SVEUČILIŠTE U RIJECI

**TEHNIČKI FAKULTET**

Diplomski sveučilišni studij strojarstva

Programski zadatak iz kolegija: Upravljanje mehatroničkim sustavima

**Nadogradnja, umjeravanje i implementacija mjernog sustava dinamometra i sEMG osjetnika**

Studenti: **Ivan Britvić**

**Krunoslav Marenić**

**Mato Štefanac**

**Luka Zvonarek**

**Luciano Kostelac**

**Sadržaj**

[1. Uvod 3](#_Toc128310682)

[2. Kalibracija dinamometra 5](#_Toc128310683)

[2.1. Vernier go direct dinamometar [3] 7](#_Toc128310684)

[2.2. Norma ASTM E74:2018 [3] 8](#_Toc128310685)

[2.3. Postupak kalibracije [3] 9](#_Toc128310686)

[3. Shimmer3 senzori 12](#_Toc128310689)

[3.1. A/D prikupljanje podataka 14](#_Toc128310690)

[4. Planiranje i provođenje eksperimentalnih ispitivanja 16](#_Toc128310691)

[4.1. Odabir pozicije elektroda 16](#_Toc128310692)

[4.2. Postupak ispitivanja 18](#_Toc128310693)

[4.3. Nasumični blok plan eksperimenta 19](#_Toc128310694)

[5. Obrada podataka 21](#_Toc128310695)

[5.1. Obrada ROS izlaza [13] 24](#_Toc128310696)

[5.2. Optimiranje parametara EMG signala [13] 28](#_Toc128310697)

[6. Analiza podataka RBD esperimenta 33](#_Toc128310698)

[7. Zaključak 35](#_Toc128310701)

[8. Literatura 37](#_Toc128310702)

[9. Popis slika 38](#_Toc128310703)

[10. Kod 1](#_Toc128310704)

[11. Prilog 13](#_Toc128310705)

# Uvod

Napredak tehnologije omogućio je korištenje novih metoda rehabilitacije osoba koji imaju problem s funkcionalnosti udova zbog nesreća, moždanog ili srčanog udara. Korištenjem senzora, tehnike 3D printa i regulacije moguće je poboljšati proces rehabilitacija. U ovome seminaru opisati će se rad na projektnom zadatku dobivanja rezultata koji će u budućnosti pridonijeti optimizaciji procesa rehabilitacije.

U okviru projektnog zadatka, nadograđuje se i implementira mjerni sustav dinamometra i sEMG osjetnika, za čije se potrebe koristiti Vernier ručni dinamometar za mjerenje sile stiska, Shimmer3 senzori za mjerenje aktivacije mišića, ROS (robotski operacijski sustav), gotovi paketi godirect\_ros i shimmer\_ros za upravljanje, komunikaciju i obradu podataka dobivenih pomoću dinamometra i senzora.

Dinamometar i sEMG osjetnici povezuju se s računalom pomoću bežične veze (Bluetooth), a za međusobnu komunikaciju računala s dinamometrom i shimmer-om zaslužan je ROS. Gotovi paketi omogućili su prikaz, rad i praćenje podataka, no bilo je potrebno i obraditi dobivene podatke kako bi se dobila korelacija između snage stiska i aktivacije mišića pa su se kodovi u gotovim paketima naknadno uređivali, a napisane su i posebne skripte za obradu tih podataka.

Kako bi se dobila prava sila na dinamometru bilo je potrebno provesti kalibraciju. Podaci dobiveni kalibracijom su se obradili, a linearnom regresijom su dobiveni koeficijenti potrebni za definiranje jednadžbe dinamometra u kodu kako bi dobili ispravne vrijednosti sile stiska.

Za dobivanje korelacije prvo su se odredile povoljne pozicije elektrode na podlaktici, a dvije pozicije elektrode odabrane su kao one s kojih će se prikupljati podaci. Odabir pozicije elektrode bitan je radi dobivanja kvalitetnih podataka i kako bi se eksperimentalno uklonio utjecaj pozicije elektrode u ovisnosti o subjektu. Iz tog razloga izvršiti će se nasumični statistički eksperiment. Nasumični statički eksperiment (engl. *randomized block desing -* RBD) je statistički eksperiment gdje se eksperimentalne jedinice dijele u grupe zvane blokove. Kod ovih eksperimenata jedna je varijabla od primarnog interesa dok su ostale varijable smetnja [16]. Provesti će se ispitivanje nad 10 subjekata, 2 različite pozicije, a za svaku poziciju izvršit će se tri ponavljanja.

Na kraju su dobiveni obrađeni podaci, te maksimalne korelacije između sile stiska i aktivacije mišića.

U prvom poglavlju je napravljen uvod u projekt u kojemu je navedena oprema korištena za provođenje mjerenja (dinamometar, sEMG osjetnici). Opisuje se način spajanja računala i opreme. Ukratko je pojašnjeno što je sve obrađeno u slijedećim poglavljima. U drugom poglavlju detaljno je opisani postupak provođenja kalibracije dinamometra. Time se određuje donja granica sile (engl. *low force limit* - LFL) koji je pokazatelj dobre kalibriranosti mjernog uređaja. Cijeli taj postupak izrađen je u skladu sa normom br. ASTM E74:2018 [14]. U trećem poglavlju opisan je Shimmer3 senzor koji služi za mjerenje površinske elektromiografije (engl. *surface Electromiography –* sEMG). Površinska elektromiografija je tehnika za procjenu električne aktivnosti skeletnih mišića te će se koristiti u ovome projektu.. Prikazane su tehničke karakteristike senzora te je objašnjen princip rada senzora. Na kraju poglavlja vidljivo je zašto je važno odabrati optimalnu brzinu uzorkovanja podataka. U četvrtom poglavlju objašnjeno je prikupljanje podataka pomoću sEMG (engl. *surface electro myography*) Shimmera, utjecaj pozicije elektroda te ostali bitni faktori. Zatim je definiran nasumični statistički eksperiment te opisan postupak ispitivanja. U petom poglavlju opisani su najčešće korišteni algoritmi za obradu sEMG signala, uređivanje i usklađivanje podataka u .csv datoteci te obrada podataka. Zatim je objašnjena brza Furieova transformacija koja je korištena za pretvorbu signala iz vremenske domene u frekvencijsku domenu. U šestom poglavlju prikazana je analiza podataka RDB (engl. *Randomized Block Design*) eksperimenta. Izvršena je usporedba maksimalnih korelacija između dviju pozicija elektrode koje su dobivene ispitivanjem na deset sudionika. Rezultati dobiveni mjerenjem prikazani su na kraju seminara u *Prilogu*.

# Kalibracija dinamometra

Osnovni princip kalibracijskog postupka je odrediti odnos između primijenjene sile i otklona instrumenta za mjerenje sile. Da bi se odredila ovisnost sile i otklona instrumenta primjenjuju se poznate sile i mjere se otkloni. Time se smanjuje mjerna nesigurnost dinamometra. Mjerna nesigurnost je definirana kao parametar pridružen rezultatu mjerenja koji opisuje rasipanje vrijednosti. Što je veća mjerna nesigurnost uređaja to je lošije mjerenje. Postupak kalibracije ili umjeravanje dinamometra izrađeno je prema normi br. ASTM E74:2018 [14].

Kako bismo izvršili očitanja sile potrebno je poznavati formulu za izračun iste. Uz poznatu masu, sila utega se izračunava prema izrazu:

(2.1)

Gdje je:

*m -* masa utega [kg]

*g -* gravitacijsko ubrzanje [m2/s]

*d -* gustoća zraka [Mg/m3]

*D -* gustoća težine [m3/kg]

Prije početka mjerenja morali smo odraditi „vježbanje“ instrumenta za mjerenje sile. To se izvodi tako da se pritisne sa što većom silom, poželjno je da se optereti dinamometar približno maksimalnom silom koju može podnijeti, najmanje dva puta u slijedu kalibracijskih sila. Na taj način ponovno uspostavljamo histerezni uzorak koji zbog razdoblja neuporabe može nestati. Pravilnim odabirom utega može se osigurati adekvatni uzorak cijelog raspona.

|  |  |  |
| --- | --- | --- |
| Težina utega [kg] | Klasa točnosti | Preciznost [g] |
| 0,1 | F1 | 0,0005 |
| 0,2 | F1 | 0,001 |
| 0,5 | F1 | 0,0025 |
| 2 | M1 | 0,1 |
| 5 | M3 | 2,5 |
| 10 | M3 | 5 |
| 20 | M3 | 10 |

Nakon početnog „vježbanja“ primjenjuju se kalibracijske sile od 5 N, 20 N, 50 N, 100 N, 150 N, 200 N, 250 N, 300 N, 350 N, 400 N, 450 N, 500 N i 550 N. Tijekom postupka umjeravanja korišteni su utezi težine 0,1 kg, 0,2 kg, 0,5 kg, 2 kg, 5 kg, 10 kg i 20 kg. U tablici 2.1 prikazana je tolerancija utega [17].



Tablica 2.1 Tolerancije utega [17]

Važno je da se sile polako primjenjuju bez izazivanja udara ili vibracija na instrument za mjerenje. Trajanje između uzastopne upotrebe sile i otklanjanja sile sa očitanjem vrijednosti instrumenta moraju biti što ujednačenije. Sam proces primjenjivanja različitih kalibracijskih sila na instrument za mjerenje sile može se vidjeti na sljedećim slikama.

Prilikom kalibracije dinamometra koristili smo za pričvršćivanje uređaja dva držača koji se stegnu vijcima za stol. Držači su izrađeni od PLA te dobiveni FDM tehnologijom 3D ispisa. Držač je ispisan na Flashforge Creator 3 3D pisaču.

|  |  |
| --- | --- |
|  |  |
| a) | b) |
|  |  |
| c) | d) |

Slika 2.1. a) kalibracijska sila od 20 N b) kalibracijska sila od 50 N c) kalibracijska sila od 150 N d) kalibracijska sila od 550 N

## Vernier go direct dinamometar [3]

Ručni dinamometar (Go Direct Hand Dynamometer) predstavlja uređaj za mjerenje sile. Koristimo ga prilikom mjerenja snage stiska te za analizu aktivacije ili zamora mišića. Potrebno je kalibrirati uređaj u rasponu od 0 N do 550 N kako bi odredili pouzdanost dinamometra. Uređaj će se kalibrirati prema normi SASO ASTM E74: 2018. Raspon mjerenja uređaja je od 0 N do 600 N s razlučivošću od ±0.05 N koju smo dobili u ROS-u. Tvornička razlučivost uređaja iznosi ±0.1 N [18]. Uspoređujući sa dobivenom razlučivošću u ROS-u, vidimo je da tvornička razlučivost lošija. U njemu se također nalazi troosni akcelerometar te troosni žiroskop. Na uređaj se može spojiti preko Bluetooth tehnologije ili izravno preko USB sučelja. Za komunikaciju s uređajem mogu se koristiti mobilne platforme kao što su Android i iOS. Također se može komunicirati s uređajem izravno preko Pythona i Javascripta. U ovom projektu za komunikaciju između računala i dinamometra korišten je *Godirect\_ros* paket. Ovaj paket omogućuje praćenje i očitavanje podataka dobivenih stiskom dinamometra u realnom vremenu. To je već gotov paket koji se nalazi unutar ROS-a. Slika 3.2. prikazuje uređaj kojeg je potrebno kalibrirati.

.

Slika 2.2 Vernier go direct dinamometar

Pretpostavka prije samog mjerenja stiska šake je da dinamometar dobro mjeri. Ne mora biti uvijek točno kalibriran, zato se i vrši sama kalibracija instrumenta. Za izvođenje postupka kalibracije s utezima potrebno je uzeti u obzir pravilno držanje dinamometra te ga je potrebno dobro pričvrstiti za stol kako bi dobili što preciznije podatke. Važno je istaknuti da se mjerna ploča okrene suprotno od stola.

## Norma ASTM E74:2018 [3]

Da bi rezultati ispitivanja bili pouzdani instrumenti se moraju kalibrirati prema točno određenim normama i procedurama. Naime, svako mjerenje, tj. ispitivanje u cilju određivanja nekog parametra nikada nije apsolutno točno. Međutim, uvijek postoji sumnja u vrijednosti do kojih se dolazi mjerenjem. Ta ograničenost se izražava putem mjerne nesigurnosti.

Mjerna nesigurnost se definira kao parametar pridružen rezultatu mjerenja koji opisuje rasipanje vrijednosti koje bi se razumno moglo pripisati mjerenoj veličini.

Prema normi ASTM E74:2018 pokazatelj dobre kalibriranosti mjernog uređaja je donja granica sile (engl. *lower force limit*) – LFL. LFL je faktor koji definira iznad koje izmjerene vrijednosti smo sigurni da uređaj ne griješi više od određenog postotka. LFL se računa kao:

|  |  |  |
| --- | --- | --- |
|  |  | (2.2) |

Gdje je:

*LFL* - donja granica sile [N],

*U* - mjerna nesigurnost (engl. *uncertainty*), računa se kao 1,96 puta dobivena

standardna devijacija za provedena mjerenja, i time obuhvaća 95% podataka,

*P* - pogreška, u ovom slučaju 0,25 (klasa A).

