**DOI: 10.58254/viti.6.2024.02.24**

УДК 621.3:623.4:629.07-044.4

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

# СУЧАСНІ МЕТОДИ І АЛГОРИТМИ ПЛАНУВАННЯ ШЛЯХУ ДЛЯ АВТОНОМНИХ МОБІЛЬНИХ РОБОТІВ, ЇХ ЦІЛЬОВІ ФУНКЦІЇ

*Сучасна ситуація в світі алгоритмів дослідження планування шляху автономними роботами показує, що існує велика кількість алгоритмів та їх модифікацій, які вимагають від дослідників та інженерів автономної робототехніки тонкого розуміння фундаментальних аспектів їх функціонування. Загальною метою дослідження є проведення аналізу існуючих алгоритмів, методів та їх модифікацій з точки зору складової математичного апарата, а саме їх цільових функцій.*

*В дослідженні розглянуто 38 сучасних алгоритмів і методів планування шляху автономними роботами. Розглянута важливість і критичність розуміння цільової функції для галузі робототехніки. Наведені приклади, що ілюструють важливість цього аспекту. В процесі дослідження алгоритми представлені в виді формалізованих узагальнених математичних формул з урахуванням можливих мінімізації/максимізації, їх вдосконалень і покращень.*

*До кожного розглянутого алгоритму і методу наданий висновок стосовно його цільової функції.*

*У дослідженні окремо виділено методи "Динамічних ігор" та розглянуто застосування фундаментального "Методу розв'язуючих функцій А. О. Чикрія" для планування шляху.*

*В загальних висновках, надана інформація переваг та важливості розуміння математичного апарата алгоритмів пошуку шляху автономними роботами задля вирішення наукових завдань. Узагальнені результати зведені до єдиної порівняльної таблиці, що надає формалізовану основну функцію алгоритмів, розкриває основну ідею, переваги/недоліки, сферу застосування і приклад використання. Порівняльна таблиця представлена в формі узагальненого допоміжного довідкового елемента дослідження.*

***Ключові слова:*** *алгоритми, дослідження, автономний робот, методи, планування шляху, ЦФ, оптимізація.*

***A. Bernatskyi Modern methods and algorithms of path planning of autonomous robots and their objective functions.***

*The current situation in the world of research algorithms for path planning by autonomous robots shows that there are a large number of algorithms and their modifications that require researchers and engineers of autonomous robotics to have a fine understanding of the fundamental aspects of their functioning. The general goal of the research is to conduct an analysis of existing algorithms, methods and their modifications from the point of view of the component of the mathematical apparatus, namely their target functions.*

*The research examines 38 modern algorithms and methods of path planning by autonomous robots. Considered the importance and criticality of understanding the objective function for the field of robotics. Examples illustrating the importance of this aspect are given. In the process of research, algorithms are presented in the form of formalized generalized mathematical formulas taking into account possible minimization/maximization, their improvements and improvements.*

*For each considered algorithm and method, a conclusion is given regarding its target function.*

*In the study, the methods of "Dynamic games" are separately highlighted and the application of the fundamental "Method of solving functions of A.O. Chikryi" for path planning is considered.*

*In general, information on the advantages and importance of understanding the mathematical apparatus of path-finding algorithms by autonomous robots for solving scientific problems is provided. The generalized results are summarized in a single comparative table, which provides the main formalized function of the algorithms, reveals the main idea, advantages/disadvantages, the scope of application and an example of use. The comparative table is presented in the form of a generalized auxiliary reference element of the study.*

***Keywords:*** *algorithms, autonomous robot, methods, path planning, research, objective function, optimization*

**Постановка завдання.** На сучасному етапі розвитку інформаційних технологій, інформаційне поле заповнено й перенасичено дослідженнями, порівняльними аналізами алгоритмів побудови шляху мобільними роботами (МР) [1].

Розуміння цільової функції є критично важливим при дослідженні алгоритмів пошуку шляху, оскільки вона визначає, що саме алгоритм намагається оптимізувати. Цільова функція (ЦФ) може включати різні критерії, такі як мінімізація відстані, часу, витрат або ризиків [2].

Наведемо кілька прикладів, які ілюструють важливість цього аспекту [1]:

– *Мінімізація відстані*. У класичному алгоритмі А* ЦФ зазвичай спрямована на мінімізацію відстані між початковою та кінцевою точками. Це досягається шляхом використання евристичної функції, яка оцінює відстань від поточної точки до цільової. Якщо ЦФ правильно визначена, алгоритм ефективно знаходить найкоротший шлях;

– *Логістика.* В логістиці ЦФ може включати мінімізацію витрат на транспортування або часу доставки. Наприклад, при плануванні маршруту для доставки товарів, алгоритм повинен враховувати не лише відстань, але й інші фактори, такі як дорожні умови, трафік та витрати на паливо. Вибір правильної цільової функції дозволяє оптимізувати ці параметри і забезпечити ефективну доставку;

– *Військові транспортні системи.* У військових транспортних системах, таких як планування військової операції з задіянням багатомодової складової автономних мобільних роботів (АМР), ЦФ може включати мінімізацію зіткнення з уникненням утворення заторів в урбанізованому середовищі та оптимізацію потоку дронів. Алгоритми повинні враховувати різні фактори, такі як час доби, погодні умови, вплив РЕБ противника, засідки. Вибір відповідної цільової функції дозволяє ефективно керувати військовим трафіком і зменшувати час виконання завдання АМР;

– *Відеоігри.* У відеоіграх ЦФ може бути більш складною і включати не лише мінімізацію відстані, але й уникнення перешкод та ворогів. Наприклад, у стратегіях реального часу алгоритми пошуку шляху повинні враховувати динамічні зміни на карті, такі як рухомі об'єкти або змінні перешкоди. Правильне визначення цільової функції дозволяє персонажам гри ефективно досягати своїх цілей, уникаючи небезпек;

– Ройові алгоритми й алгоритми оптимізації мурашиної колонії *(ant colony optimization – ACO).* АСО використовує цільову функцію для знаходження найкоротшого шляху, імітуючи поведінку мурах. Мурахи залишають феромонові сліди на шляху до їжі, і інші мурахи схильні слідувати цим слідам, що призводить до знаходження оптимального шляху. ЦФ в цьому випадку включає мінімізацію відстані та максимізацію феромонових слідів.

Розуміння *цільових функцій алгоритмів* та їх порівняння знаходження оптимального шляху з точки зору вирішення наукових завдань, надає багато важливих переваг, а саме:

– *Оцінка ефективності.* Порівняння цільових функцій дозволяє оцінити, наскільки ефективно різні алгоритми виконують свої завдання в різних умовах;

– *Вибір оптимального алгоритму.* Різні задачі можуть вимагати різних підходів до оптимізації. Порівняння цільових функцій допомагає визначити, який алгоритм найкраще підходить для конкретної задачі;

– *Виявлення сильних і слабких сторін.* Аналіз цільових функцій дозволяє виявити сильні та слабкі сторони кожного алгоритму. Це може включати виявлення ситуацій, в яких алгоритм працює неефективно, або умов, за яких він дає найкращі результати;

– *Поліпшення алгоритмів.* Порівняння цільових функцій може вказати на можливості для поліпшення існуючих алгоритмів. Це може включати модифікацію цільової функції або інтеграцію нових евристик для підвищення ефективності;

– *Теоретичний внесок.* Наукове порівняння цільових функцій сприяє розвитку теорії алгоритмів та оптимізації. Це дозволяє краще розуміти фундаментальні принципи, які лежать в основі роботи алгоритмів, і може призвести до відкриття нових методів та підходів до вирішення задач оптимізації.

**Аналіз останніх публікацій.** Відомо багато досліджень та порівнянь алгоритмів побудови шляху. Так, наприклад в дослідженні [3], Ші Вей Лі (Shi Wei Li) аналізує оптимізацію цільової функції на основі класичної оптимізації рою частинок для двовимірного планування шляху. В дослідженні [4], Каур Каріндер (Harinder, Kaur Sidhu) оцінює продуктивність різних алгоритмів пошуку шляху, зокрема, як вони оптимізують цільову функцію для знаходження найкоротшого маршруту на заданій карті. В роботі [5], Самрид Гарг

(Samridh Garg) й Бхану Деві (Bhanu Devi) пропонують модифікований алгоритм Дейкстри з адаптивною штрафною функцією для знаходження оптимального найкоротшого шляху між початковою та кінцевою точками. Аналітичне дослідження [6], Стефана Вейсер (Stephan Weiser), Ганса Вулфа (Hans Wulf) і Йорна Іхлеманна (Jörn Ihlemann), демонструє застосування алгоритму найглибшого шляху для кращого розуміння ландшафтів цільової функції. Також відомі порівняльні дослідження алгоритмів пошуку шляху [7], в якому автори Матвійчук Р. Д. і Данильчук О. М. порівнюють деякі алгоритми на рівні псевдо-кодової конструкції. В дослідженні [8] А. А. Проценко і В. Г. Іванов розглядають 35 алгоритмів пошуку шляху та методів оптимізації на понятійному рівні карт зі словесним описом виконання дії алгоритмом. Взагалі основна маса досліджень націлені на розгляд основних проблем, що виникають під час виконання задачі пошуку шляхів АМР.

Але важливим елементом будь-якого алгоритму є в першу чергу математична складова, а саме його цільова функція та її трансформації.

**Метою статті** є наукове дослідження з *урахуванням математичного апарата цільових функцій,* існуючих методів і алгоритмів, що дозволяють МР виконувати завдання в автономному режимі.

**Виклад основного матеріалу:** велика кількість сучасних алгоритмів та їх модифікацій вимагають від дослідників та інженерів автономної робототехніки тонкого розуміння фундаментальних аспектів їх функціонування. Для чого розглянемо найбільш відомі сучасні алгоритми та модифікації, а саме: VCD, BCD, Morse decomposition, Exhaustive path planning with exact cell decomposition, Approximate cell decomposition, Sensor path planning with approximate cell decomposition, QB decomposition, FQD, K-Framed quadtrees decomposition, Adaptive decomposition, Probabilistic cell decomposition, Probabilistic cell decomposition with harmonic functions, Ariadne's clew, Expansive configuration spaces, PRM, RRM, Lazy PRM, Gaussian sampling for PRM, Halton sampling for PRM, D*, Goal-directed and randomized search, SANDROS, APF, DAPF, IAPF, Artificial potential field based subgoal network, Dynamic subgoal path planner, Hierarchical motion planner, Two-layered subgoal algorithm, RRT with local trees, Obstacle-based RRT, RRT-connect, Bi-RRT, RRT*, Informed-RRT*, MBD-RRT*FFT, Динамічні ігри та Метод розв'язуючих функцій А.О. Чикрія для планування шляху. Проведемо їх детальний опис і зазначимо основні характеристики, переваги та недоліки[9–85].

1. Алгоритм *вертикального клітинного розбиття* (Vertical Cellular Decomposition) розбиває простір на вертикальні клітини, що дозволяє створити карту для планування шляху. Основні переваги включають простоту реалізації та високу точність, але можливі проблеми з масштабованістю та високі вимоги до обчислювальних ресурсів [9, 10]. ЦФ для цього алгоритму може бути виражена як математична формула (1.1), яка мінімізує або максимізує певний параметр, пов'язаний з розбиттям простору.

Зазвичай, ЦФ для таких алгоритмів може бути пов'язана з мінімізацією кількості клітин або з максимізацією ефективності покриття простору.

$$\text{Minimize } \sum_{i=1}^{n} A_i, \tag{1.1}$$

де $A_i$ – площа *i*-тої клітини, а $n$ – загальна кількість клітин.

Таким чином, розуміючи, що обчислювальна складність алгоритму залежить від кількості перешкод у просторі і може бути значною, особливо для просторів з великою кількістю перешкод. Бачимо спрямованість цільової функції на мінімізацію довжини шляху та часу виконання, що робить його ефективним для багатьох практичних застосувань, але вимагає оптимізації для роботи з великими даними.

2. Алгоритм *Boustrophedon Cellular Decomposition* (BCD) розбиває простір на клітини, які покриваються простими зворотно-поступальними рухами. Це забезпечує ефективне покриття простору та простоту реалізації, але можливі проблеми з масштабованістю та високі вимоги до обчислювальних ресурсів [11].

ЦФ цього алгоритму може бути виражена як математична формула, яка мінімізує певні параметри, такі як загальна довжина шляху або кількість переходів між клітинами.

ЦФ (2.1) що мінімізує загальну довжину шляху для алгоритму BCD має вигляд:

$$\text{Minimize } \sum_{i=1}^{n}(L_i + T_i), \tag{2.1}$$

де $L_i$ – довжина шляху для покриття і-тої клітини, $T_i$ – довжина переходу від і-тої клітини до $(i+1)$-тої клітини, $n$ – загальна кількість клітин.

Таким чином, ЦФ для алгоритму Boustrophedon Cellular Decomposition може бути виражена як математична формула, яка мінімізує загальну довжину шляху або кількість переходів між клітинами, залежно від конкретних вимог задачі.

3. Алгоритм *Morse Decomposition* використовує теорію Морзе для розбиття простору на клітини, що дозволяє аналізувати динамічні системи шляхом розбиття фазового простору на інваріантні множини, які називаються Morse sets. Основні переваги включають глибокий математичний підхід та можливість аналізу топології, але алгоритм складний у реалізації та має високі вимоги до обчислювальних ресурсів [12, 13, 14].

Цільову функцію можемо виразити як математичну формулу (3.1), що пов'язана з мінімізацією складності розбиття або з максимізацією точності виявлення інваріантних множин. тобто мінімізує або максимізує певний параметр, пов'язаний з розбиттям простору.

Цільова функція, яка мінімізує суму відхилень від інваріантних множин має вигляд (3.1):

$$\text{Minimize } \sum_{i=1}^{n}\int_{M_i} \parallel \dot{x} - f(x)^2 \parallel dx, \tag{3.1}$$

де $M_i$ – $i$-та Morse множина, $\dot{x}$ – похідна траєкторії, $f(x)$ – векторне поле динамічної системи, $n$ – загальна кількість Morse множин.

Таким чином, ЦФ для алгоритму Morse Decomposition може бути виражена як математична формула, яка мінімізує суму відхилень від інваріантних множин або кількість Morse множин, залежно від конкретних вимог задачі.

4. Алгоритм *планування шляхів з використанням точного розбиття на клітини* (Exhaustive Path Planning with Exact Cell Decomposition) забезпечує вичерпне планування шляху з точним розбиттям простору на клітини, щоб знайти оптимальний шлях від початкової до кінцевої точки, уникаючи перешкод. Це гарантує високу точність планування, але алгоритм складний у реалізації та має високі вимоги до обчислювальних ресурсів [15, 16].

ЦФ для цього алгоритму може бути виражена як математична формула, яка мінімізує певний параметр, наприклад, загальну довжину шляху або час проходження.

ЦФ, яка мінімізує загальну довжину шляху, мінімізує суму відстаней між послідовними клітинами на шляху, що дозволяє знайти найкоротший шлях через розбитий простір, має вигляд (4.1):

$$\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}), \tag{4.1}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $n$ – загальна кількість клітин на шляху.

ЦФ, що мінімізує час проходження шляху AMP, враховує як відстань, так і швидкість руху в кожній клітині, це дозволяє мінімізувати загальний час проходження шляху (4.2):

$$\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i}, \tag{4.2}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $v_i$ – швидкість руху в клітині $C_i$, $n$ – загальна кількість клітин на шляху.

Таким чином, ЦФ для алгоритму планування шляхів з використанням точного розбиття на клітини може бути виражена як математична формула, яка мінімізує загальну довжину шляху або час проходження, залежно від конкретних вимог задачі.

5. Алгоритм *Approximate Cell Decomposition* розбиває простір на регулярні клітини, такі як прямокутники або квадрати з метою спрощення задачі планування шляху. Основні переваги включають швидкість обчислень та простоту реалізації, але точність менша порівняно з точними методами, і можливі помилки через наближення [17].

ЦФ для цього алгоритму, яка мінімізує загальну довжину шляху, мінімізує суму відстаней між послідовними клітинами на шляху, що дозволяє знайти найкоротший шлях через розбитий простір має вигляд:

$$\mathcal{OCB}_j[k^u] = \bigcup_{t=1}^{k_u}\{\ \mathcal{CB}_j[[x_k, x'_k]\cdot[y_k, y'_k]\cdot(\gamma_k + l \triangle \theta)]\}, \quad (5.1)$$

де $\mathcal{CB}_j$ – відстань між центрами клітин $x_k, y_k$ та $\gamma_k + l \triangle \theta$, $k_u$ – загальна кількість клітин на шляху.

ЦФ що мінімізує кількість клітин, через які проходить шлях, спрямована на зменшення кількості клітин, через які проходить шлях, що може бути корисним для зменшення складності обчислень:

$$\mathcal{ICT}_j[k^u] = \bigcup_{t=1}^{k_u}\{\ \mathcal{CT}_j[[x_k, x'_k]\cdot[y_k, y'_k]\cdot(\gamma_k + l \triangle \theta)]\}, \quad (5.2)$$

де $k_u$ – загальна кількість клітин на шляху.

Таким чином, ЦФ для алгоритму Approximate Cell Decomposition може бути виражена як математична формула, яка мінімізує загальну довжину шляху або кількість клітин, через які проходить шлях, залежно від конкретних вимог задачі.

6. Алгоритм планування шляху сенсора з використанням наближеного розбиття на клітини (Sensor Path Planning with Approximate Cell Decomposition) використовує метод декомпозиції простору, за рахунок розбивки конфігураційного простору на дискретні клітини для планування шляху сенсором. Цей підхід дозволяє ефективно працювати в складних середовищах з різними типами перешкод, але має високі вимоги до обчислювальних ресурсів і потребує складних налаштування параметрів [18, 19].

ЦФ для цього алгоритму виражена як математична формула, що мінімізує певний параметр, такий як, загальна довжина шляху, час проходження або кількість клітин, через які проходить шлях.

ЦФ, яка мінімізує загальну довжину шляху сенсора має вигляд (6.1):

$$\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}), \quad (6.1)$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $n$ – загальна кількість клітин на шляху.

