SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Frano Petric, Ivan Rajković

PRIMJENA TEORIJE VIJČANOG GIBANJA I VIZUALNE POVRATNE VEZE U UPRAVLJANJU HODAJUĆIM ROBOTIMA

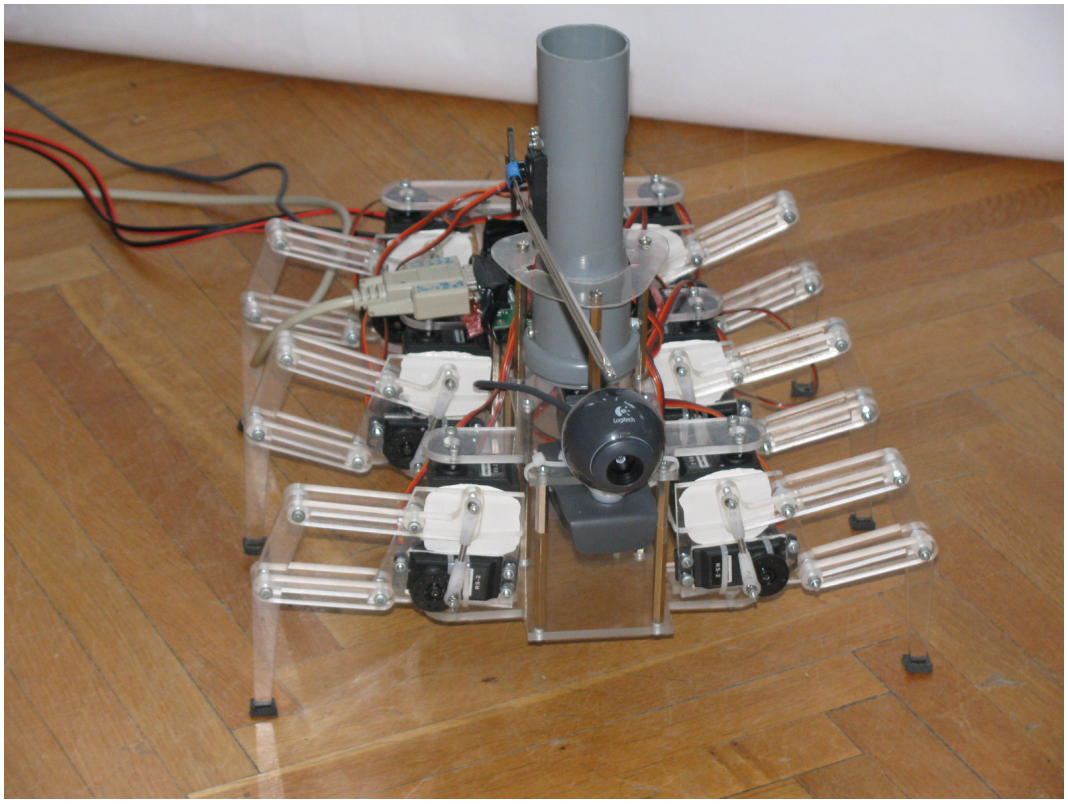
Zagreb, 2011.

Ovaj rad izrađen je u Laboratoriju za robotiku i inteligentne sustave upravljanja pod vodstvom Prof.dr.sc. Zdenka Kovačića i predan je na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2010./2011.

SADRŽAJ

1. Uvod

Šesteronožni robotski hodač razvijen je na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Razvoj robota započet je u ljetnom semestru akademske godine 2008./2009. Šesteronožni hodač izrađen je od pleksiglasa i u određenoj mjeri konstrukcijom podsjeća na kukca, što se može vidjeti na slici 1.1. Na hodača su montirani



Slika 1.1: Šesteronožni robotski hodač

web kamera i dvoosni servomehanizam (*alkar*) s ciljem ostvarivanja vizualne povratne veze te gađanja alke prilikom razvoja robotičkog alkara. Cilj je ovoga rada istražiti primjenjivost teorije vijčanog gibanja i vizualne povratne veze u upravljanju hodajućim robotima, te rezultate istraživanja prikazati na primjeu šesteronožnog robotskog hodača. U ovom radu korištena su dosadašnja znanja stečena tijekom razvoja šeste-

ronožnog hodača te su ukratko iznesena u drugom i trećem poglavlju. U drugom poglavlju ukratko je predstavljen šesteronožni robotski hodač, s naglaskom na značajke mehaničke konstrukcije, upravljačkog uređaja i montirane web kamere. Treće poglavlje rada bavi se generatorom sekvence koji je realiziran pomoću Hopfovih oscilatora. U četvrtom poglavlju iznesena je teorija vijčanog gibanja te su predloženi postupci rješavanja direktnog i inverznog kinematičkog problema. Peto poglavlje donosi rješenje direktne i inverzne kinematike šesteronožnog hodača dok je u šestom poglavlju opisana implementacija algoritama upravljanja u programskom alatu Matlab. U sedmom poglavlju fokus je postavljen na primjenu vizualne povratne veze u upravljanju hodajućim robotima posebno obraćajući pažnju na kompenzaciju kuta valjanja. Na kraju rada prikazani su ostvareni rezultati.

2. Šesteronožni robotski hodač

Cjelokupni sustav šesteronožnog robotskog hodača sastoji se od više komponenti:

- Mehanička konstrukcija
- Dvoosni servomehanizam
- Upravljački uređaj SSC-32 (Serial Servo Controller)
- Kamera

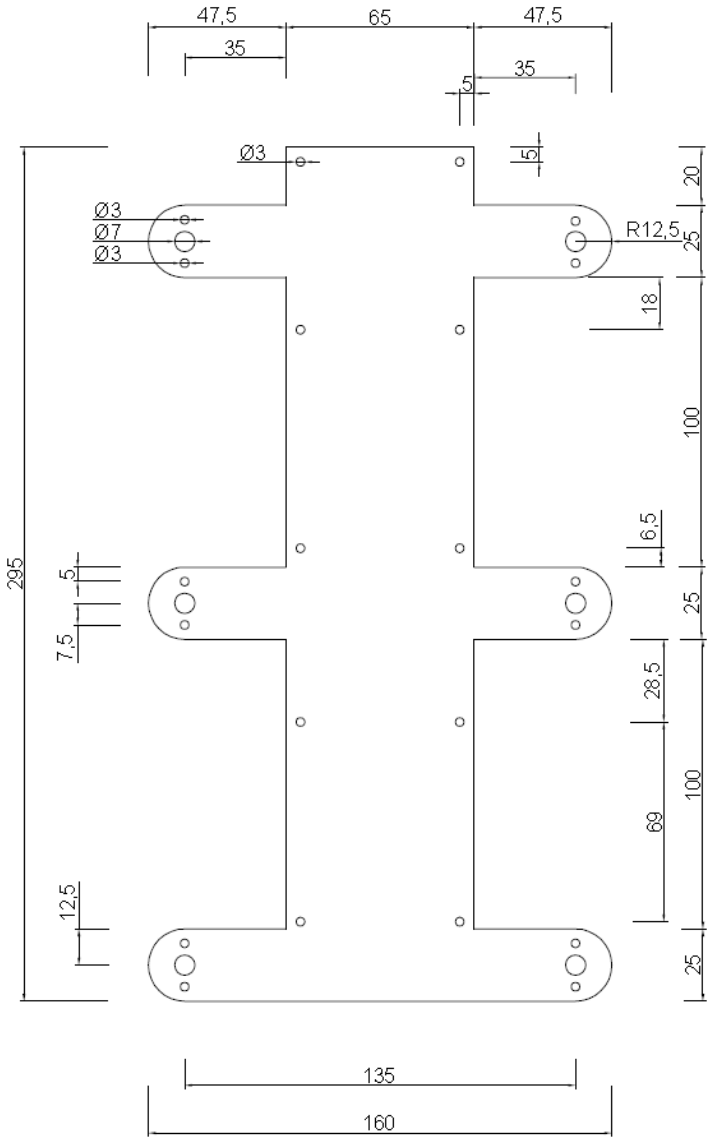
Dvoosni servomehanizam izrađen je tijekom rada na razvoju robotičkog alkara te u ovom radu neće biti korišten. Ostale komponente su obrađene u nastavku.

2.1. Mehanička konstrukcija

Mehanička konstrukcija sastoji se od trupa te 6 nogu. Trup se sastoji od dvije ploče koje su izrađene od pleksiglasa. Debljina ploča je 5mm. Na slici 2.1 prikazan je nacrt gornje ploče s pripadajućim dimenzijama. Dimenzije donje ploče identične su dimenzijama gornje ploče, a jedina razlika je u rupama koje se koriste za pričvršćivanje nogu. Noge su, kao i trup izrađene od pleksiglasa. Dijelovi noge s pripadajućim dimenzijama prikazani su na slici 2.2. Zbog konstrukcijskih značajki nogu, potreban razmak između ploča je 7.5 cm, što je postignuto korištenjem odstojnika. Motori su i na bočni nosač motora (dio noge označen brojem 4 na slici slika noga dijelovi) i na donji nosač (pločica pleksiglasa) pričvršćeni plastičnim vezicama.

2.1.1. Aktuatori

Za pokretanje robota korišteni su Modelcraftovi servo motori tipa RS-2, koji se često koriste u robotima na daljinsko upravljanje, primjerice kod modela automobila za skretanje, kod modela plovila za zakretanje kormila, kod letjelica i sl. Osnovne značajke tih motora dane su u tablici 2.1, dok je motor prikazan na slici 2.3.

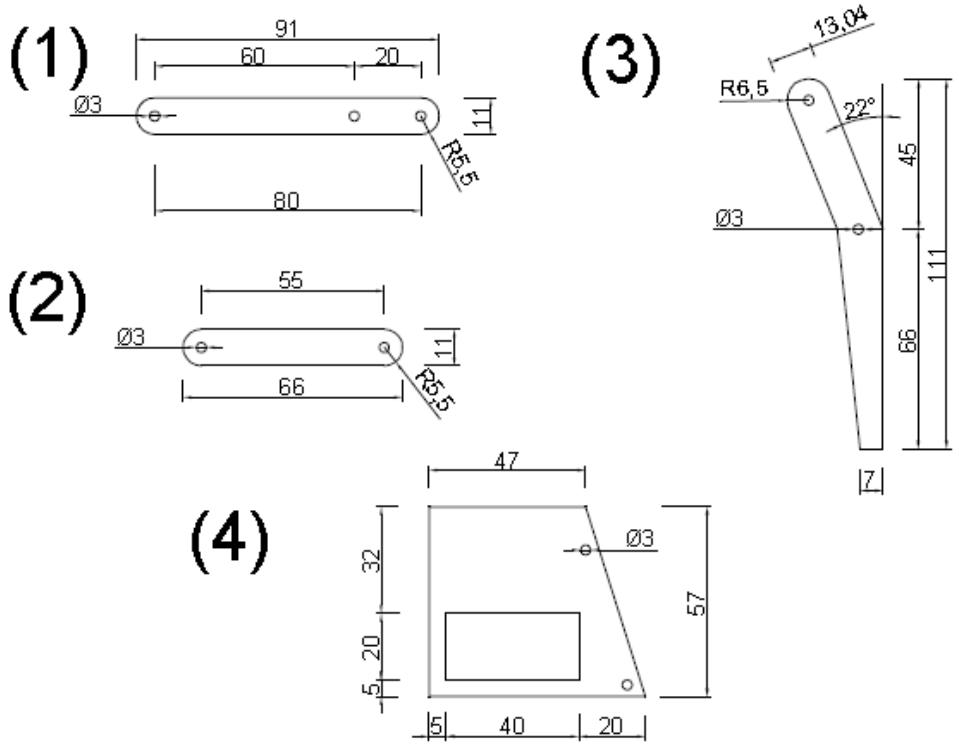


Slika 2.1: Nacrt gornje ploče tijela hodača

Motor se na PWM izlaz kontrolera spaja JR konektorom s tri priključne žice (upravljački signal (PWM), napajanje, uzemljenje).

Generiranje i značajke PWM signala

Servo motori su, za razliku od standardnih istosmjernih i izmjeničnih električnih motora, pozicijski motori. Pozicija svakog motora određena je širinom, odnosno trajanjem pulsa upravljačkog signala, što je prikazano na slici 2.3, pri čemu je bitno napomenuti kako svaki servo motor u kućištu sadrži svu potrebnu elektroniku kojom se upravljački signal pretvara u željeni zakret motora. PWM signal (engl. *PWM = Pulse Width Modulation*, odnosno impulsno-širinska modulacija) se sastoji od pulseva promjenjive širine



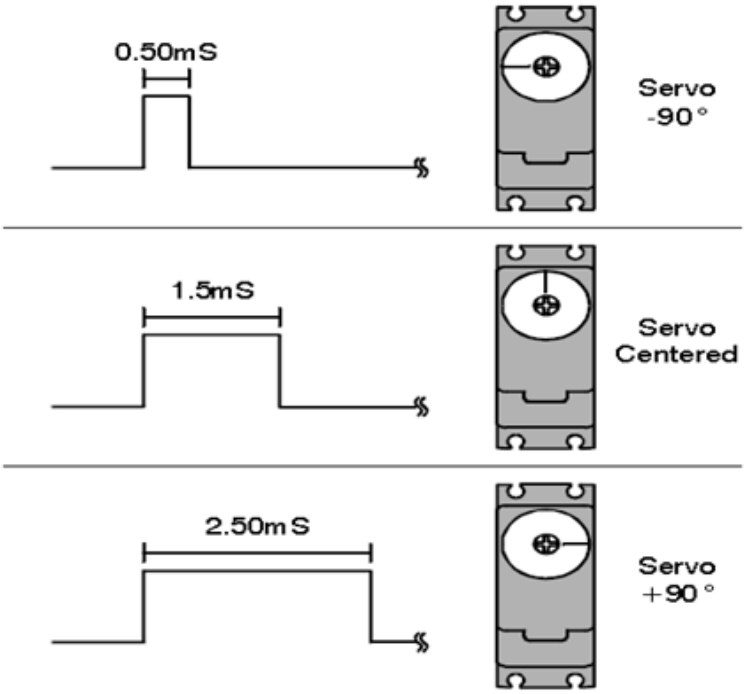
Slika 2.2: Dijelovi noge hodača

odnosno trajanja koji se pojavljuju u pravilnim razmacima. Kako se kod PWM signala modulira širina pulsa, to znači da upravo širina pulsa nosi informaciju. Taj signal se dovodi na jedan od tri priključka JR konektora (na većini upravljačkih pločica, pa tako i na onoj koja se nalazi na robotu označen je oznakom *pulse*).
Kako bi upravljačka elektronika mogla pretvoriti taj signal u zakret motora, on mora imati sljedeća obilježja:

– Frekvencija 50 Hz (pulsevi se pojavljuju u pravilnim razmacima od 20 ms)

Tablica 2.1: Osnovne značajke RS-2 motora

Značajka	Vrijednost
Dimenzije [mm]	$40.6 \times 20 \times 38.6$
Težina [g]	39.2
Moment [Ncm]	34.4 pri 6[V]



Slika 2.3: Prikaz pozicije motora u ovisnosti o trajanju pulsa [3]

– Širina, odnosno trajanje pulsa mora biti u rasponu od $500\mu s$, do $2500\mu s$, odnosno od 0.5 ms do 2.5 ms

Iako je upravljanje motorima pomoću Serial Servo Controllera moguće bez uvida u način na koji se generira PWM signal, za razumijevanje naredbi koje se koriste nužno je posjedovati barem osnovna znanja o generiranju upravljačkih signala za servo motore. Generiranje PWM signala najčešće je riješeno sklopovski za određen broj izlaznih kanala, pa tako i u slučaju Atmega168-20PU kontrolera, koji se nalazi na korištenoj pločici za upravljanje motorima. Za sklopovsko generiranje PWM signala potrebno je podesiti frekvenciju ugrađenih brojila (engl. *timer*) na 50 Hz, a zatim odgovarajućim postavkama registara mikrokontrolera odabrati način generiranja PWM signala. Atmega kontroleri nude 3 načina generiranja PWM signala:

– Brzi PWM

– PWM s ispravnom fazom

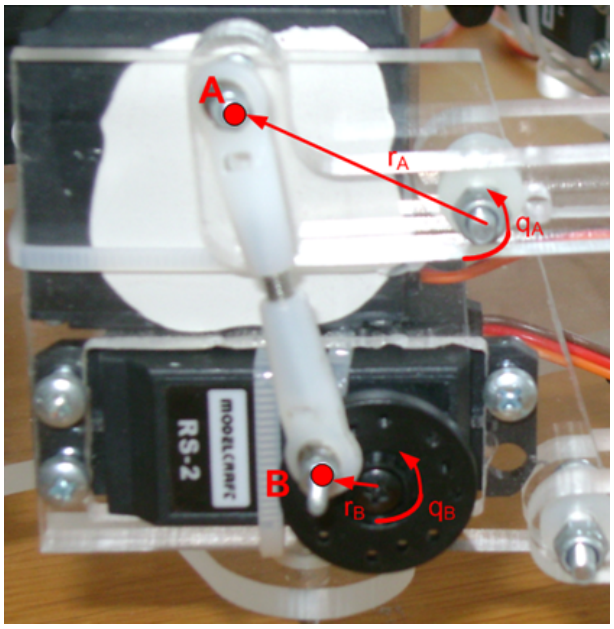
– PWM s ispravnom fazom i frekvencijom

Za upravljanje servo motorima najčešće je dovoljan brzi PWM način, jer osigurava dovoljnu točnost širine pulsa te malene devijacije frekvencije i faze upravljačkog signala. Ukoliko se koriste složeniji motori s većom razlučivosti, potrebno je odabrati PWM s ispravnom fazom i frekvencijom. Nakon postavljanja frekvencije i odabira načina

generiranja PWM signala, duljina pulsa se određuje jednostavnim upisom određene vrijednosti u OCR (engl. *Output Compare Register*). Sam signal se generira tako da se na PWM izlaz propušta visoka razina signala dok je vrijednost brojila manja od vrijednosti u OCR registru, a kad vrijednost brojila prijeđe zadanu vrijednost, na PWM izlaz se propušta niska razina signala. Ovakav način je osnovna postavka (tzv. *non-inverting mode*), a moguće je odabrati i suprotan efekt, jer Atmega kontroleri nude i tzv. *inverting mode* (invertirajući način). Formule za izračun vrijednosti kojim se postavlja frekvencija ugrađenih timera, izračun vrijednosti koje je potrebno upisivati u OCR registar daje proizvođač mikrokontrolera. Važno je napomenuti kako se PWM može generirati i programski, tako da se unutar programa mikrokontrolera uspoređuje vrijednost unutar brojila s nekom definiranom vrijednosti. Međutim, zbog kašnjenja moguće su znatne devijacije frekvencije a posebice faze PWM signala te se ovaj način, iako daje zadovoljavajuće rezultate za jednostavne servo motore, najčešće ne koristi.

2.1.2. Reduktor zakreta

Bitna značajka mehaničke konstrukcije šesteronožnog hodača je redukcija zakreta motora koji podiže odnosno spušta nogu, koja je posljedica sustava za prijenos zakreta koji je prikazan na slici 2.4. Prijenos zakreta motora vrši se pomoću elementa s kugličnim



Slika 2.4: Reduktor zakreta motora

ležajevima (engl. *Ball-links*, njem. *Kugelgelenke*). Omjer redukcije dan je sljedećim

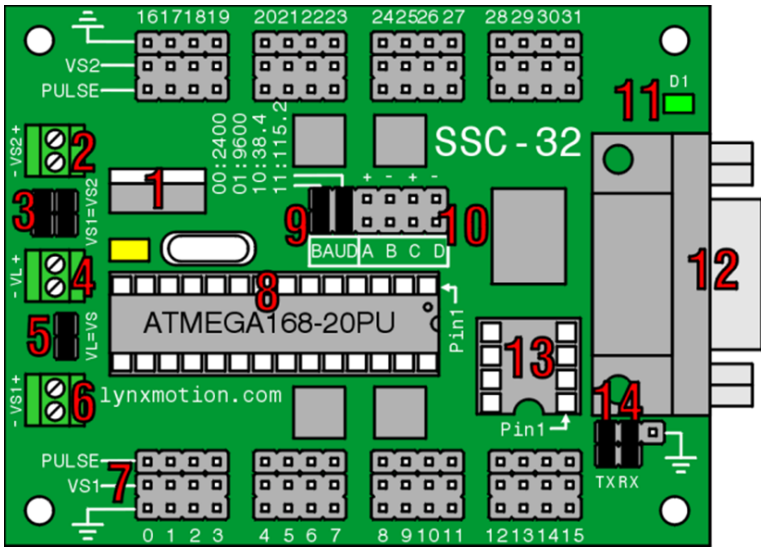
izrazom:

$$k = \frac{q_A}{q_B} = \frac{r_B}{r_A} = 0.3087 \tag{2.1}$$

pri čemu je q_B zakret motora a q_A zakret zgloba. Jako malen omjer prijenosa predstavlja značajno ograničenje prilikom upravljanja šesteronožnim hodačem, posebno u smislu kompenzacije nagiba podloge.

2.2. Upravljački uređaj

Za upravljanje zakretima motora koristi se pločica koja omogućuje istovremeno upravljanje s 32 servo motora (Serial Servo Controller, SSC-32, u daljnjem tekstu SSC). SSC je proizvod tvrtke Lynxmotion, a omogućuje više načina rada motora, od trenutnog pokretanja (engl. *immediate response*), pokretanje uz zadanu brzinu (engl. *speed control*), pokretanje uz zadano vrijeme gibanja od početnog do završnog položaja (engl. *timed motion*) do kombinacije prethodnih načina rada motora, dok se grupnom naredbom omogućuje da svi motori krenu u isto vrijeme te dosegnu konačni položaj istovremeno, što je iznimno bitno kod robota hodača pri generiranju sekvence hoda. Na slici 2.5 nalazi se pojednostavljeni prikaz SSC-a, na kojem su brojevima označeni



Slika 2.5: Serial Servo Controller - pojednostavljeni prikaz [3]

bitni dijelovi. Označeni dijelovi predstavljaju:

1. Stabilizator napona (5.0 V, 500 mA). Na izlazu daje napon od 5.0 V, pri čemu ulaz može biti u rasponu od 5.5 V do 9.0 V. Važan je zbog osiguravanja stabilnog napona za elektroničke komponente sklopa. Iako je sam stabilizator projektiran

za izlaznu struju od 500 mA, dodatno je njegov izlaz ograničen na 250 mA kako bi se smanjilo zagrijavanje.

2. VS2 terminal – konektor za dovođenje napajanja za izlazne kanale 16-31. Ukoliko se koriste standardni servo motori napon napajanja mora biti u rasponu od 4.8 V do 7.2 V.

3. VS1=VS2 Jumpers – kratkospojnici koji omogućuju prespajanje napona s VS1 na VS2 i obratno, odnosno omogućuje napajanje svih kanala iz jednog izvora, što je posebno bitno kod mobilnih robota koji koriste jednu bateriju. Ukoliko se želi postići opisani efekt, potrebno je koristiti oba kratkospojnika, odnosno kratko spojiti oba para pinova na pločici.

4. VL terminal – (engl. *Logic Voltage = VL*) - konektor za dovođenje napajanja za elektroničke komponente sklopa, odnosno svega što je spojeno na 5-voltne vodove na pločici. Ovaj ulaz se koristi kako bi se odvojilo napajanje elektronike od napajanja servo motora, što je posebno važno pri korištenju većeg broja motora koji mogu povući veću struju koja bi mogla naštetiti elektronici. Obično se na VL dovodi napon od 9.0 V.

5. VL=VS Jumper – kratkospojnik koji omogućuje napajanje elektronike i motora iz istog izvora. Ukoliko se koristi ovakav način napajanja SSC-a potrebno je osigurati minimalno 6.0 V na ulazu kako bi sklop ispravno radio. Ipak, tijekom rada s više servo motora moguće je da se mikrokontroler resetira, što je znak da je potrebno odvojiti napajanje upravljačke elektronike od napajanja izlaznih kanala.

6. VS1 terminal – konektor za dovođenje napajanja za kanale 0-15. Kao i kod VS2 potrebno je na ulaz dovesti napon u rasponu od 4.8 V do 7.2 V.

7. Izlazni pinovi na koje se spajaju servo motori, ukupno 32 kanala od kojih svaki ima tri izlazna pina. Za sve kanale vrijedi jednak raspored, odnosno:
   - Vanjski niz pinova – GND – referentna točka
   - Srednji niz pinova – VS1 ili VS2 – napajanje za motore
   - Unutranji niz pinova – PULSE – upravljački signal za motore Također, svaki od kanala moguće je koristiti kao TTL izlaz, što omogućuje dodatne podatkovne izlaze SSC-a, naravno, uz preinake programa unutar mikrokontrolera.

8. Atmega168-20PU – mikrokontroler koji je tvornički programiran za generiranje PWM signala prema naredbama koje prima sa serijskog porta.

9. BAUD inputs – omogućuju podešavanje brzine prijenosa serijske komunikacije (engl. *baud rate*).

10. ABCD inputs – ulazi koji po potrebi mogu biti i digitalni i analogni. Ukoliko se čitaju kao digitalni ulazi (naredba Read Digital Inputs) koristi se interni *pullup* otpornik od 50 $k\Omega$. Ukoliko se čitaju kao analogni ulazi, rapon od 0 V do 5 V zapisuje se u 1 oktet, što znači da se koristi 8-bitovni A/D pretvornik.

11. LED – dioda koja označava stanje procesora. Pri uključenju se pali ako je sve u redu i konstantno svijetli do primanja prve ispravne naredbe. Nakon što SSC primi prvi niz znakova preko serijske komunikacije, dioda treperi za vrijeme prijenosa podataka, dok je ostalo vrijeme ugašena. Ukoliko se tijekom rada dogodi reset mikrokontrolera, što je izgledno ukoliko se koristi veći broj motora te se ne odvoji napajanje elektronike od napajanja motora, dioda će ponovno konstantno svijetliti do primitka prve ispravne naredbe.

12. DB9 port – port za spajanje serijskog kabela na SSC.

13. EEPROM socket (8-pinski)

14. TTL serial port – za komunikaciju s SSC-om moguće je koristiti i TTL serijski port (pinovi RX, TX te GND). Ukoliko se umjesto TTL-a želi koristiti DB9 port, TTL port se mora prespojiti kratkospojnicima.

Dodatne upute za podešavanje navedenih i ostalih postavki SSC-a korištenjem kratkospojnika mogu se pronaći na engleskom jeziku u *.pdf* formatu [3]. Važno je napomenuti kako je pri spajanju napajanja na SSC potrebno obratiti pozornost na polaritet ulaznog napona, jer zamjena polariteta ulaznog napona može trajno oštetiti elektroničke komponente SSC-a. Također, iako proizvođač to ne preporuča, moguće je reprogramiranje mikrokontrolera, čemu je potrebno pristupiti s velikim oprezom, jer bi se pogrešnim postupkom mogla trajno izgubiti funkcionalnost SSC-a.

2.2.1. Upravljanje servo motorima pomoću SSC-a

Upravljanje servo motorima moguće je ostvariti slanjem naredbi putem serijske komunikacije. Osnovna naredba kojom se ostvaruje zakret motora je sljedećeg oblika:

 # <ch> P <pw> <cr>

pri čemu je:

- <ch> broj kanala na koji je spojen motor
- P naredba za postavljanje širine pulsa na odabrani kanal
- <pw> širina pulsa u mikrosekundama
- <cr> predstavlja ASCII znak 13, (engl. *carriage return*), kojim mora završiti svaka naredba za SSC.

Grupna naredba kojom se upravlja s više servo motora je sljedećeg oblika:

 # <ch> P <pw> # <ch> P <pw> ... # <ch> P <pw> <cr>

Kako je objašnjeno u prethodnom potpoglavlju, moguće je upravljanje brzinom zakreta motora, što se ostvaruje dodavanjem naredbe S u prethodno opisanu naredbu:

 # <ch> P <pw> S <spd> <cr>

pri čemu je <spd> brzina zakreta u $\mu s/s$ pri čemu $1\mu s$ širine pulsa odgovara zakretu od $0.1°$. Ukoliko se koristi grupna naredba oblika:

 # <ch> P <pw> # <ch> P <pw> S <spd> <cr>

brzinom određenom argumentom <spd> gibat će se samo drugi motor, što znači da naredba S ne utječe na cijelu grupu već samo na jedan motor. Za razliku od naredbe S, naredba T kojom se ostvaruje gibanje svih motora unutar određenog vremenskog intervala, utječe na sve motore unutar grupne naredbe. Grupna naredba s definiranim trajanjem zakreta je sljedećeg oblika:

 # <ch> P <pw> # <ch> P <pw> S <spd> # <ch> P <pw> T <time> <cr>

gdje je <time> vrijeme trajanja zakreta u milisekundama. Ovakva naredba će prouzročiti takvo gibanje da će se brzina gibanja prvog i trećeg motora unutar grupne naredbe prilagoditi tako da se osvari vrijeme trajanja zakreta definirano naredbom T, dok brzina drugog motora unutar grupne naredbe određena naredbom S zapravo predstavlja maksimalnu brzinu, dok stvarna brzina prilagođena vremenu određenu naredbom T može biti i manja. Ukoliko pak brzina definirana poljem S nije dovoljna da se zakret obavi unutar vremena T, svi motori će se gibati dulje od definiranog vremena.

Općenito, ukoliko se koristi grupna naredba s kombinacijom naredbi S i T, brzina motora određuje se prema sljedećim pravilima:

1. Svi servo motori navedeni unutar grupne naredbe krenuti će i završiti gibanje istovremeno.
2. Ukoliko je za neki motor definirana brzina naredbom S, motor se neće gibati

brzinom većom od definirane, ali se može gibati manjom brzinom ako tako zahtjeva vrijeme zadano naredbom T.

3. Ukoliko je za grupu motora zadano vrijeme trajanja zakreta naredbom T, zakret svih motora neće završiti prije isteka definiranog vremena, ali se može produžiti ukoliko brzina definirana naredbom S nije dovoljna za ostvarenje gibanja unutar zadanog vremena.

Važno je napomenuti kako se ne preporuča zadavanje naredbi S i T unutar prve naredbe. Razlog leži u činjenici da kontroler zapravo nema povratnu informaciju u kojoj se poziciji motor stvarno nalazi. Prema tome, pri uključivanju kontroler ne može znati u kojoj je poziciji motor, te će nakon primitka naredbe S ili T pretpostaviti da treba krenuti iz krajnje pozicije (koja odgovara širini pulsa od 500 μs), te će se prije obavljanja naredbe najvećom mogućom brzinom pozicionirati u taj položaj, što može biti štetno za sustav.

2.3. Kamera

Kako bi se omogućio robotski vid, postojećem robotskom sustavu dodana je web kamera. Odabrana je kamera *USB Logitech Webcam C200*, a korištena rezolucija je 640×480. Web kamera prikazana je na slici 2.6.

Slika 2.6: Web Kamera *USB Logitech Webcam C200*

2.3.1. Računalni vid

Računalni vid je dio računalne znanosti koji omogućuje stroju da *vidi*, gdje *vidi*' zapravo znači da je sposoban izvući informacije iz slike koje su mu neophodne da bi obavio zadatak. Računalni vid proučava i opisuje programske i sklopovske postupke

kojima se ostvaruje oblik umjetne inteligencije. Računalni vid je sveprisutan – sigurnosni sustavi, ispitivanje i kontroliranje kvalitete proizvodnje, medicina, bespilotne letjelice samo su neke od primjena. Kako ne postoji standardno utvrđivanje *problema računalnog vida*, ne postoji niti standardiziran način rješavanja tog problema. Umjesto toga, postoji bogatstvo metoda za rješavanje strogo definiranih zadataka, gdje su metode usko povezane sa samim zadatkom i rijetko se mogu iskoristiti za širu primjenu. U realnim izvedbama računalnog vida, računala su unaprijed programirana za rješavanje određenog zadatka. Područja analize i obrade slike te robotski vid usko su povezani s računalnim vidom. Robotski vid je inženjerska grana koja koristi računalni vid u industrijske svrhe, gdje se informacije izvučene iz slike koriste kao podrška proizvodnog procesa. Polje rada i primjene robotskog vida je široko i sveobuhvatno te je samu definiciju teško sažeti. Dvije najčešće korištene definicije su:

– Analiza slike u svrhu izdvajanja podataka za upravljanje procesom ili aktivnosti

– Prepoznavanje stvarnih objekata u slici i pripisivanje pripadajućih karakteristika tim objektima – zaključivanje što oni znače

Tehnologije senzora za slike i teorija upravljanja često su integrirane sa obradom slike s ciljem upravljanja robota te efikasnost programskih i sklopovskih izvedbi naglašava važnost obrade u stvarnom vremenu (engl. *real-time*). Primjeri ovakvih primjena su ispitivanje kvalitete, mjerenje pozicije i orijentacije, otkrivanje događaja, slijeđenje, prepoznavanje objekta, autonomna vozila i sl. No, računala ne *vide* na isti način kako su ljudska bića sposobna. Dok se ljudi mogu pouzdati na vlastitu sposobnost zaključivanja na temelju iskustva, računalni uređaji vide pomoću određivanje zasebnih piksela slike, procesiranja istih i potom donošenja zaključka uz pomoć baza podataka i implementiranih mogućnosti, kao što je mehanizam prepoznavanja oblika. Prepoznavanje, odnosno određivanje sadrži li slika određeni objekt, svojstvo ili aktivnost, predstavlja klasični problem računalnog vida, obrade slike i robotskog vida. Taj se problem može robusno riješiti i bez ljudskog djelovanja, no još ne postoje zadovoljavajuća rješenja računalnog vida za opći slučaj: proizvoljan objekt u proizvoljnoj situaciji. Postojeće metode mogu u najboljem slučaju prepoznati specifičan objekt u specifičnoj situaciji (u pravilu su strogo određeni svjetlost, pozadina, položaj objekta naspram kamere itd). Pri odrađivanju zadataka koji se mogu svrstati pod pojam računalnog vida od velike je pomoći platforma *OpenCV*.

