DOI: 10.58254/viti.5.2024.03.37

УДК 621.396

Бернацький А. П. ORCID: 0000-0003-0379-075X (ВІТІ ім. Героїв Крут)

ВДОСКОНАЛЕНИЙ МЕТОД ПЛАНУВАННЯ ШЛЯХУ АВТОНОМНОГО НАЗЕМНОГО РОБОТА З ВИКОРИСТАННЯМ АЛГОРИТМУ MBD-RRT*FFT

*В дослідженні проведено аналіз проблем, пов'язаних із плануванням шляху переміщення роботів і підвищення точності та надійності їх наведення у режимі реального часу. Задля вирішення цієї проблеми досліджено, проведено вдосконалення модифікації асимптотично оптимального алгоритму BD-RRT*FT. Розроблений алгоритм MBD-RRT*FFT при застосуванні в динамічних середовищах завдяки використанню багатопотокового обчислення має кращі можливості динамічного планування. Використання Fitch's-алгоритму вибору оптимального результату пошуку надає спрямованості вибору оптимального прямолінійного шляху.*

*Задля перевірки ефективності запропонованого методу вдосконалення алгоритму проведено імітаційне моделювання з використанням власної програми моделювання. Проведено порівняння алгоритму MBD-RRT*FFT з іншими алгоритмами за трьома картами з оцінюванням показників продуктивності.*

З метою оцінки поведінки програми, а також ідентифікації вузьких місць виконання алгоритму на всіх етапах перевірки проводився контроль навантаження на пам'ять системи прийняття рішення. Це дозволило побачити підвищення споживання ресурсів пам'яті на 12–15 %, але враховуючи отримані результати щодо оптимальності та швидкості обчислень, зроблено висновки щодо мінімальності впливу. Це надало додаткове розуміння необхідності уважного вибору апаратної складової систем прийняття рішень.

*Наукова новизна методу полягає у застосуванні науково-методичного апарату з вдосконалення алгоритму BD-RRT*FT, що надало подальший розвиток щодо розширення та доповнення відомих даних про алгоритми з плануванням шляху робота і підвищення точності та надійності його наведення у режимі реального часу. Метод охоплює способи дослідження, систематизацію, коригування нових і отриманих раніше знань.*

Ключові слова: *алгоритм, багатопотоковий, швидке дослідження, випадкове дерево, планування шляху в реальному часі, жадібний пошук, відновлення шляху, асимптотичний, Fitch's-оптимізація, БАНЗ, робот.*

A. Bernatskyi An improved method planning path of an autonomous ground robot with using the MBD-RRT*FFT algorithm.

*The study analyzed the problems associated with planning the path of robots and increasing the accuracy and reliability of their guidance in real time. To solve this problem, a practical study was conducted, based on the obtained analysis, a modification of the asymptotically optimal BD-RRT*FT algorithm and was proposed and improved the MBD-RRT*FFT algorithm. was developed. Which, when applied in dynamic environments due to the use of multi-threaded computations, has better dynamic scheduling capabilities, and the use of Fitch's optimal search result selection algorithm provides a general tendency to choose the optimal straight-line path.*

*To verify the effectiveness of the proposed algorithm improvement method, simulations were carried out using our own simulation program. A comparison of the MBD-RRT*FFT algorithm with other algorithms was carried out on three maps with an evaluation of performance indicators.*

In order to evaluate the behavior of the program, as well as to identify bottlenecks in the execution of the algorithm, the memory load of the decision-making system was monitored at all stages of the verification. This made it possible to see an increase in the consumption of memory resources by 12-15%, but taking into account the obtained results regarding the optimality and speed of calculations, conclusions were made regarding the minimal impact. Additional understanding of the need for careful selection of the hardware component of decision-making systems is provided.

*The scientific novelty of the method consists in the application of a scientific and methodological apparatus for improving the BD-RRT*FT algorithm, which ensured further development in terms of expanding and supplementing known data on algorithms with robot path planning and increasing accuracy and reliability in real time. The method includes methods of research, systematization and adjustment of new and previously acquired knowledge.*

Keywords**:** *algorithm, multiprocessing, fast exploration, random tree, real-time path planning, greedy search, path recovery, asymptotic optimality, fitch's optimization, robot, UGV.*

Постановка завдання. В розрізі парадигми Стратегії воєнної безпеки України «Воєнна безпека – всеохоплююча оборона» в короткостроковій перспективі приділяється особлива

увага до «розроблення, виробництва й оснащення сил оборони сучасним озброєнням, військовою та спеціальною технікою, забезпечення засобами ураження, у тому числі безпілотними і роботизованими...» [6].

Завдання, що покладаються на безпілотні автономні наземні засобі (БАНЗ) під час проведення бойових дій в урбанізованому просторі в умовах щільної забудови міста з постійною зміною ландшафту, мають загальну тенденцію до постійного збільшення вимог до складності в режимі реального часу, а сценарії використання БАНЗ демонструють диверсифіковану тенденцію розвитку інтелектуальних здібностей [5], що вимагає розробки військових мобільних робототехнічних систем із системами прийняття рішень, здатними на самостійне (автономне) вирішення проблем пошуку шляху.

Враховуючи вище зазначене, сучасною актуальною науковою задачею [1–6; 22] постає розробка ефективних алгоритмів пошуку маршруту переміщення БАНЗ без зіткнень в умовах вузьких ділянок урбанізованого середовища та оперативного простору з перешкодами, що динамічно змінюються [1].

Таким чином, стаття призначена опису методу модифікації алгоритму BD-RRT*FN, який можна застосовувати в динамічних середовищах з метою покращення часу планування алгоритму, довжини рішення шляху, форми шляху, швидкості конвергенції, задля задоволення вимог планування шляху в реальному часі, дозволяючи БАНЗ швидко отримати оптимальний шлях без зіткнень у динамічних середовищах у режимі реального часу.

Аналіз останніх публікацій. Відомо багато прикладів досліджень проблематики планування шляху роботів [1; 2; 7; 10–12; 14–17; 19–21]. У роботі [1] Бернацьким А. П. проведено поглиблений аналіз і запропоновано метод планування шляху автономного наземного робота з використанням модифікації динамічного двонаправленого алгоритму RRT*FN. Під час експериментальної перевірки була доведена ефективність алгоритму BD-RRT*FT, що задовольняє вимогам планування шляху в реальному часі, дозволяючи БАНЗ швидко отримати оптимальний шлях без зіткнень у динамічних середовищах.

У запропонованому методі під час модифікації акцентована увага на загальну проблематику вибору фіксованої кількості вузлів відбору точок вибірки. Таке рішення є відносно розумним завдяки використанню порівняльного вибору, мінімального за кількістю точок вибірки з використанням випадкового $p_{rand}(x)$, вибраного алгоритмом RRT*, що дозволяє уникнути впливу неправильного вибору. Але на рівні початкового припущення мінімальної оцінки на вибірковому етапі ми не розуміємо кількість майбутніх ітерацій загального пошуку довжини шляху через циклічність виконання пошуку зіткнення або закінчення виконання завдання. Це спонукає до того, що початковий шлях пошуку найменшої вибірки може бути хибним і становити «максимально велике» фінальне значення. Це також призведе до сповільнення швидкості роботи внаслідок взаємозв'язку, при якому чим більший розмір дерева, тим більше навантаження на динамічний алгоритм при оновленні в реальному часі, коли алгоритм динамічного пошуку оновлює інформацію про навколишнє середовище, кількість точок відбору проб поступово збільшуватиметься з часом.

Метою статті є модифікація алгоритму пошуку шляху BD-RRT*FT в умовах динамічного урбанізованого середовища.

Виклад основного матеріалу. В роботі [1] запропонована модифікація алгоритму RRT*FN із впровадженням модифікації алгоритму та застосуванням принципу динамічного оновлення та відновлення шляху. А саме під час кожної ітерації алгоритм оновлюватиме інформацію про середовище та виконуватиме оновлення планування на основі вихідних вузлів. Крім того, також проводиться оновлення інформації про розташування БАНЗ. Щоб зменшити розмір дерева під час процесу планування, непотрібні вузли і допоміжні гілки, через які проходить БАНЗ, маркуються як *непотрібні*, використовуючи поточне розташування БАНЗ як новий кореневий вузол. Якщо запланований шлях знищено перешкодами, алгоритм

відкине вузли на дереві, покриті перешкодами, і використає простір для перебудови дерева з повними вузлами й ефективного відновлення шляху.

