Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Anže Rezelj

Razvoj nizkocenovnega lahkega robotskega manipulatorja

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Danijel Skočaj

Ljubljana, 2017

Avtorske pravice. Rezultati magistrskega dela so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko Univerze v Ljubljani. Za objavljanje ali izkoriščanje rezultatov magistrskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

©2017 Anže Rezelj

Zahvala

Zahvaljujem se mentorju dr. Danijelu Skočaju za vodstvo in napotke pri izdelavi naloge, asistentu dr. Luki Čehovinu Zajcu za napotke in mnenja, svojim bljižnjim za vztrajno podporo in članom laboratorija ViCoS za mnenja in produktivno okolje, v katerem je delo nastalo.

Anže Rezelj, 2017

Kazalo

Povzetek

Naslov: Razvoj nizkocenovnega lahkega robotskega manipulatorja

Robotika postaja vse pomembnejša znanstvena in inženirska disciplina. Z rastočim trgom lahko pričakujemo, da se bo še povišalo povpraševanje po robotsko izobraženem kadru. Da bi olajšali dostop do (običajno zelo drage) robotske tehnologije, smo pristopili k izdelavi nizkocenovnega, a funkcionalno dovolj zmogljivega robotskega manipulatorja. Pri izdelavi smo se oprli na obstoječe rešitve, jih posodobili, nadgradili in razširili njihovo funkcionalnost. Pri izdelavi manipulatorja smo uporabili že obstoječe in cenovno dostopne komponente. Končni izdelek predstavlja celovit sistem, ki zajema robotski manipulator in manjši računalnik z vso programsko opremo. Sistem omogoča preprosto upravljanje in programiranje robotskega manipulatorja, je varen za uporabo in dovolj natančen za uporabo v pedagoške in sorodne namene.

Ključne besede

Robotski manipulator, robotika, kinematika, OpenServo, Manus, odprta koda in strojna oprema.

Abstract

Title: Development of a low-cost lightweight robot manipulator

Robotics is becoming an increasingly important scientific and engineering discipline. With the growing robotics market, it is expected that the demand for robotically educated personnel will even increase. To widen the access to the (usually very expensive) robotics technology, we decided to develop a low-cost and lightweight but functionally sufficiently powerful robot manipulator. We based our work on existing solutions; we updated and upgraded these solutions, as well as extended their functionality. In the production of the robot manipulator, we used already existing and affordable components. The end product is a complete system that includes a robotic manipulator and a small computer with all the required software. The system is easy to use and enable the user to easily program the robot manipulator. Moreover, it is safe to operate with and sufficiently precise for use in educational and similar scenarios.

Keywords

Robot manipulator, robotics, kinematics, OpenServo, Manus, open source and open hardware.

Poglavje 1

Uvod

1.1 Motivacija

Robotika postaja vse bolj priljubljena in pomembna znanstvena in inženirska disciplina in utemeljeno lahko pričakujemo, da se bo povpraševanje po robotikih in robotsko ozaveščenih kadrih le še povečalo. Zato je treba o robotiki podučiti čim več (predvsem mladih) ljudi.

Robotika je dolgo veljala za zelo zahtevno in drago tehnologijo, ki se uporablja le v industriji, kjer so v uporabi različni robotski manipulatorji za opravljanje težkih, nevarnih in zahtevnih opravil. Toda temu počasi ni več tako. Z napredkom tehnologije, pocenitvijo in lažje dostopnimi komponentami ter s pomočjo 3D tiskalnika se robotika vztrajno pomika izven industrije. Je tudi zelo atraktivna disciplina, kjer imamo opravka s fizičnimi roboti, ki so lahko opremljeni s senzorji za zaznavo okolja in interakcijo z ljudmi. Veliko raziskovalcev se je posvetilo izdelavi dostopnejše rešitve [1, 2, 3, 4, 5]. Dela se primarno osredotočajo na inženirsko plat razvoja, kot so cena in vrsta sestavnih delov ter preprosto in nezahtevno nizkonivojsko krmiljenje. Cena takih sistemov se giblje od 4000 €, kar je še vedno cenovni zalogaj, pa vse do 50 €, pri čemer so slednje rešitve nezanesljive in nenatančne.

Naš cilj je bil združiti vse dobre lastnosti zgoraj omenjenih obstoječih rešitev, jih nadgraditi in izdelati lahek nizkocenovni robotski manipulator.

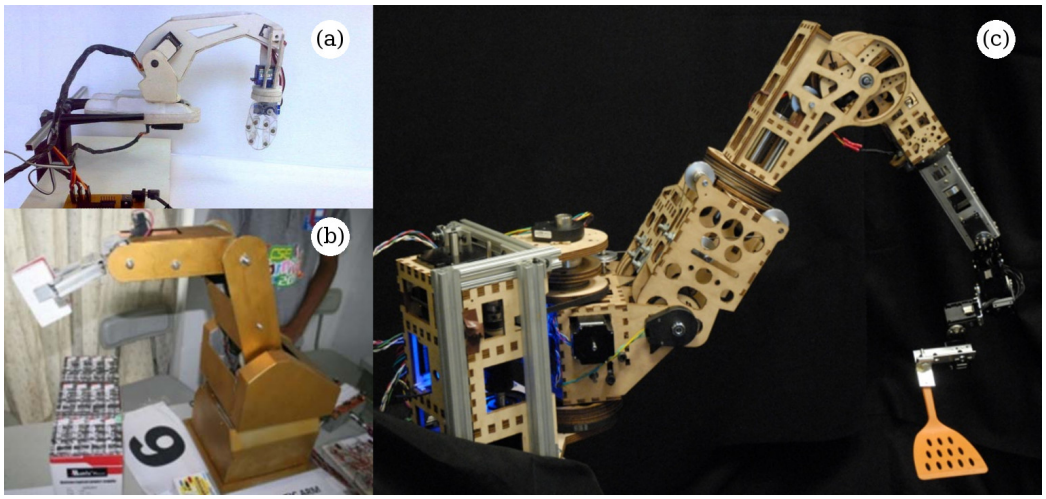
Te lastnosti so: preprostost izdelave, preprostost in varnost uporabe, zadostna prilagodljivost, robustnost in natančnost ter cenovna dostopnost. Veliko pozornosti smo posvetili nizki ceni izdelave, da si lahko robotski manipulator izdela čim širša robotsko navdušena populacija in se v izobraževalne namene omogoči dostop do dovolj robotskih manipulatorjev.

1.2 Pregled sorodnih del

Področje izdelave robotskega manipulatorja je v literaturi opisano pretežno dobro, saj je področje dejavno že nekaj desetletij in posledično obstaja veliko idej in rešitev, kako zgraditi robotski manipulator. Tako se pri izdelavi nizko-cenovnega robotskega manipulatorja pojavlja veliko rešitev. Pri izrazu "*nizko-cenovni*"moramo biti previdni, saj se uporablja praktično pri vsakem izdelanem robotskem manipulatorju, katerega cena izdelave znaša manj kakor enakovredno profesionalno izdelan robotski manipulator. Izdelave takega robotskega manipulatorja, ki je zelo zmogljiv in natančen, vendar njegova izdelava stane nekaj tisoč evrov, so se lotili avtorji članka [1] (slika 1.1 (c)). Za pogon uporablja koračne motorje, ki so sorazmerno veliki in težki, za delovanje pa potrebujejo zmogljiv napajalnik. Roka je tako primernejša za raziskovalne namene, kjer se zares lahko izrazi njena zmogljivost. Za izobraževalne namene, na primer pri predmetih, kjer se študentje prvič srečajo z robotiko, je tak robotski manipulator velik finančni zalogaj. To se hitro opazi, ko želimo, da študentje delajo z robotskim manipulatorjem samostojno ali v parih in jih želimo izdelati več. Hkrati tudi ni potrebe po zelo veliki natančnosti.

Izdelave cenovno še dostopnejšega robotskega manipulatorja so se lotili avtorji [2, 3, 4, 5] (slika 1.1 (a) in (b)). Krmilno enoto so zaupali mikrokrmilniku, na primer ATmega328, ali kar razvojni ploščici Arduino UNO [1], ki vsebuje prej omenjeni mikrokrmilnik in poenostavi njegovo programiranje. Motorji so nato ustrezno krmiljeni z uporabo namenskih močnostnih tranzi-

[1] https://www.arduino.cc/



Slika 1.1: (a) Robotski manipulator za učne namene [2], (b) cenejši večji robotski manipulator [3], (c) zmogljivi robotski manipulator [1].

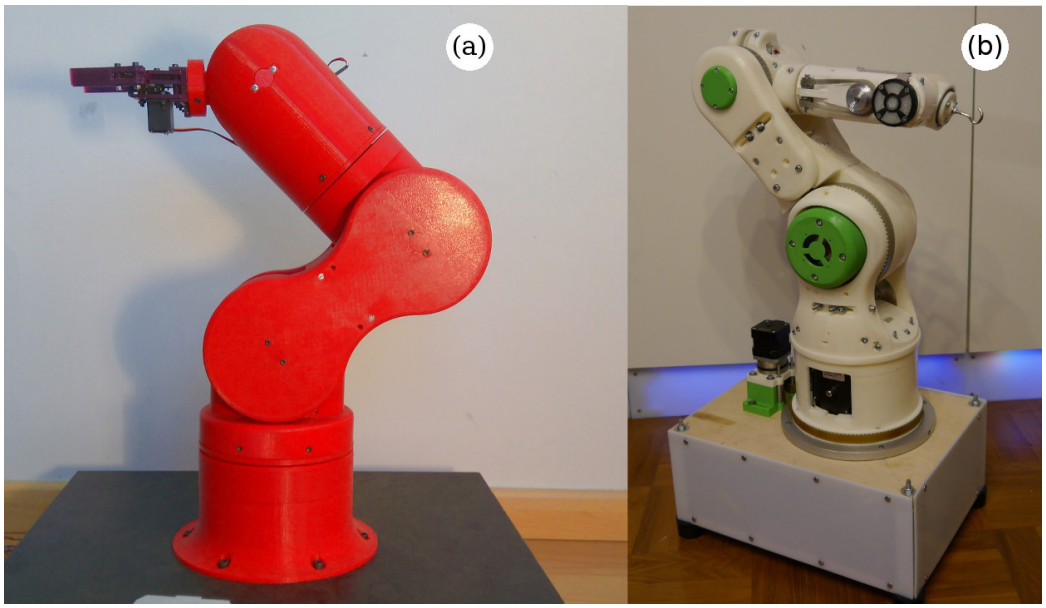
storjev (ang. H-bridge) za klasične elektromotorje ali namenskih gonilnikov za koračne motorje [3]. Ker so vse dodatne komponente na enem tiskanem vezju in jih je sorazmerno malo, tak pristop poenostavi in poceni končno izdelavo. Avtorji [2] so šli še korak dlje in za krmiljenje radijsko vodenih (ang. RC-radio control) servomotorjev uporabili kar že v servomotor vgrajen krmilnik. Pri tem so servomotor predelali le toliko, da so analogni izhod vgrajenega potenciometra povezali na mikrokrmilnik in odčitavali pozicijo. Avtorji članka [4] so uporabili podoben pristop, le da so za odčitavanje pozicije v manipulator vgradili dodatne potenciometre. Odčitavanje vrednosti v servomotor vgrajenih potenciometrov se nam zdi bolj smiselna in tudi bolj ekonomična rešitev. Prednost zunanje vgradnje potenciometra (ali drugih senzorjev za odčitavanje pozicije) se pojavi šele, ko imamo servomotor, ki lahko doseže večji kot zasuka gredi (kar se preprosto doseže z manjšo predelavo servomotorja, pri kateri se odstrani zaščito, ki omejuje zasuk gredi), kot mu to dovoljuje vgrajeni potenciometer. Tu predstavlja prednost tudi zunanja krmilna logika, ki je lahko optimizirana za specifičen namen in posledično zmogljivejša ter jo lahko po potrebi tudi nadgradimo.

