https://doi.org/10.31891/2219-9365-2026-87-54
УДК 004.67, 004.38

ІВАНКОВ Валентин
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
https://orcid.org/0009-0001-9835-8486
valentyn.ivankov@gmail.com
НОВОТАРСЬКИЙ Михайло
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
https://orcid.org/0000-0002-5653-8518
novotar@gmail.com

АДАПТИВНИЙ МЕТОД НАВІГАЦІЇ РОЮ БПЛА

У статті досліджується проблема безпечної навігації рою БПЛА у середовищі з перешкодами, наприклад, будівлями чи рельєфом місцевості. Запропонований адаптивний підхід поєднує високорівневий контролер на основі мережі Петрі та політику руху на основі навчання з підкріпленням. У рамках дослідження було розглянуто застосування методу навігації на основі Proximal Policy Optimization, який забезпечує рух у межах кожного сегмента маршруту, спираючись на результати сканування місцевості лідаром. Натомість керуюча мережа на основі високорівневих мереж Петрі зберігає повноваження щодо контролю послідовності етапів завдань, розподілу ресурсів та аварійного повернення в разі несправності. Структура мережі відносно попередньо розробленої моделі не змінюється, тому встановлені раніше властивості зберігаються, а умови переходів обчислюються за положенням БПЛА у просторі. Проведено аналіз поширених методів навчання, які використовуються для подібних задач, а також виконано навчання та моделювання в середовищі ROS 2/Gazebo для рою з 10 БПЛА. У дослідженні використовувалися два середовища, згенеровані на основі реальних даних про висоти та забудову. Навчання політики проводилося у середовищі зі складнішим рельєфом та щільнішою забудовою. Ефективність методу після перенесення було оцінено за двома критеріями: надійністю навігації та здатністю до узагальнення. Результат навченої політики становив 90 % досягнень цілі та 5 % зіткнень у раніше невідомому тестовому середовищі, і повний цикл завдання було успішно виконано.

Ключові слова: мережі Петрі високого рівня, дрони, навчання з підкріпленням, обхід перешкод, адаптивна навігація.

IVANKOV Valentyn, NOVOTARSKYI Mykhailo
National Technical University of Ukraine “Ihor Sikorsky Kyiv Polytechnic Institute”

ADAPTIVE NAVIGATION METHOD FOR UAV SWARM

The presented approach focuses on one of the fastest-growing areas of modern robotics: adaptive and autonomous navigation of swarms of unmanned aerial vehicles (UAVs) in environments with obstacles. The main goal of this research is to create an efficient approach that allows to conduct a task fully autonomously while avoiding obstacles such as terrain or buildings.

The developed method basically combines two different control mechanisms: a discrete high-level Petri net, which supervises sequencing of mission stages, resource arbitration, and emergency return, and a reinforcement learning policy, which performs movement within each route segment based on lidar scan results. The structure of the net remains unchanged relative to the previously developed model, so previously established properties are still valid, while the evaluation of transition guards is now performed based on the measured position of the UAVs. The study includes an analysis of reinforcement learning methods applied to similar tasks, including value-based, on-policy, off-policy, and model-based families. As a result, the Proximal Policy Optimization algorithm was the best fit for the task.

Training and modeling were performed in the ROS 2 and Gazebo environment for a swarm of ten UAVs on two terrains generated from real-world data. The area for training had more buildings and more complex terrain compared to the test area. The efficiency of the proposed approach was calculated based on two primary criteria: navigation reliability and generalization capability. The findings demonstrate that a policy trained exclusively on one complex terrain achieves a goal-reach rate of 90% and a collision rate of only 5% on previously unseen terrain, and moreover completes the full mission cycle there. The developed method and simulation provide a solid base for future research on hybrids of formal and learning-based control in multi-UAV systems.

Keywords: High-level Petri nets, drones, reinforcement learning, obstacle avoidance, adaptive navigation.

Стаття надійшла до редакції / Received 01.06.2026
Прийнята до друку / Accepted 16.07.2026
Опубліковано / Published 10.09.2026

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ’ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Рої БПЛА потрібні насамперед у тих випадках, коли територію слід обстежити швидко або коли політ у цій зоні небезпечний для людини. Один апарат обмежений запасом енергії та радіусом дії. У групі таку роботу можна розподілити, а місію — продовжити після відмови окремого учасника. Разом із цим з’являється інша задача: узгодити стан кожного БПЛА, спільні ресурси і рух у просторі з перешкодами.

Під час перенесення раніше розробленої моделі [1] до такого середовища виявилося, що правильна послідовність команд ще не забезпечує виконання маршруту. Мережа Петрі контролювала дискретні параметри, тобто етапи польоту, заряд акумуляторів, зарядні ресурси та включення аварійного повернення за

необхідності. Рух між заданими точками вона вважала успішним за замовчуванням. Такого спрощення достатньо, якщо маршрут пролягає на відкритому просторі або на великій висоті, але в щільній забудові прямий відрізок маршруту може проходити через будівлю або надто близько до поверхні.

Додавання даних лідара до стану мережі означає істотне ускладнення управляючої моделі. Її структура в цій роботі не змінюється: мережа передає проміжну ціль контролеру на основі Proximal Policy Optimization (PPO), а той обирає напрямок і швидкість за поточними відстанями до перешкод. Після завершення руху мережа очікує сигнал про прибуття. В цьому місці на практиці виникає практична проблема. БПЛА здатен увійти в допустиму область, пролетіти повз ціль і повернутися; за пізньої перевірки цей момент буде втрачено, і місія зависне в стані очікування допоки не завершиться аварійно.

Мета роботи — розробити адаптивний метод навігації рою БПЛА з розподілом функцій, необхідних для завершення завдання між мережею Петрі та навченою політикою. Під час моделювання завдання в середовищі ROS2 + Gazebo потрібно узгодити предикат прибуття з точністю, отриманою під час навчання, а потім перевірити перенесення, тобто застосувати політику руху на іншій місцевості (яка була невідома).

