Komparacija algoritama upravljanja za navigaciju autonomnog robota

Jelena Petrović
Katedra za Automatiku
Elektronski fakultet, Univerzitet u Nišu
Niš, Srbija
jelena.petrovic@elfak.ni.ac.rs
0009-0004-5582-3562

Miroslav Milovanović
Katedra za Automatiku
Elektronski fakultet, Univerzitet u Nišu
Niš, Srbija
miroslav.b.milovanovic@elfak.ni.ac.rs
0000-0003-0535-0790

Anđela Đorđević
Katedra za Automatiku
Elektronski fakultet, Univerzitet u Nišu
Niš, Srbija
andjela.djordjevic@elfak.ni.ac.rs
0000-0003-2611-1246

***Abstract*— U okviru istraživanja u domenu autonomne robotike, efektivnost navigacionih algoritama u kompleksnim prostornim strukturama, poput lavirinta, postavlja se kao fundamentalni parametar ocenjivanja sposobnosti autonomnih sistema da adekvatno reaguju na dinamičke i nepredvidive scenarije. Ova tematika dobija posebnu relevantnost u kontekstu rastuće implementacije autonomnih robota u aplikacijama koje mogu da uključuju operacije potrage i spasavanja, automatizovani transport i logistiku, kao i automatizaciju složenih industrijskih procesa, gde je optimalna i pouzdana navigacija neophodna. U ovom radu biće predstavljena četiri algoritma s ciljem utvrđivanja optimalnog algoritma za navigaciju robota i rešavanje lavirinta: nasumični algoritam miša, algoritam praćenja desnog zida, algoritam praćenja levog zida i Tremauxov algoritam. Rezultati analize pokazuju da algoritam praćenja zida dosledno demonstrira brzu navigaciju kroz različite konfiguracije lavirinta, dok Tremauxov algoritam, sa svojom inherentnom sposobnošću učenja i adaptacije, pokazuje izuzetne performanse u pogledu efikasnosti u ponovljenim iteracijama navigacije.**

Ključne reči—Autonomni robot, lavirint, navigacija robota, Webots simulacija

I. Uvod

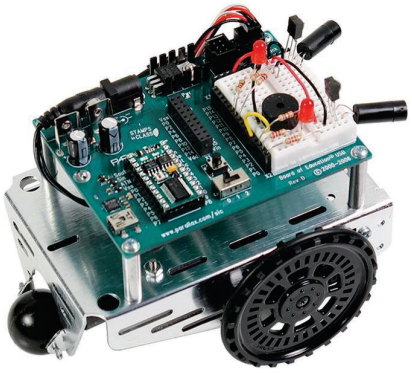
Autonomni roboti se mogu opisati kao roboti koji su sposobni da sami donose odluke u zavisnosti od trenutnog stanja okoline, umesto da samo izvršavaju unapred definisane naredbe. Oni integrišu napredne algoritme veštačke inteligencije i mašinskog učenja sa sposobnostima samostalne navigacije i izvršenja zadataka u promenljivim i nepoznatim okruženjima, kao takvi sve više se koriste u razvoju savremenih tehnoloških sistema. Izdvaja ih sposobnost da obavljaju operacije nezavisno od direktne ljudske intervencije, koristeći samo senzore i aktuatore kako bi ostvarili interakciju sa svojim okruženjem [1]. Ove sposobnosti omogućavaju širok spektar primene autonomnih robota od prenošenja teškog terata pa sve do izvršavanja misija spašavanja i istraživanja nepoznatih terena. Autonomni roboti zahtevaju kompleksne sisteme za percepciju okoline, koji kombinuju senzore i kamere, za prikupljanje podataka o okolini. Podaci dobijeni sa senzora se kasnije obrađuju primenom algoritma za lokalizaciju i mapiranje (SLAM - simultaneous localization and mapping) što robotu omogućava da kreira mapu svog okruženja i precizno odredi svoju poziciju unutar nje [2]. Razvoj autonomnih robota podrazumeva implementaciju algoritama za planiranje puta i algoritama za donošenje odluka, pa samim tim i navigaciju kroz kompleksne sredine. Lokalizacija robota se realizuje posredstvom senzora i kamera uz primenu algoritma za filtriranje podataka poput Kalmanovog filtera [2]. Mapiranje (kreiranje mape) se postiže primenom algoritama koji kombinuju podatke dobijene sa senzora i informacije o trenutnoj lokaciji robota [3]. Planiranje puta podrazumeva analizu interne mape robota u cilju rešavanja problema na što efikasniji način, uzimajući u obzir reakcije na detekciju prepreka koja se vrši putem senzora ili kamera[4]-[5].