V literaturi zasledimo, da se avtorji zaradi poenostavitev osredotočijo pretežno na izdelavo robotskega manipulatorja z nespremenjeno zasnovo. To bi predstavljalo veliko omejitev, če bi želeli spremeniti število prostostnih stopenj, saj je zasnova strojne in programske opreme vnaprej definirana le za točno določeno število prostostnih stopenj. Takšne omejitve so se lotili nasloviti avtorji članka [6, 7]. Posvetili so se izdelavi modula z eno prostostno stopnjo, ki vsebuje le RC-servomotor in krmilno vezje. Moduli so povezani na skupno vodilo, ki omogoča prenašanje informacij. Zaradi preprostosti in dizajna lahko s temi moduli sestavijo robota poljubne oblike. Za primer demonstracije tako sestavijo robota, ki ponazarja humanoida, kačo ali nestandardno obliko robota. Zaradi zasnove modula lahko z moduli sestavimo tudi robotski manipulator poljubne konfiguracije.

Ne smemo pa pozabiti tudi na ustvarjalno skupnost, ki izdeluje raznolike elektrotehniške in druge pripomočke za lastno uporabo ter svoje izkušnje in izdelke delijo z ostalimi. Primera izdelave lastnega robotskega manipulatorja sta projekta Thor [2] in 3D Printable Robot Arm [3] (slika 1.2). Projekta sta osredotočena na izdelavo večjega in zmogljivejšega robotskega manipulatorja, ki se lahko postavi ob bok manjšim industrijskim robotskim manipulatorjem. Ogrodje je izdelano s pomočjo 3D tiskalnika, kar zelo poenostavi izdelavo poljubnega ogrodja, ki je hkrati tudi dovolj močno in lahko. Za pogon sklepov uporabljajo koračne motorje z ustrezno krmilno logiko. Kljub uporabi dostopnih komponent so te drage in izdelava robotskega manipulatorja še vedno predstavlja finančni zalogaj. Zaradi njegove velikosti je treba uporabiti močnejše koračne motorje ali v paru dva manjša. Vsak koračni motor nato potrebuje ustrezno krmilno vezje, ki skrbi za pravilno premikanje. Vsa krmilna vezja mora nato krmiliti mikrokrmilnik, ki hkrati komunicira tudi z računalnikom. Celoten sistem je nato treba napajati z zmogljivim napajalnikom.

Poleg večjih in zmogljivejših robotskih manipulatorjev se pojavljajo tudi

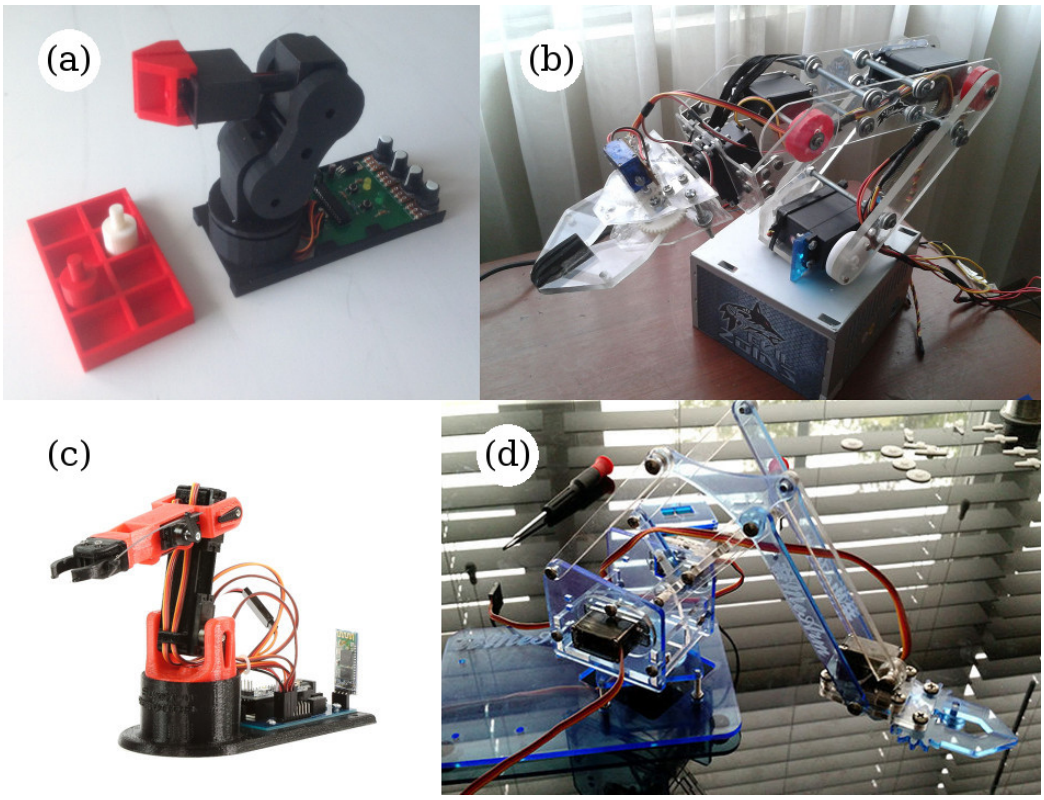
[2] https://hackaday.io/project/12989-thor
[3] https://hackaday.io/project/3800-3d-printable-robot-arm



Slika 1.2: Sliki projekta robotskega manipulatorja (a) Thor in (b) 3D Printable Robot Arm. Roki sta veliki in namenjeni zahtevnejši in naprednejši uporabi.

manjši in manj zmogljivi [4] [5] [6] [7] (slika 1.3). Med seboj so si zelo različni, preprosti za izdelavo in cenovno zelo dostopni. Za premikanje sklepov uporabljajo RC-servomotorje, ki jih krmili mikrokrmilnik, najpogosteje kar Arduino UNO. Razlikujejo se le v izbiri materiala za izdelavo ogrodja, številu prostostnih stopenj in velikosti (moči) RC-servomotorjev. Manipulatorji so izredno lahki in veliko manj nevarni za uporabo kot prej omenjeni. Pomanjkljivost predstavlja omejitev krmiljenja, saj nimajo povratne zanke, zaradi česar trenutna pozicija sklepov ni poznana. Ravno tako niso opremljeni z aktivno tokovno zaščito, česar nismo zasledili niti pri člankih [1, 2, 3, 4, 5].

[4] https://hackaday.io/project/26119-pedro-petit-robot-an-open-source-robotic-arm
[5] http://www.instructables.com/id/ROBOTIC-ARM-Arduino-Controlled/
[6] http://www.littlearmrobot.com/
[7] http://lifehacker.com/build-a-kickass-robot-arm-the-perfect-arduino-project-1700643747



Slika 1.3: Primer manjših robotskih manipulatorjev: (a) Pedro Pettit robot arm [4], (b) Arduino krmiljeni robotski manipulator [5], (c) LittleArm 2C [6] in (d) MeArm [7].

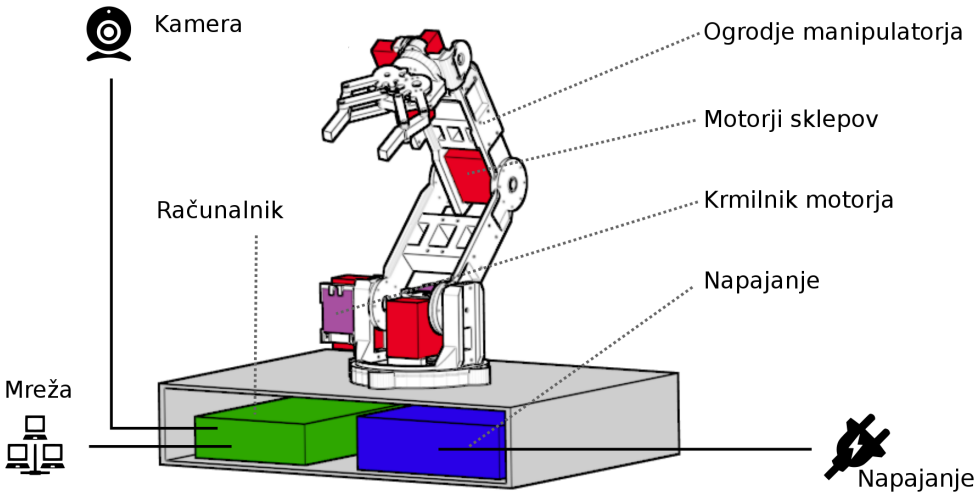
1.3 Oris sistema

Zadali smo si cilj, da se lotimo vseh zgoraj omenjenih izzivov in izdelamo lahek nizkocenovni manipulator z naslednjimi lastnostmi: preprost za izdelavo, cenovno dostopen, preprost in varen za uporabo, dovolj robusten in natančen, kompakten ter ravno dovolj velik za uporabo na mizi. Pri nosilnosti si nismo zastavili visokih ciljev in smo zadovoljni z nosilnostjo 50 g, saj bo manipulator premikal le majhne in lahke predmete. Pri številu prostostnih stopenj smo se omejili le na pet stopenj, kar je eno stopnjo manj, kot je potrebno za dosego poljubne lege vrha manipulatorja.

Poleg robotskega manipulatorja sistem sestavljata še majhen računalnik

z vso potrebno programsko opremo za njegovo krmiljenje, zajem slike s kamere in komunikacijo preko lokalne mreže ter dovolj zmogljiv napajalnik za napajanje vgrajenega računalnika in robotskega manipulatorja. Oris sistema nazorno prikazuje slika 1.4.

Mejo za ceno komponent celotnega sistema smo želeli obdržati čim nižje in smo jo določili na manj kot 300 €. To bomo poskušali doseči z uporabo cenovno dostopnih komponent in 3D tiskalnika za tiskanje ogrodja.



Slika 1.4: Oris zastavljenega sistema z vsemi sestavnimi deli.

1.4 Prispevki

Prispevek magistrskega dela je razviti sistem robotskega manipulatorja. Sestavni deli so dostopni in tudi cenovno ugodni, kljub temu pa sistem omogoča naprednejše krmiljenje servomotorjev in se po funkcionalnosti primerja z veliko dražjimi sistemi. Sistem temelji na že obstoječih, prosto dostopnih rešitvah, ki pa so bile pomanjkljive. V delu smo se posvetili njihovemu odpravljanju in nadgradnji.