ЦФ враховує як відстань, так і швидкість руху сенсора в кожній клітині, що дозволяє мінімізувати загальний час проходження шляху (6.2):

$$\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i}, \quad (6.2)$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $v_i$ – швидкість руху в клітині $C_i$, $n$ – загальна кількість клітин на шляху.

Таким чином, ЦФ для алгоритму планування шляху сенсора з використанням наближеного розбиття на клітини може бути виражена як математична формула, яка мінімізує загальну довжину шляху (6.1), час проходження (6.2) або кількість клітин, через які проходить шлях, залежно від конкретних вимог задачі.

7. Алгоритм *Quadtree-based decomposition* використовує декомпозицію простору на квадрати, з метою аналізу, зберігання або обробки даних, що дозволяє ефективно працювати у високовимірних просторах. Основні переваги включають простоту реалізації, але можливі проблеми з масштабованістю та необхідність налаштування параметрів [20, 21].

ЦФ для цього алгоритму мінімізує певний параметр, такий як неоднорідність даних у квадрантах або кількість розподілу.

Мінімізація неоднорідності даних у квадрантах:

$$\text{Minimize } \sum_{i=1}^{n} Var(Q_i), \quad (7.1)$$

де $Var(Q_i)$ – дисперсія даних у $i$-му квадранті $Q_i$, $n$ – загальна кількість квадрантів.

Таким чином, ЦФ для алгоритму Quadtree-based decomposition є математична формула, яка мінімізує неоднорідність даних у квадрантах (7.1) або кількість розбиттів, залежно від конкретних вимог задачі.

8. Алгоритм *Framed-Quadtree Decomposition* використовується для ефективного розбиття простору на клітини з метою планування шляху є вдосконаленою версією quadtree, що зменшує кількість вузлів за рахунок використання рамок. Це підвищує ефективність, але алгоритм складніший у реалізації і має високі вимоги до обчислювальних ресурсів [22, 23, 24].

ЦФ для цього алгоритму мінімізує певний параметр, такий як, загальна довжина шляху або час проходження.

8.1. Мінімізація загальної довжини шляху:

$$\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}), \tag{8.1}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $n$ – загальна кількість клітин на шляху.

Ця ЦФ мінімізує суму відстаней між послідовними клітинами на шляху, що дозволяє знайти найкоротший шлях через розбитий простір.

8.2. Мінімізація часу проходження враховує як відстань, так і швидкість руху в кожній клітині, що дозволяє мінімізувати загальний час проходження шляху:

$$\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i}, \tag{8.2}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $v_i$ – швидкість руху в клітині $C_i$, $n$ – загальна кількість клітин на шляху.

Таким чином, ЦФ для алгоритму Framed-Quadtree Decomposition може бути виражена як математична формула, яка мінімізує загальну довжину шляху (8.1), час проходження (8.2) або кількість клітин (8.3), через які проходить шлях, залежно від конкретних вимог задачі.

9. Алгоритм *K-Framed Quadtrees Decomposition* поєднує квадратно-рамкову декомпозицію, що дозволяє зменшити кількість вузлів та підвищити точність. Цей підхід забезпечує підвищену ефективність та точність у порівнянні з framed-quadtree, але також має високі вимоги до обчислювальних ресурсів і складний у реалізації [24, 25].

ЦФ для цього алгоритму є математична формула, яка мінімізує або максимізує певний параметр, залежно від конкретної задачі.

9.1. Мінімізація загальної довжини шляху при плануванні траєкторії:

$$\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}), \tag{9.1}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $n$ – загальна кількість клітин на шляху.

9.2. Мінімізація неоднорідності даних у клітинах:

$$\text{Minimize } \sum_{i=1}^{n} Var(C_i), \tag{9.2}$$

де $Var(C_i)$ – дисперсія (або інший показник неоднорідності) даних у $i$-ій клітині $C_i$, $n$ – загальна кількість клітин.

Таким чином, ЦФ для алгоритму K-Framed Quadtrees Decomposition може бути виражена як математична формула, яка мінімізує загальну довжину шляху (9.1), кількість клітин або неоднорідність даних у клітинах (9.2), залежно від конкретних вимог задачі.

10. Алгоритм *Adaptive Decomposition* використовує адаптивний підхід для декомпозиції простору і розбиття сигналів або простору на адаптивні компоненти, що дозволяє алгоритму адаптуватися до різних умов з метою аналізу, обробки або оптимізації. Основні переваги включають високу гнучкість та адаптивність, але алгоритм складний у реалізації та має високі вимоги до обчислювальних ресурсів [26, 27, 28].

ЦФ для цього алгоритму є математична формула, що мінімізує певний параметр, такий як, похибку реконструкції сигналу або неоднорідність даних у розбитих компонентах.

10.1. Мінімізація похибки реконструкції сигналу:

$$\text{Minimize } \sum_{i=1}^{n} \left| x(t) - \sum_{j=1}^{m} C_j(t) \right|^2, \tag{10.1}$$

де $x(t)$ – оригінальний сигнал, $C_j(t)$ – $j$-та адаптивна компонента сигналу, $n$ – кількість точок в сигналі, $m$ – кількість адаптивних компонент.

10.2. Мінімізація неоднорідності даних у розбитих компонентах:

$$\text{Minimize } \sum_{i=1}^{n} Var(C_i), \tag{10.2}$$

де $Var(C_i)$ – дисперсія даних у $i$-тій компоненті $C_i$, $n$ - загальна кількість компонентів.

Таким чином, ЦФ для алгоритму Adaptive Decomposition може бути виражена як математична формула, яка мінімізує похибку реконструкції сигналу (10.1), неоднорідність даних у компонентах (10.2) або кількість компонентів, залежно від конкретних вимог задачі.

11. Алгоритм *Probabilistic Cell Decomposition* (Ймовірнісне розбиття на клітини) використовує розбиття простору на комірки, які оцінюються на основі ймовірності проходження через них та використовується для планування шляху в середовищах з невизначеністю. Цей метод ефективний у високовимірних просторах і складних середовищах, але має високі вимоги до обчислювальних ресурсів і складний у реалізації [29, 30].

ЦФ для цього алгоритму є математична формула, яка мінімізує певний параметр, такий як ймовірність зіткнення або загальну довжину шляху з урахуванням ймовірностей.

11.1. Мінімізація ймовірності зіткнення:

$$\text{Minimize } \sum_{i=1}^{n} P(C_i), \tag{11.1}$$

де $P(C_i)$ – ймовірність того, що клітина $C_i$ містить перешкоди, $n$ – загальна кількість клітин на шляху.

11.2. Мінімізація загальної довжини шляху з урахуванням ймовірностей:

$$\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}) \cdot P(C_i), \tag{11.2}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $P(C_i)$ – ймовірність того, що клітина $C_i$ містить перешкоди, $n$ – загальна кількість клітин на шляху.

11.3. Мінімізація часу проходження з урахуванням ймовірностей:

$$\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i} \cdot P(C_i), \tag{11.3}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $v_i$ – швидкість руху в клітині $C_i$, $P(C_i)$ – ймовірність того, що клітина $C_i$ містить перешкоди, $n$ – загальна кількість клітин на шляху.

Таким чином, ЦФ для алгоритму Probabilistic Cell Decomposition може бути виражена як математична формула, яка мінімізує ймовірність зіткнення (11.1), загальну довжину шляху (11.2) або час проходження з урахуванням ймовірностей (11.3), залежно від конкретних вимог задачі.

12. Алгоритм *Probabilistic Cell Decomposition with Harmonic Functions,* поєднує ймовірнісне розбиття простору на ймовірнісні комірки з використанням гармонічних функцій для покращення точності планування шляху. Гармонічні функції мають властивість уникати локальних мінімумів, що робить їх корисними для планування шляху в складних середовищах. Цей метод має високі вимоги до обчислювальних ресурсів і складний у реалізації [30].

ЦФ для такого алгоритму є математична формула, яка мінімізує певний параметр, такий як, ймовірність зіткнення або загальна довжину шляху з урахуванням гармонічних потенціалів.

12.1. Мінімізація ймовірності зіткнення з урахуванням гармонічних функцій:

$$\text{Minimize } \sum_{i=1}^{n} P(C_i) \cdot H(C_i), \tag{12.1}$$

де $P(C_i)$ – ймовірність того, що клітина $C_i$ містить перешкоди, $H(C_i)$ – значення гармонічної функції в клітині $C_i$, $n$ – загальна кількість клітин на шляху.

12.2. Мінімізація загальної довжини шляху з урахуванням гармонічних функцій:

$$\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}) \cdot H(C_i), \tag{12.2}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $H(C_i)$ – значення гармонічної функції в клітині $C_i$, $n$ – загальна кількість клітин на шляху.

12.3. Мінімізація часу проходження з урахуванням гармонічних функцій:

$$\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i} \cdot H(C_i), \tag{12.3}$$

де $d(C_i, C_{i+1})$ – відстань між центрами клітин $C_i$ та $C_{i+1}$, $v_i$ – швидкість руху в клітині $C_i$, $H(C_i)$ – значення гармонічної функції в клітині $C_i$, $n$ – загальна кількість клітин на шляху.

Таким чином, ЦФ для алгоритму Probabilistic Cell Decomposition з використанням гармонічних функцій, може бути виражена як математична формула, яка мінімізує ймовірність зіткнення (12.1), загальну довжину шляху (12.2) або час проходження з урахуванням гармонічних потенціалів (12.3), залежно від конкретних вимог задачі.

13. Алгоритм *Ariadne's Clew* (нитка Аріадни) використовується для планування траєкторії в високовимірних безперервних просторах і складних середовищах та застосовується для АМР з багатьма ступенями свободи як у статичних, так і в динамічних середовищах. Цей алгоритм, будує дерево, що досліджує нові області простору в кожній ітерації. Має високі вимоги до обчислювальних ресурсів і складний у реалізації [31, 32]. Алгоритм складається з двох підалгоритмів, SEARCH (пошук) і EXPLORE (дослідження), що виконуються почергово.

ЦФ для алгоритму Ariadne's Clew визначається як оптимізаційна задача, яка мінімізує певний параметр, такий як, відстань до цілі або час проходження, з урахуванням інформації про доступний простір.

Загальна ЦФ для алгоритму Ariadne's Clew має вигляд:

$$\text{Minimize } f(q) = \sum_{i=1}^{n} \left( d(q_i, q_{goal}) + \lambda \cdot g(q_i) \right), \tag{13.1}$$

де $f(q)$ – загальна ЦФ, $q$ – поточне положення МР, $q_{goal}$ – положення цільової точки, $d(q_i, q_{goal})$ – відстань від поточного положення $q_i$ до цільової точки, $g(q_i)$ – функція, що враховує інформацію про доступний простір, зібрану підалгоритмом EXPLORE, $\lambda$ – ваговий коефіцієнт, що визначає вплив функції $g(q_i)$, $n$ – кількість кроків на шляху.

13.1. *Підалгоритм SEARCH* намагається знайти шлях до цілі, мінімізуючи відстань до цільової точки:

$$\text{Minimize } d(q_i, q_{goal}). \tag{13.2}$$

13.2. *Підалгоритм EXPLORE* збирає інформацію про доступний простір, що визначено через функцію $g(q_i)$, яка враховує параметри, такі як, щільність перешкод або доступність шляхів:

$$g(q_i) = \sum_{j=1}^{m} \left( \frac{1}{\| q - q_{obs_j} \|} \right), \tag{13.3}$$

де $q_{obs_j}$ – положення перешкоди j, а $m$ – кількість перешкод.

13.3. *Загальна ЦФ* для алгоритму Ariadne's Clew, що враховує обидва підалгоритми, має вигляд:

$$\text{Minimize } f(q) = \sum_{i=1}^{n} \left( d(q_i, q_{goal}) + \lambda \cdot \sum_{j=1}^{m} \left( \frac{1}{\| q - q_{obs_j} \|} \right) \right). \tag{13.4}$$

Таким чином, ЦФ алгоритму (13.4) Ariadne's Clew формується на основі двох підалгоритмів (13.2), (13.3), які використовують оптимізаційні методи для пошуку цілі та дослідження простору. Складність цільової функції визначається високовимірністю простору, динамічністю середовища та використанням оптимізаційних методів. Це робить алгоритм ефективним, але водночас і складним з точки зору обчислювальних ресурсів.

14. Алгоритм *Expansive Configuration Spaces* (ECS) використовує випадкове зразкування для дослідження конфігураційного простору. Алгоритм використовується для планування траєкторій у складних просторах конфігурацій, де важливо ефективно досліджувати простір і знаходити з'єднання між різними областями. Алгоритм ефективний у складних просторах й високовимірних середовищах, але має високі вимоги до обчислювальних ресурсів і складний у реалізації [33].

ЦФ для цього алгоритму зазвичай включає компоненти, які враховують як розширення простору, так і з'єднання між конфігураціями.

Загальна вигляд цільової функції для алгоритму Expansive Configuration Spaces:

$$\text{Minimize } U_{total}(q) = U_{exp}(q) + U_{conn}(q), \quad (14.1)$$

де $U_{total}(q)$ – загальний потенціал у точці $q$, $U_{exp}\ (q)$ – потенціал розширення, який стимулює дослідження нових областей простору конфігурацій, $U_{conn}(q)$ – потенціал з'єднання, який стимулює з'єднання між різними конфігураціями.

14.1. *Потенціал розширення* зазвичай визначається як функція, що стимулює вибір конфігурацій, що знаходяться на межі вже досліджених областей:

$$U_{exp}(q) = -\alpha \cdot boundary_{measure}(q), \quad (14.2)$$

де α – коефіцієнт, що визначає вагу розширення, а $boundary_{measure}(q)$ – міра, яка оцінює, наскільки конфігурація $q$ знаходиться на межі дослідженої області.

14.2. *Потенціал з'єднання* зазвичай визначається як функція, яка стимулює з'єднання між конфігураціями, що знаходяться на відстані одна від одної:

$$U_{conn}(q) = \beta \cdot \sum_{i=1}^{n} distance\ (q, q_i), \quad (14.3)$$

де $\beta$ – коефіцієнт, що визначає вагу з'єднання, $distance(q, q_i)$ – відстань між конфігурацією $q_i$ іншими конфігураціями $q_i$, $n$ – кількість конфігурацій, з якими потрібно з'єднатися.

14.3. *Загальна ЦФ* для алгоритму Expansive Configuration Spaces має вигляд:

$$\text{Minimize } U_{total}(q) = -\alpha \cdot boundary_{measure}(q) + \beta \cdot \sum_{i=1}^{n} distance\ (q, q_i). \quad (14.4)$$

Таким чином, ЦФ (14.4) алгоритму Expansive Configuration Spaces зазвичай спрямована на мінімізацію відстані або часу, необхідного для досягнення цільової конфігурації, і може включати різні компоненти вартості. Складність цільової функції залежить від розміру конфігураційного простору, кількості перешкод, вибору евристики та методів випадкового зразкування. Цей підхід дозволяє ефективно досліджувати релевантні ділянки простору, що робить його корисним для різних застосувань у робототехніці.

15. Алгоритм *Probabilistic Roadmap* (PRM) використовує випадкове зразкування для побудови графа, що представляє можливі шляхи в просторі і використовується для планування траєкторії МР у складних середовищах. Алгоритм ефективний у високовимірних просторах, але має високі вимоги до обчислювальних ресурсів і потребує налаштування параметрів [34, 35].

ЦФ для алгоритму Probabilistic Roadmap не є традиційною цільовою функцією, як у методах штучного потенційного поля. Натомість, PRM використовує графові алгоритми для пошуку найкоротшого шляху в побудованому графі. Основна мета алгоритму PRM знайти шлях, який мінімізує відстань або іншу метрику між початковою та цільовою конфігураціями.

Враховуючи, що *основні компоненти* алгоритму PRM складаються з фази *побудови* (Construction Phase) і фази *запиту* (Query Phase), ЦФ для алгоритму PRM може бути виражена як задача мінімізації відстані в графі:

$$\text{Minimize } d_{\text{total}} = \sum_{(q_i,q_j)\in P} d\left(q_i, q_j\right), \quad (15.1)$$

де $d_{total}$ – загальна відстань шляху $P$ від початкової конфігурації $q_{start}$ до цільової конфігурації $q_{goal}$, $\left(q_i, q_j\right)$ – ребро графу між конфігураціями $q_i$ та $q_j$, $d\left(q_i, q_j\right)$ – відстань між конфігураціями $q_i$ та $q_j$.

Така функція мінімізує загальну відстань шляху в графі, що дозволяє МР ефективно планувати траєкторію від початкової до цільової конфігурації, уникаючи перешкод

Таким чином, ЦФ (15.1) алгоритму Probabilistic Roadmap спрямована на мінімізацію довжини або часу шляху, а складність залежить від кількості зразків, перевірки на колізії та обраного алгоритму пошуку найкоротшого шляху. Завдяки своїй гнучкості та ефективності, PRM широко використовується для вирішення задач планування шляху в різних робототехнічних застосуваннях.

16. Алгоритм *Randomized Roadmap Method* (RRM) є одним із методів планування траєкторії для АМР у складних середовищах. Він використовує випадкові зразки для побудови

графа (roadmap), який представляє можливі шляхи від початкової точки до цілі. Ефективний у складних просторах і високовимірних середовищах, але має високі вимоги до обчислювальних ресурсів і потребує налаштування параметрів [36, 37, 38].

ЦФ для цього алгоритму включає мінімізацію відстані або часу проходження між початковою і кінцевою точками через побудований граф.

Загальна ЦФ для алгоритму Randomized Roadmap Method може бути виражена як:

$$\text{Minimize } C_{\text{total}}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c(q_i, q_j), \tag{16.1}$$

де $C_{total}(q_{start}, q_{goal})$ – загальна вартість шляху від початкової точки $q_{start}$ до цільової точки $q_{goal}$, $P$– послідовність з'єднань (шлях) від $q_{start}$ до $q_{goal}$, $c(q_i, q_j)$ – вартість переходу між двома конфігураціями $q_i$ і $q_j$.

*Вартість переходу* між двома конфігураціями зазвичай визначається як евклідова відстань між ними:

$$c(q_i, q_j) = \| q_i - q_j \|. \tag{16.2}$$

Алгоритм складається з двох основних фаз, фази побудови графа (Roadmap Construction) і фази запиту (Query Phase).

