|  |  |
| --- | --- |
|  | Zbornik radova Fakulteta tehničkih nauka, Novi Sad |

**UDK: 621.31**

**DOI:** [**https://doi.org/10.24867/17BE17Vujic**](https://doi.org/10.24867/17BE17Vujic)

**SIMULACIJA ROBOTA BAZIRANOG NA DIFERENCIJALNOM POGONU KORIŠĆENJEM GAZEBO SIMULATORA I ROS-A**

**SIMULATION OF THE ROBOT BASED ON DIFFERENTIAL DRIVE BY USING GAZEBO SIMULATOR AND ROS**

Đorđe Vujić, *Fakultet tehničkih nauka, Novi Sad*

**Oblast – ELEKTROTEHNIKA I RAČUNARSTVO**

**Kratak sadržaj –** *U ovom radu razvijen je simulator robota sa diferencijalnim pogonom, uz oslonac na Gazebo platformu za simulaciju robotskih aplikacija i robotski operativnisistem (ROS). Razvijenim simulatorom se posebno razmatra sprega ova dva alata. Analiza rezultata vršena je korišćenjem raznih alata kompatibilnih sa robotskim operativnim sistemom.*

**Ključne reči:** *Simulacija, PID, ROS, Gazebo, diferencijalni pogon, kretanje, robot*

**Abstract** – *In this paper, simulator of the differential drive robot is developed, which relies on Gazebo platform for simulation of robot applications and robot operating system (ROS). Simulator which is developed provides a common analysis between Gazebo and ROS. Analysis of the results is performed by using different tools which are compatible with ROS.*

**Keywords:** *Simulation, PID, ROS, Gazebo, differential drive, movement, robot*

# UVOD

ROS (*engl. Robot Operating System*) je set softverskih biblioteka i alata koji pomažu u pravljenju robotskih aplikacija. ROS olakšava razvoj kompleksnih i robusnih robotskih platformi različitih tipova.

Zbog toga što je pravljenje robusnog softvera za generalnu upotrebu veoma teško, rodila se ideja za kolaboracijom. ROS je rezultat kolaboracije različitih pojedinaca, laboratorija i institucija, gde svako razvija ono za šta je specijalizovan.

Na primer, jedna laboratorija ima eksperte iz oblasti mapiranja unutrašnjeg okruženja i može da učestvuje u sistemima za pravljenje mapa. Druga grupa možda ima eksperte u korišćenju mapa radi vršenja navigacije, a treća grupa će razvijati korišćenje kompjuterske vizije za detekciju objekata u toku kretanja.

ROS je dizajniran specijalno za ovakve grupe kako bi svaka mogla da vrši razvoj i dostiže svoje ciljeve uz pomoć produkata rada neke druge grupe.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**NAPOMENA:**

**Ovaj rad proistekao je iz master rada čiji mentor je bio dr Milan Rapaić, red. prof.**

# KONCEPTI

## Čvorovi

Čvorovi (*eng. nodes*) su procesi koji vrše izračunavanja. ROS je dizajniran da bude modularan, tako jedan robotski sistem uglavnom sadrži mnogo čvorova. Na primer, jedan čvor kontroliše laserski senzor, drugi čvor kontroliše točkove na motorima, a treći čvor vrši lokalizaciju.

## Poruke

Čvorovi komuniciraju jedni sa drugima prosleđujući poruke. Poruka je jednostavna struktura podataka, sa tipiziranim poljima.

## Topici

Čvorovi šalju poruke objavljujući ih topicima (*eng. topic*). Topik je ime koje se koristi za identifikaciju sadržaja poruke. Čvor koji je zainteresovan za odrađenu vrstu podataka, pretplatiće se na odgovarajući topik.

## Servisi

Servisi su par struktura poruka: jedna za zahtev, a druga za odgovor. Čvorovi koji obezbeđuju servise, nude ih pod određenim imenom, a klijenti ih koriste šaljući zahtev u vidu poruke i čekaju odgovor.

# KOMUNIKACIONA INFRASTRUKTURA

## Objavi/pretplati se

Poruke su rutirane kroz transportni sistem putem objavi/pretplati se metodologije. ROS sadrži ugrađen i dobro testiran sistem razmene poruka između čvorova, korišćenjem topika. Ovaj sistem štedi vreme razvoja jer nije potrebno obraćati pažnju na tehničku pozadinu implementacije pri korišćenju objavi/pretplati se mehanizma između anonimnih čvorova.

