Upravljanje svesmjernim robotima

Suhić, Domagoj

Master's thesis / Diplomski rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Department of Mathematics / Sveučilište Josipa Jurja Strossmayera u Osijeku, Odjel za matematiku**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:126:684535

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveuličišni diplomski studij matematike
Smjer: Matematika i računarstvo

Domagoj Suhić

Upravljanje svesmjernim robotima

Diplomski rad

Osijek, 2018.

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveuličišni diplomski studij matematike
Smjer: Matematika i računarstvo

Domagoj Suhić

Upravljanje svesmjernim robotima

Diplomski rad

Mentor: izv. prof. dr. sc. Domagoj Matijević

Osijek, 2018.

Sadržaj

Uvod

Roboti imaju sve veću prisutnost u svakodnevnom životu ljudi. Postoji niz tipova robota, ovisno o njihovoj funkciji. Tako postoje industrijski roboti čija je svrha automatiziranje poslova u proizvodnji koji su prije bili ručno obavljani. Osim njih, imamo i vojne robote, koji obavljaju širok spektar dužnosti vezanih uz ratovanje, od prijevoza resursa do izvršavanja napada na neprijatelja. Također postoje i roboti čija je primarna svrha zabavljanje ljudi. To su prvenstveno razni roboti u obliku kućnih ljubimaca ili humanoidni roboti. Tema upravljanja robotima je stoga zanimljiva ne samo iz matematičkog, odnosno računarskog, aspekta. Posebno će nam zanimljiv biti jedan podskup robota - **mobilni roboti**. Postoji nekoliko vrsta mobilnih robota ovisno o načinu na koji se kreću, no u ovom ćemo radu koristiti sljedeću definiciju mobilnog robota:

Definicija 1. *Neka je R rigidno tijelo na kotačima koje se kreće u horizontalnoj ravnini. Tada je R mobilni robot.*

Vjerojatno najzanimljivija, i najstroža u smislu zahtjeva, skupina mobilnih robota jesu autonomni mobilni roboti. Oni bi trebali biti sposobni sami donositi odluke o svom smjeru i brzini gibanja ovisno o cilju robota. Takvi roboti se često koriste u prostorima i uvjetima koji nisu prethodno poznati ili se mijenjaju za vrijeme rada robota. Zbog toga su sposobnosti određivanja položaja, navigacije i obilaženja prepreka ključni za takvog robota. Te sposobnosti se najčešće postižu korištenjem različtih senzora i algoritama. No, ni najbolji senzori i algoritmi neće biti od pretjerane pomoći ako robota u radu ograničava njegova struktura. Svakom su mobilnom robotu, pa tako i autonomnim robotima, od iznimne važnosti mobilnost i manevarska sposobnost. Stoga, roboti s mogućnošću kretanja u svim smjerovima imaju velik broj prednosti naspram robota koji nemaju tu mogućnost. Primjer upotrebe takvog robota je viličar koji se može kretati u svim smjerovima. Takvo svojstvo mu je vrlo korisno jer se uglavnom kreće u skučenim prostorima gdje se može biti teško ili nemoguće okretati. Još jedan dobar primjer su autonomni usisivači. Oni se kreću u prostorima koji su često prepuni prepreka pa im mogućnost kretanja u svim smjerovima omogućuje lakše praćenje rubova tih prepreka. U suštini, takvi roboti imaju značajne prednosti u bilo kakvom području djelovanja koje uključuje kretanje u uskim prostorima ili zahtijeva vrlo precizan smjer gibanja, poput naprimjer pirotehnike.

Robote s mogućnošću gibanja u svim smjerovima nazivamo **svesmjernim robotima**. U ovom ćemo radu obraditi temu upravljanja svesmjernim robotima. Prva dva poglavlja su posvećena teorijskim dijelom upravljanja svesmjernim robotima, a posljednje poglavlje praktičnom dijelu.

U prvom poglavlju ovog diplomskog rada ćemo se baviti problemom kinematike kod mobilnih robota općenito. Izvest ćemo uvjete gibanja robota ovisno o tipu, položaju i veličini njegovih kotača te ćemo predstaviti neke modele svesmjernih robota.

U drugom poglavlju ćemo obraditi sam problem upravljanja mobilnim robotima. Predstavit ćemo petlje povratne veze te PID regulatore. Na kraju ćemo pokazati diskretizaciju PID regulatora.

Posljednje poglavlje ovog rada posvetit ćemo implementaciji Omnibot-a, svesmjernog robota Odjela za Matematiku. Proći ćemo i kroz hardware i kroz software koji je korišten u njegovoj izradi.

1 Kinematika

Neovisno o kakvom mobilnom robotu je riječ, pod terminom "upravljanje robota" podrazumijevamo kretanje robota željenim smjerom i iznosom. Kako bismo to ostvarili, potrebno je poznavati kinematiku robota kojim upravljamo. Naime, jedini dijelovi robota koje možemo pomicati su motori i aktuatori. Stoga nam je potrebno poznavati veze između gibanja tih dijelova i gibanja samog robota. U ovom poglavlju ćemo obraditi reprezentiranje pozicije robota, kinematiku standardnog fiksnog kotača, kinematiku švedskog kotača te nekoliko primjera svesmjernih robota koji koriste švedske kotače.

1.1 Reprezentiranje pozicije robota

Neka je (X_I, Y_I) proizvoljna baza za ravninu koja definira referentni sustav s ishodištem u proizvoljnoj točki O. Taj sustav ćemo zvati **inercijskim referentnim sustavom**. Slično, neka je (X_R, Y_R) proizvoljna baza koja definira referentni sustav s ishodištem u proizvoljnoj točki P koja se nalazi na robotu. Taj sustav ćemo zvati **referentnim sustavom robota**. Ako je pozicija točke P u inercijskom referentnom sustavu dana s x i y te ako je θ kut između inercijskog i robotovog referentnog sustava, onda ćemo poziciju robota u inercijskom sustavu označavati s

$$\xi_I = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

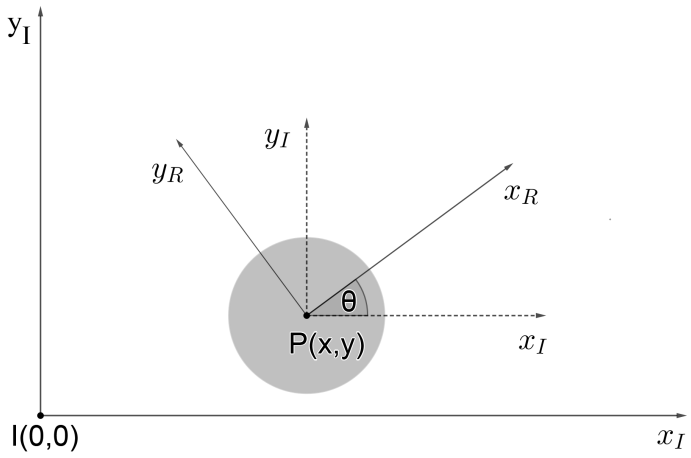
Ako pak sada označimo s

$$\dot{\xi}_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$$

brzine robota u inercijskom referentnom sustavu, tada brzine robota u robotovom referentom sustavu možemo dobiti primijenjivanjem matrice rotacije za kut θ.

$$\dot{\xi_R} = R(\theta)\dot{\xi_I} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} \tag{1}$$

Sada raspolažemo osnovnim terminima vezanim za brzinu i položaj mobilnog robota. Sljedeći korak je predstaviti i riješiti tzv. **problem povratne kinematike**.