Velik poudarek smo dali preprostosti izdelave in veliki prilagodljivosti, zaradi česar je robotski manipulator preprost za izdelavo in uporabo. Hkrati

pa sistem ni omejen na izdelavo vnaprej določene konfiguracije robotskega manipulatorja. Zaradi svoje prilagodljivosti omogoča izdelavo poljubne konfiguracije robotskega manipulatorja ali poljubnega robota. Rezultat našega dela predstavlja celovit sistem, ki vsebuje robotski manipulator z vgrajenim manjšim računalnikom, na katerem je nameščena programska oprema za delo in njegovo programiranje. Tako robotski manipulator kot programska oprema sta primerna za učne namene.

1.5 Struktura naloge

Poglavje 2 opisuje zgradbo sistema, kjer opišemo nekatere teoretične robotske koncepte, tipe robotskih manipulatorjev, kako jih matematično opišemo in računamo lego prijemala.

Poglavje 3 podrobno opisuje izhodišče pri reševanju problemov in našo implementacijo ter predstavlja glavni del našega dela. Podrobno opišemo pomanjkljivosti uporabljenih izhodiščnih rešitev in način njihove odprave, razvito programsko opremo in integracijo v različna sistema. Obsega tudi pregled vrst pogonov, ki jih lahko uporabimo za premikanje aktuatorjev, in teorijo krmiljenja z uporabo povratne zanke.

Poglavje 4 vsebuje opis evalvacijskega sistema in evalvacijo tako posameznih delov robotskega manipulatorja kot tudi celotnega sistema. Rezultat evalvacij predstavljajo vrednosti odstopanj in natančnosti robotskega manipulatorja.

V poglavju 5 povzamemo glavne lastnosti izdelanega sistema in cilje pri nadaljnjih nadgradnjah in izboljšavah.

V prilogah podamo sliko sestavnih delov ogrodja robotskega manipulatorja, shemi razvitih vezij, ter kosovnico uporabljenih materialov in komponent.

Poglavje 2

Zasnova sistema

2.1 Osnovni pojmi

Pri načrtovanju robotskega manipulatorja moramo upoštevati zakonitosti in uveljavljena pravila. Seznanili se bomo z osnovnimi pojmi in pravili, s pomočjo katerih smo zasnovali model robotskega manipulatorja.

2.1.1 Prostostne stopnje

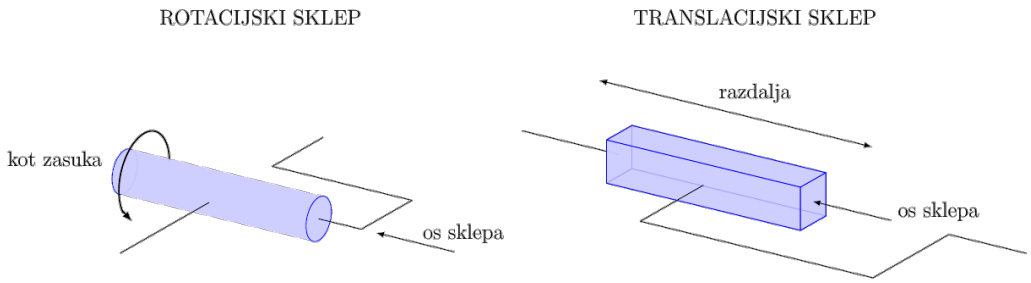
Osnovni koncept robotskega manipulatorja predstavlja prostostna stopnja (ang. DOF – degree of freedom) [8]. Vzemimo na primer zanemarljivo majhen masni delec. Število prostostnih stopenj tako definira število neodvisnih koordinat, ki so potrebne za opis pozicije masnega delca. Če lahko tak delec premikamo samo po premici, opisuje sistem z eno stopnjo prostosti, saj pozicijo podamo z razdaljo. Nihalo s togo pritrditvijo, ki se giblje v ravnini, je tudi sistem z eno prostostno stopnjo, pri čemer pozicijo podamo s kotom zasuka. Sistem zdaj razširimo tako, da se masni delec lahko premika v ravnini. Ker za opis poljubnega pozicije masnega delca v ravnini potrebujemo dva podatka, smo dobili sistem z dvema prostostnima stopnjama. Za podajanje pozicije masnega delca v prostoru tako potrebujemo tri prostostne stopnje. Za opis lahko uporabimo pravokotne koordinate x, y, z.

Za popoln opis telesa v prostoru poleg *pozicije* potrebujemo še njegovo

orientacijo. Telo lahko zasukamo okoli vseh osi in tako dobimo poleg treh translacijskih prostostnih stopenj še tri rotacijske. Položaj in orientacijo z eno besedo poimenujemo *lega* telesa v prostoru kar predstavlja 6 DOF.

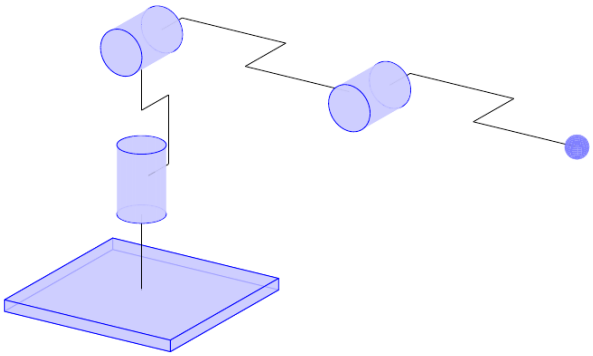
2.1.2 Vrste robotskih rok

Robotski manipulator predstavlja serijska veriga segmentov [8]. Dva sosednja segmenta med seboj povezuje sklep, ki zmanjša število prostostnih stopenj med segmentoma. Robotski sklepi imajo samo eno prostostno stopnjo in so lahko translacijski ali rotacijski, kar prikazuje slika 2.1. V robotiki so najpogosteje uporabljeni rotacijski sklepi. Imajo obliko tečaja in omejujejo gibanje dveh sosednjih segmentov na zasuk okoli skupne osi sklepa. Translacijski sklep omejuje gibanje dveh segmentov na translacijo in se ga pogosto uporablja pri 3D tiskalnikih.



Slika 2.1: Rotacijski in translacijski robotski sklep.

Spoznali smo, da obstajata le dva tipa robotskih sklepov, in sicer rotacijski in translacijski. Omenjeno dejstvo in lastnost robotskih rok, da so osi sosednjih sklepov bodisi vzporedne ali pravokotne, omeji število praktičnih izvedb različnih rok. Te so antropomorfna, sferična, SCARA, cilindrična in kartezična robotska roka. Antropomorfna roka ima vse sklepe rotacijske, shematični oris roke prikazuje slika 2.2, in je izmed vseh še najbolj podobna človeški roki. Zaradi njene zasnove smo jo izbrali tudi za osnovo našega robotskega manipulatorja.



Slika 2.2: Antropomorfna robotska roka.

2.2 Geometrijski model robota

Za ustrezno premikanje prijemala v prostoru moramo poznati geometrijske lastnosti robotskega manipulatorja. Z njegovim poznavanjem lahko določimo lego (pozicija in orientacijo) prijemala v odvisnosti od spremenljivk sklepov. To se uporablja tako za *direktno kinematiko*, kjer poznamo spremenljivike sklepov, kot tudi za *inverzno kinematiko*, torej za iskanje vrednosti spremenljivk sklepov glede na znano lego prijemala.

2.2.1 Denavit-Hartenbergovi parametri

Denavit-Hartenberg (krajše DH) je v robotiki pogosto uporabljen pristop za opis geometrije robotskega manipulatorja [9]. Sestavljen je iz pravil, ki opišejo, kako sta dva sklepa med seboj odvisna. Z rekurzivno uporabo pravila tako določimo parametre poljubnega robotskega manipulatorja.

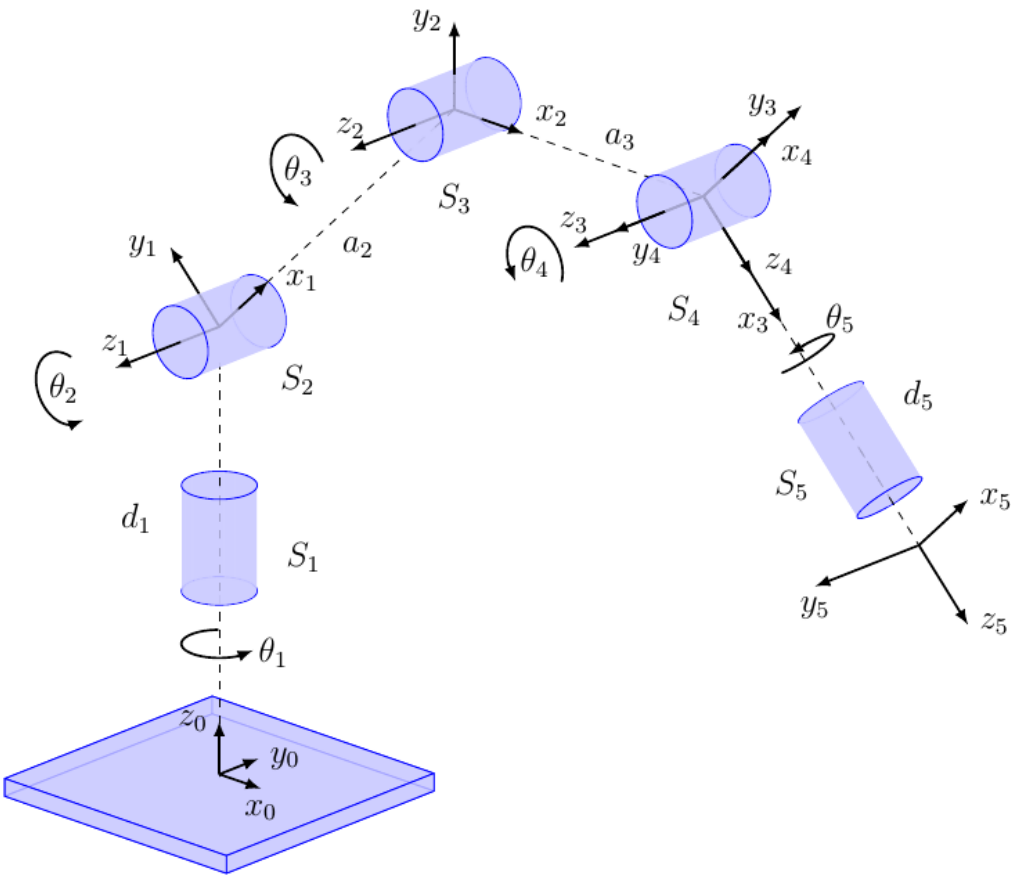
V sklopu tega dela smo razvili antropomorfni robotski manipulator s petimi prostostnimi stopnjami (5 DOF). Slika 2.3 prikazuje model manipulatorja na podlagi pravil, tabela 2.1 pa ustrezne parametre manipulatorja, potrebne za izračun lege, če s S_i povežemo x_{i-1} in z_{i-1} z x_i in z_i. Parametri, uporabljeni pri DH, so:

1. a_i – razdalja med koordinatnima izhodiščema vzdolž osi x_i;

2. d_i – razdalje med koordinatnima izhodiščema vzdolž osi z_{i-1};

3. α_i – kot med osema z_{i-1} in z_i okrog osi x_i. Kot je pozitiven pri zasuku v nasprotni smeri urinega kazalca;

4. θ_i – kot med osema x_{i-1} in x_i okrog osi z_{i-1}. Kot je pozitiven pri zasuku v nasprotni smeri urinega kazalca. V primeru rotacijskega sklepa je vrednost spremenljiva.