U kontekstu navigacije robota može se razmatrati primena autonomnih robota za navigaciju kroz lavirint. S obzirom da su lavirinti statička okruženja (ne menjaju strukturu u vremenu) istraživanje se fokusira na razvoj i optimizaciju algoritama sa ciljem da roboti mogu samostalno da identifikuju i prate najefikasniji put kroz konstruisano okruženje sa različitim preprekama. Autonomni roboti za rešavanje lavirinta su pogodni za evaluaciju različitih pristupa i algoritama u navigaciji, planiranju puta i adaptaciji na promene okruženja. Istraživanja u ovoj oblasti pružaju uvid u mogućnosti primene autonomnih robota u situacijama gde je neophodan visok nivo adaptivnosti i autonomije, kao što su misije spašavanja i istraživanje nepoznatih terena [6].

U ovom radu, cilj je implementacija algoritama za navigaciju Boe-bot robota unutar Webots simulacijskog okruženja u cilju rešavanje lavirinta i poređenje četiri implementirana algoritma navigacije: nasumični algoritam miša, algoritam praćenja levog zida, algoritam praćenja desnog zida i Tremauxov algoritam. Kroz ovu analizu, istražuje se kako različiti pristupi navigaciji utiču na efikasnost autonomnog robota u lavirintu, sa fokusom na vreme potrebno za izlazak iz lavirinta, pređeni put robota i efikasnost u dostizanju cilja. Biće predstavljena diskusija o prednostima i ograničenjima svakog od impelmentiranih algoritama.

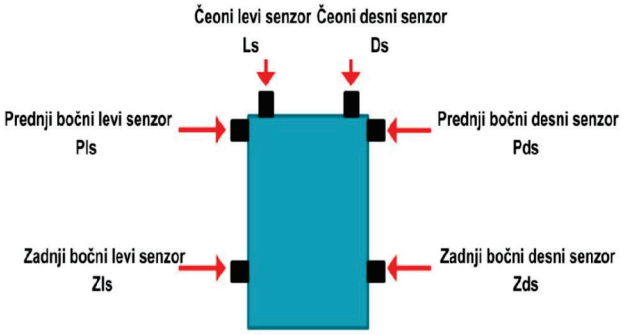
II. Postavke autonomnog robota

Za realizaciju robota implementiran je Boe-bot robot u Webots okruženju. Odabir okruženja baziran je na potrebi za platformom koja omogućava pristupačno eksperimentisanje, jednostavnu integraciju i mogućnost simulacije prilikom primene različitih algoritama. Boe-bot je robot kreiran od strane Parallax Inc. i predstavlja robota sa 3 točka (2 glavna tj.pogonska i jednim pasivnim točkom) [7]. Upravljan je BASIC Stamp 2 programabilnim mikrokontrolerom i pogodan za različite eksperimente i istraživanja Sl. 1.



Slika 1 Boe-bot robot [8]

Boe-bot podržava senzore i aktuatore koji su potrebni za implementaciju algoritma za navigaciju i upravljanje robotom u Webots okruženju. Za potrebe samolokacije i navigacije kroz lavirint implementirana su 6 laserska senzora rastojanja za prikupljanje informacija o spoljašnjem okruženju robota. Na Sl. 2 nalazi se prikaz robota sa naznačenim pozicijama svakog senzora. Postavljena su po 3 senzora sa leve i desne strane robota, po jedan sa čeone strane i po dva na bočnim stranama.