2.3.2. OpenCV

OpenCV (*Open Source Computer Vision Library*) skup je C funkcija i nekoliko C++ klasa koje se implementiraju u mnoge popularne algoritme za analizu slike i računalni vid. OpenCV se može definirati kao biblioteku računalnog vida. Omogućuje jednostavan pristup radnom okviru računalnog vida i sveobuhvatnoj biblioteci sa više od 500 funkcija koje se izvršavaju u stvarnom vremenu. U ovom radu koriste se funkcije OpenCV-a za dohvat slike s kamere, detekciju linija na slici i prikazivanje slike na ekranu. Postupak instalacije paketa preuzet je sa stranice [2].

3. Centralni generator sekvenci

Za stvaranje generatora sekvence koristi se razmatranje provedeno u [9].
Roboti hodači (engl. *legged robots*) prilikom gibanja obavljaju određenu sekvencu koja se sastoji od periodičkog ponavljanja podizanja odnosno spuštanja nogu te pokretanja nogu naprijed odnosno natrag. Kod robota hodača to periodičko ponavljanje najčešće se naziva hod. Za šesteronožne hodače najčešće se upotrebljavaju tri načina hoda:

- tronožni korak (engl. *tripod gait*)
- valni korak (engl. *wave gait, metachronal gait*)
- viševalni korak (engl. *ripple gait*)

3.1. Opis načina hoda šesteronožnog hodača

Kako bi se osigurala stabilnost kretanja robotskog hodača potrebno je da svaka generirana sekvenca zadovoljava sljedeća dva osnovna uvjeta:

- Noga se nikada ne diže s tla dok na istoj strani hodača nije postavljena druga noga u položaj na tlu kako bi držala tijelo hodača
- Nasuprotne noge strogo alterniraju - dok je jedna u zraku druga je na podlozi i obratno

Za generalni opis hoda šesteronožnog hodača potrebno je definirati vrijeme ciklusa (engl. *cycle time*), faktor popunjenja $\beta \in [0, 1]$ (engl. *duty factor*) te relativnu fazu između pojedinih nogu kojom je određena njihova međusobna koordinacija. Faktor popunjenja β definira se kao omjer vremena koje noga provodi na podlozi i ukupnog vremena koje je potrebno za obavljanje jednog perioda sekvence, odnosno vremena ciklusa:

$$\beta = \frac{T_{st}}{T_{st} + T_{sw}} \tag{3.1}$$

pri čemu je T_{sw} vrijeme trajanja dijela sekvence u kojem je noga hodača u zraku - vrijeme zamaha, dok je T_{st} vrijeme trajanja onog dijela sekvence u kojem je noga

hodača na podlozi - vrijeme upora. Tipične vrijednosti za osnovne načine hoda dane su u tablici 3.1. Vrijeme ciklusa je ukupno vrijeme koje je potrebno da noga obavi cijelu sekvencu i vrijedi:

$$T = T_{sw} + T_{st} \tag{3.2}$$

Tablica 3.1: Način hoda i faktor popunjenja

Način hoda	Faktor popunjenja β
Valni korak	$\frac{3}{4}$
Viševalni korak	$\frac{5}{8}$
Tronožni korak	$\frac{1}{2}$

Također je potrebno definirati oznake i brojeve nogu robota. Najčešće oznake su L za noge na lijevoj strani i R za noge na desnoj strani, dok se numeriranje vrši od prednjih prema stražnjima. Oznake i brojevi nogu dani su u tablici 3.2.

Tablica 3.2: Oznake i brojevi nogu

Noga	Oznaka	Broj
Prednja lijeva	L	1
Prednja desna	R	1
Srednja lijeva	L	2
Srednja desna	R	2
Stražnja lijeva	L	3
Stražnja desna	R	3

3.1.1. Valni korak

Valni korak je najsporiji oblik hoda kod šesteronožnih hodača. Karakterizira ga velik iznos faktora popunjenja $\beta = \frac{3}{4}$ što znači da je svaka noga veći dio vremena na tlu, zbog čega taj oblik hoda i jest tako spor. Prilikom gibanja valnim korakom u svakom trenutku je 5 nogu robota na tlu, što znači da je robot iznimno stabilan i statički i dinamički. Relativne faze između pojedinih nogu su:

- $\frac{\pi}{3}$ za noge koje se nalaze na istoj strani robota
- π za nasuprotne noge (zadovoljen uvjet da nasuprotne noge strogo alterniraju)

3.1.2. Viševalni korak

Viševalni korak predstavlja određeni kompromis između brzine i stabilnosti gibanja, jer su u svakom trenutku 4 noge hodača na tlu dok su dvije u zraku, pa je faktor popunjenosti manji nego kod valnog koraka.

3.1.3. Tronožni korak

Tronožni korak je najbrži način gibanja šesteronožnih robota, što sugerira i relativno malen faktor popunjenja. Zbog toga je i najmanje stabilan, ali općenito su svi načini hoda šesteronožnih hodača zbog velikog broja nogu stabilni i statički i dinamički pa tako stabilnost hodača nije ugrožena. Tronožni korak karakterizira to da su u svakom trenutku 3 noge na tlu dok se tri gibaju.

3.2. Hopfovi oscilatori

Kako pri gibanju zglobovi robota periodički ponavljaju istu trajektoriju, kao generator takve trajektorije može se koristiti oscilator. Pritom je bitno da su oscilacije koje daje oscilator stabilne te da je moguće upravljanje amplitudom i frekvencijom oscilacija. Za takav oscilator nije moguće koristiti linearne sustave jer oni nisu u stanju ostvariti stabilne trajne oscilacije. Potrebno je koristiti nelinearan sustav koji sadrži ravnotežno stanje oko kojeg se zatvara trajektorija oblika graničnog ciklusa (engl. limit cycle). Prema [15] zatvorena krivulja u faznoj ravnini, odnosno granični ciklus, može biti stabilan ili nestabilan. Nestabilan granični ciklus u velikoj mjeri ovisi o početnim uvjetima sustava te se uopće ne pojavljuje u sustavu ukoliko se početno stanje ne nalazi na graničnom ciklusu, zbog čega nije primjenjiv za implementaciju generatora sekvence. Za razliku od nestabilnog, stabilan granični ciklus se uspostavlja za sve početne uvjete sustava koji se nalaze dovoljno blizu ravnotežnog stanja oko kojeg se sam granični ciklus uspostavlja.

3.2.1. Hopfova bifurkacija

Jedan nelinearan sustav koji omogućuje prethodno opisano ponašanje je i Hopfov oscilator zasnovan na Hopfovoj bifurkaciji. Pojam bifurkacija usko je vezan uz nelinearne sustave i prema [15] može se opisati kao točka u faznoj ravnini u kojoj sustav naglo mijenja dinamičko ponašanje, odnosno točka u kojoj sustav nije strukturno stabilan. Hopfova bifurkacija je oblik bifurkacije u kojem u ovisnosti o određenom parametru

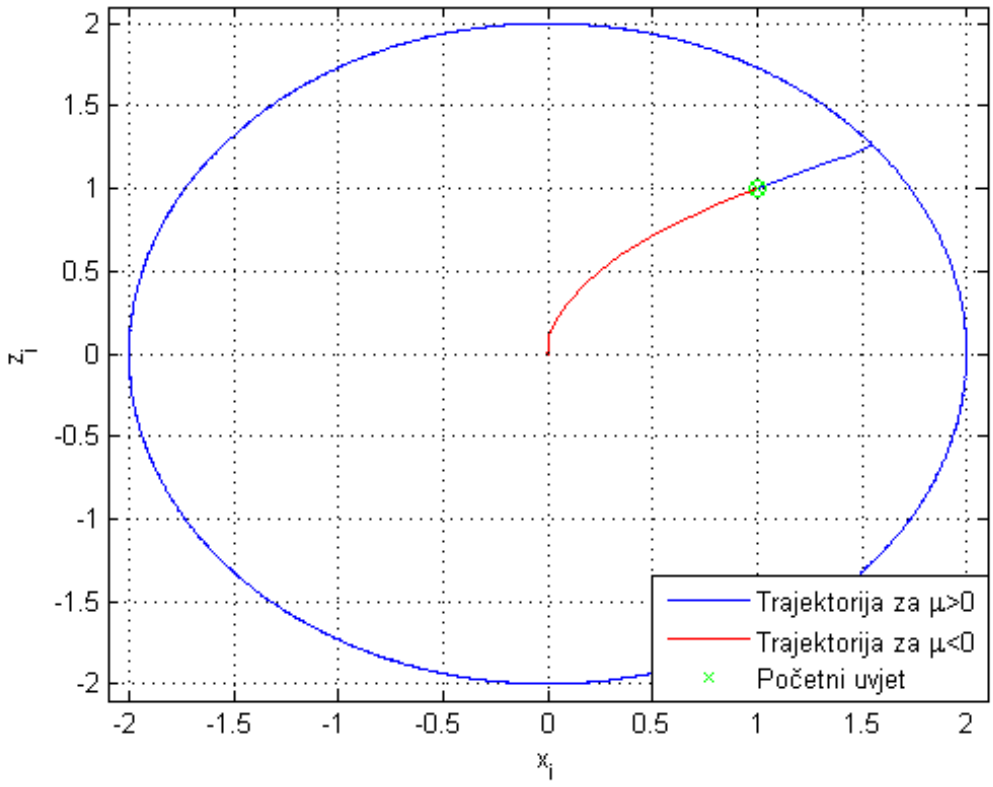
sustava svojstvene vrijednosti lineariziranog sustava u okolici ravnotežnog stanja prelaze iz desne (stabilne) poluravnine u nestabilnu i obratno. U [9] predložen je Hopfov oscilator dan sljedećim relacijama:

$$\dot{x}_i = \alpha(\mu - r_i^2)x_i - \omega z_i \tag{3.3}$$

$$\dot{z}_i = \alpha(\mu - r_i^2)z_i - \omega x_i \tag{3.4}$$

pri čemu je:

- $r_i = \sqrt{x_i^2 + z_i^2}$
- α parametar brzine konvergencije oscilatora. Uz veću vrijednost tog parametra trajektorija stanja oscilatora brže dođe u granični ciklus
- μ određuje oblik fazne trajektorije nelinearnog sustava, što znači da je to bifurkacijski parametar za Hopfov oscilator. Za $\mu < 0$ ravnotežno stanje sustava je tipa stabilnog fokusa dok se za $\mu > 0$ uspostavlja stabilan harmonički granični ciklus, što se može vidjeti na slici 3.1.



Slika 3.1: Fazna trajektorija Hopfovog oscilatora

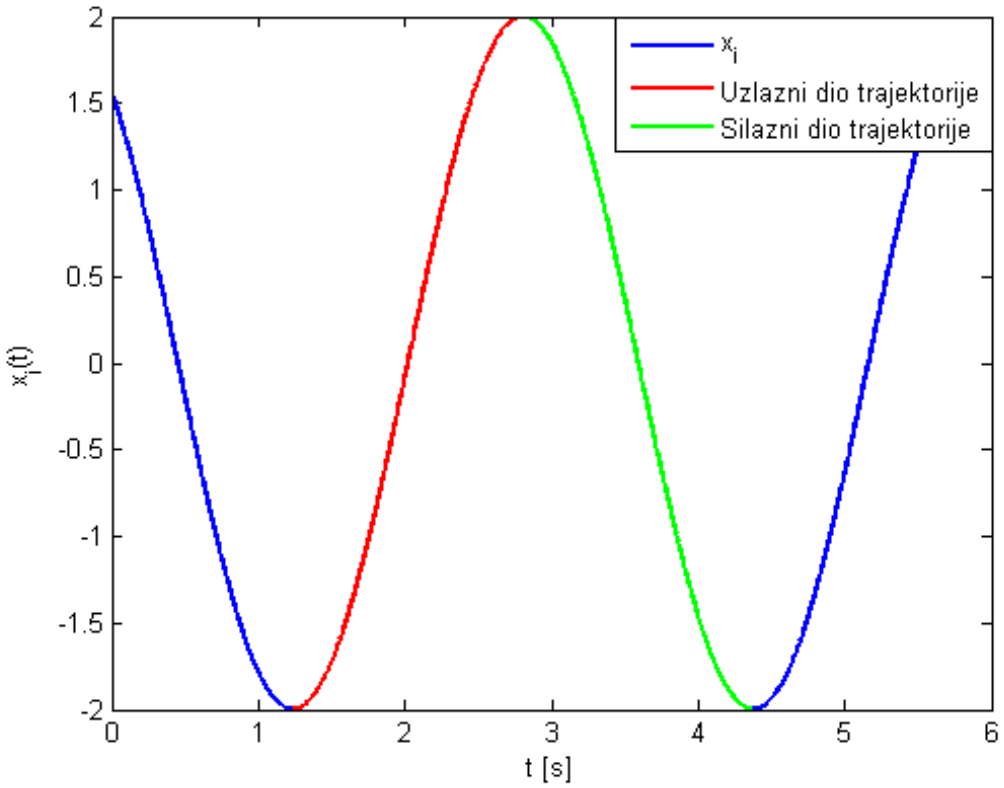
Oscilator generira glatke harmoničke oscilacije koje odgovaraju rješenjima nelinearnih diferencijalnih jednadžbi iz relacija (3.3) i (3.4). Prema [5] za $\mu > 0$ rješenja sustava

su:

$$x_i(t) = \sqrt{\mu}\sin(\omega t + \theta_0), \quad z_i(t) = \sqrt{\mu}\cos(\omega t + \theta_0) \tag{3.5}$$

pri čemu je fazni pomak θ_0 određen početnim uvjetima $x(0)$ i $z(0)$. Iz izraza (3.5) može se uočiti kako je amplituda uspostavljenih oscilacija jednaka $\sqrt{\mu}$.

Prema [9] za generiranje trajektorija zglobova koristi se varijabla stanja oscilatora x_i, dok je za generiranje hoda potrebno 6 takvih oscilatora, za svaku nogu po jedan. U obliku oscilatora koji je dan relacijama (3.3) i (3.4) nije moguće generiranje hoda zbog toga što u trajektoriji varijable stanja x_i uzlazni i silazni dio (koji odgovaraju pomicanju noge naprijed odnosno natrag) imaju jednako trajanje što se može vidjeti na slici 3.2. Jednako tako nije omogućena ni međusobna koordinacija odnosno sinkronizacija



Slika 3.2: Odziv varijable stanja x_i Hopfovog oscilatora

oscilatora. Zbog toga je potrebno proširiti model oscilatora.

Kako bi se omogućilo upravljanje vremenima gibanja noge naprijed natrag potrebno je (prema [9] i [14]) definirati frekvenciju ω na sljedeći način:

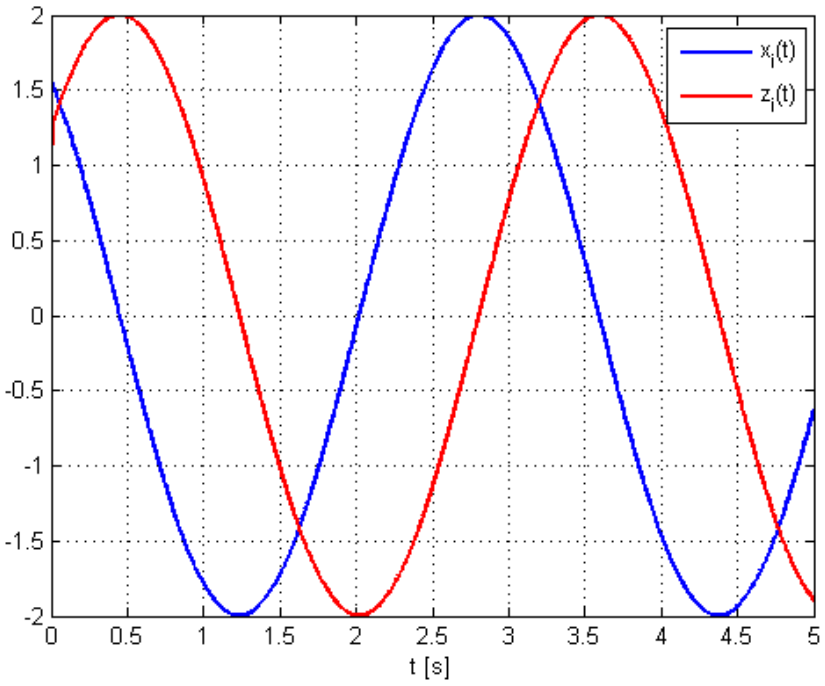
$$\omega = \frac{\omega_{st}}{e^{-az}+1} + \frac{\omega_{sw}}{e^{az}+1} \tag{3.6}$$

pri čemu je:

– $\omega_{sw} = \frac{\pi}{T_{sw}}$ frekvencija zamaha (engl.*swing frequency*)

– $\omega_{st} = \frac{\pi}{T_{st}}$ frekvencija upora (engl. *stance frequency*)

Izrazom (3.6) frekvencija ω definirana je tako da alternira između ω_{sw} i ω_{st}. Koristi se svojstvo oscilatora da je fazni pomak između x_i i z_i upravo $\frac{\pi}{2}$ što se može vidjeti na slici 3.3, a može se uočiti i iz relacije (3.5) jer je fazni pomak između funkcija $\sin$ i $\cos$ $\frac{\pi}{2}$. To znači da je za vrijeme dok je $\dot{x}_i > 0$, što odgovara fazi zamaha, varijabla stanja $z_i < 0$ pa u izrazu (3.6) prevladava frekvencija ω_{sw}. Suprotno tome, za vrijeme faze upora u kojoj je $\dot{x}_i < 0$ varjabla stanja $z_i > 0$ pa u izrazu (3.6) prevladava frekvencija ω_{st}. Parametrom $a > 0$ upravlja se razinom asimetričnosti izlaznog signala iz oscilatora. Što je njegova vrijednost veća to je razlika između ω_{sw} i ω_{st} uočljivija, ali se povećava izobličenost izlaznog signala iz oscilatora.



Slika 3.3: Varijable stanja Hopfova oscilatora

Iz izraza (3.1) slijedi:

$$\omega_{st} = \frac{1-\beta}{\beta}\omega_{sw} \tag{3.7}$$

Sada je moguće generirati hod s nekim željenim faktorom popunjenosti β na takav način da se frekvencija zamaha ω_{sw} drži konstantnom dok se frekvencija stajanja mijenja u ovisnosti o parametru β prema relaciji (3.7).

3.2.2. Sinkronizacija oscilatora

Koordinacija između oscilatora ostvaruje se povezivanjem oscilatora prema sljedećoj relaciji:

$$\begin{bmatrix} \dot{x}_i \\ \dot{z}_i \end{bmatrix} = \begin{bmatrix} \alpha(\mu - r_i^2) & -\omega \\ \omega & \alpha(\mu - r_i^2) \end{bmatrix} \begin{bmatrix} x_i \\ z_i \end{bmatrix} + \sum_{j \neq i} \mathbf{R}^{-1}(\theta_j^i) \begin{bmatrix} 0 \\ \frac{x_j + z_j}{r_j} \end{bmatrix} \tag{3.8}$$

Sinkronizacija oscilatora vrši se pomoću matrice rotacije $\mathbf{R}(\theta_j^i)$ koja je dana izrazom:

$$\mathbf{R}^{-1}(\theta_j^i) = \begin{bmatrix} \cos\theta_j^i & \sin\theta_j^i \\ -\sin\theta_j^i & \cos\theta_j^i \end{bmatrix} \tag{3.9}$$

Sama faza θ_j^i definira se kao relativna faza odnosno fazno kašnjenje noge j u odnosu na nogu i, pri čemu su $i, j \in \{L1, L2, L3, R1, R2, R3\}$. Pritom vrijedi $\theta_j^i = -\theta_i^j$. Na ovaj način postignuto je upravljanje faktorom popunjenosti i sinkronizacija Hopfovih oscilatora.

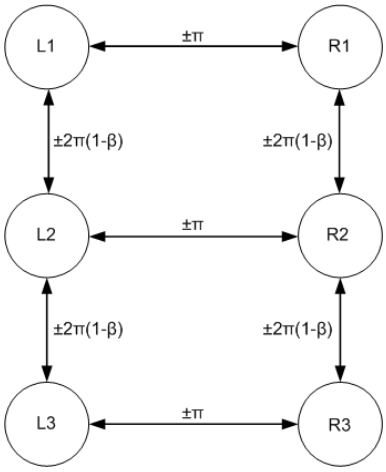
Kako bi bilo moguće kontinuirano mijenjati način gibanja potrebno je odrediti kako se fazna kašnjenja između pojedinih nogu ponašaju u ovisnosti o faktoru popunjenosti β. Uvjete koje mora zadovoljiti neka sekvenca hoda može se prema [9] zapisati i preko faznih kašnjenja na sljedeći način:

$$\theta_j^i = \begin{cases} (1-\beta)2\pi & \text{za noge na istoj strani} \\ \pi & \text{za nasuprotne noge} \end{cases} \tag{3.10}$$

Sada se može definirati matrica relativnih faza koja se koristi prilikom sinkronizacije oscilatora:

$$\begin{array}{c} \\ L1 \\ R1 \\ L2 \\ R2 \\ L3 \\ R3 \end{array} \begin{array}{c} \begin{array}{cccccc} L1 & R1 & L2 & R2 & L3 & R3 \end{array} \\ \begin{pmatrix} 0 & \pi & -2\pi(\beta-1) & -\pi(2\beta-3) & -4\pi(\beta-1) & -\pi(4\beta-5) \\ -\pi & 0 & -\pi(2\beta-1) & -2\pi(\beta-1) & -\pi(4\beta-3) & -4\pi(\beta-1) \\ 2\pi(\beta-1) & \pi(2\beta-1) & 0 & \pi & -2\pi(\beta-1) & -\pi(2\beta-3) \\ \pi(2\beta-3) & 2\pi(\beta-1) & -\pi & 0 & -\pi(2\beta-1) & -2\pi(\beta-1) \\ 4\pi(\beta-1) & \pi(4\beta-3) & 2\pi(\beta-1) & \pi(2\beta-1) & 0 & \pi \\ \pi(4\beta-5) & 4\pi(\beta-1 & \pi(2\beta-3) & 2\pi(\beta-1) & -\pi & 0 \end{pmatrix} \end{array} \tag{3.11}$$

Pojednostavljeni prikaz faznih kašnjenja dan je na slici 3.4. Relacijom (3.8) i matricom faza (3.11) u potpunosti je određen model Hopfovih oscilatora koji se koriste kao generator referentnog zakreta za zglobove koji pomiču noge naprijed natrag. Kako bi generator sekvence bio potpun, potrebno je definirati način na koji se noga podiže odnosno spušta.



Slika 3.4: Prikaz faznih kašnjenja nogu šesteronožnih hodača

Najjednostavniji način je podizanje noge do maksimalne visine za vrijeme dok se noga nalazi u fazi zamaha te spuštanje noge do minimalne visine za vrijeme dok se noga nalazi u fazi oslanjanja na podlogu. Za slučaj gibanja noge prema naprijed, faza zamaha je ona faza u kojoj noga ide od stražnjeg prema prednjem položaju, što znači da je derivacija izlaza iz oscilatora pozitivna. Za vrijeme faze stajanja noga ide od prednjeg prema stražnjem položaju pa je derivacija izlaza iz oscilatora odnosno varijable stanja x_i za određeni oscilator negativna. Ukoliko se sa c_i označi upravljačka varijabla koja označava je li noga podignuta ili spuštena, onda je ona definirana na sljedeći način:

$$c_i = \begin{cases} -1 & \dot{x_i} < 0 \\ 1 & \dot{x_i} > 0 \end{cases} \tag{3.12}$$

pri čemu vrijednost -1 označava da je noga spuštena, a vrijednost 1 da je noga podignuta. Za slučaj gibanja prema natrag upravo je obrnuta situacija. Relacijom (3.8) u potpunosti su određene trajektorije zglobova koji pomiču noge prema naprijed odnosno natrag, a relacijom (3.12) su određene trajektorije zglobova koji podižu odnosno spuštaju noge hodača, čime je u potpunosti određen model 6 sinkroniziranih Hopfovih oscilatora koji se koriste kao generator sekvence za šesteronožnog hodača.

4. Teorija vijčanog gibanja

Dijelovi teorije vijčanog gibanja (engl. *screw theory*) koji se koriste u ovom radu izneseni su u [6] i [12]. Osnovna stvar od koje polazi razvoj teorije uvoj jest promatranje gibanja krutih tijela u prostoru. Transformacija krutih tijela je prikaz gibanja tijela u prostoru, odnosno trenutne brzine, položaja i orijentacije koordinatnog sustava pridruženog tijelu koje se giba. Važno je naglasiti kako se bilo koje gibanje tijela u prostoru može prikazati familijom preslikavanja oblika $g(t) : O \to \mathbb{R}^3$ pri čemu vrijedi $O \subset \mathbb{R}^3$. Skup O može biti sastavljen bilo od vektora bilo od točaka.
Neka je $\mathbf{p}$ točka u $\mathbb{R}^3$ i neka su $[\mathbf{p}]^F$ koordinate točke $\mathbf{p}$ u ortonormiranom koordinatnom sustavu $F = \{x, y, z\}$. Ako je gibanje neke točke $\mathbf{p}$ u koordinatnom sustavu {F} definirano pozicijom u ovisnosti o vremenu t, može se definirati trajektorija točke $\mathbf{p}$:

$$[\mathbf{p}(t)]^F = [p^x(t), p^y(t), p^z(t)]^T \in \mathbb{R}^3 \tag{4.1}$$

Istovremeno gibanje mreže točaka koje su sve fiksirane na tijelu iz početnog položaja u konačni naziva se kruti pomak (engl. *rigid displacement*). Ako kruto tijelo definiramo kao podskup $O \subset \mathbb{R}^3$, kruti pomak je definiran preslikavanjem $g(t) : O \to \mathbb{R}^3$ koje opisuje gibanje tijela odnosno položaj točke u nekom trenutku t u odnosu na ishodišni koordinatni sustav {F}.
Za neko preslikavanje $g(t) : \mathbb{R}^3 \to \mathbb{R}^3$ kažemo da je kruti pomak ako posjeduje sljedeća 2 svojstva:

1. $g(t)$ čuva duljinu: $\|g(\mathbf{p}(t)) - g(\mathbf{q}(t))\| = \|\mathbf{p} - \mathbf{q}\|, \forall \mathbf{p}, \mathbf{q} \in \mathbb{R}^3$
2. $g(t)$ čuva vektorski umnožak: $g(\mathbf{v} \times \mathbf{w}) = g(\mathbf{v}) \times g(\mathbf{w}), \forall \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$

Kao posljedica dva navedena svojstva javlja se i očuvanje skalarnog produkta. Očuvanje skalarnog produkta zajedno sa svojstvima transformacije krutih tijela znači da se ortogonalni vektori primjenom transformacije prevode u ortogonalne vektore, pri čemu orijentacija među njima ostaje nepromijenjena. Zbog toga se transformacija krutih tijela može koristiti i kao reprezentacija koordinatnih sustava manipulatora, jer će

se desno orijentiran koordinatni sustav uvijek preslikati u desno orijentirani koordinatni sustav. Očuvanje duljine, vektorskog i skalarnog umnoška prilikom primjene transformacije g osnova je primjene teorije zakreta u robotskim sustavima.