Obzirom da je norma bila prestroga, tj. za provedena mjerenja dobiven je daleko prevelik LFL, korišteni su modificirani faktori.

N

Greška za sile iznad 62 N iznosi maksimalno do 4% izmjerene vrijednosti, dok za sile ispod 62 N su puno veće. Najveća greška pri manjim silama iznosi 10,5 % izmjerene vrijednosti. To je dobiveno mjerenjem kalibracijske sile od 5N.

## Postupak kalibracije [3]

Kalibracijski postupak provodi se sa rastućim ili padajućim silama. U našem projektu primijenili smo rastući niz sila kojim se opterećuje dinamometar. U nekim slučajevima kalibracija zahtijeva oba načina. Tada se izrađuje kalibracijska jednadžba za oba slučaja.

Prema normi moramo uzeti u obzir temperaturu prostorije koja je preporučena da bude 23°C. U slučaju da se provodi na drugim temperaturama potrebno je izvršiti dodatne korekcije. Primjenom duže serije primijenjenih sila na instrument donosi određenu pogrešku. Ako kalibracijsku silu prati druga kalibracijska sila manje vrijednosti prije same primjene potrebno je smanjiti primijenjenu silu na instrument za mjerenje sile na nulu. Za svaki povratak na nulu (primjenom rastuće ili padajuće sile) mora slijediti primjena najveće kalibracijske sile prije primjene iduće kalibracijske sile.

Očitanje kalibracijskih sila izvodi se na dva načina. Prvi način je da otklon izračunavamo kao razliku između otklona pri primijenjenoj sili i početnog otklona pri nultoj sili. Drugi način je da se za izračun otklona koristi prosjek dviju nultih vrijednosti.

Kalibracija mjernog instrumenta vršila se na način da se je mjerni instrument opterećivao utezima od 0,5 kg do 55 kg. Prije svakog mjerenja utega, bilježila se sila koju je dinamometar pokazivao bez opterećenja. Kao pravi rezultat mjerenja, odnosno stvarni otklon, onda se uzimala razlika očitane sile s utegom i prethodne sile dinamometra bez opterećenja. Time se nastojalo izbjeći grešku u sili koju dinamometar prikazuje bez opterećivanja. Mjerenje svakim utegom provedeno je tri puta. Mjerenjem dobiveni podaci prikazani su u *Prilogu*.



Na skupu dobivenih podataka (stvarni otklon) provedena je polinomna regresija te je dobivena kalibracijska jednadžba četvrtog reda koja glasi:

Na slici 3.3 grafički su prikazane izmjerene vrijednosti te aproksimacija pravcem.

Slika 2.3 Jednadžba kalibracije - graf polinomne aproksimacije

Da bi se kasnije mogli provoditi eksperimenti s kalibriranim uređajem, kalibracijsku jednadžbu je bilo potrebno implementirati u “godirect\_ros” paket. Na nekoliko mjesta unutar godirect\_ros datoteka “gdx\_class.py”, “godirect\_hand\_dynamometer\_config.yaml” i “godirect\_publisher.py”, bilo je potrebno dodati linije koda koje opisuju parametre kalibracijske jednadžbe dinamometra. Na kraju se dobije “config” sučelje gdje se koeficijenti kalibracijske jednadžbe četvrtog reda („lin\_coeff“, „square\_coeff“, „cubic\_coeff“, „fourth\_ord\_coeff“) mogu mijenjati.

U “godirect\_hand\_dynamometer\_config.yaml” datoteci zadaju se koeficijenti kalibracijske jednadžbe četvrtog reda. Da bi se to postiglo dodane su sljedeće linije koda (linije 26, 27, 28, 29):

|  |  |
| --- | --- |
| lin\_coeff | 1,095545806 |
| square\_coeff | -0,000302451 |
| cubic\_coeff | 0,00000141565 |
| fouth\_ord\_coeff | -1,73619x10-9 |

Tablica 2.2 Koeficijenti kalibracijske jednadžbe četvrtog reda

U “ gdx\_class.py ” datoteci definirana je funkcija „read“ koja kao argumente koristi koeficijente kalibracijske jednadžbe četvrtog reda. Ukoliko funkcija ne dobije parametre, sila („force\_data.grip\_force“) će biti izračunata pomoću unaprijed definiranih koeficijenata, odnosno sila će ispasti jednaka varijabli x (svi koeficijenti su jednaki 0, osim linearnog koji iznosi 1). Ukoliko funkcija dobije parametre zadane u .yaml datoteci, sila se izračunava prema definiranom izrazu. U “ gdx\_class.py ” dodane su sljedeće linije koda.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **def** read(self, publisher, measurement\_type='grip'  lin\_coeff=1  square\_coeff=0  cubic\_coeff=0  fourth\_ord\_coeff=0):  x = (value - sensor\_offset)  force\_data.grip\_force = lin\_coeff \* x + square\_coeff \* (x\*\*2) + cubic\_coeff \* (x\*\*3) + fourth\_ord\_coeff \* (x\*\*4) |

|  |  |
| --- | --- |
| 1  2  3  4  5 | lin\_coeff = rospy.get\_param('~lin\_coeff')  square\_coeff = rospy.get\_param('~square\_coeff')  cubic\_coeff = rospy.get\_param('~cubic\_coeff')  fourth\_ord\_coeff = rospy.get\_param('~fourth\_ord\_coeff')  gdx\_hd.read(publisher=pub, measurement\_type=measurement\_type, lin\_coeff=lin\_coeff, square\_coeff=square\_coeff, cubic\_coeff=cubic\_coeff, fourth\_ord\_coeff=fourth\_ord\_coeff) |

Pomoću “ godirect\_publisher.py ” datoteke pozivaju se koeficijenti koji su definirani u .yaml datoteci. U “ godirect\_publisher.py ” dodane su sljedeće linije koda.



|  |  |
| --- | --- |
| 1  2  3  4 | lin\_coeff: 1.095545806  square\_coeff: -0.000302451  cubic\_coeff: 0.00000141565  fourth\_ord\_coeff: -0.00000000173619 |

# Shimmer3 senzori

Shimmer senzori su nosivi senzori koji služe za praćenje, mjerenje i spremanje podataka o kinematičkoj i biofizičkoj aktivnosti osobe ovisno o konfiguraciji. Napravljeni su da budu lagani, fleksibilni i robusni. Omogućuju dobivanje pouzdanih i kvalitetnih podataka. Korišteni Shimmer3 senzor ima mogućnost mjerenja površinske elektromiografije (engl. *surface Electromiography* - sEMG) i elektrokardiografije (engl*. Electrocardiography -* ECG). Površinska elektromiografija je tehnika za procjenu električne aktivnosti skeletnih mišića te će se koristiti u ovome projektu.

Korišten je ROS/Python paket (*shimmer\_ros*) [11]. Ovo je već gotov paket koji služi za komunikaciju između računala i senzora koji će se koristit u ovome projektu za dobivanje podataka. *Shimmer\_ros* paket služi za komunikaciju računala i Shimmer senzora te također kao i prethodni paket omogućuje praćenje, očitavanje i spremanje podataka dobivenih aktivacijom mišića dobivenih na senzoru.

Koriste se i u sljedećim područjima i aktivnostima:

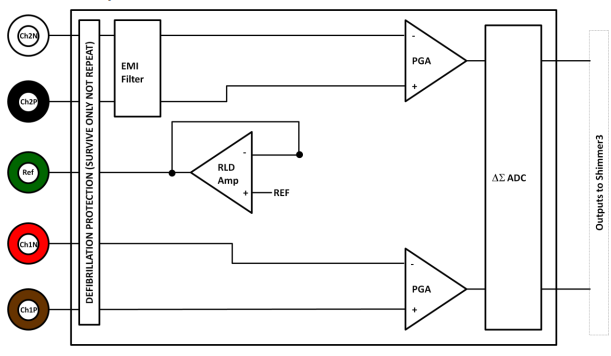
* Fibrilacija atrija
* Preuranjena ventrikularna kontrakcija
* Praćenje rada srca
* Abnormalno otkrivanje i upozorenje o ritmu
* Biomehanika, mišićna aktivnost, poremećaj hoda i držanja
* Analiza umora
* Spotska tehnika, izvedba i medicina
* Neurorehabilitacij
* Analiza tremora
* Veterinarska znanost
* Ortopedija
* Bioimpedancija prsnog koša (disanje)

Tehničke karakteristike senzora prikazane su u sljedećoj tablici:

Tablica 3.1. Tehničke karakteristike senzora

|  |  |
| --- | --- |
| Tehničke karakteristike | |
| Pojačanje | Softverski konfigurabilno (1, 2, 3, 4, 6, 8, 12) |
| Brzina prijenosa podataka | Softverski konfigurabilno (125, 250, 500, 1000, 2000, 4000, 8000 SPS) |
| Mrežna propusnost | 8.4 kHz |
| Težina | 31g |
| Dimenzije | 65 x 32 x 12 mm |
| EEPROM memorija | 2048 bajta |
| Priključci | EMG: Ulaz Ch1N, Ulaz Ch1P, Ulaz Ch2N, Ulaz Ch2P, Referentni (Ref)  ECG: Ulaz RA, Ulaz LA, Ulaz LL, Ulaz Vx, Ulaz (RL)  Svi Bolnički 1mm otporni na dodir IEC/EN 60601-1 DIN42-802 priključci |

Za mjerenje podataka korišten je jedan Shimmer3 senzor, s dva dostupna kanala za mjerenje pri čemu je korišten samo jedan kanal. Mjerenje jednim kanalom vrši se spajanjem tri elektrode na ulaze senzora. Korišten je kanal 1, gdje su elektrode spojene na *Ch1N* (kanal 1, negativna elektroda), *Ch1P* (kanal 1 pozitivna elektroda) te *ref* (referentna elektroda). Blok dijagram senzora prikazan je na slici 3.1. koji se sastoji od elektroda, zaštita od defibrilacije, EMI filter, pojačala pogona desne noge, programabilnog pojačala pojačanja i analogno digitalnih pretvarača. Zaštita od defibrilacije (engl. *defibrilation protection*) primjenjuje se za zaštitu od udara istosmjerne električne struje putem defibrilatora. EMI filter (engl. *electromagnetic interference*) smanjuje elektromagnetske smetnje. Pojačalo pogona desne noge (engl. *right leg drive amplifier*) suzbija smetnje uobičajenog načina rada (npr. iz električnih vodova ili fluorescentna svjetla). Programabilno pojačalo pojačanja (engl. *programmable gain amplifier*) služi za povećavanje amplitude ulaznog signala. Analogni digitalni pretvarači (engl. *analog to digital converters*) pretvaraju analogne signale u digitalne koji se dalje šalju u procesor Shimmer3 senzora. Potpuni princip rada senzora, rukovanje senzorom te ostale sveukupne informacije moguće je pronaći u literaturi [1].



Korištene elektrode prilikom mjerenja EMG signala

Slika 3.1 Blok dijagram Shimmer3 senzora [1].

## A/D prikupljanje podataka

Za dobro prikupljanje podataka mora se odabrati pogodna brzina uzorkovanja. Kada se mišić, o kojem prikupljamo podatke, aktivira on proizvede određeni signal. Ovaj signal ima veliki frekvencijski spektar pa ako želimo prikupiti čitavu frekvencijsku domenu tih signala onda frekvencija prikupljanja podataka mora biti barem dva puta veća od najveće frekvencije koja se očekuje prilikom aktivacije mišića. Nyquistov teorem o prikupljanju podataka opisuje vezu između ulaznog signala (signal kojeg odašilje mišić) i brzina uzorkovanja. Kada je brzina uzorkovanja niža od očekivane, rezultat je *alias*-efekt tj. preklapanja spektra. Posljedica je premala učestalost uzorkovanja ulaznog signala. Dobije se signal koji ne opisuje dovoljno dobro ulazni signal. brzina uzorkovanja odabire se prema spektru snage signala. Najveću snagu signal ima na frekvenciji između 10 i 250 Hz [2]. Prema znanstvenim preporukama cilj je izmjeriti kompletan frekvencijski spektar signala od 10 do 500 Hz. Tada brzina uzorkovanja mora biti najmanje dvostruko puta veća od najveće očekivane frekvencije, a to je 500 Hz. Zbog toga je odabrana brzina uzorkovanja od 1000 Hz, a stvarna brzina uzorkovanja je 992.9696 Hz. Efekt koji se javlja pri maloj frekvenciji prikupljanja podataka prikazan je na slici 3.2 gdje se može vidjeti da kada povećavamo brzinu uzorkovanja bolja je aproksimacija signala mišića.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Slika 3.2 Na slici je vidljivo da manja brzina uzorkovanja podataka lošije aproksimira ulazni signal, a veća bolje. [2]

U ovome poglavlju opisani su Shimmer3 senzori te zašto je važno odabrati optimalnu brzinu uzorkovanja podataka. U sljedećem poglavlju opisati će se provođenje eksperimenta, utjecaji na ispitivanje, odabir pozicije ispitivanja, nasumični statistički eksperiment te brza Fourierova transformacija.