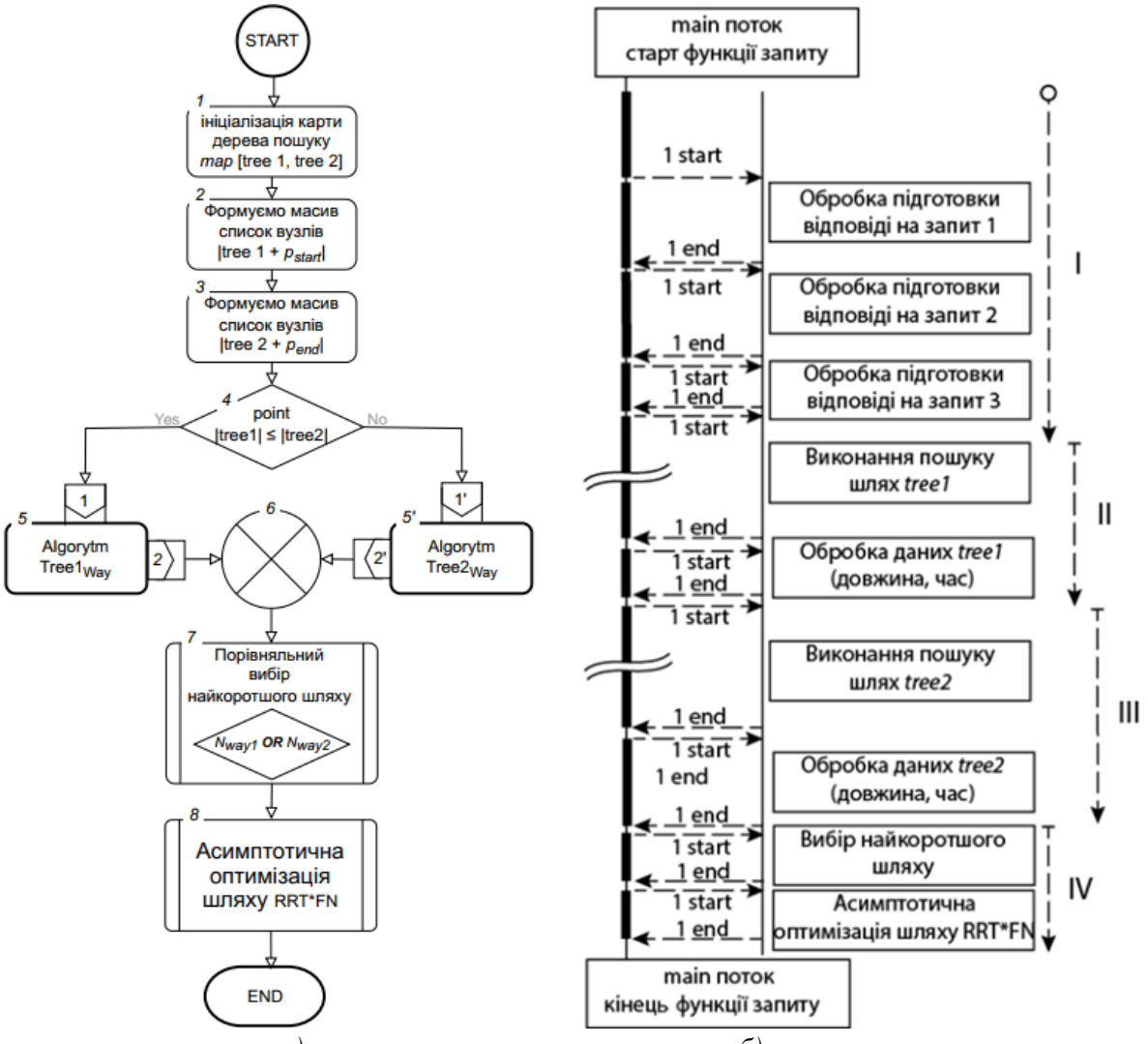
Було визначено, що алгоритм BD-RRT*FT має ймовірнісну повноту та асимптотичну оптимальність. Хоча алгоритм має ймовірнісну повноту, у діапазоні m⩽n<1, якщо кількість реперних вузлів встановлена надто малою, ймовірність отримання шляхового рішення майже дорівнює 0.

$$\begin{cases} L\{p_1, p_2, \dots, p_n\} = \sigma^*, \\ \forall p \in \sigma^*, p_{rand}(x) = p, x \in (0, iter). \end{cases} \quad (1)$$

Але якщо значення реперного вузла встановлено занадто великим, воно займатиме забагато місця в пам'яті. Зазвичай кількість реперних вузлів вибирається на основі емпіричних методів. Але для уникнення людського фактору модифікований алгоритм автоматично перевіряє кілька груп за умови фіксованого простору, станів і розміру кроку.

Через випадковість точок вибірки, вибраних алгоритмом RRT, метод вибору фіксованої кількості вузлів може бути не оптимальним, але цей метод відбору є відносно розумним і дозволяє уникнути впливу неправильного вибору фіксованої кількості вузлів для експерименту.

Загальна реалізація алгоритму BD-RRT*FT показана на рисунку 1.



а) *б)*

Рис. 1. Алгоритм BD-RRT*FT [1]:
а – блок-схема алгоритму; *б* – функціональний часовий потоковий розріз виконання алгоритму

Алгоритм BD-RRT*FT має явно виражену асинхронну властивість, яка є запозиченою з батьківського алгоритму RRT*. Така батьківська властивість вимагає послідовних однопотокових обчислень. На рівні початкового припущення мінімальної оцінки на вибірковому етапі кількість майбутніх ітерацій загального пошуку довжини шляху не відома внаслідок циклічності виконання пошуку зіткнення або закінчення виконання завдання (рис. 2). Через це початковий шлях пошуку найменшої вибірки може бути хибним і становити «максимально велике» фінальне значення, що призведе до сповільнення швидкості роботи внаслідок взаємозв'язку, при якому чим більший розмір дерева, тим більше навантаження на динамічний алгоритм при оновленні в реальному часі.

