UNIVERZA V MARIBORU
FAKULTETA ZA STROJNIŠTVO
FAKULTETA ZA ELEKTROTEHNIKO,
RAČUNALNIŠTVO IN INFORMATIKO

Dušan FISTER

RAZVOJ NAPREDNEGA ADAPTIVNEGA REGULATORJA ZA MEHATRONSKE SISTEME

Magistrsko delo
študijskega programa 2. stopnje
Mehatronika

Maribor, avgust 2017

Univerza v Mariboru

Fakulteta za elektrotehniko,
računalništvo in informatiko

Fakulteta za strojništvo

RAZVOJ NAPREDNEGA ADAPTIVNEGA REGULATORJA ZA MEHATRONSKE SISTEME

Magistrsko delo

Študent:	Dušan FISTER
Študijski program:	študijski program 2. stopnje Mehatronika
Mentor FS:	red. prof. dr. Miran BREZOČNIK
Mentor FERI:	red. prof. dr. Riko ŠAFARIČ

Maribor, avgust 2017

Univerza v Mariboru

Fakulteta za elektrotehniko, računalništvo in informatiko

Fakulteta za strojništvo

Številka: M-BM0024
Datum in kraj: 30.05.2017, Maribor

Na osnovi 330. člena Statuta Univerze v Mariboru (Statut UM-UPB11, Uradni list RS, št. 44/2015, s sprem. in dopol. do 92/2015) izdajam:

SKLEP O ZAKLJUČNEM DELU

DUŠANU FISTERJU, študentu magistrskega študijskega programa druge stopnje **MEHATRONIKA,** se dovoljuje izdelati zaključno delo.

Tema zaključnega dela je pretežno s področja **Katedre za proizvodno strojništvo.**

Mentor FS:	**red. prof. dr. Miran Brezočnik**
Mentor FERI:	**red. prof. dr. Riko Šafarič**
Zunanji delovni somentor:	/

Naslov zaključnega dela: **Razvoj naprednega adaptivnega regulatorja za mehatronske sisteme**

Naslov zaključnega dela v angleškem jeziku: **Development of advanced adaptive controller for mechatronic systems**

Rok za izdelavo in oddajo zaključnega dela je: **30.05.2018.** Zaključno delo je potrebno izdelati skladno z »Navodili za pripravo magistrskega dela« in ga v treh izvodih oddati v pristojnem referatu članice. Hkrati se odda tudi izjava mentorjev o ustreznosti zaključnega dela ter poročilo o preverjanju podobnosti z drugimi deli.

Pravni pouk: Zoper ta sklep je možna pritožba na Senat članice v roku 10 delovnih dni od dneva prejema sklepa.

UNIVERZA V MARIBORU FAKULTETA ZA STROJNIŠTVO 7

Dekan:

red. prof. dr. Bojan Dolšak

Obvestiti:
- kandidata,
- mentorja,
- odložiti v arhiv.

FS

www.fs.um.si | fs@um.si | t +386 2 220 7500 | f +386 2 220 7990 | TRR: 0110 0609 0102 935 | ID DDV: SI71674705

I Z J A V A

Podpisani _____, izjavljam, da:

- je magistrsko delo rezultat lastnega raziskovalnega dela,
- predloženo delo v celoti ali v delih ni bilo predloženo za pridobitev kakršnekoli izobrazbe po študijskem programu druge fakultete ali univerze,
- so rezultati korektno navedeni,
- nisem kršil-a avtorskih pravic in intelektualne lastnine drugih,
- soglašam z javno dostopnostjo magistrskega dela v Knjižnici tehniških fakultet ter Digitalni knjižnici Univerze v Mariboru, v skladu z Izjavo o istovetnosti tiskane in elektronske verzije zaključnega dela.

Maribor,_____ Podpis: _____

ZAHVALA

Zahvaljujem se mentorjema red. prof. dr. Miranu BREZOČNIKU ter red. prof. dr. Riku ŠAFARIČU za njuno pomoč in vodenje pri opravljanju magistrskega dela. Prav tako se zahvaljujem doc. dr. Miranu RODIČU za podane nasvete.

Zahvaljujem se tudi staršem ter bratu za njihovo podporo.

RAZVOJ NAPREDNEGA ADAPTIVNEGA REGULATORJA ZA MEHATRONSKE SISTEME

Ključne besede: online-regulacija, identifikacija, evalvacija, optimizacija, realni čas

UDK: 681.52(043.2)

POVZETEK

V magistrski nalogi predstavljamo, opisujemo ter razlagamo princip delovanja nelinearnega naprednega adaptivnega hitrostnega regulatorja, ki smo ga izdelali za potrebe napredne regulacije na enoosnem robotu. Tega krmilimo z algoritmom evolucijskih strategij, ki je sposoben dinamičnega iskanja rešitev, kjer se vrednost funkcije uspešnosti spreminja s časom. Pri tem smo vrednost funkcije uspešnosti napovedovali s pomočjo nevronske mreže (angl. Artificial Neural Network, krajše ANN). Predlagano metodo smo testirali na realnem laboratorijskem robotskem sistemu z eno stopnjo prostosti (angl. one degree of freedom, krajše 1 D.O.F.) in ugotovili, da je primeren za regulacijo v realnem času (odzivni čas 1-5 ms). Izvedena je bila primerjava z linearnim PI-hitrostnim regulatorjem, rezultati pa so pokazali uspešnejše delovanje nelinearnega hitrostnega regulatorja.

DEVELOPMENT OF ADVANCED ADAPTIVE CONTROLLER FOR MECHATRONIC SYSTEMS

Key words: online control, identification, evaluation, optimization, real-time

UDK: 681.52(043.2)

ABSTRACT

In this M.Sc. Thesis, a non-linear advanced adaptive controller is proposed and described, which was used for the purpose of velocity control on a single degree of freedom robotic mechanism. The controller is based on the algorithm of dynamic evolution strategy, which is capable of searching the global optimum dynamically, i.e. when the fitness function is changing through time. The fitness function calculation has been done by implementing an artificial neural network, which was simulating the behavior of the real system. We tested the proposed algorithm on the robotic mechanism and compared the results to the PI-velocity controller. We concluded, that the proposed approach of identification and optimization is appropriate for online real-time control (response time 1-5 ms). Furthermore, non-linear controller outperformed the linear controller, according to conducted tests.

KAZALO VSEBINE

UPORABLJENI SIMBOLI

P, PI, PD, PID	linearni regulator
K_p	proporcionalno ojačenje regulatorja
K_i	integralno ojačenje regulatorja
i_a, i_r	dejanski in želeni električni tok
ω_a, ω_r, $\hat{\omega}$	dejanska, želena in ocenjena hitrost
φ_a , φ_r	dejanski in želeni položaj
w_J, w_L	uteži umetne nevronske mreže
τ	učna konstanta algoritma evolucijskih strategij
δ	parameter algoritma evolucijskih strategij
MAX_GEN	število generacij
N(0,1)	naključno število po Gaussovi porazdelitvi s srednjo vrednostjo nič in standardno deviacijo ena
T	časovna konstanta

UPORABLJENE KRATICE

ANN	umetna nevronska mreža
D.O.F	prostostna stopnja (angl. degree of freedom)
DSC	digitalni signalni krmilnik (angl. digital signal controller)
CLA	matematični koprocesor (angl. control law accelerator)
MAC	vzporedni ukaz za zmnoži in shrani (angl. multiply and accumulate)
BPG	»backpropagation« učni algoritem umetne nevronske mreže
ES	algoritem evolucijskih strategij
(1+1)-DES	algoritem dinamičnih evolucijskih strategij

1 UVOD

1.1 Opis problema

Napredni adaptivni regulatorji lahko v veliki meri izboljšajo delovanje regulacijske zaprte zanke v primerjavi z linearnimi regulatorji (P-, PI-, PD-, PID-regulatorji) na nelinearnih sistemih. Nelinearni sistemi se razlikujejo po periodičnih ali neperiodičnih motnjah nelinearnosti, ki neugodno vplivajo na reguliran sistem. Običajni linearni regulatorji teh motenj nelinearnosti, zaradi tega, ker zgodovine ne pomnijo, ne morejo predvideti ali odpraviti, medtem ko so napredni adaptivni regulatorji zaradi pomnjenja sposobni prilagajati se (adaptirati) trenutnim razmeram. Predvidijo lahko določene motnje nelinearnosti in jih aktivno odpravijo. Slednji se lahko na trenutne razmere prilagajajo v realnem času, tj. sprotno (angl. online) oz. naknadno (angl. offline).

S principom sprotne identifikacije sistema (modeliranja) lahko z dejanskimi vhodi in izhodi posnamemo del transformacije regulacijske proge. Na ta način k realni regulacijski zanki dodamo še vzporedno virtualno regulacijsko zanko, za katero predpostavljamo, da dovolj natančno posnema obnašanje realnega sistema in je zato primerna za napovedovanje. To dejstvo omogoča uporabo optimizacijskega algoritma, ki bi bil sposoben poiskati najbolj optimalno napovedano (ocenjeno) rešitev za naslednji cikel. Težava, ki se ob tem pojavlja je, da morata biti postopka identifikacije in optimizacije, pri večini robotskih mehanizmov, zaključena v času T = 1-5 ms. Zaradi kratke električne časovne konstante je to namreč zgornja meja za zagotavljanje dovolj velike fazne rezerve in posledično stabilnosti. Ob tem se pojavlja dodatna težava, tj. potreba po visoki računski zmogljivosti mikroračunalnika, ki je v zgodovini pomembno vplivala na razvoj tovrstnih regulatorjev, ki se do danes niso pojavljali v širši javnosti.

1.2 Pregled sorodnih del

Princip identifikacije sistema z univerzalnim aproksimatorjem – umetno nevronsko mrežo [1] – je za potrebe regulacije, tako statične kot dinamične, že dolgo znan [2]. Slednja je uspešno

implementirana na različnih področjih, npr. na področju prepoznavanja slik [3], ekonomije [4] in procesiranja signalov [5].

Optimizacijski algoritmi po vzoru iz narave so močno računsko orodje za iskanje optimumov, ki še posebej v zadnjem času pridobivajo na popularnosti. Delo [6] predstavlja pregled temeljnih optimizacijskih algoritmov po vzorih iz narave, ki so bili najbolj razširjeni do leta 2013. Ti so bili uspešno implementirani na najrazličnejših področjih elektronike, npr. naknadni optimizaciji regulatorjev [7], optimizaciji DC/DC pretvornika [8] in optimizaciji oblike anten [9].

Čeprav realnih aplikacij optimizacije s kombinacijo identifikacije sistema za tako kratke čase tipanja (T= 5 ms) nismo zasledili, pa je princip poznan za počasnejše regulacijske proge [10]. Zanimiv primer sprotne adaptacije parametrov regulatorjev z metodami mehke logike predstavlja [11].

1.3 Zastavljena rešitev

V sklopu magistrske naloge želimo izdelati unikaten tip naprednega adaptivnega regulatorja, ki bi zagotavljal stabilno in dolgoročno delovanje nelinearnega robotskega mehanizma. Za vzporedni proces identifikacije in vrednotenja želimo uporabiti usmerjeno nevronsko mrežo, na kateri bi z algoritmom evolucijskih strategij preizkušali poskusne rešitve ter v vsakem optimizacijskem ciklu izbrali najprimernejšo ocenjeno rešitev, ki bi bila napovedana za naslednji cikel. Povrhu bi radi opisano rešitev izvajali sprotno na realnem robotskem mehanizmu v realnem času. Take ne želimo izdelati le adaptacije parametrov regulatorja, ampak zasnovati celotni regulator na novo (angl. from scratch).

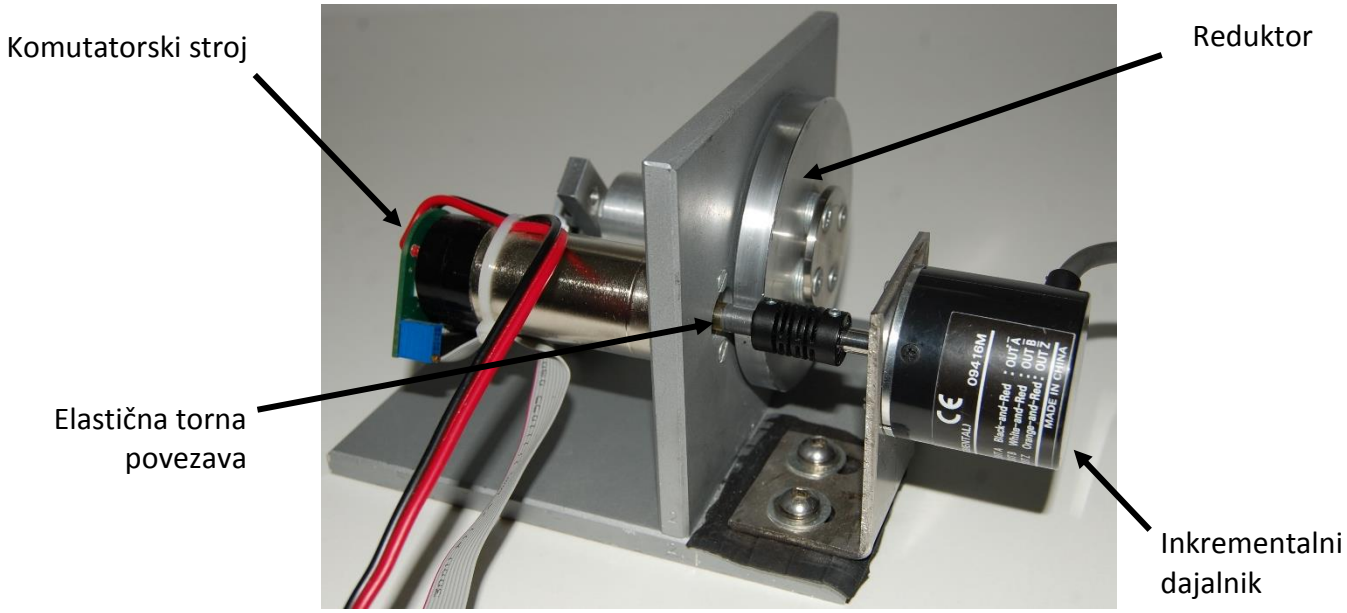
1.4 Struktura magistrske naloge

Magistrska naloga je razdeljena v štiri dele. V drugem poglavju navedemo in opisujemo bistvene sestavne komponente laboratorijskega mehanizma. Tretje poglavje govori o uporabljenih metodah – poleg regulacije in izvajanja v realnem času še optimizacijo, identifikacijo ter vrednotenje. Vsak omenjeni sklop je v tem poglavju opisan podrobneje. V četrtem poglavju predstavimo rezultate naprednega adaptivnega regulatorja na realnem

mehanizmu ter jih primerjamo s PI-hitrostnim regulatorjem. Peto poglavje povzema glavne ugotovitve ter začrta smernice za nadaljnje delo.

