Науковий вісник НЛТУ України
Scientific Bulletin of UNFU
https://doi.org/10.36930/40340518
Article received 12.05.2024
Article accepted 23.05.2024
UDC 004.[02+942]:62-50
Correspondence author: V. S. Yakovyna vitaliy.s.yakovyna@lpnu.ua
ISSN 1994-7836 (print)
ISSN 2519-2477 (online)

А. Я. Медвідь, В. С. Яковина
Національний університет "Львівська політехніка", м. Львів, Україна

ІНТЕГРАЦІЯ ДАНИХ ПРО КОЛІЗІЇ ДЛЯ ПРИШВИДШЕННЯ ОПТИМІЗАЦІЇ ТРАЄКТОРІЇ РУХУ РОБОТИЗОВАНОЇ РУКИ

Представлено новий підхід до покращення оптимізації траєкторії руху роботизованої руки шляхом інтеграції даних про колізії, отримані під час планування шляху переміщення роборуки. Планування шляху переміщення роборуки роботизованих систем є однією з ключових задач в сучасній робототехніці, особливо для роботизованої руки, яку використовують в промисловості для виконання складних маніпуляцій. У багатьох дослідженнях розглянуто розроблення алгоритмів для планування траєкторій руху, які забезпечують уникнення колізій та мінімізацію витрат часу і ресурсів на виконання обчислень. Незважаючи на чисельні вдосконалення у методах планування та оптимізації траєкторій руху руки, існує потреба у додаткових підходах, які можуть покращити ефективність цих процесів. Основна мета дослідження полягала у розробленні методу, що дає змогу пришвидшити оптимізацію траєкторії руху роботизованої руки шляхом інтеграції даних про колізії. Визначено основні завдання дослідження: розробити алгоритм, що використовує попередньо відомі дані про колізії для пришвидшення перевірок на колізії прямих відрізків; провести експериментальні дослідження ефективності запропонованого методу; оцінити вплив запропонованого підходу на загальну швидкість і якість оптимізації траєкторії. Під час проведення тестувань контролювали такі параметри, як кількість викликів функції перевірки на колізії, довжина оптимізованої траєкторії та тривалість виконання алгоритму. Вхідним параметром тестувань був поріг перевірки на колізії щодо відстані до заданого відрізка. Під час проведення експериментальних досліджень отримано результати, що підтверджують ефективність підходу. Внаслідок використання порогу відстані на рівні п'ять кроків дискретизації (0,175 рад.) загальна кількість перевірок на колізії знизилася від 82663 до 62056 шт., тобто на 24,93 %, при цьому загальна тривалість виконання скоротилась з 16,2 до 10,97 с, тобто на 32,29 %. У разі використання порогу відстані на рівні десяти кроків дискретизації (0,35 рад.) кількість перевірок знизилася до 53139 шт., тобто на 35,72 %, а тривалість виконання скоротилась до 9,1 с, тобто на 43,83 %. Запропонований підхід можна інтегрувати з іншими методами планування та оптимізації траєкторій руху роботизованої руки, що підвищує його універсальність і застосовність у різних робототехнічних системах. Результати дослідження підтверджують перспективність використання цього підходу для оптимізації траєкторій руху роботизованих систем, що може бути корисним для подальшого розвитку технологій автоматизації в промисловості.

Ключові слова: планування траєкторії руху; роботизована рука; оптимізація шляху; мінімізація витрат часу.

Вступ / Introduction

Планування шляху переміщення роборуки роботизованих систем є однією з ключових задач в сучасній робототехніці. Особливо це стосується роботизованої руки, яку використовують у промисловості для виконання складних маніпуляцій. У багатьох дослідженнях розроблено алгоритми для планування траєкторій руху, які забезпечують уникнення колізій та мінімізацію витрат часу і ресурсів на обчислення. Серед найбільш популярних підходів можна назвати Probabilistic Roadmaps (PRM) [5], Rapidly-exploring Random Trees (RRT) [7], а також їх оптимальні версії Lazy PRM [1], Adaptive Lazy PRM [6], PRM* [4], RRT* [9], Informed RRT-Connect [9] та інші.

Деякі з методів планування, зокрема RRT* [9] та PRM* [4], передбачають, окрім пошуку шляху переміщення роборуки, також його оптимізацію, завдяки повторному перепланюванню зв'язків між вузлами. Проте багато інших ефективних методів (наприклад, більшість модифікацій RRT) потребують додаткової оптимізації траєкторії для зменшення її довжини та часу виконання відповідно.

До популярних методів оптимізації довжини траєкторії можна віднести: згладжування шляху (англ. *Path Smoothing*), метод коротких замикань (англ. *Shortcutting*), оптимізацію сплайнів (англ. *Spline Optimization*) та інші [10]. Частиною більшості методів оптимізації шляху переміщення роборуки є перевірка колізій на прямих відрізках між парою точок. Під час оптимізації траєкторії алгоритми часто перевіряють наявність колізій в тих регіонах, які вже були оброблені раніше під час пошуку шляху переміщення роборуки в алгоритмі планування.

Об'єкт дослідження – оптимізація траєкторії руху роботизованої руки.