Загальна ЦФ для алгоритму Randomized Roadmap Method має вигляд:

$$\text{Minimize } C_{total}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c \| q_i - q_j \|. \tag{16.3}$$

Таким чином, ЦФ (16.3) алгоритму Randomized Roadmap Method зазвичай включає мінімізацію відстані або часу, а її складність залежить від розміру конфігураційного простору, щільності перешкод та кількості вибірок. Завдяки своїй гнучкості та ефективності, RRM широко використовується в робототехніці та інших областях, де необхідно вирішувати задачі планування руху.

17. Алгоритм *Lazy Probabilistic Roadmap* (Lazy PRM) є варіантом алгоритму Probabilistic Roadmap (PRM), який відкладає перевірку на зіткнення до моменту, коли це необхідно для відповіді на запит. Це зменшує час виконання, але можливі проблеми з масштабованістю та необхідність налаштування параметрів [39, 40].

ЦФ для алгоритму Lazy PRM може бути виражена як оптимізаційна задача, яка мінімізує загальну вартість шляху з урахуванням відстані між конфігураціями та перевірок на зіткнення.

Загальна ЦФ для алгоритму Lazy PRM має вираз:

$$\text{Minimize } C_{\text{total}}(q) = \sum_{(q_i,q_j)\in P} \left(d(q_i, q_j) + \lambda \cdot c(q_i, q_j)\right), \tag{17.1}$$

де $C_{total(q)}$ – загальна вартість шляху, $P$ – шлях, що складається з послідовності конфігурацій $q_i$, $d(q_i, q_j)$ – відстань між конфігураціями $q_i$ та $q_j$, $c(q_i, q_j)$ – вартість перевірки на зіткнення між конфігураціями $q_i$ та $q_j$, $\lambda$ – ваговий коефіцієнт, що визначає вплив вартості перевірки на зіткнення.

Враховуючи, що *відстань між конфігураціями,* $\mathrm{q_i}$ та $\mathrm{q_j}$ визначається як евклідова відстань,

$$d(q_i, q_j) = \| q_i - q_j \|, \tag{17.2}$$

а *вартість перевірки на зіткнення* $\mathrm{c(q_i, q_j)}$ може бути визначена як бінарна функція, яка приймає значення 1, якщо перевірка на зіткнення необхідна, і 0, якщо перевірка непотрібна:

$$c(q_i, q_j) = \begin{cases} 1, & \text{якщо перевірка на зіткнення необхідна} \\ 0, & \text{якщо перевірка на зіткнення не потрібна} \end{cases}. \tag{17.3}$$

Враховуючи (17.2) і (17.3), *загальна ЦФ* (20.1) для алгоритму Lazy PRM буде мати вираз:

$$\text{Minimize } C_{\text{total}}(q) = \sum_{(q_i,q_j)\in P} \left(\| q_i - q_j \| + \lambda \cdot c(q_i, q_j)\right). \tag{17.4}$$

Таким чином, ЦФ (17.4) в Lazy PRM спрямована на мінімізацію загальної вартості шляху, а складність алгоритму залежить від кількості вершин та ребер у графі. Основна

перевага алгоритму полягає в мінімізації кількості перевірок на зіткнення, що дозволяє зменшити час виконання алгоритму.

18. Алгоритм *Gaussian Sampling for PRM* є варіацією стандартного PRM. Модифікація пов'язана з можливістю краще покривати складні області простору, зменшуючи кількість необхідних зразків. Основні переваги включають ефективність у складних просторах, але алгоритм має високі вимоги до обчислювальних ресурсів і потребує налаштування параметрів [41]. Це дозволяє більш ефективно досліджувати складні області простору, особливо поблизу перешкод.

Враховуючи, що *основні компоненти* алгоритму *Gaussian Sampling for PRM* включають етапи, *вибір* і *перевірка зразків*, *побудову графу* й *пошук шляху*, ЦФ для алгоритму формалізовано може бути виражена як задача мінімізації відстані в графі, з урахуванням гауссового розподілення для вибору зразків:

$$\text{Minimize } d_{\text{total}} = \sum_{(q_i,q_j)\in P} d(q_i, q_j), \tag{18.1}$$

де $d_{total}$ – загальна відстань шляху $P$ від початкової конфігурації $q_{start}$ до цільової конфігурації $q_{goal}$, $(q_i, q_j)$ – ребро графу між конфігураціями $q_i$ та $q_j$, $d(q_i, q_j)$ – відстань між конфігураціями $q_i$ та $q_j$.

Для вибору зразків використовується гауссове розподілення. Спочатку виконується генерація випадкового зразка q1 у просторі конфігурацій й генерація другого зразка $\sigma^2$ з гауссового розподілення з центром у $q_1$:

$$q_2 \sim N(q_1, \sigma^2), \tag{18.2}$$

де $N(q_1, \sigma^2)$ – гауссове розподілення з середнім $q_1$ і дисперсією $\sigma^2$.

*Загальна ЦФ* для Gaussian Sampling for PRM буде мати вираз:

$$C_r = \{p \in C_{free} \mid \|p - q\| < d \wedge q \in \mathcal{IB} \cup \mathcal{OB}\}. \tag{18.3}$$

Таким чином, використання цільовою функцією (18.3) алгоритму Gaussian Sampling for PRM гауссового вибіркового розподілення, дозволяє краще охоплювати складні області простору, що підвищує ймовірність знаходження оптимального шляху. Хоча обчислювальна та просторова складність має завищені показники, загальна ефективність алгоритму часто виправдовує ці витрати.

19. Алгоритм *Halton Sampling for PRM* є варіацією класичного PRM, де для генерації зразків у конфігураційному просторі використовує послідовність Халтона для зразкування простору конфігурацій. Послідовність Халтона є низькодисперсійною послідовністю, яка забезпечує більш рівномірне покриття простору порівняно з випадковим зразкуванням. Це забезпечує рівномірний розподіл зразків, що зменшує кількість необхідних зразків і покращує ефективність алгоритму. Основні переваги включають рівномірний розподіл зразків і зменшення кількості зразків, але алгоритм складний у реалізації та має високі вимоги до обчислювальних ресурсів [42, 43].

Враховуючи, що *основні компоненти* алгоритму *Halton Sampling for PRM* включають етапи *генерації зразків за допомогою послідовності Халтона, перевірку на прохідність, побудова графа і пошук шляху, ЦФ* для алгоритму може бути виражена як задача мінімізації загальної вартості шляху в графі:

$$\text{Minimize } C_{\text{total}}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c(q_i, q_j), \tag{19.1}$$

де $C_{total}(q_{start}, q_{goal})$ – загальна вартість шляху від початкової точки $q_{start}$ до цільової точки $q_{goal}$, $P$ – послідовність з'єднань (шлях) від $q_{start}$ до $q_{goal}$, $c(q_i, q_j)$ – вартість переходу між двома конфігураціями $q_i$ і $q_j$.

Враховуючи, що вартість переходу між двома конфігураціями зазвичай визначається як евклідова відстань між ними:

$$c(q_i, q_j) = \| q_i - q_j \|, \tag{19.2}$$

*а послідовність Халтона генерується* за допомогою радикальної оберненої функції з різними базами для кожного виміру:

$$H(i) = \left(\phi_{b_1}(i), \phi_{b_2}(i), \dots, \phi_{b_d}(i)\right), \tag{19.3}$$

де $H(i)$ – $i$-та точка послідовності Халтона, $\phi_{b_k}(i)$ – радикальна обернена функція з базою $b_k$ для $k$-го виміру, $d$ – кількість вимірів у конфігураційному просторі.

Загальний вигляд цільової функції для *Halton Sampling for PRM* отримає вигляд:

$$\text{Minimize } C_{total}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c \parallel q_i - q_j \parallel. \tag{19.4}$$

Таким чином, ЦФ (19.4) алгоритму *Halton Sampling for PRM* завдяки своїй здатності генерувати рівномірно розподілені точки з низькою дискретністю, забезпечує краще покриття конфігураційного простору та підвищує ефективність алгоритму планування шляху.

20. Алгоритм D* (Dynamic A*) це інкрементальний пошуковий алгоритм, який використовує евристики, є розширенням алгоритму A*, призначеним для планування траєкторій у динамічних середовищах, де можуть змінюватися умови, такі як поява нових перешкод. Він підходить для частково відомих та змінних середовищ, але має високі вимоги до обчислювальних ресурсів і складний у реалізації [44, 45, 46].

Основна мета алгоритму D*, знайти найкоротший шлях від початкової точки до цільової, враховуючи зміни в середовищі в реальному часі.

*ЦФ* для алгоритму D* може бути виражена як задача мінімізації загальної вартості шляху від початкової конфігурації $q_{start}$ до цільової конфігурації $q_{goal}$:

$$\text{Minimize } C_{\text{total}}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c(q_i, q_j), \tag{20.1}$$

де $C_{total}(q_{start}, q_{goal})$ – загальна вартість шляху від початкової точки $q_{start}$ до цільової точки $q_{goal}$, $P$ – послідовність з'єднань (шлях) від $q_{start}$ до $q_{goal}$, $c(q_i, q_j)$ – вартість переходу між двома конфігураціями $q_i$ і $q_j$.

*Вартість переходу* між двома конфігураціями зазвичай визначається як евклідова відстань між ними або інша метрика, що враховує особливості середовища:

$$c(q_i, q_j) = \parallel q_i - q_j \parallel. \tag{20.2}$$

З урахуванням виразу (20.2) *ЦФ* для алгоритму D* отримаємо вираз:

$$\text{Minimize } C_{total}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} \left(\parallel q_i - q_j \parallel + h(q_i, q_{goal})\right), \tag{20.3}$$

де $h(q_i, q_{goal})$ – евристична функція, що оцінює вартість найкоротшого шляху від конфігурації $q_j$ до цільової конфігурації $q_{goal}$ .

А з урахуванням, що *евристична функція* $h(q_j, q_{goal})$ зазвичай визначається як евклідова відстань або *мангеттенська відстань* між конфігураціями:

$$h(q_j, q_{goal}) = \parallel q_j - q_{goal} \parallel. \tag{20.4}$$

Загальна ЦФ для алгоритму D* набуває вигляду:

$$\text{Minimize } C_{total}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P}(\parallel q_i - q_j \parallel + \parallel q_j - q_{goal} \parallel). \tag{20.5}$$

Таким чином, алгоритм Dynamic A* використовує адаптивну цільову функцію (20.5) для ефективного знаходження оптимальних шляхів у змінних умовах. Його складність може варіюватися залежно від конкретного застосування та умов середовища, але в цілому він пропонує значні переваги в адаптивності та ефективності порівняно з класичними методами.

21. Алгоритми цілеспрямованого та випадкового пошуку (*Goal-directed and Randomized Search*) поєднує цілеспрямований пошук, який спрямовує пошук у бік цільової точки, та випадковий пошук, який забезпечує дослідження простору конфігурацій. Використовується для вирішення задач оптимізації, де необхідно знайти оптимальне рішення шляхом мінімізації або максимізації цільової функції. Це дозволяє ефективно працювати у великих просторах, поєднуючи переваги обох підходів. Основні переваги включають ефективність у великих

просторах, але алгоритм складний у реалізації та потребує налаштування параметрів [47, 48, 49].

ЦФ виражена як математична формула, яка визначає мету оптимізації.

21.1. *Цілеспрямований пошук (Goal-directed Search),* спрямований на досягнення конкретної мети, і ЦФ для такого алгоритму може бути виражена як:

$$\text{Strong (Minimize} \vee \text{Maximize } f(x)), \tag{21.1}$$

де $f(x)$ – ЦФ, яку необхідно мінімізувати або максимізувати, а $x$ – набір змінних, що визначають стан системи або рішення задачі

21.2. *Випадковий пошук (Randomized Search),* використовує випадкові комбінації параметрів для знаходження оптимального рішення. ЦФ для випадкового пошуку може бути виражена аналогічно:

$$\text{Rand (Minimize} \vee \text{Maximize } f(x)), \tag{21.2}$$

де $f(x)$ – ЦФ, яку необхідно мінімізувати або максимізувати, а $x$ – набір випадково обраних змінних.

*Таким чином*, ЦФ для алгоритмів цілеспрямованого та випадкового пошуку може бути виражена як математична формула (21.1), (21.2), що мінімізує або максимізує певну функцію $f(x)$, залежно від конкретної задачі оптимізації.

22. Алгоритм *SANDROS* (Search And Nonuniform Dynamic Resolution Optimization Strategy) використовує комбінацію ієрархічного, нерівномірного багаторівневого та найкращого пошуку для знаходження майже оптимального рішення. Основні переваги включають високу ефективність у складних середовищах та високу точність, але алгоритм складний у реалізації та має високі вимоги до обчислювальних ресурсів [50].

ЦФ для цього алгоритму мінімізує або максимізує певний параметр, пов'язаний з оптимізацією. Нажаль автори дослідження, для алгоритму SANDROS не була явно вказана конкретна ЦФ. Тому враховуючи, що загальна форма цільової функції для оптимізаційних алгоритмів може бути виражена як:

$$\text{Minimize } f(x), \tag{22.1}$$

де $f(x)$ – це ЦФ, яку потрібно мінімізувати, а $x$ – вектор змінних, що оптимізуються.

Автор цього дослідження пропонує вважати, що ЦФ для алгоритму SANDROS має більш специфічні властивості залежно від конкретної задачі. Наприклад, якщо задача полягає в мінімізації відхилення потужності антен, ЦФ може бути виражена як:

$$\text{Minimize } \max_i \left|P_i - P_{target}\right|, \tag{22.2}$$

де $P_i$ – потужність і-тої антени, а $P_{target}$ – цільова потужність.

Враховуючи (22.3) загальна ЦФ мінімізації загальної потужності отримає вигляд:

$$\text{Minimize } \sum_{i=1}^{n} \left|P_i - P_{target}\right|, \tag{22.3}$$

де $n$ – загальна кількість антен.

Таким чином, ЦФ (22.3) алгоритму SANDROS може бути складною і включати кілька компонентів, що враховують різні аспекти руху МР. Складність цільової функції залежить від розмірності простору, кількості перешкод і динамічності середовища. Незважаючи на це, SANDROS демонструє високу ефективність у вирішенні задач планування шляхів завдяки своєму динамічному графовому пошуку.

23. Алгоритм *Artificial Potential Field* (APF) використовується для планування траєкторії МР, де робот використовує потенційні поля для планування шляху, де цільова точка притягує, а перешкоди відштовхують робота. Основні переваги включають простоту реалізації та швидкість обчислень, але алгоритм може застрягати в локальних мінімумах і не підходить для складних середовищ [29, 51].

ЦФ для цього алгоритму включає компоненти, які враховують як притягувальні, так і відштовхувальні потенціали.

Загальний формалізований вид цільової функції для алгоритму APF має вигляд:

$$\text{Minimize } U_{total}(q) = U_{att}(q) + U_{rep}(q), \tag{23.1}$$

де $U_{total}(q)$ – загальний потенціал у точці $q$, $U_{att}(q)$ – притягувальний потенціал, який притягує робота до цільової точки, $U_{rep}(q)$ – відштовхувальний потенціал, який відштовхує МР від перешкод.

23.1. *Притягуючий потенціал* зазвичай визначається як квадратична функція відстані до цілі:

$$U_{att}(q) = \frac{1}{2} k_{att} \left\| q - q_{goal} \right\|^2, \tag{23.2}$$

де $k_{att}$ – коефіцієнт привабливості, $q$ – поточне положення МР, $q_{goal}$ – положення цільової точки.

23.2. *Відштовхувальний потенціал* зазвичай визначається як функція, яка швидко зростає при наближенні до перешкоди:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\|q - q_{obs}\|} - \frac{1}{d_0} \right)^2 & if \ \|q - q_{obs}\| \le d_0, \\ 0 & if \ \|q - q_{obs}\| > d_0. \end{cases}, \tag{23.3}$$

де $k_{rep}$ – коефіцієнт відштовхування, $q_{obs}$ – положення перешкоди, $d_0$ – порогова відстань, за якою відштовхувальний потенціал дорівнює нулю.

23.3. *Загальна ЦФ* для алгоритму Artificial Potential Field має вигляд як:

$$\text{Minimize } U_{total}(q) = \frac{1}{2} k_{att} \left\| q - q_{goal} \right\|^2 + \sum_{i=1}^{m} \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\left\| q - q_{obs_i} \right\|} - \frac{1}{d_0} \right)^2 & if \ \left\| q - q_{obs_i} \right\| \le d_0, \\ 0 & if \ \left\| q - q_{obs_i} \right\| > d_0. \end{cases}, \tag{23.4}$$

де $m$ – кількість перешкод.

Складність алгоритму APF можна розглядати з кількох точок зору:

– Обчислювальна складність. APF має низьку обчислювальну складність, оскільки обчислення потенціалів і градієнтів є досить простими і можуть бути виконані в реальному часі. Це робить алгоритм привабливим для застосувань, де потрібна швидка реакція, наприклад, у МР.

– Проблеми локальних мінімумів: Однією з основних проблем APF є можливість потрапляння АМР в локальні мінімуми потенційного поля, де він може застрягти і не досягти цілі. Це обмежує застосування алгоритму в складних середовищах з багатьма перешкодами.

Таким чином, ЦФ (23.4) алгоритму Artificial Potential Field складається з притягувальних (23.2) і відштовхувальних (23.3) потенціалів, що дозволяє МР рухатися до цілі, уникаючи перешкод. Однак, основною проблемою є можливість потрапляння в локальні мінімуми, що обмежує його застосування в складних середовищах. Модифікації алгоритму можуть допомогти вирішити ці проблеми і покращити його ефективність.

24. Алгоритм *Dynamic Artificial Potential Field* (DAPF) є модіфікованою версією алгоритму Artificial Potential Field, що використовує динамічні потенційні поля для планування шляху, де цільова точка притягує, а перешкоди відштовхують АМР і використовується для планування траєкторії МР у динамічних середовищах, де перешкоди можуть змінювати своє положення з часом. Основні переваги включають простоту реалізації та швидкість обчислень, але алгоритм може застрягати в локальних мінімумах і не підходить для складних середовищ [51].