Druga korist korišćenja sistema prosleđivanja poruka jeste da korisnika primorava na implementaciju čistog interfejsa između čvorova u sistemu, i tako poboljšava enkapsulaciju i ponovno korišćenje već napisanog koda gde god je to moguće.

Struktura interfejsa poruka je definisana standardom message IDL *(eng. Interface Description Language)* [[1](#Obj21)].

## Snimanje i reprodukcija poruka

Zbog toga što je objavi/pretplati se sistem asinhron i anoniman, podaci mogu jednostavno biti snimljeni i reprodukovani bez bilo kakvih izmena u kodu. Za primer se može uzeti komunikacija između Čvora A i Čvora B. Neka Čvor A čita podatke sa senzora, a razvija se Čvor B koji će da procesira podatke koje proizvede Čvor A. ROS čini jednostavnim da se podaci koje je Čvor A objavio čuvaju u datoteku.

Apstrakcija prosleđivanja poruka omogućava da Čvor B nema potrebu za poznavanjem stvarnog izvora podataka, što bi mogao biti Čvor A ili log fajl u ovom slučaju. Ovakav dizajn omogućava da se smanji napor uložen u razvoj, i povećavaju fleksibilnost i modularnost sistema.

## Pozivi udaljenih procedura

Asinhrona priroda objavi/pretplati se sistema poruka radi za mnoge komunikacione potrebe u robotskim aplikacijama, ali su ponekad potrebne i sinhrone zahtev/odgovor (eng. request/response) interakcije između procesa. ROS kao srednji sloj (eng. middleware) omogućava korišćenje servisa. Podaci poslati između procesa u pozivu servisa su definisani u istom, jednostavnom opisu poruke (message IDL).

## Jezik za opis robota – URDF

ROS omogućava set alata za opisivanje i modeliranje robota koji je kompatibilan sa ostatkom sistema. Format za opisivanje robota zove se URDF (*eng. Unified Robot Description Format*) koji se bazira na XML notaciji u kojoj se opisuju fizičke osobine robota, kao što su dužina, pozicija, boja i slične.

# ALATI

## Alati iz komandne linije

ROS može biti korišćen u potpunosti bez grafičkog korisničkog intefrejsa. Sve ključne funkcionalnosti su dostupne kroz više od 45 alata iz komandne linije. Tu su komande za pokretanje grupa čvorova, proveru topika i servisa, pravljenje snimaka i njihova reprodukcija kao i ostale situacije. Ukoliko korisnik preferira grafičke alate, [[2](#Dav18)]i *rqt* [[3](#Tho16)]omogućuju sličnu i proširenu funkcionalnost. Više detalja o ovim alatima dajemo u nastavku.

## Rqt

ROS sadrži *rqt* [[3](#Tho16)], Qt [[4](#The21)] bazirani alat za razvoj grafičkih interfejsa za robota. Moguće je kreirati interfejse specifične za samu aplikaciju, kao i same dodatke za rqt. rqt\_graph [[5](#Dir18)] dodatak omogućava proveru i vizualizaciju ROS sistema, prikazivajući čvorove i veze između njih, omogućavajući da se jednostavanpregled trenutnog sistema.

*rqt\_plot* [[6](#Dor18)]dodatak omogućava monitoring raznih signala kao što su enkoderski signali, naponi, ili bilo koji drugi signali. Ovaj dodatak će crtati grafike izabranih signala. Slika 4.1 - Primer izgleda rqt\_plot grafičkog interfejsa ilustruje izgled grafičkog interfejsa rqt\_plot alatke.



Slika . - Primer izgleda rqt\_plot grafičkog interfejsa

## Rviz

Rviz omogućava 3-dimenzionu vizualizaciju mnogih tipova senzorskih podataka i robota opisanih u URDF formatu. Rviz može da prikaže mnoge tipove poruka ugrađenih u ROS, kao što su poruke sa laserskih skenera, trodimenzionih oblaka i slika sa kamera. On takođe koristi informaciju iz tf [[7](#Tul17)] biblioteke da bi prikazao sve podatke sa senzora u zajedničkom koordinatnom sistemu od korisnikovog izbora, zajedno sa 3D modelom robota. Vizualizacija svih podataka u istoj aplikaciji omogućava pregled toga što robot ,,vidi”, i da se uvide problemi kao što su neravnomernosti kod senzora ili nepravilnosti u modelu robota. Primer robota u rviz okruženju prikazan je na Slika 4.2 - Primer rviz korisničkog interfejsaSlika 4.2:



Slika . - Primer rviz korisničkog interfejsa

# GAZEBO

Gazebo je programski paket otvorenog koda koji se koristi za simulaciju robotskih sistema i njihovog okruženja. Od 2004. do 2011. Gazebo je integrisao fizički podsistem za simulaciju dinamičkih sistema (opisanih diferencijalnim jednačinama – ODE engine), OpenGL renderovanje, kao i podršku za simulaciju senzora i kontrolu aktuatora.