# Planiranje i provođenje eksperimentalnih ispitivanja

U ovom poglavlju objasniti će se prikupljanje podataka pomoću sEMG (engl. *surface electro myography* ) Shimmera, utjecaj pozicije elektroda te ostali bitni faktori. Zatim će se definirati nasumični statistički eksperiment eksperiment i objasniti postupak ispitivanja.

Prilikom prikupljanja podataka pomoću senzora važno je otkloniti vanjske faktore koji utječu na podatke. Faktori koji utječu na prikupljanje signala su sljedeći: dlake, masnoća na koži, vrsta tkiva (razlika je kada subjekt ima malo ili puno masnog tkiva pri čemu više masnog tkiva ima veći utjecaj na pogrešno očitani signal), temperatura, elektroda hvata signal susjednog mišića (engl. *cross talk*) , promjena geometrije tkiva na mjestu elektrode zbog kontrakcije mišića, šumovi zbog drugih uređaja te greške koje se javljaju zbog same elektrode, vodiča, bluetootha itd. Većinu ovih šumova eliminirati će se filtriranjem signala pomoću Brze Furierove transformacije (FFT), brisanjem kože medicinskim alkoholom, ispitivanjem u kontroliranim uvjetima ( temperatura: 20° ≤ t ≤ 25°) te će tijek ispitivanja za svakog subjekta biti sličan koliko god je to moguće.

## Odabir pozicije elektroda

Jedan od ciljeva ovog programskog zadatka je odrediti optimalnu poziciju elektrode da bi se dobila najveća moguća korelacija između snage stiska šake mjerene dinamometrom i aktivacije mišića mjerene sEMG-om. Mjerenje jakosti aktivacije mišića provedeno je na mišiću koji je najodgovorniji za stisak šake, a to je mišić podlaktice (lat. *Flexum Digitorum Profundus*) prikazan na sljedećoj slici.



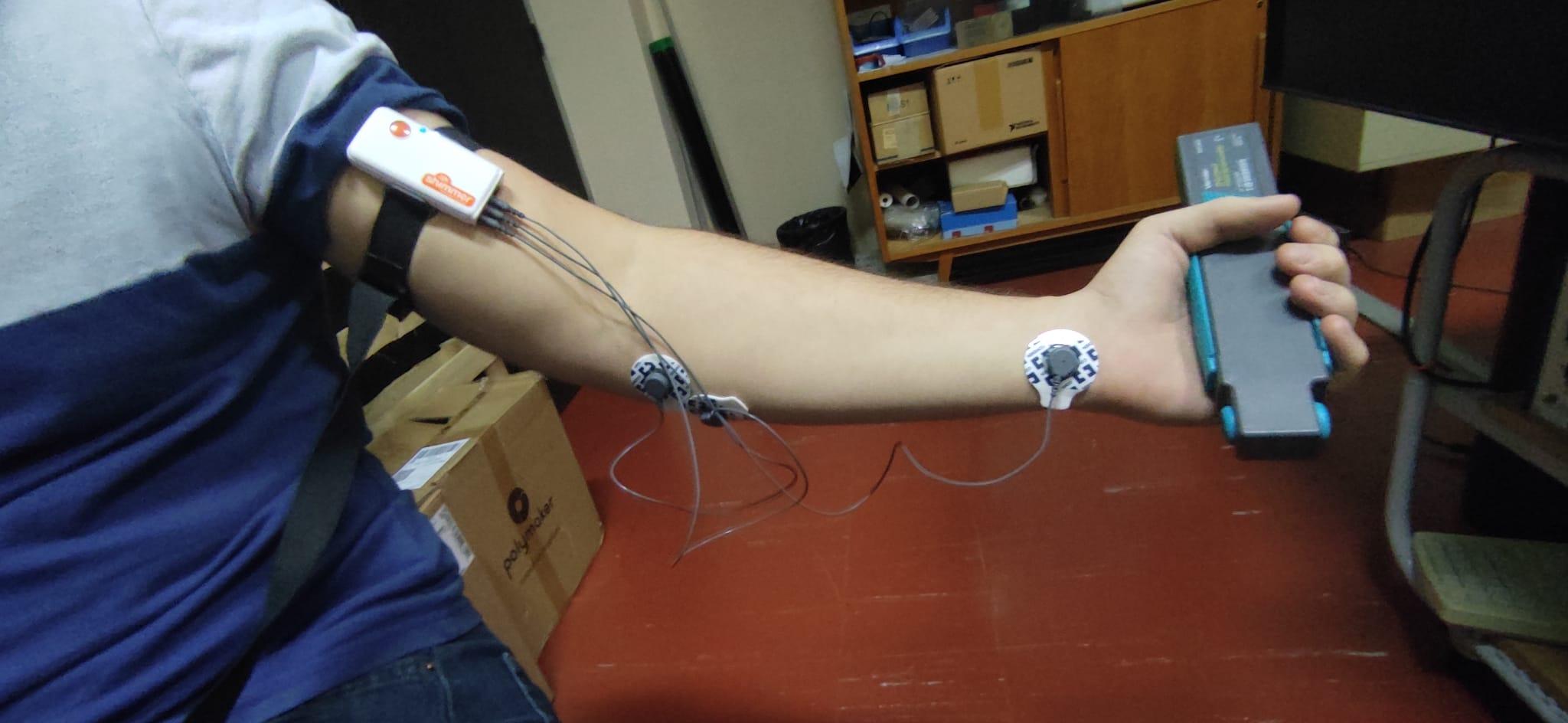
Slika 4.1 Flexor Digitorum Profundus. Najutjecajniji mišić pri stisku šake [10].

Pretpostavljeno je da će najveća korelacija biti onda kada je za neku određenu silu stiska aktivacija mišića najveća, te da se grafovi poklapaju. Prilikom odabira najbolje pozicije elektroda provedena su mjerenja kojima su eliminirana najlošije pozicije, to jest one pozicije gdje se aktivacija mišića i graf sile najmanje poklapaju, gdje je za istu silu stiska aktivacija manja. Traži se za neku silu stiska najveća aktivacija. Mjereno je pet pozicija, od toga su dvije odabrane kao pozicije koje bi potencijalno mogle dovesti do željene korelacije između stiska i aktivacije. Mjerene pozicije prikazane su na sljedećim slikama.

Slika na kojoj se prikazuje na zatvorenom, pod, namještaj

Opis je automatski generiran

Slika 4.2 Pozicija 1



Slika 4.3 Pozicija 2.

Slika na kojoj se prikazuje osoba, na zatvorenom

Opis je automatski generiran

Slika 4.4 Pozicija 3.

Slika na kojoj se prikazuje osoba

Opis je automatski generiran

Slika 4.5 Pozicija 4.

Slika na kojoj se prikazuje na zatvorenom, zid, pod, osoba

Opis je automatski generiran

Slika 4.6 Pozicija 5.

Pozicije 3 i 4 dale su najbolje aktivacije mišića za određenu silu stiska. Prema tome su ove dvije pozicije elektroda odabrane kao pozicije na kojima će se provoditi ispitivanja radi dobivanja najbolje korelacije između sile stiska šake i aktivacije mišića. Pokazalo se da je očitanje signala puno bolje kada se referentna elektroda nalazi na mjestu gdje završava tetiva palca nego kada se postavi na donju stranu dlana (vidljivo na poziciji br. 2) ili ako se postavi na lakat. Postavljanje referentne elektrode na poziciju 2 pokazalo se loše zbog toga što je prilikom stiska šake elektroda deformira te dođe do odljepljivanja, isto se događa i kada se referentna elektroda postavi na lakat. Također jako veliki utjecaj na očitanje ima i dlakavost pa se javlja slabo očitanje signala.

## Postupak ispitivanja

Ispitivanje se provodilo na deset subjekata. Ispitivale su se dvije pozicije (pozicija 3 i 4). Prije početka ispitivanja subjekti su se upoznali kakvo se ispitivanje provodi te koji je postupak koji moraju pratiti. Postupak je sljedeći:

1. Shimmer3 senzor se postavi na ispitanikovu ruku kao na slici 4.5.
2. Koža se obriše medicinskom alkoholnom maramicom
3. Elektrode se postave na kožu. Ovisno o mjerenoj poziciji
4. Ispitanik sjedne na stolicu
5. Ruka se postavi u krilo tako da je kut između nadlaktice i podlaktice 90°

Redoslijed stiskanja je sljedeći:

stisak 100%, 5s

odmor 3s

stisak 75%, 5s

odmor 3s

stisak 50%, 5s

odmor 3s

stisak 25%, 5s

Za svaku poziciju ovaj redoslijed provodi se tri puta. Ukupni broj ispitivanja je 60.

## Nasumični blok plan eksperimenta

Nasumični statistički eksperiment (engl*. Randomized Block Design - RBD*) je statistički eksperiment gdje se eksperimentalne jedinice dijele u grupe zvane blokove [12]. Kod ovih eksperimenata jedna je varijabla od primarnog interesa dok su ostale varijable smetnja. Smetnjom se smatra svaka varijabla koja utječe na mjereni rezultat, ali nije od primarnog interesa. Glavna varijabla kod koje se želi izbjeći njen utjecaj je pozicija elektrode kod ispitanika. Smetnja je u ovome slučaju ispitanik pa se želi umanjiti utjecaj ispitanika na prikupljene podatke. Način na koji ispitanik utječe na signal je taj da je geometrija ruke svakog ispitanika drugačija pa je jako teško odrediti poziciju postavljanja elektroda na specifični dio mišića. RBD se sastoji od 2 faktora (primarni faktor-pozicija elektrode, 2 faktor (smetnja) -subjekt). Model RBD s jednim faktorom smetnje računa se prema izrazu:

(4.4)

Podaci za izračun:

|  |  |  |  |
| --- | --- | --- | --- |
| Koeficijent | Formula | Vrijednosti koeficijenata | izračun |
|  |  | - | - |
| k | - | 2 | - |
|  | - | 2 | - |
|  | - | 10 | - |
|  | - | 3 | - |
|  |  |  | 60 |
|  | - | - | - |
|  |  | - | - |
|  |  | - | - |

Gdje su:

položaj elektrode (i=1…2)

subjekt (j=1…10)

vrijednost za poziciju i na subjektu j

srednja vrijednost

efekt tretmana(ispitivanja)

efekt bloka

broj faktora

broj primarnih faktora

Broj ponavljanja za jedno mjerenje (broj mjerenja jedne pozicije)

ukupni broj mjerenja

Prosjek svih Y za koje je

Prosjek svih Y za koje je

# Obrada podataka

U ovom poglavlju opisati će se najčešće korišteni algoritmi za obradu sEMG signala, uređivanje i usklađivanje podataka u .csv datoteci te obrada podataka te brza fourierova transformacija.

EMG signal je jako osjetljiv, na njega utječu mnogi faktori pa je zbog toga važno ispravno provoditi ispitivanja i obradu podataka. Čisti EMG signal (neobrađeni signal) sadrži jako puno informacija i šumova, ali da bi se dobile informacije koje je lakše interpretirati i shvatiti, signal se mora obraditi. Ispravnim provođenjem ispitivanja, postavljanjem elektroda, pripremanjem subjekata na ispitivanje te radom u povoljnim uvjetima moguće je otkloniti mnoge negativne utjecaje.

EMG elektrode mogu uhvatiti signal obližnjih uređaja i kablova što dovodi do smetnji koje se mogu vidjeti na grafu. Ove smetnje vide se kao odmak od ishodišta grafa ili kao npr. periodički tok signala (signal ima oblik sinusoide te ga je lako prepoznati na grafu), a smanjuje se ili povećava s odmakom od elektroničkih uređaja. Nagli odmaci od ishodišta javljaju se zbog naglog pomicanja žica elektroda ili loše fiksacije elektroda, a mogu se primijetiti kao kratki ''peakovi'' koji se u nekoliko milisekundi vraćaju na nulu. ECG (engl*. Electrocardiography*) signal aktivacije srca također utječu na signal. Smanjuje se odmakom elektroda od srca tj. što se elektrode postave bliže srcu to on ima veći utjecaj. Nadalje, objasniti će se brza fourierova transformacija te neki od algoritama za obradu podataka.

51

Full-wave rectification (rektifikacija) je metoda kojom se svim negativnim amplitudama neobrađenog signala dajemo apsolutnu vrijednost. Neobrađeni signal ima simetrični gornji i donji dio gledajući s obzirom na apscisu pa možemo uzeti apsolutnu vrijednost, te tako dobijemo samo pozitivni dio signala.