ЦФ для цього алгоритму включає компоненти, які враховують як притягувальні, так і відштовхувальні потенціали, щоб забезпечити безпечний і ефективний рух до цілі.

*Загальний* вигляд *цільової функції* алгоритму Dynamic Artificial Potential Field:

$$\text{Minimize } U_{total}(q) = U_{att}(q) + U_{rep}(q), \tag{24.1}$$

де $U_{total}(q)$ – загальний потенціал у точці $q$, $U_{att}(q)$ – притягуючий потенціал, який притягує робота до цільової точки, $U_{rep}(q)$ – відштовхувальний потенціал, який відштовхує робота від перешкод.

24.1. *Притягувальний потенціал* зазвичай визначається як квадратична функція відстані до цілі:

$$U_{att}(q) = \frac{1}{2} k_{att} \left\| q - q_{goal} \right\|^2, \tag{24.2}$$

де $k_{att}$ – коефіцієнт привабливості, $q$ – поточне положення МР, $q_{goal}$ – положення цільової точки.

24.2. *Відштовхувальний потенціал* зазвичай визначається як функція, яка швидко зростає при наближенні до перешкоди:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\|q - q_{obs}\|} - \frac{1}{d_0} \right)^2 & if \ \|q - q_{obs}\| \le d_0, \\ 0 & if \ \|q - q_{obs}\| > d_0. \end{cases}, \tag{24.3}$$

де $k_{rep}$ – коефіцієнт відштовхування, $q_{obs}$ – положення перешкоди, $d_0$ – порогова відстань, за якою відштовхувальний потенціал дорівнює нулю.

24.3. *Динамічний компонент.* Для врахування динамічних змін у середовищі, ЦФ може включати додатковий елемент, що враховує швидкість зміни положення перешкоди:

$$U_{dyn}(q, t) = \alpha \left\| \dot{q}_{obs}(t) \right\|, \tag{24.4}$$

де $\alpha$ – коефіцієнт, що визначає вплив динамічних змін, $\dot{q}_{obs}(t)$ – швидкість зміни положення перешкоди в момент часу $t$.

24.4. Остаточна ЦФ з урахуванням динамічного компонента, отримує наступний вираз:

$$\text{Minimize } U_{total}(q, t) = \frac{1}{2} k_{att} \left\| q - q_{goal} \right\|^2 + \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\|q - q_{obs}\|} - \frac{1}{d_0} \right)^2 & if \ \|q - q_{obs}\| \le d_0, \\ 0 & if \ \|q - q_{obs}\| > d_0. \end{cases} + \alpha \left\| \dot{q}_{obs}(t) \right\| . \tag{24.5}$$

Таким чином, ЦФ (24.5) алгоритму Dynamic Artificial Potential Field складається з атрактивних (24.2), (24.3) та репульсивних (24.4) компонентів, що дозволяє АМР ефективно досягати цілі, уникаючи перешкод. Основні виклики включають вирішення проблеми локальних мінімумів та високу обчислювальну складність, що компенсується адаптивністю та здатністю до роботи в реальному часі.

25. Алгоритм *Improved Artificial Potential Field* (IAPF) є вдосконаленою версією алгоритму штучного потенційного поля, який вирішує деякі його недоліки. Покращення включають використання додаткових методів для уникнення проблем з локальними мінімумами та підвищення ефективності уникнення перешкод [52, 53, 54].

ЦФ для цього алгоритму включає як привабливі, так і відштовхувальні потенціали, а також додаткові компоненти для уникнення локальних мінімумів.

Загальна ЦФ для алгоритму Improved Artificial Potential Field має вираз:

$$\text{Minimize } U_{total}(q) = U_{att}(q) + U_{rep}(q) + U_{improved}(q), \tag{25.1}$$

де $U_{total}(q)$ – загальний потенціал у точці $q$, $U_{att}(q)$– притягувальний потенціал, який притягує робота до цільової точки, $U_{rep}(q)$ – відштовхувальний потенціал, що відштовхує робота від перешкод, $U_{improved}(q)$ – додатковий компонент для покращення алгоритму.

25.1. *Притягувальний потенціал* зазвичай визначається як квадратична функція відстані до цілі:

$$U_{att}(q) = \frac{1}{2} k_{att} \left\| q - q_{goal} \right\|^2, \tag{25.2}$$

де $k_{att}$ – коефіцієнт притягування, $q$ – поточне положення МР, $q_{goal}$ – положення цільової точки.

25.2. *Відштовхувальний потенціал* зазвичай визначається як функція, яка швидко зростає при наближенні до перешкоди:

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\|q - q_{obs}\|} - \frac{1}{d_0} \right)^2 & if \ \|q - q_{obs}\| \le d_0, \\ 0 & if \ \|q - q_{obs}\| > d_0. \end{cases} \quad (25.3)$$

де $k_{rep}$ – коефіцієнт відштовхування, $q_{obs}$ – положення перешкоди, $d_0$ – порогова відстань, за якою відштовхувальний потенціал дорівнює нулю.

25.3. Для уникнення локальних мінімумів і покращення досяжності цілі, може бути доданий *додатковий компонент для покращення алгоритму*, такі як, використання віртуальних цілей або модифікація відштовхувального потенціалу:

$$U_{improved}(q) = \sum_{i=1}^{m} \frac{1}{2} k_{virt} \left\| q - q_{virt,i} \right\|^2, \quad (25.4)$$

де $k_{virt}$ – коефіцієнт притягування до віртуальних цілей, $q_{virt,i}$ – положення віртуальної цілі і, $m$ – кількість віртуальних цілей.

25.4. *Загальна ЦФ з урахуванням покращень* може бути виражена як:

$$\text{Minimize } U_{total}(q) = \frac{1}{2} k_{att} \left\| q - q_{goal} \right\|^2 + \sum_{j=1}^{m} \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\left\| q - q_{obs_j} \right\|} - \frac{1}{d_0} \right)^2 & if \ \|q - q_{obs_i}\| \le d_0, \\ 0 & if \ \|q - q_{obs_i}\| > d_0. \end{cases} + \sum_{i=1}^{m} \frac{1}{2} k_{virt} \left\| q - q_{virt,i} \right\|^2. \quad (25.5)$$

Таким чином, алгоритм Improved Artificial Potential Field пропонує значні покращення у порівнянні з класичним APF (24.5) завдяки модифікаціям цільової функції (25.5) та додатковим евристикам (25.4). Ці вдосконалення приводять до збільшення обчислювальної складності, вони забезпечують більш ефективне планування траєкторій та уникнення перешкод, що робить IAPF більш придатним для складних і динамічних середовищ.

26. Алгоритм *на основі штучного потенційного поля з використанням мережі підцілей* (Artificial Potential Field Based Subgoal Network) використовує потенційні поля для створення мережі підцілей, що допомагає МР уникати перешкод, є вдосконаленням традиційного методу штучного потенційного поля (APF), який використовується для планування траєкторії МР. Цей метод допомагає уникнути проблеми локальних мінімумів та забезпечує більш гладкі та ефективні траєкторії. Основні переваги включають простоту реалізації та швидкість обчислень, але алгоритм може застрягати в локальних мінімумах і не підходить для складних середовищ [55, 56, 57].

*ЦФ* для алгоритму Artificial Potential Field Based Subgoal Network може бути виражена як сума потенційних функцій, що включають притягувальні та відштовхувальні компоненти, а також додаткові компоненти для підцілей. Формула може бути записана наступним чином:

$$U(x) = U_{att}(x) + U_{rep}(x) + U_{subgoal}(x), \quad (26.1)$$

де $U_x$ – загальний потенціал у точці $x$, $U_{att}(x)$ – привабливий потенціал, що притягує робота до цільової точки, $U_{rep}(x)$ – відштовхувальний потенціал, що відштовхує робота від перешкод, $U_{subgoal}(x)$ – додатковий потенціал, що враховує підцілі для уникнення локальних мінімумів та оптимізації траєкторії.

З урахуванням, що притягувальний потенціал $U_{att}(x)$ може бути визначений як:

$$U_{att}(x) = \frac{1}{2} k_{att} \left\| x - x_{goal} \right\|^2, \quad (26.2)$$

де $k_{att}$ – коефіцієнт привабливості, і $x_{goal}$ – координати цільової точки.

*Відштовхувальний потенціал* $U_{rep}(x)$ може бути визначений як:

$$U_{rep}(x) = \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\|x - x_{obs}\|} - \frac{1}{d_0} \right)^2 & if \ \|x - x_{obs}\| \le d_0, \\ 0 & if \ \|x - x_{obs}\| > d_0. \end{cases} \quad (26.3)$$

де $k_{rep}$ – коефіцієнт відштовхування, $x_{obs}$ – координати перешкоди, $d_0$ – порогова відстань, на якій відштовхувальний потенціал починає діяти.

Додатковий потенціал підцілей $U_{subgoal}(x)$ може бути визначений як:

$$U_{subgoal}(x) = \sum_{i=1}^{n} \frac{1}{2} k_{subgoal} \left\| x - x_{subgoal_i} \right\|^2, \quad (26.4)$$

де $k_{subgoal}$ – коефіцієнт привабливості підцілей, $x_{subgoal_i}$ – координати $i$-ї підцілі, $n$ – кількість підцілей.

Враховуючи всі необхідні компоненти для ефективного планування траєкторії АМР з використанням мережі підцілей, отримаємо що, *загальна ЦФ* для алгоритму Artificial Potential Field Based Subgoal Network отримає наступний вигляд:

$$U(x) = \frac{1}{2} k_{att} \left\| x - x_{goal} \right\|^2 + \sum_{i=1}^{n} \frac{1}{2} k_{subgoal} \left\| x - x_{subgoal_i} \right\|^2 + \\ + \sum_{j=1}^{m} \begin{cases} \frac{1}{2} k_{rep} \left( \frac{1}{\left\| x - x_{obs_j} \right\|} - \frac{1}{d_0} \right)^2 & if \ \left\| x - x_{obs_j} \right\| \le d_0, \\ 0 & if \ \left\| x - x_{obs_j} \right\| > d_0. \end{cases}, \quad (26.5)$$

де $m$ – кількість перешкод.

Таким чином, ЦФ (26.5) алгоритму Artificial Potential Field Based Subgoal Network складається з атракційних та репульсивних компонент, а її складність визначається обчислювальними витратами, проблемами з локальними мінімумами та здатністю адаптуватися до змін у середовищі. Алгоритм поєднує переваги потенційних полів та субцілей, що дозволяє ефективно уникати перешкод і знаходити оптимальний шлях до цілі.

27. Алгоритм *динамічного планування шляху з підцілями* (Dynamic Subgoal Path Planner) генерує підцілі для обходу перешкод у динамічних середовищах з використанням підцілей для побудови оптимального шляху в динамічних середовищах. Алгоритм дозволяє швидко ухилятися від локальних мінімумів і ефективний у динамічних середовищах, але має високі вимоги до обчислювальних ресурсів і складний у реалізації [58].

ЦФ для такого алгоритму зазвичай включає кілька компонентів, таких як мінімізація відстані, уникнення перешкод та забезпечення плавності шляху.

*ЦФ* для Dynamic Subgoal Path Planner може бути виражена як:

$$\text{Minimize } J(x) = \alpha \cdot d(x) + \beta \cdot c(x) + \gamma \cdot s(x), \quad (27.1)$$

де $J(x)$ – загальна ЦФ, $d(x)$ – функція відстані, яка вимірює загальну відстань шляху, $c(x)$ – функція вартості, яка враховує вартість проходження через певні області (наприклад, уникнення перешкод), $s(x)$ – функція плавності, яка забезпечує плавність шляху, $\alpha, \beta, \gamma$ – вагові коефіцієнти, що визначають важливість кожного компонента.

До компонентів цільової функції відносяться:

27.1. Функція відстані $d(x)$:

$$d(x) = \sum_{i=1}^{n-1} \| x_{i+1} - x_i \|, \quad (27.2)$$

де $x_i$ – координати точки на шляху, а $n$ – кількість точок на шляху.

27.2. Функція вартості $c(x)$:

$$c(x) = \sum_{i=1}^{n} cost\ (x_i), \quad (27.3)$$

де $cost(x_i)$ – вартість проходження через точку $x_i$, яка може включати уникнення перешкод.

27.3. Функція плавност $s(x)$:

$$s(x) = \sum_{i=2}^{n-1} \| x_{i-1} - 2x_i + x_{i+1} \|, \quad (27.4)$$

де $x_{i-1}$, $x_i$, $x_{i+1}$ – послідовні точки на шляху, що забезпечують плавність.

Алгоритм включає *динамічне генерування підцілей*, що допомагає уникати локальних мінімумів і забезпечує більш ефективний пошук шляху. Підцілі можуть бути визначені на основі поточного стану та цільового стану, а також враховувати динамічні зміни в середовищі.

З урахування (27.2, 27.3, 27,4) ЦФ (27.1) отримає вигляд:

$$\text{Minimize } J(x) = \alpha \cdot \sum_{i=1}^{n-1} \|x_{i+1} - x_i\| + \beta \cdot \sum_{i=1}^{n} cost\ (x_i) + \gamma \cdot \sum_{i=2}^{n-1} \|x_{i-1} - 2x_i + x_{i+1}\|. \quad (27.5)$$

Таким чином, ЦФ (27.5) для Dynamic Subgoal Path Planner включає мінімізацію відстані, вартості та забезпечення плавності шляху, що дозволяє ефективно планувати шлях у динамічних середовищах.

28. Алгоритм *ієрархічного планування руху* (Hierarchical Motion Planner) використовує багаторівневий підхід для вирішення задач планування траєкторій, де кожен рівень відповідає за різні аспекти задачі. Алгоритм підходить для роботи у високовимірних просторах, але має високі вимоги до обчислювальних ресурсів і складний у налаштуванні параметрів [59–61].

ЦФ для такого алгоритму може включати кілька компонентів, таких як мінімізація відстані, уникнення зіткнень, оптимізація енерговитрат тощо.

ЦФ для ієрархічного планування руху може бути виражена як:

$$\text{Minimize } J = \sum_{i=1}^{n} w_i \cdot f_i(x), \quad (28.1)$$

де $J$ – загальна ЦФ, $f_i(x)$ – окремі компоненти цільової функції, що відповідають за різні аспекти задачі (наприклад, відстань, енерговитрати, уникнення зіткнень), $w_i$ – вагові коефіцієнти, що визначають важливість кожного компонента, $x$ – набір змінних, що визначають стан системи або траєкторію.

До компонентів цільової функції відносяться:

28.1. Мінімізація відстані:

$$f_1(x) = \int_0^T \|\dot{x}(t)\| dt. \quad (28.2)$$

28.2. Уникнення зіткнень:

$$f_2(x) = \sum_{j=1}^{m} \phi\left(d_j(x)\right). \quad (28.3)$$

де $\phi\left(d_j(x)\right)$ – функція штрафу за наближення до перешкод, $d_j(x)$ – відстань до $j$-ї перешкоди.

28.3. Оптимізація енерговитрат:

$$f_3(x) = \int_0^T P\big(x(t), \dot{x}(t)\big) dt, \quad (28.4)$$

де $P\big(x(t), \dot{x}(t)\big)$ – потужність, що витрачається на рух у момент часу $t$.

Об'єднавши всі компоненти, отримаємо повну цільову функцію:

$$\text{Minimize } J = w_1 \int_0^T \|\dot{x}(t)\| dt + w_2 \sum_{j=1}^{m} \phi\left(d_j(x)\right) + \text{w}_3 \int_0^T P\big(x(t), \dot{x}(t)\big) dt, \quad (28.5)$$

де $w_1$, $w_2$, $w_3$ – вагові коефіцієнти, що визначають важливість кожного компонента.

Таким чином**,** ЦФ (28.5) для алгоритму ієрархічного планування руху може бути виражена як лінійна комбінація окремих цільових функцій, що дозволяє враховувати різні аспекти задачі та забезпечує гнучкість і адаптивність алгоритму. При цьому функція є багатокомпонентною і враховує різні аспекти задачі, такі як мінімізація відстані, уникнення зіткнень та оптимізація енерговитрат.

29. *Двошаровий підцільовий алгоритм* (Two-Layered Subgoal) використовує два рівні для планування шляху, для розв'язання складних задач шляхом розбиття їх на підзадачі або підцілі. Основні переваги включають зменшення обчислювальної складності, але алгоритм складний у реалізації та має високі вимоги до обчислювальних ресурсів [62, 63].

ЦФ для такого алгоритму може бути виражена як математична формула, яка визначає мету оптимізації на кожному рівні ієрархії.

ЦФ для двошарового підцільового алгоритму може бути виражена як сукупність цільових функцій для кожного шару. Основна ЦФ може бути представлена як:

$$\text{Minimize } J = \sum_{i=1}^{n} w_i \cdot f_i(x), \quad (29.1)$$

де $J$ – загальна ЦФ, n – кількість підцілей або підзадач, $f_i(x)$ – ЦФ для кожної підцілі або підзадачі, $w_i$ – ваговий коефіцієнт для кожної підцілі або підзадачі, $x$ – набір змінних, що визначають стан системи або рішення задачі.

У двошаровому підході ЦФ може бути розділена на два основні рівні:

29.1. *Верхній рівень* (High-Level Goals):

ЦФ для верхнього рівня може включати глобальні цілі, такі як мінімізація загального часу виконання завдання або мінімізація витрат ресурсів. Має вираз:

$$J_{high} = \sum_{j=1}^{m} w_j \cdot g_j(y), \tag{29.2}$$

де $J_{high}$ – ЦФ верхнього рівня, $m$ – кількість глобальних цілей, $w_j$ – ваговий коефіцієнт для кожної глобальної цілі, $g_j(y)$ – ЦФ для кожної глобальної цілі, $y$ – набір змінних для верхнього рівня.

29.2. *Нижній рівень* (Low-Level Subgoals):

ЦФ для нижнього рівня може включати локальні цілі, такі як уникнення перешкод або оптимізація траєкторії. Формула для нижнього рівня має вираз:

$$J_{low} = \sum_{k=1}^{p} v_k \cdot h_k(z), \tag{29.3}$$

де $J_{low}$ – ЦФ нижнього рівня, $p$ – кількість локальних цілей, $v_k$ – ваговий коефіцієнт для кожної локальної цілі, $h_k(z)$ – ЦФ для кожної локальної цілі, $z$ – набір змінних для нижнього рівня.

