

Univerza v Ljubljani

Fakulteta za elektrotehniko

Timotej Gašpar

Vodenje robota v stiku s podajnim objektom

Magistrsko delo

Mentor: prof. dr. Matjaž Mihelj

Somentor: dr. Leon Žlajpah

Ljubljana, 2015

Zahvala

Vsebina

1	Uvod	5
2	Robotski manipulator PA-10	7
2.1	Kinematični model robota PA-10	7
2.2	Dinamičen model robota	12
2.3	Inverzna kinematika	14
3	ARCNET - UDP strežnik za komunikacijo s krmilnikom robota ter senzorjem sile	17
3.1	Krmilnik servo motorjev robota	18
3.2	ARCNET vmesnik	20
3.3	Senzor sile in navorov - JR3	23
3.4	Strežnik	25
3.4.1	Delovanje programa	26
3.4.2	Uporaba programa	28
3.4.3	Varnostni ukrepi	30
4	Vodenje robota PA-10	33
4.1	Kompenzacija mase prijemala ter merilnega odmika na senzorju sile JR3	33

4.2	Vodenje preko inverzne kinematike	37
4.2.1	Vodenje v notranjih koordinatah	37
4.2.2	Vodenje v zunanjih koordinatah	38
4.3	Krmiljenje z inverzno dinamiko	40
4.3.1	Vodenje v notranjih koordinatah	40
4.3.2	Vodenje v zunanjih koordinatah	42
4.4	Interakcija z okoljem	43
4.5	Admitančno vodenje	44
4.6	Impedančno vodenje	45
5	Implementacija	47
5.1	Hitrostno krmiljenje	48
5.1.1	Vodenje po trajektoriji	48
5.1.2	Admitančno vodenje	51
5.2	Navorno krmiljenje	51
5.2.1	Kompenzacija trenja	51
5.2.2	Vodenje po trajektoriji	51
5.2.3	Vodenje v kontaktu z okolico	51
6	Zaključek	53
A	Dodatek 1	59
B	Dodatek 2	61
C	Dodatek 3	63

Seznam slik

2.1	Skica robotskega mehanizma PA 10.	8
2.2	Dva zaporedna člena kinematične verige, povzeto po [1]	9
2.3	Tehnična risba robota PA-10. Iz nje se definira D-H parametre. Povzeto po [2].	10
3.1	Tehnična risba ohišja krmilnika servo motorjev. Povzeto po [2]. . .	19
3.2	Vsebina ARCNET paketa	20
3.3	Diagram stanj ARCNET modula na krmilniku robota	21
3.4	Krmiljenje robota s pošiljanjem ARCNET paketov.	21
3.5	Podatkovni del ARCNET paketa za krmiljenje robota	22
3.6	Senzor sil in navorov JR3.	23
3.7	Dostopanje do registrov na ISA kartici za digitalno obdelavo si- gnala iz senzorja senzorja JR3.	24
3.8	Strežnik kot posrednik med ARCNET ter Ethernet omrežjem. . .	26
3.9	Diagram poteka programa.	29
3.10	Nalaganje ARCNET modula v jedro Linux.	29
3.11	Zaganjanje strežnika na Linux operacijskem sistemu.	30
4.1	Koordinatni sistem senzorja sile ter koordinatni sistem težišča pri- jemala	34

4.2	Povratna zanka za krmiljenje kota v sklepu	38
4.3	Bločna shema krmiljenja referenčne lege vrha robota	39
4.4	Bločna shema vodenja po notranjih koordinatah preko inverzne dinamike.	42
4.5	Bločna shema vodenja po zunanjih koordinatah preko inverzne di- namike.	43
4.6	46
5.1	Odziv sklepa na signal stopnice.	48
5.2	Odziv sklepa na signal sinusne oblike.	49
5.3	Napaka odziva in ovojnica signala napake.	50
5.4	Sledenje referenci v obliki kroga na x in y ravnini	50

Seznam tabel

1	Veličine in simboli	xi
2.1	D-H parametri	10
3.1	Opis prejetega UDP paketa.	27
3.2	Opis ARCNET paketa.	27
3.3	Opis poslanega UDP paketa.	28

Seznam uporabljenih simbolov

V pričujočem zaključnem delu so uporabljeni naslednje veličine in simboli:

Veličina / oznaka		Enota	
Ime	Simbol	Ime	Simbol
čas	t	sekunda	s
frekvenca	f	Hertz	Hz
sila	F	Newton	N
masa telesa	m_t	kilogram	kg
kot v sklepu	q	radijan na sekundo	rad/s
lega vrha robota	\mathbf{x}	metri	m
Jacobijeva matrika	\mathbf{J}	-	-

Tabela 1: Veličine in simboli

Pri čemer so vektorji in matrike napisani s poudarjeno pisavo. Natančnejši pomen simbolov in njihovih indeksov je razviden iz ustreznih slik ali pa je pojasnjen v spremljajočem besedilu, kjer je simbol uporabljen.

Povzetek

Ključne besede: beseda1, beseda2, beseda3

Abstract

The thesis addresses ...

Key words: word1, word2, word3

1 Uvod

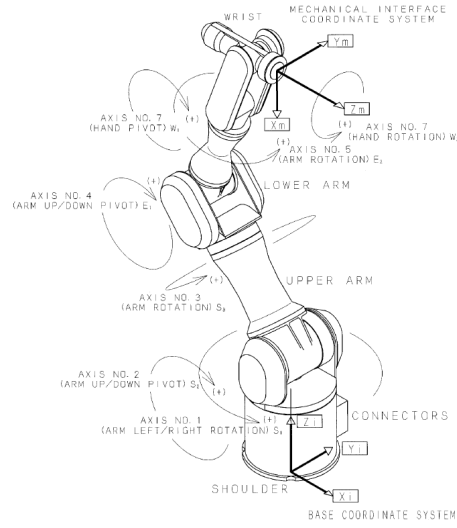
2 Robotski manipulator PA-10

Podjetje Mitsubishi Heavy Industries je leta 1992 izdelalo prvega katalogiranega industrijskega redundatnega robota [3]. Podjetje je robota poimenovalo Portable General-purpose Intelligent Arm PA-10, krajše PA-10. Gre za serijskega robota s sedmimi stopnjami prostosti (slika 2.1). Prvi trije sklepi so označeni kot ramenski sklepi (shoulder), S1, S2, S3. Naslednja dva sta označena kot komolčni sklepi (elbow), E1, E2. Zadnja dva sklepa pa sta označena kot zapestna (wrist), W1, W2. Glede na zgradbo se ga lahko uvrsti v tako imenovane antropomorfne robote [4]. Značilnost takih robotov je spretnost saj so vsi sklepi rotacijski [1].

Masa robotske roke PA-10 je 36 kg in ima nosilnost 10 kg. Servo motorji v sklepih se napajajo preko izmenične napetosti. Prenosi med v sklepih so realizirani s harmoničnimi gonili. Baza robota se lahko pritrdi v katerokoli lego. To pomeni, da se ga lahko fiksira bodisi na tla, na steno ali na strop. Robota PA-10 odlikuje relativno lahka konstrukcija, enostavno rokovanje ter odprtost njegovega krmilnika. Prav ti razlogi so povod, da je mnogo instituciji vzelo tega robota kot eksperimentalni sistem za razvijanje raznih algoritmov vodenja ([5, 6, 7, 8]).

2.1 Kinematični model robota PA-10

Robotski mehanizem obravnavamo kot kinematično verigo n med seboj povezanih togih teles, rečemo jim tudi kinematični pari. Ker je en konec kinematične verige, t.j. baza robota, toga pritrjen v bazo se premika le vrh kinematične verige. Z opisom kinematične relacije med dvema zaporednima segmentoma je mogoče defini-



Slika 2.1: Skica robotskega mehanizma PA 10.

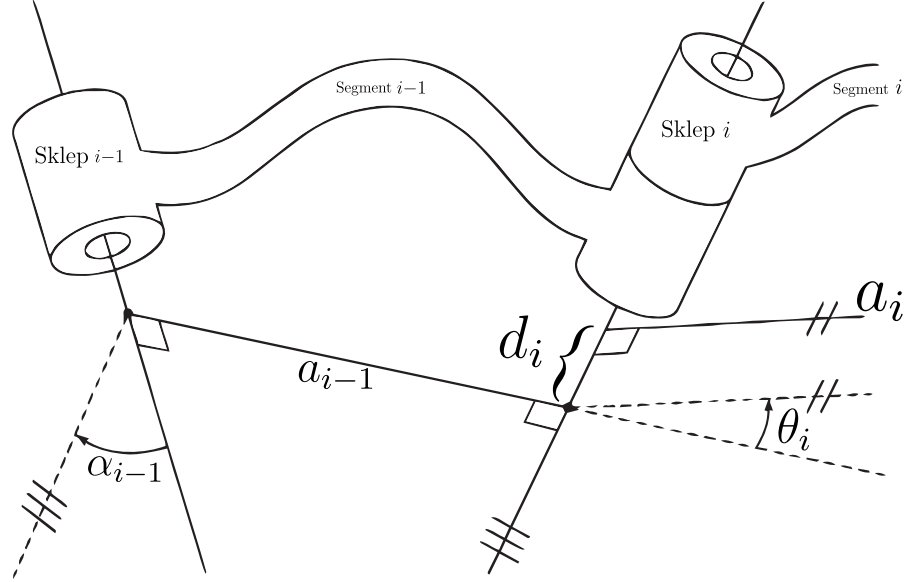
rati kinematično relacijo med bazo in vrhom robota. Homogena transformacijska matrika je operator, ki opisuje translacijo in rotacijo med dvema koordinatnima sistemoma. Definiramo jo kot [1]

$$\mathbf{T}_{i+1}^i = \begin{bmatrix} \cos(\Theta_i) & -\sin(\Theta_i) & 0 & a_i \\ \sin(\Theta_i) \cos(\alpha_i) & \cos(\Theta_i) \cos(\alpha_i) & -\sin(a_i) & -\sin(a_i)d_i \\ \sin(\Theta_i) \sin(\alpha_i) & \cos(\Theta_i) \sin(\alpha_i) & \cos(a_i) & \cos(a_i)d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (2.1)$$

kjer so a_i , α_i , d_i in Θ Denavit - Hartenberg (D-H) parametri in opisujejo relacijo med dvema koordinatnima sistemoma.

Sklicujoč na sliko 2.2 se D-H parametre parametre opiše kot:

- a_i razdalja med O_i in O_{i+1} v smeri x_i ,
- α_i kot med O_i in O_{i+1} glede na os x_i ,
- d_i razdalja med O_{i-1} in O_i glede na os z_i ,
- Θ razdalja med O_{i-1} in O_i glede na os z_i ,



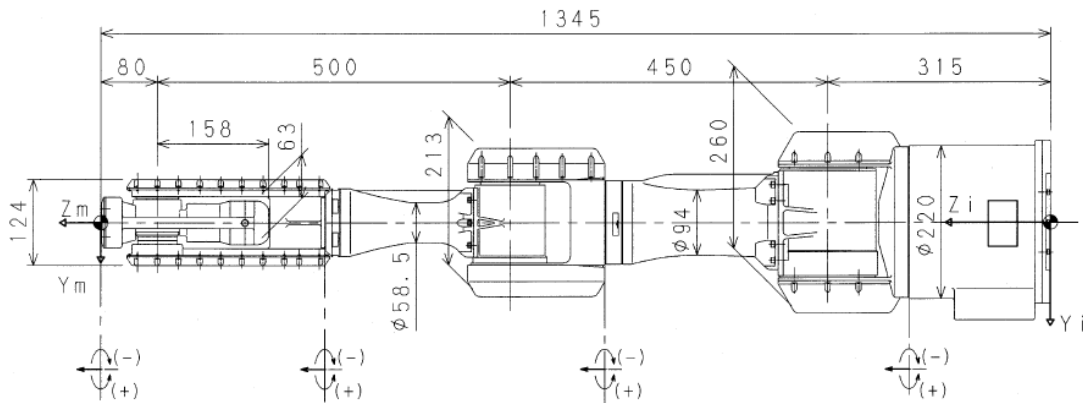
Slika 2.2: Dva zaporedna člena kinematične verige, povzeto po [1]

Z množenjem homogenih transformacijskih matrik posameznih sklepov med seboj dobimo homogeno transformacijo med vrhom robota in njegovo bazo:

$$\mathbf{T}_7^0 = \mathbf{T}_1^0 \mathbf{T}_2^1 \dots \mathbf{T}_7^6. \quad (2.2)$$

D-H parametre za robota PA-10 definiramo na osnovi podatkov proizvajalca [2] in so podani na sliki 2.3. V tabeli 2.1 so zapisani D-H parametri za vse kinematične pare. Koti v sklepih robota so zapisani kot q_n in jih imenujemo notranje koordinate. Glede na D-H sistem v našem primeru q_i ustreza parametru θ_i . Z upoštevanjem tega postane matrika \mathbf{T}_{i+1}^i funkcija kotov v sklepih $\mathbf{q} = [q_1 \ q_2 \ \dots \ q_7]^T$.

