УДК 681.518 DOI: https://doi.org/10.37434/tdnk2025.03.03

АНАЛІЗ ЕФЕКТИВНОСТІ АЛГОРИТМІВ REINFORCEMENT LEARNING ДЛЯ ПІДВИЩЕННЯ АВТОНОМНОСТІ МОБІЛЬНИХ РОБОТІВ

Д.В. Петренко, А.Г. Протасов

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського». 03056, м. Київ, Берестейський проспект, 37. E-mail: petrenkod555@gmail.com, a.g.protasov@gmail.com

Стаття присвячена проблемі підвищення автономності мобільних роботів, які сьогодні мають широке застосування в різних сферах діяльності людини. Удосконалення засобів керування рухом роботів у реальних умовах за рахунок впровадження інтелектуальних систем керування дозволить їм адаптуватись до зміни середовища, адекватно реагувати на непередбачувані ситуації та ефективніше взаємодіяти з іншими учасниками технологічного процесу. Інтелектуальна система керування рухом мобільного робота поєднує як апаратні, так і програмні складові. До програмних компонентів належать методи машинного навчання, які засновані на методах побудови алгоритмів, здатних навчатися. У статті розглядаються найпопулярніші алгоритми машинного навчання з підкріпленням (Reinforcement Learning, RL), які використовуються в інтелектуальних системах керування. У цьому методі головними компонентами є агент і середовище. Середовище являє собою динамічний світ, у якому діє агент, і з яким він постійно взаємодіє. Алгоритми машинного навчання RL умовно поділяються на дві групи – алгоритми, які використовують модель, і алгоритми без моделі. Із результатів проведеного аналізу очевидно, що для підвищення автономності руху мобільного робота у складних динамічних умовах необхідно застосовувати гібридні підходи, що поєднують навчання без моделі, як у алгоритмів PPO, SAC чи TD3, із модельними компонентами, як у алгоритмів PlaNet або MuZero. Також ефективною стратегією може бути автоматична адаптація гіперпараметрів під час навчання, наприклад коефіцієнта ентропії в алгоритмі SAC або коефіцієнта обмеження політики в алгоритмі PPO, що дає підвищену стійкість до змін у середовищі та стану спостереження, зниження потреби у великій кількості взаємодій із середовищем, гнучкість адаптації до нових задач або зміни цільової поведінки. Бібліогр. 8, табл. 1, рис. 2.

Ключові слова: машинне навчання, алгоритми навчання, мобільні роботи, системи керування, автономність роботів

Вступ. Сьогодні важко собі уявити сучасні промислові підприємства, логістику, оборонну та побутову сфери без використання мобільних роботів. Ефективність їх використання залежить від рівня їх автономності. Мобільні роботи працюють в обмеженому динамічному середовищі, тому потребують розширених можливостей прийняття рішень для безпечної, ефективної та результативної навігації. Впровадження інтелектуальних систем керування рухом дозволяє вдосконалити засоби керування мобільними роботами в реальних умовах, що дає можливість їм адаптуватись до зміни середовища, адекватно реагувати на непередбачувані ситуації та покращити взаємодію з іншими агентами.

Постановка проблеми. Інтелектуальна система керування рухом мобільного робота поєднує апаратні та програмні складові, які забезпечують здатність робота автономно орієнтуватися в просторі, ухвалювати рішення та виконувати задачі у змінному, частково невизначеному середовищі. До програмних компонентів систем керування рухом робота належать методи машинного навчання Machine Learning (ML), які засновані на методах побудови алгоритмів, здатних навчатися [1].

Методи машинного навчання стали потужним інструментом, який дозволяє роботам адаптуватися до динамічних змін, обробляти дані датчиків у реальному часі та виконувати складні маневри в обмеженому просторі. Останні дослідження показують, що найпопулярнішими алгоритмами машинного навчання в інтелектуальних системах керування є методи навчання з підкріпленням (Reinforcement Learning, RL) [2]. Ці методи оптимізують послідовне прийняття рішень, використовуючи попередній досвід робота та вдосконалюючи стратегію керування. Вони змінюють сприйняття роботом оточення та взаємодіють із ним для планування подальшого руху, вирішують завдання уникнення перешкод в умовах динамічного середовища, особливо коли середовище частково або повністю невідоме. Планування руху та уникнення перешкод як на глобальному, так і локальному рівні забезпечується за допомогою використання класичних алгоритмів A*, DWA, MPC, RRT тощо [3].

Методи глибокого навчання (DL) використовують нейронні мережі для обробки складних сенсорних даних, зокрема візуальної інформації з камер. Це дозволяє роботу здійснювати аналіз у

А.Г. Протасов – https://orcid.org/0000-0002-2965-3334, Д.В. Петренко – https://orcid.org/0009-0003-7670-555X
© Д.В. Петренко, А.Г. Протасов, 2025

реальному часі, розпізнавати об'єкти та приймати рішення. DL також може застосовуватися для інтеграції різних типів сенсорних даних (наприклад з камер, LiDAR або інерційних приладів IMU), що підвищує точність сприйняття та надійність автономного функціонування [4].