Gazebo može da koristi više fizičkih podsistema za simulaciju, visokih performansi, kao što su ODE, Bullet i slični. Omogućava realistično renderovanje okruženja uključujući visoko kvalitetno osvetljenje, senke, i teksture. Može da modeluje senzore koji ,,vide” simulirana okruženja, kao što su laserski senzori, kamere, radari i slični [[8](#Ope)].

# KINEMATIKA DIFERENCIJALNOG POGONA

Pošto točkovi mogu da se okreću različitim brzinama jedan u odnosu na drugi, može se uzeti slučaj da se centar robota kreće konstantno po zamišljenoj kružnici. U tom slučaju brzine točkova su različite jedna u odnosu na drugu, ali su konstantne. Tačka oko koje se kreću nazvana je ICC, i prikazana na Slika 6.1 u odnosu na centar diferencijalnog pogona.



Slika . - Kinematika diferencijalnog pogona

Promenom brzina točkova, moguće je varirati trajektoriju po kojoj se robot kreće. Pošto ugaona brzina rotacije ω mora biti ista za oba točka u slučaju kretanja oko tačke ICC, moguće je napisati sledeće jednačine:

gde je L distanca između centara točkova, VR i VL brzine desnog i levog točka respektivno, u odnosu na nepokretnu podlogu, i R oznaka za razdaljinu središnje tačke robota u odnosu na ICC. U bilo kom trenutku moguće je izraziti R i ω kao:

Tri slučaja su od posebnog interesa:

1. Ukoliko je VL = VR, onda je to pravolinijsko kretanje konstantnom brzinom. R postaje beskonačno, i nema rotacije, što znači da je ω = 0.
2. Ukoliko je VL = -VR, tada je R = 0, i imamo rotaciju oko centralne tačke – rotacija u mestu.
3. Ukoliko je VL = 0 i VR ≠ 0, tada imamo rotaciju oko levog točka. U tom slučaju, R = . Ista logika važi i u obrnutom slučaju kad je VR = 0 a VL≠ 0.

## DIREKTNI KINEMATSKI PROBLEM

Pitanje: kako će se robot kretati u prostoru ako znamo kretanje pojedinačnih točkova robota? Na Slika 6.1, treba primetiti da je robot na poziciji (x, y), orijentisan pod uglom θ u odnosu na X osu.

Manipulacijom parametara VL, VR robot može biti pomeran na različite pozicije i biti različito orijentisan. Poznajući VL i VR , i koristeći jednačinu 3, može se naći sledeća lokacija tačke ICC:

i u vremenu pozicija robota biće:

Ova jednačina jednostavno opisuje kretanje robota na distanci R oko ICC tačke, ugaonom brzinom *ω*.

## INVERZNI KINEMATSKI PROBLEM

Pitanje: ako je zadato željeno kretanje mobilnog robota u prostoru, kako treba da se kreću pojedinačno točkovi, da bi se to kretanje i ostvarilo? Generalno, pozicija robota koji se kreće u smeru *θ(t)* brzinom *v(t)* može se opisati kao:

Radi uprošćavanja putanje kretanja robota sa diferencijalnim pogonom, biće razmotrena dva slučaja:

1) Robot se kreće pravolinijski Jednačina kretanja postaje:

2) Robot se rotira u mestu Jednačina kretanja postaje:

Ovaj pristup motivisan je strategijom da se robot kreće pravolinijski, zatim rotira u mestu, i zatim ponovo nastavlja pravolinijski po strategiji navigacije.

# SOFTVERSKA ARHITEKTURA SISTEMA BAZIRANOG NA ROS\_CONTROL PAKETU

Paket *ros\_control* dobija stanja zglobova kao ulaze. Koristi generičku upravljačku petlju sa povratnom spregom, tipično PID regulator, da bi upravljao izlazima. Upotreba paketa se dodatno komplikuje pri upotrebi nad realnim sistemima koji nemaju jedan prema jedan mapiranje očitanih i realnih pozicija zglobova. Takvi problemi prevazilaze se korišćenjem transmisija.