Opis lege vrha robota se da skrajšati in zapisati z minimalnim številom koordinat. V matriki homogene transformacije lahko poiščemo vektor translacije \mathbf{p} in rotacijsko matriko \mathbf{R} (slika 2.3).



Slika 2.3: Tehnična risba robota PA-10. Iz nje se definira D-H parametre. Povzeto po [2].

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0.315	$q1$
2	$-\frac{\pi}{2}$	0	0	$q2$
3	$\frac{\pi}{2}$	0	0.45	$q3$
4	$-\frac{\pi}{2}$	0	0	$q4$
5	$\frac{\pi}{2}$	0	0.5	$q5$
6	$-\frac{\pi}{2}$	0	0	$q6$
7	$\frac{\pi}{2}$	0	0.08	$q7$

Tabela 2.1: D-H parametri

$$\mathbf{T} = \left[\begin{array}{c|c} \mathbf{R}^{3 \times 3} & \mathbf{p}^{1 \times 3} \\ \hline \mathbf{0}^{3 \times 1} & 1 \end{array} \right] \quad (2.3)$$

Pozicija se enostavno izpiše iz prvih treh vrstic zadnjega stolpca, vektor \mathbf{p} . Za zapis orientacije pa se je potrebno sklicati na Eulerjeve kote, ki jih lahko izračunamo iz členom podmatrike \mathbf{R} . Če omenjeno podmatriko zapišemo kot

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \quad (2.4)$$

lahko definiramo tri Eulerjeve kote na sledeč način

$$\varphi = \arctan 2(r_{21}, r_{11}), \vartheta = \arctan 2(-r_{31}, \sqrt{r_{32}^2 + r_{33}^2}), \psi = \arctan 2(-r_{32}, -r_{33}). \quad (2.5)$$

Z združitvijo treh koordinat za opis pozicije in treh koordinat za opis orientacije se lahko definira vektor, ki opisuje neko točko v prostoru baze robota:

$$\mathbf{x} = \begin{bmatrix} x & y & z & \varphi & \vartheta & \psi \end{bmatrix}^T. \quad (2.6)$$

Vektor x se imenuje tudi vektor zunanjih koordinat. Sedaj je mogoče opisati enačbo direktne kinematike kot funkcijo \mathbf{q} :

$$\mathbf{x} = \mathbf{k}(\mathbf{q}). \quad (2.7)$$

Funkcija $\mathbf{k}(\mathbf{q})$ je vektorska funkcija notranjih koordinat, v katerih so zajete kinematične enačbe pozicije in orientacije mehanizma. Problem direktne kinematike je pri serijskih mehanizmih enostavno rešljiv in ima eno rešitev [9].

Potrebno je še opisati, kakšna je hitrost vrha robota v odvisnosti od hitrosti v sklepov. Z odvajanjem enačbe 2.7 po \mathbf{q} dobimo

$$\dot{\mathbf{x}} = \frac{\partial \mathbf{k}(\mathbf{q})}{\partial \mathbf{q}} \dot{\mathbf{q}} = \mathbf{J} \dot{\mathbf{q}}, \quad (2.8)$$

kjer je \mathbf{J} Jacobijeva matrika in predstavlja parcialne odvode zunanjih koordinat.

Z odvajanjem enačbe 2.8 je mogoče zapisati še relacijo med pospeški vrha robota ter pospeški sklepov

$$\ddot{\mathbf{x}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}, \quad (2.9)$$

kjer je $\dot{\mathbf{J}}$ odvod Jacobijeve matrike.

2.2 Dinamičen model robota

Vodenje robota z referenčnimi navori veleva poznavanje njegovega dinamičnega modela. S poznavanjem dinamičnega modela je mogoče opisati silo s katero bo vrh robota deloval v kontaktu z okolico. Z razliko od kinematičnega modela je pri dinamičnem modelu število parametrov, ki vplivajo na vodenje večje. V nadaljevanju bo predstavljen splošen dinamičen model in relacija med silami, ki delujejo na vrhu robota in navori v sklepih.

Enačbe gibanja kot posledica delovanja sil in navorov se lahko zapišejo z uporabo Lagrangevih formulacij

$$\begin{aligned} \mathcal{L}(q, \dot{q}) &= \mathcal{T}(q, \dot{q}) - \mathcal{U}(q), \\ \boldsymbol{\tau} &= \frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i} \right) - \frac{\partial \mathcal{L}}{\partial q_i} \end{aligned} \quad (2.10)$$

kjer \mathcal{T} in \mathcal{U} opisujeta kinetično in potencialno energijo.

Lagrangeve formulacije so orodje za sistematičen opis dinamike posameznih segmentov [1]. Z uporabo formulacije 2.10 se lahko zapiše enačbo navorov v sklepih v odvisnosti od kotov, hitrosti in pospeškov v sklepih na sledeč način:

$$\boldsymbol{\tau}(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q}) = \mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_f(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{g}(\mathbf{q}) + \mathbf{J}^T(\mathbf{q})\mathbf{h}_o. \quad (2.11)$$

Posamezni členi zgornje enačbe bodo opisani v nadaljevanju.

Matrika $\mathbf{B}(\mathbf{q})$ predstavlja zapis vztrajnosti posameznih segmentov. Odvisna je od trenutne konfiguracije robota. Posamezni členi pa so izračunani kot

$$b_{i,j} = \sum_{i=\max(j,k)}^n Tr[(\frac{\partial}{\partial q_k} \mathbf{T}_i^0)(\frac{\partial}{\partial q_j} \mathbf{T}_i^0)]. \quad (2.12)$$

Operator Tr je sled matrike, to je vsota diagonalnih členov matrike.

Matrika $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ vključuje podatke o Coriolisovih in centripetalnih silah. Njeni členi izraženi s pomočjo kot

$$c_{i,j} = \sum_{k=1}^n c_{i,j,k} \dot{q}_k, \quad (2.13)$$

kjer so

$$c_{i,j,k} = \frac{1}{2} \left(\frac{\partial b_{ij}}{\partial q_k} + \frac{\partial b_{ik}}{\partial q_j} - \frac{\partial b_{jk}}{\partial q_i} \right), \quad (2.14)$$

Christoffelovi simboli in dodatno velja še $i, j, k = 1, \dots, n$.

Vektor $\mathbf{g}(\mathbf{q})$ predstavlja navor proizveden v sklepih manipulatorja zaradi vpliva gravitacije [1]. Posamezen člen vektorja $\mathbf{g}(\mathbf{q})$ je podan kot

$$g_i = \sum_{i=j}^n (-m_i \mathbf{g}_0 (\frac{\partial}{\partial q_j} \mathbf{T}_i^0)^T \mathbf{r}_i), \quad (2.15)$$

kjer je \mathbf{g}_0 vektor gravitacijskega pospeška $\mathbf{g}_0 = [0 \ 0 \ -9,81]^T$. Vektor \mathbf{r}_i opisuje mesto težišča v segmentu, m_i pa opisuje maso segmenta.

Vektor \mathbf{F}_f opisuje vpliv trenja. Vektor zajema tako Coulombovo trenje kot tudi viskozno trenje. Posamezni členi vektorja, f_i so prispevki trenja v i -tem sklepu. Podrobneje bo ta prispevek obravnavan v poglavju 4.3.

Vektor \mathbf{h}_o vsebuje sile in navore, ki delujejo na vrh robota, $\mathbf{h}_o = [f_x \ f_y \ f_x \ m_x \ m_y \ m_z]$, kjer prvi trije členi vektorja opisujejo sile, drugi trije pa navore.

2.3 Inverzna kinematika

Nalogo, ki jo opravlja robot tipično opišemo s časovnim potekom vektorja \mathbf{x} , kot je podan v enačbi 2.7. Da bi lahko izvedli želeno gibanje, je potrebno poiskati ustrezne vrednosti notranjih koordinat \mathbf{q} , kar predstavimo z enačbo

$$\mathbf{q} = \mathbf{k}^{-1}(\mathbf{x}), \quad (2.16)$$

kjer je \mathbf{k}^{-1} inverz funkcije \mathbf{k} . Ta funkcija predstavlja inverzno kinematiko. Čeprav je bila rešitev enačbe 2.7 enolična, pa enačba 2.16 nima vedno enolične rešitve. Še več, rešitev obstaja le, če \mathbf{x} leži v delovnem prostoru robota.

Pri vodenju robotov večinoma ne rešujemo inverzne dinamike direktno iz enačbe 2.16 ampak preko hitrosti, torej iz enačbe 2.8. Če predpostavimo, da je dimenzionalnost prostora naloge m in če velja, da je $m = n$, potem lahko določimo hitrost v sklepih iz relacije

$$\dot{\mathbf{q}} = \mathbf{J}^{-1}\dot{\mathbf{x}}, \quad (2.17)$$

kjer je \mathbf{J}^{-1} inverz Jacobijeve matrike.

V primeru, da pa ima robot več stopenj prostosti, kot jih je potrebno za izvedbo naloge, torej če je $n > m$, potem inverz \mathbf{J}^{-1} ne obstaja in je potrebno poiskati rešitev inverzne kinematike drugače. Enačbo 2.8 lahko invertiramo z uporabo naslednje zveze

$$\dot{\mathbf{q}} = \mathbf{J}^*\dot{\mathbf{p}} + \mathbf{N}\dot{\mathbf{q}}, \quad (2.18)$$

kjer je \mathbf{J}^* generaliziran inverz matrike \mathbf{J} , matrika \mathbf{N} pa predstavlja projekcijo v ničelni prostor matrike \mathbf{J} ,

$$\mathbf{N} = (\mathbf{I} - \mathbf{J}^*\mathbf{J}). \quad (2.19)$$

Prvi člen enačbe 2.18 predstavlja partikularno rešitev in zadosti osnovni nalogi, zagotovi, da je vrh robota v \mathbf{x} . Drugi člen enačbe 2.18 pa predstavlja homogeno rešitev in omogoča rekonfiguracijo robotskega mehanizma brez, da se

spremeni pozicija vrha robota \mathbf{x} . Zaradi tega se hitrost \dot{q}_n lahko uporabi za realizacijo dodatnih nalog nižje prioritete.

Kot generaliziran inverz matrike \mathbf{J}^* se pogosto uporablja Moore-Penroseov pseudoinverz

$$\mathbf{J}^+ = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T)^{-1}. \quad (2.20)$$

ali uteženi Moore-Penroseov pseudoinverz

$$\mathbf{J}^* = \mathbf{J}^\# = \mathbf{W}^{-1}\mathbf{J}^T(\mathbf{J}\mathbf{W}^{-1}\mathbf{J}^T)^{-1}, \quad (2.21)$$

kjer je \mathbf{W} utežnostna matrika.

Z odvajanjem enačbe 2.18 dobimo relacijo med pospeški v sklepih in pospeških na vrhu robota.

$$\ddot{\mathbf{q}} = \mathbf{J}^*(\ddot{\mathbf{p}} + \dot{\mathbf{J}}\dot{\mathbf{q}}) + \mathbf{N}\ddot{\mathbf{q}}_N, \quad (2.22)$$

kjer je $\ddot{\mathbf{p}}$ pospešek vrha robota, pospešek $\ddot{\mathbf{q}}_N$ pa poljubni pospešek v notranjih koordinatah, ki se uporabi za realizacijo dodatnih nalog in ne vpliva na pospešek gibanja vrha robota.

3 ARCNET - UDP strežnik za komunikacijo s krmilnikom robota ter senzorjem sile

V prejšnjem poglavju je bil opisan kinematični in dinamičen model robota PA-10. V tem poglavju pa bo predstavljen krmilnik motorjev v sklepih in program, ki je nastal za komunikacijo s krmilnikom.

V sklepih robota PA-10 so servo motorji, katerih krmiljenje se izvaja preko močnostnega krmilnika. Krmilnik omogoča vodenje servo motorjev preko referenčne hitrosti ali referenčnega navora. Komunikacija s tem krmilnikom poteka preko komunikacijskega vmesnika ARCNET, ki omogoča nastavitve močnostnih ojačevalnikov v krmilniku, izbiro načina krmiljenja in seveda pošiljanja zelenih hitrosti ali navorov v sklepih. Preko te povezave se pošiljajo nazaj tudi informacije o stanju robota in krmilnika (pozicija, navori, statusi, ...).

Za samo vodenja robota pa se uporablja dodaten vmesnik, t. i. MHI Controller, ki omogoča krmiljenje robota na različne načine. Na žalost pa ima ta vmesnik kar precej omejitev, ki onemogočajo napredno vodenje robota. Omejujoče so predvsem naslednje stvari: izvaja se lahko le vodenje po hitrosti, ki navzgor dokaj strogo omejena in največja frekvenca vzorčenja je 100 Hz.

