Politechnika
Śląska

PROJEKT INŻYNIERSKI

Budowa mapy otoczenia z wykorzystaniem robota mobilnego

Krzysztof GRĄDEK
Nr albumu: ⟨**300362**⟩

Kierunek: ⟨Automatyka i Robotyka⟩
Specjalność: ⟨Technika Informacjii⟩

PROWADZĄCY PRACĘ
⟨**dr inż. Krzysztof Jaskot**⟩
KATEDRA ⟨**Katedra Automatyki i Robotyki**⟩
Wydział Automatyki, Elektroniki i Informatyki

Gliwice 2024

Tytuł pracy

Budowa mapy otoczenia z wykorzystaniem robota mobilnego

Streszczenie

W ramach projektu wykonano mapowanie otoczenia z wykorzystaniem robota mobilnego stanowiącego połączenie obsługi mapowania, lokalizacji i nawigacji przy pomocy mikrokontrolera Raspberry pi 4 oraz Arduino nano do obsługi silników z enkoderami. W pracy przedstawiono sposób budowy mapy otoczenia z wykorzystaniem robota mobilnego. W pierwszym etapie pracy zrealizowano lokalizację robota w przestrzeni. W kolejnym etapie zrealizowano mapowanie otoczenia. W pracy wykorzystano platformę ROS oraz wizualizację Rviz.

Słowa kluczowe

Mapowanie, robot mobilny, lokalizacja, SLAM, ROS

Thesis title

Construction of an Environment Map Using a Mobile Robot

Abstract

The project involved mapping the environment using a mobile robot that combines mapping, localization, and navigation using a Raspberry pi 4 microcontroller and an Arduino nano to control motors with encoders. The work presents a way to build an environment map using a mobile robot. In the first stage of the work, the robot's location in space was realized. In the next stage, environment mapping was realized. The work used the ROS platform and Rviz visualization.

Key words

Mapping, mobile robot, localization, SLAM, ROS

Spis treści

Rozdział 1

Wstęp

Technologia jednoczesnej lokalizacji i mapowania - SLAM ("Simultaneous Localization and Mapping") polega na wykorzystaniu czujnika laserowego LiDAR do określenia pozycji robota w przestrzeni, a następnie zmapowanie pomieszczenia dla dalszej nawigacji robota.

Niniejsza praca skupia się na zbudowaniu prostego robota mobilnego wykorzystującego LiDAR do mapowania otoczenia. Robot wyposażony jest w mikrokontroler Raspberry Pi 4 stanowiący podstawę operacyjną robota, oraz Arduino Nano do obsługi silników z enkoderami. W szczególności, praca koncentruje się na implementacji rozwiązania umożliwiającego robotowi poruszanie się między zadanymi punktami w przestrzeni na utworzonej mapie.

Do realizacji założonych celów wykorzystano nowoczesne narzędzia z ekosystemu ROS 2 (Robot Operating System 2), w tym:

- Nav2 (Navigation 2) - do planowania, wykonywania ścieżek oraz lokalizacji na utworzonej mapie
- SLAM Toolbox (Simultaneous Localization and Mapping Toolbox) - do jednoczesnej lokalizacji i mapowania
- ROS2 Control (Robot Operating System 2 Control) - do sterowania napędem robota

1.1 Wprowadzenie do tematu

Roboty mobilne znajdują coraz szersze zastosowanie w różnych dziedzinach - od przemysłu przez usługi po zastosowania domowe. Kluczową umiejętnością takich robotów jest możliwość autonomicznej nawigacji w przestrzeni, co wymaga zdolności do tworzenia i wykorzystywania map otoczenia.

1.2 Osadzenie problemu w dziedzinie

Mapowanie i nawigacja robotów mobilnych stanowi fundamentalny problem w robotyce. W ostatnich latach nastąpił znaczący postęp w tej dziedzinie, głównie dzięki rozwojowi:

- Sensorów (w tym LiDAR - Light Detection and Ranging, kamery)
- Algorytmów SLAM (Simultaneous Localization and Mapping)
- Systemów kontroli ruchu
- Szkieletów programistycznych dla robotów (jak ROS - Robot Operating System)

1.3 Cel pracy

Głównym celem pracy jest zaprojektowanie i implementacja systemu mapowania otoczenia z wykorzystaniem robota mobilnego. Cele szczegółowe obejmują:

- Budowę platformy mobilnej
- Implementację systemu sterowania
- Integrację czujników
- Realizację algorytmów SLAM
- Implementację systemu nawigacji

Zakres pracy obejmuje:

- Analizę istniejących rozwiązań w dziedzinie mapowania i nawigacji robotów mobilnych
- Projekt i implementację systemu sterowania robotem
- Integrację komponentów sprzętowych i programowych
- Implementację algorytmów SLAM i nawigacji
- Testy i walidację stworzonego rozwiązania

1.4 Metodyka realizacji

Praca została zrealizowana w następujących etapach:

- Analiza literatury i istniejących rozwiązań
- Projekt i budowa platformy sprzętowej
- Implementacja oprogramowania
- Integracja komponentów
- Testy i optymalizacja

1.5 Struktura pracy

Praca składa się z sześciu rozdziałów. Po niniejszym wstępie, w rozdziale drugim przedstawiono analizę tematu i przegląd literatury. Rozdział trzeci opisuje wymagania projektowe oraz wykorzystane narzędzia. W rozdziale czwartym omówiono specyfikację zewnętrzną systemu, a w piątym - jego implementację. Rozdział szósty zawiera opis przeprowadzonych testów i ich wyniki. Pracę kończy podsumowanie i wnioski.