Інформація про авторів:
Медвідь Андрій Ярославович, аспірант, кафедра систем штучного інтелекту.
Email: andrii.y.medvid@lpnu.ua; https://orcid.org/0009-0001-4128-4973
Яковина Віталій Степанович, д-р техн. наук, професор, кафедра систем штучного інтелекту.
Email: vitaliy.s.yakovyna@lpnu.ua; https://orcid.org/0000-0003-0133-8591

Цитування за ДСТУ: Медвідь А. Я., Яковина В. С. Інтеграція даних про колізії для пришвидшення оптимізації траєкторії руху роботизованої руки. Науковий вісник НЛТУ України. 2024, т. 34, № 5. С. 136–143.
Citation APA: Medvid, A. Ya., & Yakovyna, V. S. (2024). Integration of collision data for accelerating trajectory optimization of robotic arms. *Scientific Bulletin of UNFU*, 34(5), 136–143. https://doi.org/10.36930/40340518

Предмет дослідження – методи використання даних про колізії, отриманих під час планування шляху переміщення роборуки роботизованої руки основним алгоритмом, для пришвидшення перевірок на колізії прямих відрізків у процесі оптимізації її траєкторії руху.

Мета роботи – розробити та дослідити метод, який дає змогу пришвидшити оптимізацію траєкторії руху роботизованої руки шляхом інтеграції даних про колізії, що дасть змогу верифікувати розроблений метод та підтвердити ефективність його роботи.

Для досягнення зазначеної мети визначено такі *основні завдання дослідження*:

1) розробити алгоритм, що використовує попередньо відомі стани роботизованої руки, що містять колізії з навколишнім середовищем, для пришвидшення перевірок на колізії прямих відрізків, що дасть змогу скоротити кількість викликів методу перевірки на колізії і як результат скоротить тривалість виконання методу оптимізації довжини траєкторії руху роботизованої руки;
2) провести експериментальні дослідження ефективності запропонованого методу, що дасть змогу верифікувати розроблений метод та підтвердити його ефективність;
3) оцінити вплив запропонованого підходу на загальну швидкість і якість оптимізації траєкторії руху роботизованої руки, що забезпечить скорочення тривалості обчислення оптимізованої траєкторії руху роботизованої руки шляхом інтеграції даних про колізії.

Аналіз останніх досліджень та публікацій. Останні дослідження розглядають сучасні підходи до згладжування траєкторій руху в навігації роботизованих систем. Особливу увагу приділено методам оптимізації довжини траєкторії, які необхідні для забезпечення плавного руху роботів без різких поворотів. В одному з оглядів представлено основні алгоритми згладжування траєкторій руху, в т.ч. й інтерполяційні методи, криві Безьє, B-сплайни та криві NURBS. Окрім цього, розглянуто складніші підходи, такі як використання Дубінсових кривих та клотоїд для створення траєкторій руху зі збереженням безперервності кривизни. Автори детально аналізують переваги та недоліки кожного з методів, а також окреслюють поточні та майбутні виклики у сфері оптимізації траєкторій руху, в т.ч. й інтеграцію цих підходів в реальні системи роботів для досягнення більшої ефективності та безпеки [10].

Серед найновіших досліджень у сфері оптимізації довжини шляху переміщення роботизованої руки особливу увагу привертає робота, де представлено метод згладжування траєкторії на основі B-сплайнів, що дає змогу ефективно видаляти гострі кути на шляху друку. Це має важливе значення для покращення кінематичної продуктивності роботизованої руки та забезпечення вищих кутових швидкостей суглобів, що, водночас, дає змогу збільшити швидкість подачі. Також автори пропонують стратегію планування швидкості подачі для згладженого шляху переміщення роборуки, що враховує кінематичні обмеження шести суглобів роботизованої руки. Результати комп'ютерних симуляцій та фізичних експериментів підтвердили ефективність запропонованих методів, що свідчить про їхню перевагу у сфері адитивного виробництва за допомогою роботів [17].

Робота [2] пропонує модель енергоспоживання роботизованої руки, засновану на кінематичних аналізах та експериментальних даних. Основний акцент зроблено на плануванні шляху переміщення роборуки, яке забезпечує найнижче можливе енергоспоживання під час виконання завдань, таких як маніпуляція об'єктами. Використання GPU дає змогу значно пришвидшити обчислення генетичних алгоритмів, забезпечуючи ефективне планування шляхів з урахуванням кінематичних обмежень роботизованої руки. Результати числових експериментів підтверджують, що оптимізовані шляхи, розроблені за допомогою цього підходу, мають істотні переваги щодо зменшення енергоспоживання порівняно з альтернативними методами.

Хоча тема оптимізації траєкторій руху добре вивчена, мало уваги приділяють використанню відомих даних про колізії для пришвидшення пошуку або оптимізації шляху переміщення роборуки у науковій літературі. Здебільшого дослідники зосереджуються на розробленні нових алгоритмів та методів згладжування траєкторій руху, але не використовують наявну інформацію про колізії для підвищення ефективності своїх підходів. У цій публікації застосовано відому інформацію про колізії, отриману під час планування шляху переміщення роборуки основним алгоритмом, а також інформацію, отриману під час оптимізації, для пришвидшення перевірок на колізії прямих відрізків у процесі оптимізації траєкторії руху роботизованої руки. Такий підхід дають змогу значно скоротити час оптимізації та підвищити загальну ефективність алгоритму.

Матеріали та методи дослідження. Для проведення досліджень використовували роботизовану руку UFACTORY xArm7 [15]. Дослідження виконували на моделі робота, розробленого компанією SOMATIC Holdings Ltd. Частина програмного забезпечення також належить цій компанії.

Для планування траєкторій руху використовували алгоритм RRIS (англ. *Recursive Random Intermediate State*). Основна ідея методу полягає в генеруванні випадкових проміжних станів і виборі найкращого з них на основі кількості колізій на шляхах від початкового стану до проміжного і від проміжного до остаточного стану. Алгоритм працює рекурсивно, розділяючи задачу на підзадачі, і використовує стратегію раннього виходу для підвищення швидкості. Повний опис алгоритму буде опубліковано пізніше.