Te omejitve v MHI vmesniku omejujejo kvalitetno vodenje z upoštevanjem sil in kompenzacijo dinamičnih vplivov. Ker pa je zasnova krmilnika dokaj odprta in omogoča komunikacijo preko ARCNET omrežja dokaj dobro, smo razvili lasten

krmilni sistem na višjem nivoju, ki bo omogočal implementacijo sodobnih načinov vodenja, tako s hitrostmi kot z navornim načinom vodenja.

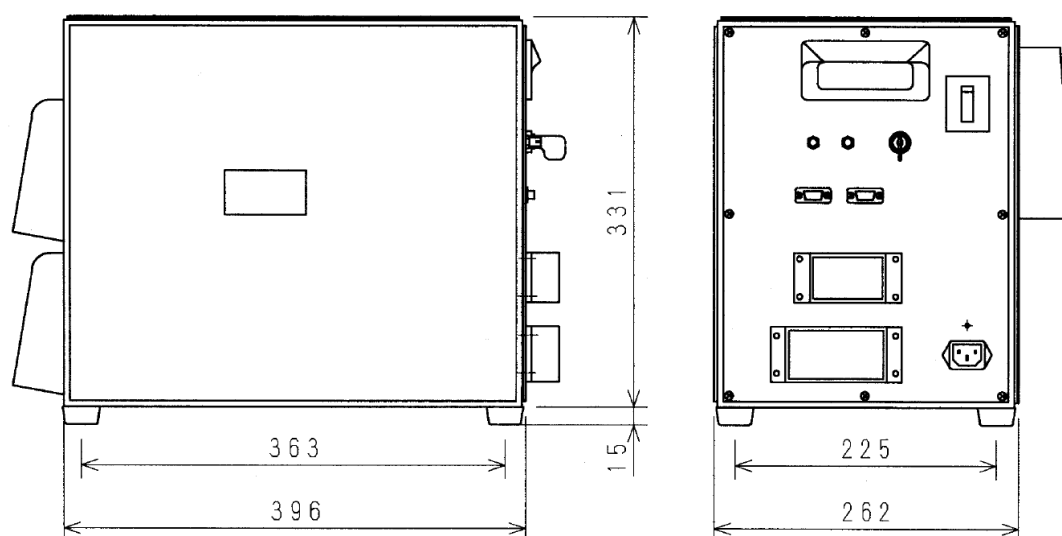
Pred začetkom izdelave lastnega programskega orodja se je definiralo dve zahtevi, ki jih mora program izpolnjevati. Prva je bila, da program deluje kar se le da hitro in kar se le da v realnem času. To izhaja iz vidika stabilnosti vodenja. Avtor v [10] namreč navaja, da vzorčni čas močno vpliva na stabilnost vodenja robota v kontaktu z okolico. Večja kot je vzorčna frekvenca večja je lahko togost okolice s katero je robot v kontaktu. Druga zahteva je bila enostavna uporaba. Program bi moral uporabniku omogočati razvijanje visokonivojskih algoritmov vodenja brez poznavanja detajlov delovanja ARCNET omrežja in krmilnika robota. Uporabnik mora imeti možnost izbire programskega okolja za realizacijo vodenja. Dodatno se je kasneje pojavila še želja, da bi programska oprema omogočala tudi posredovanje meritev izvedenih na senzorju sile JR3.

Naredili smo program, ki je zadostoval vsem navedenim kriterijem in hkrati tudi dopušča možnost nadgrajevanj. Nastali program deluje kot ARCNET-Ethernet posrednik in deluje na osebni računalniku, ki ima operacijski sistem Linux.

3.1 Krmilnik servo motorjev robota

V prvem poglavju je bilo zapisano, da robota PA-10 med drugim odlikuje dobra prenosnost. To velja tako za samo robotsko roko kot tudi za krmilnik servo motorjev. Krmilnik z ohišjem ima dimenzije $262 \times 331 \times 396$ mm in maso 22 Kg. Slika (slika 3.1) prikazuje tehnično risbo krmilnika.

V ohišju servo krmilnika so vgrajeni štirje krmilniki servo motorjev. Trije krmilijo po dva motorja, eden pa le enega. Krmilniki omogočajo vodenje motorjev na dva načina, navorno in hitrostno. Različna vodenja sta v krmilnikih drugače realizirana. Navorno vodenje je realizirano z analognim P regulatorjem toka. To izvira iz modela električnega motorja, ki pravi, da je navor na gredi



Slika 3.1: Tehnična risba ohišja krmilnika servo motorjev. Povzeto po [2].

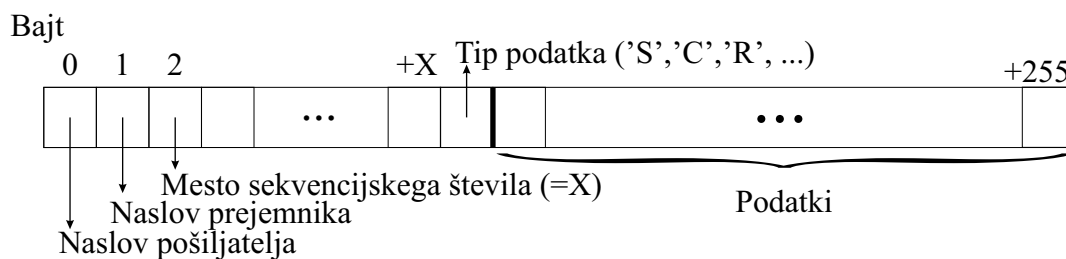
proporcionalen toku s katerim motor napajamo [11]. Hitrostna regulacija pa je realizirana z digitalnim PI regulatorjem s frekvenco približno 1500 Hz. Kot povedano, so prenosi realizirani s harmoničnimi gonili. Prestavno razmerje je 1:50. Na strani reduktorja je 14 - bitni enkoder, kar pomeni, da je resolucija merjenja kota približno $0.4 \cdot 10^{-3}$ stopinj. Krmilnik ima tudi vgrajen varnostni sistem, ki preprečuje, da bi se sklepi preveč zasukali in ob morebitnem pretiranem zasuku vklopi zavoro in tako prepreči, da bi šel izven svojih mehanskih omejitev.

Krmilnik vsebuje dva pomnilnika. Delovni pomnilnik (RAM) in nastavitveni pomnilnik (EEPROM). V EEPROM tabeli so zapisani parametri za nastavitve in vodenje servo krmilnikov. Ob zagonu krmilnika se parametri naložijo iz EEPROM tabele v RAM. Med naloženimi parametri so tudi ojačanje proporcionalnega ter integracijskega dela regulatorja, omejitve posameznih stopenj, razmerje prenosa zobnikov, itd.


3.2 ARCNET vmesnik

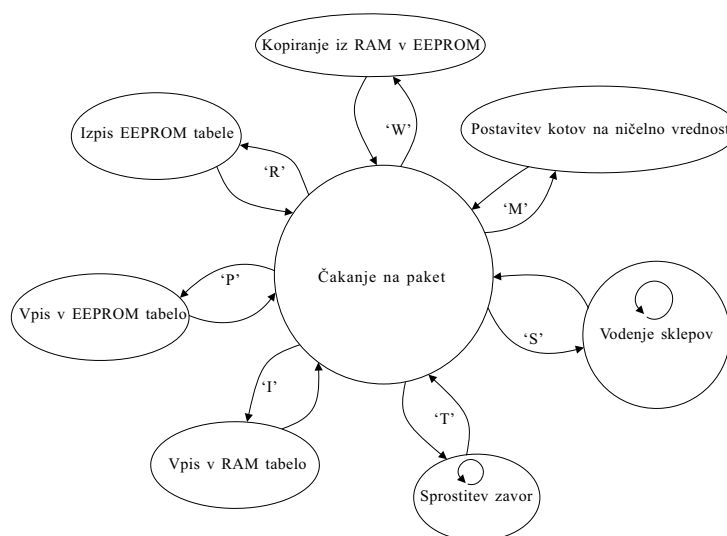
Referenčne navore in hitrosti se na krmilniku nastavlja na preko zunanjega vmesnika, ki je povezan na enako ARCNET omrežje kot servo krmilnik. Priključitev servo krmilnika na omrežje omogoča v ohišje vgrajen ARCNET modul. ARCNET je omrežje, ki vključuje podatkovni in fizični nivo po OSI modelu, komunikacija pa je serijska in paketno zastavljena. Njegova prednost pred Ethernet omrežjem je velika stopnja determinističnosti [12]. Krmilnik robota ima dva priključka za optična vodila, vhod (Rx) ter izhod (Tx). Zgornja meja hitrosti komunikacije z krmilnikom robota je 5 Mb/s [2].

Pakete, ki jih pošiljamo na krmilnik servo motorjev robota, je potrebno sestaviti tako, kot je napisano v dokumentaciji [2]. Če želimo, da bo krmilnik paket prejel, je na prvi bajt potrebno vpisati krmilnikov naslov. ARCNET modul vsebuje končni avtomat stanj, ki preklaplja med stanji glede na tip poslanih podatkov. Različni tipi podatkov so definirani z različnimi črkami, znotraj poslanega paketa. Struktura ARCNET paketa je prikazana na sliki 3.2.



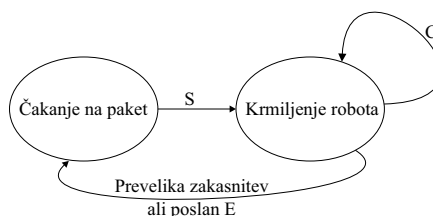
Slika 3.2: Vsebina ARCNET paketa

Preklapljanje med stanji in potrebne črke opisuje slika 3.3. Ikona  označuje stanja, v katerih se avtomat lahko zadrži dlje časa s ponavljajočim pošiljanjem paketov z enakim tipom podatkov. Iz teh stanj lahko izstopi v primeru, da nov paket ne pride pravočasno, ali pa da pride paket s črko 'E'. Ko višje nivojski vmesnik ne pošilja nobenih ukazov je ARCNET vmesnik v stanju čakanja, v katerem čaka na primerno sestavljen paket.



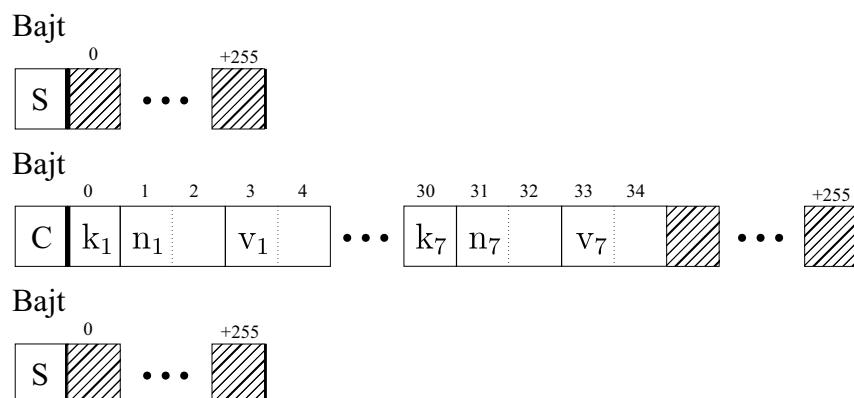
Slika 3.3: Diagram stanj ARCNET modula na krmilniku robota

Za delo z robotom je pomemben predvsem način "Vodenje sklepov", v katerem se dejansko krmili gibanje robota. V tem načinu je potrebno upoštevati predpisano maksimalno zakasnitev med zaporednima paketoma, ki je definirana v EEPROM tabeli in je tovarniško nastavljena na 300 ms. V kolikor je ta zakasnitev prekoračena, se avtomat stanj vrne v začetno stanje in je krmiljenje prekinjeno. Ta način se izbere tako, da se ARCNET vmesniku najprej pošlje ukaz, ki se začne z velikim tiskanim S, nadaljuje se s pošiljanjem paketov, ki se začnejo s veliko tiskano črko C ter zaključijo s tem, da se pošlje paket, ki se začne z velikim tiskanim E (slika 3.4). Del paketa, ki je označen z "podatki" nosi podatke, na podlagi



Slika 3.4: Krmiljenje robota s pošiljanjem ARCNET paketov.

katerih se vodi robota. Vendar pa je vsebina podatkovnega dela odvisna od črke, ki jo pošljemo.