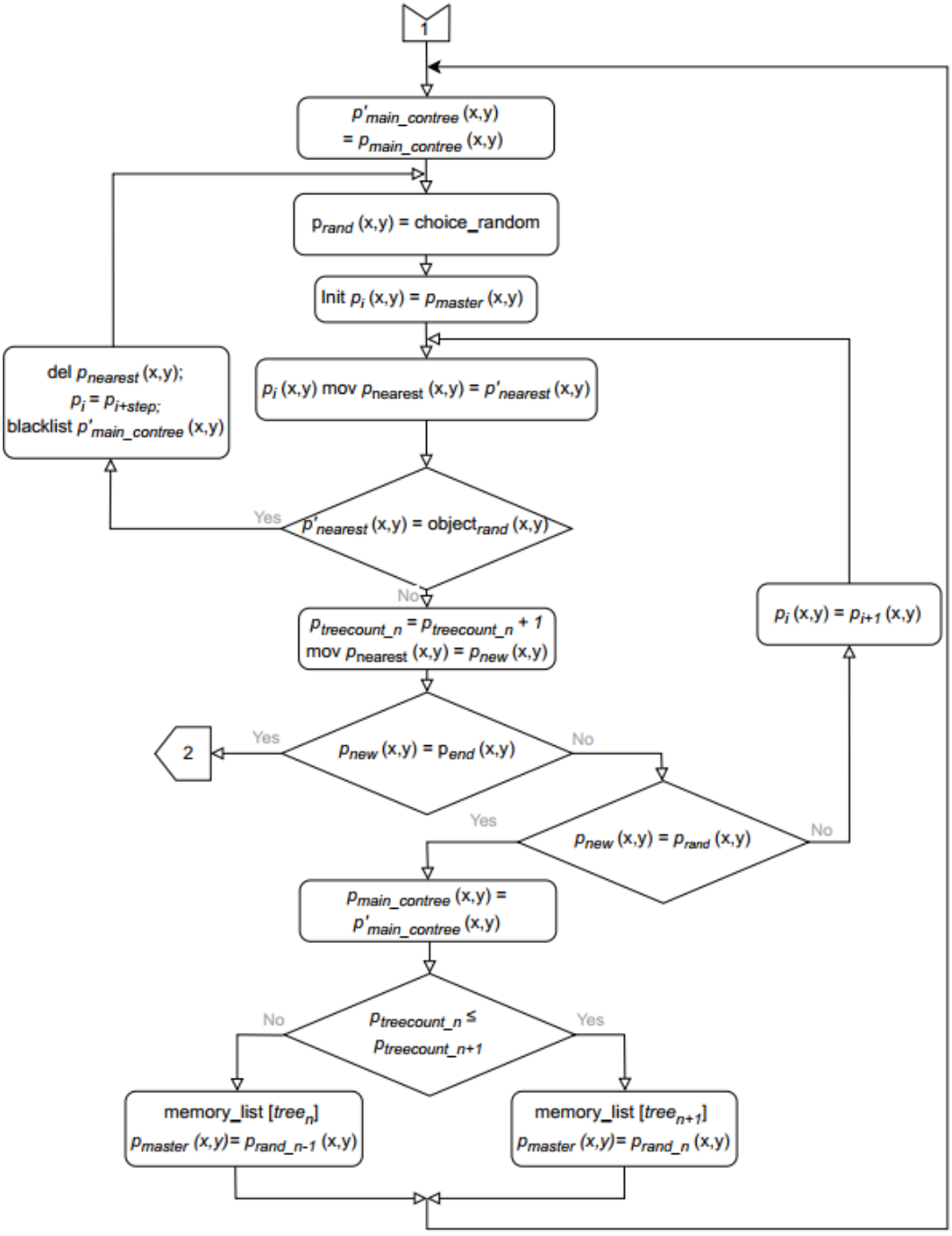


Рис. 2. Представлення у вигляді блок-схеми виконавчого блоку алгоритму пошуку шляху алгоритму BD-RRT*FT [1]

Асинхронна алгоритмізація та багатопоточність часто використовуються для досягнення схожих цілей, але між ними є ключові відмінності. Асинхронні алгоритми дозволяють продовжувати виконання операцій, не чекаючи завершення тривалої операції, що покращує чуйність. Але щоб вирішити вищеозначену проблему, ми можемо використовувати багатопотоковість, яка дозволяє алгоритму створювати кілька потоків для одночасної обробки кількох завдань пошуку шляхів.

Пропонується вдосконалений алгоритм, що використовує стратегію алгоритму BD-RRT*FT двостороннього жадібного пошуку шляху з принципом динамічного оновлення та відновлення шляху з додаванням блока паралельних багатопотокових обчислень.

Реалізація багатопокового алгоритму MBD-RRT*FFT зображена на рисунку 3.

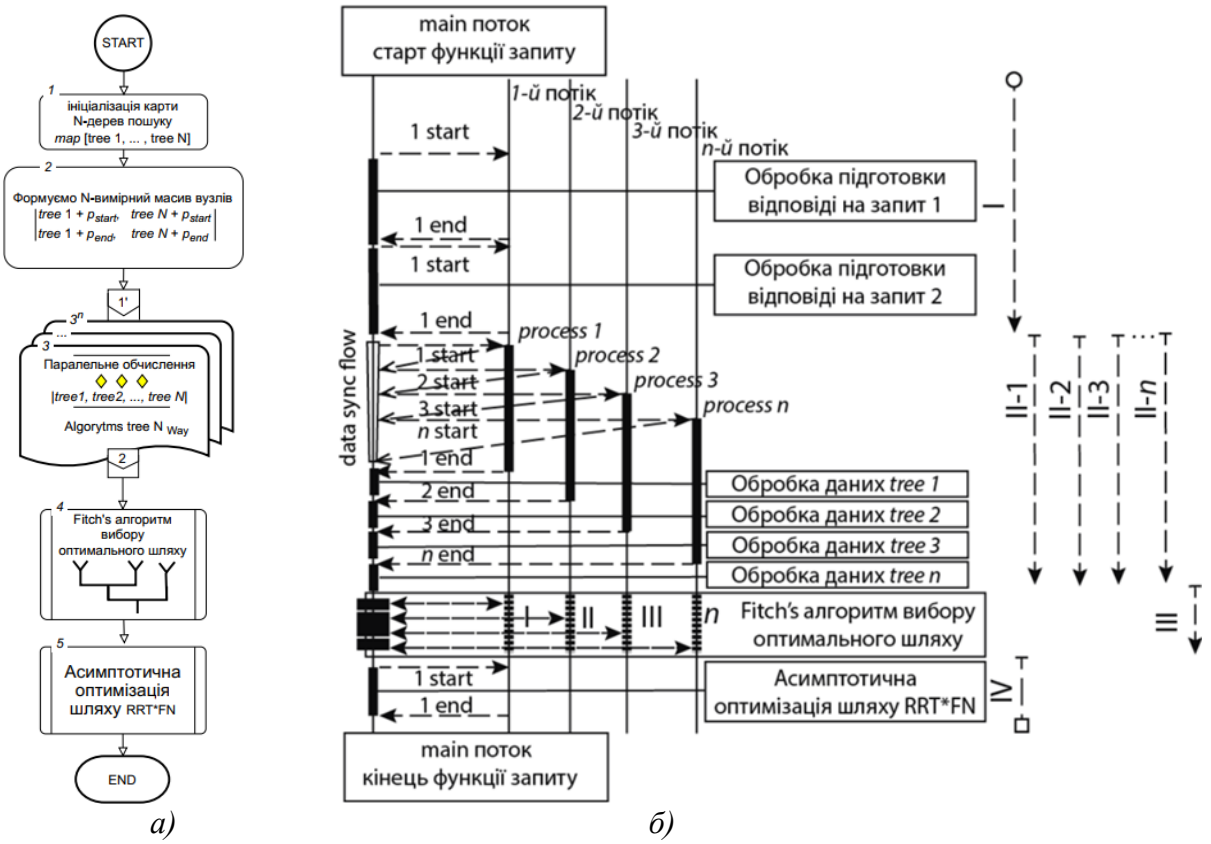


Рис. 3. Алгоритм MBD-RRT*FFT:
а – блок-схема алгоритму; *б* – функціональний часовий потоковий розріз виконання алгоритму

Опис алгоритму MBD-RRT*FFT:

Крок 1. Ініціалізуємо карту та дерева пошуку $|tree_x|$. Враховуючи, що подальший пошук планується виконувати багатопотоково в паралельних площинах, кількість дерев буде дорівнювати кількості потоків обчислення, але не менше 2, тобто $N_{tree} \vDash N_{flow} \geq 2$. Додаємо p_{start} до списку вузлів $|tree_1 ... tree_{N-1}|$ і p_{end} до списку вузлів $|tree_2 ... tree_N|$.

Крок 2. Випадковим чином вибираємо точку відбору проб p_{rand} у просторі, завдяки чому формуємо масив проб для початкової ініціалізації потоків обчислень.

Крок 3...3^n. В кожному окремому потоці вибираємо найближчий вузол $p_{nearest\_n}$ й починаємо рухатися до p_{rand_n}, зростаючи з певним кроком, і отримуємо p_{new_n}, де n – номер потоку, визначений на етапі *Крок 1*. Після чого виконується послідовність виконавчого блоку алгоритму пошуку шляху алгоритму BD-RRT*FT (див. рис. 2) [1].

Крок 4. В зв'язку з тим, що в алгоритмі BD-RRT*FT використовується асинхронна послідовна структура обчислень, для аналізу залучаються не більше 2 попередньо відібраних

вибірок шляхів, а *блок 7* (див. рис. 1) використовує метод порівняльного аналізу з вибіркою найкоротшого шляху. Але враховуючи, що модифікований алгоритм MBD-RRT*FFT на етапі находження оптимального найкоротшого шляху (*блок 4*, рис. 3) задіює багатопотокові паралельні обчислення, то використання послідовного порівняльного аналізу є недоречним і тому для аналізу *пропонується додаткова модифікація* завдяки використанню *Fitch's*-алгоритму вибору оптимального шляху із залученням в якості базового методу аналізу метод фільтрації (*filter methods*).

Крок 5. Виконуємо асимптотичну оптимізацію шляху відповідно до кроків 1–3 алгоритму RRT*FN [1; 7; 10; 14; 20].

Важливим доповненням є те, що оновлення карти після кожного завершення зростання/повторного підключення, яке ми виконували на етапі 9 алгоритму BD-RRT*FT, в цій модифікації виконується під час виконання *Кроку 3…3*^{n}. А вузли, що покрити перешкодами й вузли, лінії яких перетинаються з перешкодами, не видаляються, а маркуються як *black_list*.