Slika 2 Pozicije senzora na robotu

A. *Laserski senzori rastojanja*

Laserski senzori rastojanja (Lidar- light detection and ranging) su senzori koji koriste laserske zrake za precizno merenje udaljenosti nekog objekta u okolini. Princip rada ovih senzora zasniva se na principu emisije laserskih zraka u okruženje i merenju vremena koje je potrebno da se ti zraci reflektuju o objekat i vrate do senzora [9]. Koristeći poznatu brzinu svetlosti u vakuumu (c~299,792,458 m/s) moguće je precizno izračunati rastojanje od nekog objekta po sledećoj formuli:

$$U_d = \frac{tc}{2}, \quad (1)$$

gde t predstavlja vreme refleksije svetlosti (vreme koje je potrebno da se laserski zrak reflektuje o objekat i vrati do senzora) a U_d predstavlja udaljenost od objekta. Upotreba Lidar senzora pogodna je navigaciju robota i detekciju prepreka pa se samim tim vrlo često koriste za modeliranje percepcije robota o okolini i u različitim algoritmima za autonomnu navigaciju.

III. POSTAVKE SIMULACIJE

U ovom poglavlju će biti predstavljena četiri algoritma za navigaciju i rešavanje lavirinta, praćena simulacijom u Webots okruženju. Cilj ove analize je pružiti uvid u performanse i karakteristike algoritama u kontekstu rešavanja problema navigacije kroz kompleksne lavirinte. Kroz simulaciju u Webots okruženju, moguće je detaljno proučiti ponašanje svakog algoritma u kontrolisanom okruženju, istražujući njihove prednosti, nedostatke i primenljivost u praktičnim situacijama.

A. *Algoritmi*

1) *Nasumični algoritam miša*

Nasumični algoritam miša koristi metod slučajne selekcije putanje u procesu navigacije kroz lavirint i spada u grupu algoritama za stohastičko pretraživanje. Takođe, kao i ostala tri algoritma razmatrana u ovom radu, i ovaj algoritam je pogodan za navigaciju kroz nepoznato okruženje što je ekvivalentno lavirintu. Logika ovog algoritma zasniva se na iterativnom i nasumičnom odabiru smera kretanja, uz kontinualnu evaluaciju dostupnih putanja, na osnovu podataka dobijenih sa senzora. Nakon inicijalizacije, robot procenjuje dostupne putanje kroz lavirint i na osnovu informacija sakupljenih sa senzora po algoritmu nasumično bira smer kretanja [10]. Robot se zatim kreće u odabranom smeru i ponavlja ovaj proces sve dok ne pronađe izlaz iz lavirinta. Na Sl. 3. nalazi se se uprošćeni kod ovog algoritma.

```
1. start
2. while (nijePronadjenIzlaz) do
3.     procitajSenzore()
4.     if (imaPreprekaNapred) then
5.         nasumicnoOdaberiNoviSmer()
6.     else
7.         idiNapred()
8.     end if
9.     if (naIzlazu()) then
10.        nijePronadjenIzlaz = false
11.    end if
12. done
13. stop
```

Slika 3 Uprošćeni kod nasumičnog algoritma miša

While petlja označava glavnu kontrolnu petlju koja se nastavlja sve dok robot ne pronađe izlaz. Prilikom svakog prolaska kroz glavnu petlju očitavaju se vrednosti sa senzora pozivom funkcije za očitavanje senzora. Ukoliko robot detektuje prepreku ispred sebe nasumično će odabrati jedan od puteva koji mu se nude tj. nasumično će odabrati smer kojim će nastaviti kretanje.