На сьогодні потенціал можливостей методів машинного навчання в системах керування мобільними роботами до кінця не розкритий, тому існує потреба в подальшому дослідженні методів, які підходять для керування автономними роботами в складних умовах, підкреслюючи їхні сильні сторони, обмеження та потенціал інтеграції. Зосереджуючись на алгоритмах, адаптованих до обмежених налаштувань із динамічними перешкодами, необхідно визначити найперспективніші алгоритми для модернізації та покращення, щоб підвищити автономність і стійкість роботів.

Метою даної роботи є аналіз сучасних алгоритмів машинного навчання з підкріпленням (RL), які можуть бути застосовані в системах керування мобільними роботами.

Виклад основного матеріалу. У методі машинного навчання з підкріпленням (RL) головними компонентами є агент і середовище. Середовище являє собою динамічний світ, у якому діє агент, і з яким він постійно взаємодіє. На кожному кроці цієї взаємодії агент отримує спостереження за станом середовища (St) (часткове або повне), після чого приймає рішення про виконання певної дії (At). У результаті дії агента середовище змінює свій стан, хоча його зміна може також відбуватися незалежно від впливу агента, наприклад через природні процеси чи динаміку середовища. Ключовим елементом у цій взаємодії є сигнал винагороди (Rt), який агент отримує від середовища. Винагорода – це числовий показник, що інформує агента про якість поточного стану середовища з точки зору поставленої мети (рис. 1) [5]. Метою

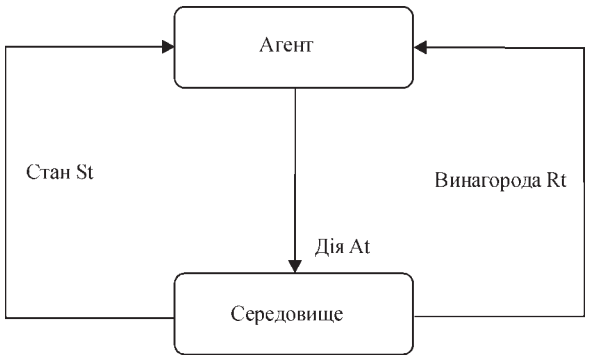

Рис. 1. Класична структура машинного навчання з підкріпленням

агента є максимізація кумулятивної винагороди, яка враховує всі винагороди, отримані протягом взаємодії з середовищем. Методи навчання з підкріпленням розроблені для того, щоб допомогти агенту навчитися ефективної поведінки, яка дозволить досягти оптимального результату. Вони базуються на поступовому покращенні стратегії дій агента через накопичення досвіду та зворотного зв'язку від середовища.

Методи RL алгоритмів можна розділити на два типи – методи на основі моделі (model-based methods) та методи без моделі (model-free methods), які відрізняються один від одного у підходах до навчання (рис. 2) [6]. Обидва типи методів знайшли широке застосування у сфері автономних мобільних роботів, які діють у динамічному середовищі [7]. Розглянемо основні принципи та алгоритми деяких підходів, їхні переваги, недоліки та сучасні тенденції інтеграції, що сприяють розширенню можливостей навчання з підкріпленням у вирішенні складних завдань.

Алгоритми RL на основі моделі. Основною перевагою використання моделі у машинному навчанні є можливість планування. Завдяки моделі агент може передбачати наслідки своїх дій у різних сценаріях, аналізувати ці наслідки та при-

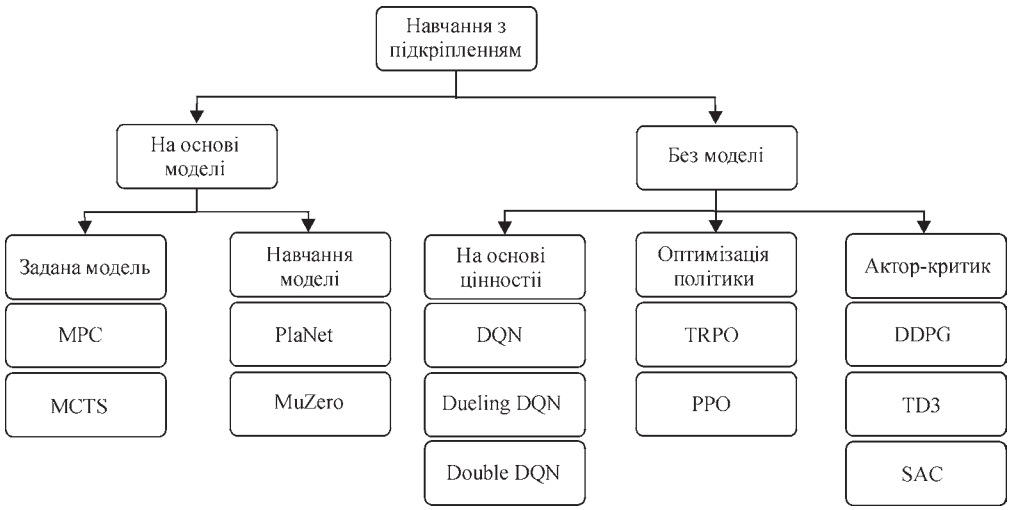

Рис. 2. Алгоритми машинного навчання з підкріпленням

ймати обґрунтованіші рішення. Результати такого планування можуть бути конвертовані у стратегічну політику, що оптимізує поведінку агента. У контексті автономних мобільних роботів модель середовища може бути використана для прогнозування динаміки руху об'єктів, оцінки ризику зіткнення або планування шляхів у реальному часі. Наприклад, методи на основі моделі можуть допомагати роботам уникати перешкоди, передбачаючи рух інших агентів у динамічному середовищі.