Враховуючи те, що модифікація алгоритму стосується тільки методів обчислення та прикінцевого вибору оптимального шляху, а описаний в роботі [1] виконавчий елемент пошуку залишається без змін, імовірнісна повнота та асимптотична оптимальність алгоритму буде наслідувати батьківську властивість [8; 9; 18; 21; 23], а саме [1; 2]:

$$\forall p \in \sigma, p_{rand}(x) = p, x \in (0, iter). \tag{2}$$

Кількість ітерацій внаслідок певної циклічності достатньо велика, рівняння (3) залишається вірним:

$$\begin{cases} L\{p_1, p_2, \dots, p_n\} = \sigma^*, \\ \forall p \in \sigma^*, p_{rand}(x) = p, x \in (0, iter). \end{cases} \tag{3}$$

Враховуючи зазначене, додаткову перевірку та повторне доведення ймовірнісної повноти та асимптотичної оптимальності алгоритму RRT* [1; 2] в розрізі цього дослідження автор описувати не буде.

На думку автора, цікавим елементом модифікації є додавання *Fitch's*-алгоритму вибору оптимального шляху із залученням категорії *Метод фільтрації* (*filter methods*) в якості базового методу аналізу на етапі пошуку оптимального найкоротшого шляху БАНЗ (*Блок 4,* рис. 3). Доведено [1; 19; 20], що якщо фінальних вибірок дуже багато, маємо суттєве збільшення часу роботи класифікатора. Тому якщо необхідно протестувати кілька класифікаторів з метою вибору кращого, то час, необхідний для обчислення, може стати просто величезним. Для подолання цієї проблематики пропонується використати *Fitch's*-алгоритм.

Загально відомо [9; 23; 24], що метод відбору *Fitch's* поділений на три категорії: *filter methods, wrapper methods* й *embedded methods.*

В зв'язку з тим, що *метод фільтрації (filter methods)* ґрунтується на статистичних методах, і, як правило, розглядають кожну вибірку незалежно, що дозволяє оцінити і ранжувати вибірки за значимістю, за яку приймається ступінь кореляції цієї вибірки з цільової змінної, задля наших цілей будемо використовувати саме цю категорію методу.

Формула ентропії точки вибірки $tree_{rand}$ для *n*-мірної мапи визначається як:

$$H(tree(X)) = -\sum_{x_i \in X} p(tree(x_i)) \times \log_2(p(tree(x_i)), \tag{4}$$

де $p(tree(x_i))$ – ймовірність того, що змінна $tree(X)$ прийме значення $tree(x_i)$. Така можливість розглядається на *Кроці 3…3*^{n} з урахування того, що $tree(X) = tree(x_i)$ і розділене на загальну кількість умовно випадкових кроків.

Для розрахунку кореляції між змінними потрібно визначити специфічну умовну ентропію (*specific conditional entropy*), відносну ентропію (*conditional entropy*) і значення інформаційного приросту (*information gain*).

$$H(tree(X_{new}) \mid tree(X) = tree(x_i)), \quad (5)$$

де ентропія $H(Tree(X_{new}))$ визначає лише ті записи, в котрих $tree(X) = tree(x_i)$.

Відносна ентропія (*conditional entropy*) вважається як:

$$H(tree(X_{new}) \mid tree(X)) = \sum_{tree(x_i) \in tree(X)} p(tree(x_i)) \times H(tree(X_{new}) | tree(X) = tree(x_i)), \quad (6)$$

У досліджені ця величина цікавить не як окреме значення, а саме як її різниця із звичайною ентропією вибірки $tree(X_{new})$.

Нам є важливим, наскільки більш упорядкованою стає змінна $tree(X_{new})$, якщо ми знаємо значення X. Або, простіше кажучи, чи існує кореляція між значеннями X і X_{new}, і наскільки вона велика. Про це говорить величина значення інформаційного приросту (*information gain) P_{IG}*:

$$P_{IG}(X_{new}|X) = H(X_{new}) - H(X_{new}|X), \quad (7)$$

Чим більший параметр P_{IG}, тим сильніша кореляція. Таким чином, ми легко можемо обчислити величину *значення інформаційного приросту* для всіх вибірок і відкинути ті, що слабко впливають на цільову змінну. Завдяки чому планується скоротити час розрахунку шляху.

Експериментальна перевірка алгоритму. Імітаційне моделювання було здійснено завдяки розробленій програмі моделювання з використанням мови програмування Python 3+ та бібліотек PyGame, NumPy і ***Multiprocessing***. В якості системи обчислювання використовувалась система 2-сокетна Workstation із системними параметрами Intel® Xeon® Processor E5-2689 v4 (25M Cache, 3.10 GHz) =2/ 4*64 Gb DDR4 2133/ SSD 250Gb/ VA Asus GeForce GTX 1650 GDDR6 4096Mb.

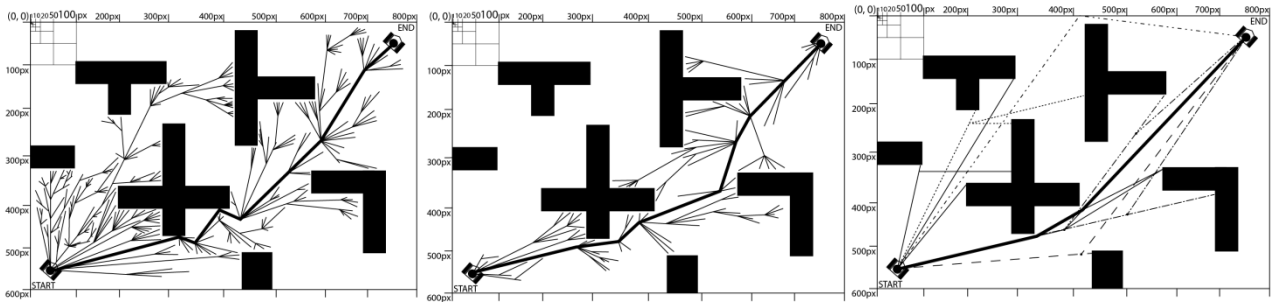
Щоб реалізувати необхідну багатопоточність в Python задля перевірки запропонованого алгоритму, було обрано бібліотеку *multiprocessing*. Цей вибір пов'язано з тим, що модуль *multiprocessing* дозволяє створювати, керувати та виконувати процеси окремо у власному просторі пам'яті. Це означає, що кожен процес може виконуватися як незалежно, так і мати залежність від результатів виконання інших процесів.

Подібно до роботи [1], задля перевірки ефективності запропонованого алгоритму проведено порівняння алгоритму MBD-RRT*FFT з іншими алгоритмами за трьома картами з оцінюванням показників продуктивності. Щоб полегшити аналіз симуляції та врахувати раціональність, перешкоди на мапах використані як блоки піксельної графіки, БАНЗ розглядається як точка піксельного розміру 4х4 px, а інші нерелевантні змінні, окрім алгоритму, контролюються, щоб бути узгодженими. Оскільки алгоритм RRT базується на випадковій вибірці, у процесі моделювання є непередбачуваність, і кожен результат вимірювання може мати відмінності, щоб усунути вплив випадковості, під час експерименту проводилось 100 незалежних експериментів для кожної ситуації та формувалися результати для порівняльного аналізу.

Перевірка роботи алгоритмів на карті 1 (Карта зі статичними перешкодами)

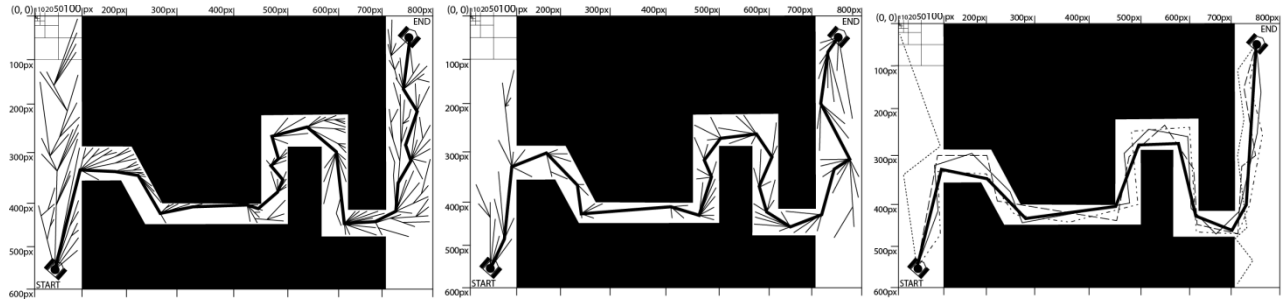
Використовуємо карту 1, розміром 800 px × 600 px, як звичайну карту статичних перешкод (рис. 4, *а*), яка використовується для перевірки швидкості реакції та індексу шляху алгоритму в нормальному середовищі перешкод. На цій карті верхній лівий кут встановлюється як початок координат [0, 0], а карта розташована в 4-му квадранті, початкова

та кінцева позиція БАНЗ [20, 580], [780, 20] позначена значками БАНЗ та надписами START, END відповідно.