Slika 2.3: Zasnovan antropomorfni robotski manipulator s petimi prostostnimi stopnjami (5 DOF).

Transformacijo med koordinatnima sistemoma i in $i-1$ opisuje formula (2.1). Z upoštevanjem formule in podatkov iz tabele 2.1 dobimo matrike

Segment	a_i	d_i	α_i	θ_i
1	0 mm	48 mm	90°	θ_1
2	108 mm	0 mm	0°	θ_2
3	112 mm	0 mm	0°	θ_3
4	0 mm	0 mm	90°	θ_4
5	0 mm	90 mm	0°	θ_5

Tabela 2.1: DH-parametri manipulatorja.

preslikav med sklepi (2.2), (2.3), (2.4), (2.5) in (2.6). Direktno preslikavo iz izhodiščnega koordinatnega sistema v koordinatni sistem vrha robota dobimo z množenjem vseh matrik po enačbi (2.7).

$$A_i^{i-1} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

$$A_1^0 = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 & 0 \\ \sin\theta_1 & 0 & -\cos\theta_i & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

$$A_2^1 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$A_3^2 = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & a_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$A_4^3 = \begin{bmatrix} \cos\theta_4 & 0 & \sin\theta_4 & 0 \\ \sin\theta_4 & 0 & -\cos\theta_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

$$A_5^4 = \begin{bmatrix} \cos\theta_5 & -\sin\theta_5 & 0 & 0 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

$$A_5^0 = A_1^0 A_2^1 A_3^2 A_4^3 A_5^4 \tag{2.7}$$

Z uporabo transformacije A_5^0 torej lahko izračunamo lego vrha robotskega manipulatorja izraženo v izhodiščnem koordinatnem sistemu in matriko uporabimo pri direktni kinematiki.

Poglavje 3

Razvoj sistema

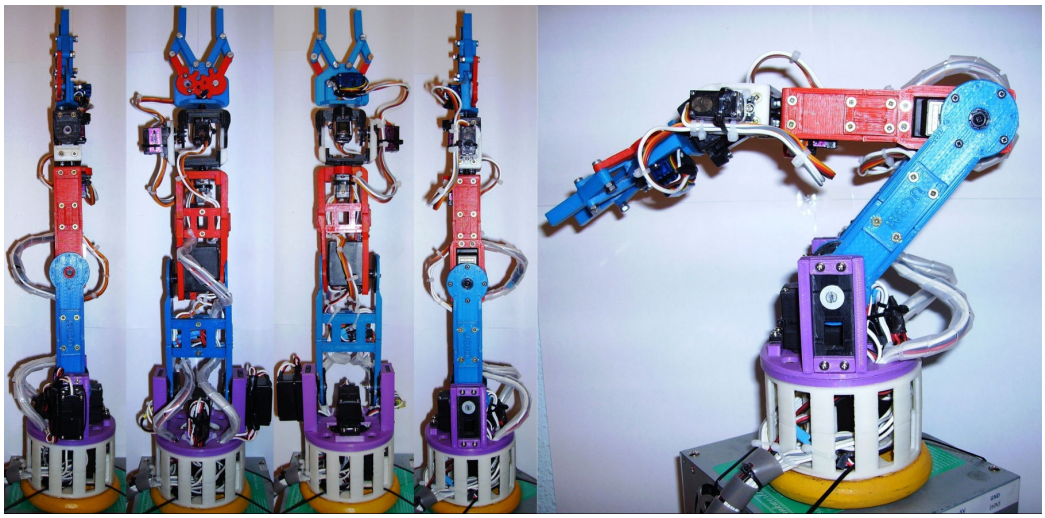
V tem poglavju se bomo posvetili implementaciji vseh posameznih delov sistema. Govorili bomo o prosto dostopnih projektnih (ang. open source and open hardware), na katere smo se oprli pri doseganju zastavljenih ciljev, in podrobno opisali vse opravljene spremembe.

3.1 Ogrodje manipulatorja

V poglavju 2.1.1 smo pisali o robotskih rokah, katere vrste poznamo in kako računamo končno lego. Ko imamo znane specifikacije modela manipulatorja, se lahko posvetimo izdelavi fizičnega modela. Tukaj imamo na voljo več pristopov. Pristop, ki se porodi najprej, je, poiskati in kupiti že obstoječi manipulator. Ta manipulator je lahko v celoti sestavljen in polno opremljen ali pa je na voljo le ogrodje in ga moramo sestaviti ter opremiti z ustreznimi senzorji, motorji in krmilnim vezjem. Cena takih manipulatorjev znaša od 50 € dalje, pri čemer za tako nizko ceno na trgu najdemo le manipulatorje vprašljive kakovosti. Naslednja možnost je, da manipulator v celoti načrtujemo in izdelamo sami. Ker imamo na razpolago 3D tiskalnik, je lastna izdelava izvedljiva in tudi ne predstavlja večjega finančnega zalogaja.

3.1.1 Izhodišče

V poglavju 1.2 smo omenili tudi delo skupnosti in različne pristope k izdelavi ogrodja robotskega manipulatorja. Pri delu smo se tako oprli na že obstoječi 3D model roke [10] [1], prikazan na sliki 3.1. Model manipulatorja ima šest prostostnih stopenj in prijemalo, poganjajo ga RC-servomotorji in je tudi dovolj majhen za uporabo na mizi. Avtor je model roke sestavil tako, da lahko vse sestavne dele natisnemo s 3D tiskalnikom. Model manipulatorja je tudi zelo preprost za izdelavo in vsebuje sorazmerno malo sestavnih delov.



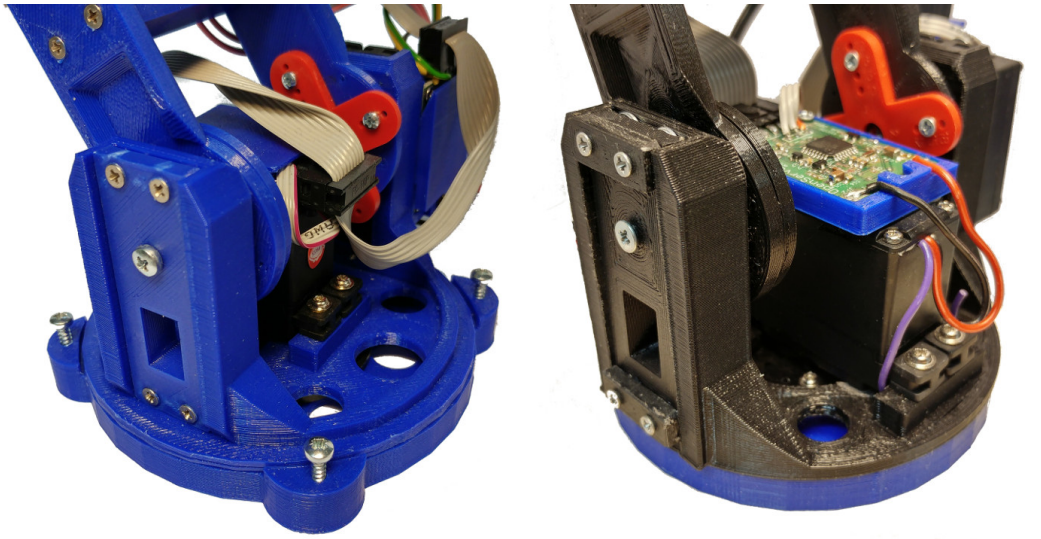
Slika 3.1: Slika prikazuje izvirni model robotske roke.[1]

3.1.2 Nadgradnja

Uporabljen model roke smo prilagodili svojim potrebam. Občutne spremembe smo opravili na bazi in podstavku robotske roke. Z okrepitvijo baze smo zmanjšali njeno deformacijo in jo za boljšo stabilnost tudi razširili. Originala podstavka ne uporabljamo, saj vsebuje dva servomotorja in je tudi izredno visok in nestabilen. Podstavek smo zato zamenjali z novim, nižjim

[1] Izvorni model je dosegljiv na Thingiverse (`http://www.thingiverse.com/thing:30163`) pod licenco Creative Commons license.

in širšim, z večjo naležno površino z bazo za boljšo stabilnost. Vgradili smo tudi aksialni ležaj in tako dobili gladek zasuk obeh delov. Za lažjo vgradnjo smo na zunanji strani dodali štiri pritrdilna mesta, ki omogočajo privitje podstavka na poljubno podlago. Opravljene so bile tudi manjše spremembe za lažjo vgradnjo uporabljenih servomotorjev.

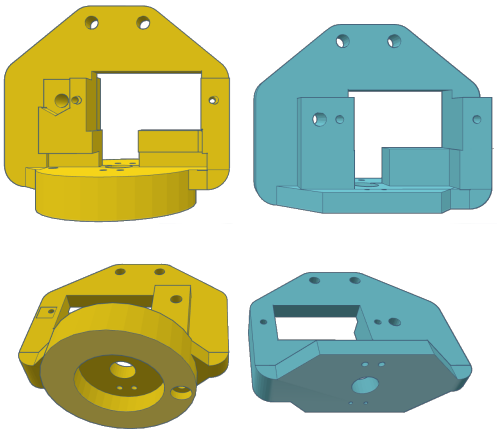


Slika 3.2: Leva stran prikazuje končno različico baze in podstavka, desna pa vmesno različico, ki je povzeta po originalnem podstavku.

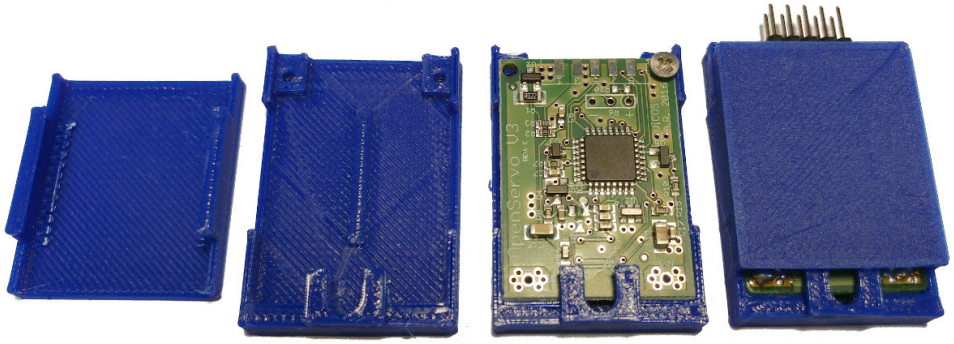
Odstranili smo četrti sklep in segmenta združili v enega. Vzrok za to je bila slaba kakovost servomotorja ES08AII, ki ima veliko zračnost, zato so bili vsi nadaljnji segmenti zelo nestabilni. Težava se je pojavila tudi pri šesti prostostni stopnji, ki skrbi za premik prijemala. Tu smo spremenili del zapestja in ga ustrezno povečali, da zagotavlja najboljšo stabilnost ne glede na zasuk zapestja. Sprememba je prikazana na sliki 3.3. Zapestje smo nato še dodatno prilagodili za vgradnjo krmilnega vezja.