2 OPIS LABORATORIJSKE OPREME

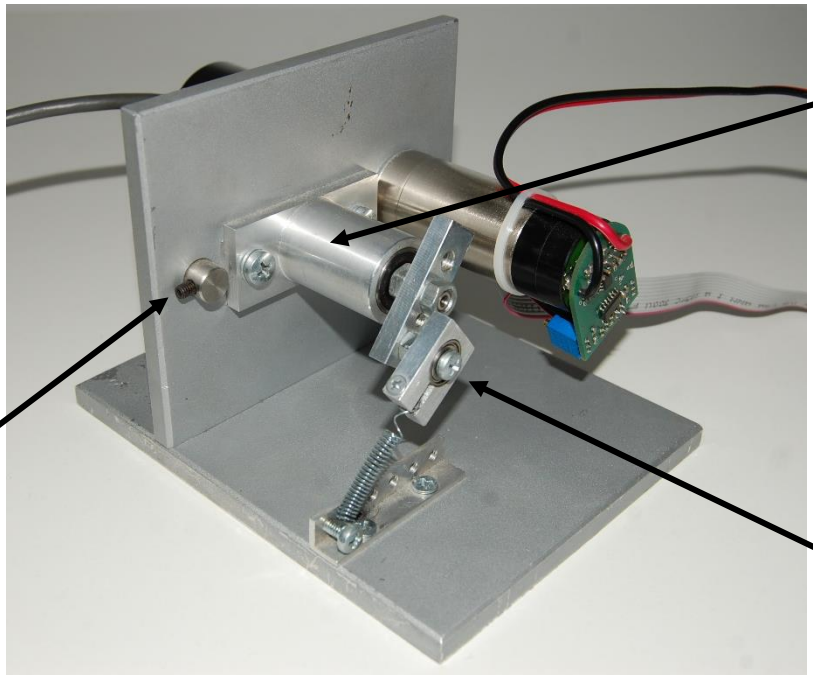
Realni robotski mehanizem sestoji iz komutatorskega stroja (v nadaljevanju motorja), ki je preko elastične torne povezave spet z reduktorjem. Zasnova pod določenimi pogoji omogoča zdrsavanje reduktorja, kar pomembno vpliva na kakovost regulacije. Zdrsavanje je mogoče z uporabo namenskega vijaka, ki je viden s slike 2.2, tudi nadzorovati. Na hrbtni strani reduktorja je ekscentrično vezana vzmet, ki v sistem vnaša nelinearno motnjo. Slednja se razteza in krči ter sili komutatorski stroj, da menjava med režimoma motor/generator. Ravno med njima se nahaja singularna točka, kjer je regulacija najzahtevnejša (slika 2.3). Motorna gred je togo povezana z inkrementalnim dajalnikom, ki meri položaj oz. zasuk gredi. Slika 2.1 prikazuje čelno stran robotskega mehanizma, medtem ko slika 2.2 hrbtno stran. V nadaljevanju predstavljamo posamezne sestavne dele sistema: motor, inkrementalni dajalnik, močnostni pretvornik, tokovni senzor in mikrokrmilnik.



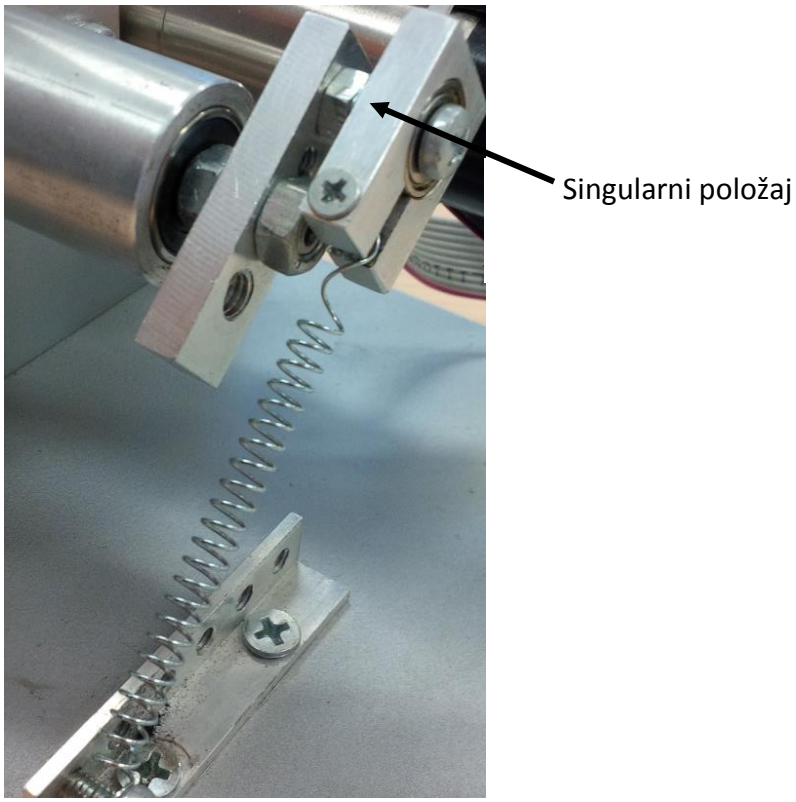
Slika 2.1: Čelna slika robotskega mehanizma

2.1 Motor ESCAP 28D11-219P

Omenjeni motor je nazivne napetosti U = 12 V DC in nazivnega toka I = 1.5 A. Zavrti se lahko do ω^{max} = 600 rad/s, kar ob prestavnem razmerju reduktorja i = 11.25 poda izhodno hitrost približno ω^{red} = 53 rad/s. Več podatkov o uporabljenem motorju je zapisanih v tabeli 2.1.



Slika 2.2: Hrbtna slika robotskega mehanizma



Slika 2.3: Singularni položaj robotskega mehanizma

Tabela 2.1: Podatki o motorju ESCAP 28D11-219P

Nazivni podatki	12 V, 1.5 A, 600 rad/s
Vztrajnostni moment	$J_m = 17.6 \cdot 10^{-7}$ kgm^2
Koeficient viskoznosti	$B_m = 1 \cdot 10^{-7}$ Nms/rad
Nazivni navor	$T_m = 28.4 \cdot 10^{-3}$ Nm
Upornost navitja	R_a = 2.5 + 0.38 Ω
Induktivnost navitja	L_a = 0.3 + 1.5 mH
Električna in mehanska časovna konstanta	K_e, K_m = 0.0195 Vs/rad, Nm/A

Za zmanjšanje tokovnih konic in dodatno omejevanje toka (prek dodatne upornosti) je k osnovnemu motorju zaporedno vezana še dodatna dušilka (lastnosti so navedene kot seštevek upornosti in induktivnosti). Dušenje je zato večje, kar proces regulacije nekoliko olajšuje, saj se lahko časi tipanja tokovne regulacije zvišajo [12]. Dodatna induktivnost v stacionarnem stanju, razen nekoliko višje upornosti, nima vpliva na obnašanje motorja.

2.2 Inkrementalni dajalnik Omron E6B2-CWZ1Y

Položaj gredi je merjen z inkrementalnim dajalnikom Omron E6B2-CWZ1Y (slika 4). Dosega ločljivost R = 2000 črtic/obrat, katero lahko z uporabo smernega diskriminatorja početverimo. Njegova napajalna napetost znaša U = 5 V DC, kar ugodno vpliva na neposredno povezavo s krmilnikom, ki ga predstavljamo v nadaljevanju. Inkrementalni dajalnik je dvokanalni, kar pomeni, da na izhodu, poleg indeksnega pulza Z in njegove negacije, generira kar štiri izhode – signal A, B ter pripadajoči negaciji. Signala A in B sta med sabo zamaknjena za fazni kot φ = 90°, zato lahko iz obeh pridobimo tudi podatek o smeri.

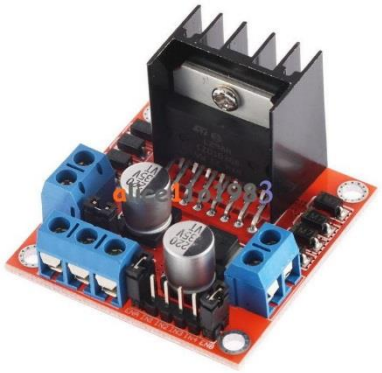
2.3 Močnostni pretvornik STM L298N

Močnostni pretvornik skrbi za pretvorbo signalnih pulzov v močnostne (ojačevalnik). Uporabljen močnostni pretvornik napajamo z napetostjo U = 12 V DC, sam pa lahko prek mostičnega vezja menjava med obema smerema vrtenja. Nazivni tok, za močnostni pretvornik znaša I = 2 A, kar je četrtino več kot motor. Pomembno je, da vzamemo pretvornik, ki je nekoliko močnejši od motorja, saj bi v nasprotnem primeru lahko prišlo do preobremenitve pretvornika, ta pa bi se lahko pregrel in postal neuporaben.

Slika 2.4: OMRON E6B2-CWZ1Y

Močnostni pretvornik L298N je pravzaprav čip, ki ga je treba povezati še z dodatnimi elektronskimi komponentami. Seveda obstajajo na tržišču že izdelane ploščice pripravljene za

takojšnjo uporabo, npr. »L298N board«. Ploščico prikazano na sliki 2.5, smo uporabili tudi v našem primeru.



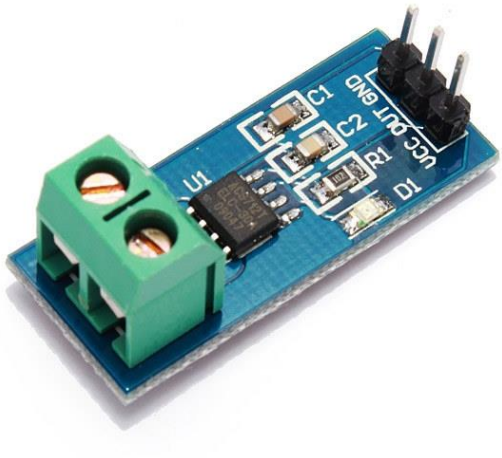
Slika 2.5: Ploščica s STM L298N

2.4 Tokovni senzor Allegro ACS712

Tokovni senzor je namenjen merjenju toka skozi motor (obremenjenosti), zato je njegove priključke treba vezati zaporedno z motorjem. ACS712 za merjenje toka izkorišča Hallov pojav, ki generira napetost v odvisnosti od smeri in jakosti toka. Slednji ima lahko definirane tri poglavitne obratovalne točke:

1. I = 0 A, izhodna napetost U_o = 2.5 V,
2. I = 5 A, izhodna napetost U_o = 5 V,
3. I = -5 A, izhodna napetost U_o = 0 V.

Iz treh točk je razvidno, da je največji razpon tokovnega senzorja I = +/- 5 A, v vmesnih točkah pa velja pravilo linearne interpolacije. Prav tako so se na tržišču razvile ploščice, t.i. »ACS712 board«, prikazane na sliki 2.6, ki smo jo uporabili tudi v našem primeru.



Slika 2.6: ACS712 z razponom merjenja toka I = +/- 5 A

2.5 Mikrokrmilnik TMS320F28377S

Digitalni signalni krmilnik (angl. Digital Signal Controller, krajše DSC) TMS320F28377S proizvajalca Texas Instruments omogoča povezavo vhodov z izhodi ter obenem zagotavlja visoko računsko zmogljivost. Programiranje poteka v razvojnem okolju Code Composer Studio, v programskih jezikih C in ASM (zbirni jezik). Procesor ima poleg glavnega jedra, tj. CPU C28X na voljo še matematični koprocesor (pospeševalnik) (angl. Control Law Accelerator, krajše CLA), zato poleg programske paralelizacije (sočasno izvajanje dveh ali več ukazov) podpira tudi strojno paralelizacijo.

Programska paralelizacija se je v glavnem uveljavila z izvajanjem dveh ukazov v enem ciklu, npr. ukazom množi-in-akumuliraj (angl. multiply-and-accumulate, krajše MAC). Tem pravimo tudi vzporedni ukazi (angl. parallel instruction), je pa za njihovo uporabo obvezno poznavanje zbirnega jezika.

Tabela 2.2 prikazuje osnovne specifikacije uporabljenega krmilnika DSC. Programska paralelizacija je razvidna iz druge in tretje kolone tabele 2.2, kjer je kot takt ure navedena vrednost 200 MIPS (angl. Milion Instructions Per Second), kar pomeni približno 200 milijonov ukazov na sekundo, medtem ko je hitrost samega procesiranja 400 MIPS. Programska paralelizacija pomeni uporabo vzporednih ukazov v slehernem ciklu, kar je dejansko dosegljivo, a v praksi težko izvedljivo, zato je dejanska računska moč ustrezno manjša.

Uporaba strojne paralelizacije z uporabo koprocesorja omogoča podvojitev računske moči procesorja. Pospeševalnik CLA je v splošnem namenjen izvajanju kratkih regulacijskih programov, saj omogoča izključno programiranje v zbirnem jeziku. Prav zadnja lastnost omogoča popoln nadzor in izkoristek nad koprocesorjem ter zahtevnejšo optimizacijo kode. Tabela 2.2 predstavlja osnovne značilnosti mikrokrmilnika DSC TMS320F28377S prikazanega na sliki 2.7 [13].