Для оптимізації траєкторії використовували алгоритм на основі динамічного програмування, заснований на ідеї методу коротких замикань. Зазначимо, що підхід до пришвидшення роботи оптимізації, який описаний тут, повинен працювати незалежно від алгоритму, яким планувалася чи оптимізувалася траєкторія.

Під час проведення експериментів контролювали такі параметри та змінні:

- кількість викликів функції перевірки на колізії: це один з основних параметрів, оскільки ця функція зазвичай має найбільшу обчислювальну складність в алгоритмах планування та оптимізації. Часті виклики функції перевірки на колізії можуть значно впливати на загальну тривалість виконання алгоритму;
- довжина оптимізованої траєкторії руху роботизованої руки: цей параметр дає змогу оцінити ефективність алгоритму з погляду мінімізації шляху переміщення роборуки та верифікації коректної роботи алгоритму. Довжина траєкторії дорівнює сумі поворотів кутів суглобів між послідовними станами роботизованої руки;
- тривалість виконання алгоритму: окрім кількості колізій відстежуємо також і час роботи алгоритму, щоб впевнитися, що час роботи додаткового коду не перевищує виграшу від зменшення кількості перевірок на колізії.

Результати дослідження та їх обговорення / Research results and their discussion

Підхід до пришвидшення оптимізації довжини траєкторії руху роботизованої руки. Експериментальні дослідження нашого підходу відбувалися на траєкторіях руху роботизованої руки з сімома степенями свободи, відповідно до кількості шарнірів. При цьому переміщення кінцевої точки роборуки відбувалися у 3-вимірному декартовому просторі. Приклад спланованої траєкторії руху можна побачити на рис. 1. Сфери зеленого кольору на рисунку відображають шлях маніпулятора.

Оскільки алгоритм оптимізації шляху переміщення роборуки застосовують після алгоритму планування, можна припустити, що перевірка на колізії відбуватиметься в тих регіонах, де вже багато станів перевірялися на колізії під час планування. Враховуючи це, запропоновано використовувати інформацію про стани, які містять колізії, для значного скорочення кількості перевірок на колізії під час оптимізації. Цей підхід дає змогу зменшити тривалість виконання алгоритму і підвищити ефективність його роботи.

Опис підходу:

1. Збереження станів з колізіями: під час планування та оптимізації шляху переміщення роборуки усі стани, що містять колізії, запам'ятовуються в окремий список.
2. Пошук найближчого стану з колізією: під час перевірки колізій на прямому відрізку, алгоритм проходить по всіх відомих станах з колізіями і знаходить найближчий стан на відрізку від стану, що містить колізію.
3. Ігнорування крайніх станів: якщо найближчий стан є одним з крайніх, ним нехтують, оскільки крайні стани у цьому алгоритмі повинні бути вільні від колізій.
4. Ігнорування далеких станів: якщо відстань до найближчого стану більша за деякий поріг (буде підібрано емпірично під час проведення дослідження), то таким станом теж нехтують, оскільки ймовірність, що на відрізку може бути та сама область колізії, є невисокою.
5. Перевірка найближчого стану на колізії: якщо відстань менша за встановлений поріг, то обирається найближчий стан на відрізку і здійснюється перевірка його на наявність колізій. Якщо стан містить колізії, його відкидають.
6. Дискретизація шляху переміщення роборуки та повторна перевірка: якщо на попередніх кроках алгоритму не знайдено колізій, то шлях дискретизується із заданим кроком і усі стани повторно перевіряють на наявність колізій. Якщо колізій немає, то відрізок вважають вільним від колізій.

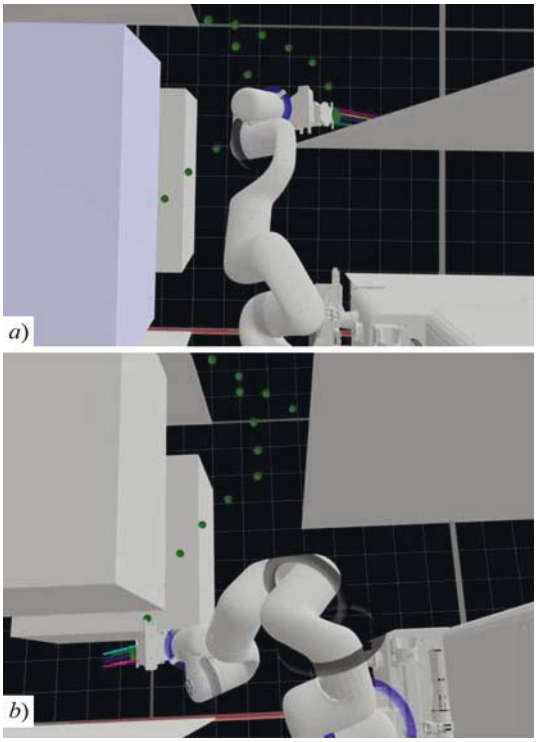

Рис. 1. Приклад спланованої траєкторії руху роботизованої руки / Example of planned robotic arm path: a) початковий стан / start state; b) остаточний стан / final state

Реалізація підходу. Частину програмного коду, яка відповідає за реалізацію цього підходу, подано нижче.

```
double Dot(const model::ArmStateDiff& diff1, const model::ArmStateDiff& diff2) {
double res = 0.0;
for (size_t joint = 0; joint < model::ArmState::DOF; joint++) {
res += diff1[joint] * diff2[joint];
}
return res;
}
```