Slika 3.5: Podatkovni del ARCNET paketa za krmiljenje robota

Paketa, ki se začeta s S in E imata podatkovni del prazen. Paket, ki se začne s C pa ima v podatkovnem delu prvih 35 bajtov zapolnjenih s podatki za vodenje robota. Slika 3.5 prikazuje vsebino podatkovnega dela omenjenih paketov. V sliki so krmilni podatki označeni z k_i , n_i in v_i . Njihov pomen pa je

- k - kontrolni bajt (velikost podatka: byte),
 - bit 0: zavora (1 - vklop, 0 - izklop),
 - bit 1: servo motor (1 - vklop, 0 - izklop),
 - bit 2: način krmiljenja (1 - navor, 0 - hitrost),
 - biti 3 - 7: se en uporabljajo,
- n - referenčni navor (velikost podatka: word) = 0.001 Nm/digit,
- v - referenčna hitrost (velikost podatka: word) = 0.0002 rad/s/digit,

Poslani referenčni navori in hitrosti gredo od vrednosti 0x0000 do 0xFFFF. Krmilnik ima zaradi varnosti nastavljene omejitve obeh veličin, zato prevelike referenčne signale poreže tako, da ustrezajo omejitvam.



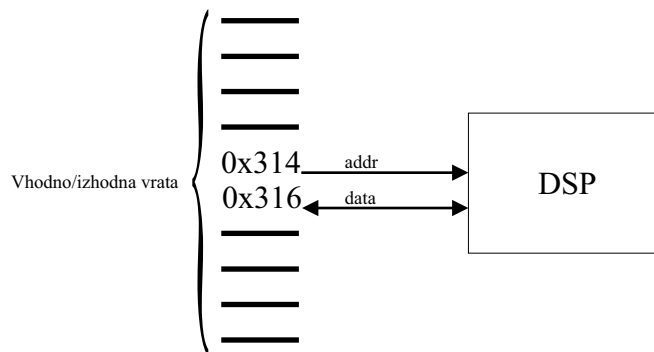
Slika 3.6: Senzor sil in navorov JR3.

3.3 Senzor sile in navorov - JR3

Za realizacijo vodenja opisanega v poglavju 4.5 smo med vrhom robota ter prijemalom pritrldilo senzor sil JR3 (slika 3.6). Senzor nam omogoča merjenje sil in navorov v x , y in z oseh. Senzor je narejen iz uporovnih lističev porazdeljenih po notranjosti senzorja, ki je v obliki križa. Iz specifikacij senzorja [13] je razvidno, da uporablja 6 uporovnih lističev. S tem, ko na senzor deluje zunanja sila ali navor, pride do deformacije materiala na katerem so nameščeni uporovni lističi in se upornost na uporovnih lističih spremeni proporcionalno s silo in togostjo senzorja. Sprememba upornosti se izmeri posredno preko spremembe napetosti na AD-pretvorniku, ki je vgrajen v sam senzor. Napetost se pretvori v silo preko množenja s kalibracijsko matriko \mathbf{K}_{calib} . Naj bo \mathbf{h}_{JR3} vektor sil, ki delujejo na senzor in naj bo \mathbf{u}_u vektor napetosti na AD-pretvorniku. Enačba za izračun sil je

$$\mathbf{h}_{JR3} = \mathbf{K}_{calib} \mathbf{u}_u \quad (3.1)$$

Senzor je priklopljen na računalnik preko 8 žilnega kabla s kontektorjem tipa RJ-45 v ISA kartico. Elektronika v samem senzorju serijsko pošilja podatke o izmerjenih silah na kartico. Kartica vsebuje vezje za digitalno obdelavo signalov (ang. *digital signal processor* - DPS). V dokumentaciji kartice [14] so opisani trije nizkoprepustni filtri z različnimi mejnimi frekvencami. Uporabili smo filter z mejno frekvenco 500 Hz.



Slika 3.7: Dostopanje do registrov na ISA kartici za digitalno obdelavo signala iz senzorja JR3.

DSP kartica na ISA vodilu zavzame dva 16-bitna naslova: osnovnega (*base address*) ter naslednjega. Osnovni se uporabi kot naslovni register, drugi pa kot podatkovni register. Preko vhodno/izhodnih vrat računalnika lahko nastavljamo vsebino teh registrov. Na ta način lahko dalje dostopamo do internih registrov DSP kartice. Na kartici so stikala preko katerih nastavimo osnovni naslov (*base address*) kartice. Če je osnovni naslov 0x314 bo naslovni register dostopen na 0x314, podatkovni pa na 0x316. Podatke na registrih DPSja preberemo tako, da na naslovni del zapišemo naslov registra. DPS nato napiše vsebino registra na podatkovni register ISA vodila. Če želimo v register DPSja nekaj zapisati, pa moramo mi zapisati podatek na podatkovni del ISA registra (slika 3.7). V sintaksi programskega jezika C bi funkciji za branje ter pisanje po registrih DSP kartice zgledalo tako:

```
#define baseAddr 0x314

int getData(int addr)
{
    outport(baseAddr, addr);
    return inport(baseAddr + 2);
}
```

```
int putData(int addr, int data)
{
    outport(baseAddr, addr);
    outport(baseAddr + 2, data);
}
```

Dokumentacija navaja, da v kolikor želi uporabnik odčitati vrednost sile ali navora F mora tako prebrati osnovno vrednost F_s na registru imenovanem (`full scale`), jo pomnožiti z vrednostjo F_f , ki je na registru izbranega filtra (`current force`). Na zadnje more to vrednost delit s konstanto K_p , da pretvori v primerne enote (N ali Nm).

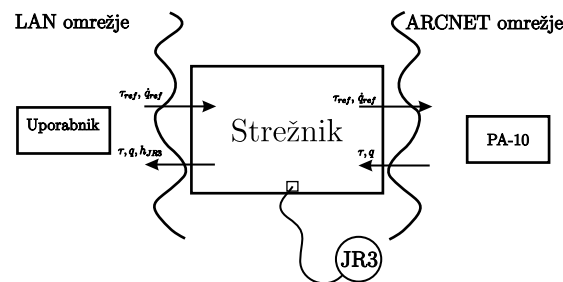
$$F = \frac{F_s F_f}{K_p} \quad (3.2)$$

Ker senzor meri silo in navor v treh smereh je potrebno opraviti izračun za vsako od izmerjenih veličin potrebno opraviti omenjeno operacijo.

3.4 Strežnik

V uvodu v poglavje smo omenili, da je proizvajalec pripravil program za krmiljenje robota, MHI Controller, vendar smo se zaradi pomanjkljivosti tega programa, avtorji odločili za izgradnjo lastnega. Definirane zahteve so bile visoka frekvenca, delovanje v realnem času ter enostavnost uporabe. Hrati pa je morabil nov program biti sposoben prebirati izmerjene sile na vrhu robota preko JR3 senzorja sil.

Ko smo se avtorji lotili izgradnje programa smo se najprej morali odločiti, na kakšni platformi bo program deloval. Zaradi enostavnosti razvijanja smo izbrali osebni računalnik, na katerega se lahko priklapi ISA kartico za senzor sil JR3 ter ARCNET mrežno kartico za komunikacijo s krmilnikom PA-10. Naslednja



Slika 3.8: Strežnik kot posrednik med ARCNET ter Ethernet omrežjem.

odločitev je bila izbira operacijskega sistema. Avtorji smo se odločali med Windows, Linux ali XPCtarget sistemi. Na koncu smo se odločili za operacijski sistem Linux, saj ga odlikujejo naslednje lastnosti:

- prosto dostopen,
- podpora za ARCNET omrežje,
- dobra podpora za dostop do izhodno/vhodnih vrat,
- dobro časovne lastnosti [15].

Nov krmilni program deluje kot posrednik med dvema različnima omrežjema (slika 3.8), ARCNET in UDP. V nadaljevanju bo opisana zgradba tega strežnika, delovanje ter uporaba. Uporabnik lahko robota PA-10 krmili s pošiljanjem UDP paketov na strežnik, ki jih ta posreduje na ARCNET omrežje.

3.4.1 Delovanje programa

Nastali program deluje kot strežnik, kar pomeni, da se odzove na klientovo zahtevo. V tem primeru je klient kakršen koli program, ki lahko pošilja UDP pakete po Ethernet omrežju. Klient mora tako formirati ustrezen UDP paket za začetek komunikacije. Paket je velik 99 bajtov. Vsebina paketa pa je opisana v tabeli 3.1.

Zap. št.	Dolžina	Opis podatka	Tip podatka
0	7	Kontrolni bajt za posamezen motor	char
6	7	Referenčni navor za posamezen motor	signed short int
20	7	Referenčna hitrost za posamezen motor	signed short int
34	62	Rezervirani / neuporabljeni bajti	char
96	1	Štartna sekvenca	char
97	1	Trenuten čas na klientu	float

Tabela 3.1: Opis prejetega UDP paketa.

Kontrolni bajt, referenčni navori ter hitrosti za posamezen motor se posredujejo naprej na krmilnik robota. Ostali bajti pa so namenjeni zagotavljanju varnosti med komunikacijo. Referenčni navori so v Nm, hitrosti pa v rad/s.

S prispelim UDP paketom strežnik pretvori dane navore in hitrosti v format, ki je naveden v dokumentaciji in opisan v 3.2. Kontrolni bajti za posamezne motorje ostanejo nespremenjeni in se posredujejo enaki kot so prispeli. Strežnik prispele podatke zapakira v ARCNET paket naveden v tabeli 3.2. Paket pošlje po ARCNET omrežju

Zap. št.	Dolžina	Opis podatka
0	1	ID pošiljatelja
1	1	ID prejemnika
2	1	Prazen / neuporabljen bajt
3	1	Mesto sekvencijskega števila
4	1	Sekvencijsko število
5	1	Tip podatka
6	1	Statusni bajt za prvi servo
7	2	Referenčni navor za prvi servo
9	2	Referenčna hitrost za prvi servo
:	:	:
36	1	Statusni bajt za sedmi servo
37	2	Referenčni navor za sedmi servo
39	2	Referenčna hitrost za sedmi servo

Tabela 3.2: Opis ARCNET paketa.

Strežnik nato počaka na odgovor servo krmilnika. Strežnik odgovor servo krmilnika najprej pretvori v želeno obliko. Hitrosti in navori v sklepih so ponovno zapisani v formatu navedenem v 3.2. Te podatke pa nato zapakira v UDP paket. Zraven doda še kontrolne bajte. UDP paket poslan klientu je opisan v tabeli 3.3.

Zap. št.	Dolžina	Opis podatka	Tip podatka
0	7	Kontrolni bajt za posamezen motor	char
7	7	Trenuten kot v sklepu	float
35	7	Trenutni proizveden navor v sklepu	signed short int
63	6	Izmerjene sile in navori na JR3	float
73 - 111	38	Rezerviran prostor	float
112	1	Čas, ki je bil porabljen za programski cikel	float
116	1	Čas, ki ga je poslal klient na bajtu 60	float

Tabela 3.3: Opis poslanega UDP paketa.

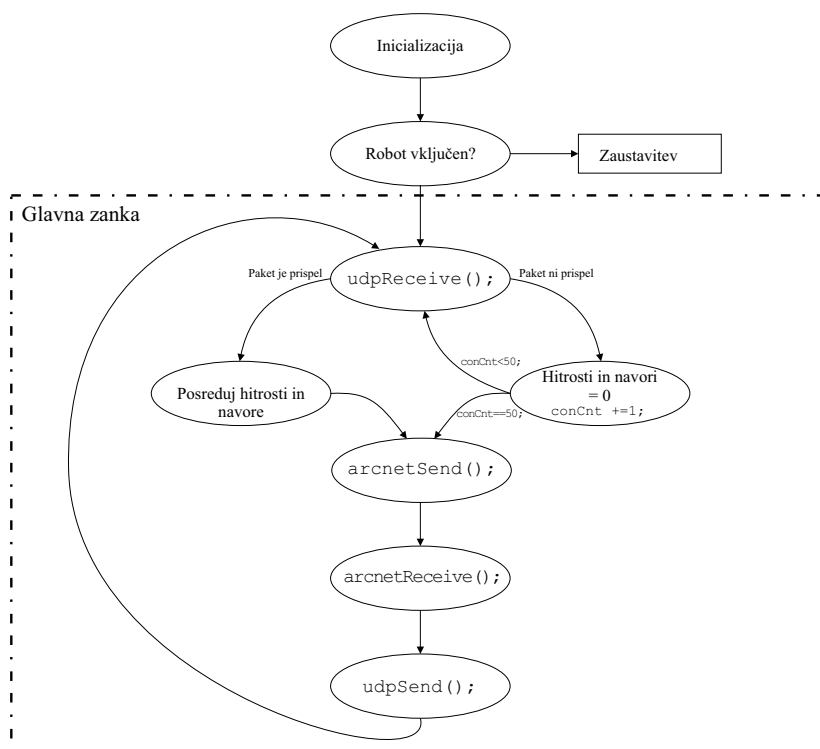
Po tistem, ko strežnik prejme in pošlje vse podatke, gre v stanje čakanja, da mine od začetka programskega cikla do konca točno 2 ms. Ohranjanje frekvence časovnega cikla je kritičnega pomena. Zato strežnik prekine komunikacijo s servo krmilnikom, v kolikor nov UDP paket ne prispe pravočasno.

