POLITECHNIKA WARSZAWSKA

Wydział Mechatroniki

Praca przejściowa

Ireneusz Szulc

Planowanie bezkolizyjnych tras dla zespołu robotów mobilnych

Opiekun pracy:
prof. dr hab. Barbara Siemiątkowska

Warszawa, 2018

Spis treści

Rozdział 1

TODO

Karta tematu:
Temat pracy: Planowanie bezkolizyjnych tras dla zespołu robotów mobilnych
Temat pracy (w jęz. ang.): Path planning for a group of mobile robots

Zakres pracy:

- Projekt algorytmu wyznaczania trajektorii dla pojedynczego robota
- Algorytm detekcji i zapobiegania kolizjom między robotami
- Implementacja oprogramowania symulacyjnego
- Przeprowadzenie testów symulacyjnych

Podstawowe wymagania:

- Aplikacja powinna umożliwiać symulację ruchu robotów oraz definiowanie położenia przeszkód przez użytkownika.
- Planowanie tras dotyczy robotów holonomicznych.

Rozdział 2

Konspekt pracy

- Wstęp teoretyczny:
  - Cooperative Pathfinding
  - algorytm A* - szczegółowo
  - przegląd metod planowania tras dla wielu robotów
  - artykuł o Cooperative Pathfinding, time-space A*
  - artykuł o wyznaczaniu priorytetów i metodach planowania tras (prezentacja): Path Coordination, time-space A*
  - metoda ładunków - problem minimów lokalnych
  - replanowanie po wykruciu kolizcji (algorytm D*)
  - algorytmy WHCA* i IADPP
  - Reciprocal Collision Avoidance
  - metody przydziału priorytetów - zwiększanie i przeliczanie
  - metody zcentralizowane vs rozproszone (porównanie)
  - time-space A*, heurystyki, Reservation Table
- generowanie mapy - labiryntu do testów: własny algorytm, teoria grafów, własności mapy
- metoda przydziału / zmiany priorytetów
- obszerne testy, porównanie wyników metod przy tych samych warunkacj początkowych
- zastosowanie: ciasne korytarze, częsty problem kolizji, szpitale, transport dokumentów, paczek

- time-space A* - pseudokod, schemat blokowy, własne heurystyki, modyfikacje
- ograniczenia - nałożone uproszczenia: ruch skośny, czas dyskretny, brak czasu na obrót
- Implementacja aplikacji - stack technologiczny: Java 8, Java FX, Spring, Spring Boot, testy jednostkowe jUnit, git, IntelliJ, Maven, Linux; schemat klas aplikacji

Rozdział 3

Wstęp teoretyczny

TODO Cooperative Pathfinding David Silver Cooperative Pathfinding is a multi-agent path planning problem where agents must find non-colliding routes to separate destinations, given full information about the routes of other agents. This paper presents three new algorithms for efficiently solving this problem, suitable for use in Real-Time Strategy games and other real-time environments. The algorithms are decoupled approaches that break down the problem into a series of single-agent searches. Cooperative A* (CA*) searches space-time for a non-colliding route. Hierarchical Cooperative A* (HCA*) uses an abstract heuristic to boost performance. Finally, Windowed Hierarchical Cooperative A* (WHCA*) limits the space-time search depth to a dynamic window, spreading computation over the duration of the route. The algorithms are applied to a series of challenging, maze-like environments, and compared to A* with Local Repair (the current video-games industry standard). The results show that the new algorithms, especially WHCA*, are robust and efficient solutions to the Cooperative Pathfinding problem, finding more successful routes and following better paths than Local Repair A*.

Kooperacyjne znajdowanie tras (ang. Cooperative Pathfinding) jest problemem planowania w układzie wieloagentowym, gdzie agenci muszą znaleźć bezkolizyjne drogi do swoich, osobnych celów. Planowanie to odbywa się mając pełną informację o środowisku oraz trasach pozostałych agentów.

Przedmiotem niniejszej pracy jest przegląd metod rozwiązujacych zagadnienie planowania bezkolizyjnych tras dla wielu robotów. Stanowi to wstęp do zaprojektowania algorytmu i implementacji oprogramowania pozwalającego na symulację jego działania w praktyce (tzn. na przykładzie *TODO*).