Код 1. Функція знаходження скалярного добутку між двома станами руки / Function for finding the scalar product between two states of a robotic arm

```
std::optional<model::ArmState> ClosestArmStateOnSegment(
const model::ArmState& segmentStart,
const model::ArmState& segmentFinish,
const model::ArmState& state,
double maxDistToSegment) {
model::ArmStateDiff segmentDiff = segmentFinish – segmentStart;
model::ArmStateDiff stateFromStartDiff = state – segmentStart;
double segmentDiff2 = std::pow(segmentDiff.Norm(model::NormType::Euclidian), 2.0);
double stateFromStartDiff2 = Dot(stateFromStartDiff, segmentDiff);
double t = stateFromStartDiff2 / segmentDiff2;
t = std::clamp(t, 0.0, 1.0);
// if closest state is edge state then skip it, because we know it has no collisions
if (t == 0.0 or t == 1.0) {
return std::nullopt;
}
// Closest point of the segment
model::ArmState closestPoint = segmentStart + segmentDiff * t;
double distance = (state – closestPoint).Norm(model::NormType::Euclidian);
if (distance > maxDistToSegment) {
return std::nullopt;
}
return closestPoint;
}
```

Код 2. Функція знаходження найближчого стану від стану, що містить колізії до сегменту / Function for finding the nearest state from a collision state to a segment

```
   bool TrajectoryBetweenStatesColliding(
   const model::RobotState& from,
   const model::RobotState& to,
   collision::CollisionEngine& collisionEngine,
   const PathFinderOptions& options) noexcept {
   // we retrieve all states that was checked during planning and optimizing and has collisions
   auto knownCollidingStates = collisionEngine.KnownCollidingStates (from.RobotBaseInWorld());
   for (auto& state: knownCollidingStates) {
   // options.CollisionCheckStep – path discretisation step for collision checking
   auto threshold = 10.0 * options.CollisionCheckStep; // try different threshold values
   auto closestPoint = ClosestArmStateOnSegment(
   from.ArmState(), to.ArmState(), state.ArmState(), threshold);
   if (not closestPoint.has_value()) {
   continue;
   }
   // check if closest point collides
   auto closestState = state;
   closestState.ArmState(closestPoint.value());
   bool collides = collisionEngine.IsColliding(closestState);
   if (collides) {
   return true;
   }
   }
   auto robotStates = arm::TraceBetweenStates(from, to, {options.TracingType, options.Colli-
sionCheckStep});
   for (int i = 1; i <= (int) robotStates.size() – 2; i++) {
   if (collisionEngine.IsColliding(robotStates[i])) {
   return true;
   }
   }
   return false;
   }
```

Код 3. Функція перевірки наявності колізій між двома станами / Function for checking the presence of collisions between two states

Під час проведення тестувань будуть змінюватись параметри порогу відстані, на якій перевіряють найближчу на відрізку точку, і отримані результати буде порівняно з простою перевіркою відрізку на колізії дискретизованого шляху переміщення роборуки між двома станами. Це дасть змогу оцінити ефективність запропонованого підходу та знайти оптимальні параметри для його застосування.

До тестового набору входять дані, що містять 45 пар точок, для яких спершу планується траєкторія без колізій, а потім виконується код оптимізації траєкторії. Зображення тестового робочого простору можна побачити на рис. **2.**

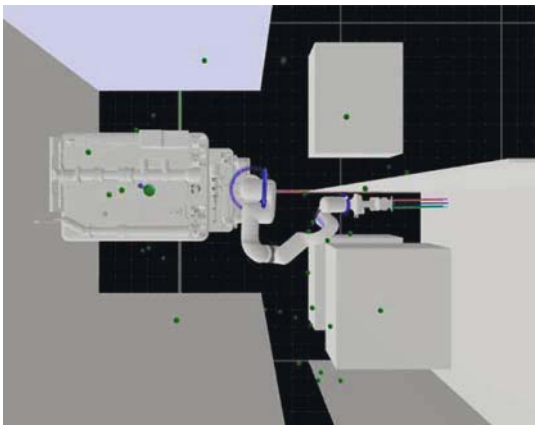

Рис. 2. Робот і робоче середовище / Robot and working environment

Відповідно до проведених експериментів, використання підходу дозволило скоротити кількість перевірок станів на колізії під час оптимізації траєкторії та зменшити тривалість виконання алгоритму оптимізації. При цьому довжина оптимізованої траєкторії не змінилася.

Результати базових експериментів з простою перевіркою на колізії дискретизованого шляху переміщення роборуки між двома станами подано в **Табл. 1.** У табл. 1-4 довжиною траєкторії слугують кути повороту суглобів роботизованої руки.

Табл. 1. Результати без врахування відомих колізій / Results without considering known collisions

Кількість перевірок на колізію, од.	Довжина траєкторії, рад.	Тривалість виконання, с
475	10,05	0,11
861	13,29	0,20
411	5,63	0,09
1165	13,38	0,30
825	13,62	0,20
3828	17,86	1,01
904	12,07	0,19
1763	16,47	0,40
2456	13,26	1,58
925	13,41	0,72
950	13,82	0,66
1657	14,28	0,50
1665	19,13	0,29
1398	15,60	0,22
533	10,97	0,08
2675	20,38	0,43
711	13,31	0,11
928	14,85	0,14
1150	12,94	0,19
272	7,73	0,04
1926	16,83	0,31
984	10,82	0,16
6642	26,34	1,06
2739	14,75	0,44
1901	12,48	0,32
1110	14,66	0,16
2632	17,12	0,41
1476	15,55	0,22
1632	16,18	0,25
3621	16,88	0,57
1517	13,46	0,23
6204	16,26	0,89
1299	15,53	0,18