Slika 1: Referentni sustavi

Definicija 2. *Neka je R mobilni robot, $\dot{\xi}_I$ brzina robota u inercijskom referentom sustavu, a neka je $\dot{\varphi} = \begin{bmatrix} \varphi_1 \\ \vdots \\ \varphi_n \end{bmatrix}$, $n \in \mathbb{N}$ vektor kutnih brzina kotača robota R. Matricu J reda $n \times 3$ za koju vrijedi:*

$$J\dot{\xi}_I = \dot{\varphi}$$

nazivamo matricom povratne kinematike robota R

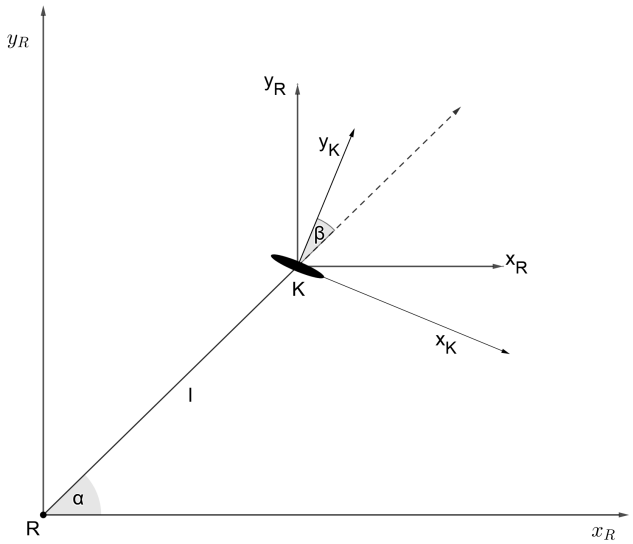
Problem povratne kinematike podrazumijeva pronalaženje matrice J. Taj problem ćemo riješavati u sljedećem potpoglavlju koristeći kinematiku kotača.

1.2 Kotači

U ovom ćemo potpoglavlju predstaviti kinematiku stadardnog fiksnog i švedskog kotača. Uvjete na gibanje robota koji koristi te kotače možemo tada dobiti kombiniranjem uvjeta na pojedine kotače.

1.2.1 Standardni fiksni kotač

Standaradni fiksni kotač je kotač koji se može gibati jedino naprijed/nazad po ravnini kotača te rotirati oko točke kontakta s ravninom podloge. Na slici 2 možemo vidjeti skicu robota i jednog fiksnog standardnog kotača koji mu pripada.



Slika 2: Fiksni standardni kotač

Središte robota se nalazi u točki R, a osi robotovog referentnog sustava su označene s X_R i Y_R. Središte kotača, K, se nalazi na udaljenosti l od R te kut koji zatvaraju $\overrightarrow{RK}$ i os X_R iznosi α. Otklon kotača iznosi β, a osi smjera u kojem se kotač okreće i njena normala su označene s X_K i Y_K.

Cilj nam je riješiti sljedeći problem: Iz poznatog gibanja robota, odnosno točke R, odrediti gibanje kotača, odnosno točke K.

Neka su s $\dot{\xi}_R = \begin{bmatrix} \dot{x_R} \\ \dot{y_R} \\ \dot{\theta} \end{bmatrix}$ označene brzine robota u robotovom referentnom sustavu. Za svaku od komponenti tog vektora ćemo raspisati utjecaj na gibanje kotača u terminima osi X_K i Y_K. Promotrimo prvo gibanje u smjeru osi X_K. Sa slike 3 je vidljivo kako je

$$\angle(X_K, X_R) = \angle(X_K, Y_K) - \angle(X_R, Y_K) = \frac{\pi}{2} - (\alpha + \beta) \tag{2}$$

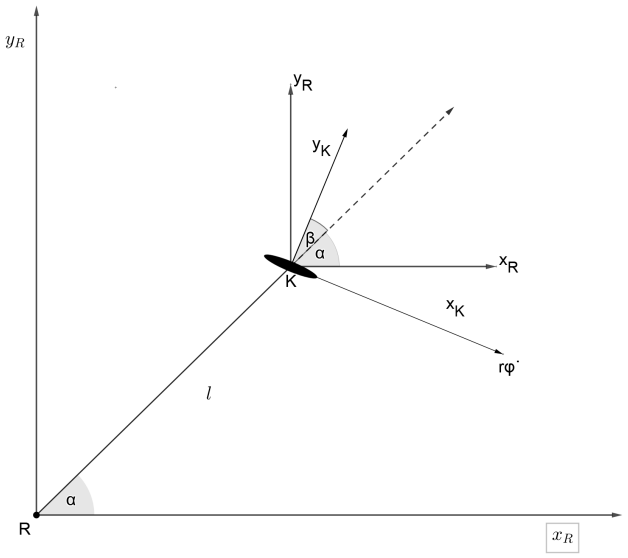
Iz (2) slijedi kako je iznos udijela $\dot{x_R}$ na gibanje kotača u smjeru X_K jednak

$$\dot{x_R} \cos(\frac{\pi}{2} - (\alpha + \beta)) = \dot{x_R} \sin(\alpha + \beta) \tag{3}$$

Koristeći isti postupak i činjenicu kako je

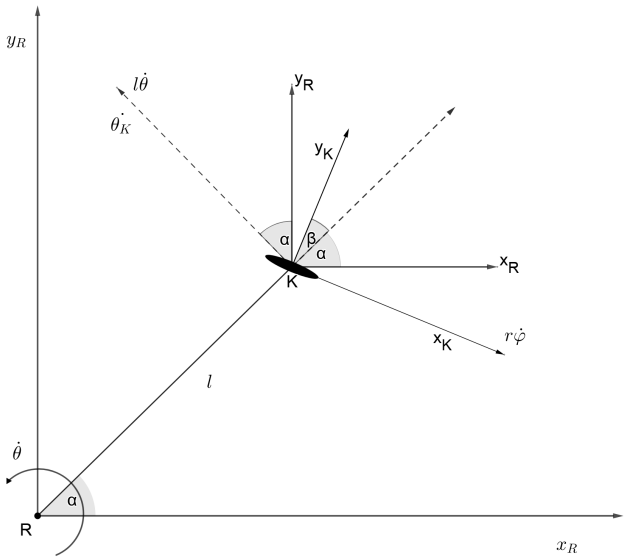
$$\angle(X_K, Y_R) = \angle(X_K, X_R) + \angle(X_R, Y_R) = \pi - (\alpha + \beta) \tag{4}$$

dobivamo kako je iznos udijela $\dot{y_R}$ na gibanje kotača u smjeru X_K jednak



Slika 3: Rastav vektora $\dot{x_R}$ u terminima osi referentnog sustava kotača

$$\dot{y_R}\cos(\pi - (\alpha + \beta)) = \dot{y_R}(-\cos(\alpha + \beta)) = -\dot{y_R}\cos(\alpha + \beta) \tag{5}$$



Slika 4: Kut između X_K i θ_K

Preostaje odrediti udio komponente $\dot{\theta}$. Kako je riječ o kružnom gibanju, poznato nam je kako će, pri rotaciji točke R kutnom brzinom $\dot{\theta}$, točka K imati brzinu $l\dot{\theta}$ u smjeru normale na $\vec{RK}$. Također, sa slike 4, jasno je vidljivo kako vrijedi

$$\angle(X_K, \theta_K) = \angle(X_K, X_R) + \alpha + \frac{\pi}{2} \overset{2}{=} \pi - \beta \tag{6}$$

Sada možemo zaključiti kako je iznos udijela $\dot{\theta}$ na gibanje kotača u smjeru X_K jednak

$$l\dot{\theta}\cos(\pi - \beta) = l\dot{\theta}(-\cos(\beta)) = -l\dot{\theta}\cos(\beta) \tag{7}$$