4.1. Rotacija u $\mathbb{R}^3$

Prilikom rješavanja kinematičkih problema, prema konvenciji Denavita i Hartenberga, zglobovima i člancima manipulatora pridružuju se desno orijentirani ortonormirani koordinatni sustavi. U skladu s općom praksom, svi koordinatni sustavi u ovom radu biti će desno orijentirani i ortonormirani. Neka pokretni koordinatni sustav {B} rotira oko fiksnog koordinatnog sustava {F} i neka su $[\mathbf{x}_B]^F, [\mathbf{y}_B]^F, [\mathbf{z}_B]^F \in \mathbb{R}^3$ koordinate osiju koordinatnog sustava {B} iskazane u odnosu na {F}. Rotacija koordinatnog sustava {B} u odnosu na {F} definira se kao rotacijska matrica dimenzija 3×3:

$$\mathbf{R}_F^B = [[\mathbf{x}_B]^F], [\mathbf{y}_B]^F], [\mathbf{z}_B]^F]] \in \mathbb{R}^{3\times 3} \tag{4.2}$$

pri čemu oznaka $\mathbb{R}^{3\times 3}$ predstavlja skup matrica dimenzija 3×3 s realnim koeficijentima. Iz relacije (4.2) uočava se kako su stupci matrice $\mathbf{R}_F^B$ međusobno ortogonalni pa je i sama matrica $\mathbf{R}_F^B$ ortogonalna. Općenito, ortogonalne matrice posjeduju dva bitna svojstva:

1. $\mathbf{R}\mathbf{R}^T = \mathbf{I} \ \rightarrow \ \mathbf{R}^{-1} = \mathbf{R}^T$
2. $det\mathbf{R} = \pm 1$

Kako u ovom slučaju matrica $\mathbf{R}$ predstavlja desno orijentirani ortonormirani koordinatni sustav {B} zapisan u koordinatama {F}, njena determinanta je uvijek 1, pa se može definirati skup specijalno ortogonalnih matrica $SO(3) \subset \mathbb{R}^{3\times 3}$:

$$SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times 3} \mid \mathbf{R}\mathbf{R}^T = \mathbf{I}, det\mathbf{R} = 1\} \tag{4.3}$$

Ovako definiran skup $SO(3)$ je grupa s obzirom na matrično množenje, pa je umnožak bilo koje dvije matrice iz tog skupa ponovno matrica iz tog skupa, što znači da se kompozicija dviju rotacija može prikazati umnoškom dviju matrica rotacije. Ukoliko matrica $\mathbf{R}_F^B$ predstavlja rotaciju koordinatnog sustava {B} u odnosu na {F}, a matrica $\mathbf{R}_B^C$ rotaciju koordinatnog sustava {C} u odnosu na {B}, onda se rotacija {C} u odnosu na {F} zapisuje u obliku $\mathbf{R}_F^C$ pri čemu je:

$$\mathbf{R}_F^C = \mathbf{R}_F^B \mathbf{R}_B^C \tag{4.4}$$

U [12] je pokazano kako je $SO(3)$ rotacijska grupa u prostoru $\mathbb{R}^3$, što znači da se konfiguracija bilo kojeg tijela koje rotira može prikazati matricom $\mathbf{R} \in SO(3)$, koja se promatra kao transformacija koja preslikava koordinate iz početnih (prije početka gibanja) u konačne koordinate odnosno konfiguraciju nakon završetka gibanja, što odgovara klasičnoj definiciji matrice rotacije. Jednako tako rotacijska matrica $\mathbf{R}_F^B \in \mathbb{R}^3$ transformira koordinate točke $\mathbf{p}^B$ iz pokretnog koordinatnog sustava {B} u fiksni {F} na način:

$$[\mathbf{p}]^F = \mathbf{R}_F^B[\mathbf{p}]^B \ , \ [\mathbf{p}]^B = [[p_x]^B, [p_y]^B, [p_z]^B] \tag{4.5}$$

Pritom je bitno naglasiti kako je rotacija $\mathbf{R} \in SO(3)$ transformacija krutog tijela jer zadovoljava dva postavljena uvjeta:

1. $\mathbf{R}$ čuva duljinu: $\|\mathbf{Rp} - \mathbf{Rq}\| = \|\mathbf{p} - \mathbf{q}\|, \ \mathbf{p}, \mathbf{q} \in \mathbb{R}^3$
2. $\mathbf{R}$ čuva vektorski umnožak: $\mathbf{R}(\mathbf{v} \times \mathbf{w}) = (\mathbf{Rv}) \times (\mathbf{Rw}), \ \mathbf{v}, \mathbf{w} \in \mathbb{R}^3$

4.1.1. Eksponencijalne koordinate za rotaciju

Neka tijelo rotira konstantnom jediničnom brzinom oko osi $[\omega]^F$. Neka je točka $[\mathbf{q}]^F$ pričvršćena na tijelo. Tada je brzina točke $[\mathbf{q}]^F$ dana izrazom:

$$[\dot{\mathbf{q}}]^F(t) = [\omega]^F \times [\mathbf{q}]^F(t) \tag{4.6}$$

Prema teoriji linearne algebre, vektorski umnožak je linearan operator te se može prikazati u obliku matrice. Za općeniti vektorski umnožak vektora $\mathbf{a} = [a_1 \ a_2 \ a_3]^T$ i $\mathbf{b} = [b_1 \ b_2 \ b_3]^T \in \mathbb{R}^3$ matrica operatora vektorskog umnoška dana je izrazom (4.7) [4]:

$$\mathbf{a} \times \mathbf{b} = \mathbf{Ab} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \mathbf{b} \tag{4.7}$$

Matrica $\mathbf{A}$ pripada klasi antisimetričnih matrica za koje vrijedi $\mathbf{A} = -\mathbf{A}^T$. Može se definirati i skup svih takvih matrica $so(3)$ (vektorski prostor nad realnim brojevima):

$$so(3) = \{\mathbf{S} \in \mathbb{R}^{3\times3} | \mathbf{S}^T = -\mathbf{S}\} \tag{4.8}$$

Prema oznakama iz [12] vektorski umnožak iz relacije (4.6) može se prikazati u obliku matričnog množenja na sljedeći način:

$$[\dot{\mathbf{q}}]^F(t) = [\omega]^F \times [\mathbf{q}]^F(t) = [\hat{\omega}]^F[\mathbf{q}]^F(t) \tag{4.9}$$

pri čemu se $[\hat{\omega}]^F$ definira s obzirom na $[\omega]^F = [\omega_1^F\ \omega_2^F\ \omega_3^F]^T \in \mathbb{R}^3$ u skladu s relacijom (4.7) na sljedeći način:

$$[\hat{\omega}]^F = \begin{bmatrix} 0 & -\omega_3^F & \omega_2^F \\ \omega_3^F & 0 & -\omega_1^F \\ -\omega_2^F & \omega_1^F & 0 \end{bmatrix} \tag{4.10}$$

Rješenje diferencijalne jednadžbe (4.9) dano je izrazom (4.11):

$$[\mathbf{q}]^F(t) = e^{[\hat{\omega}]^F t}[\mathbf{q}]^F(0) \tag{4.11}$$

pri čemu se $e^{\hat{\omega}t}$ računa razvojem u Taylorov red:

$$e^{[\hat{\omega}]^F t} = \mathbf{I} + [\hat{\omega}]^F t + \frac{([\hat{\omega}]^F t)^2}{2!} + \frac{([\hat{\omega}]^F t)^3}{3!} + \cdots \tag{4.12}$$

Za klasu antisimetričnih matrica $so(3)$ kojima pripada $\hat{\omega}$ vrijede sljedeća svojstva [12]:

1. $\hat{\omega}^2 = \omega\omega^T - \|\omega\|^2\mathbf{I}$
2. $\hat{\omega}^3 = -\|\omega\|^2\hat{\omega}$
3. Ostale potencije se izračunavaju rekurzivno

Koristeći navedena svojstva, izraz (4.12) može se analitički izračunati te vrijedi (uzimajući u obzir $t = \|[\omega]^F\|\theta,\ \|[\omega]^F\| = 1$):

$$e^{[\hat{\omega}]^F\theta} = \mathbf{I} + [\hat{\omega}]^F \sin\theta + ([\hat{\omega}]^F)^2[1 - \cos\theta] \tag{4.13}$$

Izraz (4.13) naziva se Rodriguesova formula za izračun matrice rotacije $\mathbf{R}$. Ukoliko vektor $[\omega]^F$ nije jediničnog iznosa izraz (4.13) se piše u obliku:

$$e^{[\hat{\omega}]^F\theta} = \mathbf{I} + \frac{[\hat{\omega}]^F}{\|[\omega]^F\|}\sin\left(\|[\omega]^F\|\theta\right) + \frac{([\hat{\omega}]^F)^2}{\|[\omega]^F\|^2}[1 - \cos\left(\|[\omega]^F\|\theta\right)] \tag{4.14}$$

Rodriguesova formula za izračunavanje rotacijske matrice $\mathbf{R} \in SO(3)$ može se prikazati u matričnom obliku:

$$\begin{bmatrix} \cos\theta + \omega_1^2(1-\cos\theta) & \omega_1\omega_2(1-\cos\theta) - \omega_3\sin\theta & \omega_2\sin\theta + \omega_1\omega_3(1-\cos\theta) \\ \omega_3\sin\theta + \omega_1\omega_2(1-\cos\theta) & \cos\theta + \omega_2^2(1-\cos\theta) & -\omega_1\sin\theta + \omega_1\omega_2(1-\cos\theta) \\ -\omega_2\sin\theta + \omega_1\omega_3(1-\cos\theta) & \omega_1\sin\theta + \omega_1\omega_2(1-\cos\theta) & \cos\theta + \omega_3^2(1-\cos\theta) \end{bmatrix} \tag{4.15}$$

Može se pokazati kako se eksponencijalnim preslikavanjem $e^{[\hat{\omega}]^F\theta}$, uz $\|[\omega]^F\| = 1$, dobije matrica rotacije u prostoru $\mathbb{R}^3$, tj. da eksponencijalno preslikavanje prevodi antisimetrične matrice iz skupa $so(3)$ u ortogonalne matrice iz skupa $SO(3)$ pa vrijedi:

$$e^{[\hat{\omega}]^F\theta} = \mathbf{R}([\omega]^F, \theta) \tag{4.16}$$

Antisimetrična matrica $[\hat{\omega}]^F$ je infinitezimalni generator grupe $SO(3)$. Geometrijska interpretacija relacije (4.16) je sljedeća:

- Antisimetrična matrica $[\hat{\omega}]^F$ predstavlja matrični zapis jediničnog vektora $[\omega]^F$ koji pokazuje u smjeru osi rotacije
- Eksponencijalnim preslikavanjem $e^{[\hat{\omega}]^F\theta}$ generira se zakret oko vektora $[\omega]^F$ za kut θ

Pritom se komponente vektora $[\omega]^F\theta \in \mathbb{R}^3$ nazivaju eksponencijalnim koordinatama matrice rotacije $\mathbf{R}(\omega, \theta)$.
Geometrijska interpretacija zapravo predstavlja pojednostavljeni iskaz Eulerova teorema koji glasi:
Svaka orijentacija $\mathbf{R} \in SO(3)$ *je ekvivalentna rotaciji oko neke fiksne osi* $[\omega] \in \mathbb{R}^3$ *za neki kut* $\theta \in \mathbb{R}$.
Preslikavanje $exp : so(3) \to SO(3)$ je surjektivno pa se za svaku matricu $\mathbf{R} \in SO(3)$ može pronaći par (ω, θ) takav da vrijedi relacija (4.16). Ukoliko je zadana matrica $\mathbf{R}$ jedno od rješenja dano je u [12]:

$$\theta = \arccos \frac{tr(\mathbf{R}) - 1}{2} \tag{4.17}$$

$$[\omega] = \frac{1}{2\sin\theta} \begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix} \tag{4.18}$$

pri čemu su r_{ij} , $i, j \in \{1, 2, 3\}$ komponente matrice $\mathbf{R}$, a $tr(\mathbf{R})$ trag matrice $\mathbf{R}$. Iz izraza (4.17) i (4.18) može se uočiti sljedeće:

- Postoji više mogućih rješenja koja su posljedica višeznačnosti funkcije $\arccos$
- Ukoliko se dobije $\theta = k\pi$ (što je slučaj za $\mathbf{R} = \mathbf{I}$) nije moguće odrediti ω jer se u nazivniku pojavljuje 0. To znači da postoji singularitet reprezentacije.

4.2. Gibanje u $\mathbb{R}^3$

Kako se bilo koje gibanje sastoji od rotacije i translacije, potrebno je prikazati poziciju i orijentaciju jednog koordinatnog sustava u drugom koordinatnom sustavu. Zbog toga je nužno definirati reprezentaciju koja uključuje i translaciju i rotaciju (za razliku od čiste rotacije). Kako se konfiguracija sustava sastoji od vektora pozicije i njegove orijentacije (koja je sadržana unutar matrice rotacije), definira se skup uređenih parova

vektora pozicije i matrice orijentacije u obliku specijalne Euklidske grupe $SE(3)$ na sljedeći način:

$$SE(3) = \{(\mathbf{p}, \mathbf{R}) : \mathbf{p} \in \mathbb{R}^3 \ , \ \mathbf{R} \in SO(3)\} \tag{4.19}$$

pri čemu je $\mathbf{p}$ vektor kojim je određen smjer i iznos translacije dok je matricom $\mathbf{R}$ određena os rotacije te iznos za koji se tijelo rotira. Slično kao i kod skupa $SO(3)$, bilo koji $\mathbf{T} \in SE(3)$ je transformacija krutog tijela jer čuva udaljenosti i orijentaciju.

Reprezentacija u homogenom prostoru

Neka u prostoru $\mathbb{R}^3$ transformacija $\mathbf{T} = ([\mathbf{p}]^F, \mathbf{R}) \in SE(3)$ djeluje na neku točku $[\mathbf{q}]^F$. Djelovanje transformacije $\mathbf{T}$ na točku $[\mathbf{q}]^F$ dano je izrazom:

$$\mathbf{T}[\mathbf{q}]^F = [\mathbf{p}]^F + \mathbf{R}[\mathbf{q}]^F \tag{4.20}$$

U izrazu (4.20) vidi se kako je transformacija $\mathbf{T}$ afina transformacija, odnosno nije linearna te se kao takva ne može prikazati u obliku matrice. Međutim, ona se može prikazati u linearnom obliku ukoliko se prijeđe na homogene koordinate. Time se omogućuje matrični zapis transformacije $\mathbf{T}$.
Točke u homogenom prostoru prikazuju se na način da im se doda četvrta komponenta koja je najčešće 1, a takav zapis naziva se homogenim koordinatama točke $\mathbf{q}$:

$$[\mathbf{q}]^F = \begin{bmatrix} q_1^F \\ q_2^F \\ q_3^F \end{bmatrix} \rightarrow \begin{bmatrix} q_1^F \\ q_2^F \\ q_3^F \\ 1 \end{bmatrix} \tag{4.21}$$

Kako se vektor definira kao razlika dviju točaka, četvrta komponenta za vektore je 0:

$$[\mathbf{v}]^F = \begin{bmatrix} v_1^F \\ v_2^F \\ v_3^F \end{bmatrix} \rightarrow \begin{bmatrix} v_1^F \\ v_2^F \\ v_3^F \\ 0 \end{bmatrix} \tag{4.22}$$

Posljedica ovakvog zapisa su sljedeća svojstva koja vrijede u homogenom prostoru:

- Razlika dviju točkaka je vektor
- Zbroj i razlika dvaju vektora je vektor
- Zbroj točke i vektora je točka

– Zbroj dviju točaka nema smisla

Prelaskom u homogene koordinate izraz (4.20) se može pisati na sljedeći način:

$$\mathbf{T}\begin{bmatrix}[\mathbf{q}]^F\\1\end{bmatrix}=\begin{bmatrix}\mathbf{R} & [\mathbf{p}]^F\\\mathbf{0} & 1\end{bmatrix}\begin{bmatrix}[\mathbf{q}]^F\\1\end{bmatrix} \tag{4.23}$$

odakle slijedi homogeni odnosno linearni zapis transformacije $\mathbf{T}$ u matričnom obliku:

$$\mathbf{T}=\begin{bmatrix}\mathbf{R} & [\mathbf{p}]^F\\\mathbf{0} & 1\end{bmatrix} \tag{4.24}$$

Zapis transformacija iz skupa $SE(3)$ u homogenim koordinatama omogućuje promatranje tog skupa kao grupe s obzirom na matrično množenje, što je pokazano u [12], gdje je također pokazano kako se kompozicija transformacija može prikazati kao umnožak pripadajućih matrica. Prelazak u homogene koordinate na neki način *linearizira* prostor te omogućuje zapis transformacija u obliku matrica, što je posebno važno prilikom implementacija raznih algoritama na digitalnim računalima. Važno je dodati kako je svojstvo očuvanja duljina i vektorskog umnoška očuvano prelaskom na homogene koordinate.

4.2.1. Eksponencijalne koordinate za kruto gibanje u $\mathbb{R}^3$

Neka tijelo rotira oko osi određene jediničnim vektorom $[\omega]^F \in \mathbb{R}^3, \|[\omega]^F\| = 1$. Neka je $[\mathbf{q}]^F \in \mathbb{R}^3$ jedna točka na osi rotacije, a $[\mathbf{p}]^F(t) \in \mathbb{R}^3$ položaj neke točke koja je čvrsto vezana uz tijelo koje rotira, pri čemu su koordinate točaka $[\mathbf{p}]^F$ i $[\mathbf{q}]^F$ te vektora $[\omega]^F$ izražene u istom koordinatnom sustavu {F}. Brzina točke $[\mathbf{p}]^F$ u tom istom koordinatnom sustavu dana je sljedećim izrazom:

$$[\dot{\mathbf{p}}]^F(t)=[\omega]^F\times[[\mathbf{p}]^F(t)-[\mathbf{q}]^F]=[\omega]^F\times[\mathbf{p}]^F(t)-[\omega]^F\times[\mathbf{q}]^F=[\hat{\omega}]^F[\mathbf{p}]^F(t)+[\mathbf{v}]^F \tag{4.25}$$

pri čemu je $[\mathbf{v}]^F=-[\omega]^F\times[\mathbf{q}]^F$. Prethodno je pokazano kako prelazak na homogene koordinate *linearizira* sustav odnosno omogućuje zapis transformacije u matričnom obliku, zbog čega je poželjno i izraz (4.25) zapisati u homogenim koordinatama. Prelazak na homogene koordinate vrši se na sljedeći način:

– Prema relaciji (4.21) točki $[\mathbf{p}]^F$ se dodaje 1 kao četvrta komponenta

– Derivacija točke je vektor pa se u $[\dot{\mathbf{p}}]^F$ kao četvrta komponenta dodaje 0

– $[\mathbf{v}]^F$ je vektor pa je njegova četvrta komponenta 0

Zapis relacije (4.25) u homogenim koordinatama dan je sljedećim izrazom:

$$\begin{bmatrix} [\dot{\mathbf{p}}]^F(t) \\ 0 \end{bmatrix} = \begin{bmatrix} [\hat{\omega}]^F & [\mathbf{v}]^F \\ \mathbf{0} & 0 \end{bmatrix} \begin{bmatrix} [\mathbf{p}]^F(t) \\ 1 \end{bmatrix} \tag{4.26}$$

Ukoliko se zapis vektora i točaka u homogenim koordinatama poistovjeti sa samim točkama u prostoru, može se pisati:

$$[\dot{\mathbf{p}}]^F(t) = \begin{bmatrix} [\hat{\omega}]^F & [\mathbf{v}]^F \\ \mathbf{0} & 0 \end{bmatrix} [\mathbf{p}]^F(t) = [\hat{\xi}]^F [\mathbf{p}]^F(t) \tag{4.27}$$

pri čemu matrica $[\hat{\xi}]^F = \begin{bmatrix} [\hat{\omega}]^F & [\mathbf{v}]^F \\ \mathbf{0} & 0 \end{bmatrix}$ predstavlja proširenje odnosno generalizaciju skupa antisimetričnih matrica $so(3)$ s ciljem uključivanja translacijskog gibanja u promatranje gibanja tijela u prostoru. Skup svih takvih matrica definira se kao uređeni par $([\hat{\omega}]^F, [\mathbf{v}]^F)$ na sljedeći način:

$$se(3) = \{([\omega]^F, [\mathbf{v}]^F) \ : \ [\hat{\omega}]^F \in so(3), [\mathbf{v}]^F \in \mathbb{R}^3\} \tag{4.28}$$

Matrica $[\hat{\xi}]^F \in se(3)$ reda 4 naziva se uvoj (engl. *twist*) ili infinitezimalni generator Euklidske grupe $SE(3)$.
Rješenje diferencijalne jednadžbe (4.27) dano je sljedećim izrazom:

$$[\mathbf{p}]^F(t) = e^{[\hat{\xi}]^F t} [\mathbf{p}]^F(0) \tag{4.29}$$

pri čemu se ponovno izraz $e^{[\hat{\xi}]^F t}$ računa razvojem u Taylorov red:

$$e^{[\hat{\xi}]^F t} = \mathbf{I} + [\hat{\xi}]^F t + \frac{([\hat{\xi}]^F t)^2}{2!} + \cdots \tag{4.30}$$

Da je uvoj generator Euklidske grupe pokazano je u [12], gdje je izračunat izraz (4.30) te je rješenje za $e^{[\hat{\xi}]^F t}$ uz zamjenu $t = \theta \in \mathbb{R}$ dano izrazom:

$$e^{[\hat{\xi}]^F \theta} = \begin{bmatrix} e^{[\hat{\omega}]^F \theta} & (\mathbf{I} - e^{[\hat{\omega}]^F \theta})([\omega]^F \times [\mathbf{v}]^F) + [\omega]^F [[\omega]^F]^T [\mathbf{v}]^F \theta \\ \mathbf{0} & 1 \end{bmatrix} \tag{4.31}$$

Prethodno je relacijom (4.16) pokazano kako je $e^{[\hat{\omega}]^F \theta}$ matrica rotacije $\mathbf{R}$ dimenzija 3×3 iz čega se može zaključiti kako je matrica u izrazu (4.31) dimenziija 4x4 pa je izraz (4.31) moguće zapisati u obliku:

$$e^{[\hat{\xi}]^F \theta} = \begin{bmatrix} \mathbf{R} & [\mathbf{p}]^F \\ \mathbf{0} & 1 \end{bmatrix} \tag{4.32}$$

uz $[\mathbf{p}] = (\mathbf{I} - e^{[\hat{\omega}]^F\theta})([\omega]^F \times [\mathbf{v}]^F) + [\omega]^F[[\omega]^F]^T[\mathbf{v}]^F\theta$, što odgovara relaciji (4.24) čime je pokazano da se eksponencijalnim preslikavanjem neke matrice $[\hat{\xi}]^F \in se(3)$ dobije matrica iz skupa $SE(3)$, odnosno da je matrica uvoja $[\hat{\xi}]^F$ generator Euklidske grupe $SE(3)$. Iz uvoja je moguće jednoznačno izdvojiti vektore $[\omega]^F$ i $[\mathbf{v}]^F$. Izdvojeni vektori se mogu zapisati u obliku vektora $[\xi]^F \in \mathbb{R}^6$:

$$[\xi]^F = \begin{bmatrix} [\mathbf{v}]^F \\ [\omega]^F \end{bmatrix} \tag{4.33}$$

Vektor $[\xi]^F$ naziva se uvojnim ili eksponencijalnim koordinatama uvoja $[\hat{\xi}]^F$ i moguće ga je povezati s Plückerovim koordinatama linije u prostoru.

Plückerove koordinate linije

Iz relacije (4.27) može se uočiti kako je uvoj u potpunosti određen uređenim parom $([\omega]^F, [\mathbf{v}]^F)$. Uređeni par $([\omega]^F, [\mathbf{v}]^F)$ se u literaturi ([6] i naziva Plückerovim koordinatama linije. Ukoliko se os gibanja (pri čemu se gibanje može sastojati i od rotacije i od translacije) promatra kao linija u prostoru, onda se Plückerove koordinate linije definiraju na sljedeći način:
Neka je $[\mathbf{q}]^F$ točka na promatranoj liniji te neka je $[\omega]^F$ jedinični vektor koji pokazuje u smjeru linije, odnosno u smjeru gibanja. Linija je jednoznačno određena vektorom smjera $[\omega]^F$ i prvim momentom. Prvi moment linije u odnosu na ishodište[1] *definira se kao vektorski umnožak $[\mathbf{v}]^F = [\mathbf{q}]^F \times [\omega]^F$. Uz tako definiran $[\mathbf{v}]^F$ uređeni par $([\omega]^F, [\mathbf{v}]^F)$ daje 6 Plückerovih koordinata linije.*

Ponovnim uvidom u izraz (4.25) uočava se kako se jednostavnim razmatranjem gibanja tijela u prostoru došlo do Plückerovih koordinata (potrebno uočiti kako je, zbog svojstava vektorskog produkta, $-[\omega]^F \times [\mathbf{q}]^F = [\mathbf{q}]^F \times [\omega]^F$).

Iako je jasno kako se eksponencijalnim preslikavanjem uvoja generira gibanje po i oko linije u prostoru, sama geometrijska interpretacija uvoja $[\hat{\xi}]^F$ nije intuitivna kao primjerice eksponencijalno preslikavanje antisimetričnih matrica. Jasno je kako $[\hat{\xi}]^F$ sadrži informaciju o osi gibanja odnosno liniji u obliku uređenog para $([\omega]^F, [\mathbf{v}]^F)$ odnosno Plückerovih koordinata linije, i dok je značenje vektora $[\omega]^F$ geometrijski intuitivno i jasno ($[\omega]^F$ pokazuje u smjeru gibanja), geometrijska interpretacija vektora $[\mathbf{v}]^F$ nije sasvim intuitivna. Kako bi se dobila geometrijska slika uvoja, potrebno je povezati uvoje i vijčana gibanja.

[1] Moment linije može se definirati u odnosu na bilo koju točku u prostoru

4.3. Eksponencijalne koordinate za vijčano gibanje

Prema [12] vijčano gibanje (engl. *screw motion*) je gibanje koje se sastoji od rotacije oko neke osi i istovremene translacije po toj istoj osi (u kontekstu Plückerovih koordinata, os gibanja je isto što i linija u prostoru). Definicija vijka i vijčanog gibanja, preuzeta iz [12] glasi:

*Vijak (*engl. *screw) S se sastoji od osi l, uspona h i amplitude M. Vijčano gibanje predstavlja rotaciju oko osi l za iznos koji je jednak $\theta = M$ te translaciju po osi l za iznos $d = hM = h\theta$.* [2]*Ukoliko je uspon vijka $h \to \infty$, tada se vijak sastoji od isključivo translacijskog gibanja te pomak po liniji l iznosi $d = M$ dok je iznos rotacije $\theta = 0$. Ukoliko je uspon vijka $h = 0$ vijak se sastoji samo od rotacijskog gibanja i vrijedi $d = 0,\ \theta = M$.*

Neka je $[\mathbf{p}]^F$ točka u prostoru i neka je $[\omega]^F$ jedinični vektor koji pokazuje u smjeru osi l, a točka $[\mathbf{q}]^F$ se nalazi na osi gibanja, čime je os gibanja jednoznačno određena. Točka $[\mathbf{p}]^F$ se može zapisati kao zbroj točke $[\mathbf{q}]^F$ i vektora $[\mathbf{p}]^F - [\mathbf{q}]^F$: [3]

$$[\mathbf{p}]^F = [\mathbf{q}]^F + ([\mathbf{p}]^F - [\mathbf{q}]^F) \tag{4.34}$$

Neka sada na točku $[\mathbf{p}]^F$ djeluje transformacija $\mathbf{R}_F^B(\theta)$ koja ima svojstvo rotacije, odnosno neka točka $[\mathbf{p}]^F$ rotira oko osi gibanja za kut θ. Tada vrijedi:

$$\mathbf{R}_F^B[\mathbf{p}]^F = \mathbf{R}_F^B[\mathbf{q}]^F + \mathbf{R}_F^B([\mathbf{p}]^F - [\mathbf{q}]^F) \tag{4.35}$$

Točka $[\mathbf{q}]^F$ se nalazi na osi rotacije, pa vrijedi $\mathbf{R}_F^B[\mathbf{q}]^F = [\mathbf{q}]^F$. Prema relaciji (4.16) vrijedi $\mathbf{R}(\theta, \omega) = e^{[\hat{\omega}]^F\theta}$ pa se izraz (4.35) može pisati u obliku:

$$\mathbf{R}_F^B[\mathbf{p}]^F = [\mathbf{q}]^F + e^{[\hat{\omega}]^F\theta}([\mathbf{p}]^F - [\mathbf{q}]^F) = e^{[\hat{\omega}]^F\theta}[\mathbf{p}]^F + (\mathbf{I} - e^{[\hat{\omega}]^F\theta})[\mathbf{q}]^F \tag{4.36}$$

Neka se istovremeno s rotacijom točke $[\mathbf{p}]^F$ oko l izvrši i translacija po l u smjeru $[\omega]^F$ za iznos $d = h\theta$, što odgovara vijčanom gibanju s amplitudom θ. Ukoliko se ovakva transformacija označi s $\mathbf{T}_F^B$ vrijedi:

$$\mathbf{T}_F^B[\mathbf{p}]^F = \mathbf{R}_F^B[\mathbf{p}]^F + h[\omega]^F\theta = e^{[\hat{\omega}]^F\theta}[\mathbf{p}]^F + (\mathbf{I} - e^{[\hat{\omega}]^F\theta})[\mathbf{q}]^F + h[\omega]^F\theta \tag{4.37}$$

Matrični zapis relacije (4.37) u homogenim koordinatama dan je sljedećim izrazom:

$$\mathbf{T}_F^B \begin{bmatrix} [\mathbf{p}]^F \\ 1 \end{bmatrix} = \begin{bmatrix} e^{[\hat{\omega}]^F\theta} & (\mathbf{I} - e^{[\hat{\omega}]^F\theta})[\mathbf{q}]^F + h[\omega]^F\theta \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} [\mathbf{p}]^F \\ 1 \end{bmatrix} \tag{4.38}$$

[2] Uspon vijka $h = \frac{d}{\theta}$ je dakle omjer iznosa translacijskog i rotacijskog gibanja

[3] Zbog kasnijeg prelaska u homogene koordinate gdje je zbroj točke i vektora točka

Kako izraz (4.38) vrijedi za sve $[\mathbf{p}]^F \in \mathbb{R}^3$, vijčano gibanje se može prikazati u obliku matrice transformacije:

$$\mathbf{T}_F^B = \begin{bmatrix} e^{[\hat{\omega}]^F\theta} & (\mathbf{I} - e^{[\hat{\omega}]^F\theta})[\mathbf{q}]^F + h[\omega]^F\theta \\ \mathbf{0} & 1 \end{bmatrix} \tag{4.39}$$

Uspoređujući izraze za transformaciju vijčanog gibanja (4.39) i eksponencijale uvoja (4.31), može se uočiti kako izrazi postaju identični ako se vektor $[\mathbf{v}]^F$ iz izraza (4.31) zamijeni s $-[\omega]^F \times [\mathbf{q}]^F + h[\omega]^F$:

$$[\mathbf{v}]^F = -[\omega]^F \times [\mathbf{q}]^F + h[\omega]^F \tag{4.40}$$

Uz $[\mathbf{v}]^F$ definiran prema izrazu (4.40) uvoj $[\hat{\xi}]^F = ([\mathbf{v}]^F, [\omega])^F$ s eksponencijalnim koordinatama $[\xi]^F = \begin{bmatrix} [\mathbf{v}]^F \\ [\omega]^F \end{bmatrix}$ generira upravo vijčano gibanje.

Vijčanom gibanju pripadaju i čisto rotacijsko i čisto translacijsko gibanje kao najčešći oblici gibanja zglobova u robotskim manipulatorima. U skladu s definicijom vijčanog gibanja može se pisati sljedeće:

- Rotacijski zglob: $h = 0 \rightarrow [\mathbf{v}]^F = -[\omega]^F \times [\mathbf{q}]^F$ pa je uvoj povezan sa zadanim gibanjem u eksponencijalnim koordinatama dan s $[\xi]^F = \begin{bmatrix} -[\omega]^F \times [\mathbf{q}]^F \\ [\omega]^F \end{bmatrix}$
- Translacijski zglob: $h = \infty \rightarrow [\omega]^F = 0$ pa je uvoj povezan sa zadanim gibanjem u eksponencijalnim koordinatama dan s $[\xi]^F = \begin{bmatrix} [\mathbf{v}]^F \\ 0 \end{bmatrix}$, pri čemu je $[\mathbf{v}]^F$ jedinični vektor koji pokazuje u smjeru gibanja

Iako je u primjenama u robotici puno češći postupak određivanja uvoja za neko zadano vijčano gibanje, moguće je definirati i obrat odnosno vijčane koordinate uvoja.

Vijčane koordinate uvoja

Neka je $[\hat{\xi}]^F \in se(3)$ uvoj s pripadnim eksponencijalnim koordinatama $[\xi]^F = ([\mathbf{v}]^F, [\omega]^F) \in \mathbb{R}^6$, pri čemu se ne pretpostavlja $\|[\omega]^F\| = 1$. Vijčane koordinate uvoja su:

- Uspon (omjer translacijskog i rotacijskog gibanja):

$$h = \frac{[[\omega]^F]^T[\mathbf{v}]^F}{\|[\omega]^F\|^2} \tag{4.41}$$

  Ako je $\omega = 0$ uvoj ima beskonačan uspon, odnosno opisuje translacijsko gibanje.

– Os odnosno linija l:

$$l = \begin{cases} \left\{ \frac{[\omega]^F \times [\mathbf{v}]^F}{\|[\omega]^F\|^2} + \lambda [\omega]^F \ : \ \lambda \in \mathbb{R} \right\} \ , \ [\omega]^F \neq 0 \\ \left\{ 0 + \lambda [\mathbf{v}]^F \ : \ \lambda \in \mathbb{R} \right\} \ , \ [\omega]^F = 0 \end{cases} \tag{4.42}$$

Ako je $[\omega]^F \neq 0$ os l je usmjerena linija u smjeru $[\omega]^F$ koja prolazi točkom $[\mathbf{q}]^F = \frac{[\omega]^F \times [\mathbf{v}]^F}{\|[\omega]^F\|^2}$ [4] dok je za $\omega = 0$ os usmjerena linija u smjeru $\mathbf{v}$ koja prolazi kroz ishodište.

– Amplituda (ako se gibanje sastoji isključivo od translacije odgovara iznosu translacije, u suprotnome odgovara iznosu rotacije):

$$M = \begin{cases} \|[\omega]^F\| \ , \ [\omega]^F \neq 0 \\ \|[\mathbf{v}]^F\| \ , \ [\omega]^F = 0 \end{cases} \tag{4.43}$$

Ukoliko se odabere $\|[\omega]^F\| = 1$ u slučaju kada gibanje sadrži i rotaciju, odnosno $\|[\mathbf{v}]^F\| = 1$ u slučaju kada se radi o čistoj translaciji, uvoj $[\hat{\xi}]^F$ ima jediničnu amplitudu i zove se jedinični uvoj.

Potvrda primjenjivosti teorije zakreta u robotici dolazi od Chaslesova teorema koji glasi:

Svako gibanje krutog tijela se može ostvariti kao kombinacija rotacije oko neke osi i translacije po toj istoj osi.

Prema tome, bilo koje gibanje u prostoru može se promatrati kao vijčano gibanje. Jednako tako, važno je naglasiti kako se za svako vijčano gibanje određeno s osi l, usponom h i amplitudom M može pronaći jedinični uvoj $\hat{\xi}$ takav da se zadano gibanje može generirati uvojem $\hat{\xi}M$.

4.4. Rješavanje direktnog kinematičkog problema

Definicija problema direktne kinematike, preuzeta iz [11], glasi:

Ako je zadan vektor varijabli zglobova robotskog manipulatora, tada treba odrediti položaj i orijentaciju alata prema mirnom koordinatnom sustavu pridruženom bazi robota.