Tabela 2.2: Lastnosti CPU C28x in CLA

Lastnost	CPU C28x	CLA
Hitrost procesiranja	400 MIPS	400 MIPS
Takt ure	200 MHz	200 MHz
Flash pomnilnik	1024 kB	512 kB
RAM pomnilnik	164 kB	132 kB
Št. ADC kanalov	24	24
Programski jezik	C, ASM	ASM

Slika 2.7: Mikrokrmilnik TMS320F28377S

3 NAPREDNI ADAPTIVNI REGULATOR V REALNEM ČASU

V tem poglavju opisujemo razvoj naprednega adaptivnega regulatorja v realnem času NA-OR (angl. Advanced Adaptive Online Regulator), katerega delimo na pet sklopov:

- regulacija,
- identifikacija,
- vrednotenje,
- optimizacija,
- verifikacija.

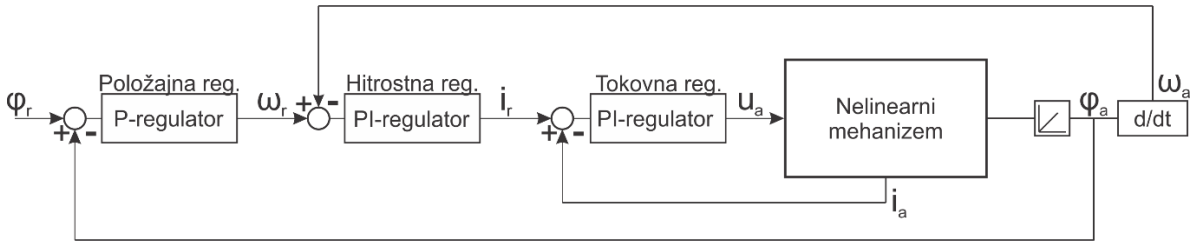
V prvem sklopu opisujemo principe linearne ter napredne adaptivne regulacije. Navedeni so najpomembnejši regulacijski podatki, predstavljena pa je tudi regulacijska histereza, ki se pojavlja na robotskem mehanizmu. Proces identifikacije zajema opis uporabljene usmerjene nevronske mreže s pripadajočimi nastavitvami. Tretji sklop zaobjema proces vrednotenja, ki je tesno povezan z uporabo nevronske mreže, kjer z njeno pomočjo napovemo vrednost ocenitvene funkcije, ki jo kliče optimizacijski algoritem. Optimizacijski algoritem temelji na evolucijski strategiji, ki je primerna za reševanje dinamičnih problemov. Zadnji sklop predstavlja integracijo vseh petih sklopov v realno računalniško zasnovo, ki omogoča adaptivno regulacijo robotskega sistema NA-OR.

3.1 Regulacija

Regulacija (vodenje) je postopek sledenja dejanske (merjene) vrednosti želeni (referenčni). Razliko med njima definira t.i. regulacijski pogrešek *e*, ki se v stabilnih regulacijskih sistemih zmanjšuje s časom. Naš regulacijski problem je tokovna, hitrostna in položajna zanka nelinearnega mehanizma. Slednjega smo poskušali reševati na dva načina.

Tokovni ter položajni regulator sta v obeh načinih identična. Prikazana sta na sliki 3.1, iz katere je razvidna delno kvazi zvezna kaskadna regulacijska proga (čas tipanja je najmanj desetkrat manjši od najmanjše časovne konstante regulacijske proge). Zaradi slednje teze je dejansko kvazi zvezna zgolj tokovna regulacijska zanka, medtem ko hitrostna položajna nista. Omenjena predpostavka otežuje regulacijski problem. Tokovna regulacija je izvedena s časom tipanja $T_{st} = 2.5$ µs (frekvenca 40 kHz), hitrostna regulacija s $T_{sh} = 5$ ms (frekvenca 200 Hz) in položajna

regulacija s časom tipanja T_{sp} = *5* ms (frekvenca 200 Hz). Prikazana regulacijska proga predstavlja prvi način reguliranja mehanizma po hitrosti/položaju,



Slika 3.1: Delno kvazi zvezna kaskadna regulacijska shema

iz katere lahko izrazimo naslednje pojme:

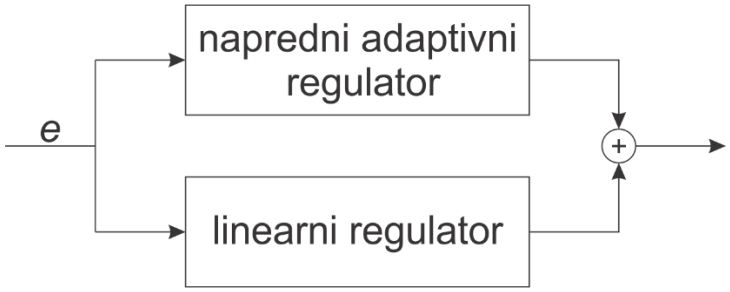
φ_r	[rad]	– želeni položaj (zasuk) reduktorja,
φ_a	[rad]	– dejanski položaj (zasuk) reduktorja,
ω_r	[rad/s]	– želena hitrost reduktorja,
ω_a	[rad/s]	– dejanska hitrost reduktorja,
i_r	[A]	– želeni tok motorja in
i_a	[A]	– dejanski tok motorja.

Razlika med posamezno želeno in dejansko vrednostjo definira regulacijski pogrešek:

- tokovni pogrešek e_t,
- hitrostni pogrešek e_h in
- položajni pogrešek e_p.

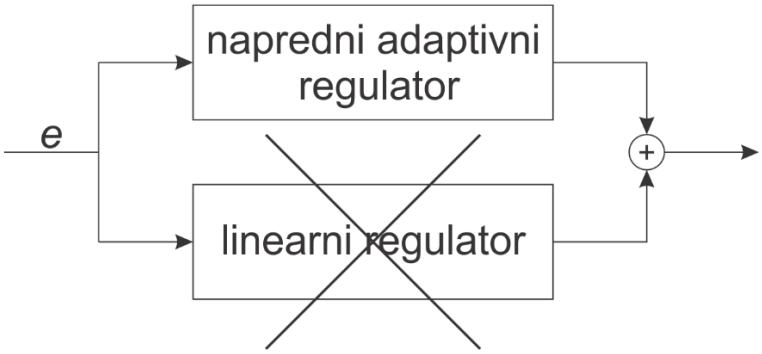
Drugi način reševanja regulacijskega problema je vključevanje adaptivnega regulatorja v kvazi zvezno regulacijsko progo. Poznanih je več principov vključevanja naprednega adaptivnega regulatorja v regulacijsko progo, med katerimi je najenostavnejša linearna kombinacija (seštevanje) z linearnim regulatorjem prikazana na sliki 3.2. V glavnem je dejaven linearni regulator, medtem ko se v posebnih okoliščinah (pri pojavu motenj) aktivira tudi napredni adaptivni regulator. Ta metoda dovoljuje določeno napako, ki jo napredni adaptivni regulator lahko vnese v sistem, saj je slednji stohastične narave in pri konstrukciji rešitev odvisen od uporabe generatorja naključnih števil. To napako v naslednjem ciklu linearni regulator lahko

kompenzira, zato je metoda še posebno uporabna v času prilagajanja naprednega adaptivnega regulatorja na realno progo. Običajno za to prilagajanje uporabljamo umetne nevronske mreže, ki pa za optimalno prilagoditev zahtevajo določeni čas.



Slika 3.2: Metoda kombiniranja linearnega in naprednega adaptivnega regulatorja

Zahtevnejša metoda vključevanja naprednega adaptivnega regulatorja v regulacijsko progo je popolna nadomestitev linearnega regulatorja z naprednim adaptivnim regulatorjem. V tem primeru mora napredni adaptivni regulator zaobjeti in upoštevati vse motnje in nelinearnosti, v procesu regulacije pa vpliv motenj odpraviti in nelinearnosti linearizirati ter razklopiti. Omenjeno metodo delovanja regulatorja v nadaljevanju uporabljamo tudi v našem primeru. Hitrostno regulacijsko zanko želimo z zaobjemanjem motenj linearizirati ter izboljšati vodenje po položaju. Rezultate, pridobljene z naprednim adaptivnim regulatorjem želimo primerjati s PI-hitrostnim regulatorjem ter preveriti jakost in vpliv nelinearnosti. Čas tipanja hitrostne in položajne zanke želimo zmanjšati kolikor se le da in s tem dodatno otežiti problem.

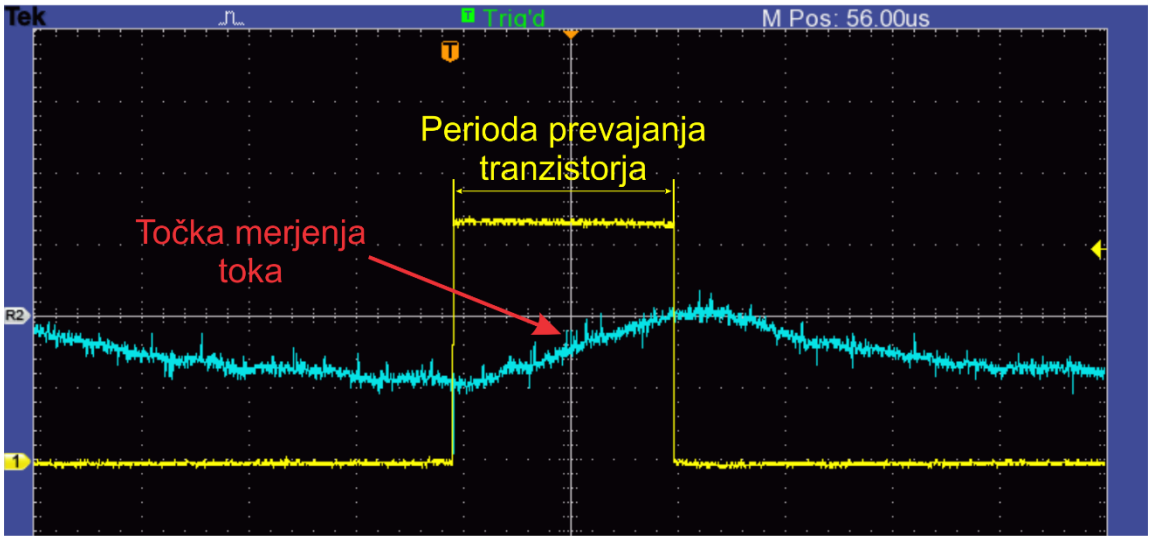


Slika 3.3: Metoda popolne zamenjave linearnega regulatorja z naprednim adaptivnim regulatorjem

Tokovna regulacija

Tokovni regulator je tipa PI z zunanjo zaščito pred prenihajem (angl. antiwindup PI-controller). Princip tokovnega PI-regulatorja temelji na izračunu tokovne napake $e_t = i_r - i_a$, pri čemer je i_r podana, i_a pa izmerimo iz t.i. trikotniške metode merjenja toka. Shema trikotniške metode je predstavljena grafično s posnetkom iz osciloskopa na sliki 3.4 (ob povečanem času tipanja tokovne regulacije za primernejši prikaz). Iz slike je razvidno naraščanje toka v času prevajanja tranzistorja na močnostnem pretvorniku, merjenje toka natanko v sredini periode ter naknadno padanje toka do naslednje periode. Vidimo, da gre za zvezni način delovanja pretvornika, saj tok skozi motor nikoli ne doseže vrednosti $i_a = 0$ A.

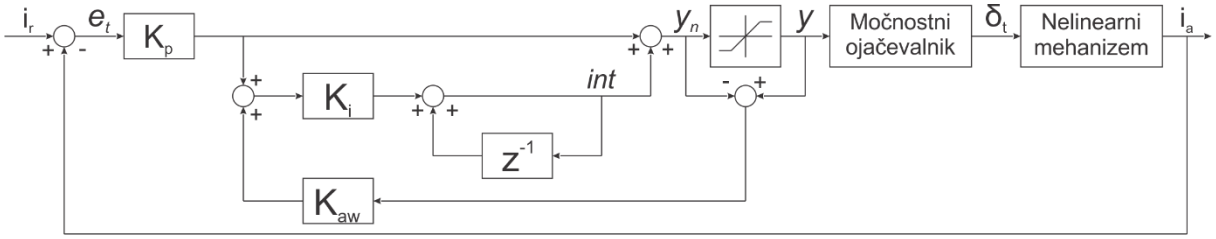
Izhod iz ACS7212 ni neposredno vezan na A/D vhod mikrokrmilnika DSC, ampak je vmes vezan še filter RC, ki filtrira neželeni visokofrekvenčni šum.



Slika 3.4: Perioda odprtja tranzistorja in točka merjenja toka

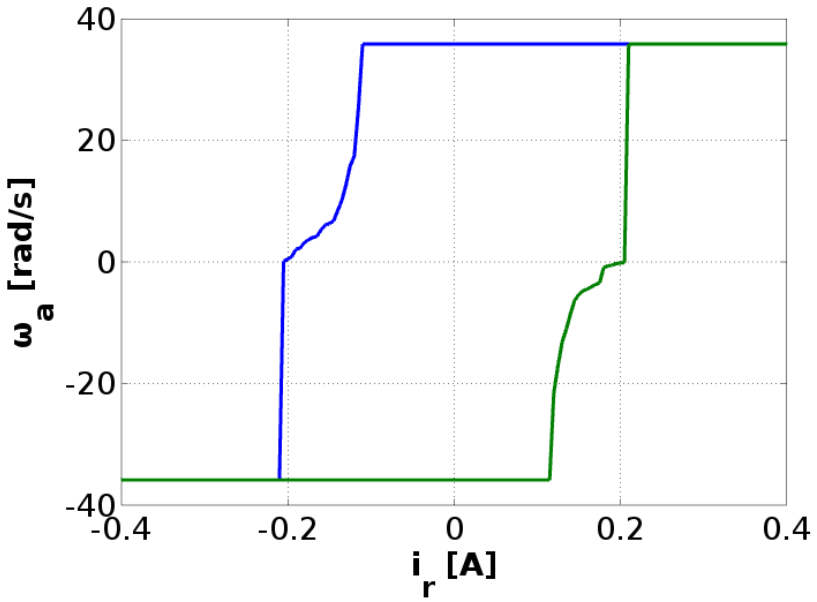
Pridobljeno tokovno napako pomnožimo z ustrezno nastavljeno proporcionalno kostanto $K_p = 0.03$ in integriramo z integracijsko konstanto $K_i = 0.005$, ki ustreza obratnemu integracijskemu času ($1/T_i$). Vhodni signal e_t preoblikujemo v izhodnega y z zunanjo limito. Dokler signal y_n (nelimitirani izhod) ne presega določene meje limite, rezultat odštevanja slednjega z limitiranim y znaša 0, konstanta $K_{aw} = 1$ pa v tem primeru ni uporabna. V kolikor pride do preseganja meje, pa razlika ni več nič. Ta razlika se pomnoži s konstanto K_{aw} in prišteje k integracijskemu delu. Ideja na zanesljiv način preprečuje integratorski pobeg (slika 3.5).