Diagram delovanja strežnika prikazuje slika 3.9.

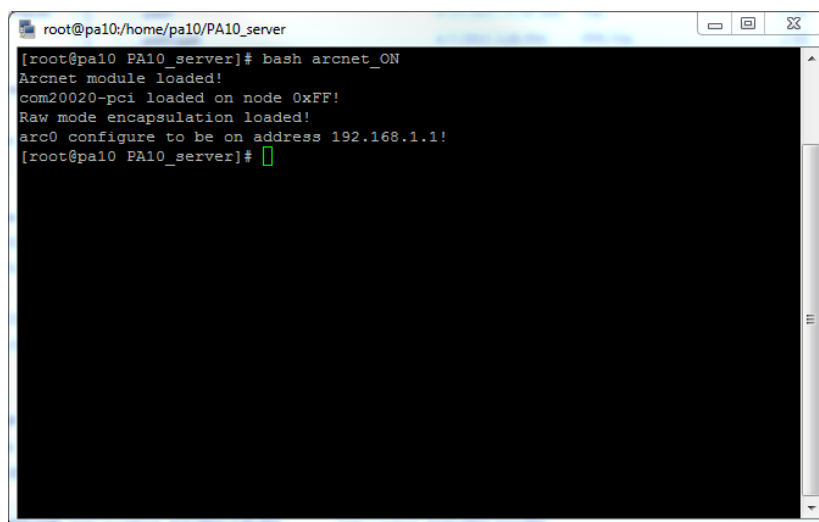
3.4.2 Uporaba programa

Kot povedano, je bil program narejen na Linux operacijskemu sistemu. Prednost tega sistema je tudi uporaba SSH (*Secure Shell*) protokola za oddaljen dostop. Uporabnik lahko na ta način zažene strežnik brez, da je fizično ob računalniku.

Pred zagonom strežnika je potrebno najprej naložiti ARCNET modul v Linux jedro. To napravi uporabnik z zagonom skripte `arcnet_ON`, ki smo jo pripravili avtorji (slika 3.10). Sledi zagon programa z ukazom `./udparcnet-server`. Gre za standardno Linux sintakso za zagon programov pri čemer `./` pomeni zagon, `udparcnet-server` pa ime programa. V kolikor želi uporabnik strežnik zagnati tako, da pričakuje ter pošilja UDP podatke na druga vrata (ang. *port*) lahko



Slika 3.9: Diagram poteka programa.

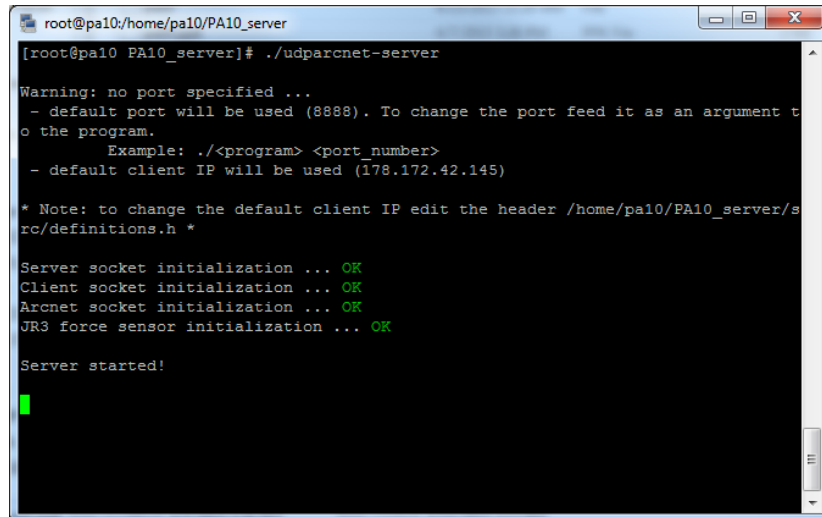


Slika 3.10: Nalaganje ARCNET modula v jedro Linux.

to doda kot argument programu. Če želi izbrati vrata 1337 to naredi tako: `./udparcnet-server 1337`. Primer zaganjanja strežnika je na sliki 3.11. Od

tukaj dalje strežnik čaka na UDP pakete s strani klienta.

Vidimo lahko, da program javi, če so bile funkcije v inicializaciji uspešne in če je senzor sil JR3 povezan. V kolikor senzorja ni, strežnik še vedno deluje, vendar so vrednosti poslanih sil enake 0.



```
root@pa10:/home/pa10/PA10_server
[root@pa10 PA10_server]# ./udparcnet-server

Warning: no port specified ...
- default port will be used (8888). To change the port feed it as an argument to
the program.
  Example: ./<program> <port_number>
- default client IP will be used (178.172.42.145)

* Note: to change the default client IP edit the header /home/pa10/PA10_server/s
rc/definitions.h *
```

```
Server socket initialization ... OK
Client socket initialization ... OK
Arcnet socket initialization ... OK
JR3 force sensor initialization ... OK

Server started!
```

Slika 3.11: Zaganjanje strežnika na Linux operacijskem sistemu.

3.4.3 Varnostni ukrepi

V tabelah, ki opisujejo poslane in prejete UDP pakete je mogoče razbrat, da so nekateri bajti namenjeni ohranjanju komunikacije oz. preverjanju ali sploh še poteka. Eden od takih ukrepov je, da v kolikor nov UDP ali ARCNET paket ne prispe v nekem vnaprej definiranem času, program robota zaustavi nato pa gre v čakanje novega paketa. Ko je program v čakanju preverja prispele UDP pakete, v kolikor imajo štartno sekvenco. Ta skrbi za to, da se program začne izvajati s prvim prispelim UDP paketom. Na ta način se poskrbi, da strežnik nebi šele petega UDP paketa prebral in na to začel z delovanjem.

Naslednji varnostni ukrep preverja stanje varnostnega gumba (*Emergency stop*). V kolikor se ta pritisne, se na servo krmilniku že sam po sebi sproži ukaz za ustavitev motorjev ter vklop zavor. Na strežniku pa se to pozna tako,

da se program zaustavi. V kolikor želimo nadaljevati z vodenjem, je potrebno program ponovno zagnat.

4 Vodenje robota PA-10

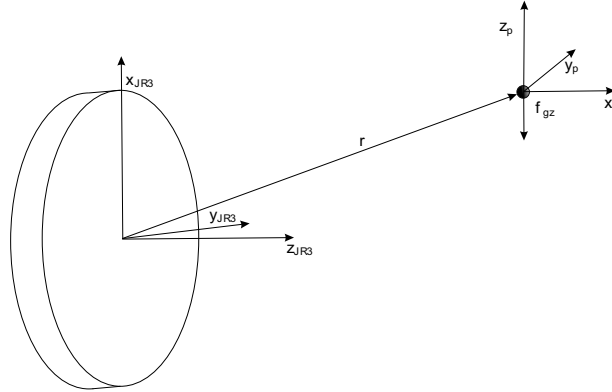
V prejšnjih poglavjih je bilo govora o pošiljanju referenčnih hitrosti in navorov na krmilnik servo motorjev robota PA-10. V poglavju, ki sledi pa bo govora o določanju teh krmilnih veličin v namene vodenja robota po prostoru. V tem poglavju bo natančneje opisano vodenje robota v zunanjih koordinatah. Opisan bo postopek kompenzacije vpliva težnosti prijemala na senzor sile JR3. Na koncu pa bosta opisana dva načina za krmiljenje robota v stiku z okolico, admitančno vodenje ter vodenje preko inverzne dinamike.

4.1 Kompenzacija mase prijemala ter merilnega odmika na senzorju sile JR3

Delovanje senzorja sile JR3 je bilo opisano v prejšnjih poglavjih. Senzor je pritrjen na vrh robota in meri navore ter sile, ki delujejo nanj v njegovem koordinatnem sistemu. Homogeno transformacijsko matriko smo definirali v poglavju 2. Sile in navore na vrhu robota lahko preslikamo v koordinatni sistem baze preko matrike homogene transformacije

$$\mathbf{h}_b = \mathbf{T}_7^0 \mathbf{h}_e \quad (4.1)$$

Robot brez prijemala v praksi ni pretirano uporaben saj lahko opravlja le malo uporabnih nalog, npr. premikanje objektov s porivanjem. V kolikor pa želimo, da bo robot lahko izvajal tudi druge naloge, je potrebno nanj pritrditi primerno



Slika 4.1: Koordinatni sistem senzorja sile ter koordinatni sistem težišča prijemala

prijemalo. Vendar pa imamo na vrhu robota najprej senzor, kar pomeni, da omenjeno prijemalo pritrdimo na senzor. Ker ima prijemalo svoje maso in težišče, bo na senzor delovala neka sila in navor kot posledica delovanja gravitacije. Kar pomeni, da bomo kljub temu, da senzor ni v kontaktu z okolico, merili sile. To pa ni dobrodošlo, saj želimo meriti le sile, ki se so posledica kontakta prijemala z okolico. Dodatno je potrebno še upoštevati merilni odmik, ki se pojavi kot posledica lezenja uporovnih mostičkov.

Sile (\mathbf{f}_s) in napore (\mathbf{m}_s), ki jih meri senzor, lahko razdelimo na tri komponente

$$\begin{aligned}\mathbf{f}_s &= \mathbf{f}_{env} + \mathbf{f}_{grav} + \mathbf{f}_{off} \\ \boldsymbol{\tau}_s &= \boldsymbol{\tau}_{env} + \boldsymbol{\tau}_{grav} + \boldsymbol{\tau}_{off},\end{aligned}\tag{4.2}$$

kjer je $(.)_{env}$ zunanja sila in navor, ki deluje na senzor, $(.)_{grav}$ vpliv mase prijemala, člen $(.)_{off}$ pa predstavlja merilni odmik, ki je posledica lezenja senzorja. Želja je, določiti $(.)_{grav}$ in $(.)_{off}$ tako, da ju lahko kompenziramo in določimo $(.)_{env}$. Ko vrh robota ni v kontaktu z okolico velja $\mathbf{f}_{env} = 0$ in $\boldsymbol{\tau}_{env} = 0$ oziroma:

$$\begin{aligned}\mathbf{f}_s &= \mathbf{f}_{grav} + \mathbf{f}_{off} \\ \boldsymbol{\tau}_s &= \boldsymbol{\tau}_{grav} + \boldsymbol{\tau}_{off}.\end{aligned}\tag{4.3}$$

Sila \mathbf{f}_{grav} je odvisna od mase (m) prijemala ter orientacije vrha robota. Koordinatni sistem prijemala naj bo zaradi poenostavitve vzporeden koordinatnemu

sistemu vrha robota. Orientacijo senzorja zapišemo s poznavanjem kinematičnega modela in bo v nadaljevanju označena z \mathbf{R} . Tako je mogoče določiti prispevek

$$\mathbf{f}_{grav} = \mathbf{R} \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} = \mathbf{R}\mathbf{f}_{gz}. \quad (4.4)$$

Ker ima \mathbf{f}_{gz} le en člen različen od nič, lahko iz matrike \mathbf{R} vzamemo le tretji stolpec in zapišemo

$$\mathbf{f}_{grav} = \begin{bmatrix} r_{31} \\ r_{32} \\ r_{33} \end{bmatrix} mg = \mathbf{R}_3 mg. \quad (4.5)$$

Avtorji v [16] pokažejo, da je mogoče izračunati \mathbf{f}_{grav} in \mathbf{f}_{off} na podlagi najmanj dveh meritev ($\mathbf{f}_{meas}^{(1)}$ in $\mathbf{f}_{meas}^{(2)}$). Zapišemo sistem enačb v matrični obliki

$$\begin{bmatrix} \mathbf{f}_{mes}^{(1)} \\ \mathbf{f}_{mes}^{(2)} \end{bmatrix} = \begin{bmatrix} \mathbf{R}^{(1)} & \mathbf{I} \\ \mathbf{R}^{(2)} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{f}_{gz} \\ \mathbf{f}_{off} \end{bmatrix}. \quad (4.6)$$

Upošteva je 4.5 se nam 4.6 poenostavi na

$$\begin{bmatrix} \mathbf{f}_{mes}^{(1)} \\ \mathbf{f}_{mes}^{(2)} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_3^{(1)} & \mathbf{I} \\ \mathbf{R}_3^{(2)} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{f}_{gz} \\ \mathbf{f}_{off} \end{bmatrix} = \mathbf{A} \begin{bmatrix} mg \\ \mathbf{f}_{off} \end{bmatrix}. \quad (4.7)$$

Izrazimo \mathbf{f}_{grav} in \mathbf{f}_{off}

$$\begin{bmatrix} mg \\ \mathbf{f}_{off} \end{bmatrix} = \left(\mathbf{A}^T \mathbf{A} \right)^{-1} \mathbf{A}^T \begin{bmatrix} \mathbf{f}_{meas}^{(1)} \\ \mathbf{f}_{meas}^{(2)} \end{bmatrix}. \quad (4.8)$$

Enačba bo veljala le, če bo matrika \mathbf{A} polnega ranga oziroma, če bosta meritvi linearno neodvisni.