ПОСТАНОВКА ПРОБЛЕМИ У ЗАГАЛЬНОМУ ВИГЛЯДІ ТА ЇЇ ЗВ'ЯЗОК ІЗ ВАЖЛИВИМИ НАУКОВИМИ ЧИ ПРАКТИЧНИМИ ЗАВДАННЯМИ

Логіка управління роєм БПЛА має задовольняти дві вимоги, які вирішуються різними способами. Перше — це правильність послідовності етапів завдання. Наприклад, БПЛА не мають злітати з недостатнім зарядом, мають пройти всі заплановані етапи або ж повинні повернутися назад, якщо стан більше не дозволяє продовжувати рух. Друге — це коректність руху: між будь-якими двома точками маршруту БПЛА має фізично прибути в правильну точку у середовищі з перешкодами, такими, як рельєф, будівлі тощо.

У попередній роботі було розроблено метод управління на основі мережі Петрі високого рівня (HLPN) та виконано моделювання завдання [1]. Друга вимога не була врахована цією моделлю, оскільки навігація відбувалася за заздалегідь розрахованим маршрутом. Такий метод підходить лише для вільного повітряного простору, де немає перешкод. У середовищі з перешкодами, наприклад, в урбанізованих районах, сегмент маршруту, що з'єднує дві точки маршруту, може проходити напряму через будівлю. Така помилка не може бути виявлена мережею, оскільки перешкоди не є частиною стану токена.

Розширення мережі з метою інтеграції перешкод не є оптимальним, оскільки рух є неперервним, а інформація про перешкоди доступна лише під час роботи через показники датчиків, наприклад, лідара. Включення перешкод як станів або перевірок можливості переходу призводить до надмірного збільшення моделі, що ускладнює її підтримку та подальшу розробку

У цій роботі пропонується адаптивний метод навігації, який поєднує розроблену модель управління на основі мереж Петрі та алгоритм Proximal Policy Optimization, що визначає напрямок і швидкість БПЛА на основі даних лідара. Комбінований підхід дозволяє не лише забезпечити коректну послідовність всіх етапів завдання, а й врахувати геометрію навколишнього середовища, яка стає відома вже безпосередньо під час польоту. В умовах щільної забудови або складного рельєфу це забезпечує уникнення перешкод, невідомих на етапі планування маршруту, із збереженням гарантій мережі Петрі, що безпосередньо впливає на безпеку та автономність виконання завдань роєм.

АНАЛІЗ ДОСЛІДЖЕНЬ ТА ПУБЛІКАЦІЙ

Мережі Петрі є популярним інструментом для опису дискретної поведінки безпілотних систем. Кольорові мережі Петрі можуть застосовуватися для інспекцій БПЛА, де токени містять дані про заряд акумулятора, а модель фіксує цикли польоту, перезарядки та технічного обслуговування [2]. Такий самий формалізм лежить в основі інструментів, які автоматично генерують мережі з певного початкового набору навичок, щоб перевірити, що система не зможе перейти до неправильного стану [3]. Стохастичні мережі Петрі застосовуються для оцінки та покращення автономності польоту в критичних сценаріях, де цільовий показник поєднує ефективність із відмовами та відновленнями в програмних процесах [4]. Публікації такого роду, включаючи модель, на якій базується дана стаття [1], мають спільне обмеження: управляюча мережа лише описує, в якому стані перебуває БПЛА і за яких умов він може переходити в інші стани. Сам рух при цьому або абстрагується, або автоматично вважається успішним.

Цю проблему можна вирішити за допомогою навчання з підкріпленням. Алгоритм Proximal Policy Optimization від OpenAI [5] став стандартним методом для неперервного управління, і його застосування в аеронавігації добре проаналізовано. Політика, навчена в симуляції та перенесена на реальне апаратне забезпечення, була досліджена як потенційний засіб уникнення перешкод в умовах підвищеного "шуму" датчиків, при цьому навмисне введення шуму та шумогасіння досліджувалися як заходи підвищення стійкості [6]. Навігація в щільному та високодинамічному середовищі за часткової спостережуваності була досліджена в роботі [7]. Також дещо схожі результати можуть бути отримані за допомогою методів комп'ютерного зору, а не за визначених відстаней до перешкод, хоча в даній роботі критик використовував "привілейовану інформацію", тобто таку, що доступна лише під час навчання [8]. У подібних дослідженнях навчена політика є контролером, тобто над нею немає дискретного шару і немає можливості дізнатися статус завдання. Спроби

поєднати мережі Петрі з навчанням з підкріпленням з'явилися відносно недавно і за межами БПЛА. Мережі Петрі використовуються як інтегрований механізм обмежень для навчання з підкріпленням, де мережа надає комбіноване представлення стану і застосовує обмеження до агента, а також проводить перевірку моделі, що продемонстровано на прикладі керування світлофорами [9]. У виробництві фреймворк PetriRL доручає мережі керування автоматизованими компонентами, а політика ухвалює рішення лише в точках вибору й лише серед дозволених мережею дій [10]. Часова мережа Петрі може використовуватися як навчальне середовище, де графова згорткова мережа зчитує топологію мережі, щоб політика могла витримувати структурні зміни, спричинені відмовами обладнання або новими замовленнями [11]. Ці результати підтверджують, що поєднання є доцільним, але досліджені домени є дискретними виробничими системами, а не системами з неперервним рухом БПЛА.

Окремим напрямом є безпечне навчання з підкріпленням через екранування, де синтезований зі специфікації темпоральної логіки «щит» на кожному кроці перевіряє запропоновану агентом дію та, за потреби, підміняє її безпечною [13]. Такий супервізор обмежує окремі дії, але не очікує завершення фізичного руху як цілісного етапу: агент зберігає керування на кожному кроці, тому критерій прибуття, від якого залежить просування дискретної моделі до наступного стану, у цій постановці не виникає.

Загалом, питання про те, чи залишається навчена політика навігації компетентною в середовищі, з яким вона не стикалася, належить до класу узагальнень без попереднього навчання на задачі і було досліджене в роботі [12]. Це дослідження відрізняє перенавчання в тестовому середовищі від справжнього перенесення та забезпечує фреймворк, у якому оцінюється політика у середовищі, досі невідомому.

Виходить, що поточні дослідження демонструють скоріше два окремі напрямки робіт — формальні моделі, які описують місії без опису руху, та навчені контролери, які забезпечують рух, але без врахування завдання. І також існує невелика кількість гібридних конструкцій, але вони обмежені дискретними подіями.