Загальна ЦФ для двошарового підцільового алгоритму може бути виражена формулою (29.4) як комбінація цільових функцій, з урахуванням обох рівнів

$$J = J_{high} + J_{low}. \tag{29.4}$$

Застосувавши (29.2) і (29.3) до (29.4) отримаємо розгорнуту цільову функцію:

$$J = \sum_{j=1}^{m} w_j \cdot g_j(y) + \sum_{k=1}^{p} v_k \cdot h_k(z). \tag{29.5}$$

Таким чином, ЦФ (29.5) для двошарового підцільового алгоритму включає цільові функції для кожного рівня, які можуть бути виражені як сума вагових компонентів, що мінімізують або максимізують певні параметри. Це забезпечує гнучкість і ефективність у вирішенні складних задач оптимізації.

30. Алгоритм *RRT with Local Trees* є варіацією класичного RRT, що використовує кілька локальних дерев для покращення ефективності пошуку в складних середовищах та зменшення обчислювальних витрат. Це зменшує кількість перевірок колізій, але алгоритм складний у реалізації та потребує налаштування параметрів [64, 65].

*ЦФ* для алгоритму RRT з локальними деревами визначає мету оптимізації руху робота, і має формалізований вигляд як:

$$\text{Minimize } J = \sum_{i=1}^{n} w_i \cdot f_i(x), \tag{30.1}$$

де $J$ – загальна ЦФ, n – кількість локальних дерев або підзадач у алгоритмі, $f_i(x)$ – ЦФ для кожного локального дерева або підзадачі, $w_i$ – ваговий коефіцієнт для кожного локального дерева або підзадачі, $x$ – набір змінних, що визначають стан системи або рішення задачі

ЦФ може включати такі компоненти, як мінімізація відстані до цілі, уникнення зіткнень, мінімізація часу виконання завдання тощо.

$$J = w_1 \cdot D(x) + w_2 \cdot C(x) + w_3 \cdot T(x), \tag{30.2}$$

де $D(x)$ – функція відстані до цілі, $C(x)$ – функція уникнення зіткнень, $T(x)$ – функція часу виконання завдання, $w1, w2, w3$ – вагові коефіцієнти для відповідних функцій.

Виконання алгоритму складається з компонентів виконання цільової функції:

1. Мінімізація відстані. Основна мета алгоритму RRT з локальними деревами полягає в мінімізації загальної відстані між початковою та кінцевою точками. Це досягається шляхом додавання відстаней між послідовними точками на шляху:

$$D(x) = \sum_{i=1}^{n} d(x_i, x_{i+1}). \tag{30.3}$$

2. Уникнення перешкод. Для уникнення перешкод вводиться штрафна функція $c(x_j)$, яка додає додатковий штраф до цільової функції, якщо точка $x_j$ знаходиться в зоні перешкод. Ваговий коефіцієнт $\lambda$ визначає важливість уникнення перешкод відносно мінімізації відстані:

$$C(x) = \lambda \cdot \sum_{j=1}^{m} c(x_j). \quad (30.4)$$

3. Баланс між відстанню та безпекою. Ваговий коефіцієнт $\lambda$ дозволяє налаштувати баланс між мінімізацією відстані та уникненням перешкод. Вищі значення $\lambda$ призводять до більшого акценту на безпеку, тоді як нижчі значення $\lambda$ зосереджуються на мінімізації відстані.

З урахуванням формул (30.2), (30.3) загальна ЦФ отримає вигляд (30.5).

$$\text{Minimize } J = \sum_{i=1}^{n}(w_1 \cdot \sum_{i=1}^{n} d(x_i, x_{i+1}) + w_2 \cdot \lambda \cdot \sum_{j=1}^{m} c(x_j) + w_3 \cdot T(x)) \ . \quad (30.5)$$

Таким чином, ЦФ (30.5) для алгоритму RRT з локальними деревами, включає мінімізацію загальної відстані між початковою та кінцевою точками, а також уникнення перешкод шляхом введення штрафної функції. Ваговий коефіцієнт $\lambda$ дозволяє налаштувати баланс між цими двома цілями, забезпечуючи ефективне та безпечне планування руху МР.

31. Алгоритм *Obstacle-based RRT* використовується для планування траєкторій у середовищах з перешкодами. Алгоритм використовує перешкоди для керування зразкуванням, що дозволяє краще обходити перешкоди. Це підвищує ефективність у складних середовищах, але алгоритм складний у реалізації та потребує налаштування параметрів [66–68].

*ЦФ* для алгоритму Obstacle-based RRT спрямована на мінімізацію відстані до цілі, одночасно уникаючи зіткнень з перешкодами , має формалізований вигляд:

$$\text{Minimize J} = \mathrm{d}(\mathrm{x_{start}}, \mathrm{x_{goal}}) + \ \lambda \cdot \sum_{i=1}^{n} CollisionCost\ (x_i) + \mathrm{C}(x_i), \quad (31.1)$$

де $J$ – загальна ЦФ, $d(x_{start}, x_{goal})$ – відстань між початковою точкою $x_{start}$ та цільовою точкою $x_{goal}$, $\lambda$ – ваговий коефіцієнт, що визначає важливість уникнення зіткнень, $CollisionCost(x_i)$ – функція вартості зіткнення для кожної точки хі на траєкторії, $n$ – кількість точок на траєкторії.

Компоненті цільової функції алгоритму:

1. Відстань до цілі $d(x_{start}, x_{goal})$ відповідає за мінімізацію відстані між початковою та цільовою точками.

2. Вартість зіткнення $\sum_{i=1}^{n} ColissionCost(x_i)$ враховує вартість зіткнень з перешкодами на траєкторії.

3. Функція вартості зіткнення $(x_i) = \begin{cases} \infty, \text{ якщо } x_i \text{ знаходиться в зоні зіткнення} \\ 0, \text{ інакше} \end{cases}$

або більш детально:

$$\mathrm{C}(x_i) = \begin{cases} \alpha \cdot d(x_i, O), \text{ якщо } x_i \text{ знаходиться поблизу перешкоди} \\ 0, \text{ інакше} \end{cases}, \quad (31.2)$$

де $\alpha$ – коефіцієнт, що визначає вплив відстані до перешкоди на вартість, а $d(x_i, O)$ – відстань від вузла $x_i$ до найближчої перешкоди О.

*Таким чином,* ЦФ (31.1) для алгоритму Obstacle-based RRT спрямована на мінімізацію відстані до цілі, одночасно уникаючи зіткнень з перешкодами. Вона включає дві основні компоненти: відстань між початковою та цільовою точками та вартість зіткнень на траєкторії. Ваговий коефіцієнт $\lambda$ дозволяє налаштувати важливість уникнення зіткнень відносно мінімізації відстані. Такий підхід забезпечує ефективне планування траєкторій у складних середовищах з перешкодами.

32. Алгоритм *RRT-Connect* (Rapidly-exploring Random Tree Connect) використовує два дерева, які ростуть від старту та цілі, намагаючись з'єднатися між собою. Це дозволяє швидко

знаходити шлях, але можливі проблеми з масштабованістю та необхідність налаштування параметрів [69, 70]. Використовується для планування траєкторій АМР у робототехніці.

*ЦФ* для алгоритму RRT-Connect спрямована на мінімізацію відстані між конфігураціями та уникнення зіткнень, може бути виражена як мінімізація довжини шляху між початковою та кінцевою точками. Формально це можна записати так:

$$\text{Minimize } L(q_{start}, q_{goal}), \tag{32.1}$$

де $L(q_{start}, q_{goal})$ – довжина шляху між початковою конфігурацією $q_{start}$ та кінцевою конфігурацією $q_{goal}$.

З урахуванням функції відстані між конфігураціями $x_i$ та $x_{i+1}$, ЦФ для алгоритму RRT-Connect отримає наступний вираз:

$$Minimize\ J = \sum_{i=1}^{n} d(x_i, x_{i+1}), \tag{32.2}$$

де $J$ – загальна ЦФ, $n$ – кількість кроків або сегментів траєкторії, $d(x_i, x_{i+1})$ – функція відстані між конфігураціями $x_i$ та $x_{i+1}$.

*Компоненти цільової функції*

1. Відстань між конфігураціями:

$$d(x_i, x_{i+1}) = \|x_{i+1} - x_i\|, \tag{32.3}$$

де $\|\ \ \|$ – евклідова відстань або інша метрика, що використовується для вимірювання відстані між конфігураціями.

2. Уникнення зіткнень:

Для забезпечення безпеки траєкторії, необхідно додати обмеження на уникнення зіткнень:

$$subject\ to\ \ x_i \in C_{free}, \tag{32.4}$$

де $C_{tree}$ – допустима область конфігурацій, вільна від перешкод.

Застосувавши (32.3, 32.4) до (32.2), ЦФ для алгоритму RRT-Connect отримає вигляд:

$$Minimize\ J = \sum_{i=1}^{n} \|x_{i+1} - x_i\|\ \ subject\ to\ \ x_i \in C_{free}. \tag{32.5}$$

*Таким чином,* ЦФ (32.5) алгоритму RRT-Connect включає суму відстаней між послідовними конфігураціями та обмеження на допустимі області конфігурацій. Це дозволяє алгоритму знаходити оптимальні та безпечні траєкторії для МР у складних середовищах.

33. Алгоритм *двонаправленого швидко-зростаючого випадкового дерева* (Bidirectional Rapidly-exploring Random Tree, Bi-RRT) є розширенням класичного алгоритму RRT, який використовує два дерева, що ростуть одночасно від початкової та кінцевої точок, поки вони не з'єднаються. Це підвищує ефективність у складних середовищах, але алгоритм складний у реалізації та потребує налаштування параметрів [71]. Алгоритм використовується для планування траєкторій у просторі станів.

ЦФ для алгоритму Bi-RRT може бути виражена як математична формула, що мінімізує відстань між початковою та кінцевою точками, а також враховує уникнення зіткнень з перешкодами. Формально, ЦФ може бути записана як:

$$\text{Minimize } J = \sum_{i=1}^{n} d(x_i, x_{i+1}), \tag{33.1}$$

де $J$ – загальна ЦФ, $n$ – кількість вузлів у траєкторії, $d(x_i, x_{i+1})$ – відстань між сусідніми вузлами $x_i$ та $x_{i+1}$.

Для забезпечення безпеки та ефективності траєкторії, ЦФ також може включати *додаткові умови*, такі як уникнення зіткнень:

$$subject\ to\ x_i \in X_{free}, \forall i, \tag{33.2}$$

де $X_{free}$ – допустима область без перешкод.

З урахуванням додаткових умов (33.2), ЦФ може бути записана як:

$$\text{Minimize } J = \sum_{i=1}^{n} d(x_i, X_{i+1}) \;\; subject\ to\ x_i \in X_{free}, \forall i. \tag{33.3}$$

Таким чином, ЦФ (33.3) алгоритму Bi-RRT зазвичай спрямована на мінімізацію відстані між початковою та кінцевою точками, враховуючи обмеження середовища забезпечуючи при цьому уникнення перешкод і дотримання кінематичних обмежень. Формально, ЦФ може бути виражена як сума відстаней між сусідніми вузлами траєкторії з додатковими умовами на допустимість станів.

34. Алгоритм *RRT** (Rapidly-exploring Random Tree Star) є вдосконаленою версією алгоритму RRT, яка забезпечує асимптотичну оптимальність. Алгоритм використовує випадкове зразкування для побудови дерева, яке поступово розширюється до цільової точки. Це означає, що з часом алгоритм знаходить шлях, який наближається до оптимального. Основні переваги включають простоту реалізації та гарантовану оптимальність шляху, але алгоритм має високі вимоги до обчислювальних ресурсів і можливі проблеми з масштабованістю [72, 73].

ЦФ для алгоритму RRT* зазвичай пов'язана з мінімізацією вартості шляху від початкової точки до цільової точки.

Розширене рішення цільової функції для алгоритму RRT* можна виразити як:

$$\text{Minimize } C\left(x_{start}, x_{goal}\right), \tag{34.1}$$

де $C\left(x_{start}, x_{goal}\right)$ – це загальна вартість шляху від початкової точки $x_{start}$ до цільової точки $x_{goal}$.

Ця вартість може бути визначена як сума вартостей окремих сегментів шляху:

$$C\left(x_{start}, x_{goal}\right) = \sum_{i=1}^{n-1} c(x_i, x_{i+1}), \tag{34.2}$$

де $x_i$ – це точка на шляху, а $c(x_i, x_{i+1})$ – це вартість переходу між точками $x_i$ й $x_{i+1}$.

*Компоненти цільової функції:*

1. Вартість переходу $c(x_i, x_{i+1})$ може включати різні метрики.

2. Довжина шляху: $c(x_i, x_{i+1}) = \|x_{i+1} - x_i\|$.

3. Енергоспоживання: вартість може враховувати енергетичні витрати на переміщення між вузлами.

4. Час виконання: вартість може враховувати час, необхідний для переміщення між вузлами.

З урахуванням компонентів, ЦФ для алгоритму RRT* отримає наступний вигляд:

$$\text{Minimize } C\left(x_{start}, x_{goal}\right) = \sum_{i=1}^{n-1} \|x_{i+1} - x_i\|, \tag{34.3}$$

де $\|x_{i+1} - x_i\|$ – евклідова відстань між вузлами $x_i$ й $x_{i+1}$.

Рівняння (34.3) мінімізує загальну довжину шляху від початкової до цільової точки, що є основною метою алгоритму RRT*.

Таким чином, ЦФ алгоритму RRT* може бути виражена як сума вартостей переходів між послідовними вузлами траєкторії, де вартість може враховувати різні метрики, такі як довжина шляху, енергоспоживання або час виконання , а також забезпечує асимптотично оптимальне рішення для задачі планування траєкторії шляхом мінімізації цільової функції,. Це дозволяє алгоритму знаходити оптимальні траєкторії в складних середовищах.

35. Алгоритм Informed-RRT* є вдосконаленням алгоритму RRT*, який спрямований на оптимізацію процесу планування шляхів шляхом фокусування вибірки на підмножині простору, яка може покращити поточне рішення. Алгоритм швидше знаходить оптимальний шлях, але складніший у реалізації та потребує налаштування параметрів [73, 74].

ЦФ для Informed-RRT* зазвичай спрямована на мінімізацію довжини шляху або іншого критерію вартості.

ЦФ для Informed-RRT* як мінімізація вартості шляху від початкової точки $x_{start}$ до кінцевої точки $x_{goal}$:

$$\text{Minimize } J(\tau) = \int_0^T c\big(\tau(t)\big)dt, \tag{35.1}$$

де $\tau$ – шлях від точки $x_{start}$ до точки $x_{goal}$, $T$ – час, необхідний для проходження шляху$\tau$, $c\big(\tau(t)\big)$ – функція вартості, яка може включати довжину шляху, енергію, час або інші критерії.

Алгоритм Informed-RRT* використовує інформоване вибіркове простір для підвищення ефективності пошуку. Це досягається шляхом обмеження вибіркового простору до еліпсоїда, який містить всі можливі покращення поточного рішення. Формально, це можна виразити як:

$$\mathcal{E} = \{x \in \mathbb{R}^n | \, \|x - x_{mid}\|_{A^{-1}} \leq 1\}, \tag{35.2}$$

де $\mathcal{E}$ – еліпсоїд вибіркового простору, $x_{mid}$ – середня точка між початковою та цільовою точками, $A$ – матриця, що визначає форму та розмір еліпсоїда.

Алгоритм Informed-RRT* використовує евристичну функцію для фокусування вибірки на підмножині простору, яка може покращити поточне рішення. Це досягається шляхом вибірки точок всередині пролатного гіперсфероїда, який визначається початковою та кінцевою точками, а також поточним найкращим шляхом.

*Пролатний гіперсфероїд* визначається як множина точок $x$, що задовольняють наступну умову:

$$(x - x_{center})^2 \cdot A(x - x_{center}) \leq 1, \tag{35.3}$$

де $x_{center}$ – центр гіперсфероїда, який знаходиться на середині відрізка між $x_{start}$ та $x_{goal}$, $A$ – матриця, яка визначає форму та розміри гіперсфероїда.

З урахуванням евристики, *ЦФ* може бути переписана як:

$$\text{Minimize } J(\tau) = \int_0^T c\big(\tau(t)\big)dt + h(x_{goal}, x), \tag{35.4}$$

де $h\big(x_{goal}, x\big)$ – евристична функція, яка оцінює вартість досягнення кінцевої точки $x_{goal}$ з поточної точки $x$.

Таким чином, ЦФ (35.4) для алгоритму Informed-RRT* спрямована на мінімізацію вартості шляху з урахуванням евристики, яка фокусує вибірку на підмножині простору, що може покращити поточне рішення. Використання еліпсоїда для обмеження вибіркового простору дозволяє значно зменшити кількість непотрібних вибірок і прискорити процес пошуку оптимального шляху. Це дозволяє алгоритму швидше знаходити оптимальні або близькі до оптимальних шляхи.

36. Алгоритм *MBD-RRT*FFT* (Multi-Bidirectional Rapidly-exploring Random Tree with Fast Fourier Transform) є складним алгоритмом для планування шляху, який використовує багатонаправлене розширення дерев, багатопотокові обчислення та алгоритм Fitch для оптимізації. Основні переваги включають високу ефективність та зменшення часу виконання завдяки паралельній обробці даних. Недоліками є складність реалізації та необхідність налаштування параметрів [75].

ЦФ цього алгоритму спрямована на мінімізацію вартості шляху з урахуванням різних критеріїв, таких як відстань, час, енергія та уникнення перешкод.

ЦФ для алгоритму MBD-RRT*FFT може бути виражена як:

$$\text{Minimize } J(x) = \sum_{i=1}^{n-1} c(x_i, x_{i+1}), \tag{36.1}$$

де $J(x)$ – загальна вартість шляху, $x_i$ – точки шляху, $c(x_i, x_{i+1})$ – функція вартості переходу між точками $x_i$ та $x_{i+1}$.