Signal *y* je nato pripeljan do močnostnega pretvornika, ki ga pretvori v signal širine odprtja tranzistorja δ_t, ojača in aplicira na motor. Regulacija je, zaradi uporabe mikrokrmilnika DSC digitalna, kar pomeni posluževanje digitalne tehnike integracije – uporabljamo Gaussovo integracijsko metodo z dodajanjem prejšnjega integracijskega signala *int*.



Slika 3.5: Digitalna regulacijska shema PI-antiwindup tokovnega regulatorja

Z ročnim spreminjanjem želenega toka i_r je moč posneti nelinearno karakteristiko motorja, napajalnika in nelinearnega mehanizma (slika 3.6) v odvisnosti od dejanske hitrosti ω_a.



Slika 3.6: Nelinearnost robotskega mehanizma

Graf na sliki 3.6 je nastal z ročnim spreminjanjem želene tokovne veličine i_r in beleženjem dejanske hitrosti ω_a. Ob držanju želenega toka na vrednosti i_r = 0 A je dejanska hitrost ω_a praktično maksimalna ali minimalna (pojav histereze). Za to je odgovorna vzmet, ki z delovanjem v smeri najmanjše prožne energije pospešuje celotni mehanizem. Ročno

nastavljanje želenega toka i_r ima učinek le v ožjem pasu, ki jo določata nastavitvi K_p in K_i tokovnega regulatorja.

Limita tokovnega regulatorja je za zagotavljanje varnosti motorja nastavljena na i_r^{lim} = +/- 1 A, kar je približno +/- 400 kvantov pretvorbe A/D glede na začetno vrednost, ki znaša okoli $i_r^{zač}$ = 3300 kvantov (+/- 15 kvantov). Začetna vrednost $i_r^{zač}$ je določena vsakič ob začetnem mirovanju motorja s povprečenjem desetih zaporednih odtipkov pretvorbe A/D.

Hitrostna regulacija

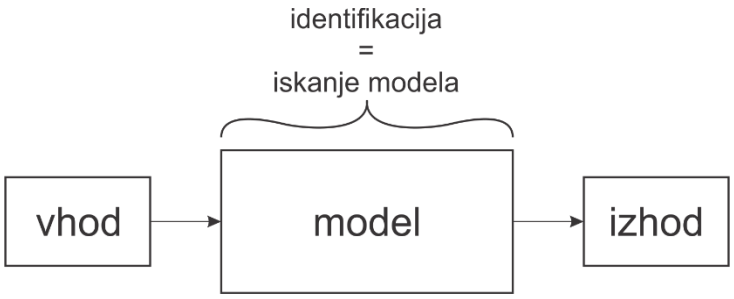
Hitrostni regulator je tipa PI-regulatorja, njegovi konstanti pa znašata K_p = 20 za ojačenje ter K_p = 0.01 za integracijsko konstanto.

Položajna regulacija

Čas tipanja položajne regulacijske zanke je identičen času tipanja hitrostne zanke (T_{sp} = T_{sh}). Zaradi prostega integratorja, ki se pojavlja z merjenjem dejanskega položaja, potrebe po položajnem I-členu ni. Položajni regulator je zato omejen le na ojačenje K_p = 0.1, kar je dodatna poenostavitev.

3.2 Identifikacija

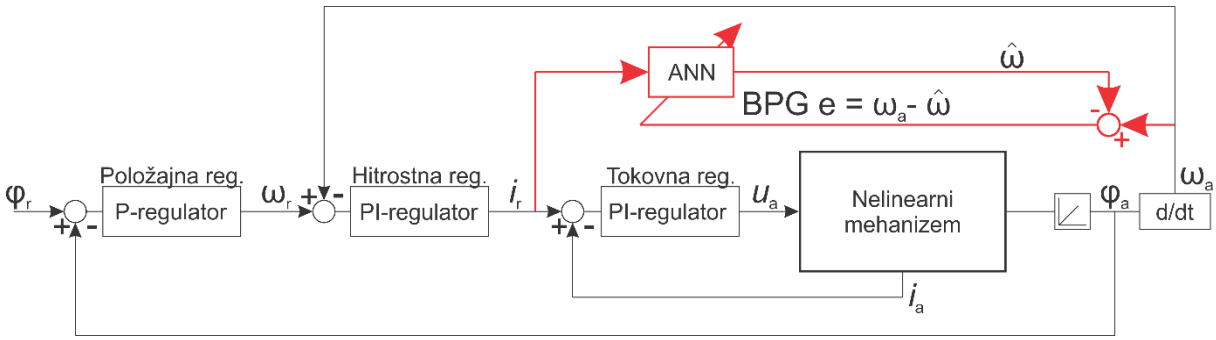
Identifikacija oz. modeliranje je postopek iskanja modela, ki čim bolj verno preslika vsako množico vhodnih spremenljivk v množico izhodnih spremenljivk, kot to prikazuje slika 3.7 [14].



Slika 3.7: Identifikacija oz. modeliranje – iskanje modela

Identifikacija je v naši regulacijski progi izvedena med količinama i_r in ω_a, kot prikazuje slika 3.8. Namen uporabe identifikacije v naši regulacijski progi je zmožnost modeliranja realno-časovne napovedi (angl. prediction) izhoda glede na vhod. Tako vhod, kot izhod se pri identifikaciji v vsakem ciklu spreminjata, zato je govora o dinamični identifikaciji. Poleg računske natančnosti je treba dodatno zagotoviti tudi časovno natančnost in predvidljivost časovnega trajanja učenja.

Identifikacija je izvedena z usmerjeno (angl. feedforward) ANN. Učenje ANN se izvaja s časom tipanja T_{sh} = 5 ms z vzvratnim učnim algoritmom (angl. backpropagation, krajše BPG). Slika 3.8 prikazuje nadgradnjo kaskadne regulacijske sheme z identifikacijo sistema.



Slika 3.8: Nadgradnja kaskadne regulacijske proge

Slika 3.8 vpeljuje novo spremenljivko – ocenjeno hitrost (angl. estimated speed). Ta je pridobljena z izračunom usmerjene transformacije vhodnih parametrov $\hat{\omega}$. Razlika med dejansko in ocenjeno hitrostjo $\omega_a - \hat{\omega}$ znaša t.i. vzvratni pogrešek (angl. backpropagation

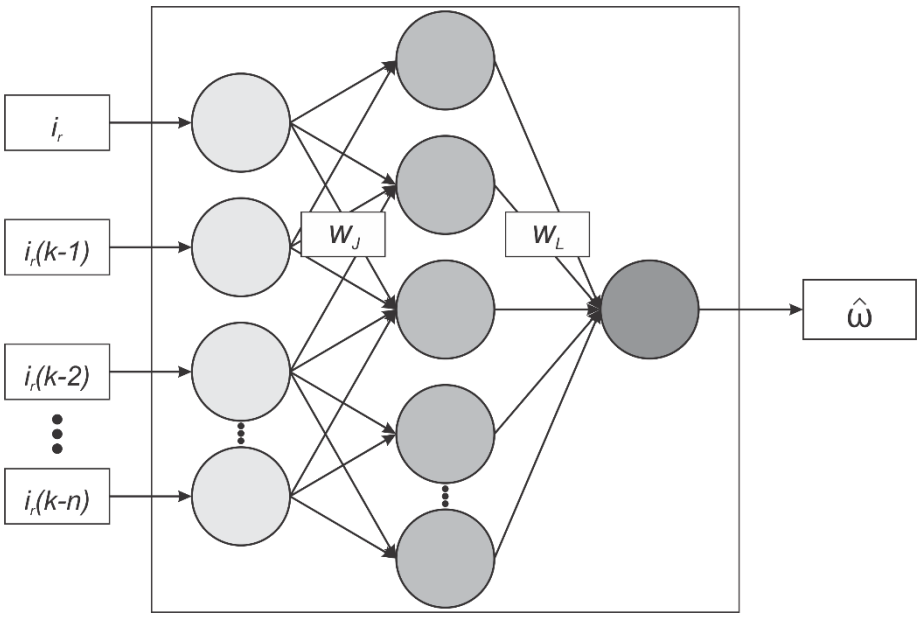
error, krajše BPG e), ki je povod za učenje nevronske mreže. Učni postopek (učenje) je v vsakem času tipanja T_{sh} izveden natanko desetkrat (deset epoh).

Kakovost identifikacije je v veliki meri odvisna od začetnih, inicializacijskih uteži. Te zagotovimo z začetnim, t.i. deviškim učenjem. ANN po tem dinamično sledi dejanski hitrosti ω_a in v vsakem trenutku zagotavlja dovolj kakovosten model za napovedovanje.

Struktura nevronske mreže

Uporabljena nevronska mreža je časovno zakasnjena (angl. time-delay ANN) [15]. To pomeni, da na vhod ANN nanašamo posamezne odtipke i_r iz prejšnjih ciklov. Ta zasnova omogoča ANN pomnjenje nelinearnosti nekaj ciklov nazaj (učinek filtra oz. vztrajnosti). Ta z večanjem števila vhodnih odtipkov (vhodov v ANN) zmanjšuje variacijo (pojavljanje izhodov okoli neke vrednosti) ocenjene hitrosti $\hat{\omega}$. Vhodov v ANN ne sme biti preveč, saj bi v tem primeru ANN napovedovala zelo majhne, nerealne spremembe $\hat{\omega}$ na izhodu.

Eksperimentalno smo tako ugotovili, da je najprimernejše število vhodnih nevronov oz. odtipkov i_r – enajst (poleg trenutnega še najnovejših deset zgodovinskih odtipkov) in je število nevronov skrite plasti sedem (tabela 3.1). Med njima se raztezajo uteži skrite plasti w_J. Izhodni nevron je izključno en in predstavlja ocenjeno hitrost $\hat{\omega}$. Iz skrite plasti ga preslikujejo uteži w_L.



Slika 3.9: Časovno zakasnjena struktura ANN

Kot prikazuje slika 3.9, odtipke iz prejšnjih ciklov (vhode v ANN) označujemo z notacijo *(k-x)*, kjer *x* pove, kateremu ciklu pripada določen odtipek.

Tabela 3.1: Struktura ANN

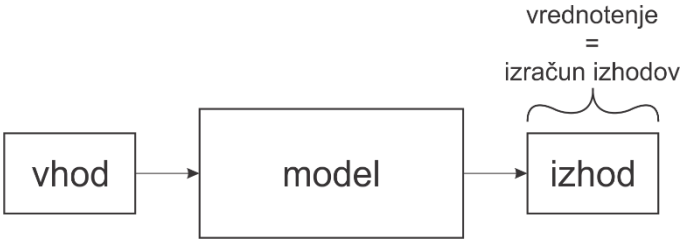
Struktura ANN	Število nevronov
Vhodna plast ANN - i_r	11
Skrita plast	7
Izhodna plast - $\hat{\omega}$	1

Vzvratni učni algoritem nevronskih mrež

Implementacijo vzvratnega učnega algoritma BGP smo povzeli iz [16]. Njegovo delovanje temelji na izračunu pogreška »BPG e« in vzvratnemu prilagajanju uteži. Uporabljeno je nadzorovano učenje s sigmoidno pragovno funkcijo v skriti in izhodni plasti, zato sta vhod in izhod skalirana med nič in ena. Prag (angl. bias) je v vseh primerih nastavljen na $b = 0$.

3.3 Vrednotenje

Vrednotenje (evalvacija) je izračun izhodnih vrednosti iz vhodnih spremenljivk s pomočjo pripadajočega modela [14]. Slika 3.10 prikazuje omenjen princip. Kot vhodne parametre uporabljamo posamezne odtipke i_r, medtem ko transformacijo predstavlja usmerjena ANN.

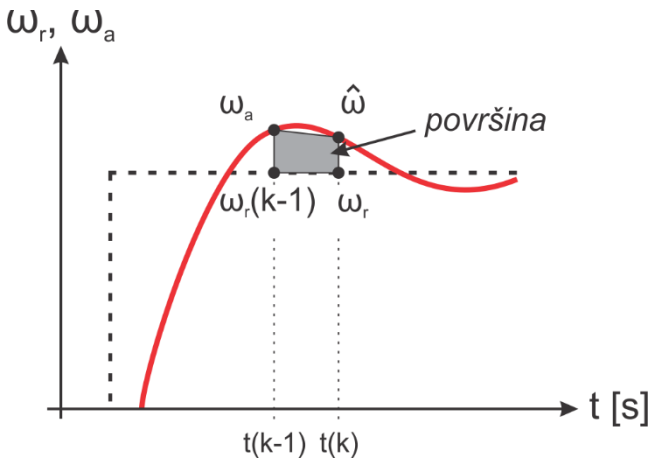


Slika 3.10: Vrednotenje – izračun izhodnih vrednosti

Ker je ANN v vsakem ciklu dovolj naučena, posnema realni sistem in služi kot simulator za preverjanje poskusnih rešitev. Na njej lahko zato ob različnih scenarijih preizkušamo različne vrednosti poskusnih rešitev, kar zaradi izpada stabilnosti na realnem sistemu ni mogoče. Postopek vrednotenja poteka na sledeči način: trenutno stanje skritih in izhodnih uteži

ohranimo, vhodne odtipke pa premaknemo za eno mesto navzdol. Pri tem pridemo do zgodovine rešitev, kjer zadnji odtipek zavržemo, na prvo mesto pa postavimo zadnjo poskusno rešitev optimizacijskega algoritma. Postopek iterativno ponavljamo, dokler vse vhodne spremenljivke niso primerne za napovedovanje vrednosti ocenitvene funkcije.