ВИДІЛЕННЯ НЕДОСЛІДЖЕНИХ ЧАСТИН ЗАГАЛЬНОЇ ПРОБЛЕМИ

Формальні моделі завдань [1]–[4] описують лише переходи між етапами завдання, але не фізичний шлях між ними, і тому не можуть включати процес уникнення перешкод як стан моделі, оскільки це надто ускладнює розробку та підтримку.

Навчені політики [6]–[8] надають таку можливість, але вони вирішують лише питання коректності переміщення, не враховуючи логіку завдання. Тобто не можна сказати, що політика ніколи не порушує задані обмеження, а лише показати, що вона поводиться прийнятно на множині, відносно якої її оцінюють. Включення обмежень завдання у функцію винагороди перетворює жорсткі вимоги на «вподобання», тобто обмеження може бути порушено, якщо штраф менший за потенційну винагороду.

Отже, інтерфейс між дискретним контролером і безперервним середовищем наразі недостатньо вивчено. Існуючі комбінації формальних і навчених підходів [9]–[11] працюють в області дискретних подій, де навчена політика вибирає підходящу дію серед наперед перелічених рішень, а не підтримує безперервну навігацію. У таких системах немає фізичного руху, який потрібно завершити, щоб дискретна модель могла перейти в наступний стан; саме тому критерій прибуття там не потрібен. А у випадку з екрануванням [13] супервізор обмежує кожну окрему дію, не очікуючи завершення руху як цілісного етапу, тому критерій прибуття там відсутній. Тому в системах, де відбувається рух у фізичному просторі, такий критерій може бути ускладнений, оскільки контролер, що використовує навчену політику, може не прилітати прямо в потрібну точку.

Також залишається недослідженим питання, чи може політика руху бути застосована дискретним контролером у середовищі, з яким ця політика досі не стикалася. Узагальнення за нульової кількості прикладів вивчається як властивість політики у роботі [12] і просто вимірюється певним коефіцієнтом успіху, а не як умова, від якої залежить дискретний контролер, що знаходиться вище за неї. Тобто не зрозуміло, чи зберігає політика певну компетенцію, наприклад, точного завершення подання команди на рух з метою переходу управляючого контролера до наступного стану.

ФОРМУЛЮВАННЯ ЦІЛЕЙ СТАТТІ

Об'єктом дослідження є процес керування роєм БПЛА в середовищі з перешкодами, де логіка завдання вимагає формальних гарантій, а неперервний рух має правильно реагувати на зміни в місцевості.

Метою дослідження є розробка та валідація гібридної архітектури керування роєм БПЛА. В розробленому методі HLPN має зберігати повний контроль над послідовністю етапів виконання завдання, а контролер руху, побудований на базі навчання з підкріпленням, має виконувати фізичний рух між точками маршруту. Також необхідно знайти умови, за яких така комбінація залишається стабільною, особливо у середовищах, які не використовувалися під час навчання політики руху.

ВИКЛАД ОСНОВНОГО МАТЕРІАЛУ

Керуюча мережа збігається з розробленою в попередній роботі [1]: сім станів, що відповідають етапам польоту (Ready, Takeoff, EnRoute, AtTarget, Returning, OnBase, Charging), спільний ресурс, який обмежує кількість одночасних зарядок, та десять переходів з умовами. Усі БПЛА рухаються одночасно: на

кожному кроці спрацьовують усі дозволені переходи (рис. 1). Структура мережі в цій роботі не змінюється — жодну позицію чи перехід не додано й не вилучено, а умови переходів не переписані, — тому встановлені в роботі [1] властивості досяжності залишаються дійсними. Змінюються два аспекти: звідки умови переходів отримують свої аргументи та що відбувається між послідовними спрацюваннями.