Neka postoji robotski manipulator s n zglobova, bilo rotacijskih bilo translacijskih. Neka je bazi manipulatora pridružen koordinatni sustav {F}, a alatu koordinatni sustav {B}. Početni položaj odnosno konfiguracija manipulatora neka je zadana za sve

[4] Prema relaciji (4.25) vrijedi $[\mathbf{v}]^F = -[\omega]^F \times [\mathbf{q}]^F$, pa je onda $[\omega]^F \times [\mathbf{v}]^F = [\omega]^F \times (-[\omega]^F \times [\mathbf{q}]^F)$ iz čega slijedi $[\mathbf{q}]^F = \frac{[\omega]^F \times [\mathbf{v}]^F}{\|[\omega]^F\|^2}$

varijable zglobova jednake 0. Tada je matricom homogene transformacije $\mathbf{T}_F^B(\mathbf{0})$ dana konfiguracija odnosno položaj i orijentacija alata u početnom položaju. Svakom zglobu odnosno osi gibanja manipulatora može se pridružiti uvoj $[\hat{\xi}_i]^F,\ i \in \{1, 2, 3,\ ...,\ n\}$. Prethodno je pokazano kako se eksponencijalnim preslikavanjem $e^{[\hat{\xi}_i]^F\theta_i}$ generira gibanje po liniji određenoj s $[\hat{\xi}_i]^F$, pri čemu je relacijom (4.29) pokazano kako se množenjem $e^{[\hat{\xi}_i]^F\theta_i}$ s početnom konfiguracijom dobije konfiguracija nakon što se gibanje i-tog zgloba završi. Tako je nakon zakreta n-tog zgloba manipulatora konfiguracija alata dana izrazom:

$$e^{[\hat{\xi}_n]^F\theta_n}\mathbf{T}_F^B(\mathbf{0}) \tag{4.44}$$

Izraz (4.44) sada predstavlja početnu konfiguraciju za gibanje ($n-1$)-og zgloba. Nakon završetka gibanja $(n-1)$-og zgloba konfiguracija sustava dana je izrazom:

$$e^{[\hat{\xi}_{n-1}]^F\theta_{n-1}}e^{[\hat{\xi}_n]^F\theta_n}\mathbf{T}_F^B(\mathbf{0}) \tag{4.45}$$

Prolazeći na takav način od n-og zgloba prema prvom, konfiguracija alata u obliku matrice homogene transformacije $\mathbf{T}_F^B$ nakon gibanja svih n zglobova manipulatora dana je izrazom:

$$\mathbf{T}_F^B = e^{[\hat{\xi}_1]^F\theta_1}e^{[\hat{\xi}_2]^F\theta_2}\cdots e^{[\hat{\xi}_n]^F\theta_n}\mathbf{T}_F^B(\mathbf{0}) = \prod_{i=1}^{n} e^{[\hat{\xi}_i]^F\theta_i}\mathbf{T}_F^B(\mathbf{0}) \tag{4.46}$$

Izraz (4.46) funkcija je svih varijabli zglobova i predstavlja rješenje direktnog kinematičkog problema za općeniti robotski manipulator. U skladu s provedenim razmatranjem, može se postupak rješavanja direktnog kinematičkog problema algoritmizirati. Postupak se sastoji od sljedećih koraka:

1. Pridružiti desno orijentirane koordinatne sustave bazi robota {F} i alatu {B}

2. Odrediti početnu konfiguraciju sustava u obliku matrice homogene transformacije $\mathbf{T}_F^B(\mathbf{0})$ (uz sve varijable zglobova jednake 0)

3. Za svaku os gibanja odrediti jedinični vektor $[\vec{o}]^F$ koji pokazuje u smjeru osi gibanja te točku $\mathbf{q}^F$ koja se nalazi na osi gibanja

4. Ukoliko je i-ti zglob rotacijski, eksponencijalne koordinate uvoja koji je povezan s tim zglobom se određuju na sljedeći način:

$$[\omega_i]^F = [\vec{o}]^F\ ,\ [\mathbf{v}_i]^F = \mathbf{q}^F \times [\omega]^F\ ,\ [\xi_i]^F = \begin{bmatrix}[\mathbf{v}_i]^F\\ [\omega_i]^F\end{bmatrix} \tag{4.47}$$

5. Ukoliko je i-ti zglob translacijski, eksponencijalne koordinate uvoja koji je povezan s tim zglobom se određuju na sljedeći način:

$$[\omega_i]^F = \mathbf{0}\ ,\ [\mathbf{v}_i]^F = [\vec{o}]^F\ ,\ [\xi_i]^F = \begin{bmatrix} [\mathbf{v}_i]^F \\ \mathbf{0} \end{bmatrix} \tag{4.48}$$

6. Iz eksponencijalnih koordinata $[\xi_i]^F$ formirati uvoje $[\hat{\xi}_i]^F$

7. Ukupna transformacija u {F} se određuje kao kompozicija, odnosno umnožak transformacija:

$$\mathbf{T}_F = \prod_{i=1}^{n} e^{[\hat{\xi}_i]^F \theta_i} = e^{[\hat{\xi}_1]^F \theta_1} e^{[\hat{\xi}_2]^F \theta_2} \dots \tag{4.49}$$

   pri čemu su $\theta_i\ ,\ i = 1, 2, ...$ varijable zglobova.

8. Matrica homogene transformacije dobije se množenjem izraza (4.49) s početnom konfiguracijom manipulatora (s desna):

$$\mathbf{T}_F^B = \mathbf{T}_F \mathbf{T}_F^B(\mathbf{0}) \tag{4.50}$$

4.5. Brzina gibanja tijela

Moguće je promatrati i brzine gibanja tijela u prostoru. Neka je {F} mirni koordinatni sustav (engl. *spatial coordinate frame*), a {B} pokretni koordinatni sustav pridružen tijelu koje se giba (engl. *body coordinate frame*).

4.5.1. Rotacijska brzina u $\mathbb{R}^3$

Neka je točka u prostoru $[\mathbf{q}]^B$ čvrsto vezana na tijelo i neka su njene koordinate izražene u sustavu {B} koji rotira oko neki osi. Putanja točke $[\mathbf{q}]^B$ u koordinatnom sustavu {F} dana je izrazom:

$$[\mathbf{q}]^F(t) = \mathbf{R}_F^B(t)[\mathbf{q}]^B \tag{4.51}$$

Rotacijska brzina točke $[\mathbf{q}]^F$ u koordinatama sustava {F} dobije se deriviranjem izraza (4.51):

$$[v_\mathbf{q}]^F(t) = \frac{d}{dt}[\mathbf{q}]^F(t) = \dot{\mathbf{R}}_F^B(t)[\mathbf{q}]^B \tag{4.52}$$

Iz izraza (4.52) može se uočiti kako $\dot{\mathbf{R}}_F^B(t)$ preslikava koordinate položaja u kutnu brzinu. Neefikasnost ovakvog načina zapisa posljedica je dimenzija matrice $\dot{\mathbf{R}}_F^B(t)$

koje su 3×3 pa je potrebno čak 9 parametara za opisivanje rotacijske brzine. Iz izraza (4.52) slijedi (uz izostavljanje funkcionalne ovisnosti o vremenu radi preglednosti):

$$[v_{\mathbf{q}}]^F = \dot{\mathbf{R}}_F^B (\mathbf{R}_F^B)^{-1} (\mathbf{R}_F^B)[\mathbf{q}]^B \tag{4.53}$$

Uvrštavanjem (4.51) u (4.53) dobije se:

$$[v_{\mathbf{q}}]^F = \dot{\mathbf{R}}_F^B (\mathbf{R}_F^B)^{-1} [\mathbf{q}]^F \tag{4.54}$$

Za kompaktniji i efikasniji zapis može se iskoristiti činjenica da je matrica $\mathbf{R}_F^B$ iz skupa specijalno ortogonalnih matrica $SO(3)$ za koje vrijede sljedeća dva bitna svojstva:

1. $\dot{\mathbf{R}}_F^B (\mathbf{R}_F^B)^{-1} \in so(3)$
2. $(\mathbf{R}_F^B)^{-1} \dot{\mathbf{R}}_F^B \in so(3)$

Koristeći ta dva svojstva može se izraz (4.54) pisati na sljedeći način:

$$[v_{\mathbf{q}}]^F = [\Omega_F^B]^F [\mathbf{q}]^F \tag{4.55}$$

pri čemu je $[\Omega_F^B]^F = \dot{\mathbf{R}}_F^B (\mathbf{R}_F^B)^{-1} \in so(3)$ antisimetrična matrica koja predstavlja trenutnu prostornu kutnu brzinu (engl. *instantaneous spatial angular velocity*) koordinatnog sustava {B} u koordinatama sustava{F}.[5]
Slično kao i u fiksnom koordinatnom sustavu {F}, kutna brzina se može definirati i u koordinatnom sustavu tijela {B}. Sličnim razmatranjem dobije se trenutna kutna brzina u koordinatnom sustavu {B} (engl. *instantaneous body angular velocity*) u obliku antisimetrične matrice $[\Omega_F^B]^B$:

$$[\Omega_F^B]^B = (\mathbf{R}_F^B)^{-1} \dot{\mathbf{R}}_F^B \tag{4.56}$$

Moguće je povezati trenutnu kutnu brzinu u {F} s trenutnom kutnom brzinom u {B}:

$$[\Omega_F^B]^B = (\mathbf{R}_F^B)^{-1} [\Omega_F^B]^F \mathbf{R}_F^B \;\rightarrow\; [\omega_F^B]^B = (\mathbf{R}_F^B)^{-1} [\omega_F^B]^F \tag{4.57}$$

4.5.2. Brzina u $\mathbb{R}^3$

Neka je točka u prostoru $[\mathbf{q}]^B$ čvrsto vezana na tijelo i neka su njene koordinate izražene u sustavu {B} koji se giba po nekom uvoju u koordinatnom sustavu {F}. Putanja točke $[\mathbf{q}]^B$ u koordinatnom sustavu {F} dana je izrazom:

$$[\mathbf{q}]^F(t) = \mathbf{T}_F^B(t) [\mathbf{q}]^B \tag{4.58}$$

[5] Izraz (4.55) se, ako se iz $[\Omega_F^B]^F$ izdvoji kutna brzina $[\omega_F^B]^F$, može pisati u obliku $v_{\mathbf{q}}^F = [\omega_F^B]^F \times \mathbf{q}^F$, što odgovara klasičnoj definiciji kutne brzine.

Brzina gibanja koordinatnog sustava u odnosu na {F} u koordinatama sustava {F} dobije se deriviranjem izraza (4.58):

$$[V_{\mathbf{q}}]^F(t) = \frac{d}{dt}[\mathbf{q}]^F(t) = \dot{\mathbf{T}}_F^B(t)[\mathbf{q}]^B \tag{4.59}$$

Može se pisati:

$$[V_{\mathbf{q}}]^F = \dot{\mathbf{T}}_F^B(\mathbf{T}_F^B)^{-1}[\mathbf{q}]^F \tag{4.60}$$

Derivacija matrice homogene transformacije $\mathbf{T}_F^B = \begin{bmatrix} \mathbf{R}_F^B & [\mathbf{p}]_F^B \\ \mathbf{0} & 1 \end{bmatrix}$ nije prikladan način zapisa brzine tijela. Analogno s razmatranjem kutnih brzina, potrebno je promotriti matricu:

$$\dot{\mathbf{T}}_F^B(\mathbf{T}_F^B)^{-1} = \begin{bmatrix} \dot{\mathbf{R}}_F^B(\mathbf{R}_F^B)^T & -\dot{\mathbf{R}}_F^B(\mathbf{R}_F^B)^T[\mathbf{p}]_F^B + [\dot{\mathbf{p}}]_F^B \\ \mathbf{0} & 0 \end{bmatrix} \tag{4.61}$$

za koju se može uočiti kako ima oblik matrice uvoja (vidi relaciju (4.27)) pa se može definirati prostorna[6] brzina koordinatnog sustava {B} u odnosu na {F} u obliku uvoja:

$$[\hat{V}_F^B]^F = \dot{\mathbf{T}}_F^B(\mathbf{T}_F^B)^{-1} = \begin{bmatrix} \dot{\mathbf{R}}_F^B(\mathbf{R}_F^B)^T & -\dot{\mathbf{R}}_F^B(\mathbf{R}_F^B)^T[\mathbf{p}]_F^B + [\dot{\mathbf{p}}]_F^B \\ \mathbf{0} & 0 \end{bmatrix} \tag{4.62}$$

Eksponencijalne koordinate uvoja $[\hat{V}_F^B]^F$ zapisuju se u obliku:

$$[V_F^B]^F = \begin{bmatrix} [v_F^B]^F \\ [\omega_F^B]^F \end{bmatrix} \tag{4.63}$$

pri čemu je $[v_F^B]^F = -\dot{\mathbf{R}}_F^B(\mathbf{R}_F^B)^T[\mathbf{p}]_F^B + [\dot{\mathbf{p}}]_F^B$, a $[\omega_F^B]^F$ se dobije izdvajanjem komponenti vektora iz antisimetrične matrice trenutne kutne brzine $[\Omega_F^B]^F = \dot{\mathbf{R}}_F^B(\mathbf{R}_F^B)^T$.
Geometrijska interpretacija uvoja povezanog s brzinom gibanja tijela u prostoru je:

- Kutna komponenta brzine $[\omega_F^B]^F$ je trenutna kutna brzine tijela odnosno koordinatnog sustava {B} u odnosu na {F} promatrana iz koordinatnog sustava {F}
- Linijska komponenta brzine $[v_F^B]^F$ nije trenutna linijska brzina sustava {B} u odnosu na {F} već linijska brzina točke (moguće i nepostojeće) koja je čvrsto vezana za koordinatni sustav {B} te prolazi ishodištem koordinatnog sustava {F}, izražena u koordinatama {F}.

Analogno razmatranju za brzinu u koordinatnom sustavu {F}, moguće je definirati i brzinu gibanja izraženu u koordinatama sustava{B}:

$$[\hat{V}_F^B]^B = (\mathbf{T}_F^B)^{-1}\dot{\mathbf{T}}_F^B = \begin{bmatrix} (\dot{\mathbf{R}_F^B})^T\mathbf{R}_F^B & (\mathbf{R}_F^B)^T[\dot{\mathbf{p}}]_F^B \\ \mathbf{0} & 0 \end{bmatrix} \tag{4.64}$$

[6]Termin prostorna označava brzinu u fiksnom koordinatnom sustavu

Eksponencijalne koordinate uvoja $[\hat{V}_F^B]^B$ dobiju se slično kao i za prostorne brzine te se zapisuju na sljedeći način:

$$[V_F^B]^B = \begin{bmatrix} [v_F^B]^B \\ [\omega_F^B]^B \end{bmatrix} \tag{4.65}$$

Geometrijska interpretacija za uvoj $[\hat{V}_F^B]^B$ je slična kao i za $[\hat{V}_F^B]^F$, s razlikom da su brzine izražene u koordinatama sustava {B}.
Veza između $[\hat{V}_F^B]^B$ i $[\hat{V}_F^B]^F$ je dana izrazom:

$$[\hat{V}_F^B]^F = \mathbf{T}_F^B [\hat{V}_F^B]^B (\mathbf{T}_F^B)^{-1} \tag{4.66}$$

4.5.3. Transformacija koordinata uvoja - adjungirana matrica transformacije

Prema izvodu provedenom u [12], izraz (4.66) je zapisan u eksponencijalnim koordinatama($[V_F^B]^B$ i $[V_F^B]^F$) i izračunat u obliku:

$$[V_F^B]^F = \begin{bmatrix} [v_F^B]^F \\ [\omega_F^B]^F \end{bmatrix} = \begin{bmatrix} \mathbf{R}_F^B & [\hat{\mathbf{p}}_F^B]^F \mathbf{R}_F^B \\ \mathbf{0} & \mathbf{R}_F^B \end{bmatrix} \begin{bmatrix} [v_F^B]^B \\ [\omega_F^B]^B \end{bmatrix} \tag{4.67}$$

pri čemu se antisimetrična matrica $[\hat{\mathbf{p}}_F^B]^F$ dobije iz vektora $[\mathbf{p}_F^B]$ prema relaciji (4.7). Matrica koja vrši transformaciju eksponencijalnih koordinata uvoja iz jednog koordinatnog sustava u drugi zove se adjungirana matrica transformacije. Ona je usko vezana uz matricu homogene transformacije pa se uvijek i definira s obzirom na neku transformaciju $\mathbf{T}_F^B$, te se označava s $\mathbf{Ad}_{\mathbf{T}_F^B}$, a njen oblik i elementi se mogu iščitati iz izraza (4.67):

$$\mathbf{Ad}_{\mathbf{T}_F^B} = \begin{bmatrix} \mathbf{R}_F^B & [\hat{\mathbf{p}}_F^B]^F \mathbf{R}_F^B \\ \mathbf{0} & \mathbf{R}_F^B \end{bmatrix} \tag{4.68}$$

Inverz adjungirane matrice transformacije dan je relacijom:

$$(\mathbf{Ad}_{\mathbf{T}_F^B})^{-1} = \begin{bmatrix} (\mathbf{R}_F^B)^T & -(\mathbf{R}_F^B)^T [\hat{\mathbf{p}}_F^B] \\ \mathbf{0} & (\mathbf{R}_F^B)^T \end{bmatrix} \tag{4.69}$$

te se može pokazati kako vrijedi:

$$(\mathbf{Ad}_{\mathbf{T}_F^B})^{-1} = \mathbf{Ad}_{(\mathbf{T}_F^B)^{-1}} \tag{4.70}$$

Brzine u prostoru se mogu brže i efikasnije izračunati direktnim korištenjem uvoja

pridruženih osima gibanja, jer proračun ne zahtjeva operacije deriviranja. Neka je $[\hat{\xi}]^F$ jedinični uvoj pridružen nekoj osi gibanja u prostoru. Neka je transformacija koju generira taj uvoj $\mathbf{T}_F^B$. Prema [12], za brzinu u fiksnom koordinatnom sustavu {F} vrijedi:

$$[\hat{V}_F^B]^F = \dot{\mathbf{T}}_F^B(\mathbf{T}_F^B)^{-1} = [\hat{\xi}]^F\dot{\theta} \tag{4.71}$$

Jednako tako za brzinu u koordinatnom sustavu tijela {B} vrijedi:

$$[\hat{V}_F^B]^B = (\mathbf{T}_F^B)^{-1}\dot{\mathbf{T}}_F^B = \left(\mathbf{Ad}_{(\mathbf{T}_F^B)^{-1}}[\xi]^F\right)^\wedge \dot{\theta} = [\hat{\xi}]^B\dot{\theta} \tag{4.72}$$

pri čemu $[\xi]^B = (\mathbf{Ad}_{(\mathbf{T}_F^B)^{-1}}[\xi]^F)$ predstavlja eksponencijalne koordinate uvoja $[\hat{\xi}]^F$ transformirane u koordinatni sustav {B}, dok izraz $[\hat{\xi}]^B = (\mathbf{Ad}_{(\mathbf{T}_F^B)^{-1}}[\xi]^F)^\wedge$ predstavlja uvoj s eksponencijalnim koordinatama $[\xi]^B$.

Transformacija brzina

Neka se tri koordinatna sustava {A}, {B} i {C} gibaju u prostoru. Ukoliko je poznat vektor $[V_A^B]^A$ (brzina {B} u odnosu na {A} u koordinatama sustava {A}) i vektor $[V_B^C]^B$ (brzina {C} u odnosu na {B} u koordinatama sustava {B}) onda se brzina {C} u odnosu na {A} u koordinatama {A} može odrediti prema izrazu:

$$[V_A^C]^A = [V_A^B]^A + \mathbf{Ad}_{\mathbf{T_A^B}}[V_B^C]^B \tag{4.73}$$

pri čemu je $\mathbf{T}_A^B$ matrica transformacije koja povezuje sustave {A} i {B}. Brzina {C} u odnosu na {A} u koordinatama {C} se određuje na sljedeći način:

$$[V_A^C]^C = \mathbf{Ad}_{(\mathbf{T_B^C})^{-1}}\left([V_B^C]^B + \mathbf{Ad}_{(\mathbf{T_A^B})^{-1}}[V_A^B]^A\right) \tag{4.74}$$

pri čemu je $\mathbf{T}_B^C$ matrica transformacije koja povezuje koordinatne sustave {B} i {C}.

4.6. Jacobian

Jacobian je matrica koja povezuje brzinu alata s brzinama zglobova. Brzina gibanja alata za neki manipulator se može izračunati iz rješenja direktne kinematike. Neka je $\mathbf{T}_F^B$ rješenje direktne kinematike za neki manipulator. Prema [12] i izrazu (4.71) brzina alata u koordinatnom sustavu baze {F} je dana izrazom:

$$[\hat{V}_F^B]^F = \dot{\mathbf{T}}_F^B\left(\mathbf{T}_F^B\right)^{-1} \tag{4.75}$$

Derivacija matrice transformacije se može izračunati na sljedeći način:

$$\dot{\mathbf{T}}_F^B = \sum_{i=1}^{n}\left(\frac{\partial \mathbf{T}_F^B}{\partial\theta_i}\dot{\theta}_i\right) \tag{4.76}$$

pa se izraz (4.75) može pisati u obliku:

$$[\hat{V}_F^B]^F = \sum_{i=1}^{n} \left(\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \dot{\theta}_i \right) \left(\mathbf{T}_F^B \right)^{-1} = \sum_{i=1}^{n} \left(\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \left(\mathbf{T}_F^B \right)^{-1} \right) \dot{\theta}_i \tag{4.77}$$

Matrica $\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \left(\mathbf{T}_F^B \right)^{-1}$ ima oblik uvoja pa se može zapisati u eksponencijalnim koordinatama. Ukoliko se cijeli izraz (4.77) prebaci u eksponencijalne koordinate, može se zapisati u matričnom obliku:

$$[\hat{V}_F^B]^F = \begin{bmatrix} [\xi_1']^F & [\xi_2']^F & \cdots & [\xi_n']^F \end{bmatrix} \dot{\theta} \tag{4.78}$$

pri čemu $[\xi_i']^F$ odgovara eksponencijalnim koordinatama uvoja $\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \left(\mathbf{T}_F^B \right)^{-1}$. Matrica $[\mathbf{J}_F^B]^F = \begin{bmatrix} [\xi_1']^F & [\xi_2']^F & \cdots & [\xi_n']^F \end{bmatrix}$ naziva se prostorni Jacobian manipulatora. Na sličan način se polazeći od izraza:

$$[\hat{V}_F^B]^B = \left(\mathbf{T}_F^B \right)^{-1} \dot{\mathbf{T}}_F^B \tag{4.79}$$

može izračunati Jacobian u koordinatama koordinatnog sustava {B} koji je pridružen alatu, $[\mathbf{J}_F^B]^B = \begin{bmatrix} [\xi_1']^B & [\xi_2']^B & \cdots & [\xi_n']^B \end{bmatrix}$.
Jacobian se znatno jednostavnije može izračunati pomoću uvoja i eksponencijalnih koordinata. Prema relacijama (4.49) i (4.50) vrijedi:

$$\mathbf{T}_F^B = e^{[\hat{\xi}_1]^F \theta_1} e^{[\hat{\xi}_2]^F \theta_2} \cdots e^{[\hat{\xi}_n]^F \theta_n} \mathbf{T}_F^B(0) \tag{4.80}$$

Izraz (4.75) se može pisati u obliku:

$$[\hat{V}_F^B]^F = \dot{\mathbf{T}}_F^B \left(\mathbf{T}_F^B \right)^{-1} = \left(\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \dot{\theta}_i \right) \left(\mathbf{T}_F^B \right)^{-1} \tag{4.81}$$

Vrijedi:

$$\left(\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \right) = e^{[\hat{\xi}_1]^F \theta_1} \cdots e^{[\hat{\xi}_{i-1}]^F \theta_{i-1}} \left(\frac{\partial e^{[\hat{\xi}_i]^F \theta_i}}{\partial \theta_i} \right) e^{[\hat{\xi}_{i+1}]^F \theta_{i+1}} \cdots e^{[\hat{\xi}_n]^F \theta_n} \mathbf{T}_F^B(0) \tag{4.82}$$

Prema pravilu za derivaciju eksponencijalne funkcije, može se pisati:

$$\left(\frac{\partial \mathbf{T}_F^B}{\partial \theta_i} \right) = e^{[\hat{\xi}_1]^F \theta_1} \cdots e^{[\hat{\xi}_{i-1}]^F \theta_{i-1}} \left([\hat{\xi}_i]^F e^{[\hat{\xi}_i]^F \theta_i} \right) e^{[\hat{\xi}_{i-1}]^F \theta_{i-1}} \cdots e^{[\hat{\xi}_n]^F \theta_n} \mathbf{T}_F^B(0) \tag{4.83}$$

Prema pravilu o inverziji umnoška matrica može se, uzimajući u obzir izraz (4.80), izračunati $\left(\mathbf{T}_F^B \right)^{-1}$ na sljedeći način:

$$\left(\mathbf{T}_F^B \right)^{-1} = \left(\mathbf{T}_F^B(0) \right)^{-1} e^{-[\hat{\xi}_n]^F \theta_n} \cdots e^{-[\hat{\xi}_2]^F \theta_2} e^{-[\hat{\xi}_1]^F \theta_1} \tag{4.84}$$

Uvrštavanjem (4.83) i (4.84) u (4.81) nakon sređivanja dobije se sljedeći izraz:

$$[\hat{V}_F^B]^F = \sum_{i=1}^{n} \left(e^{[\hat{\xi}_1]^F \theta_1} \cdots e^{[\hat{\xi}_{i-1}]^F \theta_{i-1}} [\hat{\xi}_i]^F e^{-[\hat{\xi}_{i-1}]^F \theta_{i-1}} \cdots e^{-[\hat{\xi}_1]^F \theta_1} \right) \dot{\theta}_i \tag{4.85}$$

Prelaskom u eksponencijalne koordinate, prethodni izraz može se pisati u matričnom obliku:

$$[V_F^B]^F = \begin{bmatrix} [\xi_1']^F & [\xi_2']^F & \cdots & [\xi_n']^F \end{bmatrix} \dot{\theta} = [\mathbf{J}_F^B]^F \dot{\theta} \tag{4.86}$$

pri čemu je:

$$[\xi_i']^F = \left(e^{[\hat{\xi}_1]^F \theta_1} \cdots e^{[\hat{\xi}_{i-1}]^F \theta_{i-1}} [\hat{\xi}_i]^F e^{-[\hat{\xi}_{i-1}]^F \theta_{i-1}} \cdots e^{-[\hat{\xi}_1]^F \theta_1} \right)^{\vee} \tag{4.87}$$

Slično kao i kod proračuna direktne kinematike, izrazom (4.87) dane su eksponencijalne koordinate pripadajućeg uvoja $[\hat{\xi}_i]^F$ u koordinatama sustava {F} nakon što se izvrši gibanje zglobova $1, 2, ...(i-1)$. $[\xi_i']^F$ može se računati i na sljedeći način:

$$[\xi_i']^F = \mathbf{Ad}_{\left(e^{[\hat{\xi}_1]^F \theta_1 \cdots e^{[\hat{\xi}_{i-1}]^F \theta_{i-1}}}\right)} [\xi_i]^F \tag{4.88}$$

gdje su $[\xi_i]^F$ eksponencjialne koordinate pripadajućeg uvoja u početnom položaju manipulatora, odnosno prije početka gibanja. Izrazom (4.88) su zapravo definirane eksponencijalne koordinate svih uvoja preslikane u trenutnu konfiguraciju manipulatora. Relacijama (4.77) i (4.86) definiran je prostorni Jacobian, odnosno Jacobian manipulatora u koordinatama nepomičnog koordinatnog sustava {F} pridruženog bazi manipulatora. Jacobian alata, odnosno Jacobian manipulatora u koordinatama pokretnog sustava {B}, koji je pridružen alatu, može se izračunati tako da se transformirani uvoji iz matrice prostornog Jacobiana preslikaju u koordinate sustava {B} prema relaciji (4.67), pri čemu je potrebno poznavati matricu homogene transformacije $\mathbf{T}_F^B$ koja preslikava koordinate iz {B} u {F}, odnosno prethodno riješiti direktni kinematički problem za zadani manipulator. Nakon transformiranja koordinata uvoja u koordinatni sustav {B}, Jacobian alata se formira na sljedeći način:

$$[\mathbf{J}_F^B]^B = \begin{bmatrix} [\xi_1']^B & [\xi_2']^B & \cdots & [\xi_n']^B \end{bmatrix} \tag{4.89}$$

pri čemu vrijedi:

$$[\xi_i']^B = \mathbf{Ad}_{\left(\mathbf{T}_\mathbf{F}^\mathbf{B}\right)^{-1}} [\xi_i']^F \tag{4.90}$$

Prostorni Jacobian i Jacobian u koordinatnom sustavu alata mogu se povezati adjungiranom matricom transformacije:

$$[\mathbf{J}_F^B]^B = \mathbf{Ad}_{\left(\mathbf{T}_\mathbf{F}^\mathbf{B}\right)^{-1}} [\mathbf{J}_F^B]^F \ , \ [\mathbf{J}_F^B]^F = \mathbf{Ad}_{\mathbf{T}_\mathbf{F}^\mathbf{B}} [\mathbf{J}_F^B]^B \tag{4.91}$$

5. Kinematika šesteronožnog hodača

U ovom poglavlju određena je direktna i inverzna kinematika šesteronožnog hodača zasnovana na teoriji vijčanog gibanja, pri čemu se svaka noga šesteronožnog hodača promatra kao robotski manipulator s dva stupnja slobode gibanja.

5.1. Direktna kinematika

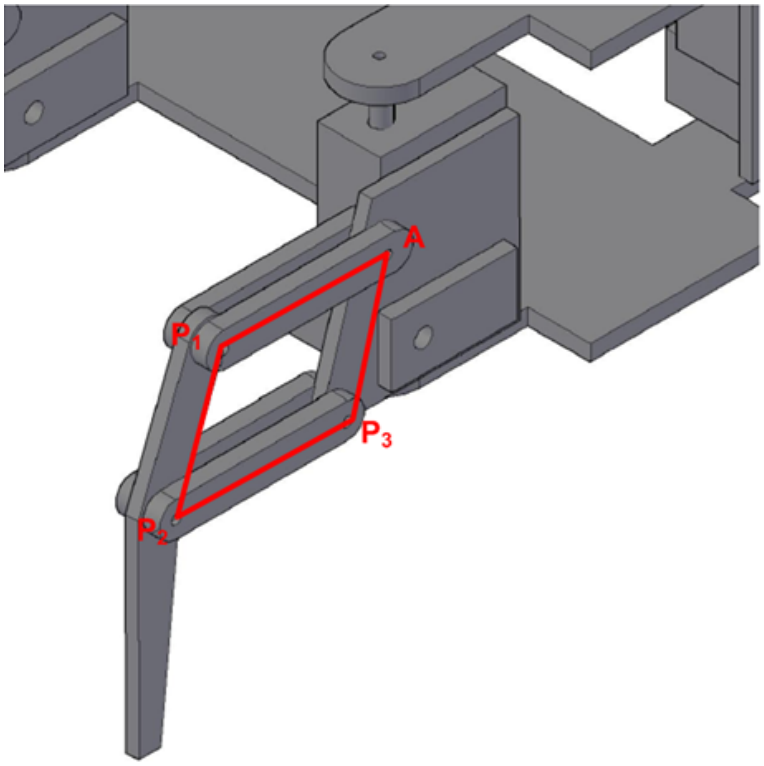
Direktni kinematički problem riješen je prema razmatranju provedenom u prethodnom poglavlju, s određenim modifikacijama koje su nužne zbog toga što noga hodača nije otvoreni kinematički lanac, već se prema slici 5.1 na njoj nalazi zatvoreni kinematički lanac, koji se sastoji od aktivnog zgloba A i tri pasivna zgloba (P_1, P_2, P_3). Pritom članci među navedenim zglobovima zatvaraju paralelogram.[1]

5.1.1. Zatvoreni kinematički lanac i virtualni aktivni zglob

Ukoliko se iskoristi činjenica da članci koji povezuju zglobove zatvorenog lanca zatvaraju paralelogram, zatvoreni lanac se može svesti na otvoreni kinematički lanac i to uvođenjem virtualnog aktivnog zgloba, na način da se gibanje jednog pasivnog zgloba opiše kao gibanje nekog od aktivnih zglobova u kinematičkom lancu. Promatrajući primjerice zglob P_1, može se uočiti kako se, zbog svojstva paralelograma da je zbroj susjednih kutova $180°$, zglob P_1 vrti u suprotnom smjeru od aktivnog zgloba A i to za identičan iznos. Prilikom rješavanja direktnog kinematičkog problema zglob P_1 je proglašen za virtualni aktivni zglob. Samo pridjeljivanje aktivnog zakreta zglobu P_1 može se izvršti na dva načina (odabir je proizvoljan jer su rezultati identični):

1. Orijentirati os kroz zglob P_1 u istom smjeru kao i kroz zglob A te mu pridjeliti zakret $-\theta_A$

[1] Zbog pogrešaka u konstrukciji ne radi se zapravo o paralelogramu, ali su odstupanja dovoljno mala da ih se može zanemariti.