Transmisije su tu da oponašaju mehaničke prenosnike, poput reduktora sa datim prenosnim odnosom *n*, tako što mapiraju promenljive ulaznog momenta na promenljive izlaznog momenta pri čemu snaga ostaje nepromenjena.

U nastavku sledi objašnjnje pojedinih delova ovakvog softvera. Controller Manager je komponenta zadužena za regulatore iz ros\_control paketa. Najbitnije naredbe su one za učitavanje i uklanjanje regulatora (load\_controller i *unload\_controller*). Sledeća je naredba za promenu regulatora (*switch\_controller*) i naredba za izlistavanje aktivnih regulatora (*list\_controllers*). Funkcija *update()* poziva se periodično i vrši ažuriranje raznih parametara koje koristi regulator, takođe i proračun upravljanja zavisno od tipa regulatora koji se koristi. Sledeća komponenta je komponenta zadužena za spregu sa spoljašnjim svetom u pogledu hardvera. Shodno ovoj komponenti, komunikacija će ići u pravcu simulacije ili realnog hardvera. Ova komponenta stanja zglobova dostavlja Controller Manager komponenti u radijanima, a od nje u ovom primeru preuzimaželjeni obrtni moment. Odatle se podaci mapiraju na klasu hardware\_interface. U slučaju simulacije, metode ove klase će da generišu samo upravljanje ka objektu simulacije, a takođe će iz njega da čitaju stanja simuliranih hardverskih komponenti. Treba još napomenuti da je u ovom slučaju robot opisan kroz URDF alat, i da je potrebno učitati Gazebo dodatak nazvan *gazebo\_ros\_control*. Kada je realni hardver u pitanju, metode klase hardware\_interface imaju zadatak da vrše njegovu apstrakciju. One su zadužene da realnom regulatoru prenesu naredbu. U ovom primeru to je moment koji je prethodno izračunat, a može da bude i pozicija ili brzina. Realni regulator može da bude mikrokontroler na kojem je implementiran PID regulator koji vrši regulaciju po parametru dobijenom iz hardverskog interfejsa. On dalje vrši upravljanje nad aktuatorima, i čita stanja sa senzora.

# PRIMER: APLIKACIJA ROS\_DIFF\_DRIVE

## UVOD

Aplikacija ros\_diff\_drive pisana je za potrebe ovog rada. Ideja je da se kreira robot koji će biti simuliran u Gazebo okruženju. Robot je pogonjen diferencijalnim pogonom. Opisan je korišćenjem URDF alata. Kompletan programski paket ove aplikacije nalazi se na Github platformi, na sledećoj web stranici: [*https://github.com/djordjevujic/ros\_diff\_drive*](https://github.com/djordjevujic/ros_diff_drive)*.*

Uprošćeni dijagram sistema softvera prikazan je na Slika 8.1:



Slika . - Uprošćeni dijagram softvera ros\_diff\_drive sistema

Pojednostavljeno objašnjenje segmenata dijagrama arhitekture sistema sledi u nastavku:

- x, y, θ – Željena pozicija robota u globalnom koordinatnom sistemu.

- APPL – Aplikativni sloj softvera. Donosi odluke o kretanju robota, u vidu koordinata tačke ka kojoj se robot kreće, željene rotacije, linearne i rotacione brzine

- cmd\_vel – Komanda koja predstavlja željenu brzinu. U ROS sistemu, to je zaseban topik.

- diff\_drive\_controller – Niži sloj softvera koji vrši proračun brzine točkova. U odnosu na trenutnu, proračunava željene brzine točkova koje prosleđuje hardverskom interfejsu. Ova softverska biblioteka je deo standarnih ROS biblioteka.

- hw\_interface – Hardverski interfejs. Služi kao apstrakcija hardvera u odnosu na ostatak sistema. Omogućava da se u odnosu na okruženje, u sistemu koristi realan hardver ili simulacija.

- odom – Topik koji predstavlja stanje odometrije diferencijalnog pogona. Vrednosti bivaju osvežene od strane sloja nazvanog ,,diff\_drive\_controller” koji vrši regulaciju brzine točkova.