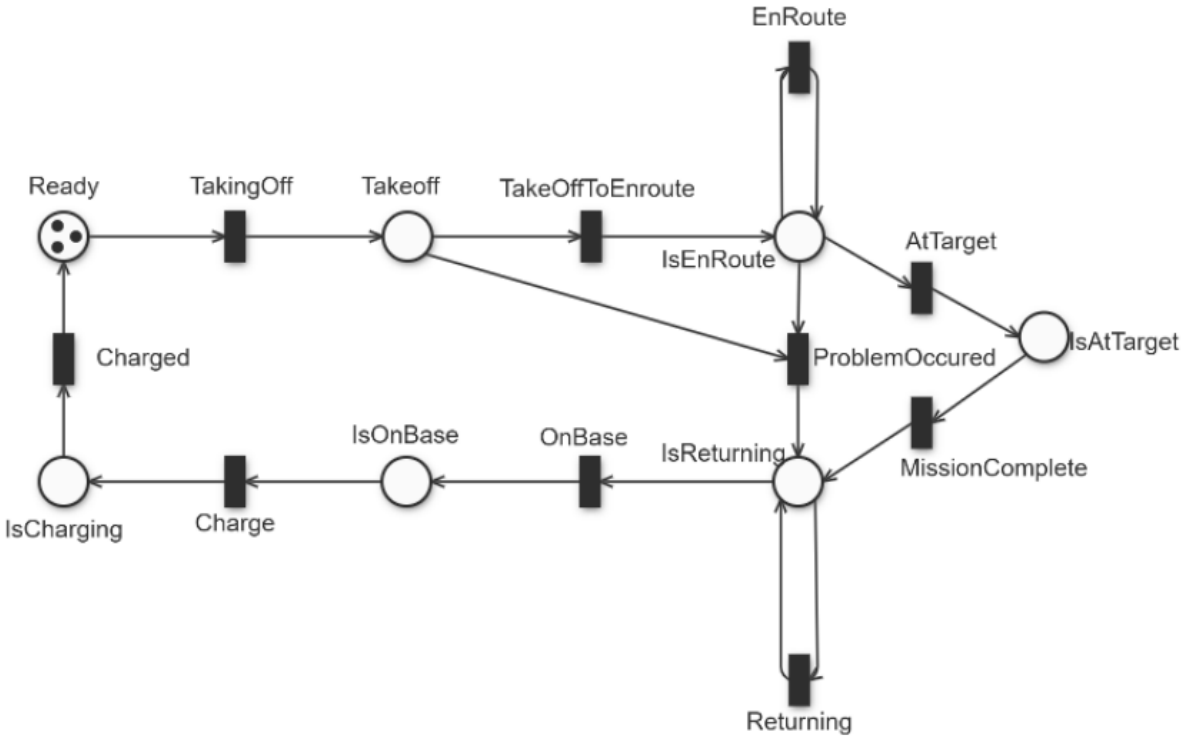

Рис. 1. Структура керуючої мережі

Відмінність полягає в тому, що в гібридному методі рух делегований, а не відбувається за замовчуванням. В попередній роботі [1] просування за маршрутом і прибуття в точку були однією операцією, оскільки модель сама переміщувала БПЛА до цільової точки. Зараз точка маршруту зберігається як проміжна ціль і передається навченій політиці руху; управляюча мережа лише дає команду на рух і очікує результат, чого раніше не було. Відповідно, прибуття є предикатом. Умови *IsAtTarget* та *IsAtBase* обчислюються за положенням БПЛА у просторі, тобто перевіряється саме фізичне положення, а не очікування мережі. Ці зміни роблять дискретний рівень (рівень мережі) більш чутливим до результату руху, який він більше не виконує самостійно. Логіка завдання залишається фіксованою, і її легко перевірити. Отриманий інтерфейс показано на рис. 2.

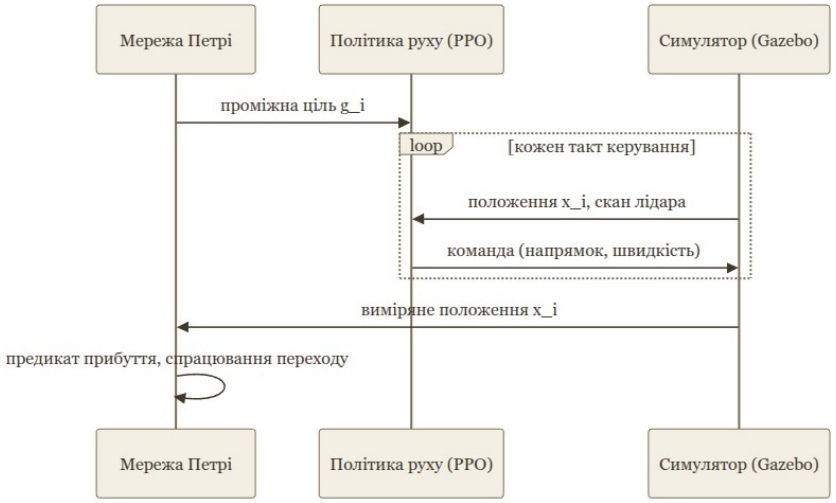

Рис. 2. Інтерфейс між дискретним (мережа) та неперервним (БПЛА в русі) рівнями

Нехай маршрут БПЛА складається з точок *wi,1, ..., wi,K*, а активна проміжна ціль — *gi = wi,k*. Політика руху зменшує відстань $\|x_i - g_i\|$, і коли виконується предикат прибуття, мережа збільшує *k*. Спостереження є вектором розмірності $4 + N_b$:

$$o_i = \left[\frac{g_i - x_i}{d_{norm}}, min\left(1, \frac{\|g_i - x_i\|}{d_{norm}}\right), \frac{r_{i,1}}{r_{max}}, \cdots, \frac{r_{i,N_b}}{r_{max}}\right] \quad (1)$$

де d_{norm} — нормувальна відстань, r_{max} — максимальна дальність огляду датчика, $r_{i,j}$ — найближча перешкода в секторі *j*. Область спостереження з 360 × 32 точок зводиться до N_b кутових секторів за мінімальною тривимірною відстанню в кожному:

$$r_{i,j} = \min_{p \in W_j} \|p\|,$$
$$W_j = \{p : \phi(p) \in [(j-1)\Delta, j\Delta), \delta_{\text{self}} < \|p\| < r_{\max}\} \quad (2)$$

де *φ(p)* — горизонтальний азимут точки, поправка $\Delta = 2\pi / N_b$, *а* δ_{self} використовується з метою відкидання відбиття сигналу датчику від корпусу БПЛА. Згортка вертикальних кілець у сектор дозволяє реєструвати поверхню та об'єкти над БПЛА як перешкоди. Схему спостереження показано на рис. 3. Вона не містить даних про абсолютне положення БПЛА у просторі, усіх даних про середовище, та даних про завдання, тобто політика не має «карти» яку було б можливо закодувати — ця властивість надалі перевіряється в експерименті.

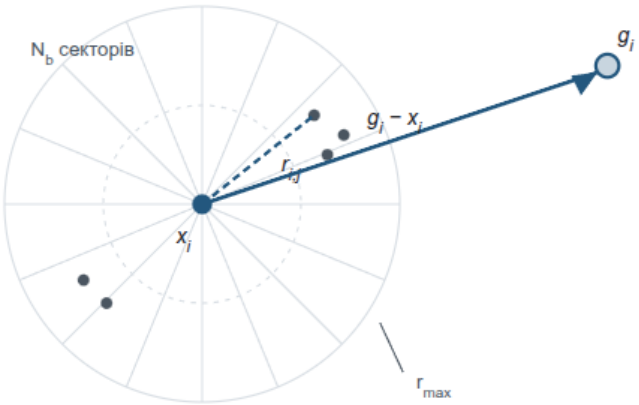

Рис. 3 Схема спостереження

Дія політики складається з напрямку і нормованої швидкості: $a_i = [\hat{d}_x, \hat{d}_y, \hat{d}_z, v]$, де $\hat{d} \in [-1,1]^3$, $v \in [0,1]$. Модуль $\hat{d}$, тобто довжина, відкидається, оскільки тут важливим є лише напрямок. Отже, командна точка задається наступною формулою:

$$c_i = x_i + \frac{\hat{d}}{\|\hat{d}\|} \cdot L, v_{\text{cmd}} = v \cdot v_{\max} \quad (3)$$

Поставлена задача ставить три вимоги до алгоритму навчання: неперервний простір дій, стійкість навчання за зашумлених даних та можливість паралельного навчання (використання декількох БПЛА в одному симуляторі). Відповідно до цих вимог було розглянуто чотири сімейства алгоритмів навчання з підкріпленням.