Метод керування Model Predictive Control (MPC) – це метод, який використовує динамічну модель системи для передбачення її майбутнього стану та знаходження оптимальної послідовності керуючих впливів, мінімізуючи функціонал вартості за обмеженнями. MPC працює за принципом рухомого горизонту прогнозування: на кожному часовому кроці система формує прогноз майбутньої поведінки робота на обмеженому горизонті часу, розв'язує задачу оптимального керування з урахуванням обмежень і обирає найкращу послідовність керуючих дій. З цієї послідовності застосовується лише перше керуюче зусилля, після чого стан системи оновлюється та весь процес повторюється з новими вхідними даними. Такий підхід забезпечує адаптивність до змін середовища в реальному часі, дотримання обмежень і мінімізацію помилок. Незважаючи на переваги (гнучкість, передбачуваність, точність), він має ряд недоліків, таких як: високі обчислювальні витрати, чутливість до похибок у математичній моделі робота, складність налаштування параметрів, обмежена швидкість реакції на раптові зміни.

Стохастичний метод пошуку Monte Carlo Tree Search (MCTS) – це метод пошуку та прийняття рішень на основі побудови ітераційного дерева пошуку за допомогою випадкового моделювання. MCTS складається з чотирьох послідовних фаз, які ітераційно повторюються для побудови та оновлення дерева пошуку. На етапі вибору (Selection) здійснюється проходження по вже існуючому дереву до вузла, який буде розширено, зазвичай із застосуванням евристики, наприклад критерію UCT. Після цього виконується розширення (Expansion), де до вибраного вузла додається один або декілька нових нащадків. У фазі моделювання (Simulation) проводиться випадкове або кероване моделювання дій від нового вузла до кінцевого стану з метою оцінки потенційної винагороди. Нарешті, у фазі зворотного оновлення (Backpropagation) отримані результати моделювання рекурсивно передаються вгору по дереву, корегуючи оцінки відвіданих вузлів. Така послідовність дій дозволяє алгоритму ефективно досліджувати простір рішень і поступово наближатися до оптимальної траєкторії чи стратегії дій у складному середовищі. MCTS ефективно використовується для планування траєкторій та прийняття рішень у динамічних середовищах, де середовище може змінюватися непередбачувано, а точне моделювання є складним для обчислювання. Попри численні переваги алгоритм має певні обмеження, такі як висока обчислювальна складність, низька ефективність у великих просторах станів, проблеми з детермінованими системами, залежність від функції винагороди.

Алгоритми з навчанням моделі. *Алгоритм планування PlaNet (Planning with Latent Networks)* – це алгоритм, що працює в середовищах із частковою або відсутньою інформацією про динаміку. Він використовує модельне навчання з підкріпленням (model-based RL) із латентним (прихованим) поданням станів та дозволяє агенту планувати дії без необхідності розгортання детальної моделі середовища. PlaNet складається з чотирьох основних кроків: спочатку сенсорні спостереження кодуються у компактні латентні представлення за допомогою варіаційного автоенкодера; потім рекурентна стохастична модель прогнозує майбутні латентні стани на основі попередніх станів і дій; далі виконується планування послідовності дій для максимізації очікуваної суми винагород у латентному просторі за допомогою Model Predictive Control; і, нарешті, параметри моделі навчаються через оптимізацію варіаційної нижньої межі (ELBO), що поєднує точність відновлення спостережень і узгодженість динамічних прогнозів. Таким чином, PlaNet ефективно планує в складних динамічних середовищах, використовуючи латентне моделювання динаміки та дозволяючи приймати рішення на основі обмежених даних. Проте алгоритм має високі обчислювальні витрати, залежить від якості латентного подання, потребує багато навчальних даних і не завжди підходить для швидкої адаптації в реальному часі. Ці обмеження можуть ускладнювати його практичне застосування для автономних мобільних роботів.

Алгоритм MuZero – це алгоритм, що поєднує планування на основі моделі з навчанням функції політики (policy function) та функції цінності (value function) без явного моделювання динаміки середовища. Він не потребує попереднього знання правил динаміки системи, а навчає приховану модель для прогнозування майбутніх станів та отриманих винагород. Основна ідея полягає в тому, що агент навчається трьом ключовим функціям: функція представлення – перетворює історію спостережень у прихований стан, який стисло описує поточний стан середовища; функція динаміки – імітує перехід між