3265	16,44	0,45
3530	18,27	0,50
3676	22,96	0,55
2040	14,50	0,29
3573	23,06	0,52
2370	15,52	0,33
229	3,93	0,03
2261	14,00	0,32
272	5,19	0,03
982	10,87	0,13
238	6,05	0,03
992	11,05	0,14
Σ = 82663	Σ = 640,96	Σ = 16,20

Під час тестування використовували крок дискретизації траєкторії 0,035 радіан. У разі використання порогу відстані на рівні п'ять кроків дискретизації (0,175 радіан) загальна кількість перевірок на колізії на всіх тестових прикладах знизилася від 82663 до 62056 шт., тобто на 24,93 %, при цьому загальна тривалість виконання алгоритму скоротилась з 16,20 до 10,97 с, тобто на 32,29 %. Результати цього експерименту подано в **Табл. 2.**

Табл. 2. Результати з порогом відстані п'ять кроків дискретизації / Results with a distance threshold of 5 discretization steps

Кількість перевірок на колізію, од.	Довжина траєкторії, рад.	Тривалість виконання, с
371	10,05	0,06
743	13,29	0,11
411	5,63	0,06
812	13,38	0,13
566	13,62	0,08
2782	17,86	0,46
576	12,07	0,08
1206	16,47	0,16
1591	13,26	0,21
799	13,41	0,19
771	13,82	0,12
1200	14,28	0,23
1435	19,13	0,31
929	15,60	0,18
445	10,97	0,07
1939	20,38	0,33
565	13,31	0,11
529	14,85	0,08
878	12,94	0,18
237	7,73	0,04
1407	16,83	0,28
811	10,82	0,13
5107	26,34	1,04
2373	14,75	0,46
1626	12,48	0,27
1077	14,66	0,15
1787	17,12	0,29
1143	15,55	0,18
1037	16,18	0,17
2779	16,88	0,47
1549	13,46	0,30
4311	16,26	0,78
1274	15,53	0,22
2274	16,44	0,41
2685	18,27	0,49
2759	22,96	0,55
1452	14,50	0,25
2461	23,06	0,47
1193	15,52	0,18
256	3,93	0,04
1644	14,00	0,28
262	5,19	0,04
921	10,87	0,14
213	6,05	0,03
870	11,05	0,14
Σ = 62056	Σ = 640,96	Σ = 10,97

У разі використання порогу відстані на рівні десяти кроків дискретизації (0,35 рад.) загальна кількість перевірок на колізії на всіх тестових прикладах знизилася від 82663 до 53139 шт., тобто на 35,72 %, при цьому загальна тривалість виконання алгоритму скоротилась з 16,20 до 9,10 с, тобто на 43,83 %. Детальні результати цього експерименту подано в **Табл. 3.**

Табл. 3. Результати з порогом відстані десять кроків дискретизації / Results with a distance threshold of 10 discretization steps

Кількість перевірок на колізію, од.	Довжина траєкторії, рад.	Тривалість виконання, с
362	10,05	0,05
986	13,29	0,13
411	5,63	0,05
702	13,38	0,10
602	13,62	0,09
2504	17,86	0,40
614	12,07	0,08
1120	16,47	0,15
1339	13,26	0,17
691	13,41	0,15
745	13,82	0,11
1247	14,28	0,23
1279	19,13	0,26
749	15,60	0,14
474	10,97	0,07
1921	20,38	0,36
671	13,31	0,10
554	14,85	0,08
645	12,94	0,12
275	7,73	0,04
1123	16,83	0,22
580	10,82	0,09
3323	26,34	0,67
1613	14,75	0,30
1670	12,48	0,27
1271	14,66	0,18
1510	17,12	0,24
861	15,55	0,13
1108	16,18	0,17
2273	16,88	0,37
1499	13,46	0,26
2707	16,26	0,50
1438	15,53	0,22
1818	16,44	0,31
1981	18,27	0,36
2242	22,96	0,43
1057	14,50	0,16
1492	23,06	0,29
1215	15,52	0,18
337	3,93	0,04
1571	14,00	0,26
361	5,19	0,05
1010	10,87	0,15
221	6,05	0,02
967	11,05	0,14
Σ = 53139	Σ = 640,96	Σ = 9,10

У разі використання порогу відстані на рівні п'ятнадцяти кроків дискретизації (0,525 рад.) загальна

кількість перевірок на колізії на всіх тестових прикладах знизилася від 82663 до 61825 шт., тобто на 25,21 %, при цьому загальна тривалість виконання алгоритму скоротилась з 16,20 до 10,09 с, тобто на 37,72 %. Результати цього експерименту подано в табл. 4.

Табл. 4. Результати з порогом відстані п'ятнадцять кроків дискретизації / Results with a distance threshold of 15 discretization steps

Кількість перевірок на колізію, од.	Довжина траєкторії, рад.	Тривалість виконання, с
342	10,05	0,05
1122	13,29	0,15
411	5,63	0,05
696	13,38	0,09
704	13,62	0,10
2119	17,86	0,33
638	12,07	0,08
1177	16,47	0,15
1396	13,26	0,17
632	13,41	0,13
799	13,82	0,12
1534	14,39	0,26
1421	19,13	0,27
1020	15,60	0,17
460	10,97	0,06
2273	20,38	0,36
1027	13,31	0,15
521	14,85	0,07
683	12,94	0,12
291	7,73	0,04
1191	16,83	0,22
733	10,82	0,11
3128	26,34	0,60
2079	16,26	0,36
1872	12,48	0,30
1558	14,66	0,22
2149	17,12	0,32
928	15,55	0,13
1322	16,18	0,21
2892	16,88	0,45
2040	13,46	0,33
3677	16,2	0,63
2159	15,53	0,33
1848	16,44	0,30
2239	18,27	0,43
2232	23,40	0,42
1138	14,50	0,17
1777	23,06	0,32
1627	15,79	0,24
516	3,93	0,07
1964	14,00	0,31
468	5,19	0,06
1556	10,87	0,22
228	6,05	0,03
1238	11,05	0,18
Σ = 61825	Σ = 640,96	Σ = 10,09

З результатів тестування можна зробити висновок, що після певного рівня підвищення порогу відстані починає погіршуватись ефективність підходу. Це зумовлено тим, що збільшується кількість перевірок на колізію для станів, що є проекціями станів, віддалених від відрізку, і як результат спроектовані стани часто не містять колізій і їх перевірка є надлишковою.