Zaglađivanje je postupak kojim ispravljenom signalu računamo srednju vrijednost, tako što eliminiramo ''peakove'', a signal dobije zaglađeni oblik (ovojnicu). Koriste se dva algoritma: pomični prosjek (engl. *Moving average* - MA) i srednja vrijednost kvadrata (engl. *Root mean square*). Za obradu signala koristi se **pomični prosjek**.

MA algoritam koristi se na rektificiranom signalu, njegova uloga je procjena amplitude, a informacija koju nam daje odnosi se na površinu ispod epohe signala. Za njegovu provedbu potrebno je definirati vremenski prozor (engl. *rolling window*) iz kojeg se uzimaju podaci. Podaci signala se usklađuju s prozorom te se dobiva moving average. Ovaj algoritam koristi se za obradu dobivenih podataka. U Yaml kodu, linija 19 definira se eksponencijalni pomični prosjek (engl. *Eksponential moving average – EMA*) čija jednadžba glasi:

Gdje je:

– trenutna vrijednost izlaznog signala

prethodna vrijednost izlaznog signala

– trenutna vrijednost ulaznog signala

– faktor smanjenja (*eng. Decreasing factor*)

1.# Window size for filtering (as s fraction of sampling\_rate)

2. window\_size\_relative: 0.5

3. # Window size for smoothing (sec) - recommended between 0.05 and 0.5 sec

4. window\_size\_smoothing: 0.3

5. # Republished topic with filtered EMG - queue size

6. repub\_queue\_size: 10

7. # Clean signal republished topic - queue size

8. clean\_queue\_size: 1000

9. # Band-stop frequencies (Hz) - a list of low.high ranges [[low, high][low, high]]

10. band\_stop: [[48, 53]]

11. # band\_stop: []

12. # High pass filter cutoff frequency (Hz), null for None

13. hpf\_cutoff: 10

14. # Low pass filter cutoff frequency (Hz), null for None

15. lpf\_cutoff: 450

16. # Rectify EMG signal after filtering (absolute value)

17. rectify: true

18. # Smoothing algorithm (rms - root mean square, ema - exponential moving average, false - none)

19. smoothing: 'ema'

20. # Exponential moving average decreasing factor (0 - 1),

21. # 0 - simple moving average

22. # Recommendation to use values < 0.01 (older values will have too low weights otherwise)

23. ema\_decay: 0.002

24. # FFT Thresholding - relative value to keep only FFT values larger or equal to

25. # fft\_threshold \* max(fft\_magnitude)

26. fft\_threshold: 0.0

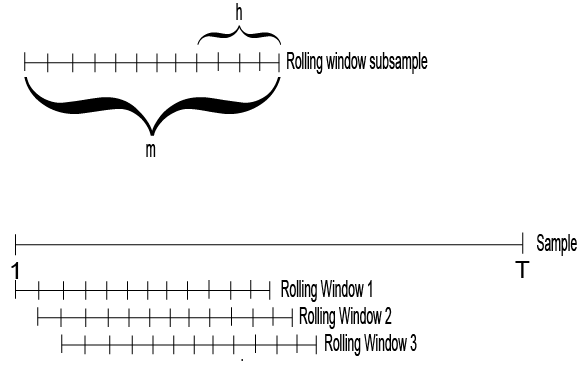
Faktor smanjenja odabran je takav da je jednak nuli te se dobije jednostavan pomični prosjek čija jednadžba glasi:

Gdje su:

prosjek podataka u jednom vremenskom prozoru

broj vremenskih prozora

Na sljedećoj slici shematski je prikazan pomični prozor.



Slika 5.2. Rolling window [9]

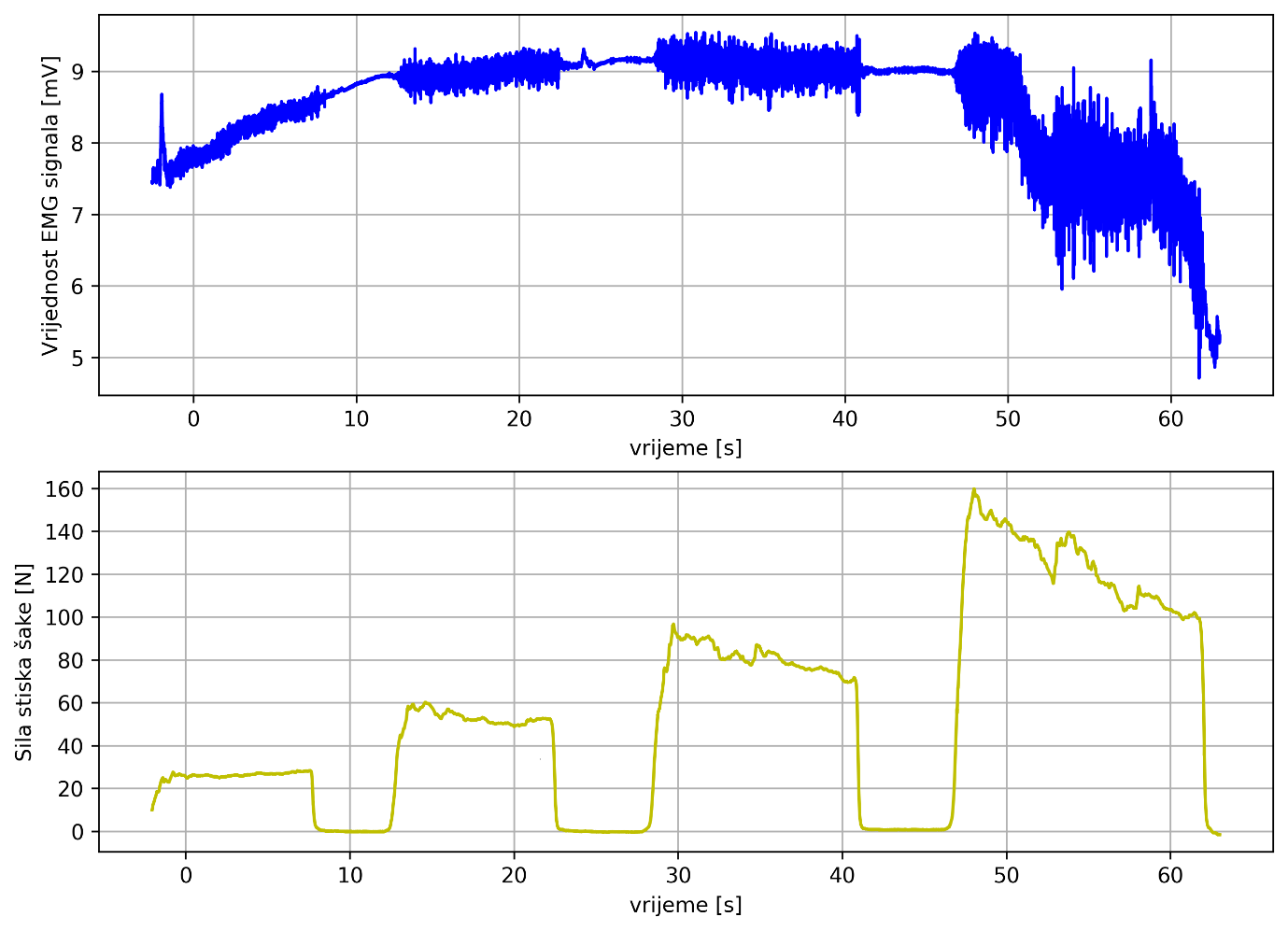
Cilj obrade podataka je dobiti optimalne parametre uz koje ćemo dobiti najbolju korelaciju za određeno mjerenje. U ROS-u su stavljeni neki parametri koji su određeni iskustvenim odabirom. Pomoću skripte je potrebno odrediti te optimalne parametre prvo za svaki set podataka posebno, a onda jedan set parametara za sve setove podataka zajedno. Inicijalni parametri su prikazani u prethodnom Yaml kodu.

Parametri koje nam je cilj optimizirati su pomični prozor (engl*. Rolling Window Size*) i faktor zaglađivanja (engl. *Smoothing factor*) te ''maska'' s kojom množimo amplitude dobivene FFT-om. Ovi su parametri zaduženi za filtriranje signala. Filtriranjem tražimo optimalne parametre koji maksimiziraju korelaciju.

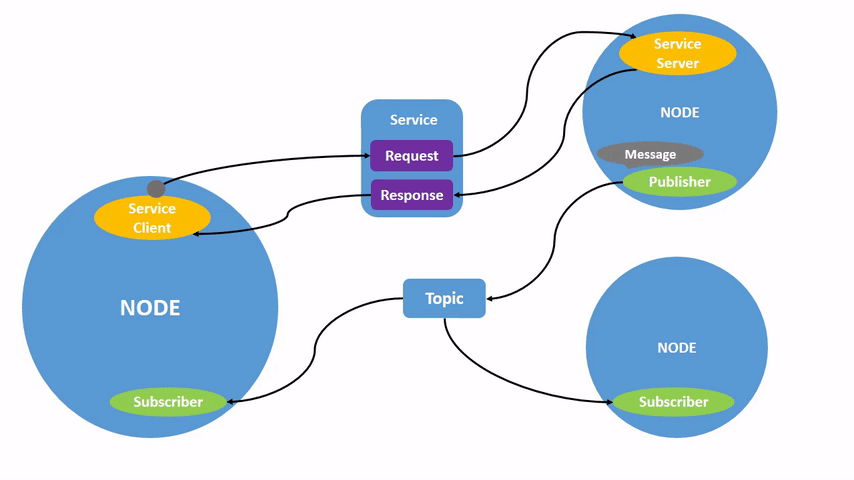
## Obrada ROS izlaza [13]

Za potrebe projekta korišten je robotski operacijski sustav (engl. *robot operating system* – ROS). ROS je set knjižnica i alata koji omogućuje upravljanje robotskim aplikacijama, otvorenog je koda te nudi usluge kao što su kontrola uređaja, slanje poruka između procesa, komunikacija između procesa itd.

Uspješnim spajanjem Shimmera i dinamometra na Linux te uspješno povezivanje, radili smo mjerenje nad subjektima već unaprijed dogovorenim postupkom. Jedno mjerenje prikazano je na slici 5.3.

*Slika 5.3 Odziv mišića podlaktice prilikom stiska šake: mjerene sile u N pomoću dinamometra (plavo), mjerenje pomoću EMG osjetnika (žuto)*

Graf prikazan na slici 5.3 prikazuje silu dinamometra u njutnima u narančastoj boji, a plavi graf prikazuje odaziv EMG signala. U ROSu biramo „teme“ (engl. *topics*) koje želimo prikazati. Plotjuggler se koristi za čistu vizualizaciju podataka koja olakšava subjektu da prati svoja snimanja tijekom eksperimenta. Za plavi graf smo mogli birati između filtriranog signala i čistog signala. Bitno je naglasiti da je filtrirani signal samo vizualna pomoć testnog subjekata u procjeni kvalitete signala, budući da mi za obradu ćemo koristiti čisti signal, tj. onaj koji nije filtriran. Tako najbolje znamo ako su elektrode dobro pričvršćene, postavljene na dobro mjesto ili pak testni subjekt dobro radi mjerenja za eksperiment. Za pohranu svakog mjerenja od svakog subjekta, koristili smo ROSBAG [11] paket koji služi za snimanje ROS topica. Njega smo koristili u slučaju oštećene .csv datoteke te kao backup datoteka u slučaju nepravilnog odabira mjerenja. Ovim programom snimani su podaci dobiveni ispitivanjem te nam je omogućilo da u bilo kojem trenutku imamo podatke dostupne za obradu ili ponovnu reprodukciju.