станами та прогнозує винагороду при виконанні певної дії; функція політики та оцінки – прогнозує ймовірності вибору дій (політику) та значення (корисність) стану. Працюючи циклічно, MuZero на основі поточного стану виконує пошук у дереві можливих майбутніх станів методом MCTS, використовуючи внутрішню модель для прогнозування наслідків дій. Це дає змогу ефективно планувати та обирати оптимальні дії навіть у випадках, коли динаміка середовища невідома або складна. Після виконання дії агент отримує нові спостереження, оновлює своє розуміння стану через функцію представлення та повторює цикл. Паралельно нейронні мережі, які реалізують ці функції оновлюються на основі отриманого досвіду, мінімізуючи сумарну функцію втрат, яка враховує три основні компоненти: помилки у передбаченні винагороди, політики та цінності. Цей підхід дозволяє автономним мобільним роботам адаптивно навчатися й планувати дії у складних, динамічних і частково невідомих середовищах, забезпечуючи високий рівень автономності та ефективності. Водночас алгоритм є обчислювально затратним через інтенсивний пошук у дереві станів і тренування складних нейронних мереж, що може ускладнювати його застосування в реальному часі на роботах з обмеженими обчислювальними ресурсами.

Попри значні переваги, методи на основі моделі стикаються з серйозними викликами, основний недолік такого підходу полягає у відсутності доступу до точної моделі середовища в більшості реальних сценаріїв. У таких випадках агент повинен самостійно навчатися моделі, ґрунтуючись виключно на зібраному досвіді. Це створює ризик упередженості моделі, коли агент може працювати оптимально у межах вивченої моделі, але демонструвати неоптимальну поведінку в реальному середовищі, слабкі результати при зміні динаміки або появі невідомих факторів. Наприклад, у динамічних середовищах, таких як жваві вулиці або складські приміщення з великою кількістю роботів, помилки моделювання можуть призвести до критичних збоїв у роботі. Також навчання моделей середовищ потребує великих обчислювальних витрат, особливо в задачах з високою розмірністю простору станів або дій.

Алгоритми RL без моделі. Безмодельні підходи, на відміну від методів на основі моделі, покладаються лише на прямий досвід взаємодії агента з середовищем. Вони не потребують явної моделі для прогнозування наслідків дій, а замість цього вивчають політику або функцію цінності через багаторазові проби та помилки [8]. Для автономних мобільних роботів у динамічних середовищах безмодельні методи демонструють високу адаптивність, особливо в умовах, де середовище постійно змінюється й агенту потрібно швидко реагувати на нові обставини. Розглянемо найперспективніші алгоритми.

Алгоритми на основі цінності (Value-Based Methods). *Метод Deep Q-Network (DQN)* поєднує глибокі нейронні мережі та *Q*-навчання для вирішення задач навчання агента в складних середовищах, що змінюються в реальному часі, з випадковими перешкодами або рухомими об'єктами. Робот навчається на основі власного досвіду, зберігаючи транзакції в буфері відтворення, що допомагає уникнути кореляції між послідовними зразками даних. Глибока нейронна мережа дозволяє наближати складні залежності між станами та діями, що робить метод придатним для великих просторових областей. Основною метою DQN є наближення функції *Q*-значень за допомогою нейронної мережі, що дозволяє агенту приймати оптимальні рішення в ситуаціях з високою розмірністю станів. Функція *Q*-значень визначається, як:

$$Q(s,a) = E\left[r + \gamma \max_{a'} \left[Q(s',a') \middle| s,a \right] \right],$$

де $Q(s,a)$ – оцінка якості дії a у стані s; r – винагорода, отримана після виконання дії; $\gamma \in [0,1]$ – коефіцієнт дисконтування, що враховує довгострокову винагороду; s' – новий стан середовища після виконання дії a; a' – наступна дія агента; E – математичне сподівання, використовується для обчислення середнього значення випадкової величини. DQN використовує нейронну мережу для апроксимації функції $Q(s,a;\theta)$, де θ – параметри цільової Q-функції. Навчання відбувається шляхом мінімізації функції витрат:

$$L(\theta) = E\left[(y - Q(s,a;\theta))^2 \right],$$

де $y = r + \gamma \max_{a'} Q(s',a',\theta^-)$ – цільове значення, θ^- – параметри цільової нейронної мережі, які оновлюються періодично для стабільності навчання.

Загальний алгоритм DQN:

1. Ініціалізація нейронної мережі $Q(s,a;\theta)$ з випадковими вагами.

2. Ініціалізація цільової мережі $Q(s,a;\theta^-) \leftarrow Q(s,a;\theta)$.

3. Повторювати для кожного епізоду:

3.1. Отримання поточного стану s.

3.2. Для кожного кроку в епізоді:

3.2.1. З імовірністю ε обираємо випадкову дію a.

3.2.2. Застосовується дія, отримуємо r, s'.

3.2.3. Зберігаємо (s, a, r, s') у буфер обміну D.

3.2.4. Обирається випадкова міні-вибірка з D та виконується градієнтний спуск для мінімізації функції втрат $L(\theta)$.

3.2.5. Кожні N кроків оновлювати цільову мережу $\theta^- \leftarrow \theta$.