Для алгоритму MBD-RRT*FFT ЦФ може включати додаткові критерії, такі як уникнення перешкод, дотримання динамічних обмежень та оптимізація за допомогою швидкого перетворення Фур'є. Тому розширене рішення цільової функції може бути виражене як:

$$\text{Minimize } J(x) = \sum_{i=1}^{n-1} |c(x_i, x_{i+1}) + \lambda \cdot h(x_i, x_{i+1}) + \mu \cdot f(x_i, x_{i+1})|, \quad (36.2)$$

де $h(x_i, x_{i+1})$ – штрафна функція за порушення обмежень (наприклад, зіткнення з перешкодами), $f(x_i, x_{i+1})$ – функція оптимізації за допомогою швидкого перетворення Фур'є, $\lambda$ й $\mu$ – коефіцієнт штрафу, що визначає важливість уникнення перешкод відносно основної вартості шляху.

ЦФ включає компоненти:

1. Функція вартості переходу визначається як евклідова відстань між точками $x_i$ та $x_{i+1}$.:

$$c(x_i, x_{i+1}) = \| x_{i+1} - x_i \|. \quad (36.3)$$

2. Штрафна функція $h(x_i, x_{i+1})$ враховує різні обмеження, такі як зіткнення з перешкодами або порушення динамічних обмежень. Вона визначається як:

$$h(x_i, x_{i+1}) = \begin{cases} 0, & \text{якщо перехід не порушує обмежень} \\ f(x_i, x_{i+1}), & \text{якщо перехід порушує обмеження} \end{cases}, \quad (36.4)$$

де $f(x_i, x_{i+1})$ – функція, що визначає величину штрафу за порушення обмежень. Ця функція може бути різною залежно від обмежень [75].

3. Функція оптимізації за допомогою швидкого перетворення Фур'є $f(x_i, x_{i+1})$:

$$f(x_i, x_{i+1}) = FFT(x_i, x_{i+1}). \quad (36.5)$$

Застосовуючи формули (36.3),(36.4) і (36.5) до (36.2) отримаємо загальний вигляд цільової функції:

$$\text{Minimize } J(x) = \sum_{i=1}^{n-1} \left| \| x_{i+1} - x_i \| + \lambda \cdot h(x_i, x_{i+1}) = \begin{cases} 0, & f_{collision}(x_i, x_{i+1}) = \infty \\ f(x_i, x_{i+1}), & f_{dynamic}(x_i, x_{i+1}) = \alpha \cdot violation_{degree(x_i, x_{i+1})} \end{cases} + \mu \cdot FFT(x_i, x_{i+1}) \right|. \quad (36.6)$$

Багатопотоковість дозволяє паралельно обробляти різні частини дерева, що значно прискорює процес пошуку. Після завершення обчислень кожним потоком, результати порівнюються для отримання загального оптимального шляху.

Таким чином, ЦФ (36.2) алгоритму MBD-RRT*FFT спрямована на мінімізацію загальної вартості шляху, враховуючи відстань між точками, штрафи за порушення обмежень та оптимізацію за допомогою швидкого перетворення Фур'є з урахуванням багатопотокової обробки, що дозволяє ефективно планувати шляхи в динамічних середовищах. Це забезпечує уникнення перешкод та дотримання інших важливих критеріїв, таких як динамічні обмеження та швидкість обчислень А також дозволяє алгоритму ефективно планувати шляхи в складних середовищах, забезпечуючи уникнення перешкод та дотримання інших важливих критерії

37. *Динамічні ігри*, це теоретичний підхід для моделювання стратегій, що дозволяє моделювати складні взаємодії. Основні переваги включають можливість моделювання складних взаємодій та застосування в різних галузях, але цей підхід вимагає глибоких знань теорії ігор і складних математичних моделей [76, 77, 78].

Динамічні ігри часто моделюються за допомогою динамічного програмування, де використовується рівняння Беллмана для визначення оптимальної стратегії.

Одним з підходів до вирішення проблеми пошуку шляху АМР є використання динамічних ігор, де ЦФ визначає оптимальний шлях для МР [79, 80].

ЦФ в контексті динамічних ігор для пошуку шляху може бути визначена як функція, яка мінімізує або максимізує певний критерій, наприклад, відстань, час або витрати енергії. Формально, ЦФ може бути записана як:

$$J(u) = \int_{t_0}^{t_f} L(x(t), u)t), t)dt + \Phi(x(t_f)), \quad (37.1)$$

де $J(u)$ – ЦФ, яку потрібно мінімізувати або максимізувати, $L(x(t), u)t)$, $t$ – функція витрат, яка залежить від стану $x(t)$, керування $u(t)$ та часу $t$, $\Phi(x(t_f)$ – кінцеві витрати, які залежать від стану в кінцевий момент часу $t_f$, $t_0$ і $t_f$ – час початку і закінчення в динамічній гри.

Таким чином, ЦФ (37.1) для методу динамічних ігор визначається як максимізація очікуваної суми винагород, враховуючи ймовірності переходів між станами та коефіцієнт дисконтування.

38. *Метод розв'язуючих функцій Чикрія* – це теоретичний підхід для розв'язання задач керування в умовах конфлікту, який базується на використанні обернених функціоналів. Цей метод дозволяє знаходити теоретично обґрунтовані рішення для задач керування в умовах конфлікту, що робить його корисним у різних галузях, таких як робототехніка, економіка та військова справа. Основні переваги включають можливість застосування в різних галузях та теоретично обґрунтовані рішення, але метод має високі вимоги до знань теорії керування та складний у реалізації [81–84].

Метод розв'язуючих функцій А. О. Чикрія є одним із фундаментальних методів, що застосовуються для розв'язання диференціальних ігор з переслідуванням [85]. Цей метод може бути адаптований для задач пошуку шляху МР.

Розглянемо АМР, який має знайти оптимальний шлях з початкової точки $A$ до цільової точки $B$ в середовищі з перешкодами. Нехай $x(t)$ – це стан робота в момент часу $t$, а $u(t)$ – керування роботом.

ЦФ в цьому контексті визначає витрати на рух МР від початкової точки до цільової точки, враховуючи перешкоди та інші фактори. Тоді ЦФ $J$ може бути визначена як:

$$J = \int_0^T L(x(t), u)t), t)dt + \Phi(x(T)), \tag{38.1}$$

де $L(x(t), u)t)$, $t$ – функція витрат, яка залежить від стану $x(t)$, керування $u(t)$, $\Phi(x(T))$ – кінцева вартість, яка залежать від кінцевого стану $T$, $T$ – час досягнення точки цільовою функцією.

Метод розв'язуючих функцій передбачає побудову функції $V(x)$, яка задовольняє рівнянню Гамільтона-Якобі-Беллмана (ГЯБ):

$$\frac{\partial V}{\partial x} + \min_u \left[L(x, u) + \frac{\partial V}{\partial x} f(x, u)\right] - 0, \tag{38.2}$$

де $f(x, u)$ – динаміка системи.

Кроки виконання алгоритму включають:

1. Ініціалізація. Встановити початкові умови для функції $V(x)$ на кінцевому стані $x(T)$:

$$V\big(x(T)\big) = \Phi\big(x(T)\big). \tag{38.3}$$

2. Розв'язання рівняння ГЯБ. Використовуючи методи чисельного розв'язання, знайти функцію $V(x)$ для всіх станів $x$ та моментів часу $t$.

3. Визначення оптимального керування. Для кожного стану $x$ та моменту часу $t$, знайти оптимальне керування $u^*(t)$, яке мінімізує вираз:

$$u^*(t) = \arg\min_u \left[L(x, u) + \frac{\partial V}{\partial x} f(x, u)\right]. \tag{38.4}$$

Побудова траєкторії: Використовуючи знайдене оптимальне керування $u^*(t)$, будується траєкторія робота від початкової точки $A$ до цільової точки $B$.

Таким чином, метод розв'язуючих функцій А. О. Чикрія, дозволяє знайти оптимальний шлях АМР, мінімізуючи витрати на рух та враховуючи динаміку системи. ЦФ (38.1) в цьому контексті визначає критерій, за яким оцінюється якість знайденого шляху. Використання часткових похідних дозволяє ефективно досліджувати унімодальні функції та знаходити оптимальні рішення, що забезпечує ефективний та безпечний рух МР в складних середовищах.

**Висновки**

В роботі розглянуто і проаналізовано 38 сучасних алгоритмів і методів планування шляху для автономних мобільних роботів та їх цільові функції.

Узагальнені результати зведені до єдиної порівняльної таблиці 1, яка надає вид формалізованої основної функції алгоритмів, розкриває основну ідею, переваги/недоліки, сферу застосування і приклад використання. Порівняльна таблиця 1, представлена в формі узагальненого допоміжного довідкового елемента дослідження.

Дослідження підтверджує, що відокремлювати роль цільової функції при дослідженні алгоритмів планування шляху мобільним роботом неможливо, оскільки вона є фундаментальним елементом, який визначає мету оптимізації, оцінює ефективність рішень, забезпечує адаптивність до умов середовища, гарантує безпеку та підвищує ефективність обчислень. Без цільової функції алгоритми планування шляху втрачають свою цілеспрямованість та ефективність, що робить їх непридатними для практичного застосування.

Розуміння цільової функції є критично важливим для алгоритмів будови шляху автономних мобільних роботів з кількох причин, а саме:

1. *Оптимізація маршруту* – ЦФ визначає, що саме алгоритм намагається оптимізувати. Це може бути мінімізація відстані, часу, енергії або уникнення перешкод. Наприклад, для безпілотних наземних засобів (БпНЗ) ЦФ може враховувати динаміку апарата, обмеження рельєфу та ентропію простору переміщення.

2. *Безпека та уникнення перешкод* – ЦФ часто включає параметри, які допомагають уникати зіткнень з перешкодами. Це особливо важливо для автономних мобільних роботів, що працюють у динамічних середовищах з рухомими об’єктами.

3. *Ефективність обчислень* – добре визначена ЦФ дозволяє алгоритму працювати більш ефективно, зменшуючи кількість необхідних обчислень для досягнення оптимального рішення. Це важливо для реального часу, коли МР повинен швидко реагувати на зміни в середовищі.

4. *Адаптивність до змін* – ЦФ може бути налаштована таким чином, щоб алгоритм міг адаптуватися до змін у середовищі або умовах завдання. Наприклад, у випадку з БпЛА, ЦФ може враховувати зміну вітрових умов або обмеження по заряду батареї.

Дослідження показало, що розуміння та правильне визначення цільової функції є ключовим для успішної реалізації алгоритмів будови шляху автономних мобільних роботів.

Отримані висновки підтверджують, що кожен з розглянутих алгоритмів має свої переваги та недоліки, що робить їх придатними для різних застосувань у робототехніці та автономних транспортних засобах. А розуміння цільової функції надає правильного вектору застосування при виборі конкретного алгоритму залежить від специфічних вимог до планування шляху, таких як складність середовища в якому працює АМР, й вимог до ефективності, точності, необхідність оптимізації та обчислювальних ресурсів [1, 2, 4, 16, 64, 75, 82, 85].

**Напрями подальших досліджень**

Враховуючи отримані результати дослідження, в подальшому планується поглиблене дослідження методу розв’язуючих функцій А. О. Чикрія, яке на думку автора, має великий потенціал для покращення автономних систем у різних аспектах, включаючи оптимізацію, уникнення конфліктів, адаптивність та координацію в мультиагентних системах [84].

Таблиця 1. **Порівняння сучасних алгоритмів планування шляху з урахуванням їх цільових функцій**

| № з/п | Алгоритм | Тип алгоритму | Основна ідея | Цільова функція алгоритму | Переваги | Недоліки | Застосування | Приклади використання |
|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | **VCD** (Vertical Cellular Decomposition) | Точне розбиття | Вертикальне розбиття простору на клітини | **Мінімізація загальної довжини шляху** Minimize $\sum_{i=1}^{n} A_i$ | – Простота реалізації; – Висока точність | – Можливі проблеми з масштабованістю – Високі вимоги до обчислювальних ресурсів | – Робототехніка – Автономні транспортні засоби | – Планування шляху для мобільних роботів – Оптимізація маршрутів у логістиці |
| 2 | **BCD** (Boustrophedon Cellular Decomposition) | Точне розбиття | Розбиття простору на клітини для покриття | **Мінімізація загальної довжини шляху** Minimize $\sum_{i=1}^{n}(L_i + T_i)$ | – Простота реалізації – Ефективне покриття простору | – Можливі проблеми з масштабованістю – Високі вимоги до обчислювальних ресурсів | – Робототехніка – Автономні транспортні засоби | – Планування шляху для мобільних роботів – Оптимізація маршрутів у логістиці |
| 3 | **Morse Decomposition** | Точне розбиття | Використання теорії Морса для розбиття простору | **Мінімізація загальної довжини шляху** Minimize $\sum_{i=1}^{n} \int_{M_i} \| \dot{x} - f(x)^2 \| \, dx$ | – Глибокий математичний підхід – Можливість аналізу топології | – Складність реалізації – Високі вимоги до обчислювальних ресурсів | – Робототехніка – Автономні транспортні засоби | – Планування шляху для мобільних роботів – Оптимізація маршрутів у логістиці |
| 4 | **Exhaustive Path Planning with Exact Cell Decomposition** | Точне розбиття | Вичерпне планування шляху з точним розбиттям простору | **Мінімізація загальної довжини шляху:** Minimize $\sum_{i=1}^{n} d(C_i, C_{i+1})$ **Мінімізація часу проходження:** Minimize $\sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i}$ | – Висока точність – Гарантована точність планування | – Високі вимоги до обчислювальних ресурсів – Складність реалізації | – Робототехніка – Автономні транспортні засоби | – Планування шляху для мобільних роботів – Оптимізація маршрутів у логістиці |
| 5 | **Approximate Cell Decomposition** | Приблизне розбиття | Приблизне розбиття простору на регулярні клітини | **Мінімізація загальної довжини шляху:** $OCB_j[k^u] = \bigcup_{t=1}^{k_u} \{ CB_j[[x_k, x'_k] \cdot [y_k, y'_k] \cdot (\gamma_k + l \triangle \theta)]\}$ | – Простота реалізації – Швидкість обчислень | – Менша точність порівняно з точними методами – Можливі помилки через наближення | – Робототехніка – Автономні транспортні засоби | – Планування шляху для мобільних роботів – Оптимізація маршрутів у логістиці |
| 6 | **Sensor path planning with approximate cell decomposition** | Глобальне планування | Декомпозиція простору на клітини | **Мінімізація загальної довжину шляху:** Minimize $\sum_{i=1}^{n} d(C_i, C_{i+1})$ **Мінімізація часу проходження:** Minimize $\sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i}$ | – Ефективність у складних середовищах – Можливість роботи з різними типами перешкод | – Високі вимоги до обчислювальних ресурсів – Необхідність налаштування параметрів | – Робототехніка – Автономні транспортні засоби | – Планування шляху для мобільних роботів – Оптимізація маршрутів у логістиці |
| 7 | **Quadtree–based decomposition** | Глобальне планування | Декомпозиція простору на квадрати | **Мінімізація неоднорідності даних у квадрантах:** Minimize $\sum_{i=1}^{n} Var(Q_i)$ | – Простота реалізації – Ефективність у високовимірних просторах | – Можливі проблеми з масштабованістю – Необхідність налаштування параметрів | – Робототехніка – Автономні транспортні засоби | – Планування шляху для дронів – Оптимізація маршрутів у логістиці |
| 8 | **Framed–quadtree decomposition** | Глобальне планування | Декомпозиція простору на квадрати з рамками | **Мінімізація загальної довжини шляху:** Minimize $\sum_{i=1}^{n} d(C_i, C_{i+1})$ **Мінімізація часу проходження:** Minimize $\sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i}$ | – Зменшення кількості вузлів – Підвищена ефективність порівняно з класичним quadtree | – Складність реалізації – Високі вимоги до обчислювальних ресурсів | – Робототехніка – Автономні транспортні засоби | – Планування шляху для дронів – Оптимізація маршрутів у логістиці |

*Продовження таблиці 1*

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 9 | **K–Framed quadtrees decomposition** | Глобальне планування | Комбінація квадратно–рамкової декомпозиції | **Мінімізація загальної довжини шляху:** $\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1})$ | – Зменшення кількості вузлів та підвищення точності<br>– Підвищена ефективність та точність порівняно з framed-quadtree | – Складність реалізації<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 10 | **Adaptive decomposition** | Глобальне планування | Адаптивна декомпозиція | **Мінімізація похибки реконструкції сигналу:** $\text{Minimize } \sum_{i=1}^{n} \left\lvert x(t) - \sum_{j=1}^{m} C_j(t) \right\rvert^2$ | – Висока гнучкість та адаптивність<br>– Можливість адаптації до різних умов | – Складність реалізації<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 11 | **Probabilistic cell decomposition** | Глобальне планування | Розбиття простору на ймовірнісні комірки | **Мінімізація ймовірності зіткнення:** $\text{Minimize } \sum_{i=1}^{n} P(C_i)$<br>**Мінімізація загальної довжини шляху з урахуванням ймовірностей:** $\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}) \cdot P(C_i)$<br>**Мінімізація часу проходження з урахуванням ймовірностей:** $\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i} \cdot P(C_i)$ | – Ефективність у високовимірних просторах<br>– Можливість роботи у складних середовищах | – Високі вимоги до обчислювальних ресурсів<br>– Складність реалізації | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 12 | **Probabilistic cell decomposition with harmonic functions** | Глобальне планування | Розбиття простору на ймовірнісні комірки з гармонічними функціями | **Мінімізація ймовірності зіткнення з урахуванням гармонічних функцій:** $\text{Minimize } \sum_{i=1}^{n} P(C_i) \cdot H(C_i)$<br>**Мінімізація загальної довжини шляху з урахуванням гармонічних функцій:** $\text{Minimize } \sum_{i=1}^{n} d(C_i, C_{i+1}) \cdot H(C_i)$<br>**Мінімізація часу проходження з урахуванням гармонічних функцій:** $\text{Minimize } \sum_{i=1}^{n} \frac{d(C_i, C_{i+1})}{v_i} \cdot H(C_i)$ | – Покращена точність завдяки гармонічним функціям<br>– Ефективність у складних середовищах | – Високі вимоги до обчислювальних ресурсів<br>– Складність реалізації | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 13 | **Ariadne's clew** | Глобальне планування | Побудова дерева, що досліджує нові області простору | **Загальна цільова функція:** $\text{Minimize } f(q) = \sum_{i=1}^{n} \left( d(q_i, q_{goal}) + \lambda \cdot \sum_{j=1}^{m} \left( \frac{1}{\lVert q - q_{obs_j} \rVert} \right) \right)$ | – Ефективне дослідження нових областей<br>– Підходить для високовимірних просторів | – Високі вимоги до обчислювальних ресурсів<br>– Складність реалізації | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 14 | **ECS** (Expansive configuration spaces) | Глобальне планування | Використання випадкового зразкування для дослідження конфігураційн ого простору | **Загальна цільова функція:** $\text{Minimize } U_{total}(q) = -\alpha \cdot boundary_{measure}(q) + + \beta \cdot \sum_{i=1}^{n} distance\,(q, q_i)$ | – Ефективність у складних просторах<br>– Можливість роботи у високовимірних просторах | – Високі вимоги до обчислювальних ресурсів<br>– Складність реалізації | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 15 | **PRM** (Probabilistic Roadmap) | Глобальне планування | Використання випадкового зразкування для побудови графа | **Загальна цільова функція:** $\text{Minimize } d_{\text{total}} = \sum_{(q_i, q_j) \in P} d(q_i, q_j)$ | – Можливість роботи у високовимірних просторах<br>– Ефективність у складних просторах | – Високі вимоги до обчислювальних ресурсів<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |

*Продовження таблиці 1*

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 16 | **RRM** (Randomized roadmap method) | Глобальне планування | Використання випадкового зразкування для побудови графа | **Загальна цільова функція:** Minimize $C_{total}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c \parallel q_i - q_j \parallel$ | – Можливість роботи у високовимірних просторах<br>– Ефективність у складних просторах | – Високі вимоги до обчислювальних ресурсів<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 17 | **Lazy PRM** | Глобальне планування | Відкладення перевірок колізій до моменту запиту | **Загальна цільова функція:** Minimize $C_{\text{total}}(q) = \sum_{(q_i,q_j)\in P} \left(d(q_i,q_j) + \lambda \cdot c(q_i,q_j)\right)$ | – Зменшення кількості перевірок колізій<br>– Зменшення часу виконання | – Можливі проблеми з масштабованістю<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 18 | **Gaussian sampling for PRM** | Глобальне планування | Використання гауссового розподілу для зразкування | **Загальна цільова функція:** $C_r = \{p \in C_{free} \mid \|p-q\| < d \wedge q \in \mathcal{IB} \cup \mathcal{OB}\}$ | – Краще покриття складних областей<br>– Зменшення кількості зразків | – Необхідність налаштування параметрів<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 19 | **Halton sampling for PRM** | Глобальне планування | Використання послідовності Халтона для зразкування | **Загальна цільова функція:** Minimize $C_{total}(q_{start}, q_{goal}) = \sum_{(q_i,q_j)\in P} c \parallel q_i - q_j \parallel$ | – Рівномірний розподіл зразків<br>– Зменшення кількості зразків | – Складність реалізації<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 20 | **D*** (Dynamic A*) | Інкрементальний пошук | Інкрементальний пошук з використанням евристик | **Загальна цільова функція:** Minimize $C_{total}(q_{start}, q_{goal}) = = \sum_{(q_i,q_j)\in P}(\parallel q_i - q_j \parallel + \parallel q_j - q_{goal} \parallel) h(q_i, q_{goal})$ | – Ефективність у динамічних середовищах<br>– Підходить для частково відомих та змінних середовищ | – Високі вимоги до обчислювальних ресурсів<br>– Складність реалізації | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 21 | **Goal-directed and randomized search** | Глобальне планування | Поєднання цілеспрямованого та випадкового пошуку | **Цілеспрямований пошук** Strong (Minimize ∨ Maximize $f(x)$) **Випадковий пошук** Rand (Minimize ∨ Maximize $f(x)$) | - Ефективність у великих просторах<br>– Поєднання переваг цілеспрямованого та випадкового пошуку | – Складність реалізації<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 22 | **SANDROS** | Динамічний графовий пошук | Ієрархічний, нерівномірний багаторівневий пошук | **Загальна цільова функція:** Minimize $\sum_{i=1}^{n} \lvert P_i - P_{target} \rvert$ | – Ефективність у складних середовищах<br>– Висока точність | – Складність реалізації<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 23 | **APF** (Artificial potential field method) | Локальне планування | Використання потенційних полів | **Загальна цільова функція** Minimize $U_{total}(q) = = \frac{1}{2} k_{att} \|q - q_{goal}\|^2 + \sum_{i=1}^{m} \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\|q-q_{obs_i}\|} - \frac{1}{d_0}\right)^2 & if \ \|q - q_{obs_i}\| \le d_0, \\ 0 & if \ \|q - q_{obs_i}\| > d_0. \end{cases}$ | – Простота реалізації<br>– Швидкість обчислень | – Проблеми з локальними мінімумами<br>– Не підходить для складних середовищ | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 24 | **DAPF** (Dynamic artificial potential field method) | Локальне планування | Використання динамічних потенційних полів | **Загальна цільова функція:** Minimize $U_{total}(q,t) = = \frac{1}{2} k_{att} \|q - q_{goal}\|^2 + \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\|q-q_{obs}\|} - \frac{1}{d_0}\right)^2 & if \ \|q - q_{obs}\| \le d_0, \\ 0 & if \ \|q - q_{obs}\| > d_0. \end{cases} + \alpha \|\dot{q}_{obs}(t)\|$ | – Простота реалізації<br>– Швидкість обчислень | – Проблеми з локальними мінімумами<br>– Не підходить для складних середовищ | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |

*Продовження таблиці 1*

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 25 | **IAPF** (Improved artificial potential field) | Локальне планування | Використання потенційних полів з покращеннями | **Загальна цільова функція:** Minimize $U_{total}(q) =$ $= \frac{1}{2}k_{att}\|q - q_{goal}\|^2 + \sum_{i=1}^{m}\begin{cases}\frac{1}{2}k_{rep}\left(\frac{1}{\|q-q_{obs_i}\|}-\frac{1}{d_0}\right)^2 & if\ \|q-q_{obs_i}\| \le d_0,\\ 0 & if\ \|q-q_{obs_i}\| > d_0.\end{cases} + \sum_{i=1}^{n}\frac{1}{2}k_{virt}\|q - q_{virt,i}\|^2$ | – Покращена ефективність уникнення перешкод<br>– Зменшення проблем з локальними мінімумами | – Можливі проблеми з масштабованістю<br>– Не підходить для дуже складних середовищ | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 26 | **Artificial potential field based subgoal network** | Потенційні поля з підцілями | Використання потенційних полів для створення мережі підцілей | **Загальна цільова функція:** $U(x) = \frac{1}{2}k_{att}\left\|x - x_{goal}\right\|^2 + + \sum_{i=1}^{n}\frac{1}{2}k_{subgoal}\left\|x - x_{subgoal_i}\right\|^2 + + \sum_{j=1}^{m}\begin{cases}\frac{1}{2}k_{rep}\left(\frac{1}{\left\|x-x_{obs_j}\right\|}-\frac{1}{d_0}\right)^2 & if\ \left\|x - x_{obs_j}\right\| \le d_0,\\ 0 & if\ \left\|x - x_{obs_j}\right\| > d_0.\end{cases}$ | – Простота реалізації<br>– Швидкість обчислень | – Проблеми з локальними мінімумами<br>– Не підходить для складних середовищ | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 27 | **Dynamic subgoal path planner** | Динамічне планування з підцілями | Генерація підцілей для обходу перешкод | **Загальна цільова функція:** Minimize $J(x) = \alpha \cdot \sum_{i=1}^{n-1}\|x_{i+1} - x_i\| + +\beta \cdot \sum_{i=1}^{n} cost\,(x_i) + \gamma \cdot \sum_{i=2}^{n-1}\|x_{i-1} - 2x_i + x_{i+1}\|$ | – Швидке ухилення від локальних мінімумів<br>– Висока ефективність у динамічних середовищах | – Складність реалізації<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 28 | **Hierarchical motion planner** | Ієрархічне планування руху | Ієрархічна структура для планування руху | **Загальна цільова функція:** Minimize $J = w_1\int_0^T\|\dot{x}(t)\|dt + w_2\sum_{j=1}^{m}\phi\left(d_j(x)\right) + +\mathrm{w}_3\int_0^T P\left(x(t),\dot{x}(t)\right)dt$ | – Зменшення обчислювальної складності<br>– Можливість роботи у високовимірних просторах | – Високі вимоги до обчислювальних ресурсів<br>– Складність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 29 | **Two-layered subgoal algorithm** | Дворівневий алгоритм з підцілями | Використання двох рівнів для планування шляху | **Загальна цільова функція:** $J = \sum_{j=1}^{m} w_j \cdot g_j(y) + \sum_{k=1}^{p} v_k \cdot h_k(z)$ | – Ефективність у складних середовищах<br>– Зменшення обчислювальної складності | – Складність реалізації<br>– Високі вимоги до обчислювальних ресурсів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 30 | **RRT with local trees** | Глобальне планування | Використання локальних дерев для покращення пошуку | **Загальна цільова функція:** Minimize $J =$ $= \sum_{i=1}^{n}(w_1 \cdot \sum_{i=1}^{n} d(x_i, x_{i+1}) + w_2 \cdot \lambda \cdot \sum_{j=1}^{m} c\left(x_j\right) + w_3 \cdot T(x))$ | – Покращена ефективність у складних середовищах<br>– Зменшення кількості перевірок колізій | – Складність реалізації<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 31 | **Obstacle-based RRT** | Глобальне планування | Використання перешкод для керування зразкуванням | **Загальна цільова функція:** Minimize $\mathrm{J} = \mathrm{d}\left(\mathrm{x}_{\mathrm{start}}, \mathrm{x}_{\mathrm{goal}}\right) + \lambda \cdot \sum_{i=1}^{n} CollisionCost\,(x_i) + \mathrm{C}(x_i)$ | – Краще обходження перешкод<br>– Підвищена ефективність у складних середовищах | – Складність реалізації<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 32 | **RRT-connect** | Глобальне планування | Бінаправлений пошук з використанням двох дерев | **Загальна цільова функція:** $Minimize\ J = \sum_{i=1}^{n}\|x_{i+1} - x_i\|\ \ subject\ to\ \ x_i \in C_{free}$ | – Швидке з'єднання старту та цілі<br>– Простота реалізації | – Можливі проблеми з масштабованістю<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |

*Закінчення таблиці 1*

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| 33 | **Bidirectional RRT** | Глобальне планування | Використання двох дерев для зменшення часу пошуку | **Загальна цільова функція:** $\text{Minimize } J = \sum_{i=1}^{n} d(x_i, X_{i+1}) \;\; subject\ to\ x_i \in X_{free}, \forall i$ | – Зменшення часу пошуку<br>– Підвищена ефективність у складних середовищах | – Складність реалізації<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів<br>– Оптимізація маршрутів у логістиці |
| 34 | **RRT*** | Глобальне планування | Використання випадкового зразкування для побудови оптимального шляху | **Загальна цільова функція:** $\text{Minimize } C(x_{start}, x_{goal}) = \sum_{i=1}^{n-1} \|x_{i+1} - x_i\|$ | – Гарантована оптимальність шляху<br>– Простота реалізації | – Високі вимоги до обчислювальних ресурсів<br>– Можливі проблеми з масштабованістю | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 35 | **Informed-RRT*** | Глобальне планування | Оптимізація RRT* з використанням евристики | **Загальна цільова функція:** $\text{Minimize } J(\tau) = \int_0^T c(\tau(t))dt + h(x_{goal}, x)$ | – Швидша конвергенція до оптимального рішення<br>– Покращена якість кінцевого рішення | – Складність реалізації<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для дронів<br>– Оптимізація маршрутів у логістиці |
| 36 | **MBD-RRT*FFT** (Multi–threaded Bidirectional RRT with FFT) | Глобальне планування | Використання багатопотоків их обчислень та алгоритму Fitch для оптимізації | **Загальна цільова функція:** $\text{Minimize } J(x) = \sum_{i=1}^{n-1} \left\| \| x_{i+1} - x_i \| + \lambda \cdot \begin{cases} 0, & f_{collision}(x_i, x_{i+1}) = \infty \\ f(x_i, x_{i+1}), & f_{dynamic}(x_i, x_{i+1}) = \alpha \cdot violation_{degree(x_i, x_{i+1})} \end{cases} + + \mu \cdot FFT(x_i, x_{i+1}) \right\|$ | – Висока ефективність завдяки багатопотоковим обчисленням<br>– Зменшення часу виконання завдяки паралельній обробці даних | – Складність реалізації багатопотокових обчислень<br>– Необхідність налаштування параметрів | – Робототехніка<br>– Автономні транспортні засоби | – Планування шляху для мобільних роботів у динаміч-них середовищах<br>– Оптимізація маршрутів у логістиці |
| 37 | **Динамічні ігри** | Теоретичний підхід для моделю-вання стратегічної взаємодії між агентами | Використання математичних моделей для аналізу стратегій у динамічних середовищах | **Загальна цільова функція:** $J(u) = \int_{t_0}^{t_f} L(x(t), u)t), t)dt + \Phi(x(t_f))$ | – Можливість моделювання складних взаємодій<br>– Використання в різних галузях (економіка, військова справа тощо) | – Складність математичних моделей<br>– Високі вимоги до знань теорії ігор | – Економіка, військова справа, соціальні науки<br>– Моделювання стратегічних взаємодій | – Аналіз ринкових стратегій<br>– Військові стратегії |
| 38 | **Метод розв'язуючих функцій Чикрія** | Теоретичний підхід для розв'язання задач керування в умовах конфлікту | Використання обернених функціоналів для розв'язання задач керування | **Загальна цільова функція:** $J = \int_0^T L(x(t), u)t), t)dt + \Phi(x(T))$ **Оптимальне керування u* (t)** $u^*(t) = \arg\min_u \left[ L(x,u) + \frac{\partial V}{\partial x} f(x,u) \right]$ | – Можливість застосування в різних галузях<br>– Теоретично обґрунтовані рішення | – Високі вимоги до знань теорії керування<br>– Складність реалізації | – Робототехніка, економіка, військова справа<br>– Задачі керування в умовах конфлікту | – Оптимізація маршрутів у логістиці<br>– Військові стратегії |

СПИСОК ЛІТЕРАТУРИ ТА ВИКОРИСТАНІ ДЖЕРЕЛА

1. Бернацький А. П. Основи робототехніки військового призначення / А. П. Бернацький. Київ: видав. ЛІРА-К, 2024. 498 с.

2. Ладієва Л. Р. Методи оптимізації та пошуку оптимальних рішень. Електронне мережне навчальне видання / У. Л. Р.Ладієва. Київ: КПІ ім. Ігоря Сікорського, 2023. 73 с.

3. Shi Wei Li. Verification and Analysis of Two-dimensional Path Planning Objective Function Optimization Based on Classical Particle Swarm Optimization Algorithm / 2021 4th International Conference on Intelligent Robotics and Control Engineering (IRCE), 18-20 September 2021. URL: https://doi.org/10.1109/IRCE53649.2021.9570874.

4. Sidhu. Performance Evaluation of Pathfinding Algorithms [Electronic resource] / Sidhu, H. Kaur // Scholarship at UWindsor. 2020. Mode of access: https://scholar.uwindsor.ca/etd/8178.

5. Samridh Garg; Bhanu Devi Shortest Path Finding using Modified Dijkstra's algorithm with Adaptive Penalty Function/ 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT). 06-08 July 2023. Mode of access: https://doi.org/10.1109/ICCCNT56998.2023.10308130.

6. Stephan Weiser, Hans Wulf, Jörn Ihlemann Application of a deepest-path algorithm to study the objective function landscape during fitting for the Yeoh and Ogden model / PAMM Proc. Appl. Math. Mech. 2021. Mode of access: https://doi.org/10.1002/pamm.202100086.

7. Матвійчук Р. Д., Данільчук О. М. Порівняння найвідоміших алгоритмів пошуку / Вісник студентського наукового товариства ДонНУ ім. Василя Стуса, 2022. С. 230–234.

8. Проценко А. А., Іванов В. Г. Класичні методи планування шляху для мобільних роботів / Системи управління навігації та зв'язку. 3(55) June 2019. С. 143–151. URL: https://doi.org/10.26906/SUNZ.2019.3.143.

9. Pankaj K. Agarwal, Esther Ezra, Micha Sharir Vertical Decomposition in 3D and 4D with Applications to Line Nearest-Neighbor Searching in 3D / arXiv:2311.01597v1 [cs.CG] 2 Nov 2023. Mode of access: https://doi.org/10.48550/arXiv.2311.01597.

10. Sleumer, Nora; Tschichold-Gürmann, Nadine Exact cell decomposition of arrangements used for path planning in robotics/Technical Report / ETH Zurich, Department of Computer Science 329. 1999. Mode of access: https://doi.org/10.3929/ethz-a-006653440.

11. Choset H. Coverage Path Planning: The Boustrophedon Cellular Decomposition / H. Choset, P.Pignon Field and Service Robotics Springer-Verlag London Limited 1998 pp 203–209. Mode of access: https://doi.org/10.1007/978-1-4471-1273-0_32.

12. Bott, Raoul. Morse Theory Indomitable / Bott, Raoul., - Publications Mathématiques de l'IHÉS. 68: 1988., pp. 99–114. Mode of access: DOI:10.1007/bf02698544.

13. Dłotko P. Computing homology and persistent homology using iterated Morse decomposition / P. Dłotko, H. Wagne: arXiv:1210.1429v2 [math.AT] 25 Oct 2012. Mode of access: https://doi.org/10.48550/arXiv.1210.1429.

14. Rasolzadah K. Morse Theory and Handle Decomposition / K. Rasolzadah. Department of Mathematics Uppsala University, Februari 2018.

15. R. Bahnemann, Revisiting Boustrophedon Coverage Path Planning as a Generalized Traveling Salesman Problem / R. Bahnemann, N. Lawrance arXiv:1907.09224v1 [cs.RO] 22 Jul 2019. URL: https://doi.org/10.1007/978-981-15-9460-1_20.