1.6 Wkład własny autora

W ramach pracy autor samodzielnie:

- Zaprojektował i zbudował platformę mobilną
- Zaimplementował sterowniki urządzeń
- Zintegrował komponenty sprzętowe i programowe
- Zaimplementował i dostosował algorytmy SLAM
- Przeprowadził testy i optymalizację systemu

Rozdział 2

Analiza tematu

2.1 Sformułowanie problemu

Problem jednoczesnej lokalizacji i mapowania (SLAM) jest fundamentalnym zagadnieniem w robotyce mobilnej. Polega on na rozwiązaniu dwóch współzależnych problemów:

- Budowy mapy nieznanego otoczenia
- Określenia pozycji robota w tym otoczeniu

Trudność polega na tym, że do stworzenia dokładnej mapy potrzebna jest precyzyjna lokalizacja robota, a do precyzyjnej lokalizacji potrzebna jest dokładna mapa. Jest to klasyczny problem "kury i jajka".

2.2 Stan aktualny dziedziny

W ostatnich latach nastąpił znaczący postęp w dziedzinie SLAM, głównie dzięki rozwojowi:

- Wydajnych algorytmów optymalizacji
- Dokładniejszych sensorów (LiDAR, kamery RGB-D)
- Mocy obliczeniowej komputerów

Współczesne rozwiązania SLAM można podzielić na kilka głównych kategorii:

- Filtracyjne (np. Extended Kalman Filter SLAM)
- Optymalizacyjne (Graph SLAM)
- Oparte na skanach laserowych
- Wizyjne (Visual SLAM)

2.3 Przegląd istniejących rozwiązań

2.3.1 SLAM Toolbox

SLAM Toolbox to nowoczesne rozwiązanie open-source dla ROS 2, które implementuje algorytm Graph SLAM [**bib:slamtoolbox**]. Jego główne cechy to:

- Tworzenie map 2D w czasie rzeczywistym
- Optymalizacja grafu pozycji robota
- Możliwość serializacji i deserializacji map
- Wsparcie dla map wielowarstwowych

SLAM Toolbox wykorzystuje skany laserowe do budowy mapy i lokalizacji. Algorytm działa w następujących krokach:

1. Akwizycja danych z czujników
2. Dopasowanie skanów metodą Iteracyjny najbliższy punk - ICP (Iterative Closest Point)
3. Budowa grafu pozycji robota
4. Optymalizacja grafu metodą Levenberga-Marquardta

2.3.2 Adaptacyjna lokalizacja Monte Carlo - AMCL (Adaptive Monte Carlo Localization)

AMCL to implementacja probabilistycznej lokalizacji wykorzystująca filtry cząsteczkowe [**bib:amcl**]. Główne cechy AMCL:

- Adaptacyjna liczba cząstek
- Efektywna implementacja filtra cząsteczkowego
- Obsługa map probabilistycznych

Algorytm AMCL działa w następujących krokach:

1. Inicjalizacja chmury cząstek
2. Predykcja ruchu cząstek na podstawie odometrii
3. Aktualizacja wag cząstek na podstawie pomiarów
4. Resampling cząstek

2.4 Wybór rozwiązania

W projekcie zdecydowano się wykorzystać połączenie SLAM Toolbox do lokalizacji w trakcie mapowania oraz zapisu mapy, oraz AMCL do lokalizacji podczas nawigacji z następujących powodów:

- Dobra integracja z ROS 2 i Nav2
- Sprawdzona skuteczność w aplikacjach robotów mobilnych
- Aktywne wsparcie społeczności
- Dostępność dokumentacji i przykładów

Rozdział 3

Wymagania i narzędzia

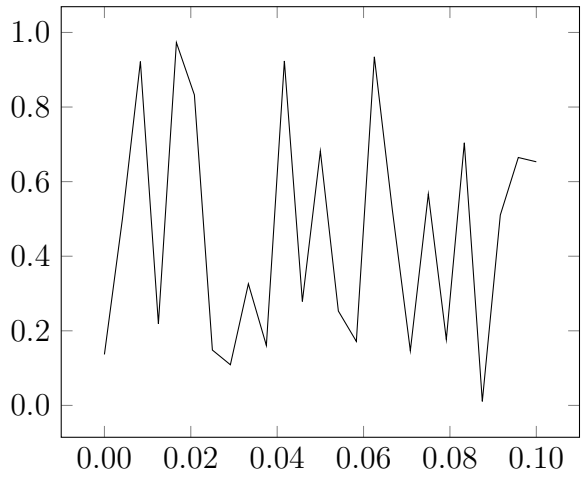
- wymagania funkcjonalne i niefunkcjonalne
- przypadki użycia (diagramy UML) – dla prac, w których mają zastosowanie
- opis narzędzi, metod eksperymentalnych, metod modelowania itp.
- metodyka pracy nad projektowaniem i implementacją – dla prac, w których ma to zastosowanie