## INKREMENTALNI PID REGULATOR

Inkrementalni PID regulator implementiran je u datoteci *regulator.py*, u okviru klase *Regulator()*. Metoda za proračun upravljanja navedena je u metodi *pid\_incremental*:

def pid\_incremental(self, error):

 # Calculate control

 dUp = self.KP \* (error - self.err\_prev)

 dUi = self.KI \* self.T \* error

 dUd = self.KDT\*(error-2\*self.err\_prev+self.err\_p\_prev)

 self.u = self.u + dUp + dUi + dUd

 # Anti wind-up

 if self.u > self.u\_limit:

 self.u = self.u\_limit

 elif self.u < -self.u\_limit:

 self.u = -self.u\_limit

 # Backup values needed for the next iteration

 self.error\_p\_prev = self.err\_prev

 self.err\_prev = error

 return self.u

## MAŠINA STANJA

Procesiranje naredbe o željenoj tački, od čekanja naredbe, do kretanja i dostizanja željene tačke, implementirano je kroz mašinu stanja. Ova mašina ima tri stanja:

1. Čekanje naredbe,
2. Rotacija ka željenoj tački,
3. Pravolinijsko kretanje ka željenoj tački.

Graf sa slike Slika 8.2 predstavlja dozvoljene tranzicije stanja tokom izvršavanja naredbe:



Slika . - Graf tranzicije stanja

Ova mašina stanja vezana je za kretanje ka željenoj tački. Ukoliko bi bilo željeno vezati više naredbi, spoljna mašina stanja bi mogla biti implementirana, i ona bi izvršavala trenutno opisanu mašinu stanja kada je to potrebno.

## REZULTATI TESTA KRETANJA KA ZADATOJ TAČKI

U nastavku je dat primer kretanja iz tačke (0.0), ka tački (5,5), pri čemu je početna orijentacija robota ugao od 0 stepeni.



Slika .- Inicijalna pozicija - čekanje naredbe

Slika . – Konzola: ulazak u stanje rotacije



Slika . - Završetak rotacije



Slika .- Završetak pravolinijskog kretanja



Slika .- Konzola: završetak pravolonijskog kretanja



Slika . - Rezultati kretanja trajektorijom kvadranog oblika

## GRAFIČKI PRIKAZ TRAJEKTORIJE KRETANJA ROBOTA

Zadavane su komande tako da se robot kreće trajektorijom kvadratnog oblika, sa početkom u (0, 0), i dostižući sledeće tačke: (5, 0), (5, 5), (0, 5), i na kraju opet početnu(0, 0). Rezultati kretanja prikazani su na Slika 8.8, gde je prikazana trajektorija u XY koordinatnom sistemu.

# LITERATURA

|  |  |
| --- | --- |
| [1] | Object Management Group®. (2021) OMG - Standards Development Organization. [Online]. HYPERLINK "https://www.omg.org/spec/IDL/" https://www.omg.org/spec/IDL/  |
| [2] | David Gossow, Josh Faust, William Woodall Dave Hershberger. (2018) ROS Wiki. [Online]. HYPERLINK "http://wiki.ros.org/rviz" http://wiki.ros.org/rviz  |
| [3] | Dorian Scholz, Aaron Blasdel Thomas. (2016, Aug.) ROS Wiki. [Online]. HYPERLINK "http://wiki.ros.org/rqt" http://wiki.ros.org/rqt  |
| [4] | The Qt Company. (2021) Qt. [Online]. HYPERLINK "https://www.qt.io/" https://www.qt.io/  |
| [5] | Aaron Blasdel Dirk Thomas. (2018, Sep.) ROS Wiki. [Online]. HYPERLINK "http://wiki.ros.org/rqt\_graph" http://wiki.ros.org/rqt\_graph  |
| [6] | Dirk Thomas Dorian Scholz. (2018, May) ROS Wiki. [Online]. HYPERLINK "http://wiki.ros.org/rqt\_plot" http://wiki.ros.org/rqt\_plot  |
| [7] | Eitan Marder-Eppstein, Wim Meeussen Tully Foote. (2017, Oct.) ROS.org/tf. [Online]. HYPERLINK "http://wiki.ros.org/tf" http://wiki.ros.org/tf  |
| [8] | Open Source Robotics Foundation. Gazebo. [Online]. HYPERLINK "https://gazebosim.org/" https://gazebosim.org/  |

**Kratka biografija:**

|  |  |
| --- | --- |
| 30000317 - Vujić Đorđe - Copy.png | **Đorđe Vujić** rođen je u Novom Sadu 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Adaptivno i napredno upravljanje odbranio je 2021. god.Kontakt: djordjevujic@yahoo.com |