Za navor pa je potrebno poznati tudi translacijski vektor od težišča do koordinatnega sistema senzorja, v nadaljevanju označen z \mathbf{r} . Prispevek navora zaradi

vpliva gravitacije je tako

$$\boldsymbol{\tau}_{grav} = \mathbf{r} \times \mathbf{f}_{grav}. \quad (4.9)$$

Dodatno bomo tukaj definirali še matričen zapis vektorja \mathbf{f}_{grav} kot

$$S(\mathbf{f}_{grav}) = \begin{bmatrix} 0 & -\mathbf{f}_{grav_z} & \mathbf{f}_{grav_y} \\ \mathbf{f}_{grav_z} & 0 & -\mathbf{f}_{grav_x} \\ -\mathbf{f}_{grav_y} & \mathbf{f}_{grav_x} & 0 \end{bmatrix}. \quad (4.10)$$

Iz treh različnih meritev navorov ($\boldsymbol{\tau}_{meas}^{(1)}$, $\boldsymbol{\tau}_{meas}^{(2)}$ in $\boldsymbol{\tau}_{meas}^{(3)}$) pri treh različnih orientacijah ($\mathbf{R}^{(1)}$, $\mathbf{R}_{meas}^{(2)}$ in $\mathbf{R}_{meas}^{(3)}$) zapišemo sistem enačb

$$\begin{bmatrix} \boldsymbol{\tau}_{meas}^{(1)} \\ \boldsymbol{\tau}_{meas}^{(2)} \\ \boldsymbol{\tau}_{meas}^{(3)} \end{bmatrix} = \begin{bmatrix} S(\mathbf{f}_{grav})^{(1)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(2)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(3)} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{r} \\ \boldsymbol{\tau}_{off} \end{bmatrix}. \quad (4.11)$$

Enačbo obrnemo in izrazimo \mathbf{r} in $\boldsymbol{\tau}_{grav}$

$$\begin{bmatrix} \mathbf{r} \\ \boldsymbol{\tau}_{off} \end{bmatrix} = \left(\begin{bmatrix} S(\mathbf{f}_{grav})^{(1)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(2)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(3)} & \mathbf{I} \end{bmatrix}^T \begin{bmatrix} S(\mathbf{f}_{grav})^{(1)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(2)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(3)} & \mathbf{I} \end{bmatrix} \right)^{-1} \begin{bmatrix} S(\mathbf{f}_{grav})^{(1)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(2)} & \mathbf{I} \\ S(\mathbf{f}_{grav})^{(3)} & \mathbf{I} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{\tau}_{meas}^{(1)} \\ \boldsymbol{\tau}_{meas}^{(2)} \\ \boldsymbol{\tau}_{meas}^{(3)} \end{bmatrix}. \quad (4.12)$$

Bistveni trije podatki, ki smo jih s temi izračuni izpostavili so merilni odmik, vpliv gravitacije ter težišče prijemala. Prednost tega postopka je, da imamo orientacijo senzorja v vseh merilnih pozicijah natančno določeno zaradi poznavanja kinematičnega modela robota. Hkrati pa lahko ta postopek popolnoma avtomatiziramo, kar smo tudi naredili. Postopek je opisan v dodatku ???. Masa ter težišče prijemala sta parametra, ki se jih identificira in sta časovno neodvisna. Zaradi efekta lezenja uporovnih lističev pa je merilni odmik časovno odvisen in ga je potrebno pogosteje izmeriti.

V nadaljevanju bo sila ter navor kot posledica interakcije z okolico zapisana

v vektorju \mathbf{h}_e kot

$$\mathbf{h}_e = \begin{bmatrix} \mathbf{f}_s - \hat{\mathbf{f}}_{grav} - \hat{\mathbf{f}}_{off} \\ \boldsymbol{\tau}_s - \hat{\boldsymbol{\tau}}_{grav} - \hat{\boldsymbol{\tau}}_{off} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_{env} \\ \boldsymbol{\tau}_{env} \end{bmatrix}, \quad (4.13)$$

kjer so $(\cdot)_s$ odčitki senzorja in $\widehat{(\cdot)}$ izračunani prispevki.

4.2 Vodenje preko inverzne kinematike

Za vodenje robota preko hitrosti v sklepih se uporablja PI regulator v krmilniku, ki poskrbi, da se vsak sklep vrti s hitrostjo čimbolj podobno referenčni. Iz tega vidika je prednost ta, da ne glede na konfiguracijo ter obremenitev robota, bo regulator poskrbel konvergenco napake proti nič. V nadaljevanju bo opisano vodenje motorjev robota v želeno lego oziroma vodenje robota v notranjih koordinatah. Nato pa se bo iz tega še izpeljalo vodenje v zunanjih koordinatah.

4.2.1 Vodenje v notranjih koordinatah

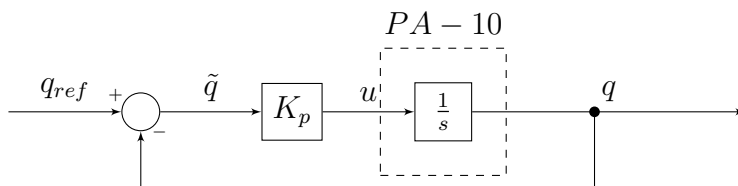
Krmilnik servo motorjev v vsakem danem trenutku vrača trenuten kot v sklepu. Lastna frekvenca zaprtozančnega sistema je dovolj visoka, da lahko servo sistem za eno stopnjo robota aproksimiramo kot integrator $\frac{1}{s}$. Naj bo referenčni kot v sklepih označen kot \mathbf{q}_{ref} in naj bo dejanski kot označen kot \mathbf{q} . Razlika med njima je napaka v kotu

$$\tilde{\mathbf{q}} = \mathbf{q}_{ref} - \mathbf{q}. \quad (4.14)$$

S preprostim P regulatorjem lahko kote v sklepu robota vodimo s povratno zanko tako, kot je prikazano na shemi 4.2

$$\mathbf{u} = \mathbf{K}_p \tilde{\mathbf{q}}. \quad (4.15)$$

Ojačanje regulatorja je označeno s \mathbf{K}_p in je diagonalna matrika saj ima vsak sklep svoje ojačanje neodvisno od ostalih. S povečevanjem ojačanja regulatorja \mathbf{K} vplivamo na dinamiko odziva [17].



Slika 4.2: Povratna zanka za krmiljenje kota v sklepu

Iz blokovnega diagrama lahko zapišemo prenosno funkcijo sistema $P(s)$ in analiziramo stabilnost:

$$P(s) = \frac{K \frac{1}{s}}{1 + K \frac{1}{s}} = \frac{K}{s + K} \quad (4.16)$$

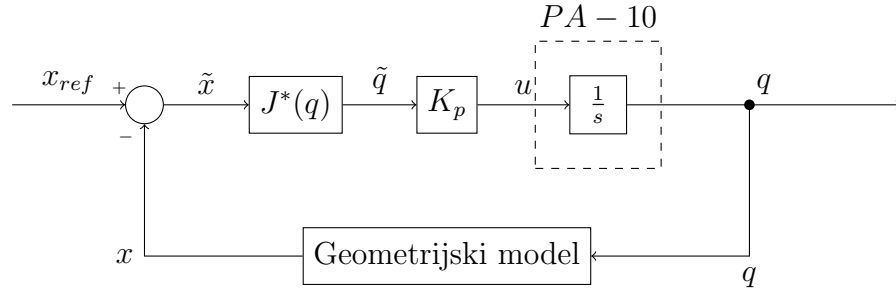
Pogoj stabilnosti je, da vsi poli sistema ležijo na desni strani s ravnine. V tem primeru je tako pogoj le, da je $K > 0$. Iz tega bi sledilo, da lahko ojačanje K povečujemo v neskončnost. Vendar pa je potrebno tukaj poudariti, da pri previsokih ojačanjih lahko pride do nestabilnosti, ki pa jih z obravnavanjem robota kot integratorja, ne moremo predvideti. Zato je smiselno, da se z ojačanji ne pretirava. Lahko pa v namene stabilnosti vnesemo še signal dušenja $\mathbf{K}_d \dot{\mathbf{q}}$. Matrika \mathbf{K}_d je tako kot \mathbf{K}_p diagonalna. Regulirni signal sedaj zapišemo kot

$$\mathbf{u} = \mathbf{K}_p \tilde{\mathbf{q}} - \mathbf{K}_d \dot{\mathbf{q}}. \quad (4.17)$$

4.2.2 Vodenje v zunanjih koordinatah

Kinematični model robota in problem inverzne kinematike je bil opisan v 2.1. Predstavljen je bil problem inverzne kinematike za redundantni sistem ter pseudoinverz Jacobijeve matrike. Zato bo v nadaljevanju uporabljen generaliziran pseudoinverz Jacobijeve matrike.

Naj bo \mathbf{x}_{ref} vektor referenčne lege vrha robota in naj bo \mathbf{x} trenutna lega vrha robota. Napaka je tako



Slika 4.3: Bločna shema krmiljenja referenčne lege vrha robota

$$\tilde{\mathbf{x}} = \mathbf{x}_{ref} - \mathbf{x} \quad (4.18)$$

Enačba direktne kinematike nam govori o odnosu med hitrostjo vrha robota ter hitrostjo v sklepih robota

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \quad (4.19)$$

To enačbo je mogoče razumeti tudi kot zvezo med spremembo položaja vrha ter sklepov robota [18]. Zato je mogoče to enačbo uporabiti za zapis relacije med napako vrha ter potrebnim odmikov v sklepih preko preko enačbe za inverzno kinematiko

$$\tilde{\mathbf{q}} = \mathbf{J}^*(\mathbf{q})\tilde{\mathbf{x}} \quad (4.20)$$

Z vstavitvijo enačbe 4.20 v 4.15 dobimo P regulator lege vrha robota.

$$\mathbf{u} = \mathbf{K}_p \mathbf{J}^*(\mathbf{q})(\mathbf{x}_{ref} - \mathbf{x}). \quad (4.21)$$

Avtor v [18] navaja, da se mehanski sistem, ki je voden na ta način obnaša kot mehanski sistem z n - dimenzionalno vzmetjo v notranjih koordinatah. Togost omenjene vzmeti pa določa ojačanje \mathbf{K}_p .

4.3 Krmiljenje z inverzno dinamiko

Navor proizveden v motorjih robota je linearno odvisen od toka, ki gre v motor. Krmilnik servo motorjev robota PA-10 uporablja P regulator toka za krmiljenje motorjev v navornem načinu. Problem vodenja robota v zunanjih koordinatah se tako prevede v problem določevanja primerne navora za opravljanje naloge. Navore lahko izračunamo s pomočjo inverznega dinamičnega modela, kar zahteva dobro poznavanje dinamičnih parametrov. Dinamičen model je bil opisan v 2.2.

Vodenje robota na v navornem načinu ima to prednost, da lahko podajnost celotnega mehanizma spreminjamo. Tako robot ni nujno podajen le na vrhu ob uporabi senzorja sil, temveč po celotni konstrukciji. Tak način vodenja omogoča dobro interakcijo z okoljem ter človekom.