Rozdział 4

[Właściwy dla kierunku – np. Specyfikacja zewnętrzna]

Jeśli „Specyfikacja zewnętrzna”:

- wymagania sprzętowe i programowe
- sposób instalacji
- sposób aktywacji
- kategorie użytkowników
- sposób obsługi
- administracja systemem
- kwestie bezpieczeństwa
- przykład działania
- scenariusze korzystania z systemu (ilustrowane zrzutami z ekranu lub generowanymi dokumentami)



Rysunek 4.1: Podpis rysunku po rysunkiem.

Rozdział 5

[Właściwy dla kierunku – np. Specyfikacja wewnętrzna]

Jeśli „Specyfikacja wewnętrzna”:

- przedstawienie idei
- architektura systemu
- opis struktur danych (i organizacji baz danych)
- komponenty, moduły, biblioteki, przegląd ważniejszych klas (jeśli występują)
- przegląd ważniejszych algorytmów (jeśli występują)
- szczegóły implementacji wybranych fragmentów, zastosowane wzorce projektowe
- diagramy UML

Krótka wstawka kodu w linii tekstu jest możliwa, np. `int a;` (biblioteka `listings`). Dłuższe fragmenty lepiej jest umieszczać jako rysunek, np. kod na rys 5.1, a naprawdę długie fragmenty – w załączniku.

```
1  class test : public basic
2  {
3      public:
4        test (int a);
5        friend std::ostream operator<<(std::ostream & s,
6                                       const test & t);
7      protected:
8        int _a;
9
10 };
```

Rysunek 5.1: Pseudokod w `listings`.

Rozdział 6

Weryfikacja i walidacja

- sposób testowania w ramach pracy (np. odniesienie do modelu V)
- organizacja eksperymentów
- przypadki testowe zakres testowania (pełny/niepełny)
- wykryte i usunięte błędy
- opcjonalnie wyniki badań eksperymentalnych

Tabela 6.1: Nagłówek tabeli jest nad tabelą.

	metoda						
			alg. 3			alg. 4, $\gamma = 2$	
ζ	alg. 1	alg. 2	$\alpha = 1.5$	$\alpha = 2$	$\alpha = 3$	$\beta = 0.1$	$\beta = -0.1$
0	8.3250	1.45305	7.5791	14.8517	20.0028	1.16396	1.1365
5	0.6111	2.27126	6.9952	13.8560	18.6064	1.18659	1.1630
10	11.6126	2.69218	6.2520	12.5202	16.8278	1.23180	1.2045
15	0.5665	2.95046	5.7753	11.4588	15.4837	1.25131	1.2614
20	15.8728	3.07225	5.3071	10.3935	13.8738	1.25307	1.2217
25	0.9791	3.19034	5.4575	9.9533	13.0721	1.27104	1.2640
30	2.0228	3.27474	5.7461	9.7164	12.2637	1.33404	1.3209
35	13.4210	3.36086	6.6735	10.0442	12.0270	1.35385	1.3059
40	13.2226	3.36420	7.7248	10.4495	12.0379	1.34919	1.2768
45	12.8445	3.47436	8.5539	10.8552	12.2773	1.42303	1.4362
50	12.9245	3.58228	9.2702	11.2183	12.3990	1.40922	1.3724

Rozdział 7

Podsumowanie i wnioski

- uzyskane wyniki w świetle postawionych celów i zdefiniowanych wyżej wymagań
- kierunki ewentualnych danych prac (rozbudowa funkcjonalna . . .)
- problemy napotkane w trakcie pracy

Dodatki

Spis skrótów i symboli

DNA kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*)

MVC model – widok – kontroler (ang. *model–view–controller*)

N liczebność zbioru danych

μ stopnień przyleżności do zbioru

$\mathbb{E}$ zbiór krawędzi grafu

$\mathcal{L}$ transformata Laplace'a

Źródła

Jeżeli w pracy konieczne jest umieszczenie długich fragmentów kodu źródłowego, należy je przenieść w to miejsce.

```
1 if (_nClusters < 1)
2     throw std::string ("unknown number of clusters");
3 if (_nIterations < 1 and _epsilon < 0)
4     throw std::string ("You should set a maximal number of
          iteration or minimal difference — epsilon.");
5 if (_nIterations > 0 and _epsilon > 0)
6     throw std::string ("Both number of iterations and minimal
          epsilon set — you should set either number of iterations
          or minimal epsilon.");
```

Lista dodatkowych plików, uzupełniających tekst pracy

W systemie do pracy dołączono dodatkowe pliki zawierające:

- źródła programu,
- dane testowe,
- film pokazujący działanie opracowanego oprogramowania lub zaprojektowanego i wykonanego urządzenia,
- itp.

Spis rysunków

Spis tabel