Konačno, iz (3), (5) i (7) možemo zaključiti kako je ukupno gibanje kotača u smjeru X_K jednako

$$\dot{x_R}\sin(\alpha + \beta) - \dot{y_R}\cos(\alpha + \beta) - l\dot{\theta}\cos(\beta) \tag{8}$$

Promotrimo sada gibanje u kotaču u smjeru osi Y_K. Koristeći već izračunate kuteve među osima, lako se pokaže kako ukupno gibanje kotača u smjeru Y_K iznosi

$$\dot{x_R}\cos(\alpha + \beta) + \dot{y_R}\sin(\alpha + \beta) + l\dot{\theta}\sin(\beta) \tag{9}$$

Izrazimo sada te vrijednosti u terminima kutne brzine kotača, $\dot{\varphi}$. Naime, kako je riječ o standardnom fiksnom kotaču, brzina u smjeru X_K iznosit će $r\dot{\varphi}$, gdje je r polumjer kotača. S druge strane, kako kotač ne može kliziti u smjeru okomitom na smjer prema kojem se kotač okreće, brzina u smjeru Y_K mora iznositi 0. Stoga, imamo sljedeće uvjete:

$$\begin{bmatrix} \sin(\alpha+\beta) & -\cos(\alpha+\beta) & -l\cos(\beta) \\ \cos(\alpha+\beta) & \sin(\alpha+\beta) & l\sin(\beta) \end{bmatrix} \begin{bmatrix} \dot{x_R} \\ \dot{y_R} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} r\dot{\varphi} \\ 0 \end{bmatrix} \tag{10}$$

Konačno, ako nas zanima gibanje kotača u inercijskom referentnom sustavu, koristeći 1 dobivamo

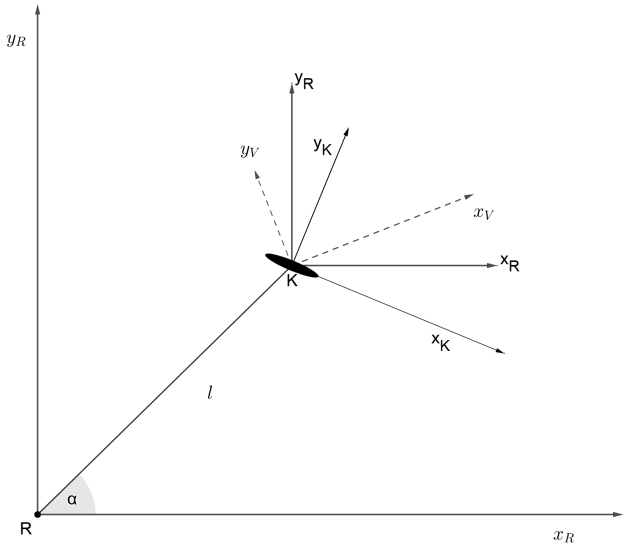
$$\begin{bmatrix} \sin(\alpha+\beta) & -\cos(\alpha+\beta) & -l\cos(\beta) \\ \cos(\alpha+\beta) & \sin(\alpha+\beta) & l\sin(\beta) \end{bmatrix} R(\theta)\dot{\xi}_I = \begin{bmatrix} r\dot{\varphi} \\ 0 \end{bmatrix} \tag{11}$$

Prvi uvjet još nazivamo **uvjetom kotrljanja**, a drugi uvjet **uvjetom klizanja**.

1.2.2 Švedski kotač

Švedski kotač, naziv je za tip kotača koji nalikuje fiksnom standardnom kotaču s pasivnim valjcima na rubu kotača postavljenim tako da kut između ravnine glavnog

Slika 5: Mecanum

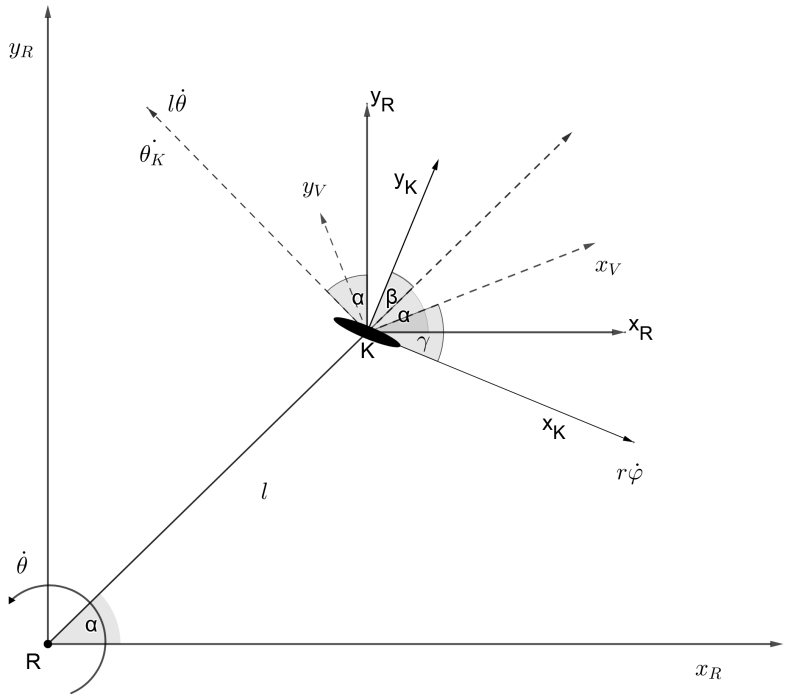


Slika 6: Švedski kotač

kotača i osi rotacije valjaka iznosi γ. Švedski kotač za kojeg vrijedi $\gamma = \frac{\pi}{4}$ još nazivamo i Mecanum.[3]

Na slici 6 možemo vidjeti skicu robota s jednim švedskim kotačem. U odnosu na sliku 2 imamo dodane osi X_V i Y_V koje predstavljaju os rotacije valjaka te njenu normalu. Također, s γ je označen kut između osi X_K i X_V. Sada ćemo provesti sličan postupak kao i za standardni fiksni kotač. Međutim, ovaj puta nećemo raspisivati utjecaje na gibanje kotača u terminima osi X_K i Y_K, već u terminima osi X_V i Y_V. Referentni sustav koji je definiran tim osima nazovimo referentni sustav valjaka. Započnimo od gibanja u smjeru osi X_V.

Sa slike 7 je vidljivo kako je



Slika 7: Rastav vektora $\dot{x_R}$ u terminima osi referentnog sustava valjaka

$$\begin{aligned}\angle(X_R, X_V) &= \angle(X_R, Y_R) - \angle(Y_T, Y_R) - \angle(X_V, Y_T) \\ &= \frac{\pi}{2} - (\frac{\pi}{2} - (\alpha + \beta)) - (\frac{\pi}{2} - \gamma) \\ &= \alpha + \beta + \gamma - \frac{\pi}{2} \end{aligned} \tag{12}$$

Iz (12) lako slijedi kako je iznos udijela $\dot{x_R}$ na gibanje kotača u smjeru X_V jednak

$$\dot{x_R}\cos(\alpha + \beta + \gamma - \frac{\pi}{2}) = \dot{x_R}\sin(\alpha + \beta + \gamma) \tag{13}$$

Koristeći isti postupak i činjenicu kako je

$$\angle(Y_R, X_V) = \angle(X_V, X_R) + \angle(X_R, Y_R) = \pi - (\alpha + \beta + \gamma) \tag{14}$$

dobivamo kako je iznos udijela $\dot{y_R}$ na gibanje kotača u smjeru X_V jednak

$$\begin{aligned}\dot{y_R}\cos(\pi - (\alpha + \beta + \gamma)) &= \dot{y_R}(-\cos(\alpha + \beta + \gamma)) \\ &= -\dot{y_R}\cos(\alpha + \beta + \gamma) \end{aligned} \tag{15}$$