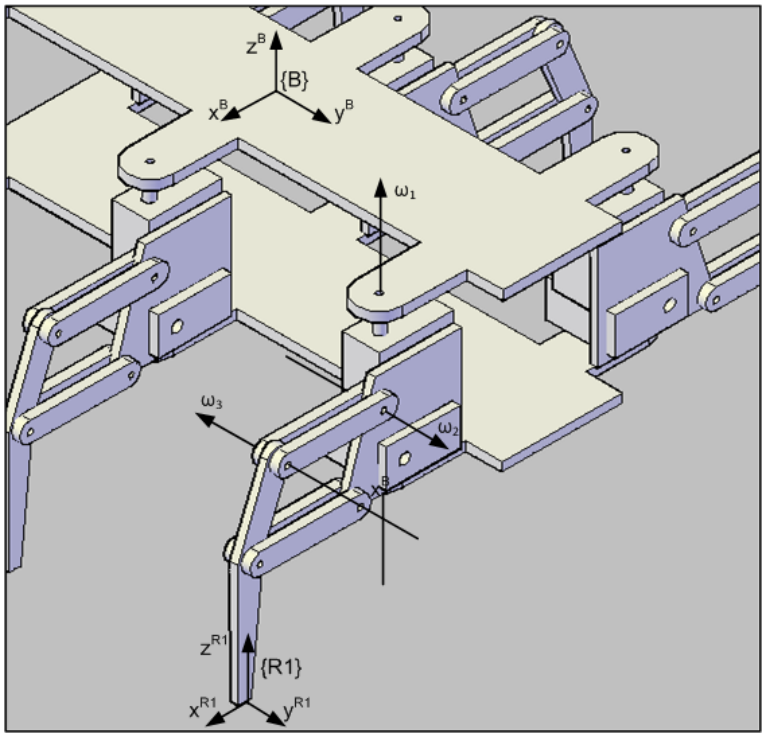
Slika 5.1: Zatvoreni kinematički lanac noge hodača

2. Orijentirati os kroz zglob P_1 u smjeru suprotnom od osi kroz zglob A te mu pridjeliti zakret θ_A

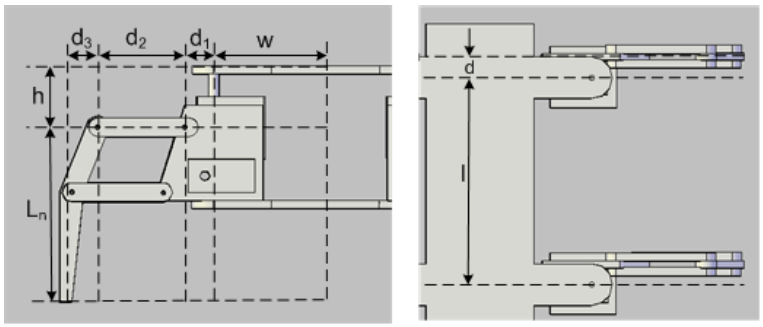
Na slici 5.2 prikazane su osi zglobova odnosno kinematički lanac jedne noge hodača (prednja desna, R1). Na slici 5.3 prikazani su kinematički parametri za kinematički lanac noge R1. Slično su definirane i osi rotacije za ostale noge hodača.
U skladu s postupkom rješevanja direktne kinematike određeni su vektori pridruženi osima, točke na osima i eksponencijalne koordinate svih uvoja pridruženih pojedinim zglobovima. Popis određenih uvoja zapisanih u eksponencijalnim koordinatama dan je u tablici 5.1. Vrhovima nogu koordinatni sustavi su pridruženi na način da im se osi podudaraju s osima baznog koordinatnog sustava, pa je matrica rotacije za svaku od nogu jednaka jediničnoj matrici. Stoga matrica homogene transformacije za početni položaj i-te noge ima oblik:

$$\mathbf{T}_B^i(0) = \begin{bmatrix} \mathbf{I} & [\mathbf{p}_i]^F \\ \mathbf{0} & 1 \end{bmatrix} \tag{5.1}$$

pri čemu je $i \in \{L1, R1, L2, R2, L3, R3\}$. Vektori početnih položaja za pojedine noge $[\mathbf{p}_i]^B$ dani su u tablici 5.2.



Slika 5.2: Prikaz osiju za nogu R1



Slika 5.3: Kinematički parametri

Kao rješenje direktne kinematike dobiju se matrice homogene transformacije koje prikaziju konfiguraciju koordinatnog sustava pridruženog vrhu noge u koordinatnom sustavu baze. Matrice su dane u nastavku.

$$\mathbf{T}_B^{L1} = \begin{bmatrix} \cos q_1^{L1} & -\sin q_1^{L1} & 0 & -w-(d_1+d_2\cos q_2^{L1}+d_3)\cos q_1^{L1} - d\sin q_1^{L1} \\ \sin q_1^{L1} & \cos q_1^{L1} & 0 & l-(d_1+d_2\cos q_2^{L1}+d_3)\sin q_1^{L1} + d\cos q_1^{L1} \\ 0 & 0 & 1 & h - L_n + d_2 \sin q_2^{L1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

Tablica 5.1: Uvoji određeni pri rješavanju direktne kinematike

Noga	$[\xi_1]^B$	$[\xi_2]^B$	$[\xi_3]^B$
L1	$[l, w, 0, 0, 1]^T$	$[-h, 0, -d_1 - w, 0, 1, 0]^T$	$[h, 0, d_1 + d_2 + w, 0, -1, 0]^T$
R1	$[l, -w, 0, 0, 1]^T$	$[-h, 0, d_1 + w, 0, 1, 0]^T$	$[h, 0, -d_1 - d_2 - w, 0, -1, 0]^T$
L2	$[0, w, 0, 0, 1]^T$	$[-h, 0, -d_1 - w, 0, 1, 0]^T$	$[h, 0, d_1 + d_2 + w, 0, -1, 0]^T$
R2	$[0, -w, 0, 0, 1]^T$	$[-h, 0, d_1 + w, 0, 1, 0]^T$	$[h, 0, -d_1 - d_2 - w, 0, -1, 0]^T$
L3	$[-l, w, 0, 0, 1]^T$	$[-h, 0, d_1 - w, 0, 1, 0]^T$	$[h, 0, d_1 + d_2 + w, 0, -1, 0]^T$
R3	$[-l, -w, 0, 0, 1]^T$	$[-h, 0, d_1 + w, 0, 1, 0]^T$	$[h, 0, -d_1 - d_2 - w, 0, -1, 0]^T$

Tablica 5.2: Vektori početnog položaja vrhova nogu

Noga	$[\mathbf{p}_i]^F$
L1	$[-(w + d_1 + d_2 + d_3), l + d, h - L_n]^T$
R1	$[w + d_1 + d_2 + d_3, l + d, h - L_n]^T$
L2	$[-(w + d_1 + d_2 + d_3), d, h - L_n]^T$
R2	$[w + d_1 + d_2 + d_3, d, h - L_n]^T$
L3	$[-(w + d_1 + d_2 + d_3), -l + d, h - L_n]^T$
R3	$[w + d_1 + d_2 + d_3, -l + d, h - L_n]^T$

$$\mathbf{T}_B^{D1} = \begin{bmatrix} \cos q_1^{D1} & -\sin q_1^{D1} & 0 & w + (d_1 + d_2 \cos q_2^{D1} + d_3) \cos q_1^{D1} - d \sin q_1^{D1} \\ \sin q_1^{D1} & \cos q_1^{D1} & 0 & l + (d_1 + d_2 \cos q_2^{D1} + d_3) \sin q_1^{D1} + d \cos q_1^{D1} \\ 0 & 0 & 1 & h - L_n - d_2 \sin q_2^{D1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.3}$$

$$\mathbf{T}_B^{L2} = \begin{bmatrix} \cos q_1^{L2} & -\sin q_1^{L2} & 0 & -w - (d_1 + d_2 \cos q_2^{L2} + d_3) \cos q_1^{L2} - d \sin q_1^{L2} \\ \sin q_1^{L2} & \cos q_1^{L2} & 0 & -(d_1 + d_2 \cos q_2^{L2} + d_3) \sin q_1^{L2} + d \cos q_1^{L2} \\ 0 & 0 & 1 & h - L_n + d_2 \sin q_2^{L2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.4}$$

$$\mathbf{T}_B^{D2} = \begin{bmatrix} \cos q_1^{D2} & -\sin q_1^{D2} & 0 & w + (d_1 + d_2 \cos q_2^{D2} + d_3) \cos q_1^{D2} - d \sin q_1^{D2} \\ \sin q_1^{D2} & \cos q_1^{D2} & 0 & (d_1 + d_2 \cos q_2^{D2} + d_3) \sin q_1^{D2} + d \cos q_1^{D2} \\ 0 & 0 & 1 & h - L_n - d_2 \sin q_2^{D2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.5}$$

$$\mathbf{T}_B^{L3} = \begin{bmatrix} \cos q_1^{L3} & -\sin q_1^{L3} & 0 & -w-(d_1+d_2\cos q_2^{L3}+d_3)\cos q_1^{L3} - d\sin q_1^{L3} \\ \sin q_1^{L3} & \cos q_1^{L3} & 0 & -l-(d_1+d_2\cos q_2^{L3}+d_3)\sin q_1^{L3} + d\cos q_1^{L3} \\ 0 & 0 & 1 & h - L_n + d_2\sin q_2^{L3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.6}$$

$$\mathbf{T}_B^{D3} = \begin{bmatrix} \cos q_1^{D3} & -\sin q_1^{D3} & 0 & w+(d_1+d_2\cos q_2^{D3}+d_3)\cos q_1^{D3} - d\sin q_1^{D3} \\ \sin q_1^{D3} & \cos q_1^{D3} & 0 & -l+(d_1+d_2\cos q_2^{D3}+d_3)\sin q_1^{D3} + d\cos q_1^{D3} \\ 0 & 0 & 1 & h - L_n - d_2\sin q_2^{D3} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.7}$$

Vrijednosti kinematičkih parametara dane su u tablici 5.3.

Tablica 5.3: Vrijednosti kinematičkih parametara

Parametar	Vrijednost [cm]
h	0
l	12.5
L_n	11.3
w	6.25
d_1	1.57
d_2	5
d_3	1.56
d	1.23

5.2. Inverzna kinematika

Inverzni kinematički problem se prema [11] definira na sljedeći način:
Ako su zadane vrijednosti položaja $\mathbf{p}$ *i orijentacije alata* $\mathbf{R}$*, tada je potrebno pronaći vrijednost varijabli u prostoru zglobova koje zadovoljavaju jednadžbu manipulatora odnosno osiguravaju da alat poprimi zadani položaj i orijentaciju.*
Iako je prema definiciji inverznog kinematičkog problema osnovni zadatak inverzne kinematike odrediti zakrete zglobova koji će osigurati da alat poprimi željenu konfiguraciju, u ovom radu se pod pojmom rješavanja inverznog kinematičkog problema podrazumijeva određivanje zakreta zglobova koji će osigurati da šesteronožni hodač poprimi zadanu orijentaciju u prostoru, s ciljem kompenzacije mogućeg nagiba podloge po kojoj se hodač giba. Prilikom rješavanja tog problema korišteni su rezultati

prikazani u [10] i [13].
Prilikom rješavanja problema inverzne kinematike korištene su sljedeće pretpostavke:

1. Inverzna kinematika se zasniva na nepomičnosti vrha noge u prostoru.

2. Kako je 1 stupanj slobode noge potreban za pokretanje noge naprijed odnosno natrag, inverzna kinematika ne uzima taj stupanj slobode kao parametar.

3. Inverznom kinematikom za šesteronožnog hodača moguće je upravljanje visinom robota te kutevima poniranja i valjanja. Funkciju pokretanja robota naprijed odnosno natrag preuzima generator sekvence. Gibanje robota u stranu nije moguće dok se skretanje vrši upravljanjem amplitudama oscilatora koji se koristi za generiranje sekvence.

4. Izlaz iz rješavača inverzne kinematike predstavlja srednju vrijednost oko koje oscilira motor koji podiže nogu.

5.2.1. Kinematički okvir za rješavanje inverzne kinematike

Za razliku od direktne kinematike u kojoj je objekt promatranja bio položaj odnosno zakret zgloba, prilikom razrade inverzne kinematike promatraju se brzine zglobova s ciljem upravljanja položajem i orijentacijom robota pomoću Jacobian matrice, pa su tako i svi definirani uvoji vezani uz brzine zglobova. Na slici 5.4 prikazan je kinematički okvir koji je polazište za rješenje problema inverzne kinematike. U usporedbi s kinematičkim okvirom prikazanim na slici 5.2, okvir je dodatno proširen fiksnim koordinatnim sustavom {F}, dok će svi uvoji biti prikazani u koordinatama sustava {B} koji je pridružen aproksimiranom centru mase šesteronožnog hodača. Brzina koordinatnog sustava {B} u odnosu na {F} može se prikazati pomoću uvoja izraženih u eksponencijalnim koordinatama sustava {B}:

– Translacijska brzina u x smjeru:

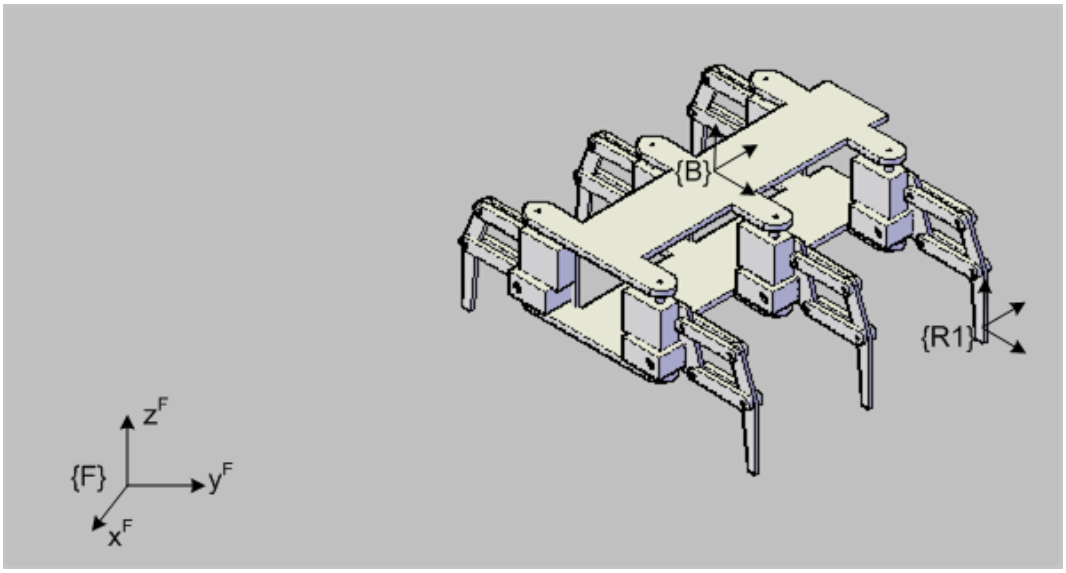
$$[\xi_F^B]_x^B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \dot{x} \tag{5.8}$$

– Translacijska brzina u y smjeru:

$$[\xi_F^B]_y^B = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T \dot{y} \tag{5.9}$$

– Translacijska brzina u z smjeru:

$$[\xi_F^B]_z^B = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T \dot{z} \tag{5.10}$$



Slika 5.4: Kinematički okvir korišten pri rješavanju problema inverzne kinematike

– Rotacijska brzina oko x - poniranje robota (engl. *pitch*):

$$[\xi_F^B]_\theta^B = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T \dot{\theta} \tag{5.11}$$

– Rotacijska brzina oko y - valjanje robota (engl. *roll*):

$$[\xi_F^B]_\phi^B = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T \dot{\phi} \tag{5.12}$$

– Rotacijska brzina oko z - skretanje robota (engl. *yaw*):

$$[\xi_F^B]_\psi^B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \dot{\psi} \tag{5.13}$$

Pošto su svi navedeni uvoji zapisani u koordinatama sustava {B}, bilo koja transformacija $\mathbf{T}_F^B$ ne mijenja eksponencijalne koordinate uvoja, pa se uvoji mogu zbrojiti te je rezultantni uvoj dan sljedećim izrazom:

$$[\xi_F^B]^B = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \dot{\theta} & \dot{\phi} & \dot{\psi} \end{bmatrix}^T \tag{5.14}$$

Izrazom (5.14) definirana je brzina koordinatnog sustava {B} odnosno brzina robota u odnosu na koordinatni sustav {F} zapisana u koordinatama sustava {B}.
Uvoji određeni za pojedine noge prethodno su prikazani u tablici 5.1. U poglavlju o određivanju Jacobian matrice navedeno je kako se pojedini uvoji moraju transformirati u ovisnosti o zakretima zglobova koji pomiču zglob kojem je uvoj pridružen. Tako za nogu $L1$ vrijedi:

$$[\xi_1]^B = [l, w, 0, 0, 1]^T$$

$$[\hat{\xi}_2']^B = e^{[\hat{\xi}_1]^B q_1}[\hat{\xi}_2]^B e^{-[\hat{\xi}_1]^B q_1} \rightarrow [\xi_2']^B$$

$$[\hat{\xi}_3']^B = e^{[\hat{\xi}_2]^B q_2} e^{[\hat{\xi}_1]^B q_1}[\hat{\xi}_3]^B e^{-[\hat{\xi}_2]^B q_2} e^{-[\hat{\xi}_1]^B q_1} \rightarrow [\xi_3']^B$$

Sličnim postupkom se dobiju eksponencijalne koordinate uvoja za ostale noge. Kako su nakon primjenjene transformacije uvoji mogu zbrajati, potrebno je zbrojiti $[\hat{\xi}_2']^B$ i $[\hat{\xi}_3']^B$ koji su povezani s istim zglobom kako bi se dobila Jacobian matrica odgovarajućih dimenzija. Rezultantni uvoj neka je označen s $[\hat{\xi}_{23}']^B$. Jacobian matrica pojedine noge formira se prema opisanom postupku na sljedeći način:

$$[\mathbf{J}_B^i]^B = \left[[\xi_1]^B, [\xi_{23}']^B\right] \tag{5.15}$$

Prema [13] i [10] ukupni odnosno rezultantni uvoj za pojedinu nogu $[\xi_F^i]^B$ se može prikazati u obliku sume transformiranih uvoja na sljedeći način:

$$[\xi_F^i]^B = [\xi_F^B]^B + [\mathbf{J}_B^i]^B \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{5.16}$$

Osnovni princip rješavanja inverzne kinematike iznesen u [10] zasniva se na pretpostavci da se vrh svake noge nalazi na podlozi i da je robot oslonjen na sve noge istovremeno te da se vrh noge ne giba u inercijalnom sustavu {F}. Zbog toga je potrebno rezultantni uvoj iz relacije (5.16) transformirati u koordinatni sustav vrha noge što može se pisati u obliku:

$$[\xi_F^i]^i = [\xi_F^B]^i + [\mathbf{J}_B^i]^i \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \tag{5.17}$$

pri čemu $i \in \{L1, R1, L2, R2, L3, R3\}$ označava neki od koordinatnih sustava pridruženih vrhovima nogu. Jacobian u koordinatnom sustavu i dobije se tako da se svi uvoji iz početnog sustava transformiraju matricom transformacije $\mathbf{Ad}_{\left(\mathbf{T}_B^i\right)^{-1}}$ i zatim prema istom principu slože u novu Jacobian matricu $[\mathbf{J}_B^i]^i$. Uzevši u obzir pretpostavku o stupnju slobode noge koji se ne koristi zbog potrebe za ostvarivanjem gibanja hodača, iz izraza (5.17) potrebno je eliminirati varijablu q_1, što se postiže na način da se iz matrice Jacobiana $[\mathbf{J}_B^i]^i$ eliminira prvi stupac, pa se može pisati:

$$[\xi_F^i]^i = [\xi_F^B]^i - [\mathbf{Q}']^i \dot{q}_2 \tag{5.18}$$

pri čemu je $[\mathbf{Q}']^i$ podmatrica matrice $-[\mathbf{J}_B^i]^i$ odnosno njen drugi stupac.
Tranformacija uvoja vezanog uz brzinu robota se prema izrazu (4.90) zapisuje na sljedeći način:

$$[\xi_F^B]^i = \mathbf{Ad}_{\left(\mathbf{T}_B^i\right)^{-1}}[\xi_F^B]^B \tag{5.19}$$

Sada je rezultantni uvoj dan izrazom:

$$[\xi_F^i]^i = \mathbf{Ad}_{\left(\mathbf{T}_B^i\right)^{-1}}[\xi_F^B]^B - [\mathbf{Q}']^i\dot{q}_2 \tag{5.20}$$

Prema pretpostavci o upravljanju samo brzinama $\dot{z}$, $\dot{\theta}$ i $\dot{\phi}$, iz matrice $\mathbf{Ad}_{\left(\mathbf{T}_B^i\right)^{-1}}$ potrebno je eliminirati stupce koji su vezani uz $\dot{x}$, $\dot{y}$ i $\dot{\psi}$ pri čemu nastaje matrica $[\mathbf{P}']^i$ koja sadrži 3., 4. i 5. stupac matrice $\mathbf{Ad}_{\left(\mathbf{T}_B^i\right)^{-1}}$. Sada se izraz (5.20) piše u sljedećem obliku:

$$[\xi_F^i]^i = [\mathbf{P}']^i \begin{bmatrix} \dot{z} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} - [\mathbf{Q}']^i\dot{q}_2 \tag{5.21}$$

Prema [10] rješenje inverzne kinematike zasniva se na nepomičnosti vrha noge u koordinatnom sustavu {F}, zbog čega je potrebno ograničiti translacijske brzine koordinatnog sustava {i} u sustavu {F}, dok se rotacijske brzine ne promatraju jer ne utječu na položaj vrha noge. Prema tome, iz matrica $[\mathbf{P}']^i$ i $[\mathbf{Q}']^i$ u izrazu (5.21) izdvajaju se prva tri retka koji su vezani uz translacijske brzine i izjednačavaju se s nul-vektorom pa se dobije:

$$[\mathbf{P}_i]^i \begin{bmatrix} \dot{z} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} - [\mathbf{Q}_i]^i\dot{q}_2 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.22}$$

Izraz (5.22) se kombiniranjem za sve noge može zapisati u obliku cjelovitog rješenja za šesteronožni robotski hodač:

$$\begin{bmatrix} [\mathbf{P}_{L1}]^{L1} \\ [\mathbf{P}_{R1}]^{R1} \\ [\mathbf{P}_{L2}]^{L2} \\ [\mathbf{P}_{R2}]^{R2} \\ [\mathbf{P}_{L3}]^{L3} \\ [\mathbf{P}_{R3}]^{R3} \end{bmatrix} \begin{bmatrix} \dot{z} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} - \begin{bmatrix} [\mathbf{Q}_{L1}]^{L1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & [\mathbf{Q}_{R1}]^{R1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & [\mathbf{Q}_{L2}]^{L2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & [\mathbf{Q}_{R2}]^{R2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & [\mathbf{Q}_{L3}]^{L3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & [\mathbf{Q}_{R3}]^{R3} \end{bmatrix} \begin{bmatrix} \dot{q}_2^{L1} \\ \dot{q}_2^{R1} \\ \dot{q}_2^{L2} \\ \dot{q}_2^{R2} \\ \dot{q}_2^{L3} \\ \dot{q}_2^{R3} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{5.23}$$

Izraz (5.23) se u skraćenom obliku piše:

$$\mathbf{P} \begin{bmatrix} \dot{z} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} - \mathbf{Q}\dot{\mathbf{q}} = \mathbf{0} \tag{5.24}$$

Rješenje inverzne kinematike predstavlja određivanje vektora brzina zglobova $\mathbf{q}$ iz izraza (5.24). Ovakav način određivanja brzina zglobova se može smatrati određenim oblikom upravljanja zasnovanog na Jacobian matrici, pošto se matrica $\mathbf{Q}$ u izrazu (5.24) sastoji od podmatrica Jacobiana za svaku od 6 nogu šesteronožnog hodača.

5.3. Upravljanje zasnovano na Jacobian matrici

Formalnim matematičkim pristupom rješenje (5.24) dano je sljedećim izrazom:

$$\dot{\mathbf{q}} = \mathbf{Q}^{-1}\mathbf{P}\begin{bmatrix}\dot{z}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix} \tag{5.25}$$

Problem se javlja ukoliko matrica $\mathbf{Q}$ nije kvadratna, što je slučaj i za šesteronožnog hodača, pa ne postoji njen inverz. Prema [7] upravljanje zasnovano na Jacobian matrici može se ostvariti na dva načina odnosno pomoću dvije metode koje su primjenjive i za matrice koje nisu kvadratične ili punog ranga:

1. Metoda transponiranja
2. Metoda pseudoinverza

Opravdanost obiju metoda detaljnije je diskutirana u [7].

5.3.1. Metoda transponiranja

Prema metodi transponiranja rješenje izraza (5.24) dano je jednadžbom:

$$\dot{\mathbf{q}} = \mathbf{Q}^{T}\mathbf{P}\begin{bmatrix}\dot{z}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix} \tag{5.26}$$

Metoda transponiranja je zapravo jedna od prvih metoda realizacije upravljačkih algoritama zasnovanih na Jacobian matrici. Metodu je opravdano koristiti jer je u [7] pokazano kako se za male pomake koji se dobiju kao:

$$\Delta\mathbf{q} = \alpha\mathbf{Q}^{T}\mathbf{P}\begin{bmatrix}\Delta z\\ \Delta\theta\\ \Delta\phi\end{bmatrix} \tag{5.27}$$

smanjuje odstupanje stvarnog od željenog položaja, što znači da se robot usmjerava prema željenoj konfiguraciji u prostoru. Pritom se koeficijentom α može upravljati brzinom konvergencije algoritma. Pritom je potrebno voditi računa o tome da se ne odabere prevelik iznos za α kako se ne bi narušila stabilnost algoritma.

5.3.2. Metoda pseudoinverza

Prema metodi pseudoinverza rješenje izraza (5.24) dano je jednadžbom:

$$\dot{\mathbf{q}} = \mathbf{Q}^{\dagger}\mathbf{P}\begin{bmatrix}\dot{z}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix} \tag{5.28}$$

pri čemu je $\mathbf{Q}^{\dagger}$ pseudoinverz matrice $\mathbf{Q}$. Općenito se pseudoinverz neke matrice $\mathbf{A}$ definira (prema [4]) na sljedeći način:
Neka je $\mathbf{A}$ *matrica dimenzija* $m \times n,\ m \neq n$ *i* $\mathbf{b}$ *vektor dimenzije* m*. Izraz* $\|\mathbf{Ax} - \mathbf{b}\|$ *poprima minimalnu vrijednost za* $\mathbf{x} = \mathbf{A}^{\dagger}\mathbf{b}$*. Matrica* $\mathbf{A}^{\dagger}$ *naziva se pseudoinverzom matrice* $\mathbf{A}$.
Problematičnost metode pseudoinverza očituje se u ponašanju blizu singulariteta. Ukoliko se manipulator nalazi točno u singularitetu, metoda pseudoinverza će dati ispravan rezulat jer neće ni pokušati pokrenuti manipulator. Suprotno tome, ukoliko se manipulator nalazi blizu singulariteta metoda pseudoinverza rezultira velikim oscilacijama u brzini zakreta zglobova čak i za jako male pomake. Zbog problematičnosti u okolini singulariteta metoda pseudoinverza se ne koristi često. Dodatan otežavajući faktor je nemogućnost simboličkog izračunavanja pseudoinverza s ciljem dobivanja direktnih izraza koji bi se implementirali u algoritmu. Zbog toga je potrebno u svakom koraku za dohvaćene vrijednosti zglobova formirati matricu $\mathbf{Q}$ i izračunati njen pseudoinverz $\mathbf{Q}^{\dagger}$, što može biti numerički zahtjevno, rezultirati sporim izvođenjem algoritma i lošim ponašanjem objekta upravljanja. Zbog toga se za upravljanje šesteronožnim hodačem koristi metoda transponiranja.

Prema tome, polazište za konačno rješenje problema inverzne kinematike dano je izrazom (5.26). Konačno rješenje uz uvrštene parametre iz tablice 5.3 dano je sljedećim izrazom:

$$\begin{bmatrix}\dot{q}_2^{L1}\\ \dot{q}_2^{R1}\\ \dot{q}_2^{L2}\\ \dot{q}_2^{D2}\\ \dot{q}_2^{L3}\\ \dot{q}_2^{R3}\end{bmatrix} = \begin{bmatrix} -5\cos q_2^{L1} & -68.65\cos q_2^{L1} & 56.5\sin q_2^{L1} - 46.9\cos q_2^{L1} - 25\\ 5\cos q_2^{R1} & 68.65\cos q_2^{R1} & -56.5\sin q_2^{R1} - 46.9\cos q_2^{R1} - 25\\ -5\cos q_2^{L2} & -6.15\cos q_2^{L2} & 56.5\sin q_2^{L1} - 46.9\cos q_2^{L1} - 25\\ 5\cos q_2^{R2} & 6.15\cos q_2^{R2} & -56.5\sin q_2^{R1} - 46.9\cos q_2^{R1} - 25\\ -5\cos q_2^{L3} & 56.35\cos q_2^{L3} & 56.5\sin q_2^{L1} - 46.9\cos q_2^{L1} - 25\\ 5\cos q_2^{R3} & -56.35\cos q_2^{R3} & -56.5\sin q_2^{R1} - 46.9\cos q_2^{R1} - 25\end{bmatrix}\begin{bmatrix}\dot{z}\\ \dot{\theta}\\ \dot{\phi}\end{bmatrix} \tag{5.29}$$

Prethodno navedeni izrazi su implementirani u rješavaču inverzne kinematike.

6. Implementacija algoritama upravljanja šesteronožnim robotom

Za implementaciju sustava upravljanja šesteronožnim hodačem odabran je programski paket *Matlab* (verzija *r2009b*) odnosno alat *Simulink*. Zbog potrebe za izvođenjem cijelog sustava upravljanja u stvarnom vremenu, dijelovi sustava upravljanja implementirani su kao Matlabove sistemske funkcije (s-funkcije, engl. *S-Functions*) koje su prema predlošcima koji dolaze uz Matlab pisane u programskom jeziku C odnosno C++[1]. Iako se za prevođenje programskih kodova s-funkcija može koristiti i Matlabov ugrađeni prevodilac (engl. *compiler*), zbog mogućnosti analize koda i otkrivanja pogrešaka korišten je alat Visual Studio 2008. Kratke upute za pisanje Matlabovih s-funkcija mogu se pronaći na stranici [1].
Sustav upravljanja u obliku sheme u Simulinku prikazan je na slici 6.1, a sastoji se od sljedećih komponenti odnosno blokova:

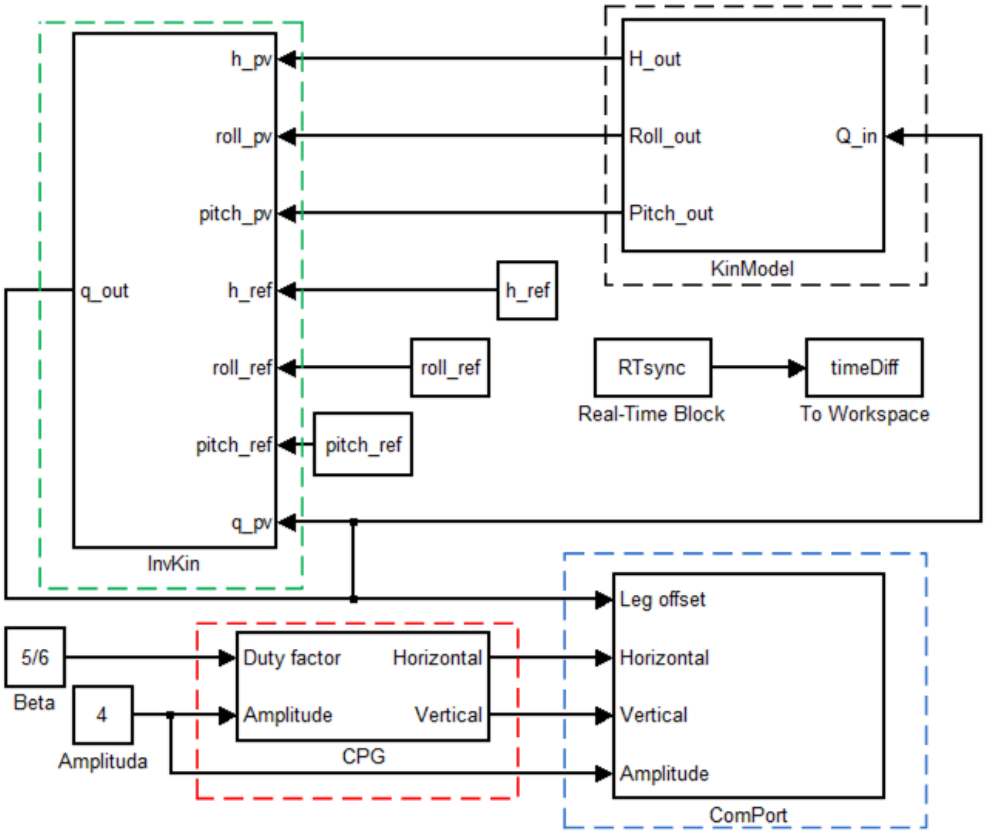
- CPG - centralni generator sekvence
- InvKin - rješavač inverzne kinematike
- KinModel - model kinematike šesteronožnog hodača
- ComPort - blok za pripremu podataka i slanje upravljačkih riječi na upravljački uređaj

Za sinkronizaciju modela sa stvarnim vremenom koristi se blok *RTsync*. Programski kodovi svih implementiranih funkcija dani su u dodatku.