Методи на основі функції цінності (DQN) працюють із дискретним простором дій; дискретизація напрямку та швидкості за прийнятної роздільності дає комбінаторно великий простір, тому це сімейство було одразу відкинуте. Off-policy методи актор-критик з неперервними діями (DDPG, TD3, SAC) ефективніші за кількістю взаємодій завдяки буферу відтворення, але чутливі до гіперпараметрів, а відтворення застарілих переходів проблематично в ситуаціях коли середовище містить інші БПЛА того самого рою, і тому нестаціонарне з погляду окремого агента. Методи на основі моделі середовища вимагають навчити передбачення показів лідара на невідомому середовищі — задачу, не простішу за саму навігацію, з накопиченням помилки моделі вздовж горизонту. On-policy градієнт політики в реалізації Proximal Policy Optimization [5] задовольняє всі три вимоги: неперервні дії, обмеження кроку оновлення, що робить навчання стабільним без тонкого налаштування, і природна векторизація збору досвіду. Також, його застосування саме до аеронавігації добре підтверджене в публікаціях [6]–[8]. Саме тому за основу взято алгоритм PPO.

Навчена політика не приводить БПЛА до конкретної точки. Вони скоріше наближаються до проміжної цілі, входять у зону допуску, пролітають повз неї і повертаються. Дане кружляння прямо випливає з команди фіксованої величини (3). Предикат прибуття може спрацювати в одному такті і не спрацьовувати вже на наступному. Оскільки мережа опрацьовує предикат у власному темпі, БПЛА, які вже прибули на точку,

можуть нескінченно не проходити перевірку, і токен ніколи не перейде далі по мережі (в наступний стан). В результаті було отримано наступний критерій прибуття:

$$arrived_i(t) = \left[\min_{t' \le t} \|x_i(t') - g_i\| < \tau_i\right] \quad (4)$$

Таблиця 1.

Порівняння сімей методів навчання з підкріпленням щодо вимог задачі

Сімейство	**Приклади**	**Неперервні дії**	**Ключове обмеження для задачі**
На основі функції цінності	DQN	ні	дискретизація напрямку і швидкості
Off-policy актор-критик	DDPG, TD3, SAC	так	чутливість до гіперпараметрів; застарілий досвід у нестаціонарному рої
На основі моделі	PETS, Dreamer	так	модель показів лідара складніша за саму політику
On-policy градієнт політики	PPO	так	нижча ефективність за кількістю взаємодій — компенсується векторизацією

Спрацювавши один раз, він залишається виконаним до отримання наступної проміжної цілі. Допуск масштабується відповідно до довжини сегмента, оскільки точки накопичуються біля кінців кривої, і фіксований допуск, що довший за короткий сегмент, виконувався б одразу:

$$\tau_i = clip\left(\alpha \cdot \|g_i - g_i^{prev}\|, \tau_{min}, \tau_{max}\right) \quad (5)$$

Нижня межа τ_{min} дорівнює радіусу цілі, використаному під час навчання: політика ніколи не оптимізувалася під жорсткіший допуск, ніж той, за який вона отримувала винагороду. Саме тут визначається умова стабільності комбінації: допуск предиката прибуття не може бути жорсткішим за точність, на якій політика руху була навчена.

Ця умова необхідна, але недостатня. Критерій прибуття передбачає, що кожна проміжна ціль фізично досяжна в межах допуску. Точка маршруту, розташована всередині перешкоди, блокує просування дискретного рівня: політика утримує БПЛА на безпечній відстані від перешкоди, предикат прибуття залишається хибним, і токен не змінює стан EnRoute. Аварійний перехід навмисно виведений з-під критерію прибуття, тому витрата заряду з часом активує повернення на базу — відмова деградує в аварійне повернення, а не в зіткнення.

Також, в управляючій мережі БПЛА знаходяться на початку координат, тоді як в симуляторі вони розміщуються де завгодно в середовищі. Системи координат відрізняються сталим зсувом, який застосовується в обох напрямках при кожному обміні:

$$g_i^{\text{sim}} = g_i^{\text{net}} + \Delta_i, x_i^{\text{net}} = x_i^{\text{sim}} - \Delta_i \quad (6)$$

Важливим є друге перетворення: воно гарантує, що умови переходів обчислюються за фактичним положенням, а не за інерціальним численням мережі.

Симулятор обгортає БПЛА у векторизоване середовище, тому один крок фізики дає n незалежних переходів. Винагорода складається з прогресу, вартості кроку, штрафу за наближення до перешкод, термінального штрафу за зіткнення та термінальної винагороди за досягнення цілі:

$$R_i = \lambda_p(d_i^{t-1} - d_i^t) - \lambda_s - \lambda_w max\left(0, \frac{d_{warn} - m_i}{d_{warn}}\right) - \lambda_c 1\left[m_i < d_{safe}\right] + \lambda_g 1[d_i^t < \rho] \quad (7)$$

де d_i^t — відстань до цілі, m_i — найближча перешкода серед секторів. Епізод завершується зіткненням або досягненням цілі та обривається через ліміт кроків.

Цілі генеруються трьома механізмами. Частка p_d — примусові обходи, розміщені за найближчою виявленою перешкодою:

$$g_i = x_i + \left(r_{i,j^*} r_{\max} + \mu\right)\left[cos\beta_{j^*}, sin\beta_{j^*}, 0\right], \quad j^* = \arg\min_j r_{i,j} \quad (8)$$

Дані цілі недосяжні по прямій, тож винагороду можна отримати лише, оминувши перешкоду; без цього випадкові цілі в переважно відкритому просторі навчають уникнення як відступ, а не як обхід. Решта цілей береться з двох варіантів відстаней:

$$\|g_i - o_i\| \sim \{U \quad (9)$$

Короткі відтворюють масштаб точок маршруту в процесі руху, довгі — тренують прокладання шляху в масштабі середовища від точки старту. Висота цілі має фіксований запас відносно поверхні середовища, $g_i^z = h(g_i^x, g_i^y) + \Delta_h$ (з тієї самої карти висот, яку симулятор використовує для фіксації зіткнень). Зіткнення з поверхнею додатково перевіряється умовою (10), оскільки за такого темпу керування зміщення за крок сумірне з порогом зіткнення, і команда на рух може переважити відгук від зіткнення, і як тільки такий БПЛА пройде наскрізь через перешкоду — лідар знову буде бачити пустий простір.

$$x_i^z < h\left(x_i^x, x_i^y\right) + \Delta_g \quad (10)$$

Умови експерименту та результати

Експерименти виконано в симуляторі Gazebo + Rviz (рис. 4-5) з роєм із десяти БПЛА; команди посилаються за допомогою ROS 2 на двох середовищах, згенерованих із реальних даних висот і забудови:

тестовий світ розміром приблизно 6,7 × 4,5 км зі значним перепадом висот та тренувальний, але щільніше забудований, з розміром приблизно 0,83 × 0,82 км. Навчання відбувалося лише на другому.