Izdelali smo tudi manjše ohišje za vgradnjo krmilne logike. Sestavljeno iz dveh delov. V ogrodje se vgradi krmilno logiko, ogrodje pa se nato pritrdili na želeno podlago. Pokrov dodatno zaščiti krmilno vezjo. Primer vgradnje prikazuje slika 3.4.

Izdelali smo tudi namenski podstavek (na sliki 3.5 pod manipulatorjem),



Slika 3.3: Na levi je prikazano končno oblikovano zapestje, na desni pa originalno zapestje.

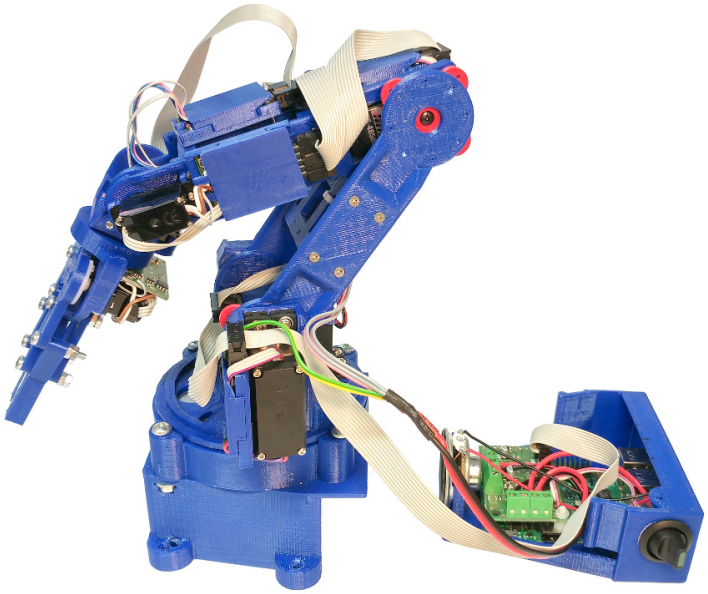


Slika 3.4: Ohišje za zaščito krmilnega vezja.

na katerega smo pritrdili robotski manipulator. Sestavljen je iz dveh delov, fiksnega in izvlečnega. Fiksni del se lahko pritrdi na podlago in je nosilni del. V izvlečni del smo vgradili majhen računalnik (Raspberry Pi 3), vtičnico za napajanje, stikalo za vklop in izklop napajanja elektromotorjev, zvočnik in

ostale dodatne komponente.

Tako smo dobili celovito, izboljšano in preprosto ogrodje robotskega manipulatorja z vsemi sestavnimi deli, ki jih lahko natisnemo s 3D tiskalnikom. Slika vseh sestavnih delov se nahaja v prilogi A.



Slika 3.5: Sestavljen robotski manipulator z namenskim podstavkom, katerega kosi so natisnjeni s 3D tiskalnikom.

3.2 Pogoni

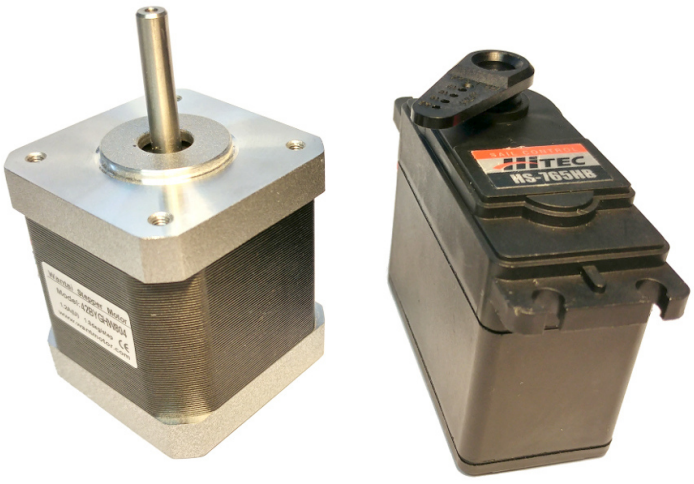
Omenili smo, da imajo robotski manipulatorji dano število prostostnih stopenj in da s premikanjem posameznih segmentov dosežemo lego v prostoru. Za premikanje se uporablja različne tehnologije, ki imajo svoje prednosti in slabosti.

3.2.1 Vrste pogonov

Vrste pogonov lahko grobo delimo na pogone s hidravliko, pnevmatiko, koračne motorje ali elektromotorje. *Hidravlika* se uporablja na primer tam, kjer so

potrebne (izredno) velike sile. Primer takega robotskega manipulatorja je teleskopska roka na delovnih strojih. Sistemi uporabljajo (običajno) eno ali več črpalk za olje, ki jih poganja motor z notranjim izgorevanjem ali elektromotor. Slabosti hidravlike sta drago vzdrževanje in nevarnost visokotlačnega brizga vročega olja. *Pnevmatski pogon* deluje na stisnjen zrak in je veliko manj nevaren, a tudi manj zmogljiv kot hidravlični pogon, vendar še vedno sorazmerno velik.

Zaradi svoje sorazmerno preproste uporabe in krmiljenja se pri krmiljenju sistemov pogosto uporablja *koračne elektromotorje* (slika 3.6). Za razliko od klasičnih se koračni lahko vrtijo (premikajo) le po korakih, zaradi česar lahko sorazmerno preprosto ocenimo relativeno pozicijo tudi brez uporabe povratne informacije. Pogosto se uporabljajo tudi *servomotorji* v eni izmed mnogo različic. Za servomotor se šteje celovit sistem z vgrajenim elektromotorjem, zobniškim prenosom in povratno informacijo. Uporabljajo se elektromotorji na izmenično (ang. AC - alternating current) ali enosmerno napetost (ang. DC - direct current).



Slika 3.6: Koračni motor Wantai 42BYGHW804 (levo) in RC-servomotor Hitec HS-765HB (desno) drug ob drugem.

Odločili smo se za uporabo *RC-servomotorjev*, ki se, kot že ime pove, upo-

rabljajo v radijsko vodenih (ang. RC – Radio Control) modelih. Priljubljeni so zaradi izredno preproste uporabe in nizke cene. Na voljo so z zelo raznolikimi specifikacijami, različnih dimenzij in jih je preprosti vgraditi. Primere RC-servomotorjev različnih moči in velikosti prikazuje slika 3.7.

Slika 3.7: Primeri RC-servomotorjev različnih velikosti in specifikacij. Z leve proti desni: Emax ES08AII in Hitec HS-311, HS-485HB, HS-645MG (enake velikosti) ter večja HS-765HB in HS-805MG.

K široki uporabnosti pripomoreta tudi njihova široka izbira in namembnost. Najpogosteje zasledimo rotacijske, obstajajo pa tudi linearni, kar sovpada z rotacijskim in translacijskim sklepom. Tako lahko z uporabo RC-servomotorjev zgradimo poljuben robotski manipulator. Poleg velikosti in teže se razlikujejo tudi v moči, kotni hitrosti in kakovosti izdelave.

Ponudba RC-servomotorjev je tako zelo raznolika, njihova izbira pa odvisna od namena uporabe. Pri izbiri servomotorjev za krmiljenje manipulatorja je tako treba upoštevati, da morajo premagovati različne obremenitve. Največja obremenitev se pojavi pri polno iztegnjenem robotskem manipulatorju. Servomotor mora biti zato dovolj zmogljiv, da lahko premaga obremenitev tudi v tej poziciji. Moč, višja od moči, potrebne za premik zgolj robotskega manipulatorja, se uporablja za premikanje dodatnih bremen (ang.

payload). Za svoj sistem smo izbrali servomotorje HS-645MG [2], HS-485HB [3], HS-311 [4], ES08AII [5], katerih specifikacije prikazuje tabela 3.1.

Servo-motor	Napetost	Rotacijska hitrost	Navor	Kratkostični tok	Teža	Material zobnikov
HS-645MG	$4,8\ V$	$0,24\ s/60°$	$7,7\ kg/cm$	$2,5\ A$	$55,2\ g$	kovina
	$6,0\ V$	$0,20\ s/60°$	$9,6\ kg/cm$			
HS-485HB	$4,8\ V$	$0,22\ s/60°$	$4,8\ kg/cm$	$1,2\ A$	$45,0\ g$	karbonit
	$6,0\ V$	$0,20\ s/60°$	$6,0\ kg/cm$			
HS-311	$4,8\ V$	$0,19\ s/60°$	$3,0\ kg/cm$	$0,8\ A$	$43,0\ g$	najlon
	$6,0\ V$	$0,15\ s/60°$	$3,5\ kg/cm$			
ES08AII	$4,8\ V$	$0,12\ s/60°$	$1,5\ kg/cm$	$0,7\ A$	$8,5\ g$	najlon
	$6,0\ V$	$0,10\ s/60°$	$1,8\ kg/cm$			

Tabela 3.1: Specifikacije uporabljenih RC-servomotorjev z navedenimi lastnosti pri obeh nazivnih napetostih delovanja.

Izbrani servomotorji se razlikujejo tako po velikosti kot tudi zmogljivosti. Glede na lastnosti smo uporabili servomotorje pri posameznih sklepih segmentov, kot navaja tabela 3.2 in je prikazano na sliki 3.8. Sklep 1 je sorazmerno malo obremenjen in ima zato vgrajen manj zmogljiv servomotor HS-311. Sklep 2 je izmed vseh najbolj obremenjen in ima zato vgrajen veliko močnejši servomotor HS-645MG. Sklep 3 je obremenjen malo manj kot sklep pred njim in ima vgrajen ustrezno zmogljiv servomotor HS-485HB. Sklepa 4 in 5 nista zelo obremenjena, a imata omejitev velikosti in imata zato

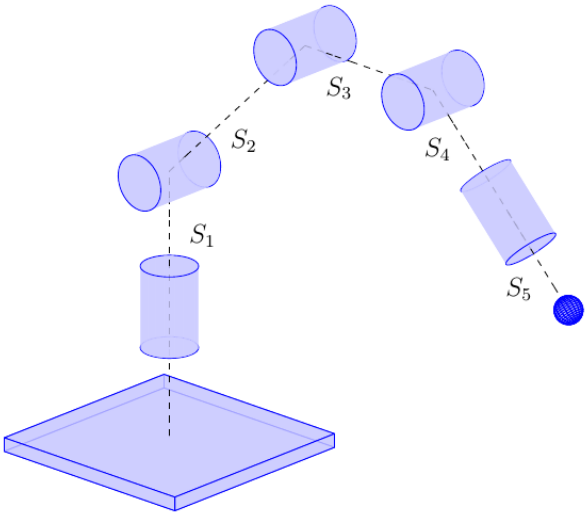
[2] http://hitecrcd.com/products/servos/sport-servos/analog-sport-servos/hs-645mg-high-torque-metal-gear-servo/product

[3] http://hitecrcd.com/products/servos/sport-servos/analog-sport-servos/hs-485hb-deluxe-hd-ball-bearing-servo/product

[4] http://hitecrcd.com/products/servos/sport-servos/analog-sport-servos/hs-311-standard-economy-servo/product

[5] https://www.emaxmodel.com/es08a-ii.html

vgrajen zelo majhen in lahek servomotor ES08AII, ki zaradi nizke teže tudi manj obremenjuje predhodne sklepe. Zaradi majhne velikosti je servomotor ES08AII vgrajen tudi v prijemalu manipulatorja.