TODO nie ma algorytmów do planowania tras w środowiskach ciasnych korytarzy (duża liczba przeszkód, częsty problem blokowania w wąskich korytarzach, zakleszczenie) *TODO* Problem do rozwiązania: Dane: mapa otoczenia i przestrzeń konfiguracyjna, określone położenie początkowe i cel dla zespołu robotów; Zadanie: Wyznaczenie możliwie najkrótszej bezkolizyjnej

trasy dla wszystkich robotów. *TODO* nie otwarte środowisko, ale zamknięte, z dużą liczbą przeszkód

- Koordynacja ruchu robotów jest jednym z fundamentalnych problemów dla systemów wielu robotów.
- Popularne podejścia omijające planowanie w wysoko wymiarowej zbiorowej przestrzeni konfiguracyjnej to techniki rozproszone i priorytetowane.
- Pomimo, że te metody są bardzo efektywne, mają dwie główne wady:
  1. nie są zupełne - czasami nie udaje się znaleźć rozwiązania, nawet gdy istnieje.
  2. Wynikowe rozwiązania są często nieoptymalne.
- Ponadto nie mówią, jak przypisywać priorytety do pojedynczych robotów.
- W tym artykule przedstawiono metodę do optymalizowania układu priorytetów dla rozproszonych i priorytetowanych technik planowania.
- Proponowana metoda wykonuje randomizowane przeszukiwanie z techniką hill-climbing do znalezienia rozwiązania i do skrócenia całkowitej długości ścieżek.
- Technika została zaimplemenotwana i przetestowana na prawdziwych robotach oraz w rozległych testach symulacyjnych.
- Wyniki eksperyentu pokazały, że metoda potrafi znacząco zmniejszyć liczbę niepowodzeń i znacznie zmniejszyć całkowitą długość tras dla różnych priorytetowanych i rozproszoncyh metod planowania dróg, nawet dla dużych zespołów robotów.

3.1 Przestrzeń konfiguracyjna

Przestrzeń konfiguracyjna to formalna, matematyczna przestrzeń będąca zbiorem możliwych stanów danego układu fizycznego. W zależności od rodzaju i liczby wyróżnionych parametrów stanu przestrzenie konfiguracyjne mogą mieć wiele wymiarów.

3.2 Metoda hill-climbing

Metoda hill-climbing jest rodzajem matematycznej optymalizacji, lokalną metodą przeszukiwania. Jest to iteracyjny algorytm, który zaczyna w wybranym rozwiązaniu problemu, następnie

próbuje znaleźć lepsze rozwiązanie poprzez przyrostowe zmiany pojedynczych elementów rozwiązania. Jeśli zmiana przynosi lepsze rozwiązanie, jest wprowadzana do nowego rozwiązania. Kroki algorytmu powtarzane są dopóki żadne "udoskonalenia" nie mogą już być znalezione.

3.3 Metody planowania tras

Spośród metod wykorzystywanych do planowania tras dla wielu robotów można wyróżnić dwie grupy:

- zcentralizowane - drogi wyznaczane są dla wszystkich agentów na raz (jednocześnie). Metody te potrafią znaleźć optymalne rozwiązanie, ale często mają bardzo dużą złożoność obliczeniową, dlatego wykorzystywane są heurystyki
- rozproszone (ang. decoupled) - dla każdego robota droga wyznaczana jest osobno, w określonej kolejności, następnie rozwiązywane są konflikty (kolizje dróg), może nie znaleźć istniejącego rozwiązania. Metoda najczęściej wiąże się z koniecznością przydzielenia robotowi priorytetu, co stanowi istotny problem, gdyż od wyboru priorytetów może zależeć zupełność algorytmu.

3.3.1 Metody zcentralizowane

Zalety:

- Planowanie w zbiorowej przestrzeni konfiguracyjnej
- Wyznaczenie **optymalnego** rozwiązania
- W praktyce: Heurystyka radzi sobie z ogromną złożonością przestrzeni konfiguracyjnej

Podejścia (bez gwarancji optymalności):

- Potential field techniques - Metoda potencjałowa
- Roadmap methods

3.3.2 Podejście "Decoupled"

Algorytm (niezupełny):

1. Wyznaczenie optymalnej ścieżki dla każdego robota **niezależnie**
2. Przydział priorytetów (opcjonalne)

3. Próba rozwiązania możliwych konfliktów między ścieżkami

Podejścia:

- Path coordination
- Planning in the configuration time-space:
  - V-Graph algorithm
  - Potential fields
  - A*

Path coordination

Idea:

- Utrzymanie robotów na ich indywidualnych, optymalnych ścieżkach
- Pozwolenie na zatrzymanie się, ruch naprzód, a nawet cofanie się, ale tylko **wzdłuż trajektorii** w celu uniknięcia kolizji

W praktyce:

- Wymagany wariant z ustaleniem priorytetów
- Złożoność $O(n \cdot m \cdot log(m))$, m - maksymalna liczba stanów podczas planowania

Konieczność wyboru priorytetów

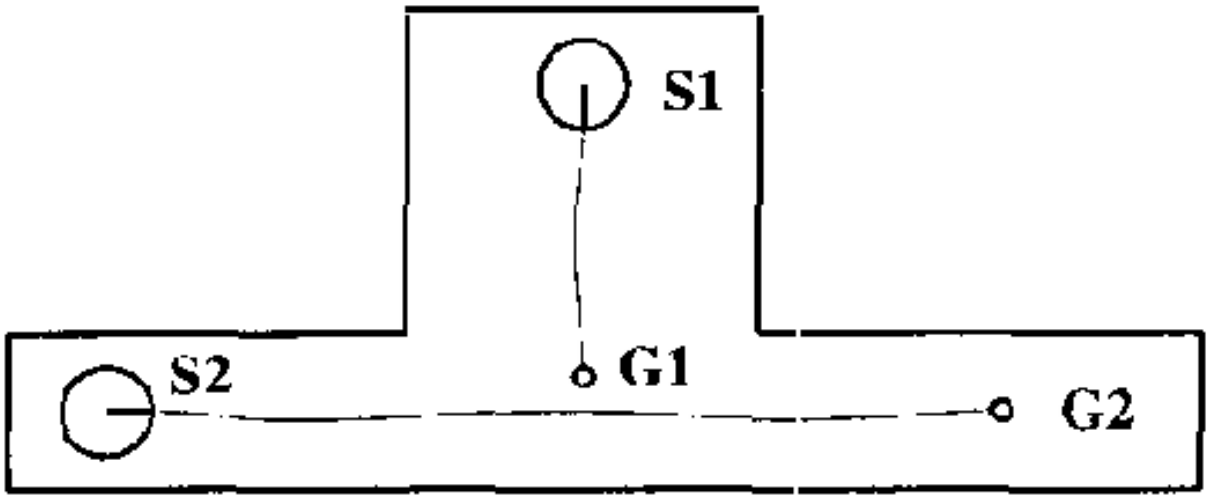
Zastosowanie A* w planowaniu dróg dla wielu robotów

Algorytm:

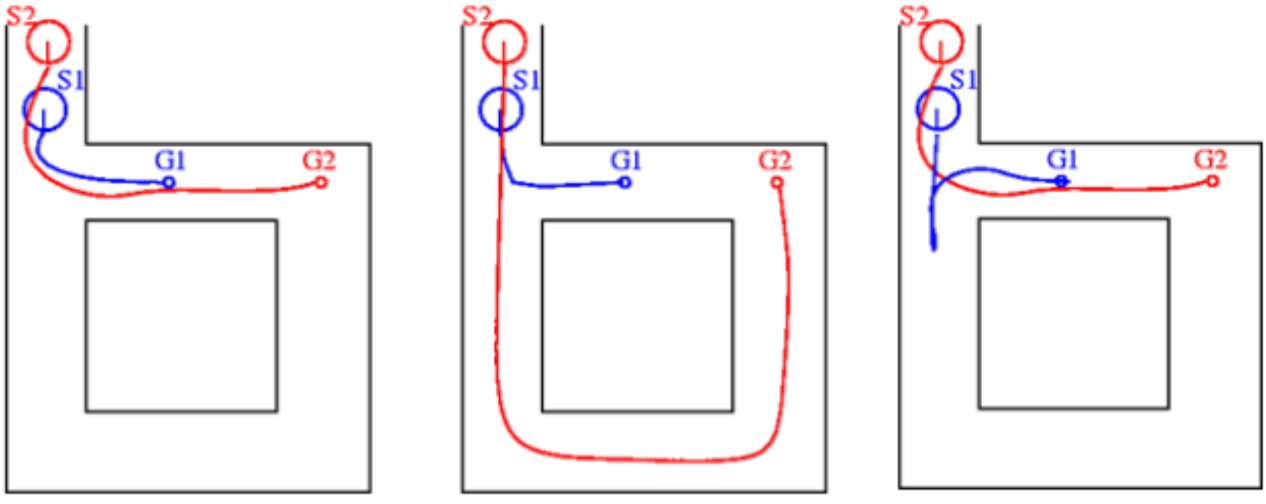
- Przypisanie priorytetów do poszczególnych robotów
- Wykonywanie kroków A* z rozpatrywaniem czasu i przestrzeni - podział otoczenia na siatkę pól, zapisywanie prawdopodobieństwa zajętości pola w danej chwili