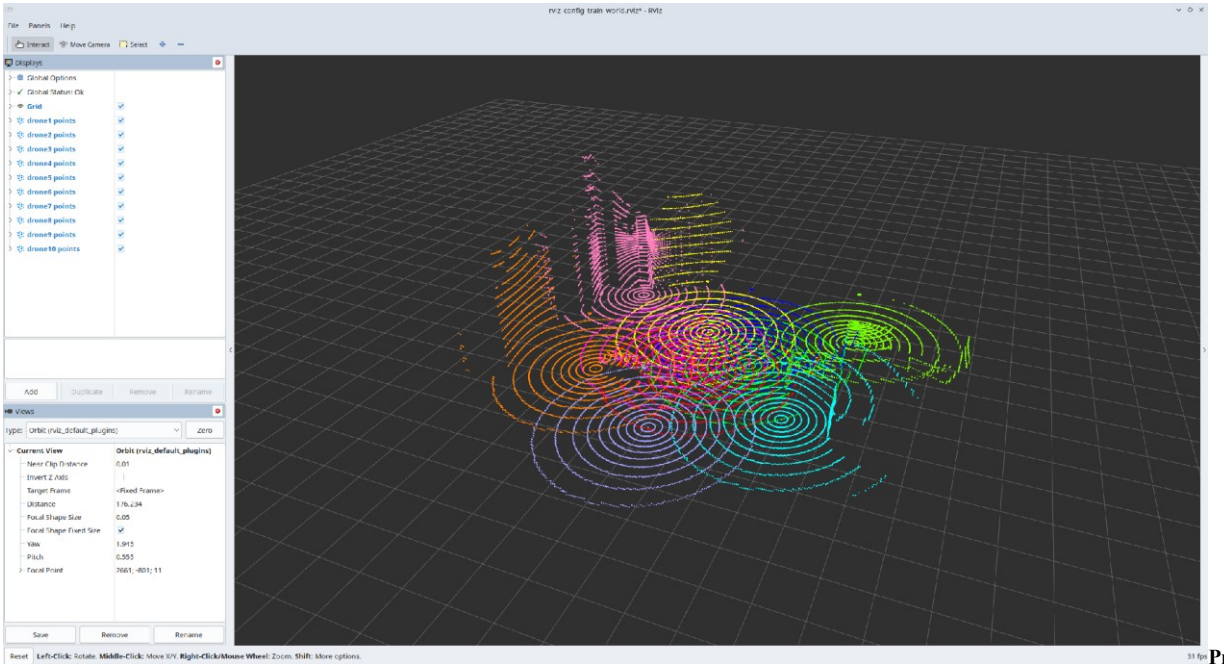

Рис. 4. Показання лідарів в середовищі rviz

Рис. 5. Рій БПЛА в Gazebo

Таблиця 2.

Конфігурація навчання

Параметр	Значення	Параметр	Значення
Алгоритм	PPO, MlpPolicy	Частота датчика	10 Гц
Довжина розгортки	256	Кутових секторів N_b	24
Розмір пакета	256	Розмірність спостереження	28
Швидкість навчання	3×10^{-4}	Випередження L	10 м
Коефіцієнт знецінення	0.99	Максимальна швидкість v_{max}	10 м/с
Обмеження кроку	0.2	Радіус цілі ρ	1.5 м
Частота керування	5 Гц	Поріг зіткнення d_{safe}	1.0 м

Для емпіричного підтвердження вибору алгоритму (таблиця 1) проведено порівняння PPO та SAC. Обидва алгоритми навчалися з нуля на навчальному середовищі по 150 000 кроків з тими самими спостереженнями, діями та функцією винагороди (7). Для коректності порівняння гіперпараметри встановлено однаковими: швидкість навчання 3×10^{-4}, розмір пакета 256 та коефіцієнт знецінення 0,99 збігаються зі значеннями, наведеними в таблиці 2. Отже, різниця результатів не може бути наслідком нерівних умов оптимізації.

Такий підхід відповідає тезі, що on-policy метод має працювати без налаштування під задачу, тоді як off-policy метод є чутливим до нього. Додатково розглянуто варіант SAC з налаштуваннями — нормалізацією винагороди для критика та відкладеним початком оновлень (*learning_starts* = 10^4), які спрямовані на усунення причини відмови — великого розкиду значень винагороди (7), що дестабілізує навчання критика.

Результати наведено на рис. 6. PPO досягає перших цілей уже на 20 000 кроків і стабілізується на рівні близько 50 % успішних досягнень цілі (коливання окремих вікон мають вибірковий характер: 10–30 епізодів), тоді як SAC зі стандартними налаштуваннями сходиться до пасивного локального оптимуму: БПЛА не рухаються, майже всі епізоди завершуються усіченням за лімітом кроків, а середня винагорода відповідає лише накопиченому штрафу за кроки нерухомого епізоду.

Налаштований SAC виходить із пасивної поведінки, проте за весь час досягає цілі лише один раз. Середня довжина епізоду ілюструє механізм: у PPO вона спочатку зростає, оскільки політика вчиться уникати зіткнень, а потім спадає завдяки успішним досягненням цілі, тоді як в обох варіантах SAC утримується біля ліміту кроків. Це узгоджується з аргументами таблиці 1 щодо чутливості off-policy методів до налаштувань та застарілого досвіду відтворення в нестаціонарному середовищі рою.