Отже, у цьому дослідженні продемонстровано ефективність методу використання попередньо відомих даних про колізії для пришвидшення оптимізації траєкторії руху роботизованої руки. Отримані результати свідчать про значне скорочення кількості перевірок на колізії та часу виконання алгоритму за збереження якості оптимізованої траєкторії.

Обговорення результатів дослідження. Запропонований підхід може бути використано разом з іншими методами, які активно розвиваються. У роботі [12] запропоновано покращений алгоритм глобального планування шляху переміщення роборуки S-RRT, заснований на ідеї жадібного алгоритму та B-сплайнової кривої. Порівняно з алгоритмом RRT покращений алгоритм S-RRT дає змогу скоротити довжину шляху переміщення роборуки, підвищити ефективність пошуку та ефективно зменшити надлишкові точки шляху переміщення роборуки, а згенеровані шляхи є набагато плавнішими. Зокрема, автори показали, що використання алгоритму S-RRT, порівняно з RRT, дає змогу зменшити довжину шляху переміщення роборуки з 40-45 см до 29-31 см, за практично незмінного часу виконання алгоритму в межах 1-4 с. Окрім цього, у дослідженні, проведеному з використанням генетичних алгоритмів для оптимізації траєкторій руху, було встановлено, що такі методи можуть значно скоротити енергоспоживання роботизованих систем [2]. Хоча це дослідження зосереджено на скороченні тривалості виконання обчислень, а не енергоспоживання, використання попередньо відомих даних про колізії може також сприяти зниженню енерговитрат, оскільки зменшуються обчислювальні навантаження на систему.

Інші дослідники також відзначають важливість використання ефективних алгоритмів планування траєкторій руху у реальних умовах. Зокрема, в роботі [13] досліджено три основні методи планування руху на основі навчання: планування руху на основі глибокого навчання, навчання з підкріпленням і навчання шляхом демонстрації. Так, автори дослідження з використанням динамічного програмування для промислових роботів показали, що алгоритми, які можуть адаптуватися до змін у середовищі, є ключовими для успішної інтеграції роботів у виробничі процеси. Запропонований у цій роботі метод, заснований на інтеграції даних про колізії, відповідає цим вимогам, оскільки дає змогу швидко адаптувати траєкторію роботизованої руки в динамічних умовах, проте не потребує значних обчислювальних ресурсів, характерних для методів машинного навчання, зокрема глибокого навчання чи навчання з підкріпленням.

Методи планування траєкторій руху викликають значну зацікавленість наукової спільноти завдяки їх важливості для покращення ефективності та безпеки мобільних роботів. Дослідження демонструють різноманітні підходи до вирішення цієї проблеми, зокрема огляд методологій планування траєкторій руху роботі [11], адаптивні алгоритми оптимізації мурашиних колоній [16], нові стратегії оптимізації шляхів для мобільних роботів [3] та вдосконалені евристичні механізми [8]. У цих роботах наголошено на важливості та актуальності досліджень у цій галузі для досягнення високої точності та швидкості планування траєкторій руху.

Так, у роботі [11] проведено докладний огляд методологій планування та оптимізації шляху переміщення мобільних роботів, здійснено огляд класичних і сучасних методів планування та оптимізації шляху пе-

реміщення роборуки, в т.ч. й штучні потенційні поля, алгоритм A*, алгоритм Дейкстри, генетичний алгоритм, ройовий інтелект і методи на основі машинного навчання. Автори показали, що слабкими аспектами класичних алгоритмів є те, що вони не практичні в реальних сценаріях через складність навколишнього середовища та обмеження давачів; точні алгоритми, можуть бути обчислювально дорогими і трудомісткими, особливо у великомасштабних середовищах; евристичні алгоритми можуть не знайти оптимальний розв'язок через застрягання в локальних мінімумах; ймовірнісні алгоритми можуть бути обчислювально дорогими та потребувати значної кількості прикладів для знаходження прийнятного розв'язку; гібридні алгоритми можуть бути складними для реалізації та потребувати значної кількості параметрів; тоді як методи оптимізації можуть потребувати значного обсягу навчальних даних та бути чутливими до їх якості. На відміну від зазначених алгоритмів, запропонований у цій роботі підхід на основі інтеграції даних про колізії дає змогу оптимізувати довжину траєкторії руху роботизованої руки, уникаючи більшості із зазначених недоліків, зокрема, він є обчислювально простішим та не потребує значної кількості навчальних даних чи параметрів.