Iznos udijela $\dot{\theta}$ dobijemo na isti način kao i kod standardnog fiksnog kotača te on iznosi

$$\begin{aligned} l\dot{\theta}\cos(\alpha+\frac{\pi}{2}-(\alpha+\beta+\gamma-\frac{\pi}{2})) &= l\dot{\theta}\cos(\pi-(\beta+\gamma)) \\ &= -l\dot{\theta}\cos(\beta+\gamma) \end{aligned} \tag{16}$$

Iz (13), (15) i (16) sada slijedi kako nam je ukupno gibanje kotača u smjeru X_V jednako

$$\dot{x_R}\sin(\alpha+\beta+\gamma) - \dot{y_R}\cos(\alpha+\beta+\gamma) - l\dot{\theta}\cos(\beta+\gamma) \tag{17}$$

Sličnim postupkom za smjer Y_V dobivamo ukupno gibanje u iznosu od

$$\dot{x_R}\cos(\alpha+\beta+\gamma) + \dot{y_R}\sin(\alpha+\beta+\gamma) + l\dot{\theta}\sin(\beta+\gamma) \tag{18}$$

Izrazimo sada (17) i (18) u terminima kutnih brzina kotača, $\dot{\varphi}$, i kutnih brzina pasivnih valjaka, $\dot{\varphi_V}$. Kako se kotač vrti u smjeru X_K, a valjci u smjeru Y_V, brzina kotača u smjeru X_V iznosit će

$$r\dot{\varphi}\cos(\angle(X_K,X_V)) + r_V\dot{\varphi_V}\cos(\angle(Y_V,X_V)) = r\dot{\varphi}\cos(\gamma) \tag{19}$$

gdje je r_V polumjer pasivnih valjaka. S druge strane, brzina kotača u smjeru Y_V iznosi

$$r\dot{\varphi}\cos(\angle(X_K,Y_V)) + r_V\dot{\varphi_V}\cos(\angle(Y_V,Y_V)) = r\dot{\varphi}\sin(\gamma) + r_V\dot{\varphi_V} \tag{20}$$

Sada imamo sljedeće uvjete na gibanje robota:

$$\begin{bmatrix} \sin(\alpha+\beta+\gamma) & -\cos(\alpha+\beta+\gamma) & -l\cos(\beta+\gamma) \\ \cos(\alpha+\beta+\gamma) & \sin(\alpha+\beta+\gamma) & l\sin(\beta+\gamma) \end{bmatrix} R(\theta)\dot{\xi}_I = \begin{bmatrix} r\dot{\varphi}\cos(\gamma) \\ r\dot{\varphi}\sin(\gamma) + r_V\dot{\varphi_V} \end{bmatrix} \tag{21}$$

Primijetimo kako u uvjetu klizanja, neovisno o tome koje vrijednosti odaberemo za $R(\theta)\dot{\xi}_I$ i $\dot{\varphi}$, uvijek možemo pronaći takav $\dot{\varphi_V}$ da uvjet bude ispunjen. Upravo zato su nam švedski kotači zanimljivi jer zbog tog svojstva možemo pomoću njih konstruirati svesmjerne robote. U sljedećem ćemo potpoglavlju stoga pri konstrukciji svesmjernih robota sa švedskim kotačima koristiti samo uvjete kotrljanja.

1.3 Svesmjerni roboti

U ovom ćemo potpoglavlju predstaviti nekoliko konfiguracija svesmjernih robota na bazi švedskih kotača te riješiti njihove probleme povratne kinematike.

1.3.1 Svesmjerni roboti s tri kotača

Najednostavniji primjer svesmjernog robota možemo dobiti korištenjem tri švedska kotača za koje je $\gamma = 0$. Svi kotači su iste veličine, imaju otklon $\beta = 0$ te su jednako udaljeni od središta robota. Također, svi kotači su međusobno jednako udaljeni. Vrijednosti α, β, l, r za svaki kotač su zapisane u tablici 1.

i	α_i	β_i	γ_i	r_i	l_i
1	0	0	0	r	l
2	$\frac{2\pi}{3}$	0	0	r	l
3	$\frac{4\pi}{3}$	0	0	r	l

Tablica 1: Parametri za svesmjernog robota s 3 kotača

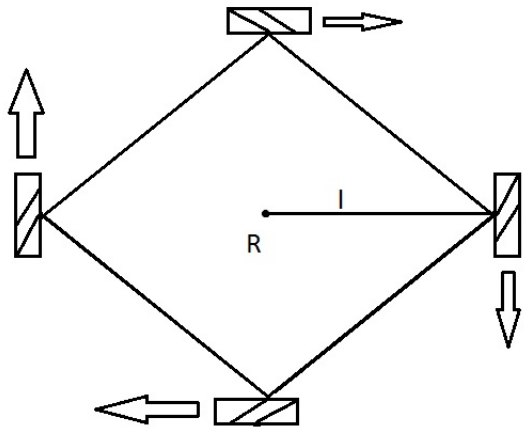
Uvrstimo li parametre za svaki kotač u uvjet kotrljanja, dobivamo matricu povratne kinematike, J za ovog robota.

$$\begin{aligned} \begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \end{bmatrix} &= \frac{1}{r} \begin{bmatrix} 0 & -1 & -l \\ \frac{\sqrt{3}}{2} & \frac{1}{2} & -l \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & -l \end{bmatrix} R(\theta) \begin{bmatrix} \dot{x_I} \\ \dot{y_I} \\ \dot{\theta} \end{bmatrix} \\ &= J \begin{bmatrix} \dot{x_I} \\ \dot{y_I} \\ \dot{\theta} \end{bmatrix} \end{aligned} \tag{22}$$

1.3.2 Svesmjerni roboti sa četiri kotača

Iako svesmjernost možemo postići, kao što smo i pokazali, sa samo tri kotača, korištenjem većeg broja kotača povećavamo stabilnost i nosivost robota. Sada ćemo predstaviti dva modela svesmjernih robota na četiri švedska kotača s parametrom $\gamma = \frac{\pi}{4}$. Prvi model je generalizacija prethodnog modela i može se vidjeti na slici 8. Vrijednosti α, β, l, r za svaki kotač su zapisane u tablici 2.

Uvrstimo li parametre za svaki kotač u uvjet kotrljanja (21), dobivamo matricu povratne kinematike, J za ovog robota.