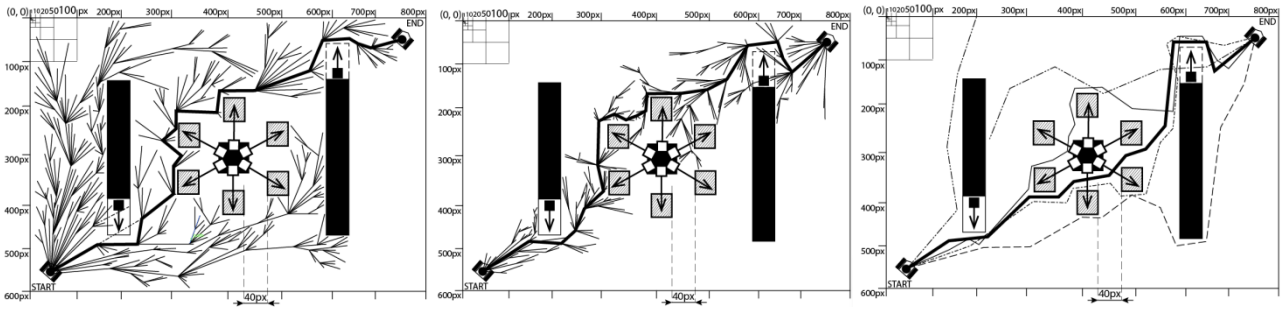
*RRT*FN* *BD-RRT*FT* *MBD-RRT*FFT*

а) візуалізація результату роботи алгоритмів на карті 1



*RRT*FN* *BD-RRT*FT* *MBD-RRT*FFT*

б) візуалізація результату роботи алгоритмів на карті 2



*RRT*DWA* *BD-RRT*FT* *MBD-RRT*FFT*

в) візуалізація результату роботи алгоритмів на карті 2

Рис. 4. Візуалізація результату роботи алгоритмів планування шляху за трьома типами карт:
а – карта 1: середовище статичних перешкод; *б* – карта 2: простір вузьких тунелів;
в – карта 3: середовище динамічних перешкод

Перевіряються алгоритми RRT*FN, BD-RRT*FT та MBD-RRT*FFT. Результати планування шляху показано на рисунку 4, *а*. Товста лінія позначає отримане рішення шляху. Інші дерева є ітераційними обчисленнями.

Як і в досліджені [1], алгоритм RRT*FN, незважаючи на двосторонність, має занадто багато надлишкових точок вибірки в процесі планування. Для інтуїтивного порівняння, середня довжина шляху рішення отримана зі 100 експериментів.

Алгоритм RRT*FN, алгоритм BD-RRT*FT і алгоритм MBD-RRT*FFT належать до алгоритмів оптимального рішення, за результатами в таблиці 1 бачимо, що багатопотоковий двонаправлений алгоритм MBD-RRT*FFT має певні переваги у швидкому пошуку рішення.

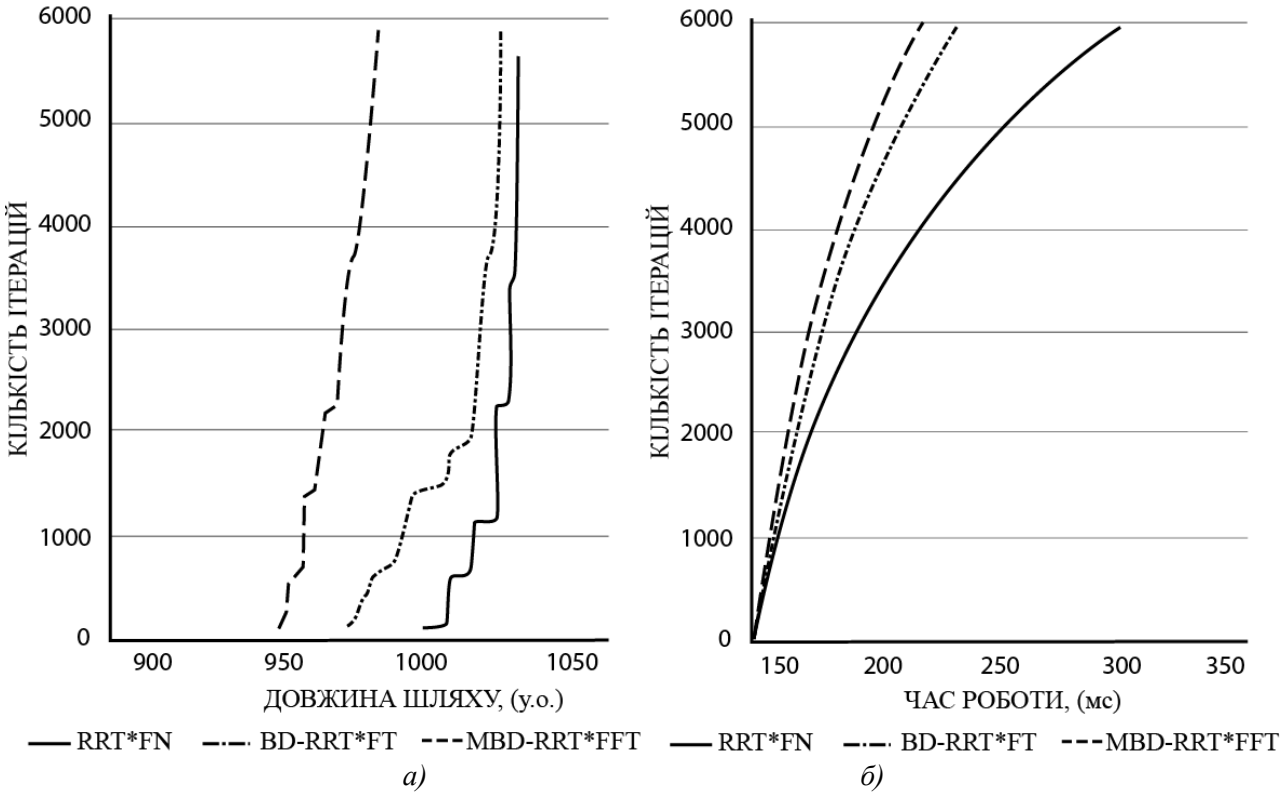
Враховуючи, що вдосконалений алгоритм використовує стратегію *Fitch’s*-вибору, загальна вартість шляху суттєво зменшується і набуває ще більш прямолінійного характеру.

На рисунку 5, *а, б* показано графічний порівняльний аналіз продуктивності кожного алгоритму на карті 1.

Таблиця 1

Дані порівняння продуктивності алгоритму на карті 1

Алгоритм	RRT*FN	BD-RRT*FT	MBD-RRT*FFT
Середня довжина шляху, *у. о.*	1032,02	1003,43	962,825
Середній час роботи, *мс*	299,4	236,6	226,9
Споживана пам’ять, *Мб*	386,53	348,51	397, 81



а) *б)*

Рис. 5. Порівняльний аналіз продуктивності кожного алгоритму на карті 1:
а – зв’язок між кількістю ітерацій та довжиною шляху;
б – зв’язок між кількістю ітерацій та часом виконання

Перевірка роботи алгоритмів на карті 2 (карта з вузьким коридором).

Використовуємо карту 2 розміром 800 px × 600 px, як карту з вузьким коридором (див. рис. 4, *б*), що імітує середовище, близьке до міського тунелю або печер. Загальні налаштування карти 2, за винятком конфігурації розташування перешкод на карті, такі самі, як і для карти 1.

Візуалізація результатів планування шляху алгоритмами RRT*FN, BD-RRT*FT та MBD-RRT*FFT наведена на рисунку 4, *б*. Як і у попередньому експерименті товста лінія позначає отримане рішення шляху. Інші дерева є ітераційними обчисленнями.