Автори [16] запропонували удосконалену версію алгоритму мурашиної колонії для задачі планування шляху переміщення мобільних роботів, який охоплює чотири нових механізми та потребує оптимізації параметрів алгоритму для встановлення їх оптимальної комбінації. Зокрема, в евристичну функцію вбудовано інформацію про початкову точку, цільову точку, поточну точку та час повороту запланованого шляху переміщення роборуки. Покращення евристичної функції дало змогу зручно регулювати швидкість пошуку від поточної до цільової точки та збалансувати інтенсивність евристичної інформації. Адаптивне псевдовипадкове правило передачі було введено для підвищення ефективності пошуку та збільшення різноманітності рою. Окрім цього, в роботі [16] представлено механізм самоадаптивного налаштування для покращення правила ймовірності переходу з одного стану в інший. Цей підхід належить до класу екристичних алгоритмів та зберігає всі недоліки класу, зазначені в попередньому абзаці. Інший підхід до вдосконалення алгоритму мурашиної колонії представлено в роботі [8]. Щоб подолати недоліки класичного алгоритму мурашиної колонії, автори навели чотири нові механізми, які містять адаптивне налаштування концентрації феромонів, евристичний механізм із спрямованим судженням, покращену стратегію псевдовипадкового перенесення та динамічне регулювання швидкості випаровування феромонів. Автори [8] порівняли запропонований підхід з 15 наявними підходами для вирішення планування шляху переміщення роборуки, в т.ч. й дев'ять варіантів алгоритму мурашиної колонії та шість найбільш поширених алгоритмів детермінованого пошуку. Експериментальні результати показали, що відносний відсоток покращення, запропонованого в роботі [8], удосконалення алгоритму мурашиної колонії з погляду часу повороту траєкторії становить 33,33, 83,33, 35,29, 38,46 і 38,46 %% відповідно. На відміну від підходів у роботах [8, 16], запропонований у цій публікації метод не потребує оптимізації параметрів, не є трудомістким, та не потерпає від проблеми локальних екстремумів.

Автори [3] розглядають покращення мета-евристичного алгоритму оптимізації Whale Optimization Algorithm – NWOA. Порівнюючи NWOA з оригінальною версією, поєданням оригінальної версії з генетичним алгоритмом та розширеним глобальним удосконаленням алгоритму (EGE-WOA), експериментальні результати роботи [3] показують, що середнє значення придатності, отримане NWOA, зменшується на 32,0, 23,2 і 8,3 %%, відповідно, а середнє значення часу планування та довжини шляху переміщення роборуки, отримане NWOA, також є найменшим серед розглянутих алгоритмів. Проте, цей алгоритм також має класичні недоліки, притаманні цьому класу методів, зокрема повільну збіжність, низьку точність оптимізації та чутливість до локальних мінімумів роботи [14].

Отже, за результатами виконаної роботи можна сформулювати такі наукову новизну та практичну значущість результатів дослідження.

Наукова новизна отриманих результатів дослідження – розроблено метод інтеграції даних про колізії для пришвидшення процесу оптимізації траєкторії руху роботизованої руки, що зменшило кількість перевірок на колізії прямих відрізків під час виконання відповідних маніпуляцій.

Практична значущість результатів дослідження – результати дослідження може бути використано для підвищення ефективності застосування промислових роботизованої руки шляхом зменшення тривалості обчислення оптимальної траєкторії роботизованої руки, зменшення довжини такої траєкторії, що дає змогу підвищити швидкодію та зменшити енергоспоживання промислових роботехнічних систем, що може привести до зменшення часу на виконання операцій та підвищення продуктивності виробництва.

Висновки / Conclusions

Розроблено метод інтеграції даних про колізії з навколишнім середовищем, отриманих під час планування траєкторії руху руки для пришвидшення роботи алгоритму оптимізації траєкторії її руху. За результатами проведеного дослідження можна зробити такі основні висновки.

1. Встановлено, що застосування цього методу дає змогу зменшити тривалість виконання алгоритму до 43 % на рівні десяти кроків дискретизації, без погіршення якості оптимізованої траєкторії за рахунок уникнення додаткових перевірок на колізії на етапі оптимізації для станів, що належать до регіонів, для яких перевірка на колізії відбувалась під час планування руху роботизованої руки.
2. Проведено експериментальні дослідження, які показали ефективність підходу та зменшили тривалість виконання алгоритму оптимізації траєкторії. З'ясовано, що використання попередньо відомих даних про колізії є перспективним напрямом для підвищення ефективності роботи роботизованої руки в промислових умовах.
3. Встановлено, що довжина оптимізованої траєкторії руху залишається незмінною у разі використання запропонованого методу за умови зменшення тривалості виконання обчислень до 43 %, що дає змогу підвищити швидкодію роботизованих систем без втрати якості їхньої роботи, а також зменшити їх енергоспоживання на виконання однієї операції.

Отже, результати дослідження підтверджують ефективність запропонованого підходу для оптимізації траєкторій руху роботизованої руки, що може бути ко-

рисним для подальшого розвитку технологій автоматизації у промисловості.

Подяка: автори висловлюють щиру подяку компанії Somatic LTD Holdings за надану підтримку та можливість використання їхнього програмного забезпечення під час проведення тестування.