Złożoność:

- $O(n \cdot m \cdot log(m))$, m - maksymalna liczba stanów podczas planowania (lista otwartych)



Rysunek 3.1: Sytuacja, w której żadne rozwiązanie nie zostanie znalezione, jeśli robot 1 ma wyższy priorytet niż robot 2



Rysunek 3.2: Niezależne planowanie optymalnych tras dla 2 robotów; suboptymalne rozwiązanie, gdy robot 1 ma wyższy priorytet; rozwiązanie, gdy robot 2 ma wyższy priorytet

Wpływ układu priorytetów na długość tras

3.4 Wybór priorytetów

Elastyczny dobór priorytetów

Obecne techniki pozostawiają dowolny wybór priorytetów lub korzystają z ustalonego z góry stałego układu kolejności robotów.

Cel:

Połączyć ze sobą planowanie tras i przypisanie priorytetów, wykorzystując randomizowane techniki.

Poszukiwanie układu priorytetów mających rozwiązanie

```
FOR tries := 1 TO  maxTries BEGIN
    select random order P
        FOR flips := 1 TO maxFlips BEGIN
        choose random i, j with i < j
         P := swap(i, j, P)
        IF solvable(P)
                return P
    END FOR
END FOR
```

Prosta technika losująca układ priorytetów (kolejności robotów) daje dobre rezultaty, ale często duża liczba iteracji jest konieczna do uzyskania rozwiązania.

3.5 Rezultaty

3.5.1 Ocena metody przez Eksperyment

Testy symulacyjne:

- Zastosowanie naszego algorytmu wyboru priorytetów:
  - A* in the configuration time-space
  - path coordination method
- Użycie 2 różnych środowisk (acykliczne / cykliczne)
- Losowe generowanie punktów startu i celu

Zoptymalizowany algorytm wyznaczania priorytetów

3.6 Podusmowanie

3.6.1 Wnioski

- Zaprezentowano metodę doboru priorytetów dla rozproszonych metod planowania dróg dla grupy robotów mobilnych.
- Zaproponowane podejście to randomizowana metoda, która cyklicznie zamienia kolejność robotów w celu znalezienia sekwencji, dla której można wyznaczyć plan dróg oraz w celu minimalizacji całkowitej długości tras.
- Jest to algorytm, który może być zatrzymany w dowolnym momencie i może zawsze zwrócić obecnie najlepsze rozwiązanie.
- Metoda została zaimplementowana i przetestowana na rzeczywistych robotach i w rozległych testach symulacyjnych dla dwóch różnych metod planowania dróg oraz dla dużej liczby robotów.
- Wyniki eksperyentu pokazały, że metoda potrafi znacząco zmniejszyć liczbę niepowodzeń (gdy żadne rozwiązanie nie zostaje znalezione) i znacznie zmniejszyć całkowitą długość tras.

```
FOR tries := 1 TO  maxTries BEGIN
    select random order P
    IF (tries = 1)
            P* := P
    FOR flips := 1 TO maxFlips BEGIN
        choose random i, j with i < j
        P´ := swap(i, j, P)
        IF moveCosts(P´) <  moveCosts(P)
                P := P´
    END FOR
    IF moveCosts(P) < moveCosts(P*)
        P* := P
END FOR
RETURN P*
```

Rozdział 4

Algorytm A*

TODO

Rozdział 5

Podsumowanie

Bibliografia

[1] Thrun S. Bennewitz M., Burgard W. *Optimizing Schedules for Prioritized Path Planning of Multi-Robot Systems*. 2001.

[2] Roszkowska E. Mówiński K. *Sterowanie hybrydowe ruchem robotów mobilnych w systemach wielorobotycznych*. Postępy Robotyki, 2016.

[3] Siemiątkowska B. *Uniwersalna metoda modelowania zachowań robota mobilnego wykorzystująca architekturę uogólnionych sieci komórkowych*. 2009.

Wykaz skrótów

API	Application Programming Interface
SDK	Software Development Kit

Spis rysunków

Spis tabel