Na sliki 3.11 je prikazan izračun ocenitvene funkcije poskusne rešitve. Ocenitvena funkcija je nastavljena tako, da v vsakem trenutku zmanjšuje hitrostni pogrešek pri sledenju. V izračunu sodelujejo trenutna ω_r in prejšnja referenca $\omega_r(k-1)$, ocenjena hitrost $\hat{\omega}$ in dejanska hitrost ω_a.



Slika 3.11: Izračun površine

Ocenitvena vrednost je enaka površini med štirimi točkami, namen optimizacije pa je to površino zmanjšati ali izničiti. Iz slike 3.11 vidimo napako ostrih mej izračunanega lika, saj ocenitveno vrednost zaradi hitrejšega procesiranja le aproksimiramo. Napaka, ki nastane pri izračunu je sicer pri dovolj majhnem času tipanja T_{sv} zanemarljiva.

Občutljivost ocenitvene funkcije lahko z upoštevanjem odtipkov iz več prejšnjih ciklov povišamo. S tem povečamo ločljivost napake (napaka je blizu želene vrednosti zelo majhna, npr. en kvant, kar pomeni 50% pogrešek inkrementalnega dajalnika), a upoštevamo več odtipkov v zgodovini rešitev, kar lahko ob menjavi želene vrednosti, zaradi dolgega iznihavanja, neugodno vpliva na vodenje robotskega mehanizma. Prav tako s povišanjem števila odtipkov v zgodovini rešitev dosežemo filtrski učinek, ki lahko zmanjša odzivnost sistema.

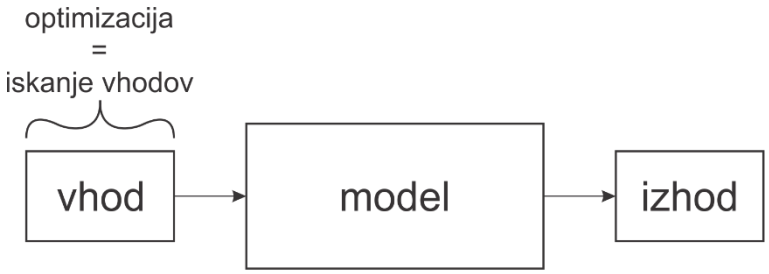
3.4 Optimizacija

Optimizacija je postopek iskanja optimalnih vhodnih parametrov pri znanem modelu in izhodnih vrednosti spremenljivk [14] (slika 3.12). Naš problem optimizacije je poiskati optimalno vhodno rešitev i_r^{trial} tako, da je pogrešek med ω_r in ω_a minimalen.

Ta problem rešujemo z evolucijskimi algoritmi (EA), ki se sestojijo iz naslednjih komponent [17]:

- predstavitve rešitev,
- evolucijskih operatorjev (mutacija in križanje),
- selekcije staršev,
- selekcije preživelih,
- ocenitvene funkcije in
- pogoja ustavljanja.

Problem, ki ga rešujemo na robotskem mehanizmu je dinamične narave, kar pomeni, da se iskalni prostor s časom spreminja – obstaja verjetnost ujetja v lokalni optimum. Izbrani optimizacijski algoritem mora zato izkazovati neobčutljivost na lokalne optimume.

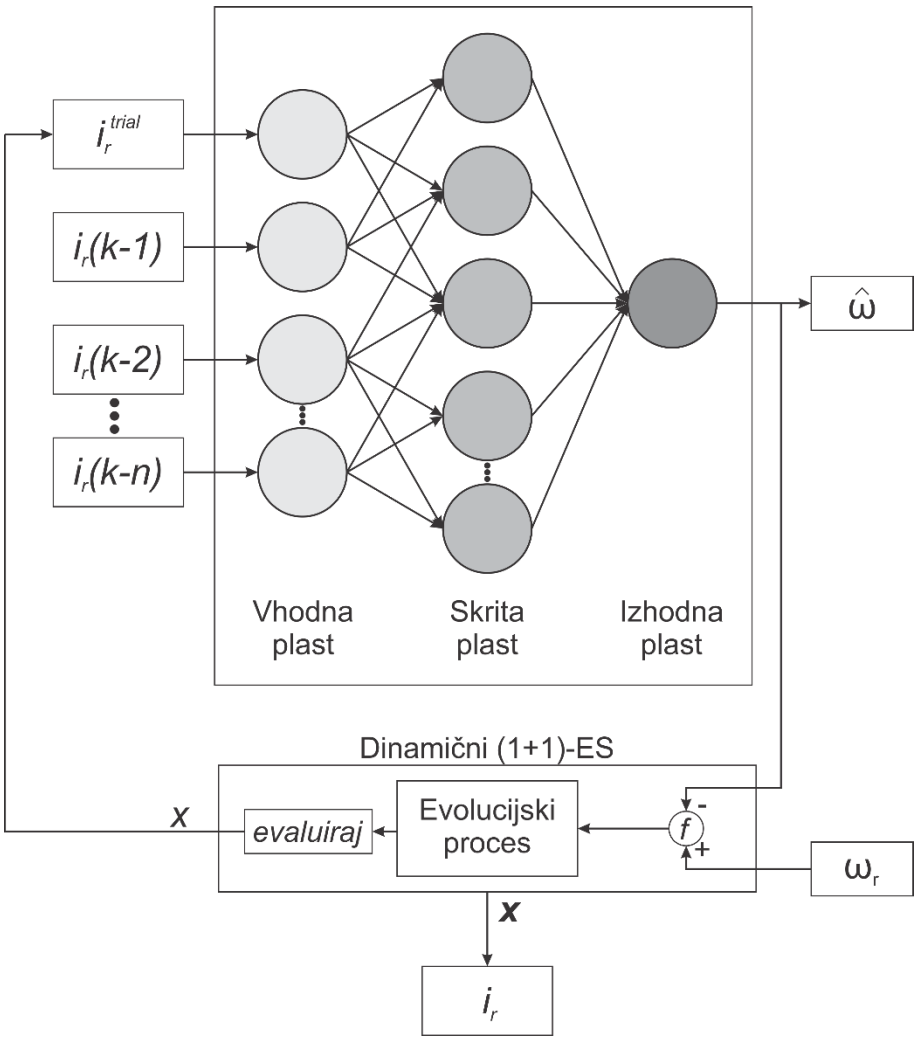


Slika 3.12: Optimizacija – iskanje vhodov

Kot že povedano, se uteži skrite in izhodne plasti ohranijo med procesom optimizacije, vhodi se pomaknejo za eno mesto navzdol, na prvo mesto pa postavimo vrednost poskusne rešitve. Pri vrednotenju dobimo oceno poskusne rešitve, postopek pa se zaključi z ustavitvenim pogojem maksimalnim številom generacij MAX_GEN. Optimizacijski algoritem je zadolžen za spreminjanje poskusnih rešitev tako, da je po koncu optimizacije najdena optimalna ali vsaj

blizu-optimalna poskusna rešitev, katera je uporabljena na tokovnem regulatorju v prihodnjem ciklu. Velja omeniti, da v enem ciklu časa tipanja T_{sv} dvakrat premaknemo vhode v ANN (prvič za učenje ANN in drugič za optimizacijo/evalvacijo).

Slika 3.13 podrobneje predstavlja postopek vrednotenja poskusne rešitve i_r^{trial}. Napovedovanje je ena izmed temeljnih lastnosti in glavna prednost uporabe ANN. Blok »Dinamični (1+1)-ES« predstavlja izvedenko dinamičnega algoritma evolucijskih strategij, ki ga predstavljamo v nadaljevanju.



Slika 3.13: Končna rešitev optimizacije je vektor ***x***

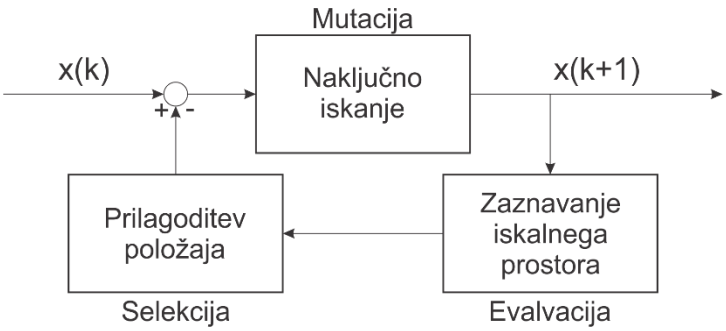
Optimizacijski problem, ki ga ponazarja robotski mehanizem je dinamične narave. Pomeni, da se globalni optimum s časom spreminja, temu pa mora optimizacijski algoritem slediti. Za

uspešno reševanje dinamičnih problemov sta potrebni naslednji dve lastnosti evolucijskih algoritmov:

- zaznati spremembe globalnega optimuma,
- odzvati se na to spremembo.

Prvo lastnost lahko obvladamo v evolucijskem algoritmu s pomočjo uvedbe nadzornega posameznika, katerega vrednost ocenitvene funkcije se spreminja samo ob spremembah globalnega optimuma. Druga lastnost zahteva vzdrževanje večje raznolikosti populacije, ki zagotavlja iskanje v širino in omogoča populaciji v dinamičnih okoljih konvergenco proti novemu lokalnemu optimumu. Zadnja lastnost običajno ne drži za evolucijske algoritme, ki z izboljševanjem ocenitvene funkcije (približevanjem globalnemu optimumu) z večanjem selekcijskega pritiska namreč zmanjšujejo iskalni prostor.

Kakovost trenutnih posameznikov s postopkom optimizacije zato poleg vpliva na njihove potomce vpliva tudi na iskalni prostor. Na sliki 3.14 prikazujemo evolucijski algoritem za dinamična okolja v vlogi zaprtozančnega regulacijskega sistema.



Slika 3.14: Optimizacijska povratna zanka

Pomembna lastnost vsakega algoritma, ki ga uporabimo za problem regulacije v realnem času, je odzivnost. Uporabljeni algoritem mora zato poiskati dobro rešitev dovolj hitro. Dovoljen čas za napredno adaptivno optimizacijo v realnem času je identičen času tipanja T_{sv} = 5 ms.

Primeren predstavnik algoritmov, neobčutljivih na lokalne optimume in z lastnostjo hitre konvergence je algoritem evolucijskih strategij (angl. Evolution Strategies, krajše ES). To je eden izmed družine evolucijskih algoritmov, ki ga je razvil Rechenberg [18], [19]. Tradicionalni

algoritem ES uporablja za globalno preiskovanje prostora izključno populacijo z enim posameznikom, ki ga v vsaki evalvaciji mutira s pomočjo Gaussove perturbacije. Z uporabo tega algoritma lahko dosežemo krajši čas izvajanja.

Osnove evolucijskih strategij

V splošnem je posameznik v algoritmu ES predstavljen kot:

$$\boldsymbol{x}_i^{(t)} = \left(x_{i,1}^{(t)}, ..., x_{i,n}^{(t)} \right)^T, \tag{3.1}$$

kjer sta

i – število posameznikov v populaciji,

n – dimenzija problema.

Predstavljeni vektor lahko nadgradimo s parametrom iskanja v širino δ, ki istočasno določa tudi korak mutacije (enačba 3.2). Ta se spreminja v povezavi s širjenjem/krčenjem iskalnega prostora po enačbi (3.3). Nova poskusna rešitev se izračuna po enačbah (3.3) in (3.4):

$$\boldsymbol{x}_i^{(t)} = (x_{i,1}^{(t)}, ..., x_{i,n}^{(t)}, \sigma_i^{(t)})^T \tag{3.2}$$

$$\sigma_i^{(t+1)} = \sigma_i^{(t)} \cdot e^{\tau \cdot N(0,1)} \tag{3.3}$$

$$\boldsymbol{x}_{i,j}^{(t+1)} = x_{i,j}^{(t)} + \sigma_i^{(t+1)} \cdot N_i(0,1) \tag{3.4}$$

kjer sta:

τ – učna konstanta,

N(0,1) – naključno število po Gaussovi porazdelitvi s srednjo vrednostjo nič in standardno deviacijo ena.

Proces, ki ga opisujeta enačbi (3.3) in (3.4), imenujemo tudi samoprilagajanje (angl. self-adaptation).

Obstajata dva tipična populacijska modela algoritma ES: (μ,λ)-ES in (μ+λ)-ES. Pri prvem modelu generiramo λ potomcev iz populacije μ staršev, pri čemer se najboljših μ potomcev uvrsti v naslednjo generacijo. Pri drugem modelu λ generiranih potomcev tekmuje z μ starši za preživetje. Naš algoritem lahko zapišemo kot (1+1)-ES, kar pomeni, da v vsaki evalvaciji

nastopa le en starš, ki kreira enega potomca, v naslednjo generacijo pa se prenese boljši izmed njiju.

Čeprav algoritem (1+1)-ES odlikujejo majhna časovna kompleksnost, samoprilagodljivost in neobčutljivost na lokalne optimume, za dinamično optimizacijo, zaradi nenehnih sprememb globalnega optimuma, ni primeren. Algoritem smo zato nekoliko modificirali, modificirano izpeljanko pa poimenovali (1+1)-DES, kjer začetnica D predstavlja zmožnost reševanja dinamičnega problema. Slednja je predstavljena v naslednjem podpoglavju.

Dinamična evolucijska strategija (1+1)-DES

Namen (1+1)-DES je povišanje robustnosti preiskovanja prostora rešitev z dodajanjem nadzornega posameznika v originalno populacijo. Ta predstavlja varovalko pred izpadom stabilnosti robotskega mehanizma. Medtem ko se parameter iskanja v širino δ pri originalnem osebku ohranja iz generacije v generacijo, se parameter δ_0 pri nadzornemu posamezniku ponastavi na začetno fiksno vrednost ob vsakem zagonu (1+1)-DES in s tem zagotavlja večjo širino preiskovanja prostora rešitev. Pri tem sta oba posameznika podvržena delovanju operatorjev mutacije in selekcije. Vrednosti ocenitvenih funkcij beležijo spremenljivke f_{trial}, f_{dyn} in f_{best}, rezultat optimizacijskega postopka pa, z najboljšo vrednostjo ocenitvene funkcije f_{best} predstavlja končna rešitev $\boldsymbol{x}$.