2) *Algoritam praćenja desnog zida*

Ovaj algoritam se zasniva na principu kontinualnog kontakta sa zidom sve vreme prolaska kroz lavirint. Na ovaj

način robot održava stalnu orjentaciju u odnosu na zid, čime se minimizuje mogućnost odabira pogrešnog puta i povećava efikasnost navigacije. Za razliku od slučajnog algoritma miša, ovaj algoritam koristi deterministički prisup navigaciji, što znaci da se odluke o kretanju robota donose na osnovu konkretnih informacija iz okruženja. U tom kontekstu se misli na odabir smera kretanja prilikom nailaženja na prepreku, gde u ovom slučaju robot na osnovu informacija sa senzora bira put sa primarnim zahtevom da održi kontakt sa desnim zidom [12]. U nastavku se nalazi uprošćeni kod koji opisuje navigaciju robota po algoritmu za praćenje desnog zida.

```
1. start
2. while (nijeIzlaz) do
3.     procitajSenzore()
4.     if (imaPreprekaNapred) then
5.         okreniSeUDesno()
6.     else
7.         idiNapred()
8.     end if
9.     if (naIzlazu()) then
10.        nijeIzlaz = false
11.    end if
12. done
13. stop
```

Slika 4 Uprošćeni kod algoritma za praćenje desnog zida

Slično kao i u nasumičnom algoritmu miša glavna petlja je while petlja kojom se proverava da li je robot izašao iz lavirinta i prilikom svakog prolaska kroz petlju, proveravaju se informacije sa senzora. Ukoliko robot detektuje prepreku nastaviće desno kako bi održao kontakt sa desnim zidom.

3) Algoritam praćenja levog zida

Algoritam za praćenje levog zida je poput algoritma za praćenje desnog zida varijacija algoritma za praćenje zida. Logika i princip rada su u osnovi isti, dok je razlika jedino u zidu koji robot prati prilikom kretanja kroz lavirint. U slučaju gde je robot pratio desni zid, ukoliko se detektuje prepreka robot bi nastavio desno kako bi održao kontakt sa desnim zidom, međutim kod praćenja levog zida, robot će nastaviti levo i održavati kontakt sa levim zidom [12].

4) Tremauxov algoritam

Tremauxov algoritam se zasniva na principu sistema praćenja tragova. Oslanja se na dva osnovna principa: sistematizovano pretraživanje prostora i upotrebu tragova kao navigacionih informacija. Prva ideja je da se lavirint istražuje sistematično, s ciljem otkrivanja svih mogućih puteva i identifikacije svih putanja do izlaza. Druga ideja je da se koriste tragovi, koji se ostavljaju na svakoj raskrsnici koju robot poseti, kako bi se sprečio ponovni prolazak kroz iste putanje i olakšala navigacija kroz lavirint. Robot zapravo postavlja i ažurira tragove na svim raskrsnicama koje poseti, beležeći broj prolaza kroz svaku putanju. Na osnovu ovih informacija, robot donosi odluke o izboru putanja koje će istražiti, dajući prednost neistraženim putevima ili onima s manjim brojem prolaza [12].

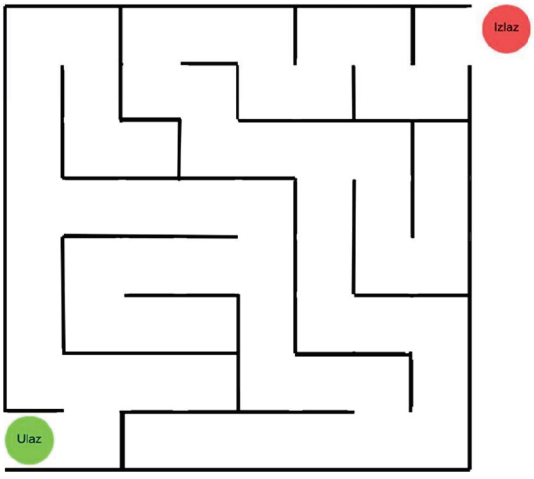
```
1. Inicijalizacija: Postavi robota na početnu tačku lavirinta.
2. while (nijeIzlaz) do
3.     if (neposećenaRaskrsnica()) then
4.         postaviTragoveNaSvePuteve()
5.     else if (posećenaRaskrsnicaSaNeistraženimPutem()) then
6.         odaberiNeistraženiPut()
7.     else
8.         vratiseNaPrethodnuRaskrsnicu()
9.     end if
10.    proveraIzlaza()
11. end while
```