DQN є потужним методом для керування автономними мобільними роботами у складних динамічних середовищах завдяки здатності ефективно обробляти великі простори станів і адаптуватися до змін у середовищі. Серед основних переваг є: здатність до навчання без повного моделювання середовища, можливість узагальнення знань для нових ситуацій, покращена стабільність завдяки використанню цільової мережі та буфера відтворення. Однак метод має низку недоліків: велика обчислювальна складність і вимоги до обчислювальних ресурсів; труднощі з вибором гіперпараметрів, що впливають на ефективність навчання; можливість нестабільності при надмірному навчанні або недостатньому дослідженні середовища. Таким чином, DQN є перспективним підходом для навігації автономних роботів, але вимагає оптимізації та адаптації до конкретних задач.

Алгоритми Dueling DQN та Double DQN є модифікаціями алгоритму DQN і запропоновані для покращення стабільності та точності навчання. Double DQN усуває проблему переоцінювання *Q*-значень шляхом розділення процесів вибору дії та її оцінки за допомогою двох нейронних мереж. Dueling DQN, у свою чергу, використовує окремі гілки мережі для оцінки цінності стану та переваги дії, що дозволяє точніше оцінювати важливість станів навіть тоді, коли конкретна дія не має великого впливу. Обидва підходи демонструють підвищену ефективність у задачах навігації автономних мобільних роботів у складних середовищах. Основними недоліками даних модифікацій є вища обчислювальна складність через використання додаткових мережевих компонентів, потреба в більшій кількості даних для ефективного навчання, обмежений ефект у задачах з неперервними просторами дій.

Алгоритми оптимізації політики (Policy Optimization). *Метод Trust Region Policy Optimization (TRPO)* використовується в рамках глибокого навчання з підкріпленням, спрямований на стабільну та ефективну оптимізацію політики шляхом обмеження величини оновлення параметрів політики в межах області довіри (trust region). TRPO забезпечує гарантоване покращення політики на кожному кроці навчання, що робить його привабливим для застосування в задачах керування автономними мобільними роботами в складних і динамічних середовищах. Алгоритм TRPO працює ітераційно: на кожному кроці агент збирає траєкторії і одночасно взаємодіє з середовищем за поточною політикою, після чого оцінюється перевага дій (advantage estimation), що вказує на їхню ефективність. Далі формується сурогатна функція винагороди, яка максимізується під обмеженням на середню KL-дивергенцію між старою та новою політикою, що гарантує стабільність оновлення. Для розв'язання цієї обмеженої задачі застосовується метод кон'югованих градієнтів з квадратичною апроксимацією KL-дивергенції та лінійним пошуком для визначення кроку оновлення параметрів політики. Процес повторюється до збіжності або досягнення заданої продуктивності. Ключовою перевагою TRPO є його здатність зберігати стабільність оновлень політики, запобігаючи великим стрибкам у політиці. Добре працює в умовах високої варіативності середовища, таких як динамічні середовища автономних мобільних роботів, де точність і передбачуваність критичні для безпеки. Алгоритм здатний оптимізувати політику для задач з великою кількістю етапів (наприклад де роботи мають виконувати тривалі завдання). Однак через високу обчислювальну складність і потребу у великих обсягах даних його застосування на реальних роботах може бути обмеженим.

Градієнтний алгоритм Proximal Policy Optimization (PPO) – це сучасний алгоритм RL, він був розроблений для забезпечення ефективного та стабільного навчання агентів, поєднує ідеї TRPO та стохастичної оптимізації, але є значно простішим у реалізації та має менші затрати на обчислювання. PPO базується на оновленні політики шляхом обмеження зміни між послідовними оновленнями, що запобігає надмірним корегуванням стратегії агента та сприяє стабільному навчанню. Він вводить обмеження на зміну політики за допомогою clip-функції для уникнення нестабільних оновлень. Основна цільова функція PPO виглядає наступним чином:

$$L(\theta) = E_t\left[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)A_t)\right],$$

де $r_t(\theta)$ – відношення ймовірностей нової до старої політики; A_t – функція переваги (advantage function), яка оцінює, наскільки дія була кращою; ε – коефіцієнт обмеження, який контролює зміну політики (гіперпараметр). Ця функція обмежує величину оновлення політики, що сприяє стабільному та безпечному навчанню. Якщо нова політика сильно відрізняється від старої, clip запобігає занадто великим змінам градієнта. Перевагами даного алгоритму є: значно легший за TRPO; не потребує обчислення Гессе або складних обмежень; забезпечує контроль над зміною політики через механізм кліпінгу; може використовувати міні-батчі та стохастичну оптимізацію, що робить його придатним для великих наборів даних; придатний до навчання великих нейромереж при моделюванні та в реальних системах. Недоліками є: неможливість гарантувати поступове підвищення продуктивності алгоритму, нестабільність при виборі гіперпараметрів, розміри батчів і навчальна

швидкість суттєво впливають на продуктивність, обмежене управління політикою.

Попри недоліки, PPO є одним з найуспішніших алгоритмів оптимізації політики, особливо у випадках, де необхідно забезпечити баланс між стабільністю, ефективністю та обчислювальною простотою. Для задач автономного керування мобільними роботами в складних динамічних середовищах PPO часто є кращим вибором, ніж TRPO, завдяки гнучкості, швидкому навчанню та практичній ефективності.