Algoritem 1 prikazuje psevdokod algoritma (1+1)-DES.

Algoritem 1 Psevdokod algoritma (1+1)-DES

Vhod: želena hitrost v prejšnjem ciklu $\omega_r(k-1)$, želena hitrost ω_r, dejanska hitrost ω_a in poskusna rešitev $\boldsymbol{x} = \{x, \delta\}$

Izhod: najboljša rešitev $\boldsymbol{x}$ in pripadajoča ocenitvena vrednost

```
1.  y = {x, δ}
2.  z = {x, δ0}
3.  f_best = evaluiraj(ω_r(k-1), ω_r, ω_a, x)
4.  for t = 1 to MAX_GEN do
5.      y = mutiraj(y)
6.      f_trial = evaluiraj(ω_r(k-1), ω_r, ω_a, y)
7.      z = mutiraj(z)
8.      f_dyn = evaluiraj(ω_r(k-1), ω_r, ω_a, z)
9.      if f_dyn < f_trial then
10.         y = z
11.         f_dyn = f_trial
12.     end if
13.     if f_best ≤ f_dyn then
14.         x = y
15.         f_best = f_dyn
16.     end if
17. end for
18. return f_best
```

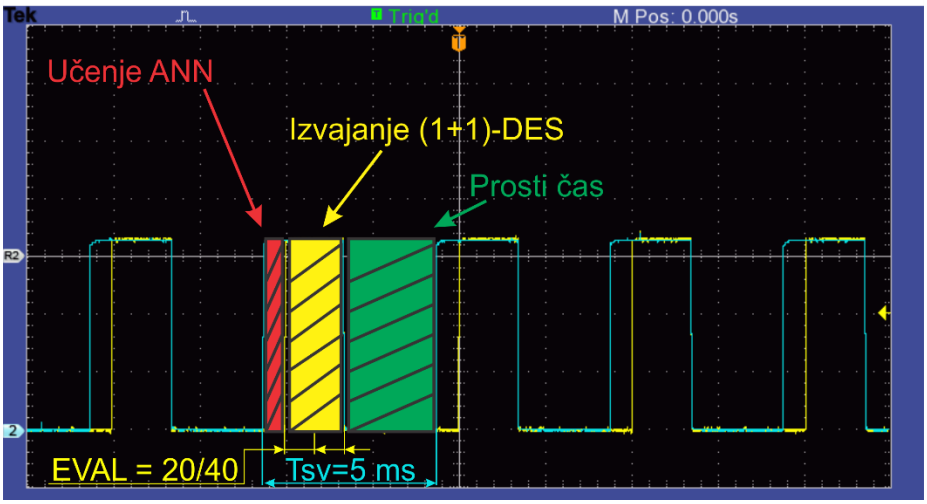
kjer so $\boldsymbol{x}$, $\boldsymbol{y}$ in $\boldsymbol{z}$ vektorji poskusnih rešitev, sestavljeni iz poskusne vhodne vrednosti x, y in z parametra δ oz. δ_0. Vrednost funkcije »evaluiraj« je pridobljena z izračunom ocenitvene funkcije (glej poglavje 3.3 Vrednotenje). Spremenljivka MAX_GEN predstavlja maksimalno število generacij.

3.5 Verifikacija

Izvajanje v realnem času (angl. real-time execution) je pomembna lastnost adaptivnega regulacijskega krmilnika, s katerim zagotavljamo, da se vsa opravila, to so tokovna in položajna regulacija, učenje ANN ter optimizacija z vmesnim ovrednotenjem končajo v predvidenem času (čas tipanja za hitrostno regulacijo T_{sv} = 5 ms). V nasprotnem primeru lahko pride do nestabilnosti regulacijske zanke. V naslednjih treh podpoglavjih predstavljamo realno-časovno analizo, konfiguracijo mikrokrmilnika ter integracijo vseh sklopov.

Realno-časovna analiza

Izvajanje v realnem času dokazujemo z realno-časovno analizo [21]. Čeprav jo dokazujemo analitično, smo dokaz v našem primeru poenostavili, ter ga prikazali z realnimi meritvami. Slika 3.15 predstavlja posnetek realno-časovnega izvajanja z osciloskopa. Prikazani sta dve opravili, učenje ANN in izvajanje (1+1)-DES, iz slike pa je razvidno, da se v periodi T_{sv} = 5 ms vsa opravila končajo znotraj predpisanega intervala, še več, izkaže se, da po izvajanju obeh procesov ostane krmilniku še nekaj prostega časa. Učenje ANN je najmanj časovno zahteven postopek. Nekoliko zahtevnejše je izvajanje (1+1)-DES, pri katerem je izvedena grafična primerjava za dva različna pogoja ustavljanja, tj. MAX_GEN=20 in MAX_GEN=40. Čas tipanja hitrostne regulacije bi lahko, na podlagi predstavljenih rezultatov, zmanjšali na približno T_{sv} = 2 ms, kar bi hitrostno regulacijo bistveno izboljšalo. Lahko pa število generacij podvojimo ali celo potrojimo, s čimer izboljšamo rezultate optimizacije ter tako zagotovimo popolnoma stabilno delovanje v vseh režimih dela. Velja omeniti, da sta opravili za tokovni in položajni regulator zaradi paralelnega izvajanja, ki ne moti hitrostne regulacije, že všteti v realno-časovno analizo, a je njun časovni vpliv na regulacijski algoritem kot celoto zanemarljiv.



Slika 3.15: Realno-časovna analiza

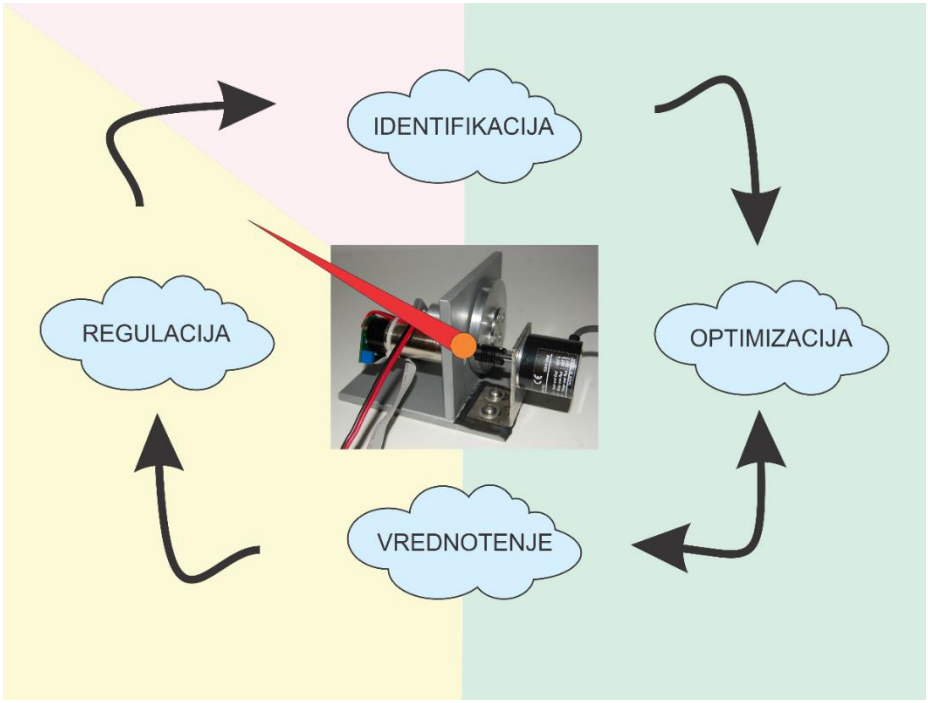
Konfiguracija krmilnika

Oba procesorja si lahko podatke med seboj poljubno izmenjujeta. Glavni program se izvaja na CPU C28x, ki opravi inicializacijo, saj CLA ne more dostopati do vseh registrov. Organizacija glavnega programa je razdeljena v tri programske rutine, ki jih periodično proži enota EPwm. Prva je namenjena za tokovno regulacijo, druga za PI-hitrostno ali NA-OR regulacijo ter tretja

za položajno regulacijo. Vse tri linearne regulacije so izvedene v koprocesorju, zato je glavni procesor popolnoma prost za izvajanje adaptivnega regulatorja NA-OR.

3.6 Integracija vseh opisanih sklopov

Integracija oz. vključevanje pomeni sestavljanje posameznih sklopov (regulacija, identifikacija, vrednotenje, optimizacija, verifikacija) v celoto. Je najpomembnejše delo te magistrske naloge, saj z integracijo dobi celotna magistrska naloge svoj pomen. Integracija je računalniški program, ki vključene sklope medsebojno povezuje in določa prenos podatkov med njimi. Integracija je predstavljena na sliki 3.16, kjer sodeluje vseh pet sklopov. S fizikalnega stališča je identifikacija pravzaprav zaznavanje oz. posnemanje dejanske proge, optimizacija in vrednotenje iskanje optimalne rešitve ter regulacija izkoriščanje najdenih rešitev. Potek računalniškega programa se dogaja periodično v smeri urinega kazalca, ki je predstavljen kot pokazatelj realnega časa. V ozadju je narisana ura, ki v zelenem polju simbolizira več kot polovico prostega časa procesorja, rumena barva od polovice do zadnje četrtine, medtem ko je zadnja četrtina prostega časa roza. Z načrtovanjem izvajanja v realnem času bi bilo treba procesor maksimalno izkoristiti tako, da bi bil v roza področju.



Slika 3.16: Integracija opisanih sklopov z računalniškim programom v obliki ure

4 REZULTATI IN DISKUSIJA

Cilj našega eksperimentalnega dela je bil pokazati kakovost delovanja adaptivnega regulatorja NA-OR v primerjavi s PI-hitrostnim regulatorjem ter pokazati, da je z našim pristopom moč regulacijsko progo linearizirati ter tako izboljšati hitrostne in položajne odzive na enoosnem robotskem mehanizmu. Za dokazovanje smo izvedli dva primerjalna testa obeh algoritmov ter k temu dodali še dodatno testiranje adaptivnega regulatorja NA-OR pri posebnih pogojih (pri povišanem številu generacij). Izvedli smo naslednje teste [16]:

1. test sledenja hitrosti (primerjava adaptivnega NA-OR in PI-hitrostnega regulatorja),
2. test sledenja položaja (primerjava adaptivnega NA-OR in PI-hitrostnega regulatorja),
3. test nemirnosti (angl. restlessness).

Rezultate testiranj prikazujejo grafi, ki jih bomo komentirali v nadaljevanju. V graf smo pri hitrostni regulaciji vključili želeno in dejansko hitrost, kakor tudi njun regulacijski pogrešek. Pri položajni regulaciji smo dodali še želeni in dejanski položaj ter njuno razliko. Med testiranji smo uporabili parametre algoritma (1+1)-ES, kot jih prikazuje tabela 4.1.

Tabela 3.1: Tabela parametrov algoritma (1+1)-DES

Test / Parameter	test 1 (hitrost)	test 2 (položaj)	test 3 (nemirnost)
MAX_GEN	20	20	40
δ_0	50	50	50
T	0.2	0.2	0.2
x_0	3300 (+/- 15)	3300 (+/- 15)	3300 (+/- 15)

V vseh primerih je bil edini pogoj ustavljanja izključno doseganje maksimalnega števila generacij MAX_GEN, medtem ko smo za učenje ANN uporabili maksimalno število epoh. Med testiranji smo zagotovili konstantne pogoje (čase tipanja, konstante tokovnega in položajnega regulatorja), tako za PI-hitrostni regulator, kakor tudi za adaptivni regulator NA-OR. Prav tako smo zagotovili enake fizične začetne pogoje, to pomeni enak položaj robotskega mehanizma ob začetku izvajanja eksperimentov in identično menjavanje želenega položaja φ_r in želene

hitrosti ω_r. Sledenje položaja ter hitrosti smo želeli testirati med dvema skrajnima legama razpona φ = 180°, zato se motnja singularnega položaja mora pojavljati približno na sredini v obe smeri.

Test sledenja položaja in hitrosti smo, zaradi popolnoma zadovoljivih rezultatov, opravili pri znižanem maksimalnem številu generacij, tj. MAX_GEN=20, medtem ko test nemirnosti hitrosti pri MAX_GEN=40. Z nižanjem pogoja ustavljanja na MAX_GEN=20 smo pridobili precej prostega časa, ki bi ga lahko porabili za pogostejšo izvajanje regulacije oz. znižanje časov tipanja (v testiranjih smo poskušali preizkusiti adaptivni regulator NA-OR na najslabšem možnem scenariju).

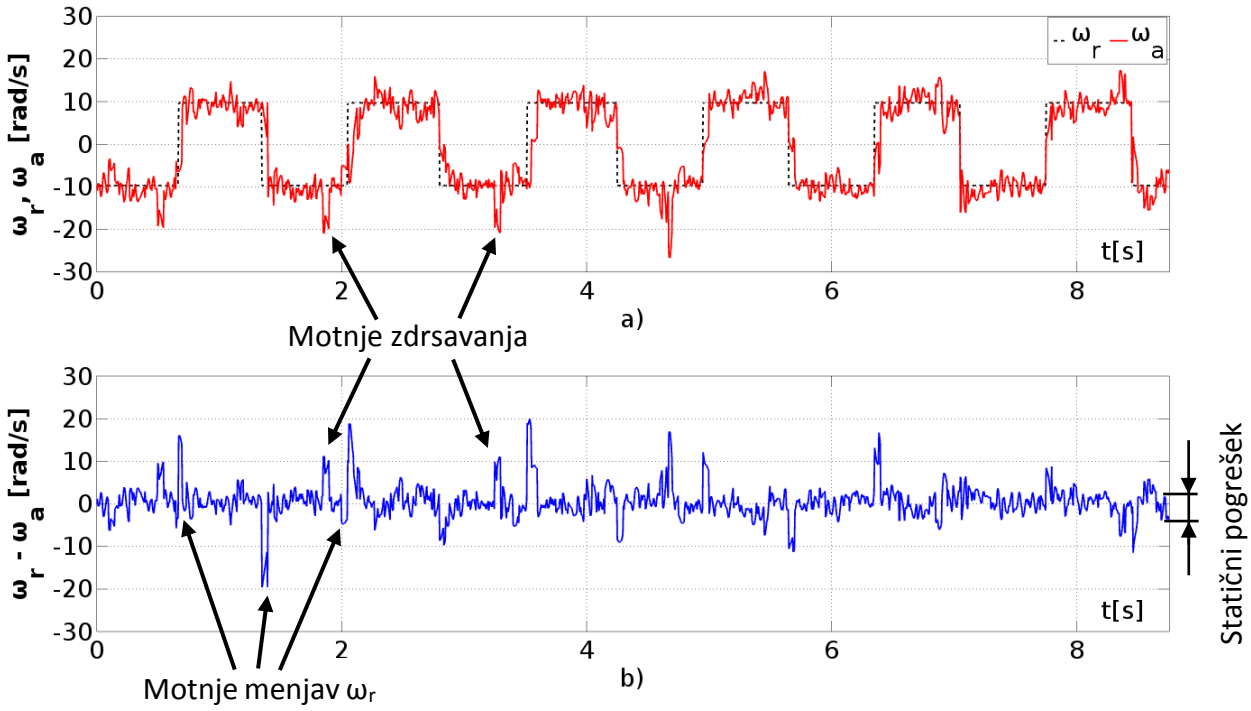
4.1 Test sledenja hitrosti

Rezultate hitrostne regulacije smo posneli z odklopljeno položajno regulacijsko zanko, kar pomeni, da položajna regulacija ne vpliva na delovanje hitrostnega regulatorja. Želena hitrost ω_r je zato konstantna oz. jo menjuje programska rutina, ki se proži periodično skoraj vsako sekundo.