*Slika 5.4 Prikazuje izmjenu poruka između čvorova (eng. Nodes) i teme (eng. Topic). U našem slučaju događa se komunikacija između računala, dinamometra i senzora.*

Nakon provođenja tri mjerenja po poziciji (odabrane su dvije pozicije) na deset subjekata, potrebno je obraditi podatke koje su „snimiljene“ u ROSu, tj. za sveukupno 60 mjerenja i .csv datoteka. Otvaranjem jedne .csv datoteke, nailazi se na prvi problem u obradi podataka.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| \_time | /gdx/grip\_force\_stream/grip\_force | /gdx/grip\_force\_stream/header/seq | /gdx/grip\_force\_stream/header/stamp | /shimmer0/raw\_emg\_stream/emg\_ch1 | /shimmer0/clean\_emg\_stream/emg\_ch2 |
| 167031905,75316 |  |  |  | 0,017493 | 0,017365 |
| 167031905,75328 | -0,376633 | 1338157 | 1670319054 |  |  |
| 167031905,75536 |  |  |  | 0,017501 | 0,017359 |
| 167031905,75756 |  |  |  | 0,017516 | 0,017355 |
| 167031905,75886 |  |  |  | 0,017519 | 0,017363 |
| 167031905,75998 |  |  |  |  |  |
| 167031905,76085 |  |  |  | 0,017526 | 0,017387 |
| 167031905,76195 |  |  |  |  |  |
| 167031905,76305 |  |  |  | 0,017497 | 0,017355 |
| 167031905,76415 |  |  |  | 0,017428 | 0,01728 |
| 167031905,76525 |  |  |  | 0,017365 | 0,017206 |
| 167031905,76635 |  |  |  |  |  |
| 167031905,76744 |  |  |  | 0,017333 | 0,017172 |
| 167031905,76854 |  |  |  |  |  |
| 167031905,76964 |  |  |  | 0,017279 | 0,017124 |
| 167031905,77074 |  |  |  | 0,017154 | 0,017004 |
| 167031905,77184 |  |  |  |  |  |
| 167031905,77294 |  |  |  | 0,016925 | 0,016783 |
| 167031905,77404 |  |  |  |  |  |
| 167031905,77514 |  |  |  | 0,016908 | 0,016782 |
| 167031905,77623 |  |  |  | 0,016954 | 0,016825 |
| 167031905,77733 |  |  |  |  |  |

Slika 5.5 Primjer neporavnatih podataka

Slika 5.5. prikazuje ROS podatke koji su vremenski sinkronizirani, kako stižu na ROS topic (lijevi stupac je poredan od najmanje do najviše vrijednosti). Prema slici 5.5., ROS izbacuje vrijednosti globalno vrijeme, vrijeme dinamometra, silu dinamometra, kanale shimmera, očišćene kanale shimmera te vrijeme shimmera. Exportana .csv datoteka ima jako puno stupaca i redova koji nam ne trebaju za obradu podataka. Problem nastaje u različitim „tickovima“ samog shimmera i dinamometra. Drugim riječima, dinamometar i shimmer objavljuju podatke u različitim frekvencijama i različitim vremenima te su spremljeni kako stižu na ROS topic. To rezultira „rupama“ u .csv datoteki koje treba zakrpati.

Princip „krpanja“ jest prvo uskladiti vremena dinamometra i shimmera te pomoću interpolacije popuniti podatke koji nedostaju. Vremenski se poravnavaju prve dvije točke EMG signala i sile stiska. R skripta [11] treba učitati sve .csv datoteke koje smo spremili u ROSu, poravnati ih, te spremiti pod drugim imenom. Na kraju, dobivamo tri stupca koji nam trebaju za daljnju obradu podataka; vrijeme, nefiltrirani EMG kanal te sila u dinamometru u tom vremenu. Izgled sređene datoteke prikazuje slika 5.6.

|  |  |  |
| --- | --- | --- |
| measure\_time | emg\_ch1 | grip\_force\_inter |
| 0.0000000000 | 3.236048 | 1.208014 |
| 0.0010101795 | 3.235543 | 1.173086553 |
| 0.0020101070 | 3.240375 | 1.350483455 |
| 0.0030200481 | 3.245976 | 1.191063649 |
| 0.0040299892 | 3.248044 | 1.244192 |
| 0.0050301552 | 3.248308 | 1.244260448 |
| 0.0060400963 | 3.24814 | 1.246240762 |
| 0.0070500374 | 3.24778 | 1.248220606 |
| 0.0080599785 | 3.244726 | 1.25020092 |
| 0.0090601444 | 3.267084 | 1.252181233 |
| 0.0100700855 | 3.288961 | 1.254161077 |
| 0.0110800266 | 3.297183 | 1.256141391 |
| 0.0120799541 | 3.299371 | 1.258123579 |
| 0.0130901337 | 3.289514 | 1.260103893 |
| 0.0141000748 | 3.268911 | 1.262083737 |
| 0.0151100159 | 3.254222 | 1.264064051 |
| 0.0161099434 | 3.233307 | 1.266044364 |
| 0.0171201229 | 3.211142 | 1.268024208 |
| 0.0181300640 | 3.208569 | 1.270004522 |
| 0.0191299915 | 3.22059 | 1.271984835 |
| 0.0201401711 | 3.221239 | 1.27396468 |
| 0.0211501122 | 3.225205 | 1.275944993 |
| 0.0221500397 | 3.228619 | 1.277925306 |
| 0.0231599808 | 3.233019 | 1.279905151 |
| 0.0241701603 | 3.237827 | 1.281885464 |
| 0.0271899700 | 3.236264 | 1.287827811 |
| 0.0282001495 | 3.240279 | 1.289808124 |

Slika 5.6 Poravnati parametri

## Optimiranje parametara EMG signala [13]

Nakon dobivanja potrebnih .csv datoteka, potrebno je taj signal filtrirati te ga usporediti s vrijednostima dinamometra. Za to koristimo Python skriptu koja filtrira signal pomoću Fourierove transformacije koja je već detaljno opisana u prošlom poglavlju.

Cilj je dobiti parametre koji će pokriti svaki set podataka ta za njih naći prosječnu najveću korelaciju te optimalne parametre koji se tada mogu implementirati u ROS. Skripta koja se koristi ima nekoliko funkcija. Prva funkcija joj je da generira parametre *smoothing factor* te *rolling window size*. Zatim se definira *fft relative window size* koji je jednak onome kao u ROS skripti. Sljedeće se definiraju *sampling rate* i *fft window size* koji se dobivaju iz podataka. *FFT windows size* se dobiva na način da se uhvati frekvencijski spektar koji se definirao u inicijalnim parametrima ROSa (slika 6.2) i on izosi 496. *Sampling rate* je dobiven na temelju podataka kao recipročna vrijednost medijana razlike vremenskih podataka te iznosi 992,96969 Hz. Parametri se generiraju kao brojevi od 0.01 do 0.5 za *rolling window size* te od 0.001 do 0.05 za *smoothing factor*. Zbog prirode postupka na koji se dobivaju parametri *FFT window size* i *sampling rate*, definirani su kao „hard coded“, tj. budući da su isti za bilo koji set podataka, ne učitavaju se sa seta podataka nego se definiraju kao konstante tako da nije potrebno dva puta učitavati .csv datoteke preko petlje. Sljedeće se radi petlja koja učitava podatke mjerenja. To se dobiva definiranjem radne mape „new\_data“ pomoću naredbe glob.glob. i jedne for petlje koja to radi za svaku datoteku u radnoj mapi. Zatim, iz učitane datoteke se definiraju varijable koje odgovaraju pojedinim stupcima, tj. vrijeme, EMG podaci i podaci s dinamometra. Bitno je naglasiti da ukupan broj podataka mora biti višekratnik veličine prozora da bi obrada podataka funkcionirala. Nakon toga, potrebno je preoblikovati podatke da funkcioniraju u daljnjoj obradi, tj da podaci odgovaraju veličini prozora.

Vrši se diskretna pretvorba „emg\_dat“ varijable u FFT pomoću naredbe numpy.fft.rfft. Dimenzija tog FFTa iznosi 249 što govori koliko je dugačak taj FFT. Sljedeći zadatak skripte je generirati FFT masku koja sadrži 249 nasumičnih vrijednost od 0 – 1. Radi se iterativna petlja pomoću paketa itertools koja traži optimalne za maksimalnu korelaciju. Nakon definiranja tih parametara, koristi se obrnuta Fourierova transformacija da bi se vratila funkcija iz frekvencijske u vremensku domenu, nalazi se EMA za neku FFT masku generiranu prije druge petlje, uspoređuje sa prijašnjom kombinacijom parametara *smoothing factor* i *rolling window size* te, ako je postignuta maksimalna vrijednost korelacije, izlazi van iz petlje.

Kao preliminarni test petlje, radi se skripta za maksimalnu korelaciju samo za jedan set podataka. Ovakav test služi za predviđanje i komentiranje konačnih dobivenih podataka te potencijalno ubrzavanje glavne skripte, ako se vidi da se optimalne vrijednosti nalaze u nekom užem intervalu. Ova skripta radi obrnuto od glavne skripte; prvo učitava set podataka pa tada traži optimalne parametre. Slika 6.6 prikazuje postupak koji se vrši nad jednim takvim setom podataka, te prikazuje se ovisnost maksimalne korelacije o *smoothing factoru*, *rolling window size* te kašnjenju.

a)

b)

Slika 5.7 a) Ovisnost maksimalne korelacije o smoothing factoru b) Ovisnost maksimalne korelacije o rolling windowu size za jedan set podataka

Optimalni parametri su: faktor zaglađivanja, pomični prozor i vrijeme kašnjenja. Faktor zaglađivanja (engl. *smoothing factor*) se koristi u postupku kojim ispravljenom signalu računamo srednju vrijednost, tako što eliminiramo ''peakove'', a signal dobije zaglađeni oblik. Pomični prozor (engl. *rolling window size*) je definiran kao vremenski prozor iz kojeg se uzimaju podaci. Veličina pomičnog prozora ovisi o veličini uzorka, tj. periodičnosti podataka. Vrijeme kašnjenja (engl. *time lag*) predstavlja vremensko razdoblje između dvije povezane radnje. Na slici 5.7 u prvom prozoru se vidi da možemo očekivati da će za maksimalnu korelaciju vrijednost *smoothing factor* biti između 0 i 0.01 te vrijednost *rolling window size* između 0.15 i 0.4. Rezultati se spremaju u .csv datoteku je se nalaze konačne vrijednosti za *smoothing factor* i *rolling window size* da se uvrste u ROS. Također, pronađene su maksimalne korelacije za optimalne parametre od svakog seta podataka, tj. datoteke i najveće kašnjenje između dinamometra i Shimmera. Maksimalna korelacija je postignuta tad, kad je set podataka najbolje filtriran. Grafički je prikazano na slici 5.8.

Chart

Description automatically generated

*Slika 5.8 signali za optimalne parametre a) Nefiltrirani signal (plavo) b)Poravnati signal(crveno) c) Zaglađeni signal (zeleno) d) Stisak šake (žuto)*

Graf „Nefiltrirani signal“ prikazuje čisti, ne profiltrirani EMG signal kakav se dobiva kao rezultat snimljenih podataka. Graf „Pomaknuti signal“ prikazuje taj isti graf, samo pomaknuti u ishodište, tj. da je prva vrijednost uvijek nula. Graf „Zaglađeni signal“ prikazuje optimalno profiltrirani signal za najveću korelaciju. Na kraju, graf „Stisak“ prikazuje signal dinamometra u njutnima. Za taj set podataka, *smoothing factor* iznosi 0.001, a *relative window size* iznosi 0.32. Maksimalna korelacija iznosi 0.957, što je više nego zadovoljavajuće.

Na temelju slike 6.7. može se usporediti kakvo će biti zaglađivanje s faktorima *smoothing factor, rolling window size* te FFT maskom kada se pokrene glavna skripta koja učitava sve podatke te ih obrađuje istovremeno. Sada, *smoothing factor* iznosi 0.001, *rolling window size* iznosi 0.46 te najveća korelacija iznosi 0.946. FFT maska je prikazana slikom 6.8.

Slika 5.9 Vektor od 250 nasumičnih generiranih vrijednosti od 0 do 1 dok je nulta pozicija uvijek nula

Vrši se diskretna pretvorba „emg\_dat“ varijable u FFT pomoću naredbe numpy.fft.rfft. Dimenzija tog FFTa iznosi 249 što govori koliko je dugačak taj FFT. Sljedeći zadatak skripte je generirati FFT masku koja sadrži 249 nasumičnih vrijednost od 0 – 1. Radi se iterativna petlja pomoću paketa itertools koja traži optimalne za maksimalnu korelaciju. Nakon definiranja tih parametara, koristi se obrnuta Fourierova transformacija da bi se vratila funkcija iz frekvencijske u vremensku domenu, nalazi se EMA za neku FFT masku generiranu prije druge petlje, uspoređuje sa prijašnjom kombinacijom parametara *smoothing factor* i *rolling window size* te, ako je postignuta maksimalna vrijednost korelacije, izlazi van iz petlje.

# Analiza podataka RBD esperimenta

Ispitivanje se provodilo na deset subjekata. Ispitivale su se dvije pozicije (pozicija 3 i 4). Za svaku poziciju vršilo se mjerenje po tri puta. Ukupan broj ispitivanja za deset ispitanika je bio 60. RBD (*engl. Randomized Block Design*) se sastoji od 2 faktora (primarni faktor-pozicija elektrode, 2 faktor (smetnja) -subjekt).

Primarni faktor [Yj] čiji utjecaj ima pozicija elektrode određuje se na način da se zbroje izmjerene korelacije za poziciju 3 i 4 te se podijeli sa ukupnim brojem mjerenja za svakog ispitanika. Drugi faktor [Yi] predstavlja smetnju koju uzrokuje ispitanik. Računa se kao aritmetička sredina ukupnih izmjerenih korelacija za svakog ispitanika za jednu poziciju. Isti postupak vrijedi i za drugu poziciju. U ovom slučaju imamo deset ispitanika po tri mjerenja za svaku poziciju (ukupno 30). Potom se određuje srednja vrijednost [µ] računa se kao aritmetička sredina ukupnog broja ispitivanja za deset ispitanika, a to je 60.