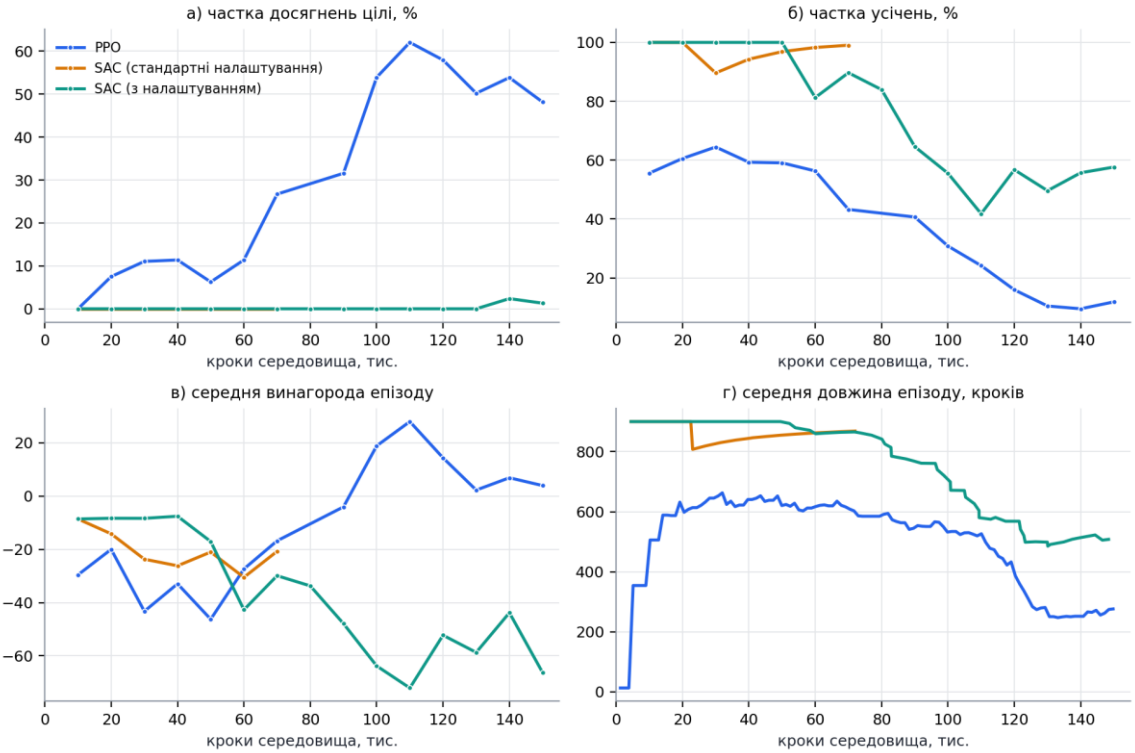

Рис. 6. Порівняння PPO та SAC.
а) частка досягнень цілі; б) частка усічень; в) середня винагорода епізоду; г) середня довжина епізоду

Проміжна PPO політика, навчена на мільйоні кроків на довгій дистанції (9), заблокувала завдання: кожен БПЛА перейшов у стан *EnRoute* і залишався в ньому з хибним предикатом прибуття. У довгій смузі нормована відстань у (1) насичена майже весь епізод, і у винагороді (7) домінує складова прогресу, тому проходження останніх метрів у межах ρ було втрачено під час навчання. Під час тестування політика отримувала проміжні цілі, а допуск (5) був обмежений знизу величиною ρ. У результаті вийшло, що БПЛА підлітали на більшу відстань, ніж радіус навчання, і нескінченно кружили навколо, ніколи не виконуючи умови (4), і дискретний рівень очікував нескінченно. Відмова невидима в метриках навчання, які описують лише навчальний розподіл: explained variance трималася близько 0.85, а стандартне відхилення політики спадало рівномірно. Повернення до навчання на короткій дистанції усунуло відмову.

Політика зі змішаним розподілом цілей, навчена сумарно 10×10^6 кроків, була оцінена на обох середовищах, по 50 епізодів на кожному. Середовища суттєво відрізняються за складністю: навчальне щільніше забудоване і містить більше перешкод на одиницю довжини маршруту, ніж тестове.

Таблиця 3.

Оцінка роботи

Результат	Навчальне середовище	Тестове середовище
Ціль досягнуто	40 %	90 %
Зіткнення	30 %	5 %
Усічення	10 %	0 %

У середовищі, яке не зустрічалося під час навчання, політика зберегла компетенцію: 90 % досягнень цілі за 5 % зіткнень. Це узгоджується з конструкцією спостереження: без знання абсолютного положення та карти місцевості політика не має представлення, на яке можна спиратися, і результат є реакцією на локальну геометрію перешкод. Навчальне середовище суттєво щільніше забудоване, тому вищий рівень зіткнень на ньому відображає складнішу задачу; рівень усічень у навчальному середовищі є артефактом довгої відстані (9), протяжність якої могла бути зіставною з розміром цього середовища.

Також завдання включає несправність, що стохастично вводиться невдовзі після зльоту, тобто в БПЛА спрацьовує перехід IsProblemOccured. Зворотна траєкторія вже є попередньо обчисленою, але індекс маршруту скидався на нуль при переході - а нульовий індекс є самою ціллю місії, тому БПЛА, що переривав місію невдовзі після зльоту, спершу скеровувався в іншу сторону від бази. Тому було змінено обчислення входу в траєкторію звороту (в точку, найближчу до фактичного положення)

$$k^* = \arg\min_k \|x_i - w_k^{\text{ret}}\| \tag{11}$$

Для БПЛА, що досяг цілі штатно, (11) дорівнює нулю і поведінка не змінюється, а для аварійного переривання час повернення скоротився з приблизно 40 до менш ніж 4 секунд.

З інтеграцією обох шарів, дискретного та неперервного, завдання завершувалося для всіх БПЛА у кожному прогоні: зліт, політ до цілі з реактивним оминанням перешкод, прибуття, повернення, посадка та зарядка за обмеженням у два слоти, разом з аварійним поверненням несправного БПЛА.

ВИСНОВКИ З ДОСЛІДЖЕННЯ ТА ПОТЕНЦІЙНІ НАПРЯМКИ ДЛЯ ПОДАЛЬШИХ ДОСЛІДЖЕНЬ

Результати показують, що мережі Петрі високого рівня та політику руху на основі навчання з підкріпленням можна поєднати так, щоб кожна з частин зберегла свої сильні сторони. Мережа Петрі забезпечує те, що здатна дати лише формальна модель. Наприклад, вирішення задач розподілу спільних ресурсів забезпечене структурно, а не процедурно; включення аварійного сценарію відбувається автоматично, внаслідок того що спрацювала така умова переходу, а не як робота окремої гілки обробки помилок; паралельний рух кількох дронів. Політика руху при цьому дає те, що не здатна забезпечити формальна модель: реактивне уникнення перешкод, інформація про які з’являється під час руху на основі показань датчиків (в нашому випадку — лідар).