Slika 8: Svesmjerni robot sa 4 švedska kotača

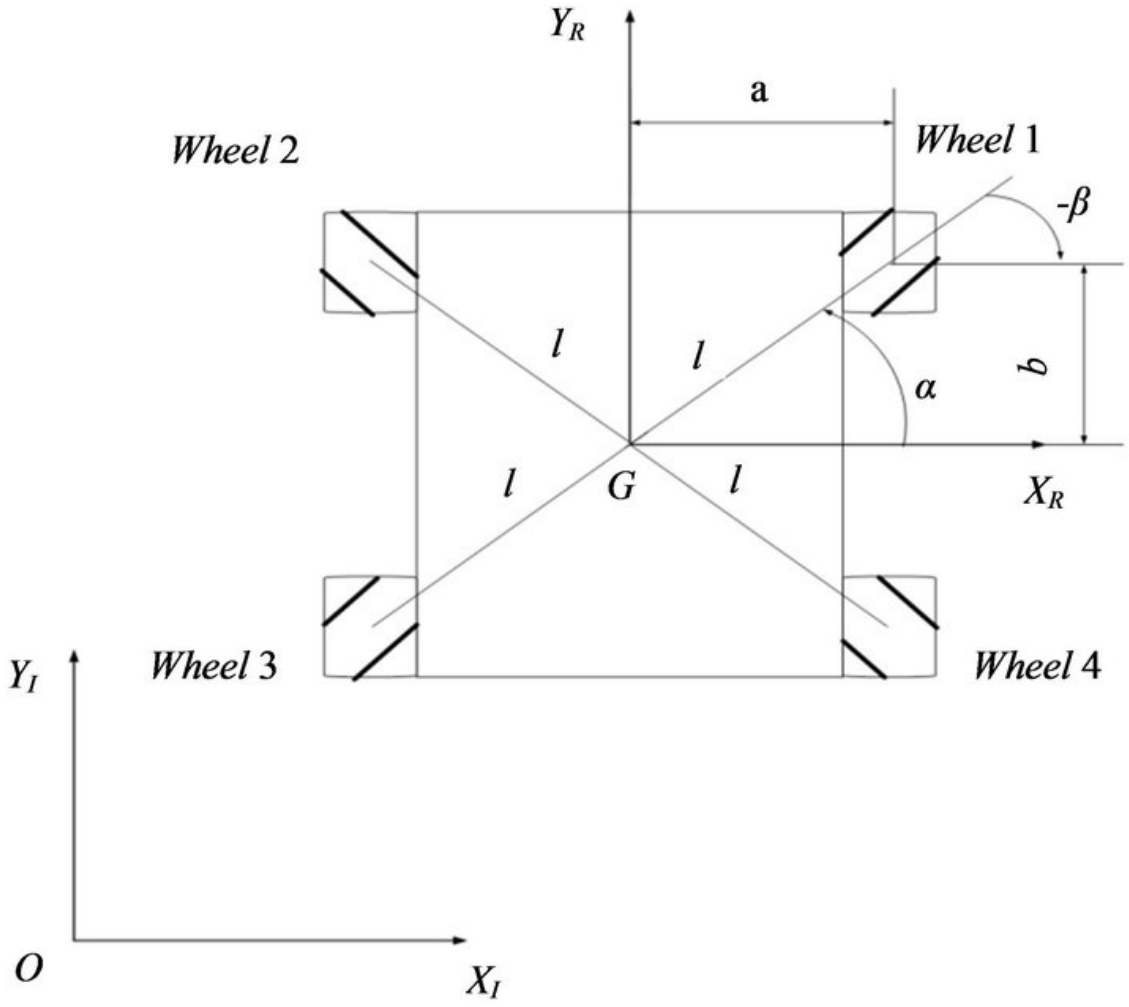
i	α_i	β_i	γ_i	r_i	l_i
1	0	0	$\frac{\pi}{4}$	r	l
2	$\frac{\pi}{2}$	0	$\frac{\pi}{4}$	r	l
3	π	0	$\frac{\pi}{4}$	r	l
4	$\frac{3\pi}{2}$	0	$\frac{\pi}{4}$	r	l

Tablica 2: Parametri za svesmjernog robota sa 4 kotača

$$\begin{aligned}\begin{bmatrix}\dot{\varphi}_1\\ \dot{\varphi}_2\\ \dot{\varphi}_3\\ \dot{\varphi}_4\end{bmatrix} &= \frac{1}{r}\begin{bmatrix}1 & -1 & -l\\ 1 & 1 & -l\\ -1 & 1 & -l\\ -1 & -1 & -l\end{bmatrix} R(\theta)\begin{bmatrix}\dot{x_I}\\ \dot{y_I}\\ \dot{\theta}\end{bmatrix}\\ &= J\begin{bmatrix}\dot{x_I}\\ \dot{y_I}\\ \dot{\theta}\end{bmatrix}\end{aligned} \tag{23}$$

Drugi model je baziran na četverokotačnom svesmjernom robotu predstavljenim u [3]. Model možemo vidjeti na slici 9 Vrijednosti α, β, l, r za svaki kotač su zapisane u tablici 3, gdje $\zeta = \tan^{-1}(\frac{b}{a})$.

Uvrstimo li parametre za svaki kotač u uvjet kotrljanja u (21), dobivamo matricu povratne kinematike, J za ovog robota.



Slika 9: Drugi model svesmjernog robota sa 4 švedska kotača[3]

$$\begin{bmatrix} \dot{\varphi}_1 \\ \dot{\varphi}_2 \\ \dot{\varphi}_3 \\ \dot{\varphi}_4 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} -1 & -1 & -l\sqrt{2}\sin(\frac{\pi}{4}-\zeta) \\ -1 & 1 & -l\sqrt{2}\sin(\frac{\pi}{4}-\zeta) \\ 1 & 1 & -l\sqrt{2}\sin(\frac{\pi}{4}-\zeta) \\ 1 & -1 & -l\sqrt{2}\sin(\frac{\pi}{4}-\zeta) \end{bmatrix} R(\theta) \begin{bmatrix} \dot{x_I} \\ \dot{y_I} \\ \dot{\theta} \end{bmatrix}$$
$$= J \begin{bmatrix} \dot{x_I} \\ \dot{y_I} \\ \dot{\theta} \end{bmatrix} \tag{24}$$

i	α_i	β_i	γ_i	r_i	l_i
1	ζ	$-\zeta$	$\left(\frac{\pi}{2}+\frac{\pi}{4}\right)$	r	l
2	$\pi-\zeta$	ζ	$-\left(\frac{\pi}{2}+\frac{\pi}{4}\right)$	r	l
3	$\pi+\zeta$	$-\zeta$	$\left(\frac{\pi}{2}+\frac{\pi}{4}\right)$	r	l
4	$2\pi-\zeta$	ζ	$-\left(\frac{\pi}{2}+\frac{\pi}{4}\right)$	r	l

Tablica 3: Parametri za drugi svesmjerni robot sa 4 kotača

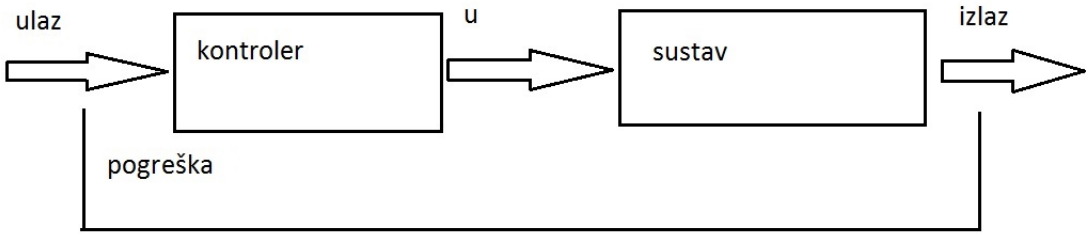
2 Upravljanje motorima

Samo poznavanje kinematike mobilnog robota nije dovoljno kako bismo mogli upravljati njime. Naime, ne možemo mobilnim komponentama robota (aktuatorima i motorima) jednostavno "reći" ili zadati brzinu kojom želimo da se gibaju. Oni su u konačnici elektroničke komponente i jedini način na koji možemo njima upravljati jest manipuliranjem njihove ulazne električne struje. Naivno rješenje problema upravljanja motorom je potpuno parametriziranje sustava, odnosno točnog određivanja odnosa napona ulazne električne struje u motor i momenta sile koji se tada stvori. Dodatno, bilo bi potrebno odrediti koeficijente trenja za svaku podlogu po kojoj bi se kotač kretao i niz drugih parametara. Već na prvi pogled je jasno kako takav pristup nije održiv. Uz to što bi određivanje parametara potencijalno oduzelo veliku količinu vremena, dodatan je problem to što bi se pri bilo kakvoj promijeni uvjeta, poput zamijene motora ili promijene podloge, moralo nanovo određivati parametre sustava. Stoga nam je potrebno neko drugo rješenje.