V nadaljevanju bomo opisali vodenje po zunanjih in po notranjih koordinatah. Enačbo 2.11 zapišemo drugače in sicer brez upoštevanja zunanjih sil na vrh manipulatorja, $\mathbf{h}_o = 0$. Dodatno bomo vpeljali še novo veličino

$$\mathbf{n}(q, \dot{q}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{F}_f(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{g}(\mathbf{q}), \quad (4.22)$$

ki združuje vse dinamične veličine razen vpliva vztrajnosti. Sedaj zapišemo dinamičen model robota v skrajšani obliki

$$\boldsymbol{\tau}(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q}) = \mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) \quad (4.23)$$

4.3.1 Vodenje v notranjih koordinatah

Preden se lotimo vodenja v zunanjih koordinatah je najprej potrebno definirati vodenje robota po notranjih koordinatah. Vhodna veličina je navor zato bomo zapisali

$$\boldsymbol{\tau}(q, \dot{q}, \ddot{q}) = \mathbf{u}. \quad (4.24)$$

Predpostavimo, da poznamo približen dinamični model robota, t.j. matriko $\hat{\mathbf{B}}$ in $\hat{\mathbf{n}}$. Navor za sledenje referenčnemu pospešku $\ddot{\mathbf{q}}_{ref}$ sedaj izračunamo z

$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})\ddot{\mathbf{q}}_{ref} + \hat{\mathbf{n}}(q, \dot{q}). \quad (4.25)$$

Vstavimo 4.25 in 4.23 v 4.24 in dobimo

$$\hat{\mathbf{B}}(\mathbf{q})\ddot{\mathbf{q}}_{ref} + \hat{\mathbf{n}}(q, \dot{q}) = \mathbf{B}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}). \quad (4.26)$$

Če predpostavimo, da poznamo natančen model, velja $\hat{\mathbf{B}} = \mathbf{B}$ in $\hat{\mathbf{n}} = \mathbf{n}$ in lahko zapišemo

$$\ddot{\mathbf{q}}_{ref} = \ddot{\mathbf{q}}. \quad (4.27)$$

Enačba nam pravi, da v kolikor poznamo natančen dinamičen model, bo vodenje 4.25 zagotovilo želeno gibanje robota [18]. Ker pa dinamičnih parametrov robota v praksi nikoli ne poznamo dovolj natančno, 4.27 ne velja in se pojavi napaka v poziciji sklepov. Zato je smiselno krmilnemu signalu dodati še regulacijo hitrosti ter pozicije in definiramo krmilne pospeške $\ddot{\mathbf{q}}_u$ kot

$$\ddot{\mathbf{q}}_u = \ddot{\mathbf{q}}_{ref} + \mathbf{K}_p(q_{ref} - q) + \mathbf{K}_d(\dot{\mathbf{q}} - \dot{\mathbf{q}}). \quad (4.28)$$

in jih vstavimo v enačbo 4.25 namesto $\ddot{\mathbf{q}}_{ref}$. Končna enačba za vodenje preko inverzne dinamike je tako

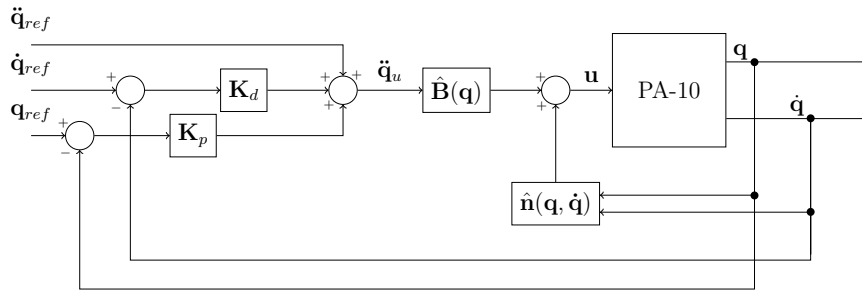
$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})(\ddot{\mathbf{q}}_{ref} + \mathbf{K}_p(q_{ref} - q) + \mathbf{K}_d(\dot{\mathbf{q}} - \dot{\mathbf{q}})) + \hat{\mathbf{n}}(q, \dot{q}). \quad (4.29)$$

Če sedaj primerjamo $\ddot{\mathbf{q}}_u$ z dejanskim pospeškom v sklepih lahko zapišemo dinamiko signala napake, kot

$$\ddot{\tilde{\mathbf{q}}} + \mathbf{K}_d\dot{\tilde{\mathbf{q}}} + \mathbf{K}_p\tilde{\mathbf{q}} = 0. \quad (4.30)$$

Enačbo lahko prevedemo v Laplace-ov prostor

$$s^2 + \mathbf{K}_d s + \mathbf{K}_p = 0 \quad (4.31)$$



Slika 4.4: Bločna shema vodenja po notranjih koordinatah preko inverzne dinamike.

in opazimo, da je to prenosna funkcija drugega reda. Sistem bo asimptotično stabilen, če sta \mathbf{K}_p in \mathbf{K}_d pozitivno definitni. Nesklopljenost sistema dobimo tako, da sta matriki diagonalni. Diagonalne člene zapišemo v obliki $\mathbf{K}_{pii} = \omega^2 n_{ii}$ in $\mathbf{K}_{dii} = 2\xi_i \omega n_{ii}$ [18], kjer je ω lastna frekvenca in ξ dušenje sistema drugega reda.

Shema takega vodenja je prikazana na sliki 4.4.

4.3.2 Vodenje v zunanjih koordinatah

V 4.2.2 smo robota vodili po hitrostih v sklepih in je bilo zato potrebno definirati tudi hitrost vrha robota. Pri vodenju preko inverzne dinamike pa smo pokazali, da je potrebno definirati ustrezne pospeške. Tako je sedaj potrebno izpeljati takšen $\ddot{\mathbf{q}}_u$, da bo robot sledil referenčnim pospeškom v zunanjih koordinatah. Z upoštevanjem relacije med hitrostmi v sklepih in hitrostmi na vrhu robota

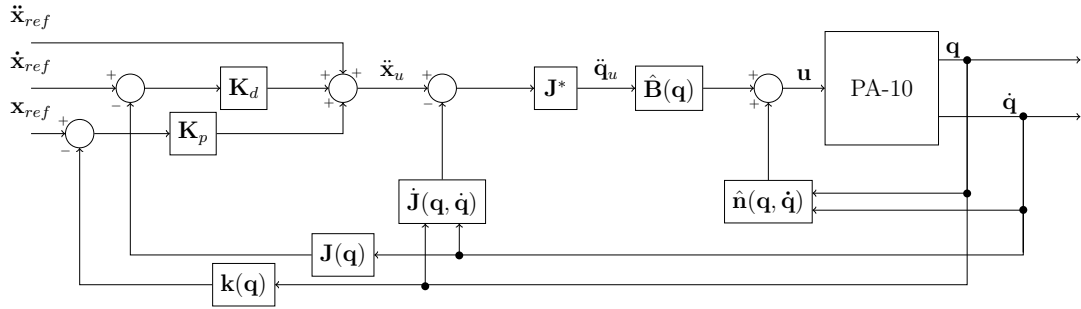
$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \quad (4.32)$$

in njenega odvoda

$$\ddot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{J}}(\mathbf{q})\dot{\mathbf{q}}. \quad (4.33)$$

dobimo

$$\ddot{\mathbf{q}}_u = \mathbf{J}^*(\mathbf{q})(\ddot{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}) \quad (4.34)$$



Slika 4.5: Bločna shema vodenja po zunanjih koordinatah preko inverzne dinamike.

Sedaj lahko definiramo regulator, ki bo dajal take pospeške v sklepih, da bo vrh robota sledil referenčni legi. Avtor v [1] predlaga sledeč regulator pospeškov

$$\ddot{\mathbf{x}}_u = \ddot{\mathbf{x}}_{ref} + \mathbf{K}_d \dot{\tilde{\mathbf{x}}} + \mathbf{K}_p \tilde{\mathbf{x}} \quad (4.35)$$

Z vstavitvijo 4.35 v 4.34 dobimo končno enačbo pospeškov v sklepih

$$\ddot{\mathbf{q}}_u = \mathbf{J}^*(\mathbf{q})(\ddot{\mathbf{x}}_{ref} + \mathbf{K}_d \dot{\tilde{\mathbf{x}}} + \mathbf{K}_p \tilde{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}). \quad (4.36)$$

Regulator zagotavlja konvergenco signala napake proti nič v stacionarnem stanju

$$\mathbf{K}_p \tilde{\mathbf{x}} = 0. \quad (4.37)$$

Shema takega vodenja je prikazana na sliki 4.5.

4.4 Interakcija z okoljem

Robotski mehanizmi so v večini primerov med opravljanjem nalog v kontaktu z okolico. Vendar pa obstaja veliko takih nalog, kjer nas sila med okolico in robotom ne zanima, npr. pobiranje in odlaganje. So pa primeri nalog, ko ključnega pomena. Primer take naloge je recimo brušenje površine ali pa interakcija s človekom. V nadaljevanju tega poglavja bomo opisali dva načina vodenja robota v kontaktu z okolico, admitančno ter impedančno.

4.5 Admitančno vodenje

Robotski mehanizem PA-10 je industrijski robot s segmenti iz pretežno litega železa. Segmenti imajo relativno veliko maso, sklepi pa relativno visoko trenje. Lahko predpostavimo, da ima manipulator veliko lastno impedanco. Pojem mehanske impedance izvira iz analogije električne impedance. Iz tega izraza izvira tudi izraz za admitančno vodenje robota. Tako vodenje je uporabno predvsem ob interakciji s človekom, saj se celoten robot giblje v skladu z gibanjem človeške roke [10]. Predstavlja pa razmerje med silo (F) in hitrostjo (v).

$$Z(s) = \frac{F}{v} = ms + b + \frac{k}{s} \quad (4.38)$$

oziroma razmerje med silo in pozicijo (x)

$$Z(s) = \frac{F}{x} = ms^2 + bs + k, \quad (4.39)$$

kjer je m masa, b viskozno dušenje, k togost [10].

Za admitančno vodenje je potrebno upoštevati še podatek o silah in navorih, ki delujejo na vrhu robota. Izmerjene sile in navori bodo označene s \mathbf{h}_e , referenčne sile in navori pa z \mathbf{h}_{ref} . Napaka sile je tako definirana kot:

$$\tilde{\mathbf{h}} = \mathbf{h}_{ref} - \mathbf{h}_e. \quad (4.40)$$

Sila kot rezultat delovanja robota na okolico je odvisna od hitrosti oz. pozicije vrha robota. Zato lahko napako sile dodamo v regulator lege 4.21. Definiramo referenčno pozicijo v odvisnosti od sile

$$\mathbf{x}_{ref} = \mathbf{K}_{hP}\tilde{\mathbf{h}} + \mathbf{K}_{hI} \int_0^t \tilde{\mathbf{h}} d\xi. \quad (4.41)$$

Z vstavitvijo enačbe 4.41 v 4.21 je mogoče zapisati celotno regulacijsko enačbo za vodenje robota v zunanjih koordinatah z upoštevanjem delovanja zunanjih sil na vrh robota.

$$\mathbf{u} = \mathbf{K}_p \mathbf{J}^{-1} (\mathbf{K}_{hP} \tilde{\mathbf{h}} + \mathbf{K}_{hI} \int_0^t \tilde{\mathbf{h}} d\xi - \mathbf{x}) \quad (4.42)$$

Dobili smo PI regulator sile interakcije. Čeprav je bilo pri definiciji mehanske impedance definirano, da je to razmerje med silo ter hitrostjo, se raje uporablja razmerje med silo ter pozicijo. Meritve sile so podvržene šumu, integrator pa deluje kot nizkopasovni filter.

Admitančno vodenje se uporablja takrat, ko lahko robota vodimo le hitrostno ali pozicijsko in želimo vplivati na okolico z neko silo. Silo interakcije pa je določimo z modelom okolja. Kontakt si lahko predstavljamo kot interakcijo med togim robotom in podajnim okoljem.

4.6 Impedančno vodenje

Pri izpeljavi vodenja lege robota z inverzno dinamiko smo predpostavili, da so sile interakcije \mathbf{h}_o enake nič. Ker pa to v primeru, da je robot v kontaktu z okoljem ni res, je potrebno pogledati kaj se zgodi takrat, ko te sile delujejo na robota. Vzemimo spet dinamičen model robota, ki ga vodimo preko reguliranega pospeška $\ddot{\mathbf{x}}_u$ in dodamo silo interakcije

$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})\ddot{\mathbf{q}}_u + \hat{\mathbf{n}}(q, \dot{q}) + \mathbf{J}^T \mathbf{h}_o. \quad (4.43)$$

Vidimo, da bodo sedaj navori v sklepi različni od izračunanih preko inverznega dinamičnega modela in posledično bo med želenim vrhom robota in dejanskim vrhom robota nastala napaka. Vrh robota bo z neko silo deloval na okolje, ki pa jo lahko s senzorjem na vrhu robota oz. med prijemalom in vrhom, izmerimo.

Za izhodišče vzamemo enačbo 4.37 in 4.43 ter zapišemo dinamiko napake z upoštevanjem delovanja zunanjih sil

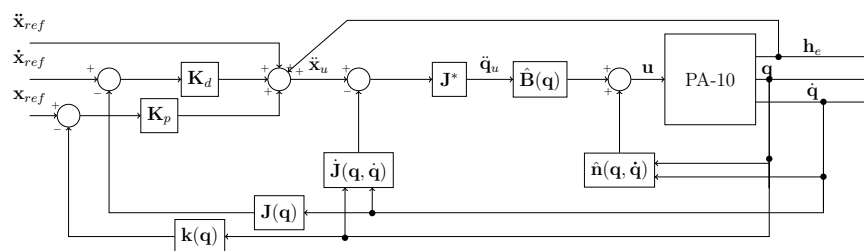
$$\ddot{\mathbf{x}}_{ref} + \mathbf{K}_d \dot{\tilde{\mathbf{x}}} + \mathbf{K}_p \tilde{\mathbf{x}}. \quad (4.44)$$

Če želimo napako kompenzirati je potrebno v regulator vključiti še podatek o izmerjenih silah na vrhu robota (\mathbf{h}_e). Na ta način dosežemo to, da se vrh robota obnaša kot neskončno tog sistem [1]. To pomeni, da če vodimo robota pozicijsko in pritisnemo na njegov vrh, bo kompenziral nastalo napako.