6.1. Generator sekvence

Blok generatora sekvence prikazan je na slici 6.1 i označen isprekidanim crvenim pravokutnikom. Ulazi u blok odnosno funkciju su:

[1] Matlabove s-funkcije moguće je pisati i u Matlabovom programskom jeziku M, ali se one sporije izvršavaju.



Slika 6.1: Blokovska shema sustava upravljanja šesteronožnim hodačem

– *Duty factor* - Faktor popunjenja β

– *Amplitude* - Kvadrat amplitude oscilacija (parametar oscilatora μ)

Izlazi iz funkcije su:

– *Horizontal* - trajektorija za motore koji pomiču noge naprijed odnosno natrag, vektor dimenzije 6

– *Vertical* - upravljačke vrijednosti za spuštanje odnosno podizanje nogu, vektor dimenzije 6

Parametri koje funkcija prima su:

– α - faktor brzine konvergenije oscilatora

– T_{sw} - vrijeme zamaha nogu

– a - parametar nesimetričnosti oscilacija

Hopfovi oscilatori od kojih se sastoji generator sekvence implementirani su u prostoru stanja zbog čega funkcija ima 12 kontinuiranih stanja. U svakom koraku izračunavaju se brzine promjene varijabli stanja prema relacijama Hopfovog oscilatora dok se na izlaz funkcije prosljeđuju dohvaćene trenutne vrijednosti varijabli stanja. Integracija

se odvija unutarnjim procedurama *Simulinka* dok se funkcijom dohvaćaju trenutne vrijednosti varijabli stanja oscilatora x_i koje se prosljeđuju na izlaz bloka. Također, na osnovu dovaćenih varijabli stanja formiraju se upravljačke vrijednosti za spuštanje odnosno podizanje nogu.

6.2. Rješavač inverzne kinematike

Blok rješavača inverzne kinematike prikazan je na slici 6.1 i označen isprekidanim zelenim pravokutnikom. Ulazi u blok odnosno pripadajuću s-funkciju su:

- h_{pv} - vrijednost visine šesteronožnog hodača dobivena preko povratne veze
- $roll_{pv}$ - vrijednost kuta valjanja hodača
- $pitch_{pv}$ - vrijednost kuta poniranja hodača
- h_{ref} - postavna (referentna) vrijednost visine hodača
- $roll_{ref}$ - postavna vrijednost kuta valjanja
- $pitch_{ref}$ - postavna vrijednost kuta poniranja
- q_{pv} - trenutna srednja vrijednost zakreta vertikalnih motora, vektor dimenzije 6

Izlaz iz bloka rješavača inverzne kinematike je vektor q_{out} koji je također dimenzije 6, a predstavlja nove srednje vrijednosti zakreta vertikalnih motora. Funkcija rješavača inverzne kinemtike ne prima parametre. Sama funkcija je izvedena kao blok s kontinuiranim stanjima, imajući u vidu kako postupak inverzne kinematike daje brzine zglobova. Implementacijom funkcije s kontinuiranim stanjima integracija brzine odnosno izračun trenutnog položaja pojedinog zgloba obavlja se unutarnjim procedurama unutar Simulinka dok se funkcijom dohvaća trenutna vrijednost te prosljeđuje na izlaz. Kako ne postoji povratna veza po poziciji motora, za trenutnu poziciju motora funkciji se prosljeđuje vrijednost koja je zadnja poslana na upravljački uređaj.

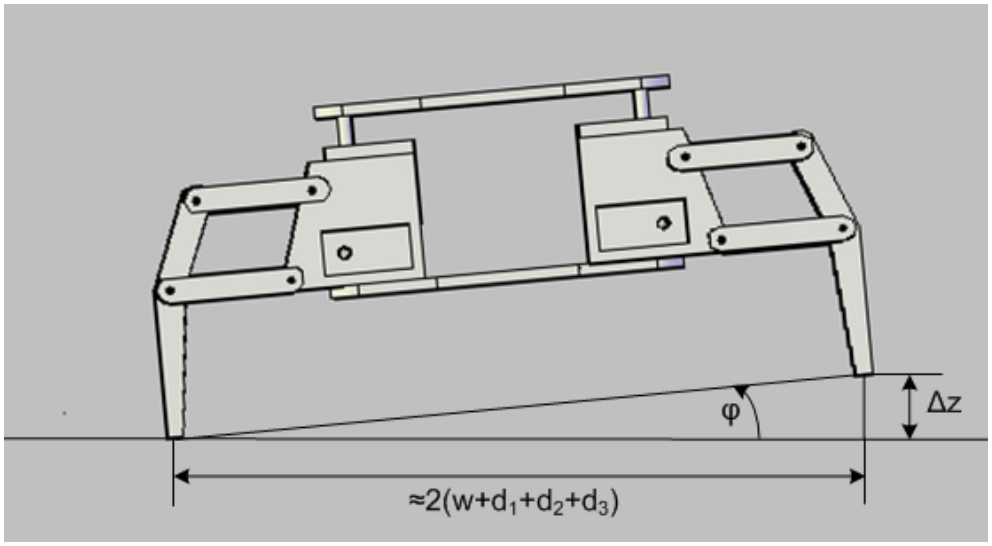
6.3. Kinematički model šesteronožnog hodača

Kako u trenutnoj konfiguraciji šesteronožnog hodača ne postoji senzor koji bi dao informaciju o položaju i orijentaciji hodača, s ciljem provjeravanja ispravnosti rada izrađen je model koji za trenutne srednje vrijednosti oko kojih osciliraju vertikalni motori daje aproksimiranu visinu hodača te kutove valjanja i poniranja. Pritom je potrebno naglasiti kako model ispravno radi u jako uskom području te služi isključivo u svrhe

provjeravanja ispravnosti rada rješavača inverzne kinematike. Blok je na slici 6.1 označen isprekidanim crnim pravokutnikom. Ulaz u blok kinematičkog modela je trenutni položaj odnosno srednja vrijednost zakreta vertikalnih motora. Jednako kao i kod bloka rješavača inverzne kinematike, zbog nepostojanja povratnih veza se na ulaz bloka prosljeđuju vrijednosti koje su zadnje poslane na upravljački uređaj. Izlazi iz bloka kinematičkog modela su:

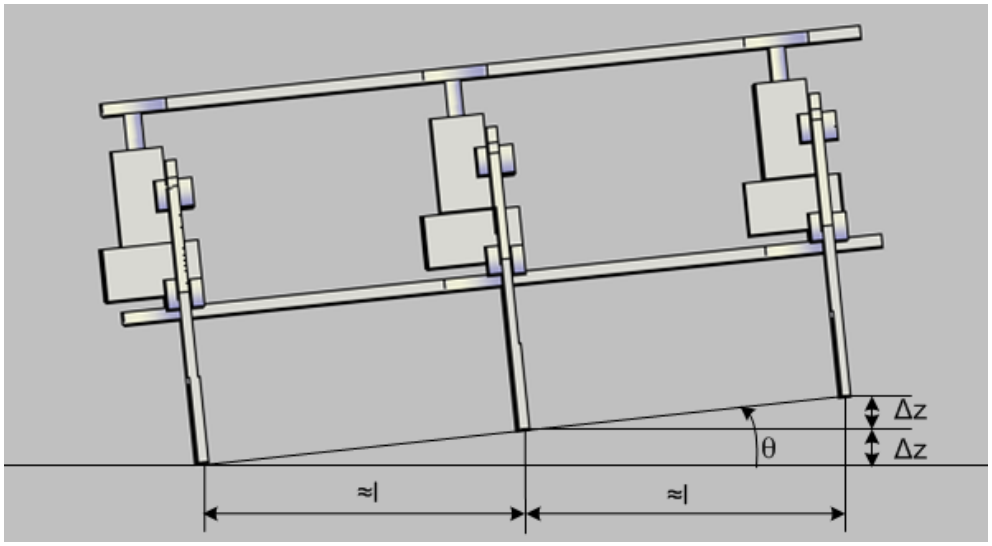
- *H_out* - aproksimirana vrijednost visine centra mase robotskog hodača u odnosu na podlogu
- *Roll_out* - aproksimirana vrijednost kuta valjanja
- *Pitch_out* - aproksimirana vrijednost kuta poniranja

Sam kinematički model zasniva se na geometriji samog robota pri čemu se iz odnosa z koordinata svih nogu izračunavaju kutevi valjanja i poniranja te visina centra mase robota. Kut valjanja određuje se iz razlike z koordinata nogu s nasuprotnih strana na način da se za svaki par nogu izračuna kut valjanja prema slici 6.2 te se na izraz proslijedi srednja vrijednost triju određenih kuteva. Kut poniranja određuje se na način da



Slika 6.2: Geometrijski prikaz kuta poniranja hodača

se za svaki par nasuprotnih nogu odredi srednja vrijednost z koordinata te se prema slici 6.3 određuje kut poniranja iz odnosa srednjih vrijednosti visina parova nasuprotnih nogu. Na izlaz se prosljeđuje srednja vrijednost određenih kutova poniranja. Sama visina centra mase određuje se kao srednja vrijednost z koordinata svih nogu hodača.



Slika 6.3: Geometrijski prikaz kuta poniranja hodača

6.4. Slanje naredbi na upravljački uređaj

Zakrete za motore koji se dobiju iz generatora sekvence i rješavača inverzne kinematike prosljeđuje se funkciji koja pretvara primljene podatke u oblik koji prepoznaje upravljački uređaj na samom robotu te ih prosljeđuje putem serijske komunikacije. Blok za obradu podataka i slanje naredbi upravljačkom uređaju na slici 6.1 je označen plavim pravokutnikom. Prilikom obrade podataka vodi se računa o tome da su motori na desnoj strani hodača (R1, R2, R3) okrenuti suprotno od osiju pretpostavljenih u kinematičkom okviru (slika 5.2) zbog čega se zakreti poslani na njih inveritraju. Jednako tako, prilikom slanja upravljačkih vrijednosti uzet je u obzir reduktor zakreta prikazan na slici 2.4 pa se vrijednosti zakreta koji se dobiju na ulazu u funkciju skaliraju s faktorom $1/r$ koji je određen relacijom (2.1) i iznosi 3.2394. Kao parametar funkcija prima vrijeme uzorkovanja bloka za slanje podataka.

7. Kompenzacija kuta valjanja pomoću kamere

Kako ne postoji povratna informacija o orijentaciji hodača u prostoru, istražena je mogućnost korištenja postojeće kamere za dobivanje dijela informacije o orijentaciji. Potreba za kompenzacijom kuta valjanja česta je u robotici (pri operaterskom navođenju kretanja robota, kada robot dohvaća izduženi predmet i sl.). U ovom radu kompenzacija kuta valjanja obavlja se zbog načina kretanja šesteronožnog hodača pri čemu dolazi do naginjanja robota oko pravca napredovanja. Istražene su dvije moguće primjene kamere u detekciji kuta valjanja:

- Detekcija kuta valjanja i rotiranje slike s ciljem pružanja operateru točnije informacije o okolini robota neovisno o orijentaciji robota
- Detekcija kuta valjanja i prosljeđivanje informacije rješavaču inverzne kinematike s ciljem kompenzacije nagiba podloge od strane samog robota

7.1. Detekcija kuta valjanja

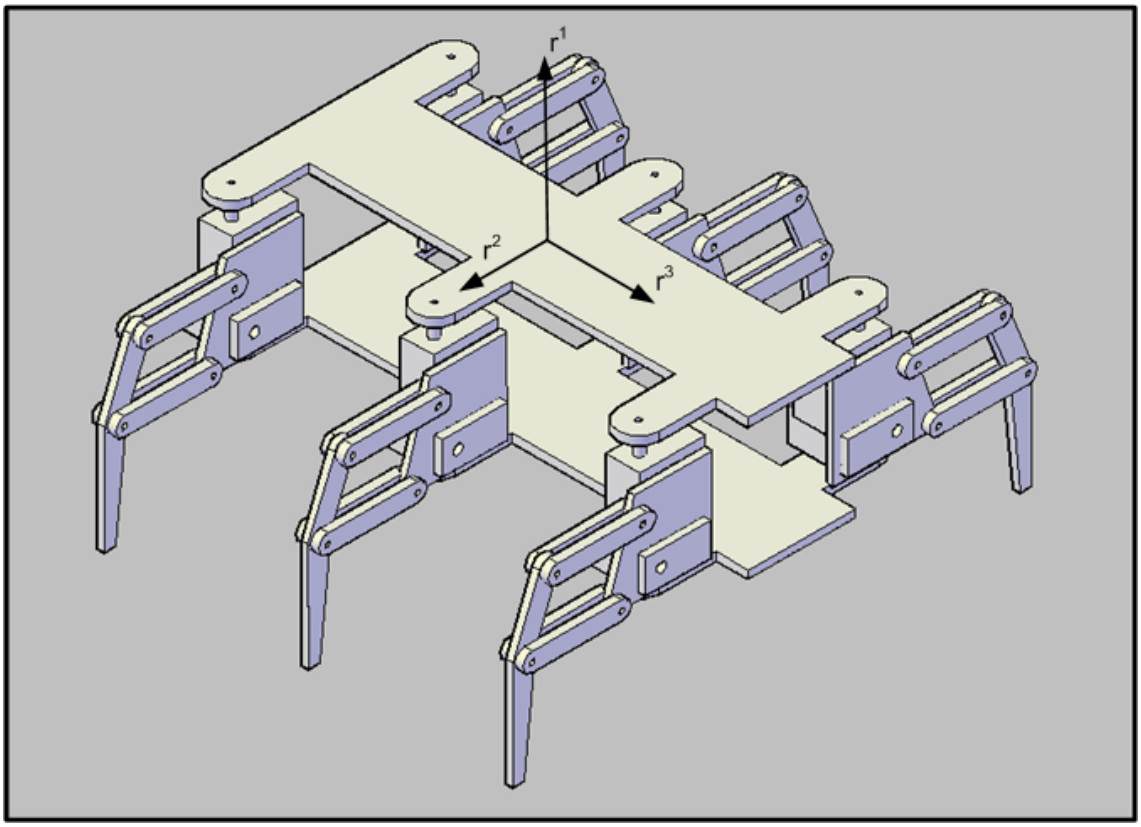
Kako je kamera robota smještena s prednje strane hodača, pomoću kamere moguće je detektirat samo kut valjanja hodača.

7.1.1. Kut valjanja

Kut valjanja može se definirati kao rotaciju oko pravca napredovanja robota. Položaj i orijentaciju slobodnog gibljivog tijela u prostoru jednoznačno se opisuje pomoću 6 vrijednosti:

- 3 vrijednosti vezane uz položaj (x, y, z),
- 3 vrijednosti vezane uz orijentaciju (ϕ, θ, ψ).

Zbog specifičnosti zadatka u ovom poglavlju razmatra se samo orijentacija alata odnosno robotskog hodača. Dakle, najprije je potrebno objasniti orijentaciju alata te definirati pojam kuta valjanja. Za opis orijentacije alata koriste se tri ortonormirana vektor-stupca (vektori su prikazani na slici 7.1) iz matrice rotacije $\mathbf{R}$:



Slika 7.1: Okomiti vektor, vektor približavanja i vektor klizanja za šesteronožnog hodača

- $\mathbf{r}^3$ - vektor približavanja u smjeru djelovanja alata
- $\mathbf{r}^2$ - vektor klizanja ili pomicanja
- $\mathbf{r}^1$ - okomit vektor, okomit na ravninu koju razapinju vektor približavanja i vektor klizanja

Pritom je orijentacija vektora $\mathbf{r}^1$ određena tako da spomenuti vektori čine desno orijentirani koordinatni sustav. Potrebno je također uočiti kako se vektori ($\mathbf{r}^3$, $\mathbf{r}^2$, $\mathbf{r}^1$) podudaraju redom s (y, x, z) osima koordinatnog sustava $\{\mathrm{B}\}$ definiranog pri rješavanju direktnog i inverznog kinematičkog problema. Prema tome vrijedi:

- kretanje u smjeru vektora $\mathbf{r}^3$ odgovara kretanju robota unaprijed
- kretanje u smjeru vektora $\mathbf{r}^2$ odgovara kretanju robota udesno
- kretanje u smjeru vektora $\mathbf{r}^1$ odgovara kretanju robota prema gore

Prema navedenom, orijentacija robota se može definirati na sljedeći način:

- rotacija oko vektora $\mathbf{r}^3$ odgovara valjanju robota
- rotacija oko vektora $\mathbf{r}^2$ odgovara poniranju robota
- rotacija oko vektora $\mathbf{r}^1$ odgovara skretanju robota

Jedan zadatak je pomoću digitalne kamere koja se nalazi na robotu pronaći horizont, izračunati kut valjanja robota te rotacijom slike kompenzirati izračunati kut. Drugi zadatak je poslati podatak o kutu zakreta rješavaču inverzne kinematike. Za rješavanje obaju problema prethodno je potrebno prepoznati horizont. Razvijena aplikacija radi uz pretpostavku da horizont predstavlja jasna linija čiji je položaj u stvarnosti poznat (ne smanjuje se općenitost algoritma ako se pretpostavi da je linija horizontalna). Općenito govoreći, glavni problem kod razlikovanja oblika od okoline predstavljaju oblici i kontrast. Osim o osvjetljenju i oblicima u okolini, u smislu efikasnosti algoritma najvažnije je postojanje horizonta kao jasne linije u vidokrugu. U nastavku poglavlja detaljno je objašnjena izrada algoritma detekcije i kompenzacije kuta valjanja. Tako složen zadatak podijeljen je u sljedeće faze:

- prihvat slike i izdvajanje *frameova*
- prepoznavanje rubova
- traženje linija i odabir linije kao horizonta
- zakretanje slike za izračunati kut
- slanje podatka o kutu valjanja rješavaču inverzne kinematike

7.2. Prihvat slike s kamere

OpenCV funkcija *cvCreateCameraCapture()* omogućava jednostavno očitavanje slike s kamere. Naredbi se predaje ID broj kamere kao argument (početna vrijednost je '-1', što znači *samo odaberi jednu kameru*), dok ona vraća pokazivač na strukturu tipa *CvCapture** koja sadrži informacije o učitanoj slici s kamere. Također, koristeći *cvSetCaptureProperty()* postavljaju se promjenjiva svojstva stvorene *CvCapture* strukture. Argumentima *CV_CAP_PROP_FRAME_WIDTH/HEIGHT* postavlja se način čitanja u jedinicama okvir (engl. *frame*) te definiraju dimenzije pojedinog okvira, odnosno rezolucija kamere. Pristup kameri i postavljanje parametara ostvaruje se sljedećim isječkom koda:

```
CvCapture* kamera =cvCreateCameraCapture(0);
```

```
cvSetCaptureProperty(kamera,CV_CAP_PROP_FRAME_WIDTH,640);
cvSetCaptureProperty(kamera,CV_CAP_PROP_FRAME_HEIGHT,480);
if (kamera==0) return -1;
```

Da bi se obradilo sliku na željeni način potrebno je izdvojiti pojedine okvire slike. To se ostvaruje pomoću OpenCV naredbi *cvGrabFrame()* i *cvRetrieveFrame()*. Prva funkcija na najbrži mogući način kopira sliku u internu memoriju nevidljivu korisniku te vraća 1 ako je uspješno obavljeno dohvaćanje slike, a 0 ako je dohvaćanje neuspješno. Druga funkcija vraća *IplImage** pokazivač koji pokazuje na dohvaćeni okvir slike. Za izdvajanje pojedinog okvira koristi se i kombinaciju tih dviju funkcija ostvarenu funkcijom *cvQueryFrame()*. *cvQueryFrame()* kao argument prihvaća pokazivač na strukturu *CvCapture*. Potom dohvaća sljedeći okvir slike u memoriju već otprije dodijeljenu *CvCapture* strukturi. Izdvajanja okvira ostvaruje se sljedećim naredbama u kodu:

```
if(!cvGrabFrame(kamera)) exit(0);
slika = cvRetrieveFrame(kamera);
```

Rezultat prihvaćanja slike s kamere prikazan je na slici 7.2.

Slika 7.2: Rezultat prihvaćanja slike

7.2.1. Strukture podataka slike i pristup podacima

Osnovni element slike je piksel. Piksel (engl. *picture element*) u smislu računalnog vida predstavlja skalarnu brojčanu vrijednost (siva slika) ili vektor brojčanih vrijednosti (boje ili spektri) koja prikazuje informaciju o jednoj točki slike. Slika je prikazana kao polje piksela. Najčešća korištena struktura OpenCV-a za rad sa različitim vrstama slika je *IplImage*. *IplImage* struktura podataka zapravo je slična matričnoj strukturi *CvMat* s dodatnim svojstvima koja omogućavaju interpretaciju matrice kao slike. Struktura *IplImage* pokazana je u nastavku.

```
typedef struct _IplImage {
int nSize;
int ID;
int nChannels;
int alphaChannel;
int depth;
char colorModel[4];
char channelSeq[4];
int dataOrder;
int c;
int align;
int width;
int height;
struct _IplROI* roi;
struct _IplImage* maskROI;
void* imageId;
struct _IplTileInfo* tileInfo;
int imageSize;
char* imageData;
int widthStep;
int BorderMode[4];
int BorderConst[4];
char* imageDataOrigin;
} IplImage;
```

Shvaćanje ove strukture i njenih parametara bitno je za razumijevanje operacija nad slikom, zbog čega su pojašnjeni osnovni parametri. Parametri *width* i *height* predstavljaju broj piksela po širini i po visini. Parametrom *depth* zadaje se skup vrijednosti

kojima se zadaju pojedini elementi piksela, dok se parametrom *nChannels* postavlja broj elemenata koji definiraju piksel, odnosno broj kanala (engl. channels). Parametrom *origin* definira se način čitanja slike, počevši od gornjeg lijevog ili od donjeg desnog rubnog piksela. Pomoću *dataOrder* definiran je redoslijed elemenata piksela u matričnom zapisu slike. Vrijednost *widthStep* odgovara broju okteta u retku slike. Pristup podacima omogućuje parametar *imageData* koji predstavlja pokazivač na matrični zapis piksela, točnije na njegov prvi redak. Pokazivač na *IplImage* strukturu koristi se za daljnju manipulaciju nad slikom.

7.3. Siva slika i filtriranje

Kako bi se ubrzala obrada slike, ulazna slika s kamere pretvara se u sivu sliku. Radi se o pretvorbi ulazne RGB[1] slike u sivu sliku na način da se svakom pikselu dodjeljuje vrijednost koja karakterizira određene vizualne karakteristike slike. Kako funkcija prima pokazivače na ulaznu i izlaznu strukturu mora se rezervirati mjesto u memoriji za strukturu izlazne slike. Koristi se OpenCV funkcija *cvCreateImage()*. Prvi argument koji se predaje funkciji je veličina strukture, *CvSize*, koju se dobije pozivom funkcije *cvGetSize(image)* koja vraća veličinu postojeće strukture slike. Drugim argumentom zadaje se koji tip podataka je korišten za svaki kanal na svakom pikselu. Posljednji argument definira broj kanala. Pretvaranje u sivu sliku ostvaruje se pomoću gotove implementirane OpenCV funkcije *cvCvtColor()*. Ostvarenje u kodu:

```
cvCvtColor(img, img_gray, CV_RGB2GRAY);
```

U svrhu obrade slike, iz slike se nastoje izvući korisne informacije. Mjestimične nagle promjene intenziteta u nekom području slike ujednačenog intenziteta najčešće predstavljaju šum u slici. Otklanjanje šuma je najvažniji razlog filtriranju. Filtriranje, odnosno izglađivanje, je jednostavna i česta korištena radnja pri obradi slike. Također, veoma bitna funkcija filtriranja je i pojednostavljenje slike, odnosno zanemarenje manje bitnih, a isticanje važnijih dijelova slike. OpenCV omogućuje 5 različitih načina filtriranja pomoću funkcije *cvSmooth()*:

1. *CV_BLUR*
2. *CV_BLUR_NO_SCALE*
3. *CV_MEDIAN*

[1] RGB označava Red-Green-Blue odnosno sliku u boji

4. *CV_GAUSSIAN*

5. *CV_BILATERAL*

Kao argumente funkcija *cvSmooth()* prima pokazivače na ulaznu i izlaznu strukturu, odabrani tip filtriranja te parametre koji se koriste u izračunu izlaza, a njihov značaj ovisi o odabiru samog tipa filtriranja. Za ostvarenje zadatka brzina algoritma nije od presudne važnosti, važnija je točnost. Stoga, iz tablice za odabir tipa filtriranja odabere se Gausova metoda, koja se eksperimentalno pokazala kao najbolja. Odabranom metodom za svaki piksel računa se nova vrijednost kao težinski prosjek vrijednosti koje se nalaze na susjednim poljima. Težine se pak dobivaju normalizacijom vrijednosti zadanih dvodimenzionalnom Gausovom razdiobom prema eksponencijalnoj formuli:

$$f(x, y) = e^{-\frac{dx^2+dy^2}{3.125}} \tag{7.1}$$

Kao broj polja koji utječu na vrijednost odabrano je polje dimenzija 7×7. Navedeni algoritam implementiran je u kod funkcijom:

```
cvSmooth(img_gray,img_gray,CV_GAUSSIAN,7,7);
```

Rezultat pretvaranja slike s kamere u sivu sliku i obradu Gausovim operatorom može se vidjeti na slici 7.3.

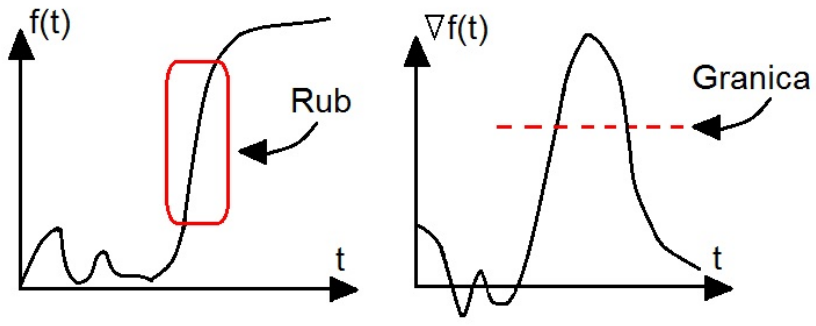
Slika 7.3: Rezultat primjene Gaussova filtra na sivu sliku

7.4. Detekcija rubova na slici

Kako bi se što lakše mogao pronaći horizont potrebno je smanjiti količinu podataka i filtrirati nepotrebne informacije, no pri tome istovremeno očuvati važna strukturna svojstva slike. To se postiže upravo algoritmom za detekciju rubova. Rubovi određuju granice ploha ujednačenog intenziteta i zato njihova detekcija predstavlja najvažniji problem u analizi slike. Rubovi na slici područja su s jakim kontrastom intenziteta, odnosno skokovima u intenzitetu od jednog do drugog piksela. Ako su stranice objekta u slici glatke, homogene i neprozirne, te ako je svjetlo uniformno i bez sjena, intenzitet svjetla naglo se mijenja samo na rubovima, odnosno gradijent na tim mjestima je velik. Upravo na izračunu gradijenta, odnosno njegove aproksimacije, zasnivaju se tehnike prepoznavanja rubova. Postoji više metoda, među kojima valja spomenuti Laplaceovu i Sobelovu metodu. Kako Sobelova metoda daje puno bolje rezultate, češće se koristi te joj je posvećeno više pažnje.