Slika 5 Uprošćeni kod Tremauxovog algoritma

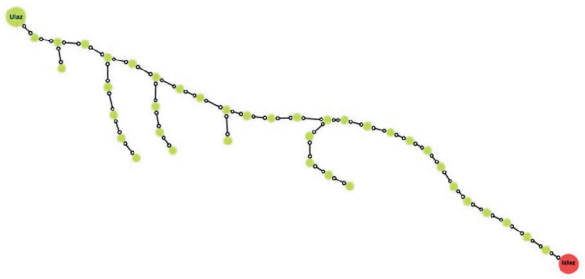
Inicijalizacijom algoritma, robot se postavlja na početnu tačku lavirinta i ulazi u while petlju koja omogućava kontinuirano napredovanje kroz lavirint do pronalaska izlaza. U procesu istraživanja, funkcija neposećenaRaskrsnica() igra ulogu u identifikaciji novih, neposećenih raskrsnica. Robot implementira strategiju postavljanja tragova na sve puteve raskrsnice preko funkcije postavi TragoveNaSvePuteve(), označavajući ih kao istražene. Kada robot naiđe na raskrsnicu koja je već ranije bila posećena ali još sadrži neistražene puteve, poziva se funkcija posećena RaskrsnicaSaNeistraženimPutem(), navodeći robota da odabere jedan od preostalih neistraženih puteva za dalje istraživanje. U situaciji kada su svi putevi na trenutnoj raskrsnici već istraženi, robot se usmerava na povratak prema prethodnoj raskrsnici u potrazi za novim neistraženim putevima, dok paralelno proverava mogućnost izlaska na kraju svake iteracije. Proces se završava kada je izlaz iz lavirinta uspešno pronađen.

B. Konstrukcija i inicijalizacija algoritma

Lavirinti su kompleksne strukture koje se često koriste u robotici za testiranje i razvoj navigacionih algoritama i autonomnih sistema. Lavirinti predstavljaju oblasti koje se koriste za simulaciju realnih okruženja, čime se omogućava analiza performansi i efikasnosti algoritama kroz virtuelnu ili fizičku interakciju robota sa okolinom. U Webots okruženju kreiran je tzv. savršeni lavirint koji ima osobinu da su svi zidovi lavirinta međusobno spojeni ili spojeni sa granicom lavirinta. Ovakvi lavirinti ne sadrže petlje i kao takvi su ekvivalentni stablima u teoriji grafova, pa se samim tim tako mogu i predstaviti. Na Sl. 6. nalazi se izgled lavirinta koji je kreiran nasumičnim i slobodnim odabirom, modeliranjem dostupnih objekata u programskom paketu. Lavirint je nasumično kreiran kako ne bi bio pogodniji za neki od tipova razmatranih algoritama za navigaciju i rešavanje lavirinta. Graf kreiranog lavirinta nalazi se na Sl. 7.