Для карт із вузьким коридором простору, через сліпоту зростання й завдяки односторонньому зростанню, RRT*FN алгоритму важко отримати точки вибірки у вільному просторі в межах вузького каналу, в результаті чого кількість ітерацій і час виконання набагато вищі, ніж у алгоритмів з двостороннім характером BD-RRT*FT та MBD-RRT*FFT, а точки

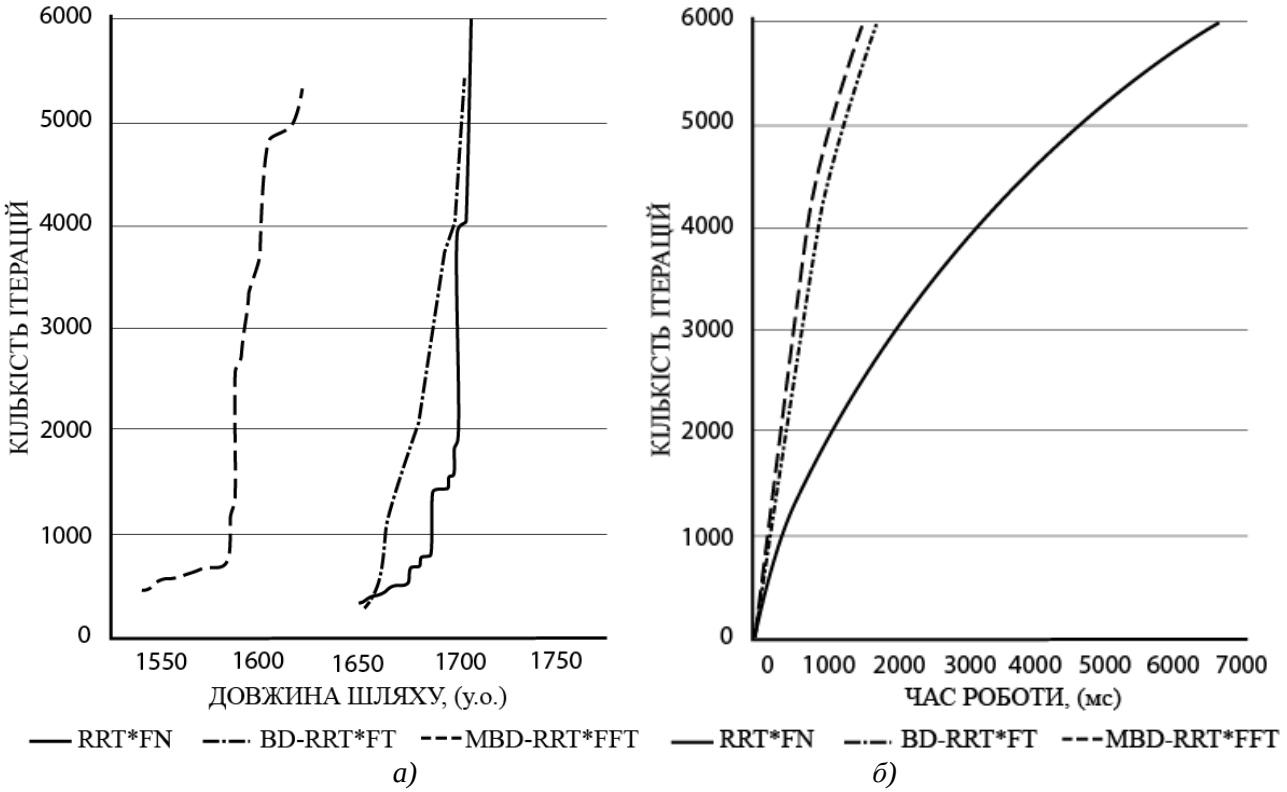
вибірки здебільшого зосереджені в лівому вільному просторі, як і в [1], що підтверджує правильність напрямку модифікації алгоритму.

В таблиці 2 наведено отримані під час експерименту результати дослідження пошуку шляху в умовах вузького середовища.

Таблиця 2

Порівняння показників ефективності алгоритму на карті 2

Алгоритм	RRT*FN	BD-RRT*FT	MBD-RRT*FFT
Середня довжина шляху, *у. о.*	1712,95	1711,99	1591,67
Середній час роботи, *мс*	6797	1955	1817
Споживана пам’ять, *Мб*	387,84	339,52	356,75

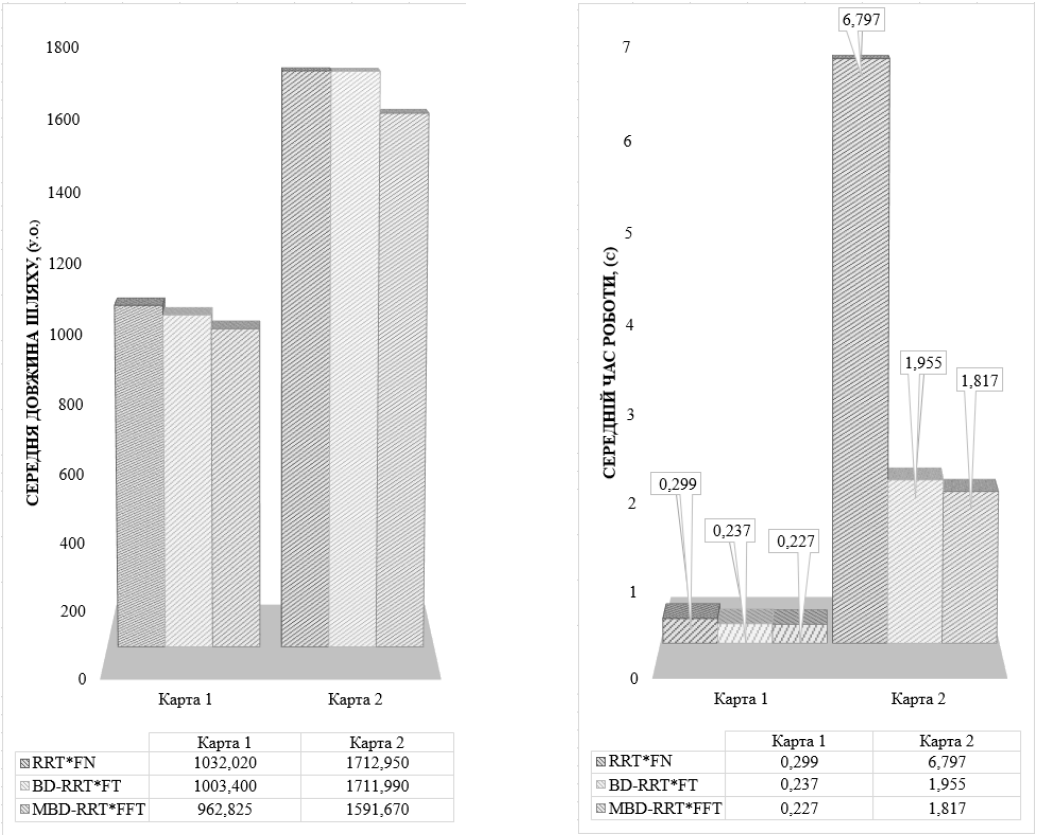


а) *б)*

Рис. 6. Порівняльний аналіз продуктивності кожного алгоритму на карті 2:
а – зв’язок між кількістю ітерацій та довжиною шляху;
б – зв’язок між кількістю ітерацій та часом виконання

Порівняння продуктивності алгоритмів пошуку оптимального рішення шляху для карти 2 та результати ітераційного процесу з урахування виконання асимптотичної оптимізації в 3 алгоритмах показано на рисунку 6, *а, б*. З чого видно, що довжина та швидкість пошуку шляху, отриманого алгоритмом MBD-RRT*FFT, кращі, ніж у алгоритмів RRT*FN і BD-RRT*FT.

При перевірці роботи методу вдосконалення алгоритму, застосовуючи карти зі статично встановленими перешкодами (карта 1 і карта 2), модифікований алгоритм MBD-RRT*FFT, маючи спадкові зв’язки з алгоритмом BD-RRT*FT, показав високу продуктивність порівняно з іншими алгоритмами (рис. 7), через що для подальшої перевірки роботи вдосконаленого алгоритму в режимі наявності динамічних об’єктів було прийнято рішення на часткову заміну набору перевірочних алгоритмів для карти 3.



а) *б)*

Рис. 7. Порівняльні діаграми продуктивності алгоритмів на карті 1 і карті 2:
а – зв’язок між кількістю ітерацій та довжиною шляху;
б – зв’язок між кількістю ітерацій та часом виконання

Перевірка роботи алгоритмів на карті 3 (карта з динамічними об’єктами).

Для подальшого вивчення продуктивності алгоритму в динамічному середовищі створено динамічну карту розміром 600 px × 800 px, що містить три рухомі об’єкти-перешкоди. Так як результати в дослідженні [1] були занадто «приємними», було прийнято рішення карту 3 зробити більш складною. Тому перший і третій об’єкти залишились подібними до карти 3 дослідження [1].