Алгоритми «актора-критика» (Actor-Critic). *Алгоритм Deep Deterministic Policy Gradient (DDPG)* – це off-policy алгоритм RL, що поєднує детерміновану політику з функціональним апроксиматором і градієнтами політики. Його особливістю є здатність ефективно працювати у неперервному просторі дій, що робить його придатним для керування роботами з точним керуванням. DDPG є основою для подальших алгоритмів, таких як TD3 та SAC. Ключовими ідеями алгоритму DDPG є використання детермінованої політики для роботи в неперервних просторах дій, що дозволяє безпосередньо відображати стан у точну дію, а також застосування архітектури «актор-критик», де актор навчається приймати оптимальні дії, а критик оцінює їх якість. Для стабілізації навчання використовуються цільові мережі з повільним оновленням і буфер повторів, який дозволяє повторно використовувати попередній досвід і розривати часову кореляцію між зразками. Крім того, до дій додається шум для забезпечення дослідження середовища в процесі навчання.

Оновлення критика здійснюється мінімізацією помилки Беллмана:

$$L(\theta^Q) = E_{(s,a,r,s)}\left[(\theta^Q(s,a|\theta^Q) - y)^2\right],$$

де $y = r + \gamma Q'(s, \mu'(s|\theta^{\mu'})|\theta^Q)$, γ – коефіцієнт дисконтування, Q'та μ' – цільові функції критика та актора відповідно.

Оновлення актора через максимізацію очікуваної цінності дій:

$$\nabla_{\theta^\mu} J \approx E_s\left[\nabla_a Q(s,a|\theta^Q)\Big|_{a=\mu(s)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)\right].$$

Алгоритм DDPG є потужним інструментом для навчання автономних мобільних роботів у неперервних просторах дій завдяки використанню «актор-критик»-архітектури та детермінованої політики, що забезпечує швидке й ефективне прийняття рішень у реальному часі. Проте його ефективність значною мірою залежить від налаштування гіперпараметрів і може бути нестабільним під час навчання, особливо у складних середовищах із шумом. Крім того, DDPG потребує великої кількості взаємодій із середовищем для досягнення належної якості політики, що може бути ресурсно затратним. Незважаючи на ці недоліки, DDPG залишається одним із найефективніших методів для задач неперервного керування в автономній робототехніці.

Алгоритм Twin Delayed Deep Deterministic Policy Gradient (TD3) – це вдосконалений «актор-критик»-алгоритм, розроблений як покращення DDPG для просторів неперервних дій (рух вперед, назад, ліворуч, праворуч і т.п.). Він зменшує переоцінку функції цінності (Q-функції), стабілізує навчання та надмірну чутливість до шуму. TD3 зберігає архітектуру «актор-критик», але додає ключові механізми для усунення нестабільності, характерної для DDPG. TD3 покращує стабільність і точність навчання шляхом використання двох Q-функцій (twin critics) для усунення систематичної переоцінки значень та ймовірності розривів у політиці, затриманого оновлення політики (delayed policy update), що дозволяє стабільніше корегувати параметри актора, та модифікованої таргетної політики з додаванням шуму (target policy smoothing). Шум згладжує політику, запобігаючи перенавчанню Q-функції на різких змінах політики, дуже корисний у фізичних роботах із шумом у сенсорах. Оновлення політики (актора) виконується рідше, ніж критиків (наприклад, кожен другий або третій крок), це дозволяє критикам краще оцінити поточну політику та уникнути дестабілізуючого зворотного зв'язку. Цільове значення для критика:

$$y = r + \gamma \min_{i=1,2} Q'_{\theta_i}(s', \mu'_\Phi(s') + \varepsilon),$$

де ε – додавання шуму для стабільності (target policy smoothing $\mu'_\Phi(s')+\varepsilon$), Φ – параметри цільової політики $\mu'_\Phi(s')$.

Оновлення критиків (обох Q-функцій):

$$L(\theta_i) = E_{(s,a,r,s)}\left[(Q_{\theta_i}(s,a) - y)^2\right], i = 1,2.$$

Оновлення актора (рідше за критика):

$$\nabla_\Phi J \approx E_s\left[\nabla_a Q_{\theta_1}(s,a)\Big|_{a=\mu_{\theta\Phi(s)}} \nabla_\Phi \mu_\Phi(s)\right].$$

TD3 демонструє значне покращення порівняно з DDPG завдяки стабільнішому навчанню, меншій переоцінці Q-функції та точнішому оновленню політики. Його основними перевагами є стійкість до шуму, стабільність у складних умовах і покращена точність у виборі дій. Серед недоліків – збільшена обчислювальна складність через використання двох Q-функцій та необхідність точного налаштування кількості затриманих оновлень актора.

Алгоритм Soft Actor-Critic (SAC) – це off-policy «актор-критик»-алгоритм, який використовує стохастичну політику та ентропійну регуляризацію для досягнення балансу між дослідженням і екс-

плуатацією. Основна ідея полягає в тому, щоб не лише максимізувати очікувану винагороду, але й ентропію політики, що дозволяє агенту балансувати між експлуатацією відомих стратегій і дослідженням нових. Це особливо важливо для автономних мобільних роботів, що функціонують у динамічних, частково спостережуваних і шумних середовищах, де адаптивність і стійкість до несподіваних ситуацій є ключовими вимогами. Крім того, SAC використовує два *Q*-критики для зменшення переоцінки, як і в TD3, а також має автоматичне налаштування коефіцієнта ентропії, що робить алгоритм менш чутливим до гіперпараметрів.