Slika 4.1 prikazuje hitrostno regulacijo z implementiranim adaptivnim regulatorjem NA-OR. Iz hitrostnega odziva in napake pri sledenju lahko opazimo dva tipa motenj – motnje zaradi menjavanja želene hitrosti ω_r (tranzientne motnje) in motnje zaradi zdrsavanja reduktorja v bližini singularne točke. Čeprav so motnje zdrsavanja večinoma konstantne in se pojavlja periodično, se včasih zgodi, da slednje izginejo. Obstajata dve razlagi za izginotje: prva je uspešno učenje in prilagajanje ANN na motnje nelinearnosti, ki jo uspe upoštevati ter odpraviti in druga, ki je bolj naključne narave. Motnje zdrsavanja morajo biti vidne v obeh smereh vožnje robotskega mehanizma, tj. v pozitivni in negativni smeri, saj se zdrsavanje (in singularni položaj) pojavlja v obe smeri. Adaptivni regulator NA-OR beleži majhen statični pogrešek, ki ga v času ene periode ne uspe izničiti. Praktično je to zaradi potrebe po variabilnem navoru na robotski mehanizem, zato lahko sklepamo, da slednjega s trenutno aplikacijo ni mogoče odpraviti.

Če primerjamo odziv adaptivnega regulatorja NA-OR s PI-hitrostnim regulatorjem ugotovimo (slika 4.2), da NA-OR beleži manjši stacionarni pogrešek kot PI-hitrostni regulator. Prav tako se

pri adaptivnem regulatorju NA-OR pojavljajo manjše ali vsaj podobne (a nikoli večje) konice zaradi motenj zdrsavanja in menjavanja želene hitrosti ω_r. Opazimo, da PI-hitrostni regulator pri času tipanja T_{sv} = 5 ms v splošnem slabše sledi želeni hitrosti kot NA-OR, kar pomeni, da slednji uspešno odpravlja večino motenj. Kljub temu v negativni smeri PI-hitrostni regulator kakovostneje odpravlja motnje zdrsavanja. Te na odzivu PI-hitrostnega regulatorja niso niti opazne, zato je PI-hitrostni regulator v tem primeru uspešnejši (kar je tudi edina prednost napram adaptivnemu regulatorju NA-OR).



Slika 4.1: Hitrostna regulacija z NA-OR

4.2 Test sledenja položaja

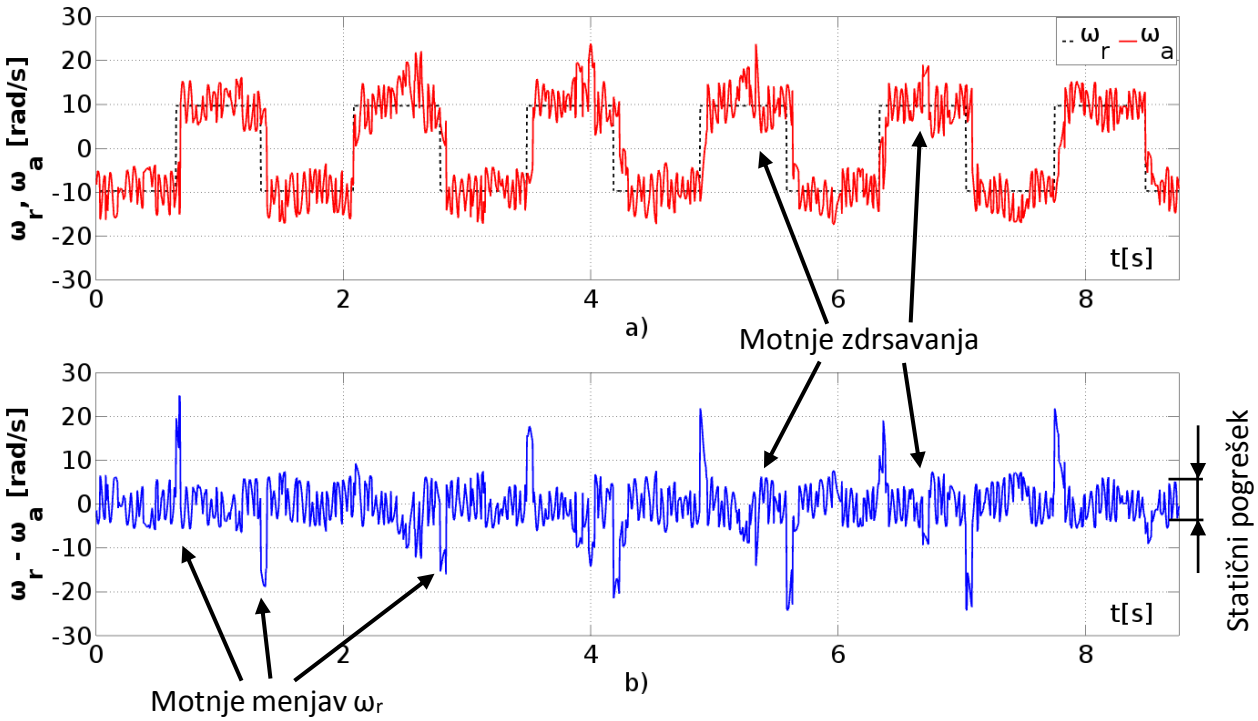
Rezultate položajne regulacije lahko opišemo s slikama 4.3 in 4.4. V tem primeru so aktivne vse tri regulacijske zanke. Programska rutina namesto želene hitrosti ω_r v tem primeru spreminja φ_r, prav tako med dvema fiksnima točkama z vmesnim singularnim položajem.

Iz odzivov je na prvi pogled vidno stabilno in solidno delovanje obeh regulatorjev. Kar se tiče položajne regulacije in pripadajoče napake sta oba posnela približno enako kakovostne rezultate. Položajni statični pogrešek je v obeh primerih blizu nič, je pa treba omeniti dejstvo,

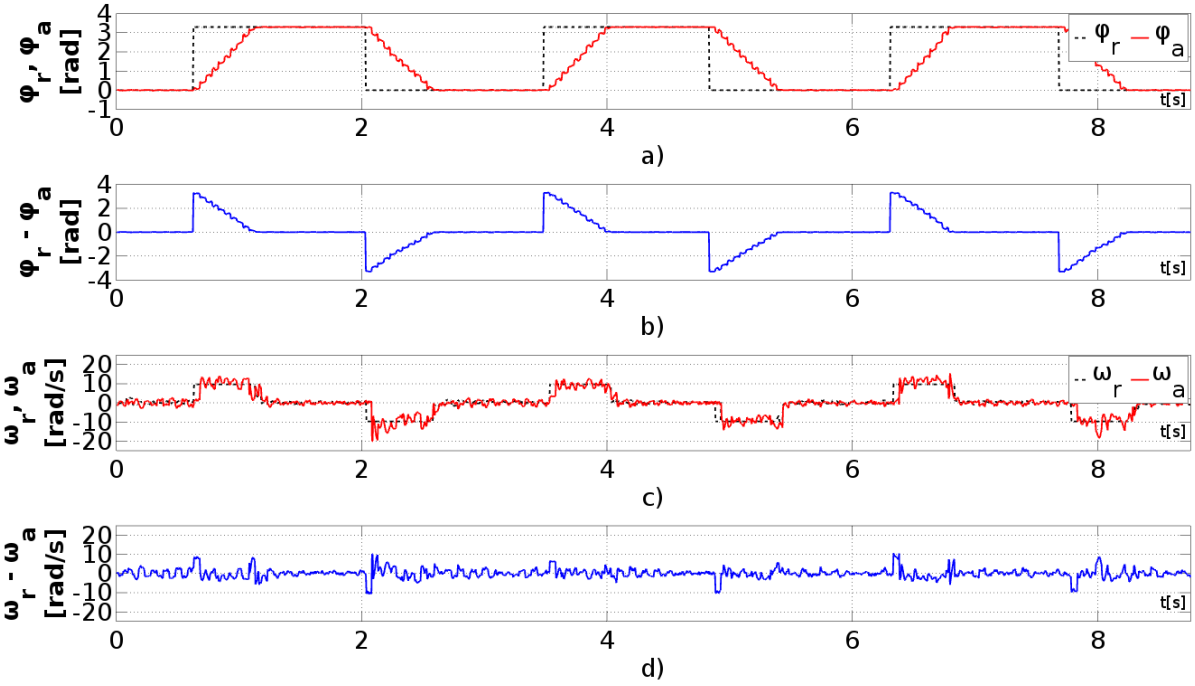
da je ta pri adaptivnem regulatorju NA-OR nekoliko večji. Pomeni, da v položajni zaprti zanki ta nekoliko slabše sledi želeni vrednosti in da za zadovoljivo sledenje hitrosti okoli $\omega_r = 0$ rad/s potrebuje konstantno vzbujanje. Če slednjega ni, NA-OR deluje nemirno in se v točki $\omega_a = 0$ rad/s ne more ustaliti, kar je splošno znani problem regulacij z ANN. Čas sledenja ω_r pri menjavi želene hitrosti je pri obeh algoritmih podoben, zato morebitnih zaključkov iz slednjega ne gre sklepati.

4.3 Test nemirnosti

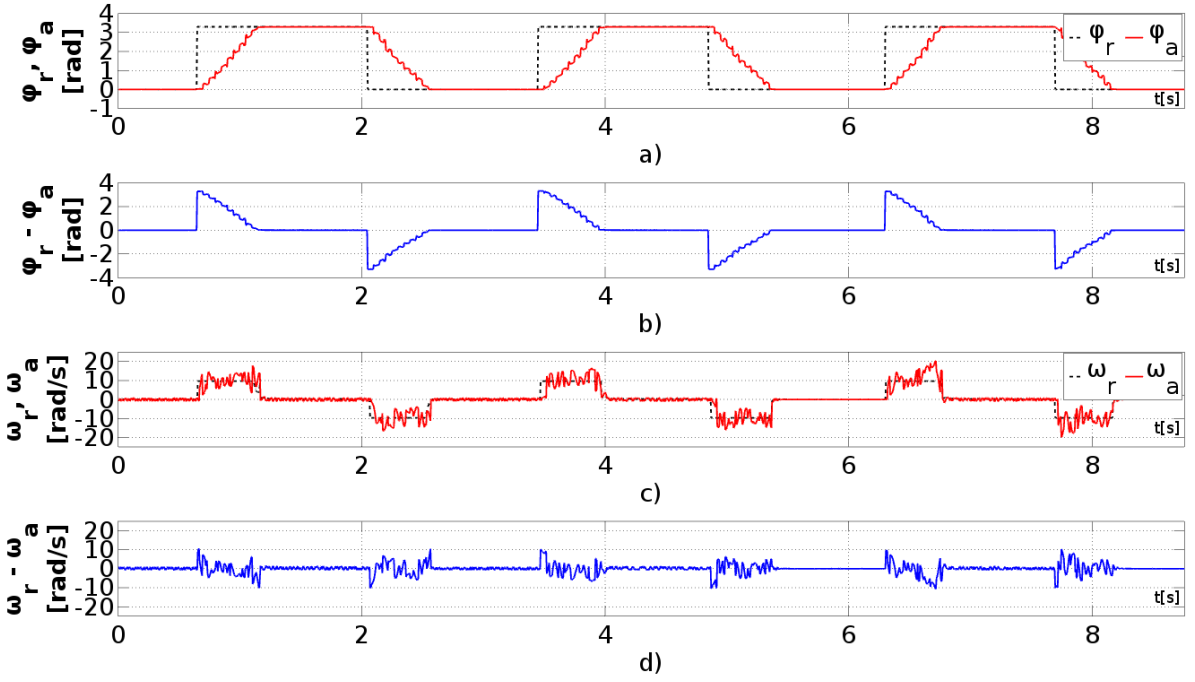
Namen testa nemirnosti je dokazati bistveno izboljšanje sledenja pri želeni hitrosti $\omega_r = 0$ rad/s in izklopljeni položajni regulacijski zanki. Potekal je pri višji vrednosti MAX_GEN, kar pomeni, da je algoritem dlje časa iskal globalni optimum, zato se mu je lahko bolj približal. Mehanizem smo vzbujali z ročnim gibanjem in ga odmikali od želene hitrosti, adaptivni regulator NA-OR pa je v vsakem primeru poskrbel, da se je mehanizem aktivno vrnil nazaj na želeno hitrost in tam tudi ostal. Slika 4.5 prikazuje zmanjšano nemirnost oz. drhtenje napram odzivu položajne regulacije. Treba je omeniti, da ni popolnoma odpravljena.