Перший X_1:[180] і третій об’єкти-перешкоди X_3:[580] візуально позначені як чорні прямокутники розміром 40 px × 200 px, що рухаються вперед-назад уздовж осі Y, а діапазон руху становить Y_1:[110–490] px і Y_2:[50–500] px відповідно. Другий об’єкт-перешкода має складний характер, а саме блок-перешкода розміром 40 px × 40 px (центр X_{20}:[400], Y_{20}:[300]) розпадається на шість незалежних рівновіддалених об’єктів з радіусом центрів 100 px.

В якості порівняльних алгоритмів вибрано алгоритм BD-RRT*FT та алгоритм RRT*DWA. Оскільки в динамічному середовищі довжина рішення шляху змінюється нерегулярно, то задля спрощення аналізу приймемо, що лише довжина шляху переміщення динамічної перешкоди буде впливати на глобальний шлях БАНЗ. Середній час виконання побудови шляху та довжина наведені в таблиці 3, графічна візуалізація результатів планування шляху показані на рисунку 4, *в*, а порівняння продуктивності алгоритмів пошуку оптимального рішення шляху для карти 3 та результати ітераційного процесу з урахування виконання асимптотичної оптимізації в 3 алгоритмах показано на рисунку 8, *а*, *б*.

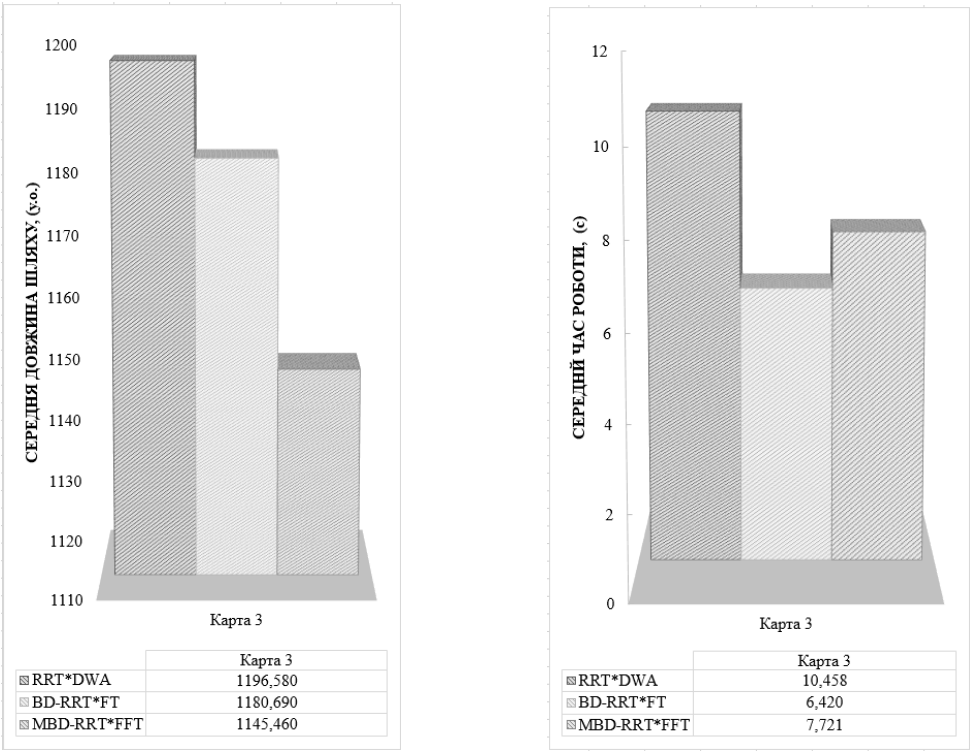
Таблиця 3

Порівняння показників ефективності алгоритму на карті 3

Алгоритм	RRT*DWA	BD-RRT*FT	MBD-RRT*FFT
Середня довжина шляху, *у. о.*	1196,58	1180,69	1145,46
Середній час роботи, *мс*	10458,4	6420,2	7720,9
Споживана пам’ять, *Мб*	470,66	387,04	433,14

На рисунку 4, *в* явно помітна наявна властивість всіх алгоритмів можливості планування шляху в реальному часі та з корекцією шляху задля уникнення динамічних (рухомих) об’єктів-перешкод.

Відомо [7; 8; 10; 16; 21], що алгоритм RRT*DWA має початкову спадкову властивість планування шляху від алгоритмів RRT і RRT*, що саме й використовують для глобального планування шляху на перших етапах, і тільки потім, під час руху БАНЗ, використовують алгоритм DWA.



а) *б)*

Рис. 8. Порівняльна діаграма продуктивності алгоритмів з динамічними об’єктами на карті 3:
а – зв’язок між кількістю ітерацій та довжиною шляху;
б – зв’язок між кількістю ітерацій та часом виконання

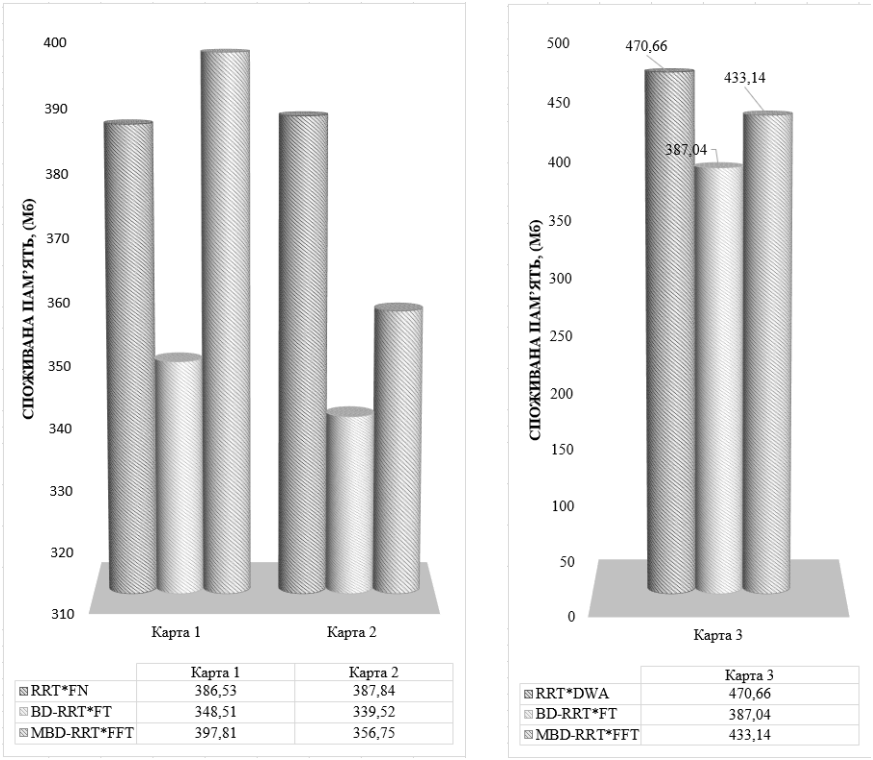
Локальне планування шляху використовується для досягнення динамічного уникнення перешкод, але через це легко впасти в локальну оптимальність, а середня довжина шляху та глобальний час планування шляху довші, ніж запропонований автором модифікований алгоритм BD-RRT*FT в роботі [1]. Вдосконалений алгоритм MBD-RRT*FFT, маючи загальну спадковість пошуку від алгоритму BD-RRT*FT, має загальну тенденцію до того, що запланований шлях може бути тимчасово поганим через те, що деякі вузли вибірки відкидаються й маркуються в *black_list*, після того, як шлях знищено перешкодами. Але завдяки використанню багатопотокового обчислення експеримент реально показав, що вдосконалений алгоритм MBD-RRT*FFT має кращі можливості динамічного планування,

а використання *Fitch’s*-алгоритму вибору оптимального результату пошуку надає загальній тенденції прагнення до *вибору більш прямолінійного оптимального шляху*.

Як і в дослідженні [1] під час проведення експериментальної частини, на всіх етапах перевірки проводився контроль навантаження на пам’ять системи прийняття рішення.

Середні результати споживання при роботі всіх алгоритмів показані в таблицях 1–3.