Zapišemo sedaj celotno enačbo, ki opisuje navore v sklepih pri impedančnem vodenju

$$\mathbf{u} = \hat{\mathbf{B}}(\mathbf{q})\mathbf{J}^*(\mathbf{q})(\ddot{\mathbf{x}}_{ref} + \mathbf{K}_d\dot{\tilde{\mathbf{x}}} + \mathbf{K}_p\tilde{\mathbf{x}} - \dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{h}_e) + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{J}^T\mathbf{h}_o. \quad (4.45)$$

Shema takega vodenja je prikazana na sliki 4.6.



Slika 4.6:

Impedančno vodenje se uporablja takrat, ko lahko robota vodimo preko vhodnih navorov. Z razliko od admitančnega vodenja, si lahko kontakt predstavljamo kot interakcijo med zelo togo okolico in podajnim robotom. Podajnost robota pa je odvisna od parametrov vodenja (\mathbf{K}_p , \mathbf{K}_d) in kompenzacije sile.

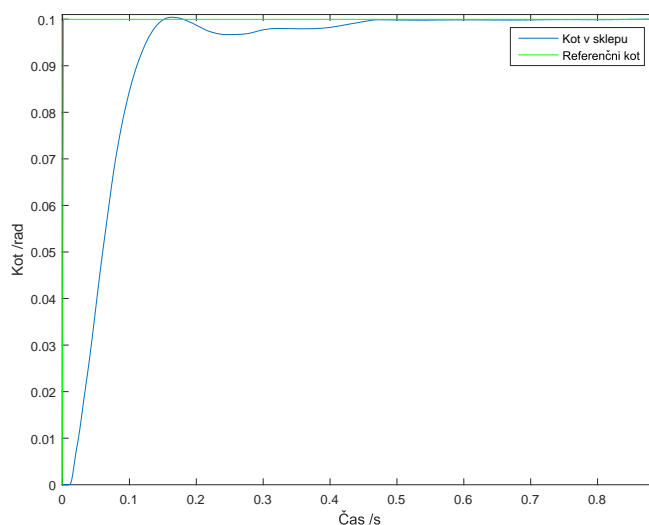
5 Implementacija

Poznavanje teoretičnih osnov za vodenje robotov nam omogoča realizacijo krmiljenja na način, ki je najbolj primeren zadani nalogi. V zgornjem poglavju so bile te osnove opisane in prikazane so bile blokovne sheme vodenja. Iz tistih shem, je videti, da je za implementacijo različnih vodenj potrebno poznati različno količino parametrov. Pri hitrostnem vodenju je enostavneje, saj je potrebno določiti le parametre ojačanj regulatorja. Pri navornem vodenju robota pa določitev parametrov še zdaleč ni enostaven problem saj je potrebno poznati dinamične parametre samega robota.

V nadaljevanju tega poglavja, bo opisana implementacija algoritmov vodenja, ki so bili opisani v poglavju 4. Implementacija vodenj je hkrati tudi preizkus s katerim se bo ovrednotilo namensko narejen strežnik. Preverilo se bo do katere frekvence vodenje v kontaktu še dobro sledi referenci in kdaj je še stabilno.

Vodenje bo je bilo realizirano z uporabo strežnika opisanega v poglavju 3. Za komunikacijo s strežnikom je bilo narejeno visoko nivojsko vodenje v programskem paketu MATLAB® SIMULINK®. Razlogi za izbiro omenjenega programskega paketa so bile dobre predhodne izkušnje ter dobro poznavanje delovanja programa s strani avtorjev. SIMULINK® bloki, ki se so se uporabljali in bodo v nadaljevanju prikazani na slikah, so podrobneje opisani v dodatku B.

Pri odzivu sistema na signal stopnico se bo preverilo tri pomembne parametre: mrtvi čas, čas ustalitve ter prenihaj.



Slika 5.1: Odziv sklepa na signal stopnice.

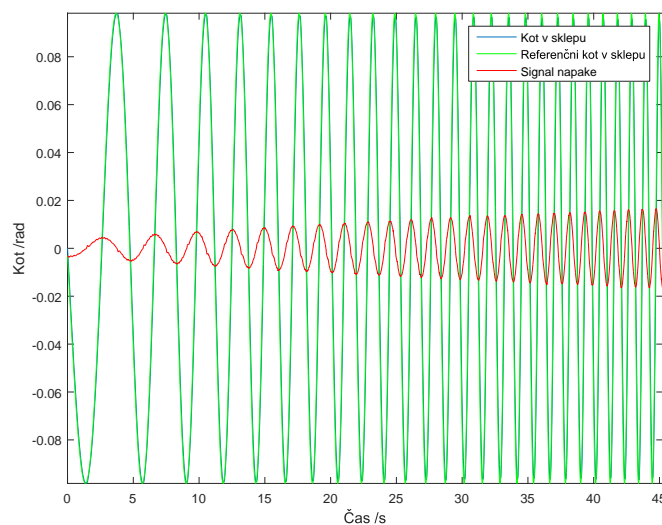
5.1 Hitrostno krmiljenje

Implementacija hitrostnega krmiljenja robota je potekala postopoma. Najprej je bilo realizirano pozicijsko vodenje sklepov nato pa vodenje v zunanjih trajektorijah. Ovrednotilo se je, kakšna je dinamika odziva na signal referenčni signal stopnice in kakšna je napaka ob sledenju signalu sinusne oblike s spreminjajočo se frekvenco.

5.1.1 Vodenje po trajektoriji

Odziv tretjega sklepa na referenčni signal stopnice je viden na sliki 5.1. Preizkus je bil postavljen tako, da je referenca najprej skočila na 0.1 rad, nato pa spet na nič. Na ta način se je sklep vrnil v prvotno lego. Iz grafa se je vidno, da se sistem ustali po VSTAVIČAS s. Hkrati pa je potrebno opozoriti na mrtvi čas, ki pa je 0,011 s. Videti je tudi, da pride do prenehaja. Ta je posledica vztrajnosti sklepa, ki se pri veliki hitrosti ne more nemudoma ustaviti.

Graf na sliki 5.2 prikazuje odziv tretjega sklepa na referenčni signal sinusne oblike. Referenčnemu signalu je frekvenca naraščala od 0.1 Hz do 1 Hz v časovnem

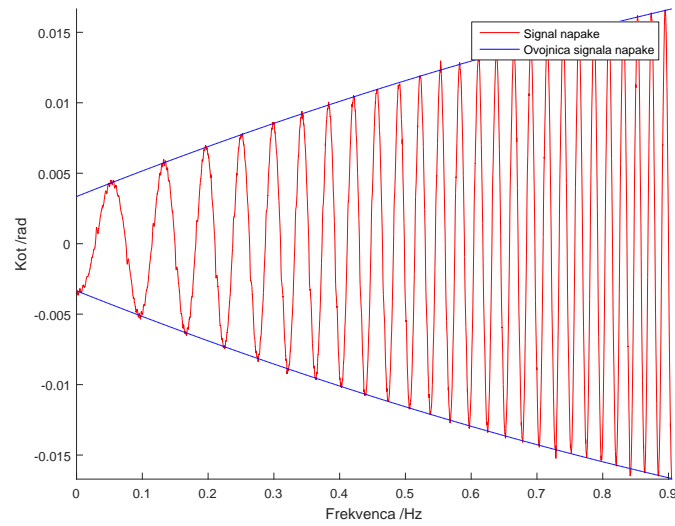


Slika 5.2: Odziv sklepa na signal sinusne oblike.

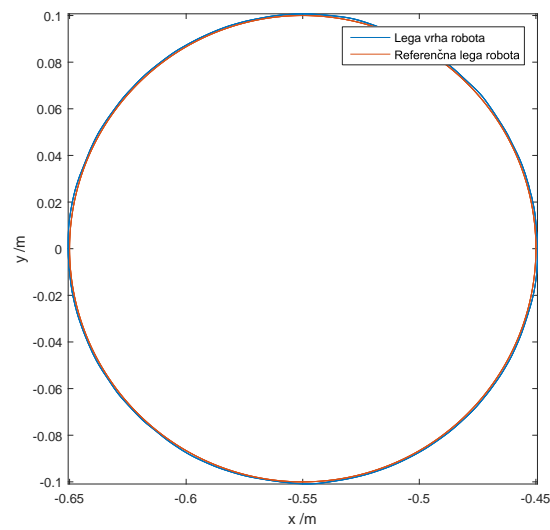
intervalu 50 sekund. Na ta način se je preverilo, koliko dobro še sledi referenci, ko se ta hitreje spreminja. Z rdečo obarvan signal napake prikazuje, da je sledenje s povečevanjem hitrosti slabše. Amplituda napake narašča v odvisnosti od frekvence. Graf 5.3 prikazuje povečan signal napake. Funkcija ovojnice je bila aproksimirana z polinomom drugega in sicer

$$y(f) = (0.0004f + 0.0033f^2) \frac{1}{50}. \quad (5.1)$$

Tretji poizkus vodenja robota je temeljil na vodenju v zunanjih koordinatah. Referenčna trajektorija je bila v obliki kroga na x in y ravnini. Graf je na sliki 5.4. Na grafu je videti, da vrh robota dobro sledi referenci. Največja napaka v sledenju je bila 0.006 m.



Slika 5.3: Napaka odziva in ovojnica signala napake.



Slika 5.4: Sledenje referenci v obliki kroga na x in y ravnini

5.1.2 Admitančno vodenje

5.2 Navorno krmiljenje

5.2.1 Kompenzacija trenja

5.2.2 Vodenje po trajektoriji

5.2.3 Vodenje v kontaktu z okolico

6 Zaključek

Literatura

- [1] B. Siciliano, L. Sciavicco, L. Villani in G. Oriolo, *Robotics, Modelling, Planning and Control*. London: Springer, 2009.
- [2] Mitsubishi Heavy Industries, *Portable General Purpose Intelligent Arm - Operating Manual*, rev. 1 izd.
- [3] M. H. I. Ltd., "Mitsubishi clean room robot," *Seminarska naloga*, 2003.
- [4] J. J. Craig, *Introduction to Robotics*. Z.D.A.: Pearson Education International, 2005.
- [5] N. D. Vuong in M. H. J. Ang, "Dynamic model identification for industrial robots," *Acta Polytechnica Hungarica*, 2009.
- [6] R. van der Aalst, M. H. J. Ang in H. Nijmeijer, "Dynamic identification of a mitsubishi pa-10 robotic manipulator," *ICAR*, 1999.
- [7] R. Jamisola, M. J. Ang, T. M. Lim, O. Khatib in S. Y. Lim, "Dynamics identification and control of an industrial robot," *ICAR*, 1999.
- [8] T. Petrič in M. Munih, "Kompenzacija trenja in gravitacije na robotu mitsubishi pa-10," *Seminarska naloga*, 2010.
- [9] J. Lenarčič in T. Bajd, *Robotski mehanizmi*. Ljubljana: Univerza v Ljubljani, Založba FE in FRI, 2003.
- [10] M. Mihelj, *Vodenje robotov*. Ljubljana: Univerza v Ljubljani, Založba FE in FRI, 2011.

- [11] D. Miljavec in P. Jereb, *Električni stroji: temeljna znanja*. Ljubljana: Univerza v Ljubljani, Založba FE in FRI, 2008.
- [12] Contemporary Controls, *ARCNET Tutorial*.
- [13] I. JR3, “JR3, Installation manual for force - torque sensors with internal electronics.” Dosegljivo: <http://www.jr3.com/product-manuals.html>. [Dostopano: 6. 17. 2015].
- [14] I. JR3, “JR3, DPS-based force sensor receivers, software and installation manual.” Dosegljivo: <http://www.jr3.com/product-manuals.html>. [Dostopano: 6. 17. 2015].
- [15] R. V. Aroca in G. Cauri, “A real time operating systems (rtos) comparison,” *XXIX Congresso da Sociedade Brasileira de Computação*, 2009.
- [16] D. Omerjen in B. Nemec, “Meritev gibanja kolena z industrijskim robotom - avtomatska kompenzacija gravitacije prijemala,” *Strojniški vestnik*, str. 87 – 97, 2002.
- [17] B. Zupančič, *Teorija regulacij*. Ljubljana: Univerza v Ljubljani, Založba FE in FRI.
- [18] M. Mihelj, T. Bajd in M. Munih, *Vodenje robotov*. Ljubljana: Univerza v Ljubljani, Založba FE in FRI, 2011.

Dodatek

A Dodatek 1

B Dodatek 2

C Dodatek 3

There is no spoon.