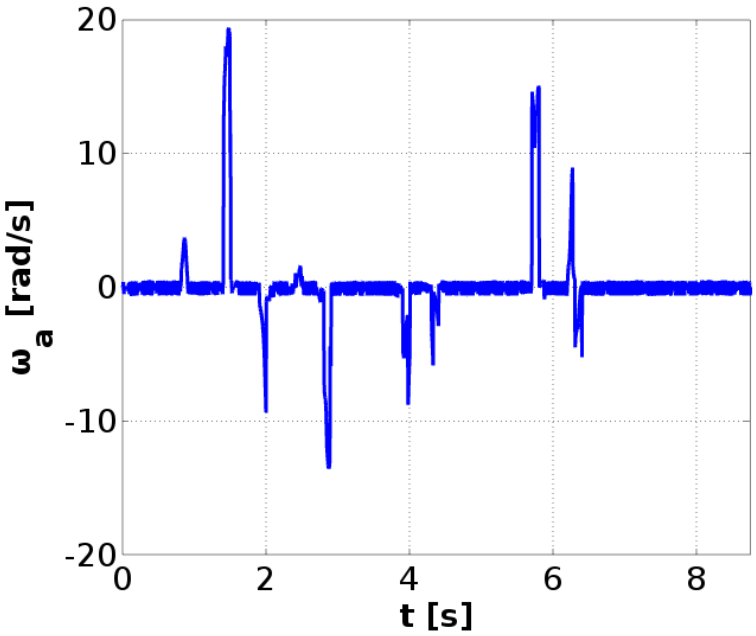
Slika 4.2: Hitrostna regulacija s PI-hitrostnim regulatorjem



Slika 4.3: Položajna regulacija z adaptivnim regulatorjem NA-OR



Slika 4.4: Položajna regulacija s PI-hitrostnim regulatorjem



Slika 4.5: Vzbujanje hitrostnega regulatorja z ročnimi motnjami

4.4 Diskusija

V diskusiji navajamo tegobe, ki se pojavljajo pri testiranju na realnem robotskem mehanizmu. Navedene so iztočnice, s katerimi bi te bilo moč zmanjšati ali celo odpraviti. Glede na to, da gre za »začetno« verzijo adaptivnega regulatorja NA-OR, je v prvi vrsti treba omeniti, da je že delovanje NA-OR kot celote svojevrsten uspeh. Uspelo nam je združiti identifikacijo realne proge ter optimizacijo poskusnih rešitev z napovedovanjem ocenitvenih vrednosti. Praktično smo izdelali svoj tip online regulatorja, ki deluje v realnem času (čas tipanja med 1-5 ms) ter pokazali njegovo stabilnost med delovanjem.

Opazili smo težave pri deviškem učenju ANN, čeprav smo uteži inicializirali iz predhodno opravljenega učenja. Če sistem ni razpadel v fazi zagona, je regulator deloval stabilno dalj časa. Zanesljivost bi bilo moč izboljšati z regularnim kopiranjem trenutnih uteži v trajni pomnilnik flash in njihovim branjem ob zagonu. V regulacijsko progo bi bilo treba vgraditi še eno ANN, ki bi bila daljšega spomina, tj. vsaj dve sekundi, kar ustreza času periode menjavanja referenc. Tako bi bilo treba v procesu regulacije izdelati tudi odločitveno strukturo. Kljub temu je adaptivni regulator NA-OR v vseh testiranjih, razen sledenju $\omega_r = 0$ rad/s pokazal uspešnejše

sledenje želeni vrednosti hitrosti in položaja. Dobljeni rezultati so, ob enakih pogojih, za razred primernejši od PI-hitrostnega regulatorja.

Oteževalno okoliščino je predstavljal dolgi čas tipanja (ki je bil konstanten za oba regulatorja), katerega bi bilo moč vsaj prepoloviti (izvajanje adaptivnega regulatorja NA-OR je časovno dovolj kratko). Delovanje regulacije bi se v tem primeru precej izboljšalo, smo pa želeli posneti rezultate ob najbolj oteževalnih okoliščinah, tj. pri najdaljšem času tipanja in minimalnim številom generacij. S povečevanjem pogoja ustavljanja MAX_GEN smo sicer težavo nemirnosti hitrosti pri majhnih želenih hitrostih zmanjšali, nismo pa ponudili celovite rešitve za odpravo tega problema, kar pomeni, da torej tudi povečanje pogoja ustavljanja v popolnosti ne rešuje te težave. Slednji naj bo omogočal učinkoviteje iskanje globalnega iskanja, saj iskanje poteka dvakrat dlje časa. Poskusne rešitve se lahko zato v tem času bolj približajo globalnemu optimumu.

Alternativna možnost odprave nemirnosti bi bil vklop linearnega regulatorja za majhne želene hitrosti. Nenazadnje bi bilo treba mikrokrmilnik DSC bolje izkoristiti s krajšimi časi tipanja in ne dopustiti, da je bil več kot polovico časa v brezdelju.

Ugotovili smo, da nelinearni mehanizem ni tako nelinearen, kot se zdi. PI-hitrostni regulator ob zmanjšanem času tipanja namreč popolnoma odpravlja vse motnje. Težava, ki nastaja ob spremembah časov tipanja je spreminjanje dejanske hitrosti, ki se meri v enoti odvoda položaja brez upoštevanja časa programske rutine (ločljivost inkrementalnega dajalnika se čedalje bolj manjša). V kolikor se spreminja čas tipanja, to vpliva tudi na interpretacijo hitrosti. Da bi poslabšali regulacijske odzive, bi bilo treba uporabiti nekoliko manj močan komutatorski stroj oz. na reduktor pritrditi utež. Ta bi sicer nudila dodatni filtrski učinek, ki bi nelinearnost motnje zdrsavanja in pospeške ob spremembah želenih vrednosti zmanjševal, povečeval pa vztrajnosti moment.

Izdelali smo vodenje izključno ene prostostne stopnje. Da bi zasnovo omenjenega adaptivnega regulatorja dokazali, bi jo bilo treba testirati na realnem, večosnem robotu. Prepričani smo, da bi tamkajšnje odzive vgrajenih regulatorjev bilo nemogoče izboljšati, saj nenazadnje trenutno vodimo robota glede na rezultate simulacije ANN.

Ne glede na končni razplet vodenja robotskega mehanizma, smo pri tej magistrski nalogi pridobili širša znanja o dinamični identifikaciji sistema in pripadajoči optimizaciji. Prav tako smo izvedeli marsikaj novega o realno-časovnem izvajanju in vrednotenju poskusnih rešitev s pomočjo ANN. V začetku smo v primerjavo želeli vključiti tudi algoritem PSO, vendar smo ugotovili, da je zaradi nepredvidljivega časovnega obnašanja in dolgotrajnega optimizacijskega postopka neprimeren. Katerikoli algoritem že uporabimo na mikrokrmilniku DSC, mora biti ta efektiven, in časovno nezahteven.

Nenazadnje, podrobno smo spoznali tudi mikrokrmilnik DSC družine Delfino. Slednji je eden izmed večjih konkurenčnih upov za namene adaptivne regulacije v prihodnjih letih. Kot študent sem se pri tej nalogi prvič srečal z zbirnim programskim jezikom. Prav tako je bilo zanimivo delo s programskim orodjem Code Composer Studio, ki je razen spontanega snemanja rezultatov v živo, nudil popolno podporo pri razhroščevanju in pomoč pri programiranju z zbirko primerov. V fazi verifikacije smo večkrat uporabili tudi osciloskop Tektronix TBS1052B in razširitveno vezje, na katerega smo namestili vse potrebne komponente. Programskega filtriranja pridobljenih vrednosti vhodov se nismo poslužili.

Smatramo, da bi za povečanje zanesljivosti sistema bilo treba implementirati dodatna orodja. Slednji namreč lahko vsak trenutek, zaradi ene optimizacijske napake izpade iz stabilnosti, kar lahko ima v praksi za posledico morebitno uničenje robotskih mehanizmov. Eden izmed takšnih primerov izpada je hitra spremenitev vztrajnostnega momenta reduktorja.

5 SKLEP

V tej magistrski nalogi smo predstavili zgradbo, delovanje in rezultate lastnega naprednega adaptivnega regulatorja. Uporabili smo princip identifikacije realne regulacijske proge z umetno nevronsko mrežo ter na njej optimizirali poskusne rešitve. Predstavili smo delovanje dinamičnega optimizacijskega algoritma evolucijskih strategij in pokazali izvajanje našega regulatorja v realnem času.

Napredni adaptivni regulator smo testirali na realnem enoosnem robotskem nelinearnem mehanizmu. Pridobljene rezultate smo primerjali s PI-hitrostnim regulatorjem in ugotovili, da napredni adaptivni regulator kakovostneje sledi želeni vrednosti. Rezultati naprednega adaptivnega regulatorja so, razen odpravljanja motnje singularnega položaja, vsaj toliko dobri ali celo boljši od PI-hitrostnega regulatorja. Glavna prednost naprednega adaptivnega regulatorja napram linearnemu je pomnjenje zgodovine in napovedovanje dogajanja v naslednjem ciklu.

Glede na to, da smo izdelali začetno verzijo naprednega adaptivnega regulatorja, je prostora za napredek še veliko. Naš napredni adaptivni regulator bi lahko izboljšali z uvedbo dodatne umetne nevronske mreže za pomnjenje širše zgodovine, ki je v začetku dela še nismo načrtovali, potreba po njej pa je prišla med testiranji.

Prav tako bi bilo treba ugotoviti način za vodenje nelinearnega sistema pri hitrih spremembah vztrajnostnega momenta, npr. pri ročnem zaustavljanju in pospeševanju, kjer se je napredni adaptivni regulator razmeroma slabo odrezal. Čas tipanja bi bilo treba znižati za oba regulatorja, saj se zaradi predolgega čakanja na novo tipanje oba regulatorja bližata delovanju regulatorja »bang-bang«, kar je glavna slabost obeh regulatorjev.

Zagotavljanje realnega časa je bilo, zaradi slabega poznavanja napovedljivosti, v magistrski nalogi najbolj vprašljivo. Dejstvo, ki položaj olajšuje, je čedalje bolj napredna in zmogljiva strojna oprema, pri kateri si lahko privoščimo časovno rezervo. Za minimizacijo računalniškega programa je bilo zelo pomembno usposobiti lastno verzijo programske in strojne opreme. Pričakujemo, da se bodo tovrstni napredni regulatorji v prihodnosti še bolj izoblikovali. Je pa treba omeniti, da slednji praktično ne dovoljujejo večjega uporabniškega poseganja v

regulacijsko logiko, kot npr. mehka logika, kjer lahko uporabnik zelo natančno določi regulacijski potek.

Princip bi lahko testirali z apliciranjem drugih algoritmov, ampak se rezultati verjetno ne bi bistveno spremenili, saj težav s slabim iskanjem globalnega optimuma, sodeč po ugodnih vrednostih ocenjevalne funkcije, nismo imeli.

6 VIRI

[1] H. Demuth, M. H. Beale, O. De Jesus, M. Hagan. Neural network design. Martin Hagan, 2014.

[2] K. S. Narendra, P. Kannan. Identification and control of dynamical systems using neural networks. IEEE Transactions on neural networks 1.1 (1990): 4-27.

[3] H. A. Rowley, B. Shumeet, K. Takeo. Neural network-based face detection. IEEE Transactions on pattern analysis and machine intelligence 20.1 (1998): 23-38.

[4] S. Moshiri, E. C. Norman. Neural network versus econometric models in forecasting inflation. (1999).

[5] A. Cochocki, R. Unbehauen. Neural networks for optimization and signal processing. John Wiley & Sons, Inc., 1993.

[6] I. Fister Jr., X. S. Yang, I. Fister, J. Brest, D. Fister. A brief review of nature-inspired algorithms for optimization. Elektrotehniški vestnik, ISSN 0013-5852. 2013, vol. 80, no. 3, p. 116-122.

[7] D. Fister. Načrtovanje samonastavljivega regulatorja 2 DOF robota s pomočjo BA algoritma. Diplomsko delo, Univerza v Mariboru, Fakulteta za strojništvo. Maribor, 2015.

[8] D. Gjura. Samonastavljivi PD regulator za DC/DC pretvornik navzdol osnovan na algoritmu teorije rojev delcev. Magistrsko delo, Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko. Maribor, 2016.

[9] J. M. Johnson, R. Yahya. Genetic algorithm optimization and its application to antenna design. Antennas and Propagation Society International Symposium, 1994. AP-S. Digest, vol. 1. IEEE, 1994.

[10] L. Frank, J. Campos, R. Selmic. Neuro-fuzzy control of industrial systems with actuator nonlinearities. Society for Industrial and Applied Mathematics, 2002.

[11] B. Brečko. Načrtovanje vodenja nelinearne regulacijske proge z adaptivnim regulatorjem, zasnovanim na kombinaciji mehke logike in gradientne tehnike

optimizacije. Magistrsko delo, Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko. Maribor, 2016.

[12] D. Dolinar, G. Štumberger, M. Milanovič, Z. Maljković. Modeliranje in vodenje elektromehanskih sistemov. Založniška dejavnost FERI, 2006.

[13] Texas Instruments. TMS320F2837xS Delfino Microcontrollers. Texas Instruments, 2015.

[14] A. E. Eiben, J. E. Smith. Introduction to evolutionary computing. Vol. 53. Heidelberg: Springer, 2003.

[15] A. P. Engelbrecht. Computational intelligence: an introduction. John Wiley & Sons, 2007.

[16] R. Šafarič, A. Rojko. Inteligentne regulacijske tehnike v mehatroniki. Fakulteta za elektrotehniko, računalništvo in informatiko, 2007.

[17] S. Russell, P. Norvig. A modern approach. Prentice-Hall, Englewood Cliffs 25 (1995): 27.

[18] I. Rechenberg. Evolutionsstrategien. Simulationsmethoden in der Medizin und Biologie. Springer, Berlin, Heidelberg, 1978. 83-114.

[19] M. Eigen. Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologishen Evolution. Frommann Verlag, Struttgart-Bad Cannstatt 45 (1973): 46-47.

[20] T. Bäck. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford university press, 1996.

[21] M. Colnarič. Procesi in opravila. Študijska literatura za predmet Vgrajeni sistemi. Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko. Maribor, 2017.