7.4.1. Sobelov operator

Sobelov operator je zapravo diskretni operator diferenciranja koji računa aproksimaciju gradijenta funkcije intenziteta analizirane slike. Metoda gradijenta prepoznaje rub slike tražeći maksimum i minimum prve derivacije slike. Razmatra se najprije ideja na jednodimenzionalnoj funkciji rampe. Uz pretpostavku da f(t) (slika 7.4) predstavlja intenzitet, područje skokovite promjene predstavlja rub. Upravo deriviranjem funkcije f(t) dobije se područje na kojem se intenzitet najbrže mijenja. Dakle, rubovi će imati veće vrijednosti gradijenta od svoje okoline. Nadalje, usporedbom vrijednosti gradijenta može se detektirati rub te proglasiti piksel rubnim pikselom ukoliko vrijednost gradijenta premašuje određenu granicu. Potrebno je proširiti tu ideju na dvije



Slika 7.4: Grafički prikaz detekcije ruba slike

dimenzije, jer se ulazna siva slika čita kao dvodimenzionalna matrica. Pri tome je

potrebno imati na umu kako se elemente promatra diskontinuirano, piksel po piksel. Iako se često kaže da se pri detekciji ruba na ulaznoj sivoj slici izračunava gradijent, zapravo se vrši diskretno diferenciranje. Gradijent funkcije dvije varijable je 2D vektor sa komponentama dobivenim derivacijom u horizontalnom i vertikalnom smjeru. Diferenciranjem u svakoj točki slike, dobiveni vektor pokazuje u smjeru najvećeg mogućeg porasta intenziteta, a duljina vektora odgovara iznosu promjene intenziteta u tom smjeru. Dakle, potreban je operator koji u svakoj točki u ulaznoj sivoj slici računa apsolutnu veličinu te smjer gradijenta intenziteta. Rezultat takvog proračuna pokazuje kako se *naglo* ili *glatko* intenzitet na slici mijenja u točki, a samim time upućuje i na to kolika je vjerojatnost da je u tom dijelu slike rub. Odgovarajući alat za izračun gradijenta je Sobelov operator koji koristi par konvolucijskih maski, veličine 3×3, za proračun aproksimacije gradijenta. Konvolucijske matrice su znatno manje od same slike, te tako prelazeći preko originalne slike manipuliraju pikselima. Jedna u smjeru redaka, a druga u smjeru stupaca procjenjuju gradijent. Ako se definira matrica $\mathbf{I}$ kao originalna slika, onda su $\mathbf{D}_X$ i $\mathbf{D}_Y$ rezultat konvolucije originalne slike i odgovarajuće konvolucijske maske, odnosno sadrže horizontalnu i vertikalnu aproksimaciju derivacija u svakoj točki prema sljedećim izrazima:

– Aproksimacija vertikalne derivacije:

$$\mathbf{D}_X = \frac{\partial}{\partial x}\mathbf{I} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} .\times \begin{bmatrix} \mathbf{I}(i-1,j-1) & \mathbf{I}(i-1,j) & \mathbf{I}(i-1,j+1) \\ \mathbf{I}(i,j-1) & \mathbf{I}(i,j) & \mathbf{I}(i,j+1) \\ \mathbf{I}(i+1,j-1) & \mathbf{I}(i+1,j) & \mathbf{I}(i+1,j+1) \end{bmatrix} \tag{7.2}$$

– Aproksimacija horizontalne derivacije:

$$\mathbf{D}_Y = \frac{\partial}{\partial y}\mathbf{I} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} .\times \begin{bmatrix} \mathbf{I}(i-1,j-1) & \mathbf{I}(i-1,j) & \mathbf{I}(i-1,j+1) \\ \mathbf{I}(i,j-1) & \mathbf{I}(i,j) & \mathbf{I}(i,j+1) \\ \mathbf{I}(i+1,j-1) & \mathbf{I}(i+1,j) & \mathbf{I}(i+1,j+1) \end{bmatrix} \tag{7.3}$$

Rezultat množenja može se zapisati i u drugom obliku pogodnijem za programsku implementaciju:

$$\mathbf{D}_X = \mathbf{I}(i-1,j-1)+2\mathbf{I}(i,j-1)+\mathbf{I}(i+1,j-1)-\mathbf{I}(i-1,j+1)-2\mathbf{I}(i,j+1)-\mathbf{I}(i+1,j+1) \tag{7.4}$$

$$\mathbf{D}_Y = \mathbf{I}(i-1,j-1)+2\mathbf{I}(i-1,j)+\mathbf{I}(i-1,j+1)-\mathbf{I}(i+1,j-1)-2\mathbf{I}(i+1,j)-\mathbf{I}(i+1,j+1) \tag{7.5}$$

Za izračun amplitude gradijenta u svakoj točki slike može se upotrijebiti rezultat prethodnih aproksimacija gradijenta. Koristi se formula:

$$|D| = \sqrt{\mathbf{D}_X^2 + \mathbf{D}_Y^2} \tag{7.6}$$

dok se smjer gradijenta računa pomoću izraza:

$$\phi = \arctan \frac{\mathbf{D}_X}{\mathbf{D}_Y} \tag{7.7}$$

Korjenovanje i kvadriranje decimalnih brojeva uzimaju puno procesorskog vremena te se usporava izvođenje algoritma. Stoga se s ciljem ubrzavanja proračuna koristi aproksimacija amplitude gradijenta zbrajanjem apsolutnih iznosa komponenti:

$$|\mathbf{D}| = |\mathbf{D}_X| + |\mathbf{D}_Y| \tag{7.8}$$

Nakon što se odredi matrica amplitude gradijenta, potrebno je istaknuti prepoznati rub, odnosno istaknuti objekte na slici koji su od interesa. Najjednostavnije je koristiti binarno ograničenje. Binarnim ograničenjem dobiva se binarna slika na kojoj su željeni objekti prikazani jedinicom, a ostali nulom. Iz matrice gradijenta odbacuju se pikseli koji imaju manji gradijent od nekog zadanog i njima se dodjeljuje vrijednost 0 (crna boja), a onima s gradijentom većim od zadanog dodjeljuje se vrijednost 255 (bijela boja). Što je prag odluke veći, intenzitet rubova na slici će biti manji. Dakle, općenito gledajući, algoritam detekcije rubova kao ulaz dobiva sliku sa nijansama sive boje $\mathbf{I}(i, j)$, a na izlazu daje binarnu sliku $\mathbf{L}(i, j)$ u kojoj su rubne točke označene sa '1'. Pri tome, ključni parametar je eksperimentalno utvrđena granica ϵ. Prema tome, koraci u algoritmu detekcije rubova su:

1. Postaviti $i = 1, j = 1, \epsilon > 0$
2. Izračunati $|\nabla \mathbf{I}(i, j)|$
3. Ako je $|\nabla \mathbf{I}(i, j)| > \epsilon$, $\mathbf{L}(i, j) = 1$, inače $\mathbf{L}(i, j) = 0$
4. $j = j + 1$, ako je $j < n$ vratiti se na korak 2
5. $\mathbf{L}(i, n) = 0$ - rubni pikseli
6. $j = 1, i = i + 1$, ako je $i < m$ vratiti se na korak 2
7. Za $j = 1, 2, ..., n$ postaviti $\mathbf{L}(m, j) = 0$ - rubni pikseli

OpenCV ima implementiranu funkciju Sobelovog operatora, *cvSobel()*:

```
cvSobel( const CvArr* src, CvArr* dst,
 int xorder, int yorder, int aperture_size = 3 );
```

Funkcija cvSobel() nije davala zadovoljavajuće rezultate i zbog same nemogućnosti podešavanja određenih parametara (primjerice parametra koji služi kao prag za prepoznavanje rubnih piksela i o kojem značajno ovisi kvaliteta rezultata), samim time i povećavanja robusnosti algoritma, izvodi se programski. Najvažniji dio programskog koda koji ostvaruje Sobelov operator:

```
for(i=granicaLijevoX; i<granicaDesnoX; i++)
 {
  prethodniRedak+=pomakStupac;
  trenutniRedak+=pomakStupac;
  sljedeciRedak+=pomakStupac;
    for(j=granicaLijevoY; j<granicaDesnoY; j++)
  {
   prvi_prvi=*prethodniRedak; prethodniRedak++;
   prvi_srednji=*prethodniRedak; prethodniRedak++;
   prvi_zadnji=*prethodniRedak; prethodniRedak--;
   srednji_prvi=*trenutniRedak; trenutniRedak++;
   srednji_srednji=*trenutniRedak; trenutniRedak++;
   srednji_zadnji=*trenutniRedak; trenutniRedak--;
   zadnji_prvi=*sljedeciRedak; sljedeciRedak++;
   zadnji_srednji=*sljedeciRedak; sljedeciRedak++;
   zadnji_zadnji=*sljedeciRedak; sljedeciRedak--;
   dx = (prvi_prvi + 2*srednji_prvi + zadnji_prvi)-
   (prvi_zadnji + 2*srednji_zadnji + zadnji_zadnji);
   dy = (prvi_prvi + 2*prvi_srednji + prvi_zadnji)-
   (zadnji_prvi + 2*zadnji_srednji + zadnji_zadnji);
   D=(abs(dx)+abs(dy));
   if (D<granicaSOBEL)
   {
    argument[i*(sivaSlika->width)+j]=0;
   }
   else
   {
    argument[i*(sivaSlika->width)+j]=255;
```

```
    }
  }
  prethodniRedak=baza+pomakRedak;
  baza=prethodniRedak;
  trenutniRedak=prethodniRedak+pomakRedak;
  sljedeciRedak=trenutniRedak+pomakRedak;
 }
```

Obrada primljene sive slike počinje uklanjanjem rubnih piksela kako bi se umanjio njihov značaj na daljnje pretraživanje slike. Potom se Sobelovim operatorom prelazi preko slike te prema formulama (7.7) i (7.8) izračunava aproksimacija okomitih i vertikalnih parcijalnih derivacija za svaku točku. U kodu dx predstavlja derivaciju u vertikalnom smjeru (smjeru stupaca), dy u horizontalnom (u smjeru redaka), dok vrijednosti prvi/srednji/zadnji_prvi/srednji/zadnji predstavljaju vrijednosti intenziteta piksela ulazne sive slike. Na osnovu izračunatih derivacija u vertikalnom i horizontalnom smjeru se računa aproksimacija amplitude gradijenta. Nadalje, uspoređuje se izračunata amplituda sa eksperimentalno određenom granicom. Granica je određena tako da je dovoljno niska da obuhvaća sve potrebne rubove za daljnje određivanje linija, a dovoljno visoka da ne obuhvati lažne rubove. Ako je amplituda gradijenta veća od postavljene granice, detektiran je rub te odgovarajući element u izlaznom polju argument se postavlja na iznos 255. Ako je amplituda gradijenta manja od postavljene granice, element poprima vrijednost nula. Rezultat obrade Sobelovim operatorom prikazan je na slici 7.5. Potrebno je napomenuti kako se uz minimalnu promjenu te korištenje izraza (7.7) može proširiti funkcija da računa smjer gradijenta u točki u kojoj je vrijednost amplitude gradijenta iznad granične vrijednosti. Takva bi promjena bila nužna primjerice pri detektiranju kružnica na slici.

7.5. Pronalazak linija na slici

Zadatak pronalaska horizonta na slici može se poistovjetiti sa zadatkom pronalaska linija. Za pronalazak ravnih linija koristila se Houghova transformacija koja predstavlja relativno brz način pretrage binarne slike za pronalazak pravaca.

7.5.1. Houghova transformacija

Houghova transformacija je metoda obrade slike koja se koristi za prepoznavanje oblika unutar slike. Najčešće se koristi za prepoznavanje linija, kružnica, elipsi i sličnih jed-

Slika 7.5: Rezultat obrade sobeovim operatorom

nostavnih oblika. Houghova metoda prepoznavanja zahtijeva da su traženi oblici matematički opisivi. Ulazni podatak za Houghovu transformaciju je ravnina s dvije vrste piksela: pikseli koji predstavljaju rub i pikseli koji predstavljaju pozadinu, stoga su njeni ulazi često monokromatske slike. Implementacija Houghove transformacije u kodu algoritma kompenzacije kuta valjanja nastavlja se na izlaz Sobelovog operatora. Stoga, osnovna ideja Houghove transformacije pravca glasi:

- Svaka točka u slici nalazi se na nekom pravcu te beskonačno pravaca može prolaziti kroz tu točku
- Na pravcu se nalazi više točaka pa kako svaka točka čije koordinate zadovoljavaju jednadžbu pravca predstavlja jedan glas za taj pravac, tim je zastupljeniji u slici što je više njegovih glasova

Jednadžba pravca je:

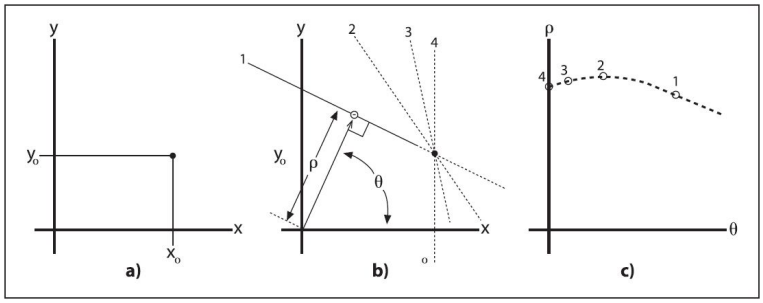
$$y = ax + b \tag{7.9}$$

U slučaju okomitog pravca parametri a i b poprimaju beskonačne iznose pa se uvodi parametarski zapis s ograničenim parametrima, ilustriran na slici 7.6:

$$y = -\frac{\cos\theta}{\sin\theta}x + \frac{\rho}{\sin\theta} \tag{7.10}$$

gdje $\rho \in [-D, D]$, D-dijagonala slike predstavlja udaljenost od ishodišta do pravca duž okomice na pravac, a $\theta \in [0, \pi]$ predstavlja kut koji vektor ρ zatvara s x osi.

Dakle, svaki je pravac predstavljen jedinstvenim parom (ρ, θ). Formiranjem dvodi-



Slika 7.6: Parametarski prikaz pravca [8]

menzionalnog akumulatora koji će pamtiti glasove za pojedini par (ρ, θ), dobiva se zastupljenost pojedinog pravca na slici. OpenCV ima gotovu funkciju koja ostvaruje algoritam Houghove transformacije. Funkcija vraća maksimume u (ρ, θ) ravnini, pri čemu izračun nije eksplicitno prikazan korisniku. Zbog mnogobrojnih načina izvedbe, jednostavnog i preglednog skupa rješenja te same brzine algoritma, korištena je gotova OpenCV funkcija, čije detaljno objašnjenje slijedi u nastavku.

7.5.2. Primjena Houghove transformacije za pronalazak horizonta

OpenCV omogućuje dva različita načina Houghove transformacije linija, standardnu jednokanalnu i višekanalnu Houghovu transformaciju (SHT) i progresivnu vjerojatnosnu Houghovu transformaciju (PPHT). SHT je prethodno opisan algoritam. PPHT promatra ponavljanje linija kroz manje dijelove, čime se smanjuje vrijeme obrade. Oba algoritma koriste istu OpenCV funkciju:

```
CvSeq* cvHoughLines2(
        CvArr* image,
        void* line_storage,
        int method,
        double rho,
        double theta,
        int threshold,
        double param1 = 0,
        double param2 = 0 );
```

Prvi argument je ulazna slika koja se promatra se kao binarna, no mora biti 8-bitna. Drugi argument je pokazivač na mjesto spremanja rezultata. Može biti tipa *memory*

storage ili stupčani vektor, o čemu ovisi način spremanja rezultata. Ako se pri pozivu koristi matrica, funkcija vraća NULL pokazivač. Za SHT metodu tip matrice je *CV_32FC2* u kojoj su spremljene vrijednosti ρ i θ za prepoznate linije. U slučaju korištenja PPHT metode, tip matrice je *CV_32FC4*, gdje su spremljene koordinate početnih i krajnjih točaka prepoznatih linija. Ukoliko se za spremanje odabere *memory storage*, funkcija vraća pokazivač na *CvSeq* strukturu niza kojoj se pristupa na sljedeći način:

```
float* line = (float*) cvGetSeqElem( lines , i );
```

Pri implementaciji, odabrana je druga metoda te se primjer primjene može vidjeti u nastavku.

Za odabir metode služi treći argument, a metoda se može zadati kao:

- *CV_HOUGH_STANDARD*
- *CV_HOUGH_PROBABILISTIC*
- *CV_HOUGH_MULTI_SCALE*

Pomoću iduća dva argumenta, *rho* i *theta* postavlja se razlučivost polja. Jedinične veličine su redom piksel i radijan. Parametrom *threshold* zadaje se granična vrijednost potrebnog broja točaka da bi linija bila prepoznata. Posljednji argument nije normaliziran te uvelike ovisi o veličini slike i veličini same linije koja se traži. PPHT koristi još *param1* i *param2* gdje se prvim postavlja minimalna duljina linije, dok se drugim postavlja razmak između kolinearnih isječaka linije potreban da ih algoritam ne spoji. U nastavku slijedi isječak koda u kojem se koristi Houghova transformacija:

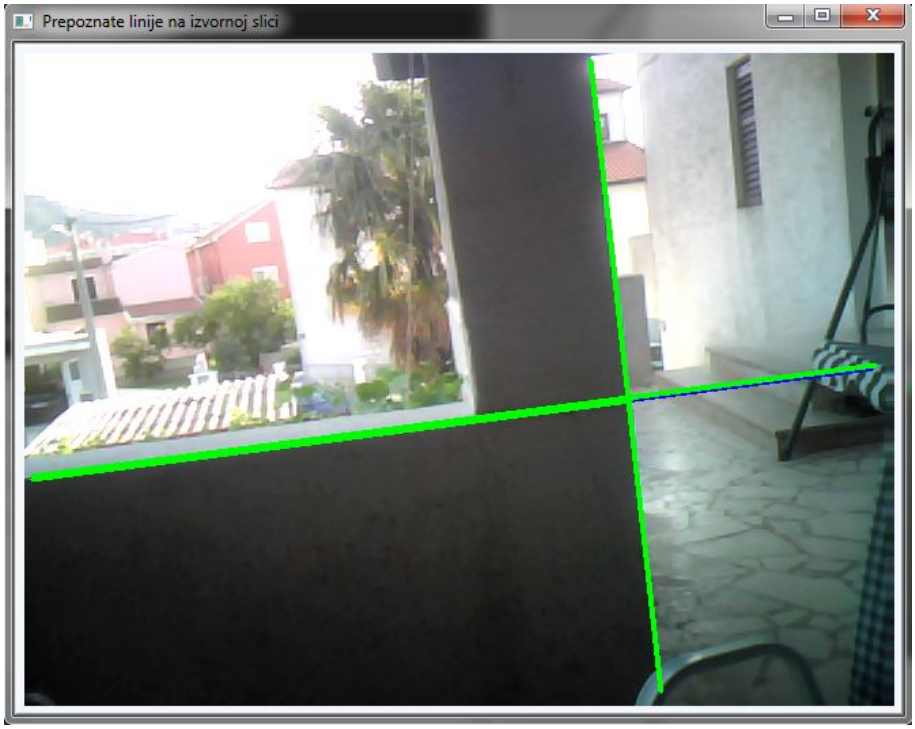
```
CvMemStorage* line_storage = cvCreateMemStorage(0);
if (crtanje)
{
CvSeq* line_seq=cvHoughLines2(sivaSlika,line_storage,
CV_HOUGH_PROBABILISTIC,1,0.01,granicaHOUGH,50,150);
//krajnje tocke linija
for( int i=0; i<line_seq->total; ++i ) {
CvPoint* p = (CvPoint*)cvGetSeqElem ( line_seq, i );
if ((i==0)&&(!medijan))
cvLine(slika,p[0],p[1],cvScalar(255,0,0),3,8);
else cvLine(slika,p[0],p[1],cvScalar(0,255,0),3,8);
}
}
```

```
CvSeq* seq_line=cvHoughLines2(sivaSlika,line_storage,
CV_HOUGH_STANDARD,1,0.01,granicaHOUGH,50,150);
//udaljenost i kut linije
double *kutevi=(double*)malloc((seq_line->total)*sizeof(double));
for( int i=0; i<seq_line->total; ++i ) {
float* rf = (float*)cvGetSeqElem ( seq_line, i );
kutevi[i]=rf[1];
printf("ro=%.4f\nfi=%.4f\n",rf[0],rf[1]);
if (i==0) kutZakreta=rf[1];//citam prvi kao kut zakreta
}
if ((seq_line->total>3)&&(medijan)) kutZakreta=MEDIJAN(kutevi,seq_line
cvReleaseMemStorage(&line_storage);
free(kutevi);
```

Algoritam je napravljen tako da se omogući crtanje prepoznatih linija. Također, omogućena su dva načina odabira konačnog horizonta iz skupa prepoznatih linija. Jedan je prepisivanje argumenta *theta* prvog člana niza rezultata. Naime, funkcija vraća prepoznate linije redoslijedom po broju glasova, stoga, najjasnija linija je prva prepoznata linija. Stoga, uz pretpostavku da horizont predstavlja najjasniju liniju na slici, jasno je prepoznavanje prve linije kao horizonta. Bitno je napomenuti kako je uz istu pretpostavku parametar *threshold* postavljen na visoku vrijednost čime se znatno smanjuje broj prepoznatih linija. Također, mogući su drugačiji odabiri horizonta što uvelike ovisi o okolini po kojoj se orijentira (implementiran je odabir horizonta kao medijana kuteva pronađenih linija). Rezultat prepoznavanja linija može se vidjeti na slici 7.7.

7.6. Zakretanje slike

Nakon pronalaska horizonta potrebno je zarotirati sliku. Skup metoda kojima se prikazuju preslikavanja na koji promatrač vidi ravninu u trodimenzionalnom prostoru (pri tom je može gledati iz svih mogućih kutova) zovu se perspektivne transformacije. Perspektivne transformacije zbog gubljenja dimenzije tijekom procesa nisu linearan operator. Međutim, perspektivne transformacije su fleksibilne, njima se mogu opisati sve moguće veze između pogleda na ravninu u trodimenzionalnom prostoru i same ravnine. Općenito, njima se preslikavaju paralelogrami u trapezoid. Kako je potrebno ostvariti geometrijsku transformaciju rotiranja, može se koristiti uža klasa transformacija, afine transformacije. Njima se oblici mogu stezati, rastezati i rotirati. Glavna ka-



Slika 7.7: Rezultat prepoznavanja linija

rakteristika takvim preslikavanjima je da se čuva paralelnost stranica, što je pokazano na slici 7.8. Afine transformacije mogu se zamisliti kao preslikavanje proizvoljnog paralelograma u proizvoljni paralelogram. Sve afine transformacije mogu se izraziti pomoću matričnog množenja i dodavanja vektora. Standardni prikaz takvih transformacija je matrica tipa 2×3:

$$\mathbf{T} = \begin{bmatrix} a_{00} & a_{01} & b_0 \\ a_{10} & a_{11} & b_1 \end{bmatrix} \tag{7.11}$$

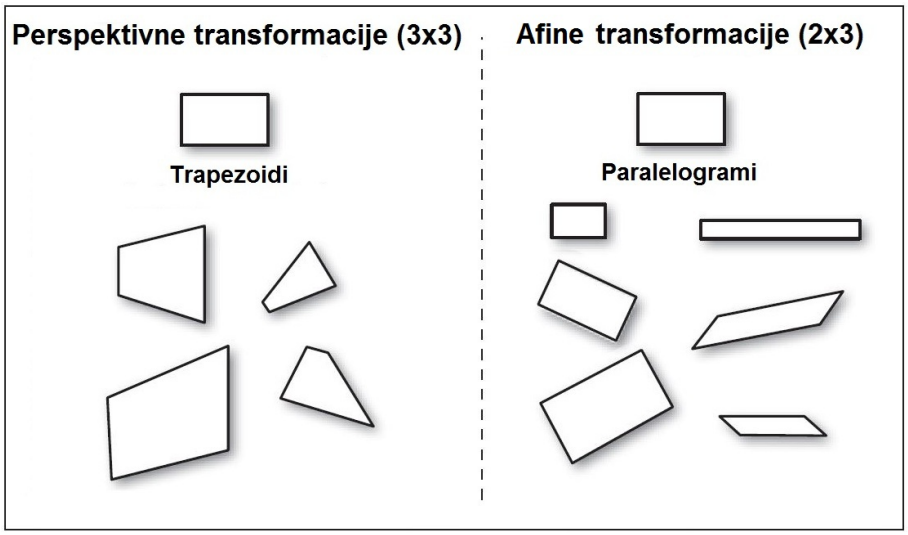
Uz:

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{7.12}$$

vrijedi:

$$\mathbf{x}' = \mathbf{T}\mathbf{x} \tag{7.13}$$

Ideja je originalnu sliku postaviti u koordinatni sustav tako da središtu slike odgovaraju koordinate $(0, 0)$. Potrebno je realizirati linearni operator rotacije za kut ϕ u spomenutom koordinatnom sustavu. Neka x i y predstavljanju koordinate proizvoljne točke na originalnoj slici, a neka x' i y' predstavljaju koordinate te točke nakon rotacije. Vezu između tih koordinata moguće je izraziti preko ortogonalne matrice operatora rotacije



Slika 7.8: Perspektivne i afine transformacije

$\mathbf{A}$:

$$\mathbf{A} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \tag{7.14}$$

Sada vrijedi:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \end{bmatrix} \tag{7.15}$$

Kako je dobiven koordinatni sustav koji središte slike preslikava u točku $(0, 0)$, potrebno je transformirati dobiveni u koordinatni sustav slike. To će se postići dodavanjem vektora središta slike, $\mathbf{b} = [x_c\ y_c]^T$. Dakle, željenu transformaciju realizira se na sljedeći način:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_c \\ y_c \end{bmatrix} \tag{7.16}$$

Ukoliko se vektor $\mathbf{x}$ proširi dodavanjem jedinice kao treće komponente, $\mathbf{x} = [x\ y\ 1]^T$ izrazi (7.13) i (7.16) se mogu poistvojetiti ukoliko se matrica afine transformacije zapiše u sljedećem obliku:

$$\mathbf{T} = \begin{bmatrix} \cos\phi & -\sin\phi & x_c \\ \sin\phi & \cos\phi & y_c \end{bmatrix} \tag{7.17}$$

OpenCV ima gotovu funkciju za afine transformacije:

```
void cvWarpAffine(
const CvArr* src,
CvArr* dst,
const CvMat* map_matrix,
```

```
int flags = CV_INTER_LINEAR | CV_WARP_FILL_OUTLIERS,
CvScalar fillval = cvScalarAll(0) );
```

Parametri predstavljaju ulaznu i izlaznu sliku. Kao parametar *map_matrix* zadaje se matrica željene transformacije tipa 2×3. Osim direktnim unošenjem matricu transformacije moguće je izračunati drugim OpenCV funkcijama, primjerice matricu rotacija funkcijom:

```
CvMat* cv2DRotationMatrix(
CvPoint2D32f center,
double angle,
double scale,
CvMat* map_matrix);
```

Parametrom *center* zadaje se točka središta rotacije. Parametrom *angle* zadaje se kut rotacije, dok se faktorom *scale* skalira konačna slika. Ako se definira $\alpha = scale \times \cos angle$ i $\beta = scale \times \sin angle$ onda funkcija izračunava *map_matrix* prema sljedećem izrazu:

$$\mathbf{M} = \begin{bmatrix} \alpha & \beta & (1-\alpha)centar_x - \beta centar_y \\ -\beta & \alpha & \beta centar_x + (1-\alpha)centar_y \end{bmatrix} \tag{7.18}$$

Da bi se omogućilo preslikavanje cijele slike neovisno o kutu rotacije i stekao bolji uvid u afine transformacije, izrađena je funkcija *zakret* koje je objašnjena u nastavku.

7.6.1. Implementacija operatora zakretanja

Operator zakretanja realiziran je tako da za svaki piksel polazne slike prolazi kroz sljedeće korake:

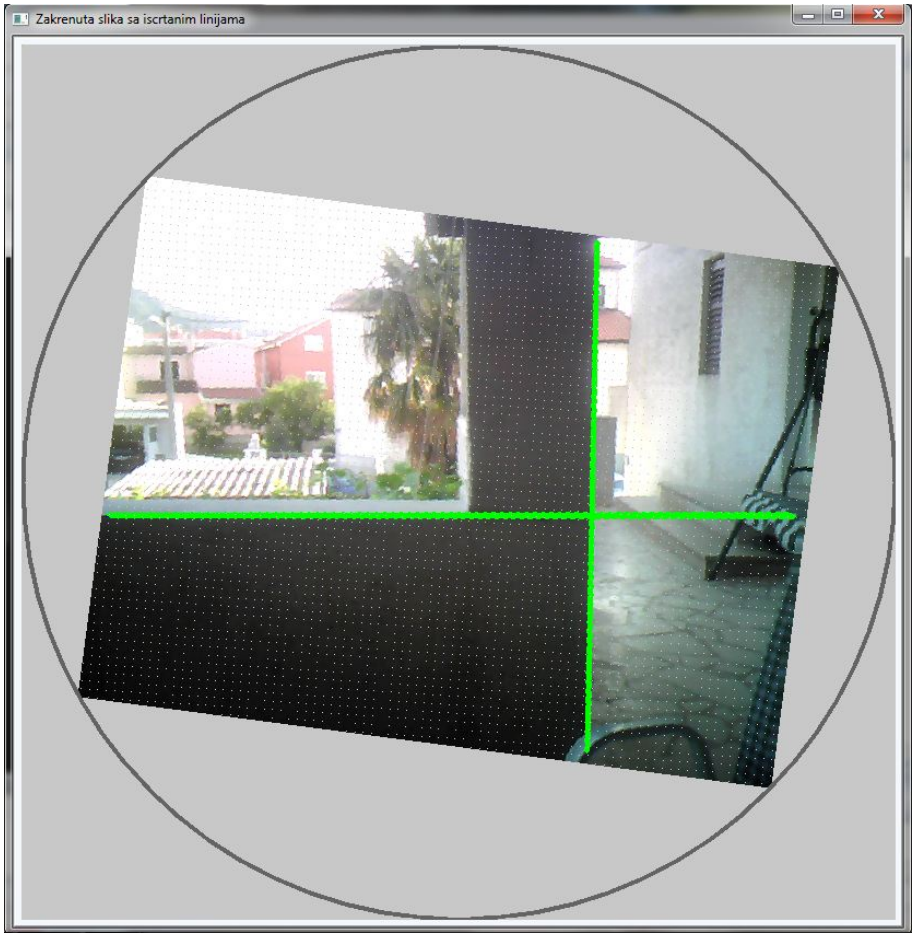
1. Računanje koordinate na polaznoj slici
2. Računanje koordinata na novoj slici na osnovu izračunatih koordinata sa polazne slike prema izrazu (7.16)
3. Preslikavanje vrijednost komponenti piksela

U nastavku slijedi isječak koda kojim je implementiran operator zakretanja.

```
for(x=0; x<slika->height; x++)
 {
  char* pokazivac = (char*) (slika->imageData +
x * slika->widthStep);
```

```
  for( y=0; y<slika->width; y++ )
  {
   //racunanje starih koordinata prema centru slike
   x0=x-stariCentarX;
   y0=y-stariCentarY;

   //racunanje novih koordinata -
   //mjesta u matrici u koje se upisuju elementi
  x1=(int)floor(matrica11*x0+matrica12*y0+noviCentarX+0.5);
    x1_pom=x1-noviCentarX;
 y1=(int)floor((matrica21*x0+matrica22*y0+noviCentarY)+0.5);
    y1_pom=y1-noviCentarY;

   if ((y1>=0)&&(y1<zakrenuto->width*3)
&&(x1>=0)&&(x1<konVisina))
   {
    for (i=0; i<slika->nChannels; i++)
    {
     //prepisivanje odgovarajucih elemenata
     zakrenuto->imageData[x1*(zakrenuto->widthStep)+
(slika->nChannels)*y1+i]=pokazivac[(slika->nChannels)*y+i];
    }
   }
  }
 }
```

Rezultati primjene operatora zakretanja prikazani su slikom 7.9. Na slici se mogu vidjeti *prazni* pikseli, odnosno pikseli unutar nove slike koji nisu preslikani. Uzrok pojave *praznih* piksela je zaokruživanje pri računanju transformacije koordinata. Za kutove $\phi \in [10° + k90°, 80° + k90°]$, $k = 0, 1, 2\ldots$, izrađeni algoritam daje nezadovoljavajuće rezultate. Stoga je s ciljem boljeg preslikavanja razvijen algoritam koji prolazi po novoj slici te računa koordinate točke koju bi trebao preslikati u promatrane koordinate. Na taj način se osigurava da nema praznih piksela. To je svojevrstan



Slika 7.9: Rezultat operatora zakretanja

inverzni operator, zbog čega se koristi inverz ortogonalne matrica rotacija:

$$\mathbf{A}^{-1} = \mathbf{A}^{T} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \tag{7.19}$$

Iz sličnosti relacija (7.18) i (7.19) može se zaključiti kako je OpenCV funkcija implementirana na ovaj *(inverzni)* način. Isječak koda u kojem je implementirana *inverzna* verzija zakretanja slijedi u nastavku.

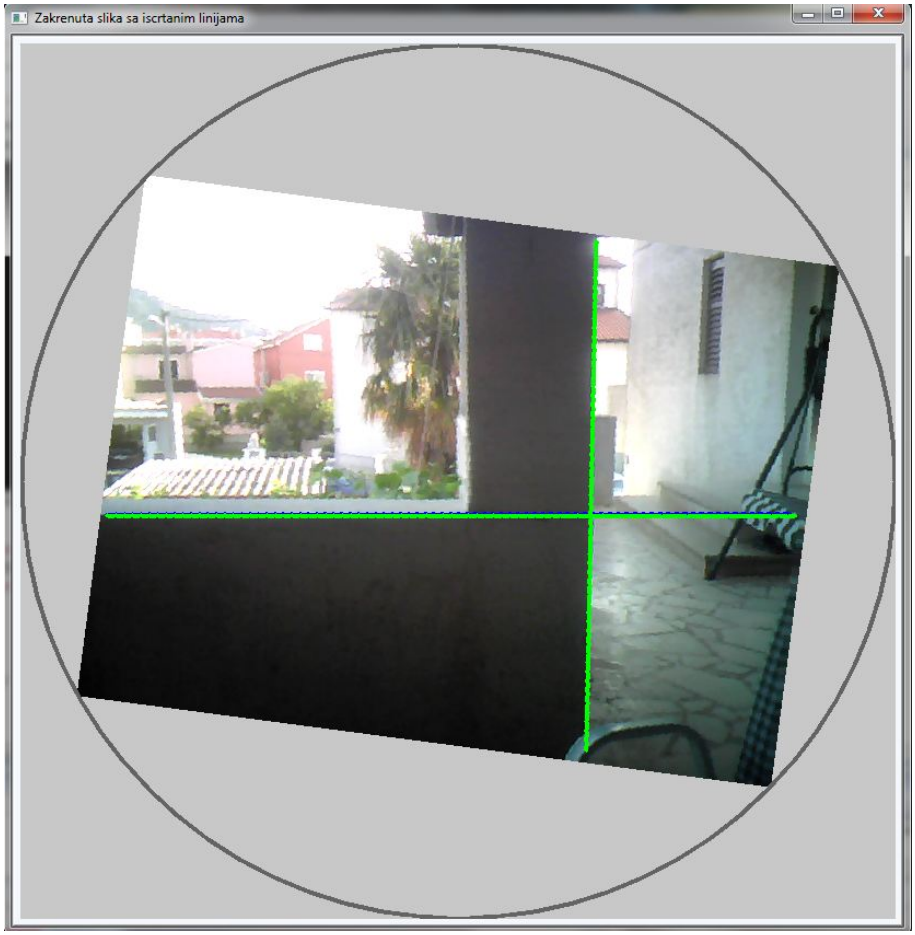
```
for(red=0; red<zakrenuto->height; red++)
 {
  char* pokazivac = (char*) (zakrenuto->imageData +
red * zakrenuto->widthStep);

  for( stup=0; stup<zakrenuto->width; stup++ )
  {
   //izracunavanje koordinata na zakrenutoj slici
```

```
    x1=red-noviCentarX;
    y1=stup-noviCentarY;
    //izracunavanje koordinata na orginalnoj slici
 x=(int)floor(matrica11*x1+matrica21*y1+stariCentarX+0.5);
 y=(int)floor(matrica12*x1+matrica22*y1+stariCentarY+0.5);
    if ((y>=0)&&(y<slika->width)&&(x>=0)&&(x<slika->height))
    {
     for (i=0; i<slika->nChannels; i++)
     {
//prepisivanje odgovarajucih elemenata
//na odgovarajuce pozicije
       zakrenuto->imageData[red*(zakrenuto->widthStep)+
(slika->nChannels)*stup+i]=slika->imageData[x*
(slika->widthStep)+y*(slika->nChannels)+i];
     }
    }
   }
 }
```

Rezultati operatora zakretanja prikazani su na slici 7.10. Slika 7.10 ujedno je i rezultat izrađenog algoritma kompenzacije kuta valjanja robota. Na slikama 7.5, 7.7 i 7.10 vidljivo je kako algoritam radi ispravno. Kao konačan rezultat, vidljivo je kako je slika zakrenuta upravo za kut koji je prepoznat kao horizont. Kako je prosječan broj obrađenih okvira u sekundi (FPS, engl. *Frames Per Second*) jednak 5, može se zaključiti kako izrađeni algoritam zadovoljava kriterije operatora kompenzacije kuta valjanja robota u stvarnom vremenu.