16. Laumond, J.-P. (Ed.) Robot Motion Planning and Control. London: Spring-Verlag Limited. 1998.

17. Chenghui Cai, Silvia Ferrari Information-Driven Sensor Path Planning by Approximate Cell Decomposition / IEEE transactions on systems, man, and cybernetics, vol. 39, No. 3, June 2009. URL: https://doi.org/10.1109/TSMCB.2008.2008561.

18. Chenghui Cai, Silvia Ferrari Information-driven sensor path planning and the treasure hunt problem / Dissertation Duke University 2008. 112 p.

19. Ramon Gonzalez; Marius Kloetzer; Cristian Mahulea Comparative study of trajectories resulted from cell decomposition path planning approaches / 2017 21st International Conference on System Theory, Control and Computing (ICSTCC). URL: https://doi.org/10.1109/ICSTCC.2017.8107010.

20. Prabhakar Reddy G.V.S., Hubert J. Montas, Hanan Samet, Adel Shirmohammadi Quadtree-Based Triangular Mesh Generation for Finite Element Analysis of Heterogeneous Spatial Data / 2001 ASAE Annual International Meeting Sacramento, California, USA, July 30-August 1, 2001. 25 p.

21. Huiwei Wang; Yaqian Huang; Huaqing Li Quadtree-Based Adaptive Spatial Decomposition for Range Queries Under Local Differential Privacy / IEEE Transactions on Emerging Topics in Computing (Volume: 11, Issue: 4, Oct.-Dec. 2023). pp 1045-1056. URL: https://doi.org/10.1109/TETC.2023.3317393.

22. José Lima The K-Framed Quadtrees Approach for Path Planning Through a Known Environment / ROBOT 2017: Third Iberian Robotics Conference. 2017.

23. Zhou Yijun, Xi Jiadong, Xi Jiadong, Luo Chen A Fast Bi-Directional A* Algorithm Based on Quad-Tree Decomposition and Hierarchical Map / Preparation of Papers for IEEE Access (February 2017). URL: http://dx.doi.org/10.1109/ACCESS.2021.3094854.

24. Ana Rodrigues, Pedro Costa, José Lima The K-Framed Quadtrees Approach for Path Planning Through a Known Environment / November 2018 Advances in Intelligent Systems and Computing 2018. URL: http://dx.doi.org/10.1007/978-3-319-70833-1_5.

25. Ana Rodrigues, Pedro Costa, José Lima The K-framed quadtrees approach for path planning through a known environment / Advances in Intelligent Systems and Computing Search for Book Publications. 2018., pp 49–59. URL: http://doi.org/10.1007/978-3-319-70833-1_5.

26. J. W. Burns; N. S. Subotic; D. Pandelis Adaptive decomposition in electromagnetics / IEEE Antennas and Propagation Society International Symposium. 1997. URL: https://doi.org/10.1109/APS.1997.631726.

27. Xiaomei Zhang, Xiangyu Yun, Zhe Wang, Mohan Li, Jinming Hu, Chengmin Wang, Cunfeng Wei An adaptive decomposition algorithm for quantitative spectral CT imaging / X-RAY Spectrometry Vol.53, Iss.4., August 2024., pp. 282–293. URL: https://doi.org/10.1002/xrs.3365.

28. Tao Liu, Zhijun Luo, Jiahong Huang, Shaoze Yan, A Comparative Study of Four Kinds of Adaptive Decomposition Algorithms and Their Applications / PMC PubMed Central 2018 Jul; 18(7): 2120. URL: https://doi.org/10.3390 s18072120.

29. Sabudin E. N, Omar. R and Che Ku Melor C. Potential Field Methods And Their Inherent Approaches For Path Planning / ARPN Journal of Engineering and Applied Sciences Vol. 11, No. 18, Sept. 2016., pp. 10801-10805.

30. Jan Rosell, Pedro Iñiguez Path planning using Harmonic Functions and Probabilistic Cell Decomposition / IEEE Xplore, Conference: Robotics and Automation, ICRA 2005. URL: http://dx.doi.org/10.1109/ROBOT.2005.1570375.

31. Emmanuel f Mazer, Juan-Manuel Ahuactzin, Pierre Bessiere The Ariadne's Clew Algorithm / Journal of Artificial Intelligence Research. May 1998., pp. 295–316. URL: http://dx.doi.org/10.1613/jair.468.

32. J. M. Ahuactzin, P. Bessiere, E. Mazer The Ariadne's Clew Algorithm / Journal of Artificial Intelligence Research 1998. pp. 295–316. URL: https://doi.org/10.48550/arXiv.1105.5440.

33. David Hsu, Jean-claude Latombe, Rajeev Motwani Path Planning in Expansive Configuration Spaces / International Journal of Computational Geometry & Applications 09(04n05) March 1997. URL: http://dx.doi.org/10.1142/S0218195999000285.

34. Arushi Khokhar Probabilistic Roadmap (PRM) for Path Planning in Robotics / Medium., Feb 12, 2021.

35. Lijun Qiao, Xiao Luo, Qingsheng Luo An Optimized Probabilistic Roadmap Algorithm for Path Planning of Mobile Robots in Complex Environments with Narrow Channels / Sensors 2022, 22(22), 8983. URL: https://doi.org/10.3390/s22228983.

36. S. Carpin Algorithmic Motion Planning: The Randomized Approach/ General Theory of Information Transfer and Combinatorics. 2006. pp. 740–768.

37. Jinbao Chen, Yimin Zhou; Jin Gong; Yu Deng An Improved Probabilistic Roadmap Algorithm with Potential Field Function for Path Planning of Quadrotor / 2019 Chinese Control Conference (CCC). July 2019. URL: https://doi.org/10.23919/ChiCC.2019.8865585.

38. Nancy M. Amato, Yan Wu Randomized roadmap method for path and manipulation planning / IEEE International Conference on Robotics and Automation vol. 1., May 1996. pp. 113–120. URL: http://dx.doi.org/10.1109/ROBOT.1996.503582.

39. Robert Bohlin, Lidia E. Kavraki Path Planning Using Lazy / IEEE International Conference on Robotics & Automation, San Francisco, CA., April 2000. pp. 521–528. URL: https://doi.org/10.1109/ROBOT.2000.844107.

40. Jory Denny; Kensen Shi; Nancy M. Amato Lazy Toggle PRM: A single-query approach to motion planning / 2013 IEEE International Conference on Robotics and Automation., Karlsruhe, Germany. 2013. URL: https://doi.org/10.1109/ICRA.2013.6630904.

41. Boor, V., Overmars, M.H., Van Der Stappen, A.F.: The Gaussian sampling strategyfor probabilistic roadmap planners. In: RSS. vol. 2, pp. 1018–1023. IEEE (1999). URL: https://doi.org/10.1109/ROBOT.1999.772447.

42. Steven M. LaValle, Michael S. Branicky On the Relationship Between Classical Grid Search and Probabilistic Roadmaps / The International Journal of Robotics Research Vol.23, Iss. 7-8 2004. URL: https://doi.org/10.1177/0278364904045481.

43. Huageng Zhong, Ming Cong, Minghao Wang, Yu Du, Dong Liu HB-RRT:A path planning algorithm for mobile robots using Halton sequence-based rapidly-exploring random tree / Engineering Applications of Artificial Intelligence., Vol. 133, Part E., July 2024. URL: https://doi.org/10.1016/j.engappai.2024.108362.

44. Anthony Stentz The D* Algorithm for Real-Time Planning of Optimal Traverses / The Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania 15213, September 1994.

45. Firas A. Raheema, Umniah I. Hameed Path Planning Algorithm using D* Heuristic Method Based on PSO in Dynamic Environment / American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS) (2018) Volume 49, No 1, pp. 257–271.

46. Dibyendu Biswas D*, D* Lite & LPA* / Medium Jun 27, 2021.

47. Reinhard Bauer, Daniel Delling, Peter Sanders, Dennis Schieferdecker, Dominik, Schultes, Dorothea Wagner Combining Hierarchical and Goal-Directed Speed-Up Techniques for Dijkstra's Algorithm / Experimental Algorithms (WEA 2008). pp.303–318.

48. Akash Lai, Shaz Qadeer A program transformation for faster goal-directed search / 2014 Formal Methods in Computer-Aided Design (FMCAD) October 2014. URL: https://doi.org/10.1109/FMCAD.2014.6987607.

49. Akash Lal, Shaz Qadeer A Program Transformation for Faster Goal-Directed Search / FMCAD '14: Proceedings of the 14th Conference on Formal Methods in Computer-Aided Design 2014, pp. 147–154.

50. Pang C. Chen, Y. Hwang SANDROS: a motion planner with performance proportional to task difficulty/ IEEE Transactions on Robotics and Automation (Vol: 14, Iss. 3, June 1998) pp. 390–403. URL: https://doi.org/10.1109/70.678449.

51. Iswanto, Alfian Ma'arif, Oyas Wahyunggoro, Adha Imam Cahyadi Artificial Potential Field Algorithm Implementation for Quadrotor Path Planning / (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, No. 8, 2019. URL: https://dx.doi.org/10.14569/IJACSA.2019.0100876.

52. Eryi Zhang Path planning algorithm based on Improved Artificial Potential Field method / Applied and Computational Engineering September 2023. pp. 167–174. URL: http://dx.doi.org/10.54254/2755-2721/10/20230170.

53. Wenrui Wang, Mingchao Zhu, Zhenbang Xu An improved artificial potential field method of trajectory planning and obstacle avoidance for redundant manipulators / International Journal of Advanced Robotic Systems, September 2018. URL: https://doi.org/10.1177/1729881418799562.

54. Hu Hongyu, Zhang Chi, Sheng Yuhuan, Zhou Bin, Gao Fei An Improved Artificial Potential Field Model Considering Vehicle Velocity for Autonomous Driving / IFAC-PapersOnLine Vol. 51, Iss. 31., 2018. pp. 863–867. URL: https://doi.org/10.1016/j.ifacol.2018.10.095.

55. Kaddour Messaoudi, Noureddine Chaib A survey of UAV-based data collection: Challenges, solutions and future perspectives / Journal of Network and Computer Applications Vol. 216, July 2023. URL: https://doi.org/10.1016/j.jnca.2023.103670.

56. Lijuan Xie, Huanwen Chen, Guangrong Xie Artificial Potential Field Based Path Planning for Mobile Robots Using Virtual Water-Flow Method / Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques (ICIC 2007). Pp. 588–595. URL: http://dx.doi.org/10.1007/978-3-540-74282-1_66.

57. Jixue Mo, Gao Changqing, Fei Liu, Qingkai Yang A Modified Artificial Potential Field Method Based on Subgoal Points for Mobile Robot / ICIRA2023-The 16th international conference on intelligent robotics and applications., July 2023. URL: http://dx.doi.org/10.1007/978-981-99-6483-3_26.

58. Hong Liu; Weiwei Wan; Hongbin Zha A dynamic subgoal path planner for unpredictable environments / 2010 IEEE International Conference on Robotics and Automation., Anchorage, AK, USA., May 2010. URL: https://doi.org/10.1109/ROBOT.2010.5509324.

59. David J. Grymin, Charles B. Neas, Mazen Farhood A hierarchical approach for primitive-based motion planning and control of autonomous vehicles / Robotics and Autonomous Systems., Vol. 62, Iss. 2, Feb. 2014, pp. 214–228. URL: https://doi.org/10.1016/j.robot.2013.10.003.

60. Hanlin Chen, Xizhe Zang, Yubin Liu, Xuehe Zhang, Jie Zhao A Hierarchical Motion Planning Method for Mobile Manipulator / Sensors 2023, 23(15), 6952. URL: https://doi.org/10.3390/s23156952.

61. Yao Qi, Binbing He, Rendong Wang, Le Wang, Youchun Xu Hierarchical Motion Planning for Autonomous Vehicles in Unstructured Dynamic Environments / IEEE Robotics and Automation Letters ( Volume: 8, Issue: 2, February 2023). pp. 496–503. URL: https://doi.org/10.1109/LRA.2022.3228159.

62. Amitava Chatterjee, Anjan Rakshit, N. Nirmal Singh Vision-Based Mobile Robot Navigation Using Subgoals/ Vision Based Autonomous Robot Navigation., vol. 44, issue 4, May 2011. pp. 620–641.

63. Nirmal Singh, Avishek Chatterjee, Amitava Chatterjee, Anjan Rakshit A two-layered subgoal based mobile robot navigation algorithm with vision system and IR sensors / May Measurement 44(4) 2011. pp. 620–641. URL: http://dx.doi.org/10.1016/j.measurement.2010.12.002.

64. LaValle, Steven M. Rapidly-exploring random trees: A new tool for path planning / Technical Report (TR 98–11). Computer Science Department, Iowa State University. October 1998.

65. J. J. Kuffner; S. M. LaValle RRT-connect: An efficient approach to single-query path planning / Millennium Conference. IEEE International Conference on Robotics and Automation. April 2000. https://doi.org/10.1109/ROBOT.2000.844730

66. Fan Yang, Xi Fang, Fei Gao, Xianjin Zhou, Hao Li, Hongbin Jin, Yu Song Obstacle Avoidance Path Planning for UAV Based on Improved RRT Algorithm / Discrete Dynamics in Nature and Society, Iss. 1., 2022. URL: https://doi.org/10.1155/2022/4544499.

67. Xiong Yin, Wentao Dong, Xiaoming Wang, Yongxiang Yu, Daojin Yao Route planning of mobile robot based on improved RRT star and TEB algorithm / Scientific reports 8942. 2024. URL: https://doi.org/10.1038/s41598-024-59413-9.

68. Xinyu Tang, Jyh-Ming Lien, Nancy Amato OBRRT: Obstacle-Based RRT/ IEEE International Conference on Robotics and Automation. 2006. pp. 895–900. URL: http://dx.doi.org/10.1109/ROBOT.2006.1641823.

69. Xin Cheng, Jingmei Zhou, Zhou Zhou, Xiangmo Zhao An improved RRT-Connect path planning algorithm of robotic arm for automatic sampling of exhaust emission detection in Industry 4.0 / Journal of Industrial Information Integration Vol. 33., June 2023. URL: https://doi.org/10.1016/j.jii.2023.100436.

70. J. J. Kuffner and S. M. LaValle RRT-Connect path solving/ CRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065). URL: https://doi.org/10.1109/ROBOT.2000.844730.

71. Rui Zhang, He Guo, Darius Andriukaitis, Yongbo Li Intelligent path planning by an improved RRT algorithm with dual grid map / Alexandria Engineering Journal Vol. 88., February 2024, pp. 91–104. URL: https://doi.org/10.1016/j.aej.2023.12.044.

72. Jun Ding,Yinxuan Zhou, Xia Huang An improved RRT* algorithm for robot path planning based on path expansion heuristic sampling / Journal of Computational Science Vol. 67, March 2023. URL: https://doi.org/10.1016/j.jocs.2022.101937.

73. James J. Kuffner, Jr. Steven M. LaValle RRT-Connect: An Efficient Approach to Single-Query Path Planning / In Proc. 2000 IEEE Int’l Conf. on Robotics and Automation (ICRA 2000) pp. 995–1001. URL: http://dx.doi.org/10.1109/ROBOT.2000.844730.

74. Zhe Huang, Hongyu Chen, John Pohovey, Katherine Driggs-Campbe Neural Informed RRT*: Learning-based Path Planning with Point Cloud State Representations under Admissible Ellipsoidal Constraints / arXiv:2309.14595v2 [cs.RO] 07 Mar 2024. URL: http://dx.doi.org/10.1109/ICRA57147.2024.10611099.

75. Бернацький А. П. Вдосконалений метод планування шляху автономного наземного робота з використанням алгоритму MBD-RRT*FFT / Системи і технології зв’язку, інформатизації та кібербезпеки. Випуск № 5, 2024. URL: https://doi.org/10.58254/viti.5.2024.03.37.

76. Anita Garhwal, Partha Pratim A Survey on Dynamic Spectrum Access Techniques for Cognitive Radio / International Journal of NextGeneration Networks (IJNGN). Vol. 3, No. 4, 2012. URL: http://dx.doi.org/10.5121/ijngn.2011.3402.

77. P. Taylor, L. Jonker, Evolutionary stable strategies and game dynamics /Mathematical Biosciences 40 (1–2) (1978). pp.145–156.

78. Чикрій А. О. Вимірні багатозначні відображення та їх селектори в динамічних іграх переслідування / А. А. Чикрій, І. С. Раппопорт // Пробл. упр. та інформатики. № 1-2. 2006. С. 60–70.

79. Shaobo Zhang, Qinxiang Xia, Mingxing Chen, Sizhu Cheng Multi-Objective Optimal Trajectory Planning for Robotic Arms Using Deep Reinforcement Learning / Sensors 2023, 23(13), 5974. URL: https://doi.org/10.3390/s23135974.

80. Masoud Fetanat; Sajjad Haghzad; Saeed Bagheri Shouraki Optimization of dynamic mobile robot path planning based on evolutionary methods / AI & Robotics (IRANOPEN). 2015. URL: https://doi.org/10.1109/RIOS.2015.7270743.

81. Альбус Дж. Аналітичний метод вирішення ігрового завдання про "М’яку посадку" для рухомих об’єктів / Дж. Альбус, А. Мейстел, А. О. Чикрій, А. А. Білоусов, А. І. Козлов // Кібернетика та систем. аналіз. № 1. 2001. С. 97–115

82. Чикрій А. О Квазилінійні конфліктно керовані процеси зі змінною структурою / А. А. Чикрій, І. І. Матичин // Пробл. упр. та інформатики. № 6. 1998. С. 31–41.

83. Чикрій А. О. Квазилінійні позиційні інтегральні ігри зближення / А. О. Чикрій, Г. Ц. Чикрій, К. Ю. Волянський // Пробл. упр. та інформатики. № 6. 2001. С. 5–28.

84. Chikrii, A. Conflict-Controlled Processes / Springer Science &Business Media. 2013.

85. Барановська Л. В., Гирявець Д. М., Барановська Г. Г. Візуалізація групової гри переслідування на площині / Conference: Information Technologies and Security (ITS-2020). 2021.