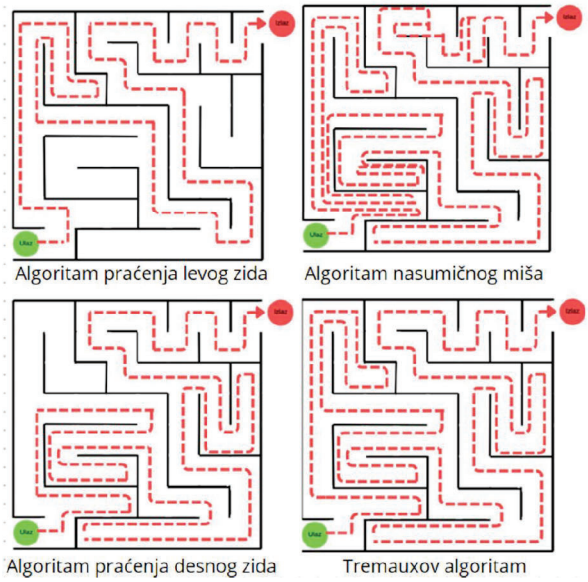
Slika 6 Konstrukcija lavirinta



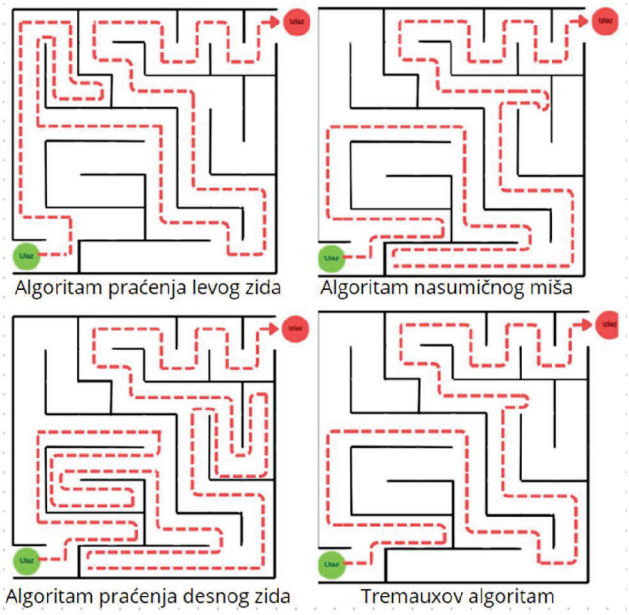
Slika 7 Konstrukcija lavirinta predstavljena grafom

IV. REZULTATI SIMULACIJA

U okviru ovog istraživanja, fokusiramo se na uporednu analizu performansi četiri predstavljena algoritma za rešavanje i navigaciju kroz lavirint. Svaki od ovih algoritama implementiran je na Boe-bot robotu unutar Webots okruženja, pri čemu je svaki podvrgnut dvostrukom testiranju kroz identičan lavirint. Cilj ovog segmenta istraživanja jeste da se, putem simulacija, evidentiraju i uporede ključni parametri performansi: vreme potrebno da robot pronađe izlaz iz lavirinta, kao i ukupna dužina puta kojim je robot prošao, uzimajući u obzir fiksnu udaljenost između početne i krajnje tačke od 1950 metara. Analizom prikupljenih podataka, težimo da odredimo prednosti i mane svakog od testiranih algoritama, pružajući tako uvid u njihovu efikasnost i praktičnu primenu. Snimljeno je 8 simulacija, po dve za svaki algoritam. Na Sl. 8 i 9, prikazane su putanje kretanja robota za svaku od osam simulacija. Na Sl. 8 prikazane su putanje kojima se robot kretao, zabeležene tokom prve opservacije, po algoritmu za praćenje levog zida, algoritmu za praćenje desnog zida, nasumičnom algoritmu miša i Tremauxovom algoritmu respektivno. Na Sl. 9 nalaze se putanje zabeležene tokom druge opservacije, u istom redosledu kao i na Sl. 8. Vreme potrebno za nalaženje izlaza i pređeni put su predstavljeniu tabeli I. i tabeli II.



Slika 8 Putanje zabeležene tokom prve opservacije



Slika 9 Putanje zabeležene tokom druge opservacije

TABELA I. EKSPERIMENTALNI REZULTATI PRVE OPSERVACIJE

R.br	*Algoritam za rešavanje lavirinta*	*Vreme potrebno za nalaženje izlaza*	*Udaljenost ciljne tačke*	*Pređeni put*
1	Levi zid	5:28 min	1950 m	3750 m
2	Desni zid	7:48 min	1950 m	5650 m
3	Nasumični miš	12 min	1950 m	7850 m
4	Treamaux	9:48 min	1950 m	6300 m

