-7_11.

12. Pygame [Електронний ресурс]. – 2024. – URL: https://www.pygame.org/ (дата звернення: 11.05.2025).

13. Stable-baselines3 [Електронний ресурс]. – 2024. – URL: https://stable-baselines3.readthedocs.io/ (дата звернення: 11.05.2025).

14. Raffin A.; Hill A.; Gleave A.; Kanervisto A.; Ernestus M.; Dormann N. Stable-baselines3: reliable reinforcement learning implementations // Journal of Machine Learning Research. – 2021. – Vol. 22, Art. 268. – P. 1–8. – URL: http://jmlr.org/papers/v22/20-1364.html (дата звернення: 11.05.2025).

15. Coordinates and transformations [Електронний ресурс]. – 2024. – URL: https://motion.cs.illinois.edu/RoboticSystems/CoordinateTransformations.html (дата звернення: 11.05.2025).

References

1. Mohsan S. A. H.; Othman N. Q. H.; Li Y.; Alsharif M. H.; Khan M. A. Unmanned aerial vehicles (UAVs): practical aspects, applications, open challenges, security issues, and future trends // Intelligent Service Robotics. – 2023. – Vol. 16. – P. 109–137. – DOI: 10.1007/s11370-022-00452-4.

2. Arulkumaran K.; Deisenroth M. P.; Brundage M.; Bharath A. A. Deep reinforcement learning: a brief survey // IEEE Signal Processing Magazine. – 2017. – Vol. 34. – P. 26–38. – DOI: 10.1109/MSP.2017.2743240.

3. Mnih V.; Kavukcuoglu K.; Silver D. et al. Playing Atari with deep reinforcement learning [Electronic resource] // arXiv preprint. – 2013. – URL: http://arxiv.org/abs/1312.5602 (accessed 11.05.2025).

4. Schulman J.; Wolski F.; Dhariwal P.; Radford A.; Klimov O. Proximal policy optimization algorithms [Electronic resource]. – 2017. – URL: http://arxiv.org/abs/1707.06347 (accessed 11.05.2025).

5. Reuf K.; Stefan W.; Simon H. Deep Q-learning versus proximal policy optimization: performance comparison in a material sorting task // Proc. IEEE Int. Symp. on Industrial Electronics (ISIE). – 2023. – P. 1–6. – DOI: 10.1109/ISIE51358.2023.10228056.

6. Kalidas A. P.; Joshua C. J.; Md A. Q.; Basheer S.; Mohan S.; Sakri S. Deep reinforcement learning for vision-based navigation of UAVs in avoiding stationary and mobile obstacles // Drones. – 2023. – Vol. 7, No. 4. – Art. 245. – DOI: 10.3390/drones7040245.

7. Haarnoja T.; Zhou A.; Hartikainen K. et al. Soft actor-critic algorithms and applications [Electronic resource]. – 2019. – arXiv:1812.05905. – URL: https://arxiv.org/abs/1812.05905 (accessed 11.05.2025).

8. Jang B.; Kim M.; Harerimana G.; Kim J. W. Q-learning algorithms: a comprehensive classification and applications // IEEE Access. – 2019. – Vol. 7. – P. 133653–133667. – DOI: 10.1109/ACCESS.2019.2941229.

9. Zhao D.; Wang H.; Shao K.; Zhu Y. Deep reinforcement learning with experience replay based on SARSA // Proc. IEEE Symposium Series on Computational Intelligence (SSCI). – 2016. – P. 1–6. – DOI: 10.1109/SSCI.2016.7849837.

10. Mnih V.; Kavukcuoglu K.; Silver D. et al. Human-level control through deep reinforcement learning // Nature. – 2015. – Vol. 518. – P. 529–533. – DOI: 10.1038/nature14236.

11. Szandała T. Review and comparison of commonly used activation functions for deep neural networks // Lecture Notes in Networks and Systems. – 2021. – P. 203–224. – DOI: 10.1007/978-981-15-5495-7_11.

12. Pygame [Electronic resource]. – 2024. – URL: https://www.pygame.org/ (accessed 11.05.2025).

13. Stable-baselines3 [Electronic resource]. – 2024. – URL: https://stable-baselines3.readthedocs.io/ (accessed 11.05.2025).

14. Raffin A.; Hill A.; Gleave A.; Kanervisto A.; Ernestus M.; Dormann N. Stable-baselines3: reliable reinforcement learning implementations // Journal of Machine Learning Research. – 2021. – Vol. 22, Art. 268. – P. 1–8. – URL: http://jmlr.org/papers/v22/20-1364.html (accessed 11.05.2025).

15. Coordinates and transformations [Electronic resource]. – 2024. – URL: https://motion.cs.illinois.edu/RoboticSystems/CoordinateTransformations.html (accessed 11.05.2025).