На рисунку 9. представлено порівняльні діаграми споживання об’єму пам’яті алгоритмами для всіх карт, що досліджувалися. Ми бачимо, що для вдосконаленого алгоритму присутнє *підвищене споживання пам’яті*, що свідчить про використання *багатопоточності* при виконанні обчислень, але враховуючи отримані результати щодо оптимальності та швидкості обчислень, ми розуміємо, що, порівнюючи з батьківським алгоритмом, приріст споживання в 12–15 % можна *вважати незначним*, але необхідно враховувати при виборі апаратної складової керуючого контролера та систем прийняття рішень.



	Карта 1	Карта 2
RRT*FN	386,53	387,84
BD-RRT*FT	348,51	339,52
MBD-RRT*FFT	397,81	356,75

	Карта 3
RRT*DWA	470,66
BD-RRT*FT	387,04
MBD-RRT*FFT	433,14

а) *б)*

Рис. 9. Порівняльна діаграма ефективності завантаження пам’яті системи прийняття рішень при виконанні пошуку шляху алгоритмами:
а – робота алгоритмів на картах 1 і 2; *б* – робота алгоритмів на картах 3

Висновки

У задачах навігації рухомих об’єктів одним із головних напрямів є вирішення проблеми планування шляху переміщення роботів і підвищення точності та надійності їх наведення у режимі реального часу. Розуміючи складність завдань під час проведення бойових дій в урбанізованому просторі щільної забудови міста з постійною зміною ландшафту, такі завдання, що покладаються на БАНЗ, мають тенденцію до постійного ускладнення.

Задля вирішення цієї важливої практичної проблеми з урахуванням проведеного дослідження [1] на основі отриманої аналітики було запропоновано і проведено вдосконалення модифікації асимптотично оптимального алгоритму BD-RRT*FT та розроблено алгоритм MBD-RRT*FFT, який при застосуванні в динамічних середовищах, не дивлячись на загальну спадковість алгоритму BD-RRT*FT, має загальну тенденцію до того,

що запланований шлях може бути тимчасово поганим, але завдяки використанню багатопотокового обчислення має кращі можливості динамічного планування, а використання *Fitch's*-алгоритму вибору оптимального результату пошуку надає загальну тенденцію до вибору оптимального прямолінійного шляху.

Експериментально доведено ефективність алгоритму MBD-RRT*FFT задля задоволення вимог планування шляху в реальному часі, дозволяючи БАНЗ швидко отримати оптимальний шлях без зіткнень у *складних динамічних* середовищах у режимі реального часу.

Важливо зауважити, що завдяки використанню в алгоритмі MBD-RRT*FFT багатопоточності з'являється підвищене споживання пам'яті в 12–15 %, але враховуючи отримані результати щодо оптимальності та швидкості обчислень, ми розуміємо, що, порівнюючи з батьківським алгоритмом, такій приріст можна вважати незначним, але необхідним для розуміння при виборі апаратної складової керуючого контролера та систем прийняття рішень

Напрями подальших досліджень

Враховуючи отримані результати дослідження, в подальшому планується розробити й описати *методику* застосування вдосконаленого модифікованого алгоритму MBD-RRT*FFT для застосування в реальних умовах середовища з динамічними перешкодами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бернацький А. П. Метод планування шляху автономного наземного робота з використанням модифікації динамічного двонаправленого RRT-алгоритму // Системи і технології зв'язку, інформатизації та кібербезпеки. 2023. № 4. С. 16–31. DOI: 10.58254/viti.4.2023.02.16.

2. Бернацький А. П. Основи робототехніки військового призначення / А. П. Бернацький, І. В. Панченко, О. І. Восколович. Київ: ВІТІ, 2021. 496 с.

3. Бернацький А. П., Панченко І. В., Восколович О. І. Розширена математична модель руху автономного наземного робота розвідника в умовах бойових дій в урбанізованому просторі // Озброєння та військова техніка. 2021. № 30 (2). С. 121–129.

4. Бідюк П. І. Системи і методи підтримки прийняття рішень / П. І. Бідюк, О. Л. Тимощук. Київ: КПІ ім. Ігоря Сікорського, 2022.

5. Про нарощування спроможностей сил оборони: Указ Президента України від 06.02.2024 № 51/2024. URL: https://www.president.gov.ua/documents/512024-49625.

6. Про рішення Ради національної безпеки і оборони України від 25 березня 2021 року «Про Стратегію воєнної безпеки України»: Указ Президента України від 21.03.2021 № 121.

7. Adiyatov, Olzhas; Varol, Huseyin Atakan. A novel RRT-based algorithm for motion planning in Dynamic environments. In Mechatronics and Automation (ICMA), 2017 IEEE International Conference on, 2017. P. 1416–1421. URL: https://doi:10.1109/ICMA.2017.8016024.

8. Black, Paul E. Greedy algorithm. Dictionary of Algorithms and Data Structures, US National Institute of Standards and Technology* PE Black. 2010. URL: http://www.nist. gov/dads/HTML/greedyalgo.html.

9. Cormen T., Leiserson C., Rivest R., Stein C. Introduction to Algorithms 3rd ed. Cambridge, London: The MIT Press, 2009. 1312 p.

10. Gammell J. D., Srinivasa S. S., Barfoot T. D. Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic // Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems; Chicago, IL, USA. 14–18 September. 2014. URL: https://doi.org/10.1109/IROS.2014.6942976.

11. Jeong I. B., Lee S. J., Kim J. H. Quick-RRT*: Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate. Expert Syst. Appl. – 2019; 123. P. 82–90. URL: https://doi:10.1016/j.eswa.2019.01.032.

12. Kadry S., Alferov G., Fedorov V. D-Star Algorithm Modification // International Journal of Online and Biomedical Engineering (iJOE). 2020. Vol. 16, No. 8. P. 108–113. URL: https://doi.org/10.3991/ijoe.v16i08.14243.

13. Kagan E., Ben-Gal I. A Group-Testing Algorithm with Online Informational Learning // IIE Transactions (Institute of Industrial Engineers). No. 46 (2). P. 164–184. URL: https://doi.org/10.1080/0740817X.2013.803639.

14. Klemm S., Oberländer J., Hermann A., Roennau A., Schamm T., Zollner J. M., Dillmann R. RT-Connect: Faster, asymptotically optimal motion planning // 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO 2015) At: Zhuhai, China. Volume 12. URL: https://doi.org/10.1109/ ROBIO.2015.7419012.

15. Li B., Chen B. An Adaptive Rapidly-Exploring Random Tree // IEEE/CAA Journal of Automatica Sinica. 2021. URL: https://doi.org/10.1109/JAS.2021.1004252.

16. Majeed A, Hwang SO. Path planning method for UAVs based on constrained polygonal space and an extremely sparse waypoint graph // Applied Sciences. 2021. No. 11 (12). P. 5340. URL: https://doi.org/10.3390/app11125340.

17. Maw A. A., Tyan M., Nguyen T. A. et al. iADA*-RL: Anytime graph-based path planning with deep reinforcement learning for an autonomous UAV // Applied Sciences. 2021. No. 11 (9). P. 1–18. URL: https://doi.org/10.3390/app11093948.

18. Schapire R., Freund Y. Boosting: Foundations and Algorithms. MIT, 2012.

19. Spanogianopoulos, Sotirios and Sirlantzis, Konstantinos Non-holonomic Path Planning of Car-like Robot using RRT*FN // In: 2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI). 2015 IEEE, pp. 53–57. URL: https://doi:10.1109/URAI.2015.7358927.

20. Taheri E., Ferdowsi M. H., Danesh M. Fuzzy greedy RRT path planning algorithm in a complex configuration space // Int. J. Control. Autom. Syst. 2018. No. 16. P. 3026–3035. URL: https://doi:10.1007/ s12555-018-0037-6.

21. Thomas H. Corman, Charles I. Leiserson, Ronald L. Rivest, Clifford Stein. Algorithms: construction and analysis. Introduction to Algorithms. Michigan U.: Williams, 2006.

22. Winnfield J. A. Unmanned Systems Integrated Roadmap FY2011-2036 / Winnfield J. A. Jr., Kendall F. Washington, DC: U.S. Department of Defense, March 9, 2012.

23. Wirth N. Algorithms + Data Structures = Programs (Prentice-Hall Series in Automatic Computation) / N. Wirth. Hoboken: Prentice Hall, 1976. 366 p.

24. Zhang L., Shen J., Yang J., Li G. Analyzing the fitch method for reconstructing ancestral states on ultrametric phylogenetic trees // Bulletin of Mathematical Biology. 2010. P. 1760–1782. DOI: 10.1007/s11538-010-9505-8.