TABELA II. EKSPERIMENTALNI REZULTATI DRUGE OPSERVACIJE

R.br	*Algoritam za rešavanje lavirinta*	*Vreme potrebno za nalaženje izlaza*	*Udaljenost ciljne tačke*	*Pređeni put*
1	Levi zid	5:30 min	1950 m	3750 m
2	Desni zid	7:32 min	1950 m	5650 m
3	Nasumični miš	4:12 min	1950 m	2850 m
4	Treamaux	3:12 min	1950 m	2000 m

U ovoj analizi performansi primetno je da su vremena potrebna za pronalaženje izlaza iz lavirinta varirala u zavisnosti od primenjenog algoritma. Algoritam praćenja levog zida se konstantno isticao kao najefikasniji, sa minimalnim vremenom potrebnim za izlazak, što sugeriše njegovu pouzdanost u ovom „savršenom“ lavirintu.

Nasuprot tome, algoritam praćenja desnog zida, iako sličan u pristupu, pokazao je nešto duže vreme za izlazak, što ukazuje na to da izbor strane može imati značajan uticaj na efikasnost rešenja, posebno u lavirintima gde struktura lavirinta više odgovara praćenju jedne strane lavirinta. Lavirint kroz obe opservacije ostaje nepromenjen, što u slučaju algoritama za praćenje levog i desnog zida rezultira minimalnim oscilacijama u vremenima potrebnim za pronalaženje izlaza. Može se zaključiti da primenom jednog od algoritama za praćenje zida, robot prelaziti iste putanje, bez obzira na to koliko puta prolazi kroz lavirint, pa će samim tim i vreme potrebno za pronalaženje izlaza biti jednako u svakoj opservaciji. Nasumični algoritam miša pokazao je veliku varijabilnost u performansama, što ukazuje na njegovu nepredvidivost. U drugoj simulaciji, ovaj algoritam je zabeležio vrlo dobre rezultate i za 4 min i 12 s pronašao izlaz, dok je u prvoj bio znatno sporiji, ističući nasumičnost kao faktor koji može biti koristan u nekim situacijama, ali generalno manje pouzdan za konzistentno brzo rešavanje lavirinata. Na osnovu putanje date na Sl. 9 može se zaključiti da je robot upravljan po algoritmu slučajnog miša zalutao samo jednom, dok bi najbolji rezultat ovog algoritma prikazao putanju poput putanje dobijene Tremauxovim algoritmom pri drugoj opservaciji. Tremauxov algoritam, sa druge strane, pokazao je značajnu razliku (6 min i 36 s) u vremenima između dve simulacije, sa poboljšanjem u drugom pokušaju. Ovo ukazuje da njegova sposobnost da uči iz prethodnih iskustava može dosta uticati na njegovu efikasnost, čineći ga potencijalno najboljim izborom za situacije gde je moguće primeniti naučene informacije iz prvog prolaza kroz lavirint.

V. Zaključak

U okviru autonomne robotike, navigacija robota predstavlja važan parametar za ocenjivanje performansi robota. S obzirom da lavirinti predstavljaju kompleksne strukture, često se koriste za testiranje i razvoj navigacionih algoritama u autonomnim sistemima. U ovom radu su analizirane performanse četiri algoritama za navigaciju robota (nasumični algoritam miša, algoritam praćenja desnog zida, algoritam praćenja levog zida i Tremauxov algoritam) sa ciljem utvrđivanja optimalnog algoritma za rešavanje lavirinta.

Kada se ovi algoritmi uporede, jasno je da svaki ima specifične prednosti koje mogu biti korisne u određenim problemima i situacijama. Algoritmi za praćenja zida pokazali su se kao pouzdani i najkonstantniji kada postoji zahtev za efikasnim izlaskom iz potpuno nepoznatog lavirinta i može se reći da pružaju zadovoljavajuće performanse. Nasumični algoritam miša nudi mogućnost iznenađujuće brzih rešenja, ali je i suviše nepredvidiv i nije pogodan za probleme koji zahtevaju visok nivo efikasnosti i pouzdanosti. Tremauxov algoritam zapravo predstavlja primer adaptivnog pristupa koji može maksimalno iskoristiti prethodna iskustva za efikasnije rešavanje problema. Ova uporedna analiza ističe važnost odabira pravog algoritma u zavisnosti od specifičnih karakteristika lavirinta i ciljeva rešavanja.