7.7. Integriranje rješavača inverzne kinematike u aplikaciju za obradu slike

Kako je već navedeno, drugi dio zadatka primjene vizualne povratne veze u upravljanju robotom je kompenzacija nagiba podloge zasnovana na informaciji o kutu valjanja s kamere. Razvijena aplikacija za liniju koja je odabrana kao referentna odnosno horizont vraća kut ρ. Kako je koordinatni sustav kamere zarotiran za kut $\pi/2$ u odnosu na koordinatni sustav robota, rješavaču inverzne kinematike porsljeđuje se detektirani kut umanjen za vrijednost $\pi/2$. Funkcija rješavača inverzne kinematike implementi-



Slika 7.10: Rezultat operatora zakretanja - *inverzna* verzija

rana je prema izrazu (5.29) pri čemu se koristi samo dio izraza vezan uz kompenzaciju kuta valjanja, odnosno samo treći stupac upravljačke matrice iz navedenog izraza. Kod funkcije dan je u nastavku.

```
void inverzna(double *q_tren, double theta){
 double dq[6];
 if(theta>-0.02 && theta<0.02) theta=0;
 dq[0]= (56.5*sin(q_tren[0])- 46.9*cos(q_tren[0])-25)*theta;
 dq[1]=(-56.5*sin(q_tren[1])- 46.9*cos(q_tren[1])-25)*theta;
 dq[2]= (56.5*sin(q_tren[2])- 46.9*cos(q_tren[2])-25)*theta;
 dq[3]=(-56.5*sin(q_tren[3])- 46.9*cos(q_tren[3])-25)*theta;
 dq[4]= (56.5*sin(q_tren[4])- 46.9*cos(q_tren[4])-25)*theta;
 dq[5]=(-56.5*sin(q_tren[5])- 46.9*cos(q_tren[5])-25)*theta;
 for(int i=0;i<6;i++) {
  q_tren[i]+=dq[i]/50;
```

```
  }
  CreateSendCommand(q_tren, send_data);
  if(!WriteFile(hSerial, send_data,
  sizeof(send_data), &dwBytesSent, NULL)) {
        return;
    }
}
```

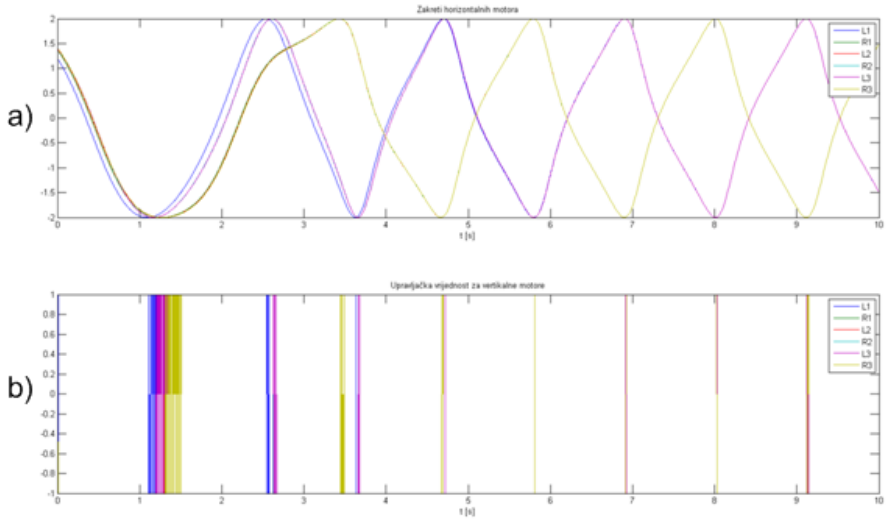
Funkcija *inverzna* poziva funkciju *CreateSendCommand* koja pretvara izračunate zakrete u upravljačku naredbu za SSC, dok je funkcija *WriteFile* standardna funkcija za pisanje na serijski port. Prirast zakreta vertikalnih motora skaliran je s faktorom $1/50$ koji je određen eksperimentalno. Skaliranje je potrebno zbog toga što algoritam računa brzinu promjene zakreta zgloba dok se upravlja pozicijom, pa se zbog brzog izvođenja aplikacije (prosječno između 4 i 5 FPS-a) s prevelikim pojačanjem gubi svojstvo konvergencije prema cilju odnosno ulazi u nestabilno područje rada. Dodavanjem funkcije rješavača inverzne kinematike neznatno je usporena brzina izvedbe cjelokupne aplikacije, ali se ipak zadržalo svojstvo izvedbe u stvarnom vremenu. Programski kod cjelokupne aplikacije zajedno s funkcijom za inicijalizaciju serijske veze dan je u dodatku.

8. Upravljanje šesteronožnim hodačem

U ovom poglavlju prikazani su rezultati implementacije algoritama upravljanja šesteronožnim hodačem. Korišteni su simulacijski odzivi snimljeni pomoću Matlaba. Video isječci s prikazom rezultata implementacije algoritama na šesteronožnom hodaču priloženi su na CD-u.

8.1. Generirana sekvenca

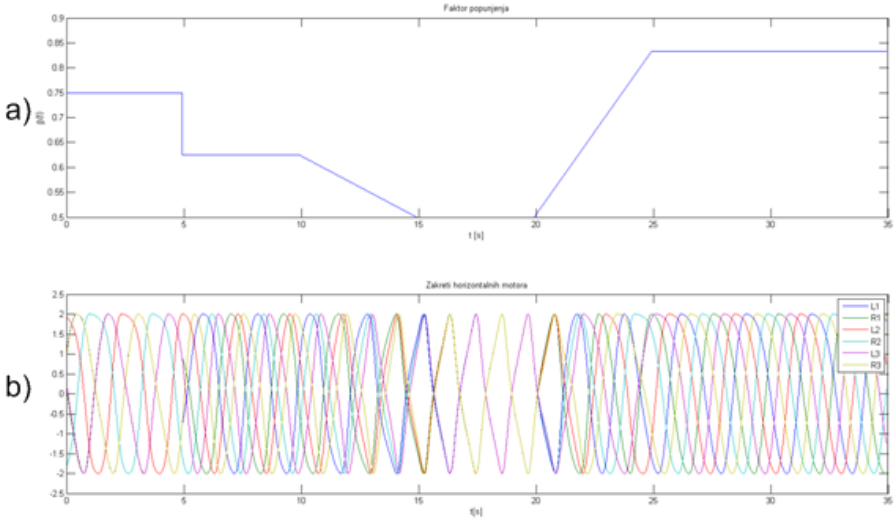
Na slici 8.1 prikazan je odziv bloka oscilatora za faktor popunjenja $\beta = 0.5$ što odgovara tronožnom koraku, pri čemu su na grafu $a)$ iscrtane trajektorije za horizontalne motore dok su na grafu $b)$ iscrtani upravljački signali za vertikalne motore. Vrijeme zamaha T_{sw} postavljeno je na vrijednost $1\ s$. Može se uočiti kako je potrebno odre-



Slika 8.1: Odziv generatora sekvence - tronožni korak

đeno vrijeme da se uspostave oscilacije nakon čega se noge grupiraju u dvije grupe, u svakoj grupi po 3 noge, koje odrađuju istu trajektoriju što je u skladu s tronožnim

korakom. Iz toga se može zaključiti kako generator sekvence radi ispravno. Na vrijeme potrebno da se uspostave oscilacije utječe faktor brzine konvergencije oscilatora α, faktor popunjenja β, ali i početne vrijednosti matrice faznih kašnjenja kao i sami početni uvjeti oscilatora. Promjena načina gibanja šesteronožnog hodača postiže se promjenom faktora popunjenja. Rezultati su prikazani na slici 8.2. Na grafu $a)$ iscrtan je faktor popunjenja β koji se mijenja u vremenu dok su na grafu $b)$ iscrtane trajektorije horizontalnih motora. Upravljački signal za vertikalne motore je izostavljen, jednako kao i dio u kojem se uspostavljaju oscilacije.

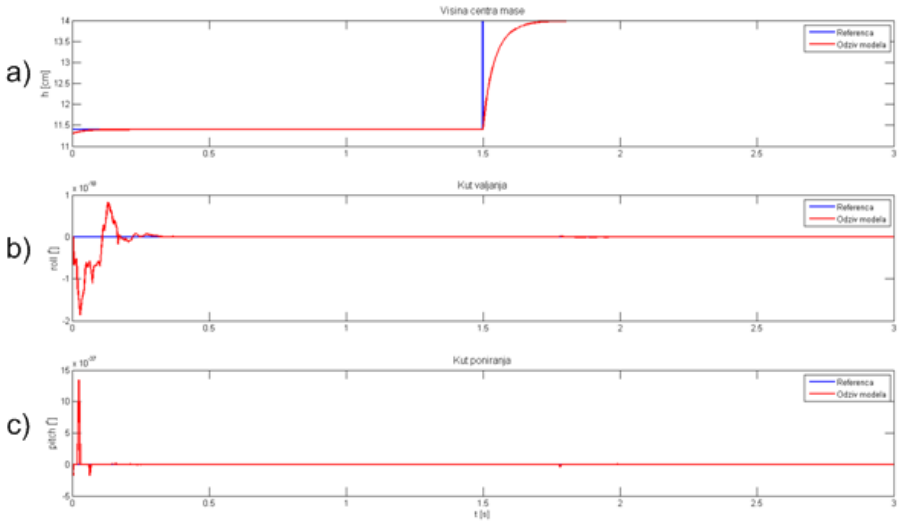


Slika 8.2: Promjena načina gibanja hodača promjenom faktora popunjenja β

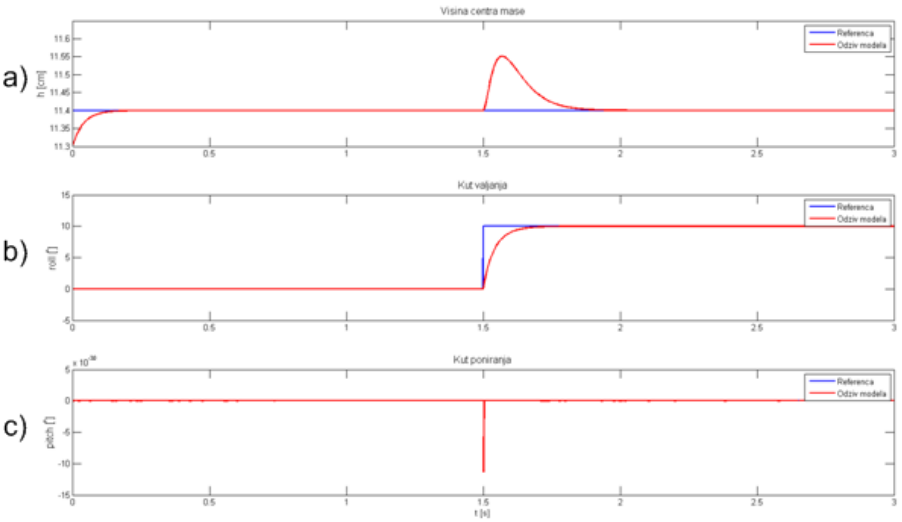
8.2. Inverzna kinematika

Prilikom prikaza simulacijskih rezultata sve su slike podijeljene u tri grafa, $a)$, $b)$ i $c)$. Na grafu $a)$ na svim slikama iscrtane su referentne i mjerene vrijednosti visine hodača, pri čemu se za referentnu vrijednost koristi plava boja, a za mjerenu vrijednost crvena. Na grafu $b)$ na svim slikama iscrtane su referentne i mjerene vrijednosti kuta valjanja hodača, pri čemu se za referentnu vrijednost koristi plava boja, a za mjerenu vrijednost crvena. Na grafu $c)$ na svim slikama iscrtane su referentne i mjerene vrijednosti kuta poniranja hodača, pri čemu se za referentnu vrijednost koristi plava boja, a za mjerenu vrijednost crvena. Rezultat upravljanja visinom hodača dan je slikom 8.3. Može se uočiti kako se zadana vrijednost visine postiže u kratkom vremenu (otprilike $0.25s$) dok se istovremeno održavaju zadane vrijednosti za kuteve valjanja i poniranja.

Rezultat upravljanja kutom valjanja hodača dan je slikom 8.4. Brzina postizanja za-



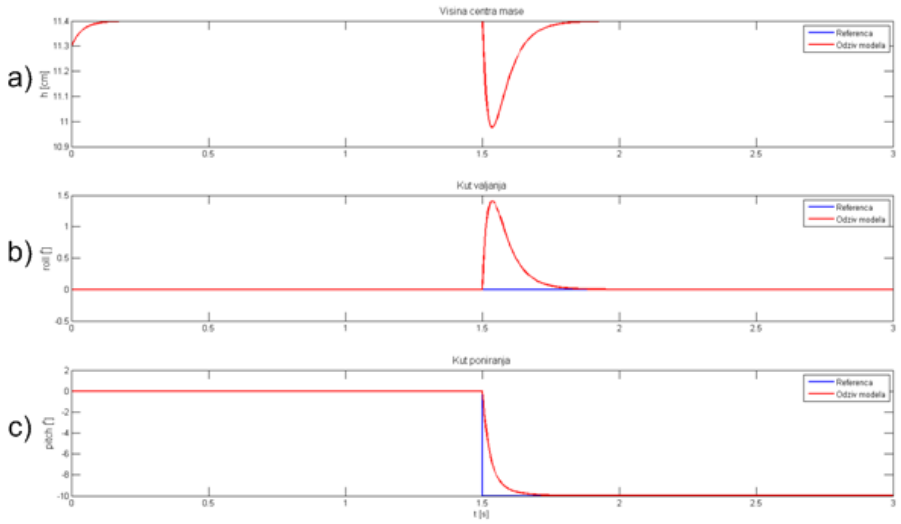
Slika 8.3: Upravljanje visinom hodača



Slika 8.4: Upravljanje kutom valjanja hodača

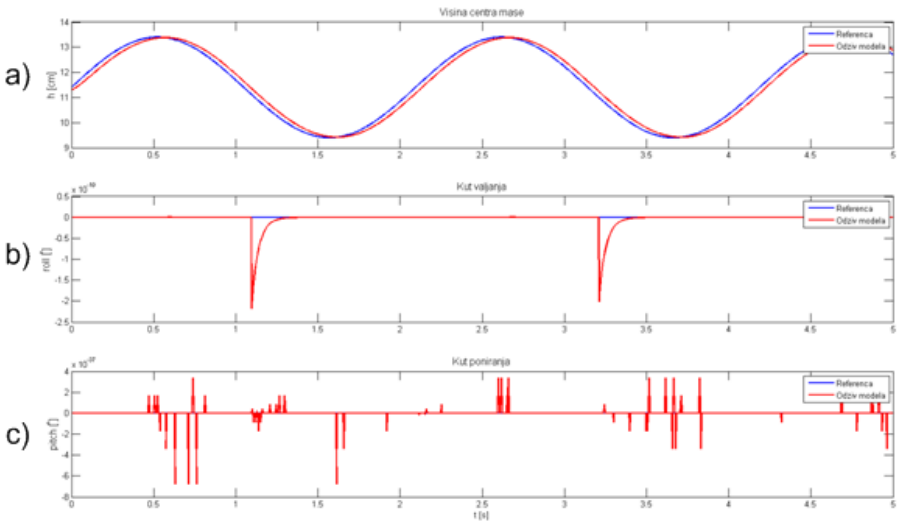
dane vrijednosti gotovo je identična kao i kod upravljanja visinom. Može se uočiti da u trenutku skokovite promjene referentne vrijednosti kuta valjanja dolazi do odstupanja mjerene vrijednosti visine hodača od zadane. Ipak, kako je promjena po iznosu neznatna (otprilike $1\ mm$) te se vrlo brzo kompenzira može se zaključiti kako je upravljanje zadovoljavajuće. Prilikom skokovite promjene referentne vrijednosti kuta valjanja nema značajnije promjene mjerene vrijednosti kuta poniranja.

Rezultat upravljanja kutom poniranja hodača dan je slikom 8.5. Ponovno je brzina



Slika 8.5: Upravljanje kutom poniranja hodača

odziva jednaka kao i u slučaju skokovite promjene reference visine i kuta poniranja. Također dolazi do odstupanja u visini centra mase hodača, koje iznosi $2\ mm$, ali se odstupanje uspješno eliminira. Jednako tako eliminira se odstupanje mjerene vrijednosti kuta valjanja od referentne vrijednosti koje iznosi $1.5°$.
Na slici 8.6 prikazano je slijeđenje periodičke reference visine centra mase hodača.

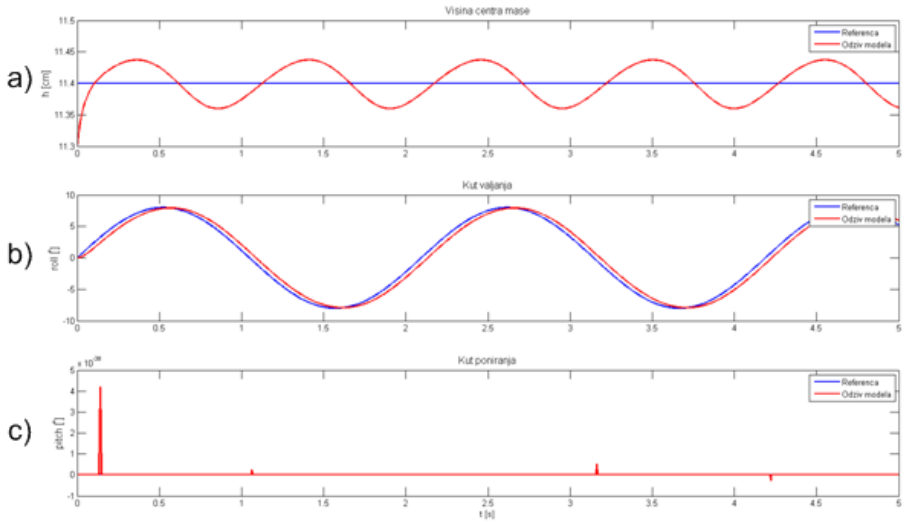


Slika 8.6: Slijeđenje periodičke reference visine

Uočava se kako hodač zadovoljavajuće prati zadanu referencu visine istovremeno odr-

žavajući zadane vrijednosti kutova valjanja i poniranja.

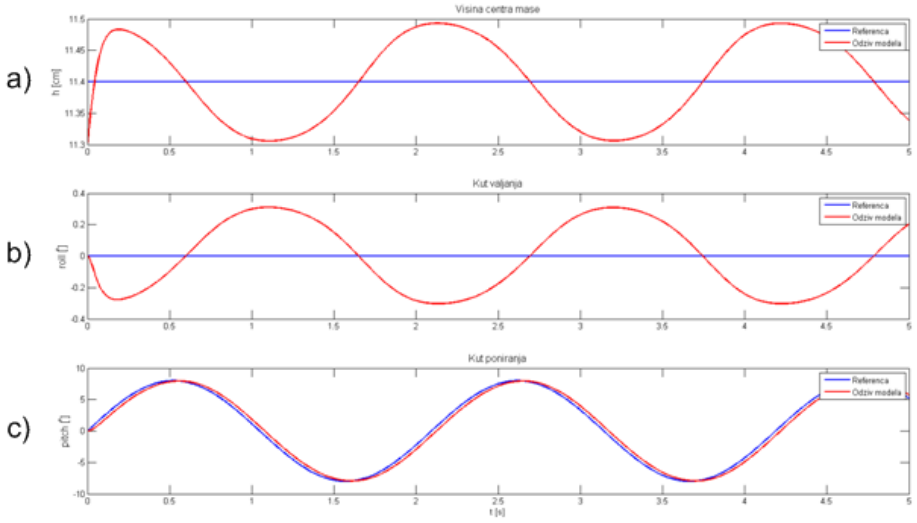
Na slici 8.7 prikazano je slijeđenje periodičke reference kuta valjanja. Slijeđenje refe-



Slika 8.7: Slijeđenje periodičke reference kuta valjanja

retne vrijednosti je zadovoljavajuće. Pojavljuje se odstupanje od zadane visine centra mase koje je zanemarivo. Šiljci u odzivu kuta poniranja zanemarivog su iznosa i smatraju se posljedicom numeričkog postupka izračunavanja.

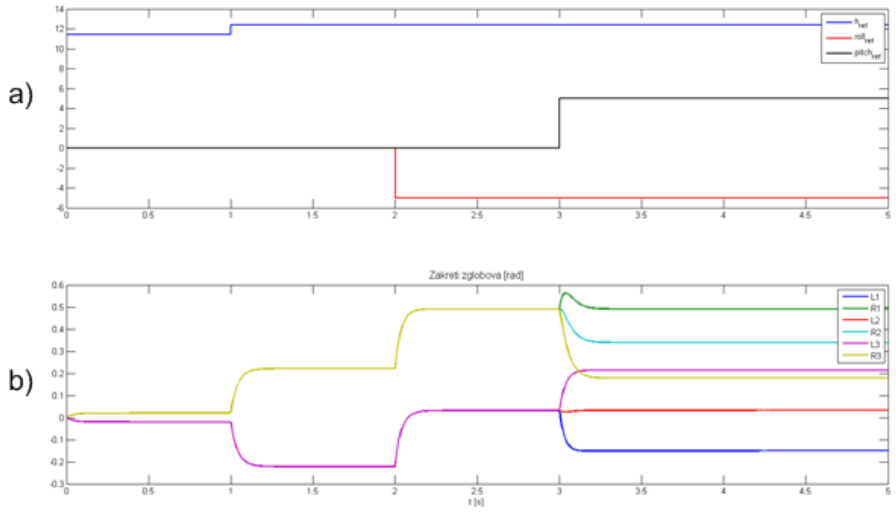
Na slici 8.8 prikazano je slijeđenje periodičke reference kuta poniranja. Uočava se



Slika 8.8: Slijeđenje periodičke reference kuta poniranja

kako mjerena vrijednost kuta valjanja zadovoljavajuće slijedi referentnu. U slučaju pe-

riodičke referentne vrijednosti kuta poniranja pojavljuju se odstupanja od referentnih vrijednosti i za visinu centra mase i za kut valjanja, pri čemu su oba odstupanja periodičkog oblika. Međutim, ponovno su odstupanja zanemariva te se može zaključiti kako implementirani rješavač inverzne kinematike radi ispravno.
Dodatno je slikom 8.9 prikazano ponašanje zakreta vertikalnih motora ovisno o promjeni referentne vrijednosti veličina kojima se upravlja. Na grafu $a)$ prikazane su referentna vrijednost visine hodača (plava boja), referetna vrijednost kuta valjanja (crvena boja) i referetna vrijednost kuta poniranja (crna boja). Na grafu $b)$ prikazane su vrijednosti zakreta vertikalnih motora.



Slika 8.9: Zakreti vertikalnih motora

8.3. Kompenzacija kuta valjanja pomoću vizualne povratne veze

Integracijom rješavača inverzne kinematike u aplikaciju za dohvat i obradu slike ostvarena je kompenzacija detektiranog kuta valjanja. S brzinom između 3 i 5 okvira u sekundi algoritam obrade slike i rješavač inverzne kinematike zadovoljavajuće rade u stvarnom vremenu. Ostvareni rezultati prikazani su u obliku video isječaka koji su priloženi na CD-u.

9. Zaključak

Istražena je mogućnost primjene nelinearnih Hopfovih oscilatora za generiranje sekvence hodajućih robota. Zbog stabilnih rješenja nelinearnih diferencijalnih jednadžbi oscilatora kojima se generiraju gotovo harmoničke oscilacije, zaključeno je kako su Hopfovi oscilatori primjenjivi, što je pokazano na primjeru šesteronožnog hodača. Detaljno je proučena teorija vijčanih gibanja za koju se pokazalo kako omogućuje relativno jednostavan način rješavanja kinematičkih problema kako za robotske manipulatore tako i za hodajuće robote, naravno s određenim modifikacijama. Od posebnog je značaja postupak određivanja Jacobian matrice koja se često koristi u upravljanju robotskim sustavima. Na primjeru šesteronožnog hodača pokazano je kako inverzna kinematika zasnovana na teoriji vijčanog gibanja, odnosno Jacobian matrici daje zadovoljavajuće rezultate čak i uz nepostojanje povratne veze po položaju zglobova samog robota. Zbog nepostojanja povratne veze po položaju i orijentaciji robota istražena je mogućnost dobivanja djelomične informacije o orijentaciji robotskog sustava u prostoru pomoću vizalne povratne veze ostvarene web kamerom. Pritom je razrađen općeniti algoritam koji se zasniva na detekciji linije horizonta. Primjena vizualne povratne veze razrađena je u dva smjera, rotiranje slike na operatorskom panelu s ciljem olakšavanja upravljanja nekim robotom samom operateru te prosljeđivanje informacije o detektiranom kutu valjanja upravljačkim algoritmima robota koji se giba u prostoru. Za slučaj šesteronožnog hodača koji je autonoman hodajući robot pokazano je kako se integriranjem vizualne povratne veze i rješavača inverzne kinematike zasnovanog na teoriji vijčanog gibanja mogu postići zadvoljavajući rezultati kompenzacije detektiranog kuta valjanja. Uspješnost implementacije algoritama i relativno zadovoljavajuće ponašanje šesteronožnog hodača, koji je zbog manjkavosti u mehaničkoj kostrukciji vrlo ograničen robotski sustav, znak su kako su teorija vijčanog gibanja i vizualna povratna veza *moćni alati* kojima se može puno postići u upravljanju hodajućim robotima.

10. Literatura

[1] Matlab and simulink tips and tricks. `http://161.53.68.66/wiki/index.php/Matlab_and_Simulink`, Travanj 2011.

[2] Opencv library wiki. `http://opencv.willowgarage.com/wiki/`, Travanj 2011.

[3] Ssc-32 manual. `http://www.lynxmotion.com/images/html/build136.htm`, Travanj 2011.

[4] A. Aglić-Aljinović, N. Elezović, i D. Žubrinić. *Linearna Algebra*. Element, 2011.

[5] A. Ahmadi, E. Mangieri, K. Maharatna, i M. Zwolinski. Physical realizable circuit structure for adaptive frequency hopf oscillator. 2009.

[6] Jorge Angeles. *Fundamentals of Robotic Mechanical Systems: Theory, Methods and Algorithms*. Springer, 2007.

[7] Samuel R. Bass. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. 2009.

[8] G. Bradski i A. Kaebler. *Learning OpenCV - Computer Vision with the OpenCV Library*. O'Reilly Media, Inc - Safari books online (Internet edition), 2008.

[9] R. Campos, V. Matos, i C. Santos. *Hexapod Locomotion: a Nonlinear Dynamical Systems Approach*. Industrial Electronics Department, University of Minho, Guimaraes, Portugal, 2010.

[10] Qiushi Fu. *Kinematics of Articulated Wheeled Robots: Exploiting Reconfigurability and Redundancy*. Department of Mechanical and Aerospace Engineering, State University of New York at Buffalo, 2008.

[11] Z. Kovačić, S. Bogdan, i V. Krajči. *Osnove robotike*. Graphis, 2002.

[12] R.M Murray, Z. Li, i S.S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[13] A. Mutka i Z. Kovacic. A leg-wheel robot-based approach to the solution of flipper-track robot kinematics. 2011.

[14] L. Righetti i A.J. Ijspeert. Pattern generators with sensory feedback for the control of quadruped locomotion. 2008.

[15] Z. Vukić, LJ. Kuljača, D. Đonlagić, i S. Tešnjak. *Nonlinear Control Systems*. Marcel Dekker, Inc., 2003.

Primjena teorije vijčanog gibanja i vizualne povratne veze u upravljanju hodajućim robotima

Sažetak

Opisan je centralni generator sekvence. Generiranje hoda zasnovano je na sinkroniziranim Hopfovim oscilatorima koji generiraju trajektorije za noge šesteronožnog robotskog hodača. Sinkronizacija i koordinacija vrše se promjenom relativnih faza između oscilatora. Glatki prijelazi među načinima hoda ostvaruju se promjenom faktora popunjenja. Iznesene su osnove teorije vijčanog gibanja. Pokazano je kako je teorija vijčanog gibanja primjenjiva u robotici, kako na robotskim manipulatorima tako i na mobilnim ili hodajućim robotima. Korištenje teorije vijčanog gibanja pri rješavanju direktnog i inverznog kinematičkog problema prikazano je na primjeru šesteronožnog robotskog hodača. Kako bi se ostvarilo upravljanje robotom, upravljački algoritmi implementirani su u Matlabu (Simulink) u obliku sistemskih funkcija (s-funkcija). Za ostvarenje vizualne povratne veze koristi se web kamera. Za obradu slike s ciljem detekcije linije horizonta, koriste se funkcije i procedure OpenCV biblioteke. Prilikom detekcije linije horizonta izračunava se i kut valjanja kamere i robota. Slika se zatim rotira za izračunati kut kako bi se operateru olakšalo upravljanje robotom. Kut valjanja se također prosljeđuje rješavaču inverzne kinematike s ciljem kompenzacije nagiba podloge. Rezultati su prikazani u obliku simulacijskih odziva iz Simulinka te slika koje su rezultat obrade slike funkcijama OpenCV-a. Primjena algoritama upravljanja na šesteronožnom hodaču prikazana je u obliku video isječaka koji su priloženi na CD-u.

Ključne riječi: šesteronožni robotski hodač, oscilator, vijčano gibanje, teorija vijčanog gibanja, kinematika, Jacobian, kamera

The use of screw theory and visual feedback in control of walking robots

Abstract

Central pattern generator is presented. Gait generation is based on sinchronized Hopf oscillators which generate trajectories for legs of the six-legged robot. Synchronization and coordination are achieved by changing relative phases between oscillators. Smooth gait transition is achieved by changing the duty factor. Basics of screw theory are introduced. It is shown that screw theory is applicable in robotics, both on robotic manipulators and mobile or walking robots. Use of screw theory in forward and inverse kinematics problems for walking robots is presented on the six-legged robot. To achieve control over the robot, control algorithms are implemented in Matlab (Simulink) in form of S-Functions. A web-camera is used to support the visual feedback. OpenCV functions and procedures are used for image processing in order to detect the line of the horizon. Through detection of the line of the horizon roll angle is calculated. Picture is then rotated by retrieved angle to ease the control over the robot for the operator. Retrieved angle is also forwarded to inverse kinematics solver in order to achieve lateral posture control of the robot. Finally, results are shown in form of simulation responses retrieved from Simulink and images retrieved from OpenCV functions. Application of developed algorithms on six-legged robot is shown in form of video clips which are enclosed on the CD.

Keywords: six-legged robot, oscillator, screw motion, screw theory, kinematics, Jacobian, camera