Slika 3.8: Sklepi in segmenti robotskega manipulatorja.

Sklep	Servomotor
S1	HS-311
S2	HS-645MG
S3	HS-485HB
S4	ES08AII
S5	ES08AII
Prijemalo	ES08AII

Tabela 3.2: Servomotorji na posameznih sklepih.

Poleg RC-servomotorjev poznamo tudi robotske servomotorje, ki so zmogljivejši in vsebujejo dvosmerno komunikacijo ter omogočajo branje pozicije in ostalih parametrov. Pomanjkljivost predstavljata le majhna izbira robotskih servomotorjev in sorazmerno visoka cena, saj so precej novi in po njih

ni tolikšnega povpraševanja kot po RC-servomotorjih. Kljub temu pa počasi postajajo vse bolj popularni.

3.2.2 Predelava RC-servomotorja

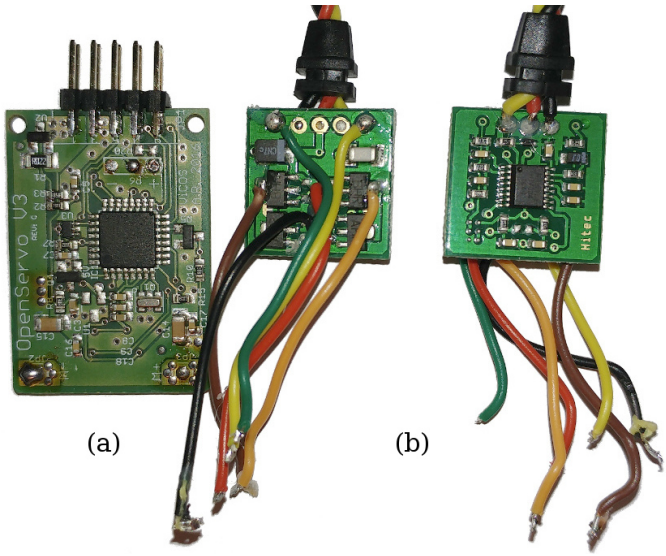
RC-servomotor je sestavljen iz elektromotorja, zobniških prenosov, povratne zanke in krmilne logike. Njegovo delovanje je preprosto. Krmilna logika na podlagi krmilnega signala in povratne zanke skrbi za ustrezno krmiljenje elektromotorja. Zobniški prenosi nato vrtenje gredi elektromotorja prenesejo na gred servomotorja in pri tem zmanjšajo število vrtenj ter povečajo navor. Povratno zanko predstavlja rotacijski potenciometer, ki je neposredno povezan z gredjo servomotorja.

Čeprav je preprosta uporaba RC-servomotorjev tudi razlog za njihovo popularnost, nam predstavlja omejitev, saj lahko nastavljamo le kot zasuka gredi, kar pa ni dovolj. Za dosego ciljev, ki smo si jih zastavili v poglavju 1.3, mora krmilna logika poleg nastavljanja pozicije omogočati tudi branje trenutne pozicije, nastavljanje vrednosti regulatorja in še veliko več. RC-servomotor smo zato opremili z lastno (zmogljivo) krmilno logiko, ki je podrobneje opisana v poglavju 3.3.

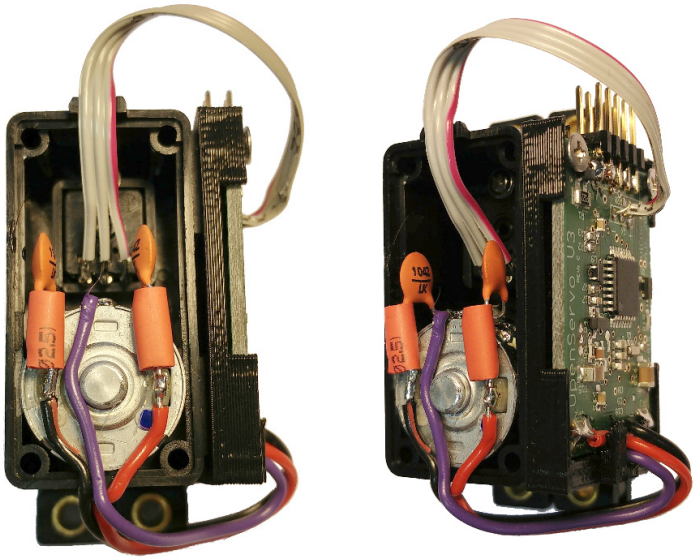
Vgradnja krmilne logike

Pred vgradnjo krmilne logike je treba odstraniti originalno krmilno logiko servomotorja, saj nam ne koristi. Povezana je s potenciometrom in elektromotorjem. Primer odstranjenega originalnega krmilnega vezja prikazuje slika 3.9.

Novo vezje je priporočljivo vgraditi ob servomotor ali ga uporabiti samostojno, saj je v večini primerov novo krmilno vezje preveliko za vgradnjo v ohišje. Slika 3.10 prikazuje primer predelave servomotorja HS-485HB s krmilno logiko, pritrjeno na zunanjo steno ohišja servomotorja. Pri vgradnji se lahko potenciometer in elektromotor priključita v poljubni orientaciji, saj OpenServo omogoča spremembo orientacije tudi programsko. Priporočljivo je vgraditi tudi dodatne keramične kondenzatorje, kar je razvidno tudi s



Slika 3.9: Primerjava velikosti (a) izdelanega krmilnega vezja in (b) iz HS-645 odstranjenega krmilnega vezja s spodnje in zgornje strani.



Slika 3.10: Primer vgradnje krmilne ploščice in priključitve potenciometra in elektromotorja. Dodana sta dva keramična kondenzatorja za glajenje elektromagnetnega valovanja.

slike 3.10, saj so namenjeni zmanjševanju elektromagnetnega šuma, ki ga povzročajo krtačni elektromotorji.

3.3 Krmilna logika

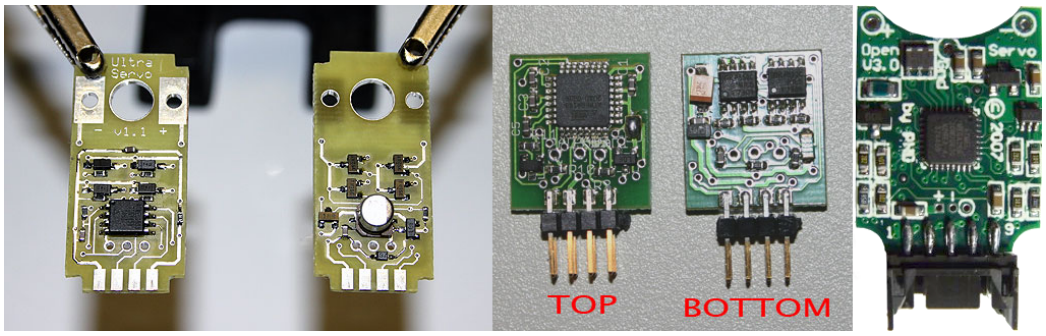
Tu bomo podrobno opisali, kako smo pristopili k realizaciji zastavljenega cilja. Pri doseganju cilja smo se namreč oprli na že obstoječe rešitve. Slednje pohitri razvoj končnega izdelka. Hkrati pa z deljenjem naših rešitev, tudi pripomoremo skupnosti in olajšamo delo drugim, ki se soočajo s podobnimi težavami.

3.3.1 Izhodišče

OpenServo [11] je projekt z odprto programsko in strojno opremo, ki ga razvijajo posamezniki s celega sveta z začetkom razvoja leta 2006. Ideja in cilj projekta sta omogočiti dostop do cenovno dostopne strojne in programske opreme ter komponent, s katerimi se nadomesti originalno analogno vezje v radijsko vodenih servomotorjih in tako nadgradi servomotor z zmogljivejšo krmilno logiko.

Jedro projekta OpenServo predstavlja mikrokrmilnik družine AVR podjetja Atmel, natančneje različica OpenServo 1.1 uporablja ATtiny45 ali ATtiny85 [12]. Oba mikrokrmilnika sta majhna in zato primerna tam kjer je prostor za vgradnjo omejen. Njuna velikost predstavlja tudi omejitev, saj imata na razpolago omejeno število vhodov in izhodov ter malo dolgoročnega spomina, kar ima za posledico manj zmogljivo programsko opremo. Pomanjkljivosti so nato delno odpravljene v različici OpenServo 2.1, ki uporablja ATmega8 [13] ali ATmega16 [14], z več vhodi in izhodi ter več pomnilnika za večjo funkcionalnost programske opreme, a ima zaradi vseh razširitev večje tiskano vezje. Zadnja uradna različica OpenServo 3.0 temelji izključno na ATmega16, ki poleg merjenja napetosti in toka omogoča še merjenje temperature in EMF (ang. Electromotive force). Vsa komunikacija z mikrokrmilnikom poteka preko vodila I^2C [15]. Primerjavo različic prikazuje slika

3.11.



Slika 3.11: Tiskana vezja projekta OpenServo, od leve proti desni: različica 1.1, 2.1 in 3.0 [11].

Veliki prednosti projekta OpenServo sta njegova modularnost in vsestranska uporabnost. Uporaba vodila za komunikacijo omogoča krmiljenje različnega števila servomotorjev. To število je omejeno le z lastnostmi uporabljenega vodila. Za nameček ima lahko vsak servomotor drugačno konfiguracijo, tako da nismo omejeni na točno določen scenarij, kot pri rešitvah [2, 4, 1, 5]. Komunikacija preko vodila I^2C je preprosta in omogoča branje in nastavljanje parametrov servomotorja. Zaradi modularnosti v primeru okvare preprosto zamenjamo kontrolni del z novim, kar poenostavi in poceni vzdrževanje sistema. Slabost predstavlja le večji začetni vložek, saj je treba izdelati toliko kontrolnih vezij, kolikor servomotorjev želimo krmiliti. Kljub temu pa so njegove prednosti veliko večje in, po našem mnenju, odtehtajo pomanjkljivosti. Dodatno, ker vsa komunikacija poteka preko vodila, lahko zamenjamo obstoječi mikrokrmilnik z zmogljivejšim in pri tem uporabimo že obstoječi sistem. Poleg RC-servomotorjev lahko uporabimo tudi druge, večje servomotorje ali elektromotor, ki mu dodamo povratno zanko, pri čemer ni nujno, da jo predstavlja potenciometer.