2.1 Petlja povratne veze

Petlja povratne veze (engl. **feedback loop**) je situacija u kojoj su dva ili više dinamička sustava povezana na način da utječu jedno na drugo. Dinamičkim sustavom smatramo sustav čije se ponašanje mijenja s vremenom, često kao odgovor na vanjske podražaje.[2] Na slici 10 možemo vidjeti skicu povratne veze. Moguće je koristiti petlje povratne veze u svrhu upravljanja motorima. Naime, ako kao ulaz na slici 10 stavimo napon električne struje koja ulazi u motor, a kao izlaz izmjerenu brzinu motora, tada imamo petlju povratne veze koju možemo kontrolirati kako bismo dostigli željenu brzinu.

Sada ovisno o grešci $e = v_{zeljena} - v_{izmjerena}$ i prethodnoj vrijednosti ulaza, možemo izračunati novu vrijednost ulaza, u, koristeći tzv. funkciju prijenosa (engl.



Slika 10: Petlja povratne veze

transfer function). Takve sustave nazivamo kontrolerima. Neka od ključnih svojstava koje želimo imati kod kontrolera su stabilnost, odnosno da male smetnje dovode do malih pogrešaka, te robustnost. Robustnošću smatramo sposobnost dostizanja željene vrijednosti sa različitim parametrima sustava. Primjer može biti mobilni robot koji postiže željenu brzinu sa različitim snagama motora. Postoji širok spektar različitih kontrolera baziranih na petljama povratnih veza. U sljedećem potpoglavlju ćemo detaljnije pogledati jedan od njih - PID regulator.

2.2 PID regulatori

PID je akronim za "proportional, integral, and derivative".[1] Dakle, PID regulator je kontroler baziran na petlji povratnih veza čija se funkcija prijenosa sastoji od tri elementa:

- P element: Proporcionalan je sadašnjoj grešci, tj. grešci u trenutku t.
- I element: Proporcionalan je integralu greške do trenutka t, što se može interpretirati kao akumulacija prošlih greški.
- D element: Proporcionalan je derivaciji greške u trenutku t, što se može interpretirati kao predviđanje buduće greške.[1]

Ako sa k_P, k_I i k_D označimo koeficijente proporcionalnosti P, I i D elemenata, tada funkciju prijenosa PID regulatora možemo zapisati na sljedeći način:

$$u(t) = k_P e(t) + k_I \int_0^t e(T)dT + k_D \frac{de(t)}{dt} \tag{25}$$

Prikladne vrijednosti koeficijenata proporcionalnosti možemo dobiti korištenjem jedne od mnogih metoda za tzv. "tuning". To je bitan proces jer lošim izborom koeficijenata možemo dobiti oscilirajući sustav ili sustav čiji izlaz ne teži željenoj vrijednosti.

Ako želimo implementirati PID regulator u mikrokontroleru za potrebe upravljanja motorom mobilnog robota, potrebno je diskretizirati ga. Naime, moći ćemo računati novu vrijednost funkcije prijenosa samo u nekim diskretnim trenutcima t_k, $k \in \mathbb{N}$. U tu svrhu, integral aproksimiramo sumom, a derivaciju konačnim diferencijama. Tada za određivanje vrijednosti funkcije prijenosa u trenutku t_{k+1} koristimo sljedeći algoritam:

Algoritam 1. *(Računanje vrijednosti funkcije prijenosa diskretnog PID regulatora)*

1. *Inicijaliziramo vrijednosti akumulacije grešaka i stare greške*

$$E, e_{old} = 0$$

2. *U beskonačnoj petlji provodimo preostale korake algoritma, gdje je e izmjerena greška u trenutnoj iteraciji, a Δt vremenski interval od zadnjeg računanja vrijednosti funkcije prijenosa:*

3. $E = E + e\Delta t$

4. $\dot{e} = \frac{e - e_{old}}{\Delta t}$

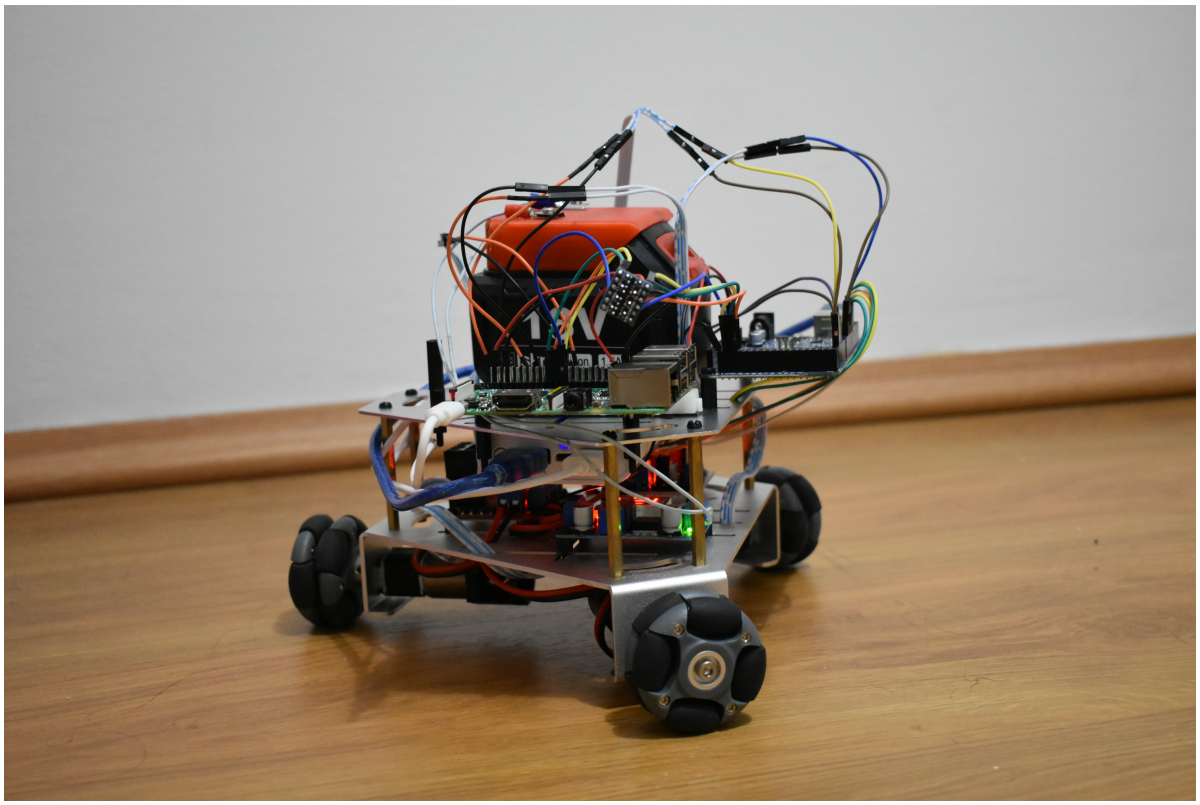
5. $u = k_P e + k_I E + k_D \dot{e}$

6. $e_{old} = e$

Sada imamo dovoljno teorijskog znanja kako bismo mogli modelirati i konstruirati svesmjernog robota. U sljedećem poglavlju ćemo predstaviti jednu implementaciju svesmjernog robota.