Цільове значення для *Q*-функції:

$$y = r + \gamma E_{a'\sim\pi}\left[\min_{i=1,2} Q_{\theta'_i}(s',a') - a\log\pi_\Phi(a',|s')\right]$$

де $Q_{\theta'i}$ – цільова *Q*-функція (target network), logπ у цільовому значенні спонукає до стохастичності, α – коефіцієнт, що контролює баланс між експлуатацією та дослідженням.

Оновлення політики (актора): $J_\pi(\Phi) = E_s\left[E_{\alpha\sim\pi}\left[\alpha\log\pi_\Phi(a|s) - Q_{\theta_i}(s,a)\right]\right]$. Це – баланс між експлуатацією (максимізація *Q*) та дослідженням (через ентропію). Оновлення коефіцієнта α автоматичне, що дозволяє підтримувати бажаний рівень стохастичності.

SAC поєднує високу стабільність, ефективне дослідження та автоматичне балансування між експлуатацією й експлорацією, що робить його одним з найефективніших алгоритмів для неперервного керування в складних умовах. Його основні переваги: стійке навчання, висока продуктивність при обмеженій кількості зразків, стійкість до шуму та адаптивність. Серед недоліків – підвищені обчислювальні витрати через використання кількох нейронних мереж і складність реалізації, зокрема при використанні ентропійного регулятора та автоматичного налаштування параметра α.

Отже алгоритми RL без використання моделі середовища (model-free) відзначаються своєю гнучкістю та здатністю працювати у складних, динамічних і частково спостережуваних середовищах без попереднього знання динаміки. Ці методи, зокрема SAC, TD3, PPO, DQN та їхні модифікації, безпосередньо навчаються через взаємодію агента з середовищем, що дозволяє ефективно адаптуватися до змін. Хоча такі алгоритми зазвичай потребують значної кількості взаємодій та обчислювальних ресурсів, вони демонструють високу надійність, стабільність і простоту реалізації, що робить їх особливо придатними для автономних мобільних роботів у реальних умовах.

Результати аналізу ефективності алгоритмів машинного навчання RL. Узагальнюючи результати порівняння, можна зробити висновок, що для задач автономних мобільних роботів у складних динамічних середовищах найперспективнішими є алгоритми з класу «актор-критик» з підтримкою неперервних дій і високою стабільністю навчання. До таких алгоритмів належать Soft Actor-Critic (SAC), Twin Delayed Deep Deterministic Policy Gradient (TD3), а також Proximal Policy Optimization (PPO) у поєднанні з добре підібраними гіперпараметрами. Ці методи забезпечують високу ефективність навчання, здатні працювати з багатовимірним простором дій і демонструють високу стійкість до нестабільності, яка характерна для реального фізичного середовища (див. таблицю).

Із результатів проведеного аналізу очевидно, що для покращення результатів керування мобільними роботами у складних динамічних умовах необхідно застосовувати гібридні підходи, що поєднують навчання без моделі (як PPO, SAC чи TD3) з модельними компонентами, наприклад із використанням моделі динаміки для планування або для імітаційного попереднього тренування (як у PlaNet або MuZero). Також ефективною стратегією може бути автоматична адаптація гіперпараметрів під час навчання, наприклад коефіцієнта ентропії в SAC або коефіцієнта обмеження політики в PPO.

Використання запропонованих підходів може дати наступні ефекти:

- швидку збіжність політики в реальних умовах;
- підвищену стійкість до змін у середовищі та стану спостереження;
- зниження потреби у великій кількості взаємодій із середовищем, що критично для фізичних роботів;
- гнучкість адаптації до нових задач або зміни цільової поведінки.

Загалом ефективна комбінація сучасних алгоритмів навчання з підкріпленням, структурованих планувальних методів і технік перенесення навчання становить основу для розробки надійних систем автономної навігації у динамічному світі.

Висновки

Результати проведеного аналізу ефективності алгоритмів машинного навчання RL свідчать, що для підвищення автономності мобільних роботів у складних динамічних середовищах найперспективнішими можуть бути алгоритми з класу «актор-критик» з підтримкою неперервних дій, а саме: Soft Actor-Critic (SAC), Twin Delayed Deep Deterministic Policy Gradient (TD3), Proximal Policy Optimization (PPO).