References

1. Bohlin, R., & Kavraki, L. E. (2000). Path planning using lazy PRM. In Proceedings 2000 ICRA. Millennium conference. *IEEE international conference on robotics and automation. Symposia proceedings* (Cat. No. 00CH37065), 1, 521–528. https://doi.org/10.1109/ROBOT.2000.844107
2. Cao, Y. (2023). GPU-Enabled Genetic Algorithm Optimization and Path Planning of Robotic Arm for Minimizing Energy Consumption. *University of South Carolina*. Scholar Commons. (Master's thesis). URL: https://scholarcommons.sc.edu/cgi/viewcontent.cgi? article=8312&context=etd
3. Dai, Y., Yu, J., Zhang, C., Zhan, B., & Zheng, X. (2023). A novel whale optimization algorithm of path planning strategy for mobile robots. *Applied Intelligence*, 53(9), 10843–10857. https://doi.org/10.1007/s10489-022-04030-0
4. Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, *30*(7), 846–894. https://doi.org/10.1177/0278364911406761
5. Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580. https://doi.org/10.1109/70.508439
6. Kim, D., Kwon, Y., & Yoon, S.-E. (2018). Adaptive Lazy Collision Checking for Optimal Sampling-based Motion Planning. 2018 15th *International Conference on Ubiquitous Robots* (UR), Honolulu, HI, USA, 320–327. https://doi.org/10.1109/URAI.2018.8442203
7. LaValle, Steven, M. (1998). Rapidly-exploring Random Trees: A New Tool for Path Planning, 1–4. URL: https://lavalle.pl/papers/Lav98 c.pdf
8. Liu, C., Wu, L., Xiao, W., Li, G., Xu, D., Guo, J., & Li, W. (2023). An improved heuristic mechanism ant colony optimization algorithm for solving path planning. *Knowledge-Based Systems*, 271. https://doi.org/10.1016/j.knosys.2023.110540
9. Mashayekhi, R., Idris, M. Y. I., Anisi, M. H., Ahmedy, I., & Ali, I. (2020). Informed RRT*-connect: An asymptotically optimal single-query path planning method, 8, 19842–19852. https://doi.org/10.1109/ACCESS.2020.2969316
10. Ravankar, A., Ravankar, A. A., Kobayashi, Y., Hoshino, Y., & Peng, C.-C. (2018). Path Smoothing Techniques in Robot Navigation: State-of-the-Art, Current and Future Challenges. *Sensors*, 18. https://doi.org/10.3390/s18093170
11. Sahoo, S. K., & Choudhury, B. B. (2023). A review of methodologies for path planning and optimization of mobile robots. *Journal of process management and new technologies*, 11(1-2), 122–140. https://doi.org/10.5937/jouproman2301122S
12. Song, Q., Li, S., Yang, J., Bai, Q., Hu, J., Zhang, X., & Zhang, A. (2021). Intelligent Optimization Algorithm-Based Path Planning for a Mobile Robot. *Computational intelligence and neuroscience*. https://doi.org/10.1155/2021/8025730
13. Tamizi, M. G., Yaghoubi, M., & Najjaran, H. (2023). A review of recent trend in motion planning of industrial robots. *International Journal of Intelligent Robotics and Applications*, 7(2), 253–274. https://doi.org/10.1007/s41315-023-00274-2
14. Tang, J., & Wang, L. (2024). A whale optimization algorithm based on atom-like structure differential evolution for solving engineering design problems. *Scientific Reports*, 14, 795. https://doi.org/10.1038/s41598-023-51135-8
15. Wang Daniel. (2024). The difference between UFACTORY xArm5, UFACTORY xArm6 and UFACTORY xArm7. UFACTORY Help Center. URL: http://help.ufactory.cc/en/articles/4491842-the-difference-between-ufactory-xarm5-ufactory-xarm6-and-ufactory-xarm7
16. Wu, L., Huang, X., Cui, J., Liu, C., & Xiao, W. (2023). Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. Expert Systems with Applications, 215. https://doi.org/10.1016/j.eswa.2022.119410
17. Xie, F., Chen, L., Li, Z., & Tang, K. (2020). Path Smoothing and Feed Rate Planning for Robotic Curved Layer Additive Manufacturing. *Robotics and Computer-Integrated Manufacturing*, 65. https://doi.org/10.1016/j.rcim.2020.101967

A. Ya. Medvid, V. S. Yakovyna

Lviv Polytechnic National University, Lviv, Ukraine

INTEGRATION OF COLLISION DATA FOR ACCELERATING TRAJECTORY OPTIMIZATION OF ROBOTIC ARMS

The article presents a novel approach to improve the optimization of robotic arm movement trajectory by integrating collision data obtained during path planning. Path planning of robotic systems is one of the key tasks in modern robotics, especially for robotic arms used in industry to perform complex manipulations. Many studies are dedicated to developing algorithms for movement trajectory planning that ensure collision avoidance and minimize computational time and resource consumption. Despite numerous advancements in movement trajectory planning and optimization methods, there is a need for additional approaches that can improve the efficiency of these processes. The main goal of the research was to develop a method to accelerate the movement trajectory optimization of a robotic arm by integrating collision data. The main research tasks were defined as follows: to develop an algorithm that uses pre-known collision data to accelerate collision checks of straight segments; to conduct experimental studies on the proposed method effectiveness; and to evaluate the impact of the proposed approach on the overall speed and quality of movement trajectory length optimization. During the testing, parameters such as the number of collision check function calls, the length of the optimized movement trajectory, and the algorithm execution time were controlled. The input parameter of the tests was the collision check threshold based on the distance to the given segment. Experimental studies yielded results confirming the approach effectiveness. As a result of using a distance threshold of 5 discretization steps (0.175 radians), the total number of collision checks decreased from 82663 to 62056, which is 24.93 %, while the total execution time decreased from 16.2 s to 10.97 s, which is 32.29 %. Using a distance threshold of 10 discretization steps (0.35 radians), the number of checks decreased to 53139, which is 35.72 %, and the execution time decreased to 9.1 s, which is 43.83 %. The proposed approach can be integrated with other movement trajectory planning and optimization methods, enhancing its versatility and applicability in various robotic systems. The research results confirm the potential of this approach for optimizing the movement trajectory length of robotic systems, which can be beneficial for the further development of automation technologies in the industry.

Keywords: planning the trajectory of movement; robotic arms; path optimization; time minimization.