Zatim se računa efekt bloka Bj prema izrazu:

(6.1)

Gdje je:

*Bj - efekt bloka*

*Yj - primarni faktor*

*- srednja vrijednost sustava*

Nakon toga se dobiva prema konačna vrijednost koja je određena sa pozicijom elektrode i subjekta na kojem se vrši mjerenje. Računa se prema izrazu:

(6.2)

Gdje je:

*vrijednost za poziciju i na subjektu j*

*efekt tretmana(ispitivanja)*

Prema tablici 6.1. možemo vidjeti da bolju korelaciju dobivamo na drugoj poziciji (pozicija 4) koja iznosi 0,949352. Uspoređujući je sa drugom dobivenom korelacijom iz prve pozicije (pozicija 3) koja iznosi 0,946126, vidimo da je malo veća prema tablici 6.1.

|  |  |  |
| --- | --- | --- |
| µ=0,947739 | | |
| Yij | i |  |
| j | 1 | 2 |
| 1 | 0,925928 | 0,929153 |
| 2 | 0,940174 | 0,9434 |
| 3 | 0,949667 | 0,952892 |
| 4 | 0,963657 | 0,966883 |
| 5 | 0,956384 | 0,959609 |
| 6 | 0,940822 | 0,944047 |
| 7 | 0,929993 | 0,933218 |
| 8 | 0,958393 | 0,961618 |
| 9 | 0,949005 | 0,95223 |
| 10 | 0,947239 | 0,950464 |
|  | 0,946126 | 0,949352 |

*Tablica 6.1. Usporedba korelacija između dviju pozicija elektrode temeljem RDB eksperimenta*

Rezultate koje smo dobili prilikom ispitivanja na deset sudionika prikazani su *Prilozima*.

# Zaključak

Projektni zadatak je bio nadograditi implementaciju mjernog sustava dinamometra i Shimmera3 sEMG osjetnika. Korišteni su Vernier dinamometar kao osjetnih stiska i Shimmer3 sEMG senzor za aktivaciju mišića. Shimmer3 sEMG se koristi u medicinskim svrhama. Velika je pomoć kod rehabilitacije bolesnika koji su doživjeli moždani udar. Neki bolesnici imaju problema sa otkazivanjem mišića pojedinih dijelova tijela. Upravo sEMG se koristi za računanje jačine stiska šake.

Na početku se vršila kalibracija dinamometra, kada je dobivena kalibracijska jednadžba iz čega su iščitani parametri koji su se upisali u kod već gotovog paketa godirect\_ros, kako bi uređaj na izlazu pokazivao ispravne vrijednosti.

Zatim se je određivala povoljna pozicija elektrode za provedbu daljnjih ispitivanja. Na početku provodilo se ispitivanje na pet pozicija, od toga su dvije pozicije pokazale dobru vezu između sile stiska i aktivacije mišića. Ove dvije pozicije odabrane kao najbolje za sljedeći korak. RDB eksperimentom je upravo određena najbolja pozicija. Sljedeći korak bio je provesti 60 ispitivanja nad 10 ispitanika kako bi se dobiveni podaci mogli obraditi. Za obradu prikupljenih podataka napisan je Python kod.

Pomoću Python koda obradili su se podaci, a na kraju su dobiveni željeni rezultati kojima možemo interpretirati vezu između sile stiska i aktivacije mišića. Dobiveni rezultati ukazuju na jako veliku korelaciju sile i aktivacije koja u prosjeku iznosi 0.946. Ovaj podatak ukazuje nam na to da je zadatak dobro odrađen te da bi sljedeći cilj bio povezivanje Shimmer senzora i uređaja za rehabilitaciju.

Sljedeći korak u ovom projektu je implementirati dobivene parametre i FFT masku u ROS te ponoviti svih 60 mjerenja na svih 10 subjekata te iz tih mjerenja izvući novu maksimalnu korelaciju.

Glavni problem bili su nepoznavanje ili nedovoljno poznavanje programskih jezika pomoću kojih su se obrađivali podaci. Problemi su nastajali u vezi između Shimemer3-a i računala. Uređaj bi radio kako treba i nakon nekog vremena bi prestao raditi. To nam je oduzelo jako puno vremena pri prikupljanju podataka. Dlakavost ruku, mokre ruke ili znojne i loše lijepljenje elektroda je dovodilo do grešaka u rezultatima ili nekada nebi uopće pokazivao rezultate.

Sudionici su se susreli s novim područjima poput programiranja, principa rada mišića itd. Neka područja bila su teška za savladati i shvatiti pa je utrošeno jako puno vremena za jako mali napredak.

# 

# 8. Literatura

[1] Unknown author.: ‘’EMG User Guide Revision 1.12’’, Retrieved from Internet, <https://shimmersensing.com/wp-content/docs/support/documentation/EMG_User_Guide_Rev1.12.pdf#page=13&zoom=100,92,389>. 1.2.2023.

[2] Konrad, Peter.: ''The ABC of EMG'', Norax. USA,

[3] Umjeravanje dinamometra za mjerenje hvata šakom, L. Zvonarek, M. Blagdan, 2022.

[4] S interneta: <https://www.youtube.com/watch?v=spUNpyF58BY>, kanal 3brown1blue, preuzeto 7.2.2023.

[5] S interneta: <https://betterexplained.com/articles/an-interactive-guide-to-the-fourier-transform/>, preuzeto 7.2.2023.

[6] The Scientist and Engineer's Guide to Digital Signal Processing, Steven W. Smith

[7] Sa intereneta: <https://dewesoft.com/daq/guide-to-fft-analysis>, preuzeto 7.2.2023.

[8] Sa interneta: <https://github.com/tbazina/shimmer_ros>, preuzeto 7.2.2023.

[9] Sa interneta: <https://www.mathworks.com/help/econ/rolling-window-estimation-of-state-space-models.html> , preuzeto 6.2.2023.

[10] S interneta, <https://www.kenhub.com/en/library/anatomy/flexor-digitorum-profundus-muscle> , 14.2.2023.

[11] S interneta, Bazina, Tomislav.: godirect\_ros & godirect\_shimmer\_ros'' <https://github.com/tbazina/godirect_ros> , S interneta, 14.2.2023.

[12] S interneta, <https://www.itl.nist.gov/div898/handbook/pri/section3/pri332.htm>, 14.2.2023.

[13] S interneta, Zvonarek, Luka, <https://github.com/kittnznstuff/UMS-RiTeh>, 14.2.2023.

[14] S interneta, [https://www.document-center.com/standards/show/ASTM-E74, 17.02.2023](https://www.document-center.com/standards/show/ASTM-E74,%2017.02.2023).

[15] S interneta, https://dewesoft.com/daq/guide-to-fft-analysis , 17.02.2023.

[16] S interneta, <https://link.springer.com/referenceworkentry/10.1007/978-0-387-32833-1_344>, 17.02.2023.

[17] S interneta, <https://weighing.andonline.com/sites/default/files/documents/Selecting%20Weights%20and%20Certificates_1.pdf> , 20.02.2023.

[18] S interneta, <https://www.vernier.com/files/manuals/gdx-hd/gdx-hd.pdf> , 24.02.2023.

# Popis slika

[Slika 2.1 Kalibracijske sile od 20N, 50N, 150N i 550 N 7](https://d.docs.live.net/69d3b2950e9a9cba/Radna%20površina/UMSG3_2023_britvic%20i%20stef%20dio%20rjesen%20(1).docx" \l "_Toc127409800)

[Slika 2.2 Vernier go direct dinamometar 8](#_Toc127409804)

[Slika 2.3 Graf polinomne aproksimacije 12](#_Toc127409805)

[Slika 3.1 Blok dijagram Shimmer3 senzora [1]. 17](#_Toc127409806)

[Slika 3.2 Na slici je vidljivo da manja brzina uzorkovanja podataka lošije aproksimira ulazni signal, a veća bolje. [2] 18](#_Toc127409807)

[Slika 4.1 Flexor Digitorum Profundus. Najutjecajniji mišić pri stisku šake [10]. 19](#_Toc127409808)

[Slika 4.2 Pozicija 1 20](#_Toc127409809)

[Slika 4.3 Pozicija 2. 20](#_Toc127409810)

[Slika 4.4 Pozicija 3. 20](#_Toc127409811)

[Slika 4.5 Pozicija 4. 21](#_Toc127409812)

[Slika 4.6 Pozicija 5. 21](#_Toc127409813)

[Slika 5.1. Shematski prikaz procesa [15] 26](#_Toc127409814)

[Slika 5.2. Rolling window [9] 27](#_Toc127409815)

[Slika 5.3. Odziv mišića podlaktice prilikom stiska šake: mjerene sile u N pomoću dinamometra (narančasto), mjerenje pomoću EMG osjetnika (plavo) 27](#_Toc127409816)

[Slika 5.4. Prikazuje izmjenu poruka između čvorova (eng. Nodes) i teme (eng. Topic). U našem slučaju događa se komunikacija između računala, dinamometra i senzora. 28](#_Toc127409816)

[Slika 5.5. Primjer neporavnatih podataka 28](#_Toc127409817)

[Slika 5.6 Poravnati parametri 29](#_Toc127409818)

[Slika 5.7 Ovisnost maksimalne korelacije o faktoru zaglađivanja, pomičnom prozoru i vremenu kašnjenja za jedan set podataka 31](#_Toc127409819)

[Slika 5.8 Signali za optimalne parametre nefiltrirani signal, poravnati signal, zaglađeni signal i stisak šake 32](#_Toc127409820)

[Slika 5.9 Vektor od 250 nasumičnih generiranih vrijednosti od 0 do 1 dok je nulta pozicija uvijek 0 33](#_Toc127409821)

|  |  |
| --- | --- |
| Tablica doprinosa | |
| Luciano Kostelac | Umjeravanje dinamometra, provedba ispitivanja, spajanje seminara, |
| Luka Zvonarek | Obrada podataka, umjeravanje dinamometra, provedba ispitivanja |
| Mato Štefanac | Obrada podataka, umjeravanje dinamometra, provedba ispitivanja |
| Krunoslav Marenić | sEMG, odabir elektroda |
| Ivan Britvić | Dinamometar, odabir elektroda |