3.3.2 Nadgradnja projekta OpenServo

Čeprav na spletni strani projekta OpenServo oglašujejo, da je kontrolno logiko mogoče kupiti pri različnih distributerjih, se je izkazalo, da temu ni več tako. Projekt je namreč v zatonu in ni aktiven. Posledica je, da so vse zaloge razprodane in novih serij ni predvidenih. Odločili smo se za alternativno rešitev in sami pristopili k izdelavi tiskanega vezja krmilne logike, opirajoč se na načrt projekta OpenServo. Tako smo se lahko podrobneje spoznali z zgradbo in delovanjem sistema ter odpravili pomanjkljivosti.

Vsa uradna dokumentacija projekta je dosegljiva na njegovi spletni strani [11]. [6] Odločili smo se za različico OpenServo 3.0 z zadnjo stabilno različico programske opreme. Izbrana različica ima največ razširitev in je tudi edina, ki jo podpira zadnja različica programa za mikrokrmilnik (po navedbi na spletni strani). Izkaže se, da so informacije na spletni strani pomanjkljive in tudi zavajajoče. Tako stabilna različica programske opreme ponuja le omejen nabor funkcionalnosti.

3.3.3 Tiskano vezje

Prednost lastne izdelave je, da lahko odpravimo odkrite napake ter po potrebi spremenimo in posodobimo tiskano vezje. Tako smo se po testiranju servomotorjev pri izdelavi tiskanega vezja odločili za manjše spremembe. Mikrokrmilnik ATmega16 smo zamenjali z ATmega328P [16], ki mu je skoraj enak. Namesto 16 kB ima 32 kB spomina, namesto 1 kB ima 2 kB statičnega rama in je tudi lažje dostopen in cenejši.

Nadaljnje izboljšave so prisotne pri segmentih, ki so tokovno najbolj obremenjeni. Močnejši servomotorji pri obremenitvah za delovanje potrebujejo sorazmerno visoke tokove, kar je razvidno iz tabele 3.1. Z analizo tokovne poti na shemi smo našli dve mesti za potencialno nadgradnjo. Tranzistor za krmiljenje motorja smo nadgradili z močnejšim, z boljšimi karakteristikami. Nadgradili smo tudi segment za merjenje toka, ki zmore zdaj meriti

[6] https://openservo.org/StepByStep

tokove do 3 A in z manj izgubami. Toleranco vhodne napetosti za napajanje elektromotorjev smo iz 7,5 V dvignili na 10 V.

Med testiranjem smo se odločili tudi za spremembo pri razporeditvi konektorjev in dodali možnost, da se kontrolni del vezja lahko napaja z zunanjim virom napetosti 5 V. Za to smo se odločili, ker smo pri napajanju za motorje kasneje dodali varnostno stikalo, s katerim fizično prekinemo napajanje in tako v primeru nevarnosti izklopimo napajanje motorjev.

Tabeli 3.3 in 3.4 prikazujeta združene osnovne lastnosti tiskanega vezja. Stolpca *Priporočljivo* in *Tipično* navajata priporočljive vrednosti oziroma, vrednosti, za katere je bilo vezje izdelano. Stolpec *Največ* podaja absolutno najvišje vrednosti, pri katerih še deluje, vse vrednosti višje od navedenih pa lahko dolgoročno poškodujejo ali uničijo komponente na vezju. Pravilnost delovanja pri njih ni zagotovljena.

Lastnost	Priporočljivo	Največ
Delovna napetost	5,0 V	5,5 V
Tokovna poraba	16,9 mA	17,0 mA
Najvišja vhodna napetost motorjev	5,0 V	2x delovna napetost
Tokovna obremenitev elektromotorja		3000 mA

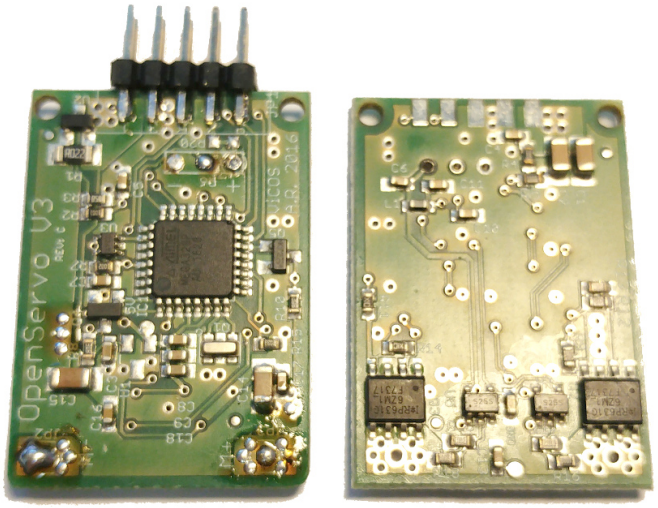
Tabela 3.3: Električne karakteristike tiskanega vezja.

Lastnost	Tipično
Teža	4,3 g
Širina	35,0 mm
Širina s konektorjem	43,0 mm
Višina	24,8 mm
Debelina	5,0 mm

Tabela 3.4: Fizične karakteristike tiskanega vezja.

Zaradi cenejšega razvoja in izdelave tiskanega vezja smo načrtovali dvostransko vezje, čeprav so v projektu uporabili štiriplastno vezje. Posledica

je njegova večja dimenzija, saj je težje razporediti in med seboj ustrezno povezati vse komponente.



Slika 3.12: Tiskano vezje, pripravljeno za vgradnjo. Na levi je prikazana zgornja, na desni pa spodnja stran tiskanega vezja.

Slika 3.12 prikazuje izdelano vezje z vsemi komponentami, ki je pripravljeno za nadaljnjo uporabo. Uporabljene komponente so dovolj velike in jih lahko s tanko spajkalno konico prispajkamo ročno. Priporočljivo je uporabiti tanko spajkalno konico ali spajkalno postajo za spajkanje komponent SMD (ang. surface mounted device). Tiskano vezje lahko s komponentami opremijo tudi že v podjetju, ki izdeluje tiskana vezja. To je uporabno predvsem pri njihovem večjem naročilu.

3.3.4 Koda mikrokrmilnika

Srce projekta OpenServo predstavlja mikrokrmilnik družine AVR. Vsebuje centralno procesno enoto (krajše CPE, ang. CPU – Central Processing Unit), ki temelji na 8-bitni arhitekturi RISC (Reduced Instruction Set Computer), le z aritmetično logično enoto (ang. ALU – Arithmetic Logic Unit) in izvedbo

operacije v eni urini periodi [17] (v večini primerov). Povedano drugače, v eni urini periodi lahko sešteje ali odšteje vrednosti dveh 8-bitnih registrov in vrednost shrani v podani register. Množenje dveh 8-bitnih registrov zahteva dodatno urino periodo, saj je dobljen rezultat 16-bitna vrednost. Prikrajšani pa smo tako za operacijo deljenja kot tudi za operacije s številom s fiksno ali plavajočo vejico. Omejitve lahko zaobidemo s programsko implementacijo manjkajočih operacij. Pri tem moramo imeti v mislih, da so programsko implementirane operacije počasnejše in porabijo več spomina, kar ni moteče, če hitrost procesiranja ni kritična.

Koda mikrokrmilnika je pisana v programskem jeziku C in uporablja odprtokodne knjižnice za mikrokrmilnike družine AVR. Pisanje v jeziku C omogoča hitrejši razvoj kode, kot če bi pisali v zbirnem jeziku. Pri tem žrtvujemo visoko optimizacijo kode, a so sodobni prevajalniki tudi pri optimizaciji zelo dobri. Tudi za prevajanje kode se uporablja prosto dostopen prevajalnik, imenovan avr-gcc [7]. Ko je koda prevedena, moramo za njen prenos uporabiti programator, ki zna prevedeno kodo pravilno zapisati v dolgoročni spomin mikrokrmilnika. Na trgu obstaja kar veliko namenskih programatorjev, ki pa niso ravno poceni. Namesto namenskega smo kot programator uporabili Arduino UNO [8], saj je njegov nakup cenejši (če ga že nimamo). Za prenos prevedene kode smo nato uporabili še odprtokodno programsko opremo AVRDUDE [9] (AVR Download/UploaDEr).

3.3.5 Tokovna zaščita

Mikrokrmilnik ima zdaj vgrajeno tudi programsko kodo za nadzor tokovne porabe in ob prekoračitvi ustrezno ukrepa. Implementirana sta dva načina odzivanja. V prvem ob prekoračitvi nastavljene tokovne vrednosti mikrokrmilnik za dlje izklopi napajanje elektromotorja. Čas je trenutno nastavljen na 3 sekunde in se ga bo dalo v prihodnje nastavljati. Namen zaščite je, da

[7] https://gcc.gnu.org/wiki/avr-gcc
[8] https://www.arduino.cc/en/Tutorial/ArduinoISP
[9] http://www.nongnu.org/avrdude/

v primeru preobremenitve za dlje izklopimo motorje in se lahko sklep premakne v poljubno pozicijo ali pa se roka počasi začne spuščati proti tlom. Čeprav je krmiljenje motorjev izklopljeno, je konfiguracija krmilnega dela motorjev nastavljena tako, da na elektromotorju naredi kratek stik. Zaradi prisotnosti magnetnega polja in induktivnosti tuljave se elektromotor začne upirati vrtenju. Rezultat je počasnejše premikanje, kar omili nenadzorovano padanje.

Druga konfiguracija se od prve razlikuje le po tem, da krmiljenje motorjev izklopi za eno periodo krmiljenja. Primer uporabe je servomotor, ki skrbi za premikanje prstov. Konfiguracija omogoča, da se objekt zagrabi z ravno dovolj veliko silo ter se pri tem ne poškoduje in se elektromotor ne preobremeni.

Pri obeh pristopih se po pretečenem času krmiljenje elektromotorja samodejno omogoči in nastavi želeni cilj na trenutno pozicijo. Želeni tokovni meji se lahko nastavljata poljubno tudi med delovanjem, pri čemer se aktivira konfiguracija, ki ima najnižjo nastavljeno vrednost.