Dalje istraživanje biće usmereno na poboljšanje postojećih algoritama za navigaciju autonomnih robota. Fokus će biti na impelementaciji inteligentnih metoda u cilju formiranja efikasnijih algoritama. Takođe jedan od ciljeva je testiranje algoritama na realnim sistemima.

Reference/Literatura

[1] B. Siciliano, O. Khatib Robotics and the Handbook. In Springer Handbook of Robotics (pp. 1-6). Cham: Springer International Publishing. (2016)

[2] S. Alamri, S. Alshehri, W. Alshehri, H. Alamri, A. Alaklabi, T. Alhmiedat, Autonomous maze solving robotics: Algorithms and systems. International Journal of Mechanical Engineering and Robotics Research, 10(12), 668-675, (2021)

[3] S. Thrun, Probabilistic robotics. Communications of the ACM, 45(3), 52-57 (2022)

[4] Russell, J. Stuart, P. Norvig, Artificial intelligence: a modern approach. Pearson. (2016) Z. Jian, Z. Yan, X. Lei, Z. Lu, B. Lan, X. Wang, B. Liang, Dynamic control barrier function-based model predictive control to safety-critical obstacle-avoidance of mobile robot. In 2023 IEEE International Conference on Robotics and Automation (ICRA) (pp. 3679-3685). IEEE. (2023)

[5] Murphy, R. Robin "International cooperation in deploying robots for disasters: Lessons for the future from the Great East Japan Earthquake." 日本ロボット学会誌 32.2 (2014): 104-109.

[6] R. Balogh, Basic activities with the Boe-Bot mobile robot. In Proceedings of conference DidInfo (2008)

[7] A. Lindsay, Robotics with the Boe-Bot, Parallax Inc.

[8] A. Valle, Bilby Rover autonomous mobile robot in the context of Industry 4.0-Survey on sensor-enabled applications through Webots simulations and implementation of Hector-SLAM algorithm for autonomous navigation (Doctoral dissertation, Politecnico di Torino) (2021)

[9] S. Yadav., K. K. Verma, S. Mahanta, The Maze problem solved by Micro mouse. International Journal of Engineering and Advanced Technology (IJEAT) ISSN, 2249, 8958. (2012)

[10] I. Hammad, I, K. El-Sankary, J. Gu, A comparative study on machine learning algorithms for the control of a wall following robot. In 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO) (pp. 2995-3000). IEEE. (2019)

[11] L. Bienias, K. Szczepański, P. Duch, Maze exploration algorithm for small mobile platforms. Image Processing & Communications, 21(3), 15. (2016)

ABSTRACT

Within the research domain of autonomous robotics, autonomous system ability to adequately respond to dynamic and unpredictable scenarios is evaluated by observing the effectiveness of navigational algorithms in complex spatial structures such as mazes. This topic is especially relevant in situations where optimal and reliable navigation is necessary, such as implementation of autonomous robots for search and rescue operations, automated transport and logistics, as well as the automation of complex industrial processes. This paper presents a comparative analysis of four algorithms for robot navigation and maze solving: *random mouse*, *follow-the-right-wall*, *follow-the-left-wall* and *Tremaux's* algorithm. The main goal in this research is determining the optimal navigation algorithm. The results of the analysis show that the wall-following algorithm consistently demonstrates rapid navigation through various maze configurations, while Tremaux's algorithm, with its inherent learning and adaptation capability, exhibits exceptional performance in terms of efficiency in repeated iterations of navigation.

Comparison of Control Algorithms for Autonomous Robot Navigation

Jelena Petrović, Miroslav B. Milovanović, Anđela Đorđević