Інтерфейс між рівнями складається з проміжної цілі, що передається від мережі до політики управління рухом, та предикату прибуття, що повертається від політики руху до мережі. Під час тестування стало зрозуміло, що найбільша складність проєктування полягає саме в предикаті прибуття. Навчений контролер кружляв навколо цілі, а не потрапляв прямо в неї, а тому миттєва перевірка умови прибуття може бути настільки неточною, що здатна повністю заблокувати роботу дискретного рівня. Така комбінація буде залишатися стабільною за умови, що допуск предиката прибуття не жорсткіший за точність, на яку була навчена політика руху.

Також цікавий результат дало перенесення. Оскільки актор не має абсолютної інформації про положення, політика не може запам'ятовувати рельєф місцевості. Таким чином, політика, натренована в спеціальному середовищі, зберегла компетенцію в тестовому середовищі, яке до цього не зустрічалось. Результат перенесення це 90 % успішних досягнень цілі при лише 5 % зіткнень і завершення повного циклу місії. Оскільки вектор спостереження не містить абсолютного положення, навчена реактивна поведінка залежить лише від локальної геометрії перешкод, а не від конкретного середовища.

Наукова новизна роботи полягає в тому, що вперше було запропоновано гібридний метод навігації рою БПЛА на основі мереж Петрі високого рівня та політики руху на основі PPO. В розробленому методі формальна модель місії делегує навченій політиці етап руху між сегментами. Структура мережі залишилася незмінною порівняно з моделюванням, у якому частина руху була умовною. Було додано лише зв'язок між дискретним та неперервним рівнями, а саме проміжну ціль, що передається від мережі вниз, та предикат прибуття, що обчислюється за виміряним положенням БПЛА. На відміну від підходів із покроковим обмеженням дій агента формальним супервізором [13], запропонований інтерфейс не втручається в керування всередині сегмента маршруту, що робить критерій прибуття центральним елементом між рівнями.

Для подальшого розвитку дослідження, пропонується зосередитись на наступних напрямках:

По-перше, збільшення кількості дронів у рою. Це не потребує внесення змін у саму мережу чи політику руху, проте потребує змін у навколишній інфраструктурі. Наприклад, протокол MAVLink підтримує передачу сигналу від наземної станції до груп із 255 дронів і завдяки невеликому розміру пакетів підходить для повільних або зашумлених каналів зв’язку.

По-друге, інтеграція складніших сценаріїв, наприклад, зміна формацій з лінійної на ієрархічну або повнозв'язну залежно від перешкод, або зміна лідера в рою знову ж через перешкоди чи внутрішні несправності. По-третє, виявлення недосяжних проміжних цілей — точок маршруту, розташованих усередині перешкод, та їх пропуск або перепланування, оскільки наразі така точка блокує просування місії до спрацювання аварійного повернення.

Література

1. Ivankov, V., & Novotarskyi, M. (2025). Drone Swarm Control Model Based on High-Level Petri Nets. Information, Computing and Intelligent Systems, (6), 152–163. https://doi.org/10.20535/2786-8729.6.2025.333220

2. Fedorova, A., Beliautsou, V., & Zimmermann, A. (2022). Colored Petri Net Modelling and Evaluation of Drone Inspection Methods for Distribution Networks. Sensors, 22(9), 3418. https://doi.org/10.3390/s22093418

3. Pelletier, B., Lesire, C., Doose, D., Godary-Dejean, K., & Dramé-Maigné, C. (2022). SkiNet, A Petri Net Generation Tool for the Verification of Skillset-based Autonomous Systems. Electronic Proceedings in Theoretical Computer Science, 371, 120–138. https://doi.org/10.4204/EPTCS.371.9

4. Andrade, E., & Machida, F. (2024). Assuring Autonomy of UAVs in Mission-critical Scenarios by Performability Modeling and Analysis. ACM Transactions on Cyber-Physical Systems, 8(3), Article 30. https://doi.org/10.1145/3624572

5. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint https://doi.org/10.48550/arXiv.1707.06347

6. Joshi, B., Kapur, D., & Kandath, H. (2024). Sim-to-Real Deep Reinforcement Learning Based Obstacle Avoidance for UAVs Under Measurement Uncertainty. In 2024 10th International Conference on Automation, Robotics and Applications (ICARA) (pp. 278–284). IEEE. https://doi.org/10.1109/ICARA60736.2024.10553074

7. Sheng, Y., Liu, H., Li, J., & Han, Q. (2024). UAV Autonomous Navigation Based on Deep Reinforcement Learning in Highly Dynamic and High-Density Environments. Drones, 8(9), 516. https://doi.org/10.3390/drones8090516

8. Wang, J., Yu, Z., Zhou, D., Shi, J., & Deng, R. (2024). Vision-Based Deep Reinforcement Learning of Unmanned Aerial Vehicle (UAV) Autonomous Navigation Using Privileged Information. Drones, 8(12), 782. https://doi.org/10.3390/drones8120782

9. Sachweh, T., Haritz, P., & Liebig, T. (2024). Using Petri Nets as an Integrated Constraint Mechanism for Reinforcement Learning Tasks. In 2024 IEEE Intelligent Vehicles Symposium (IV) (pp. 1686–1692). IEEE. https://doi.org/10.1109/IV55156.2024.10588380

10. Lassoued, S., & Schwung, A. (2024). Introducing PetriRL: An innovative framework for JSSP resolution integrating Petri nets and event-based reinforcement learning. Journal of Manufacturing Systems, 74, 690–702. https://doi.org/10.1016/j.jmsy.2024.04.028

11. Luo, J., Yi, S., Lin, Z., Zhang, H., & Zhou, J. (2024). Petri-net-based deep reinforcement learning for real-time scheduling of automated manufacturing systems. Journal of Manufacturing Systems, 74, 995–1008. https://doi.org/10.1016/j.jmsy.2024.05.006

12. Kirk, R., Zhang, A., Grefenstette, E., & Rocktäschel, T. (2023). A Survey of Zero-shot Generalisation in Deep Reinforcement Learning. Journal of Artificial Intelligence Research, 76, 201–264. https://doi.org/10.1613/jair.1.14174

13. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., & Topcu, U. (2018). Safe Reinforcement Learning via Shielding. Proceedings of the AAAI Conference on Artificial Intelligence, 32(1), 2669–2678. https://doi.org/10.1609/aaai.v32i1.11797