# 10. Kod

|  |  |
| --- | --- |
| 1  2  3  4 | lin\_coeff: 1.095545806  square\_coeff: -0.000302451  cubic\_coeff: 0.00000141565  fourth\_ord\_coeff: -0.00000000173619 |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | **import** **scipy.fft**  **import** **numpy** **as** **np**  **import** **scipy.signal**  **import** **random**  **import** **matplotlib.pyplot** **as** **plt**  **import** **seaborn** **as** **sns**  **import** **matplotlib** **as** **mpl**  **import** **matplotlib.pyplot** **as** **plt**  **import** **glob**  **import** **os**  **import** **pandas** **as** **pd**  **import** **itertools** |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | *# kod za sve da povleče:*  print('Get current working directory : ', os.getcwd()) *#help function*  *# cvs\_files = glob.glob("D:\\Luka\\Documents\\Riteh\\UMS\\new\_data\\\*.csv")*  cvs\_files = glob.glob("new\_data/\*.csv")  print(cvs\_files)  *# FFT treba napraviti na prozorima od 0.5s, kao što smo i u real-time-u s ROS-om radili*  *# Calculate window size for FFT - always round to nearest even number*  fft\_relative\_window\_size = 0.5  *# Geenriranje parametara za optimizaciju*  rolling\_window\_relative\_size = [i/100 **for** i **in** range(1, 50)]  *# smoothing\_factor = [i/1000 for i in range(50)]*  smoothing\_factor = [i/1000 **for** i **in** range(10)]  *#HARD\_CODING ()*  *# Ovo je hard-coded (inače treba iz datoteke očitati)*  fft\_dim = 249  display(f'{fft\_dim=}')  *# Sampling rate - median daje bolje rezultate od prosjeka*  sampling\_rate = 992.969696969697  display(f'{sampling\_rate=}')  *# Zaokružimo na najbliži cijeli broj - mora biti cijeli broj podataka pod prozorom*  fft\_window\_size = round(fft\_relative\_window\_size \* sampling\_rate)  *# Parni broj točaka za FFT - 496 po mom izračunu ((990 / 2 = 495) + 1)*  fft\_window\_size += (fft\_window\_size % 2)  display(f'{fft\_window\_size=}')  *# FFT resolution*  fft\_resolution = sampling\_rate / fft\_window\_size  emg\_fft\_grip\_dat\_file = []  **for** file **in** cvs\_files:  datoteka=np.loadtxt(file, delimiter=",", skiprows=1)  emg\_dat = datoteka[:, 1]  grip\_dat = datoteka[:, 2]  time\_dat = datoteka[:, 0]  *# Data length*  dat\_len = time\_dat.shape[0]  *# display(f'{dat\_len=}')*  *######################*  *# Ukupan broj podataka mora biti višekratnik veličine prozora da bi obrada funkcionirala*  *# U real-time-u se podaci tako i obrađuju (kad ih se skupi 496)*  *# Reshape data to window\_size × n array (disregard starting few points)*  time\_dat = time\_dat[dat\_len%fft\_window\_size:].reshape((-1, fft\_window\_size))  emg\_dat = emg\_dat[dat\_len%fft\_window\_size:].reshape((-1, fft\_window\_size))  grip\_dat = grip\_dat[dat\_len%fft\_window\_size:].reshape((-1, fft\_window\_size))  *# Calculate RFFT on emg\_data*  emg\_fft = scipy.fft.rfft(emg\_dat)  emg\_fft\_grip\_dat\_file.append((file, emg\_fft, grip\_dat))  *# emg\_fft\_grip\_dat\_file* |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62 | roll\_wind\_size = []  smooth\_fact = []  mean\_max\_corrs = []  all\_max\_corrs = []  fft\_mask = []  **for** params **in** itertools.product(rolling\_window\_relative\_size, smoothing\_factor):  *# Random 0 - 1, 249 kom*  optimization\_params = np.random.random(fft\_dim)  optimization\_params = np.append(optimization\_params, params)  optimization\_params[0] = 0 *#uvijek nultu vrednost maknem skroz*  *#display(f'{params=}')*  *# display(f'{optimization\_params[:fft\_dim]=}')*  max\_corrs = []  file\_name = []  **for** file, emg\_fft, grip\_dat **in** emg\_fft\_grip\_dat\_file:  emg\_ifft = scipy.fft.irfft(emg\_fft \* optimization\_params[:fft\_dim])  rolling\_window\_size = round(optimization\_params[fft\_dim] \* sampling\_rate)  emg\_abs = np.abs(emg\_ifft.ravel())  window\_ema = np.array([(1-optimization\_params[fft\_dim+1])\*\*i **for** i **in** range(rolling\_window\_size)])  window\_ema = window\_ema / window\_ema.sum()  emg\_ema = scipy.signal.fftconvolve(emg\_abs, window\_ema[::-1], mode='valid')  grip\_flat = grip\_dat.ravel()[rolling\_window\_size-1:]  emg\_ema = emg\_ema - emg\_ema.mean()  grip\_flat = grip\_flat - grip\_flat.mean()  corr\_ema = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')  corr\_ema /= (len(grip\_flat) \* emg\_ema.std() \* grip\_flat.std())  lags = scipy.signal.correlation\_lags(len(emg\_ema), len(grip\_flat), mode='full')  max\_corr\_lag\_ema = lags[np.argmax(np.abs(corr\_ema))]  max\_corr = np.abs(corr\_ema).max()  max\_corrs.append(max\_corr)  file\_name.append(file)  *# Calculate mean max correlation*  mean\_max\_corrs.append(np.mean(max\_corrs))  all\_max\_corrs.append(max\_corrs)  roll\_wind\_size.append(params[0])  smooth\_fact.append(params[1])  fft\_mask.append(optimization\_params[:fft\_dim])    max\_corr\_ind = np.argmax(mean\_max\_corrs)  display(f'{max\_corr\_ind=}')  display(f'{mean\_max\_corrs[max\_corr\_ind]=}')  display(f'{roll\_wind\_size[max\_corr\_ind]=}')  display(f'{smooth\_fact[max\_corr\_ind]=}')  display(f'{all\_max\_corrs[max\_corr\_ind]}')  display(f'{fft\_mask[max\_corr\_ind]=}')  df=pd.DataFrame({  "Max\_corrs mean": mean\_max\_corrs[max\_corr\_ind],  "Rolling window size": roll\_wind\_size[max\_corr\_ind],  "Smoothing factor":smooth\_fact[max\_corr\_ind],  "All\_corrs": all\_max\_corrs[max\_corr\_ind],  "File": file\_name  })  df.to\_csv("Rezultati\_Bazina.csv", index=**False**)  df = pd.DataFrame({  "FFT maska": fft\_mask[max\_corr\_ind]  })  df.to\_csv("Rezultati\_FFT\_Bazina.csv", index=**False**) |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84 | *#ponovno vrtimo kod za sve podatke*  print('Get current working directory : ', os.getcwd()) *#help function*  *# cvs\_files = glob.glob("D:\\Luka\\Documents\\Riteh\\UMS\\new\_data\\\*.csv")*  cvs\_files = glob.glob("new\_data/\*.csv")  print(cvs\_files)  array\_for\_output1 = []  array\_for\_output2 =[]  array\_for\_output3 =[]  array\_for\_output4 = []  max\_corrs\_lags = []  max\_corrs = []  rolling\_window\_size=roll\_wind\_size[max\_corr\_ind]  smoothing\_factor =smooth\_fact[max\_corr\_ind]  optimization\_params=fft\_mask[max\_corr\_ind]  *# optimization\_params[0]=0*  **for** file **in** cvs\_files:  datoteka=np.loadtxt(file, delimiter=",", skiprows=1)  emg\_dat = datoteka[:, 1]  grip\_dat = datoteka[:, 2]  time\_dat = datoteka[:, 0]  *# Data length*  dat\_len = time\_dat.shape[0]  display(dat\_len)  *# Ispis*  display(time\_dat)  display(emg\_dat)  display(grip\_dat)  *# Sampling rate - median daje bolje rezultate od prosjeka*  sampling\_rate = 1./np.median(np.diff(time\_dat[10:-10])) *# iz ROS skripte*  display(sampling\_rate)  *######################*  *# FFT treba napraviti na prozorima od 0.5s, kao što smo i u real-time-u s ROS-om radili*  *# Calculate window size for FFT - always round to nearest even number*  fft\_relative\_window\_size = 0.5  *# Zaokružimo na najbliži cijeli broj - mora biti cijeli broj podataka pod prozorom*  fft\_window\_size = round(fft\_relative\_window\_size \* sampling\_rate)  *# Parni broj točaka za FFT - 496 po mom izračunu ((990 / 2 = 495) + 1)*  fft\_window\_size += (fft\_window\_size % 2)  display(f'fft\_window: {fft\_window\_size}')  *# FFT resolution*  fft\_resolution = sampling\_rate / fft\_window\_size  display(f'fft\_resolution: {fft\_resolution}')  *# Ukupan broj podataka mora biti višekratnik veličine prozora da bi obrada funkcionirala*  *# U real-time-u se podaci tako i obrađuju (kad ih se skupi 496)*  *# Reshape data to window\_size × n array (disregard starting few points)*  time\_dat = time\_dat[dat\_len%fft\_window\_size:].reshape((-1, fft\_window\_size))  emg\_dat = emg\_dat[dat\_len%fft\_window\_size:].reshape((-1, fft\_window\_size))  grip\_dat = grip\_dat[dat\_len%fft\_window\_size:].reshape((-1, fft\_window\_size))  *# Calculate RFFT on emg\_data*  emg\_fft = scipy.fft.rfft(emg\_dat)  *# Ispis oblika matrica*  display(f'time: {time\_dat.shape}')  display(f'emg\_dat: {emg\_dat.shape}')  display(f'grip\_dat: {grip\_dat.shape}')  display(f'emg\_fft: {emg\_fft.shape}')  fft\_dim = emg\_fft.shape[1] *# koliko je dugačak redak u fft*  print (f'fft\_dim: {fft\_dim}')  emg\_ifft = scipy.fft.irfft(emg\_fft \* optimization\_params[:fft\_dim])  rolling\_window\_size = round(rolling\_window\_size \* sampling\_rate)  emg\_abs = np.abs(emg\_ifft.ravel())  window\_ema = np.array([(1-optimization\_params[fft\_dim+1])\*\*i **for** i **in** range(rolling\_window\_size)])  window\_ema = window\_ema / window\_ema.sum()  emg\_ema = scipy.signal.fftconvolve(emg\_abs, window\_ema[::-1], mode='valid')  grip\_flat = grip\_dat.ravel()[rolling\_window\_size-1:]  emg\_ema = emg\_ema - emg\_ema.mean()  grip\_flat = grip\_flat - grip\_flat.mean()  corr\_ema = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')  corr\_ema /= (len(grip\_flat) \* emg\_ema.std() \* grip\_flat.std())  lags = scipy.signal.correlation\_lags(len(emg\_ema), len(grip\_flat), mode='full')  max\_corr\_lag\_ema = lags[np.argmax(np.abs(corr\_ema))]  max\_corr = np.abs(corr\_ema).max()  max\_corrs\_lags.append(max\_corr\_lag\_ema)  max\_corrs.append(max\_corr)    array\_for\_output1.append(np.max(max\_corrs))    array\_for\_output3.append(max\_corrs\_lags[0])  df=pd.DataFrame({"Max\_corrs": array\_for\_output1, "Lagovi": array\_for\_output3})  df.to\_csv ("Rezultati2.csv", index=**False**) |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | **for** params **in** itertools.product(rolling\_window\_relative\_size, smoothing\_factor):  i += 1  *# Random 0 - 1, 250 kom*  optimization\_params = np.random.random(fft\_dim)  optimization\_params = np.append(optimization\_params, params)  optimization\_params[0] = 0 *#uvijek nultu vrednost maknem skroz*  emg\_ifft = scipy.fft.irfft(emg\_fft \* optimization\_params[:fft\_dim])  rolling\_window\_size = round(optimization\_params[fft\_dim] \* sampling\_rate)  emg\_abs = np.abs(emg\_ifft.ravel())  window\_ema = np.array([(1-optimization\_params[fft\_dim+1])\*\*i **for** i **in** range(rolling\_window\_size)])  window\_ema = window\_ema / window\_ema.sum()  emg\_ema = scipy.signal.fftconvolve(emg\_abs, window\_ema[::-1], mode='valid')  grip\_flat = grip\_dat.ravel()[rolling\_window\_size-1:]  emg\_ema = emg\_ema - emg\_ema.mean()  grip\_flat = grip\_flat - grip\_flat.mean()  corr\_ema = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')  corr\_ema /= (len(grip\_flat) \* emg\_ema.std() \* grip\_flat.std())  lags = scipy.signal.correlation\_lags(len(emg\_ema), len(grip\_flat), mode='full')  max\_corr\_lag\_ema = lags[np.argmax(np.abs(corr\_ema))]  max\_corr = np.abs(corr\_ema).max()  max\_corrs\_lags.append(max\_corr\_lag\_ema)  **if** max\_corrs:  **if** max\_corr > np.max(max\_corrs):  optimal\_params = optimization\_params  max\_corrs.append(max\_corr)  roll\_wind\_size.append(params[0])  smooth\_fact.append(params[1])  array\_for\_output1.append(np.max(max\_corrs))  array\_for\_output2.append(optimal\_params[-1])  array\_for\_output3.append(max\_corrs\_lags[0])  array\_for\_output4.append(optimal\_params[-2])  *#print(array\_for\_output1)*  *#print(array\_for\_output2)*  *#print(array\_for\_output3)*  *#print(array\_for\_output4)*  prosjek2 = np.average(array\_for\_output2[0])  prosjek4 = np.average(array\_for\_output4[0])  *#np.savetxt("rez.csv", array\_for\_output1, header= ,delimiter=",")*  *#samo pandas*  df=pd.DataFrame({"Max\_corrs": array\_for\_output1, "Rolling window size": array\_for\_output4, "Lagovi": array\_for\_output3, "Smoothing factor":array\_for\_output2,"Prosjek roll wind size":prosjek4, "Prosjek smth fact": prosjek2})  df.to\_csv ("Rezultati.csv", index=**False**) |