3 Omnibot

U ovom ćemo poglavlju predstaviti Omnibot-a. Omnibot je svesmjerni mobilni robot na 3 švedska kotača i ima istu kinematiku kao i robot iz potpoglavlja 1.3.1. Napravljen je u sklopu kolegija Ugrađeni sustavi. Prvo ćemo predstaviti elektroničke komponente koje ga sačinjavaju, a zatim ćemo se posvetiti proučavanju implementacije kontrolera.



Slika 11: Omnibot

3.1 Hardware

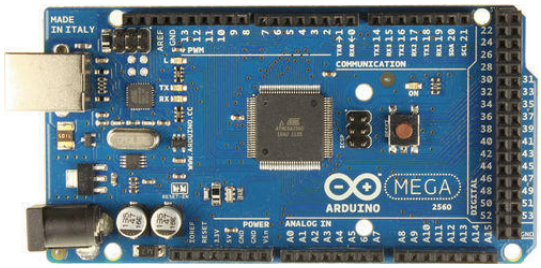
Omnibot se satoji od trokutastog okvira podijeljenog na dvije razine. Robot ima tri švedska kotača s parametrom $\gamma = 0$, svaki su nalazi pri jednom od vrhova trokuta. Kotači su mu spojeni na 12V DC motore, koji su pak spojeni na H-bridge komponente pomoću kojih se jednostavno može obrnuti polaritet napona i time mijenjati smjer okretanja motora. Njih napaja 18V baterija spojena na konverter napona kako bi se isti spustio na prikladnih 12V. Svaki motor također posjeduje

wheel encoder - komponenta pomoću koje se može detektirati pomak, odnosno brzina motora. Svi motori i *encoderi* su spojeni s Arduinom koji je pak spojen s Raspberry Pi-jem.

3.1.1 Arduino

Arduino je mikrokontroler - minijaturno računalo koje se nalazi na jednoj integriranoj pločici. Model koji se koristi na Omnibot-u je Mega2560. On dolazi s nizom ulazni i izlaznih *pin*-ova pomoću kojih se može upravljati elektroničkim komponentama. Unutar Omnibot-a, Arduino radi baš to. Na njemu se nalazi implementacija PID regulatora koji računa napon te ga šalje prema motorima. Također, očitava vrijednosti sa *encoder*-a u svrhu aproksimiranja trenutne brzine motora. Sa Raspberry Pi-jem je spojen putem *pin*-ova namijenjenih za komunikaciju SPI protokolom te na taj način mogu izmijenjivati poruke.

3.1.2 Raspberry Pi



Slika 12: Arduino Mega2560

Slika 13: Raspberry Pi 3

Raspberry Pi je kompletno računalo izgrađeno na jednoj pločici. Ujedno ima i nekoliko *pin*-ova za upravljanje elektroničkim komponentama. Model koji se koristi na Omnibot-u je Raspberry Pi 3. Na njemu smo instalirali Raspbian operativni sustav, Raspberry inačicu Debian sustava. Raspberry funkcionira kao glavni kontroler. On prima željene brzine od korisnika te ih pretvara u prikladne kutne brzine i zatim putem SPI protokola šalje te brzine Arduinu.

3.2 Software

Naposlijetku, pogledajmo implementaciju Omnibot-a sa strane *software*-a. Generalno, tok je sljedeći:

1. Raspberry Pi i uređaj pomoću kojega se planira upravljati robotom (računalo ili mobilni telefon) se spajaju na istu internetsku konekciju.

2. Na Raspberry-ju se pokreće Python skripta koja otvara HTTP poslužitelj na IP adresi Raspberry Pi-ja.

3. Korisnik odlazi na IP adresu Raspberry Pi-ja te dobiva sučelje pomoću kojega može unijeti željenu brzinu.

4. Raspberry očita dobivnu vrijednost, pretvara je u prikladne kutne brzine te putem SPI protokola šalje iste Arduinu.

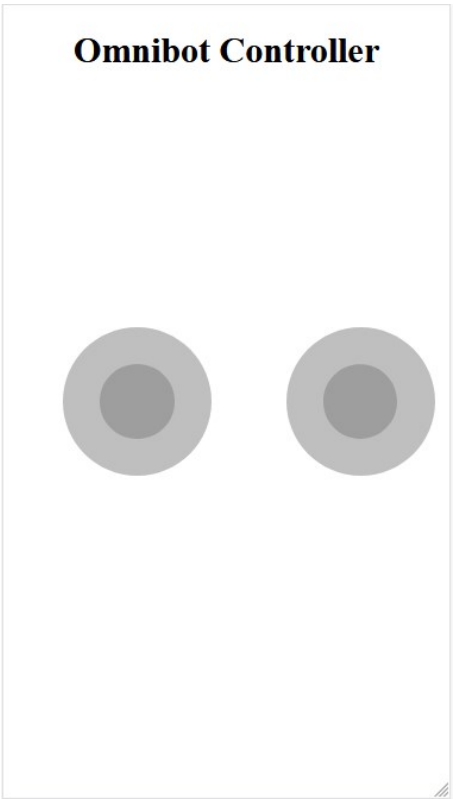
5. Arduino očita kutne brzine putem SPI protokola te koristeći PID regulator računa napon koji treba poslati na motor i šalje ga.

Sada ćemo detaljnije obraditi pojedine dijelove ovoga toka.

3.2.1 Http Server

Na Raspberry-ju se pokrene HTTP server sa 2 *endpoint*-a. Pri zahtjevima HTTP GET metode, poslužitelj šalje .html i .js datoteke koje čine korisničko sučelje za upravljanje Omnibot-om koje sadrži dva upravljača (slika 14). Lijevi određuje $\dot{x_R}$ i $\dot{y_R}$, a desni $\dot{\theta}$. Oko središta svakog upravljača je implementirana zona unutar koje se smatra da upravljač nije pomaknut. Upravljače se može pomicati ili mišem ili dodirom, ovisno o uređaju s kojeg se pristupa sučelju.

Pri zahtjevima HTTP POST metode, poslužitelj očitava poslane brzine. No, poslužitelj se izvršava unutar posebne niti izvršanja. Zbog tog razloga ne možemo odmah preračunati brzine i početi ih slati Arduinu. Naime, moguće je da u tom trenutku već komuniciramo s Arduinom. Zbog načina na koji SPI protokol funkcionira, počeli bismo Arduinu slati krive brzine i bilo bi se teško vratiti u normalno stanje bez ponovnog pokretanja Arduina. Stoga, poslužitelj poziva *callback* metodu koja mu je proslijeđena iz glavne niti izvršavanja pri inicijalizaciji te joj šalje primljene brzine kao parametre. Odluka o tome što će se raditi s brzinama sada više neće biti u rukama poslužitelja. S druge strane, i klijent i poslužitelj su asinkroni. Kako



Slika 14: Korisničko sučelje Omnibot-a

bismo spriječili slanje stotina zahtjeva prema poslužitelju svake sekunde, na klijentu je implementirano blokiranje slanja novih zahtjeva dok se za prijašnji zahtjev nije dobio odgovor od poslužitelja. Ovo pravilo ima jednu iznimku, a to je pri resetiranju upravljača. Ako se zahtjev za resetiranje, odnosno zaustavljanje robota, blokira, ponovno šaljemo zahtjev čim dobijemo odgovor od poslužitelja za prijašnji zahtjev.