Результати аналізу ефективності алгоритмів RL

Алгоритм	Тип	Підтримка неперервних дій	Ефективність навчання	Стабільність навчання	Витрати на обчислювання
MPC	модельний	так	висока	висока	високі
MCTS	модельний	обмежена	середня	висока	високі
PlaNet	модельний	так	висока	середня	високі
MuZero	модельний	так	висока	висока	дуже високі
DQN	безмодельний	ні	середня	середня	середні
Double DQN	безмодельний	ні	середня	покращена	середні
Dueling DQN	безмодельний	ні	середня	покращена	середні
TRPO	безмодельний	так	висока	висока	високі
PPO	безмодельний	так	висока	висока	помірні
DDPG	безмодельний	так	висока	низька	високі
TD3	безмодельний	так	висока	висока	високі
SAC	безмодельний	так	висока	висока	високі

Використання під час навчання автоматичної адаптації таких гіперпараметрів, як коефіцієнта ентропії в SAC або коефіцієнта обмеження політики в PPO, може дати наступні ефекти: швидку збіжність політики в реальних умовах, підвищену стійкість до змін у середовищі та стану спостереження, зниження потреби у великій кількості взаємодій із середовищем, гнучкість адаптації до нових задач або зміни цільової поведінки.

Список літератури/Reference

1. Петренко Д.В., Протасов А.Г. (2024) Огляд сучасних технологій підвищення автономності мобільних колісних роботів. *Вчені записки ТНУ імені В.І. Вернадського. Серія: Технічні науки*. 35(**74**), 2, 122–128. DOI: https://doi.org/10.32782/2663-5941/2024.2/17
   Petrenko, D.V., Protasov, A.G. (2024) Overview of modern technologies for increasing the autonomy of mobile wheeled robot. *Vcheni Zapysky TNU*, *Seriya: Texnichni nauky*, 35(**74**), 2, 122–128. [in Ukrainian]. DOI: https://doi.org/10.32782/2663-5941/2024.2/17
2. Akalin N., Loutfi A. (2021) Reinforcement Learning Approaches in Social Robotics. *Sensors*, 21(**4**), 1292. DOI: https://doi.org/10.3390/s21041292
3. Liu, Y. et al. (2023) Mobile robot path planning based on kinematically constrained a-star algorithm and DWA fusion. *Algorithm Mathematics,* 11(**21**), 4552. DOI: https://doi.org/10.3390/math11214552
4. Faseeh, M. et al. (2024) Deep Learning assisted real-time object recognition and depth estimation for enhancing emergency response in adaptive environment. *Results in Engineering,* **24**, 103482. DOI: https://doi.org/10.1016/j.rineng.2024.103482
5. Zhang, T., Mo, H. (2021) Reinforcement learning for robot research: A comprehensive review and open issues. *International J. of Advanced Robotic Systems*, 18(**3**). DOI: https://doi.org/10.1177/17298814211007305
6. Rybczak, M., Popowniak, N., Lazarowska, A. (2024) A survey of machine learning approaches for mobile robot control. *Robotics,* 13(**1**), 12. DOI: https://doi.org/10.3390/robotics13010012
7. Lee, M.-F.R., Yusuf, S.H. (2022) Mobile robot navigation using deep reinforcement learning. Processes, 10(**12**), 2748. DOI: https://doi.org/10.3390/pr10122748
8. Yang, L., Bi, J., Yuan, H. (2022) Dynamic path planning for mobile robots with deep reinforcement learning. *IFAC-PapersOnLine,* 55(**11**), 19–24. DOI: https://doi.org/10.1016/j.ifacol.2022.08.042

ANALYSIS OF THE EFFECTIVENESS OF REINFORCEMENT LEARNING ALGORITHMS FOR INCREASING THE MOBILE ROBOTS AUTONOMY

D.V. Petrenko, A.G. Protasov

National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute». 37 Beresteysky Ave., 03056, Kyiv, Ukraine.
E-mail: petrenkod555@gmail.com, a.g.protasov@gmail.com

The paper is devoted to the problem of increasing the autonomy of mobile robots, which are widely used in various spheres of human activity today. Improving the means of controlling the movement of robots in real conditions, through the introduction of intelligent control systems, will allow them to adapt to changes in the environment, adequately respond to unforeseen situations and more effectively interact with other participants in the technological process. The intelligent system of controlling the movement of a mobile robot combines both hardware and software components. The software components of robot movement control systems include machine learning methods, which are based on methods of constructing algorithms capable of learning. The paper considers the most popular machine learning algorithms with reinforcement (Reinforcement Learning, RL), which are used in intelligent control systems. In this method, the main components are the agent and the environment. The environment is a dynamic world in which the agent operates and with which it constantly interacts. RL machine learning algorithms are conventionally divided into two groups - algorithms that use a model and algorithms without a model. From the results of the analysis it is obvious that to increase the autonomy of mobile robot movement in complex dynamic conditions, it is necessary to apply hybrid approaches that combine model-free learning, as in the PPO, SAC or TD3 algorithms, with model components, as in the PlaNet or MuZero algorithms. Also, an effective strategy can be the automatic adaptation of hyperparameters during training, for example, the entropy coefficient in the SAC algorithm or the policy constraint coefficient in the PPO algorithm, which provides increased resistance to changes in the environment and the observation state, reducing the need for a large number of interactions with the environment, and flexibility of adaptation to new tasks or changes in target behavior. 8 Ref., 1 Tabl., 2 Fig.

Keywords: machine learning, learning algorithms, mobile robots, control systems, robot autonomy

Отримано 23.05.25
Отримано у переглянутому вигляді 19.06.25
Прийнято 01.09.25