|  |  |
| --- | --- |
| 1  2  3  4  5 | lin\_coeff = rospy.get\_param('~lin\_coeff')  square\_coeff = rospy.get\_param('~square\_coeff')  cubic\_coeff = rospy.get\_param('~cubic\_coeff')  fourth\_ord\_coeff = rospy.get\_param('~fourth\_ord\_coeff')  gdx\_hd.read(publisher=pub, measurement\_type=measurement\_type, lin\_coeff=lin\_coeff, square\_coeff=square\_coeff, cubic\_coeff=cubic\_coeff, fourth\_ord\_coeff=fourth\_ord\_coeff) |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | **import** **glob**  **import** **os**  **import** **pandas** **as** **pd**  **import** **scipy.fft**  **import** **numpy** **as** **np**  **import** **scipy.signal**  **import** **random**  **import** **matplotlib.pyplot** **as** **plt**  **import** **seaborn** **as** **sns**  **import** **matplotlib** **as** **mpl**  **import** **matplotlib.pyplot** **as** **plt**  print('Get current working directory : ', os.getcwd()) *#help function*  cvs\_files = glob.glob("D:**\\**Luka**\\**Documents**\\**Riteh**\\**UMS**\\**new\_data**\\**\*.csv")  print(cvs\_files)  array\_for\_output1 = []  array\_for\_output2 =[]  array\_for\_output3 =[]  array\_for\_output4 = []  prosjek2 = []  prosjek4 = []  **import** **itertools**  rolling\_window\_relative\_size = [i/100 **for** i **in** range(1, 50)]  smoothing\_factor = [i/1000 **for** i **in** range(50)]  *#display(len(rolling\_window\_relative\_size))*  *#display(len(smoothing\_factor)* |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | **import** **itertools**  rolling\_window\_relative\_size = [i/100 **for** i **in** range(1, 50)]  smoothing\_factor = [i/1000 **for** i **in** range(50)]  *#display(len(rolling\_window\_relative\_size))*  *#display(len(smoothing\_factor))*    max\_corrs\_lags = []  max\_corrs = []  roll\_wind\_size = []  smooth\_fact = []  i = 0  **for** params **in** itertools.product(rolling\_window\_relative\_size, smoothing\_factor):  i += 1  *# Random 0 - 1, 250 kom*  optimization\_params = np.random.random(fft\_dim)  optimization\_params = np.append(optimization\_params, params)  optimization\_params[0] = 0 *#uvijek nultu vrednost maknem skroz*  emg\_ifft = scipy.fft.irfft(emg\_fft \* optimization\_params[:fft\_dim])  rolling\_window\_size = round(optimization\_params[fft\_dim] \* sampling\_rate)  emg\_abs = np.abs(emg\_ifft.ravel())  window\_ema = np.array([(1-optimization\_params[fft\_dim+1])\*\*i **for** i **in** range(rolling\_window\_size)])  window\_ema = window\_ema / window\_ema.sum()  emg\_ema = scipy.signal.fftconvolve(emg\_abs, window\_ema[::-1], mode='valid')  grip\_flat = grip\_dat.ravel()[rolling\_window\_size-1:]  emg\_ema = emg\_ema - emg\_ema.mean()  grip\_flat = grip\_flat - grip\_flat.mean()  corr\_ema = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')  corr\_ema /= (len(grip\_flat) \* emg\_ema.std() \* grip\_flat.std())  lags = scipy.signal.correlation\_lags(len(emg\_ema), len(grip\_flat), mode='full')  max\_corr\_lag\_ema = lags[np.argmax(np.abs(corr\_ema))]  max\_corr = np.abs(corr\_ema).max()  max\_corrs\_lags.append(max\_corr\_lag\_ema)  **if** max\_corrs:  **if** max\_corr > np.max(max\_corrs):  optimal\_params = optimization\_params  max\_corrs.append(max\_corr)  roll\_wind\_size.append(params[0])  smooth\_fact.append(params[1])  *# if i < 5:*  *# display(optimization\_params)* |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | display(f'Max correlation: {np.max(max\_corrs)}')  display(f'Lag at amx corr: {max\_corrs\_lags[np.argmax(max\_corrs)]/sampling\_rate}')  display(f'Optimal rolling\_window\_relative\_size: {optimal\_params[-2]}')  display(f'Optimal smoothing factor: {optimal\_params[-1]}')  sns.set\_theme(context='paper', style='whitegrid', rc={})  sns.despine()  fig, ax = plt.subplots(3, 1)  fig.set\_size\_inches(w=6., h=6.)  sns.scatterplot(  x=smooth\_fact,  y=max\_corrs,  size=2.,  legend=**None**,  ax=ax[0]  )  ax[0].set\_title('Correlation - Smoothing factor', fontsize='small', fontweight='bold')  sns.scatterplot(  x=roll\_wind\_size,  y=max\_corrs,  size=2.,  legend=**None**,  ax=ax[1]  )  ax[1].set\_title('Correlation - Rolling window size', fontsize='small', fontweight='bold')  sns.scatterplot(  x=max\_corrs\_lags/sampling\_rate,  y=max\_corrs,  size=2.,  legend=**None**,  ax=ax[2]  )  ax[2].set\_title('Correlation - Time lag', fontsize='small', fontweight='bold')  *# plt.plot(max\_corrs\_lags, max\_corrs, 'o', size=2.)*  *# plt.grid(which='both')*  plt.suptitle(t=r'Parameter optimization', fontsize='small')  plt.tight\_layout()  fig.savefig('optimizacija.png', dpi=320)  plt.show() |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63 | *# Attenuation/amplification of single frequencies*  *# mijenajati FFT frekvencije množeći s vrijednostima [0, 1.5 - pojačanje]*  *# Kod vas je malo manji sampling rate pa umjesto 250 ima 249 frekvencija*  *# optimization\_params = np.random.random(250)*  *# optimization\_params = np.random.random(fft\_dim)*  *# EMA relative window size - u sekundama + ovaj random.random zapravo svaki*  *# put kad pokrenete kod malo poveća/smanji rolling window jer ga pomnoži s*  *# random u rasponu (0, 1)*  *# rolling\_window\_relative\_size = 0.25\*random.random()*  *# optimization\_params = np.append(optimization\_params, rolling\_window\_relative\_size)*  *# EMA smoothing factor - isto potebno optimirati, svaki put je malo drugačiji faktor*  *# zbog množenja*  *# smoothing\_factor = 0.025 \* random.random()*  *# optimization\_params = np.append(optimization\_params, smoothing\_factor)*  *# DC offset always removed*  *# optimization\_params[0] = 0 #uvijek nultu vrednost maknem skroz*  *# Optimal parameters after loop*  optimization\_params = optimal\_params  *# emg\_fft se pomnoži s nekim brojevima [0, 1.5] da se zapravo filtrira signal*  *# i onda ide inverz da se rekonstruira početni signal + nekakvo filtriranje*  emg\_ifft = scipy.fft.irfft(emg\_fft \* optimization\_params[:fft\_dim])  *# emg\_ifft = scipy.fft.irfft (np.dot(emg\_fft[:,None],optimization\_params[None,:fft\_dim]))*  display(f'emg\_ifft: {emg\_ifft.shape}')  *# Odredi se duljina prozora za EMA pomoću relativne duljine (u sekundama) i sampling*  *# rate-a*  rolling\_window\_size = round(optimization\_params[fft\_dim] \* sampling\_rate)  *# Ako je slučajno više od prozora za fft, zaokruži se na tu veličinu*  display(f'rolling\_window\_size: {rolling\_window\_size}')  **if** rolling\_window\_size > fft\_window\_size:  rolling\_window\_size = fft\_window\_size  *# Apsolutna vrijednost emg signala nakon filtriranja*  *# Ravel ga spljošti u 1D signal (spoji sve retke matrice u jedan)*  emg\_abs = np.abs(emg\_ifft.ravel())  *# EMA window - težine s kojima se množi signal*  window\_ema = np.array([(1-optimization\_params[fft\_dim+1])\*\*i **for** i **in** range(rolling\_window\_size)])  *# Normalizacija težina pomoću sume (suma svih težina = 1)*  window\_ema = window\_ema / window\_ema.sum()  *# EMA zaglađen emg signal (filtriranje + apsolutna + zaglađivanje)*  emg\_ema = scipy.signal.fftconvolve(emg\_abs, window\_ema[::-1], mode='valid')  *# Plot*  plt.figure(figsize=(8, 6), dpi=320)  plt.subplot(4, 1, 1)  *# Original input signal*  plt.plot(time\_dat.ravel(), emg\_dat.ravel())  plt.grid(which='both')  plt.subplot(4, 1, 2)  *# Nakon filtriranja + rekostrukcije*  plt.plot(time\_dat.ravel(), emg\_ifft.ravel())  plt.grid(which='both')  plt.subplot(4, 1, 3)  *# Nakon apsolutne vrijednosti + zaglađivanja*  plt.plot(time\_dat.ravel()[rolling\_window\_size-1:], emg\_ema)  plt.grid(which='both')  plt.subplot(4, 1, 4)  *# Grip za usporedbu*  plt.plot(time\_dat.ravel()[rolling\_window\_size-1:], grip\_dat.ravel()[rolling\_window\_size-1:])  plt.grid(which='both')  plt.show() |

|  |  |
| --- | --- |
| 1  2  3  4  5 | *# Vi zapravo optimirate ovih prvih 249 vrijednosti + jedna za rolling window size + jedna za smoothing factor*  *# Možete napraviti petlju ili nešto slično da ih pretražuje*  display(optimization\_params[:fft\_dim].shape)  display(optimization\_params[fft\_dim])  display(optimization\_params[fft\_dim+1]) |

|  |  |
| --- | --- |
| 1  2  3 | display(time\_dat.ravel()[rolling\_window\_size-1:].shape)  display(emg\_ema.shape)  display(grip\_dat.ravel()[rolling\_window\_size-1:].shape) |

|  |  |
| --- | --- |
| 1  2  3  4 | grip\_flat = grip\_dat.ravel()[rolling\_window\_size-1:]  *# Demean signals for*  emg\_ema = emg\_ema - emg\_ema.mean()  grip\_flat = grip\_flat - grip\_flat.mean() |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | *# corr\_ema = scipy.signal.correlate(emg\_ema, grip\_flat, method='fft', mode='full')*  *# display(corr\_ema)*  corr\_ema = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')  corr\_ema /= (len(grip\_flat) \* emg\_ema.std() \* grip\_flat.std())  *# display(corr\_ema)*  lags = scipy.signal.correlation\_lags(len(emg\_ema), len(grip\_flat), mode='full')  print(f'EMA - GRIP:')  print(scipy.stats.pearsonr(emg\_ema, grip\_flat)[0])  max\_corr\_lag\_ema = lags[np.argmax(np.abs(corr\_ema))]  print(f'EMA -> Corr at 0 lag: {corr\_ema[lags == 0]}')  print(f'EMA -> Max corr: {np.abs(corr\_ema).max()}, Lag: {max\_corr\_lag\_ema} or {max\_corr\_lag\_ema/sampling\_rate} sec')  plt.figure(figsize=(8, 6), dpi=320)  plt.plot(lags, corr\_ema)  plt.legend(['EMA - GRIP'])  plt.grid(which='both')  plt.show() |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | *# %timeit corr\_ema = scipy.signal.correlate(emg\_ema, grip\_flat, method='fft', mode='full')*  corr\_ema1 = scipy.signal.correlate(emg\_ema, grip\_flat, method='fft', mode='full')  display(corr\_ema)  *# %timeit corr\_ema = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')*  corr\_ema2 = scipy.signal.fftconvolve(emg\_ema, grip\_flat[::-1], mode='full')  display(corr\_ema)  *# %timeit corr\_ema = np.convolve(emg\_ema, grip\_flat[::-1])*  corr\_ema3 = np.convolve(emg\_ema, grip\_flat[::-1])  display(corr\_ema)  *# %timeit corr\_ema = np.correlate(emg\_ema, grip\_flat, mode='full')*  corr\_ema4 = np.correlate(emg\_ema, grip\_flat, mode='full')  np.allclose(corr\_ema1, corr\_ema4) |

# Prilog

1.

2. Podaci dobiveni RDB eksperimentom

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

2. Podaci dobiveni RDB eksperimentom

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | X1 = 1 | X1 = 2 |  |  |
|  | j | Pozicija 1 | Pozicija 2 | Yj | Bj |
| X2 = 1 | S1 | 0,888365 | 0,901968 | 0,927541 | -0,0202 |
|  |  | 0,951445 | 0,959487 |  |  |
|  |  | 0,957508 | 0,90647 |  |  |
| X2 = 2 | S2 | 0,906292 | 0,964466 | 0,941787 | -0,00595 |
|  |  | 0,905572 | 0,952191 |  |  |
|  |  | 0,966331 | 0,95587 |  |  |
| X2 = 3 | S3 | 0,941162 | 0,958664 | 0,95128 | 0,003541 |
|  |  | 0,958844 | 0,969771 |  |  |
|  |  | 0,910035 | 0,969203 |  |  |
| X2 = 4 | S4 | 0,948201 | 0,976617 | 0,96527 | 0,017531 |
|  |  | 0,956579 | 0,974684 |  |  |
|  |  | 0,969693 | 0,965846 |  |  |
| X2 = 5 | S5 | 0,962822 | 0,949054 | 0,957996 | 0,010257 |
|  |  | 0,97041 | 0,959686 |  |  |
|  |  | 0,955415 | 0,950591 |  |  |
| X2 = 6 | S6 | 0,955301 | 0,909735 | 0,942435 | -0,0053 |
|  |  | 0,96015 | 0,964407 |  |  |
|  |  | 0,93576 | 0,929256 |  |  |
| X2 = 7 | S7 | 0,956043 | 0,972525 | 0,931606 | -0,01613 |
|  |  | 0,938036 | 0,903132 |  |  |
|  |  | 0,878056 | 0,941842 |  |  |
| X2 = 8 | S8 | 0,975598 | 0,983074 | 0,960006 | 0,012267 |
|  |  | 0,947281 | 0,955947 |  |  |
|  |  | 0,953289 | 0,944846 |  |  |
| X2 = 9 | S9 | 0,948521 | 0,941208 | 0,950618 | 0,002879 |
|  |  | 0,953454 | 0,970735 |  |  |
|  |  | 0,92006 | 0,969728 |  |  |
| X2 = 10 | S10 | 0,975409 | 0,942812 | 0,948851 | 0,001113 |
|  |  | 0,963344 | 0,939349 |  |  |
|  |  | 0,974814 | 0,897381 |  |  |
|  | Yi | 0,946126 | 0,949352 |  |  |
|  | Ti | -0,00161 | 0,001613 |  |  |