Kako svako otvaranje HTTP konekcije stvara određeni *overhead*, a zahtjev s novim brzinama se može slati i desetak puta u sekundi, problem se mitigira ostavljanjem konekcije otvorenom.

```
1  var onDone;
2  var size = Math.min(window.innerWidth / 3, window.innerHeight / 2);
3
4  var left = nipplejs.create({
5    zone: document.getElementById('left'),
6    mode: 'static',
7    position: {left: '30%', top: '50%'},
8    color: 'black',
9    size: size
10 });
```

```
11
12 var right = nipplejs.create({
13   zone: document.getElementById('right'),
14   mode: 'static',
15   position: {left: '80%', top: '50%'},
16   color: 'black',
17   size: size
18 });
19
20 var positions = {
21   left: {
22     x: 0,
23     y: 0
24   },
25   right: {
26     x: 0,
27     y: 0
28   }
29 }
30
31 var locked = false;
32 var tol = 0.15;
33
34 left.on('move', onMove);
35 right.on('move', onMove);
36 left.on('end', onReset);
37 right.on('end', onReset);
38
39 function onMove(evt, data) {
40   let key = evt.target.id ? 'right' : 'left';
41   let x = 2.0 * Math.cos(data.angle.radian) * data.distance / size;
42   let y = 2.0 * Math.sin(data.angle.radian) * data.distance / size;
43
44   if (x*x + y*y < tol*tol) {
45     positions[key].x = 0.0;
46     positions[key].y = 0.0;
47   }
48   else {
49     positions[key].x = x
50     positions[key].y = y;
51   }
52
53   sendPosition();
54 }
55
56 function onReset(evt, data) {
57   let key = evt.target.id ? 'right' : 'left';
58   positions[key].x = 0.0;
59   positions[key].y = 0.0;
```

```
60
61    sendPosition(true);
62  }
63
64  function sendPosition(force) {
65    if (!locked) {
66      onDone = null;
67      locked = true;
68      let xhttp = new XMLHttpRequest();
69      xhttp.onreadystatechange = function() {
70        if (this.readyState === 4) {
71          locked = false;
72          if (onDone) {
73            onDone();
74          }
75        }
76      }
77      xhttp.open('POST', '', true);
78      xhttp.setRequestHeader('Content-type', 'application/json');
79      xhttp.send(JSON.stringify(positions));
80    }
81    else if (force) {
82      onDone = function() {
83        onDone = null;
84        sendPosition();
85      };
86    }
87  }
```

Listing 1: Implementacija klijenta bazirana na nippleJS biblioteci

3.2.2 Obrada podataka

Obrada podataka se odvija u glavnoj niti izvršavanja. Kada nam poslužitelj proslijedi nove brzine, brzine pretvaramo u kutne brzine kotača te ih spremamo sa strane. Unutar beskonačne petlje, provjeravamo imamo li brzine koje trebamo poslati te, ako ih imamo, je li komunikacijski kanal s Arduinom kroz SPI protokol slobodan. Ako je sve u redu, šaljemo brzine. Bitno je napomenuti kako nije bezrazložno to što brzine šaljemo samo ako ih je u međuvremenu poslužitelj proslijedio. Naime, svaki puta kada putem SPI protokola Arduinu šaljemo nove brzine, greške na PID regulatoru se resetiraju. Kada bismo u svakoj iteraciji petlje slali brzine, imali bismo vrlo *trzavo* gibanje robota.

Također, prije slanja brzina, moramo ih validirati. Svaki motor ima ograničenja na maksimalnu i minimalnu brzinu. Zbog toga je moguće, čak i ako imamo sves-

mjerni robot, dogoditi se slučaj da se ne možemo kretati željenim brzinama. Zbog toga je potrebno prilaogidit iznose brzina tako da smjer ostane isti, a da robot može postići te brzine.

3.2.3 Kontroler

Nakon što putem SPI protokola Arduino dobije nove željene brzine, proslijeđuje ih tzv. upraviteljima. Imamo tri upravitelja, po jedan za svaki motor. Svaki upravitelj također ima svoj PID regulator. Uloga upravitelja je slanje novo izračunatog napona motoru. Glavna petlja programa ažurira brojeve okretaja na *encoderima*, računa količinu vremena koja je protekla od zadnjeg izvršavanja petlje te poziva metode izvršavanja upravitelja.

3.2.4 Zaključak

Robot kojeg smo ovdje obradili je bio prezentiran na Osijek Mini Maker Faire-u kao dio izlaganja "MathBot2 strikes back". Tamo je postigao poprilično značajnu razinu pažnje, što od strane onih koje je zanimalo kako je robot napravljen, a što od onih koji su samo htjeli njime upravljati. Time je predstavljen još jedan aspekt iz kojeg je tema upravljanja robotima zanimljiva - edukacijski aspekt. U razvoju Omnibot-a pomogli su izv. prof. dr. sc. Domagoj Matijević, vanjski suradnik Matija Bogdanović i asistent Jurica Maltar.

Literatura

[1] M. Araki, *PID Control*, Control Systems, Robotics, and Automation **2** (2009), 58–79.

[2] K.J. Åström, R.M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*, Princeton University Press, Princeton, 2010.

[3] L. Lin, and H. Shih, *Modeling and Adaptive Control of an Omni-Mecanum-Wheeled Robot*, Intelligent Control and Automation **4** (2013), 166–179.

[4] R. Siegwart, I.R. Nourbakhsh, D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, MIT Press, London, 2011.

Sažetak

U ovom radu bavili smo se upravljanjem svesmjernih robota. Prvo smo se posvetili teorijskim aspektima upravljanja mobilnim robotima. Izrazili smo uvjete kotrljanja i uvjete klizanja za fiksni standardni te švedski kotač. Nakon toga smo predstavili problem povratne kinematike te ga riješili za tri primjera svesmjernih robota. Zatim smo se bavili problemom upravljanja mobilnim robotima te smo predstavili petlje povratne veze i PID regulatore. Obradili smo i diskretizaciju PID regulatora. Na kraju smo se posvetili predstavljanju jedne konkretne implementacije svesmjernog robota. Prvo smo predstavili strukturu samog robota te elektroničke komponente koje ga sačinjavaju, a onda smo predstavili *software* pomoću kojega upravljamo samim robotom.

Ključne riječi: upravljanje, robot, svesmjerni, švedski kotač, povratna kinematika, petlja povratne veze, PID, Python, C++, Raspberry PI, Arduino.

Summary

This paper discussed controlling omnidirectional robots. The first part of the paper was dedicated to the theoretical aspects of controlling mobile robots. Rolling and sliding constraints were presented for both the fixed standard wheel and the Swedish wheel. Next, the inverse kinematic problem was described and solved for three omnidirectional robot examples. After that, the paper discussed the problem of controlling mobile robots. Feedback loops and PID controllers were presented. The discrete version of the PID controller was also shown. Finally, an implementation of an omnidirectional robot was presented. The paper first discussed its structure and electronic components and after that the software used to control it.

Key words: control, robot, omnidirectional, Swedish wheel, inverse kinematics, feedback loop, PID, Python, C++, Raspberry PI, Arduino.

Životopis

Rođen sam 29. listopada 1994. u Osijeku. Pohađao sam Osnovnu školu Ljudevita Gaja u Osijeku, a nakon toga II. gimnaziju u Osijeku. Gimnaziju završavam 2013. godine te tada upisujem Sveučilišni preddiplomski studij matematike na Odjelu za matematiku u Osijeku. Studij završavam 2016. godine sa završnim radom na temu Ruby programski jezik te time stječem akademski stupanj prvostupnika matematike. Iste godine upisujem diplomski studij na Odjelu za matematiku, smjer Matematika i računarstvo. U listopadu 2016. sudjelujem na IEEEXtreme natjecanju. Tijekom ljeta 2017. odrađujem praksu u tvrtki Mono d.o.o te se iste jeseni tamo zapošljavam.