3.4 Regulator PID

3.4.1 Teoretična podlaga

V krmiljenih sistemih pogosto naletimo na kontrolerje s povratno zanko. V industriji je regulator PID (angl. proportional-integral-derivtive) med najpogosteje uporabljenimi kontrolerji s povratno zanko za krmiljenje procesov. Regulator neprekinjeno računa napako kot razliko med trenutnim in ciljnim stanjem sistema ter glede na podane parametre izračuna novo vrednost za krmiljenje. Sestavljen je iz treh med seboj neodvisnih delov, in sicer proporcionalnega, integralnega in odvodnega. Njegovo delovanje prikazuje enačba

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}, \tag{3.1}$$

kjer je $e(t)$ napaka v času t ter $u(t)$ kontrolna vrednost v času t in je vhod v sistem, ki ga želimo pripeljati v ciljno stanje. Vrednosti K_p, K_i in K_d

predstavljajo nastavljive uteži regulatorja PID in močno vplivajo na njegovo delovanje. Napačno nastavljene vrednosti lahko privedejo do zelo nestabilnega sistema z močnimi oscilacijami ali sistema, ki se odziva zelo počasi.

Proporcionalni del, sestavljen iz $K_p e(t)$, napako v času t skalira z vrednostjo K_p in tako na izhodu povzroči ustrezno spremembo (minimizira trenutno napako). Velike vrednosti povečajo odzivnost sistema in hkrati povečajo oscilacije, za razliko od manjših vrednosti, ki so lahko brez oscilacij, a se zaradi premajhne odzivnosti ne približajo ciljnemu stanju.

Integralni del, sestavljen iz $K_i \int_0^t e(\tau)d\tau$, integrira napako skozi čas in jo poskuša odpraviti (minimizira stacionarno napako). Velike vrednosti K_i skrajšajo odzivni čas, a je ta kljub temu počasnejši kot pri proporcionalnem delu.

Odvodni del, sestavljen iz $K_d \frac{de(t)}{dt}$, poskuša predvidevati obnašanje sistema in zmanjšuje oscilacijo (minimizira napako v prihodnosti), toda zaradi prevelike vrednosti K_d sistem nikoli ne doseže želenega stanja.

V diskretnih sistemih integralni in odvodni del enačbe (3.1) poenostavimo in nadomestimo z

$$K_i \int_0^t e(\tau)d\tau \approx K_i \sum_{n=1}^{k} e(t_n)\Delta t \tag{3.2}$$

$$K_d \frac{de(t)}{dt} \approx \frac{e(t_k) - e(t_{k-1})}{\Delta t}, \tag{3.3}$$

kjer Δt predstavlja časovni korak, $e(t_k)$ pa napako regulatorja v času t_k. Enačba regulatorja PID za diskretni sistem je tako

$$u(t_k) = K_p e(t_k) + K_i \sum_{n=1}^{k} e(t_n)\Delta t + \frac{e(t_k) - e(t_{k-1})}{\Delta t}. \tag{3.4}$$

3.4.2 Implementacija regulatorja PID

V poglavju 3.4.1 smo govorili o zgradbi regulatorja PID, sedaj pa bomo obravnavali njegovo implementacijo na mikrokrmilniku ATmega328P. Kot smo zasledili v poglavju 3.3.4, ima mikrokrmilnik omejeno zmogljivost procesiranja, kar smo pri implementaciji regulatorja tudi upoštevali.

Implementacija proporcionalnega in odvodnega dela je preprosta in prikazana v algoritmu 1. Primera vrednosti koeficientov iz enačbe 3.4 sta $K_p = 300$ in $K_d = 5$. Za natančnejše računanje prispevka odvoda bi bilo smiselno uporabiti manjšo vrednost faktorja spremembe časa in ustrezno večjo vrednost parametra K_d. Na primer vrednosti 256 (algoritem 1, vrstica 5) in $K_d = 5$ (algoritem 1, vrstica 6, *d_gain*) podata isti rezultat kot vrednosti 16 in $K_d = 80$. To bi omogočalo natančnejše nastavljanje parametra regulatorja in posledično tudi boljše krmiljenje.

Algorithm 1 Implementacija proporcionalnega, odvodnega in integralnega dela

```
1:  // Proporcialni del
2:  p_component = seek_position − current_position;
3:  pwm_output = (int32_t)p_component ∗ (int32_t)p_gain;
4:  // Odvodni del
5:  d_component = (p_component − p_component_old) ∗ 256;
6:  pwm_output+ = (int32_t)d_component ∗ (int32_t)d_gain;
7:  // Integralni del
8:  i_component+ = p_component;
9:  pwm_output+ = (((int32_t)i_component ∗ (int32_t)i_gain) >> 8);
10: //
11: pwm = ((int16_t)(pwm_output >> 8));
```

Večji izziv predstavlja implementacija integralnega dela. Težavo predstavlja omejenost mikrokrmilnika, saj v povratni zanki težko dosežemo želeno natančnost pri računanju parametra. Omejena natančnost je zelo izrazita pri računanju integralnega dela, kjer so vrednosti pogosto manjše od 1. Ker mikrokrmilnik nima strojne podpore za računanje števil s plavajočo ali fiksno vejico in je programska implementacija počasna, smo enačbo (3.2) prilagodili in izpostavili čas. Tako pri računanju vsote seštevamo le cela števila in čas upoštevamo šele pri računanju vrednosti prispevka integralnega dela. Implementacijo prikazuje algoritem 1, kjer vrstica 5 predstavlja skozi čas akumulirano vsoto, vrstica 6 pa prispevek integralnega dela. Šli smo še korak dlje in namesto deljenja s časom trajanja periode uporabili operacijo zamika

bitov v desno za 8 mest. To je ekvivalentno deljenju števila z vrednostjo 256, kar je tudi zelo dober približek frekvence posodabljanja. AVR žal podpira aritmetični ali logični pomik le za eno mesto. Kljub temu prevajalnik pri prevajanju pri uporabi pomika generira le 8, pri uporabi deljenja pa kar 76 bajtov kode. Pri programiranju v programskem jeziku C ne moremo napisati zelo optimalne kode, lahko pa prevajalnik usmerimo, kako naj pri prevajanju kodo optimizira.

Tabela 3.5 prikazuje vrednosti, ki smo jih uporabili za regulator PID (sklicujoč se na sliko 3.8 in tabelo 3.2). V algoritmu 1 v zadnji vrstici sledi še deljenje z 256 (pomik v desno za 8 bitov) za skaliranje vrednosti in kasneje (zaradi 8–bitnega PWM–signala) omejitev vrednosti na območje od -255 do 255 (predznak poda smer vrtenja elektromotorja).

Sklep	K_p	K_i	K_d
S1	500	300	5
S2	500	300	5
S3	500	300	5
S4	300	200	5
S5	150	200	5
Prijemalo	500	200	5

Tabela 3.5: Uporabljeni parametri regulatorja PID.

3.4.3 Frekvenca posodabljanja in analogno-digitalni pretvornik

Pri računanju vrednosti regulatorja PID se uporablja tudi čas, ki preteče od zadnjega računanja vrednosti, oziroma število ponovitev računanj v eni sekundi. Vrednosti regulatorja PID se pri projektu OpenServo računajo s frekvenco 100 Hz, kar je enkrat višja frekvenca kot pri originalnem krmilnem vezju servomotorja. Višja frekvenca pomeni hitrejšo regulacijo in hi-

trejše popravljanje (krajši odzivni čas), kar privede do manjših oscilacij in napak. Večjo natančnost in posledično tudi manjše oscilacije pridobimo tudi z natančnejšim analogno-digitalnim pretvornikom (krajše AD-pretvornik oz. v ang. ADC–analog to digital converter). ATmega328P ima le 10–bitni AD-pretvornik z napako kvantizacije $0,5$. To pomeni, da v primeru prave vrednosti 200,5 AD-pretvornik izmenično vrača vrednosti 200 in 201. Napake kvantizacije ne moremo odpraviti, lahko le zmanjšamo njen vpliv, kar dosežemo s povečanjem ločljivosti AD-pretvornika.

Povečanje ločljivosti lahko dosežemo z vgradnjo namenskega AD-pretvornika, česar nočemo, saj podraži izdelavo sistema. Želen rezultat lahko dosežemo tudi s postopkom večkratnega vzorčenja in decimacije [18]. Teorija v ozadju je dokaj zapletena in jo podrobneje opisuje članek [19]. Uporaba je preprostejša in v osnovi za vsak dodaten bit potrebujemo štirikrat več meritev. Primer izračuna prikazuje enačba (3.5). ADC_{10bit} predstavlja 10–bitno vrednost, ki jo vrne AD-pretvornik, n predstavlja želeno število dodatnih bitov in $ADC_{(10+n)bit}$ dobljeni rezultat.

$$ADC_{(10+n)bit} = \frac{\sum_{k=1}^{4^n} ADC_{10bit}}{2^n} \tag{3.5}$$

Pri trenutni konfiguraciji mikrokrmilnika eno branje AD-pretvornika traja približno $86,4\ \mu s$, kar omogoča frekvenco branj pri različni resoluciji, kot prikazuje tabela 3.6. Z znižanjem frekvence meritev lahko torej dosežemo višjo ločljivost.

Ločljivost	Čas branja	Največja frekvenca
10 bitov	$\approx 0,0864\ ms$	$\approx 11574\ Hz$
11 bitov	$\approx 0,3456\ ms$	$\approx 2893\ Hz$
12 bitov	$\approx 1,3824\ ms$	$\approx 723\ Hz$
13 bitov	$\approx 5,5296\ ms$	$\approx 180\ Hz$

Tabela 3.6: Z večanjem ločljivosti meritve se zmanjša frekvenca meritev.

Na podlagi tabele 3.6 in optimizacije pri računanju vrednosti regulatorja

PID smo izbrali 12–bitno resolucijo in frekvenco branja 256 Hz namesto prvotnih 100 Hz. Dejanska frekvenca branj je 256,99 Hz, saj je to najbližja frekvenca želeni, ki se jo da nastaviti na mikrokrmilniku. Izbrani vrednosti predstavljata zelo dober kompromis med natančnostjo in odzivnostjo.

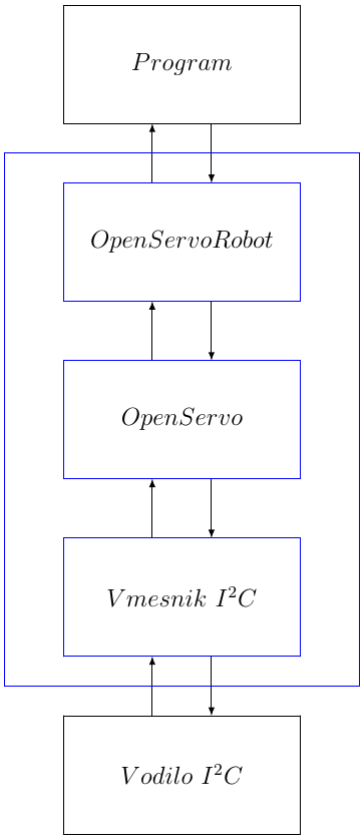
3.5 Komunikacija

3.5.1 Programska implementacija

Da bi zagotovili najboljšo prilagodljivost smo implementirali lastno programsko opremo. Razvita knjižnica tako skrbi za komunikacijo z napravami na vodilu. Zaradi modularnosti in lažje nadgradnje ter razširljivosti je implementirana v treh nivojih: komunikacija preko vodila I^2C , razreda OpenServo in OpenServoRobot, kar nazorno prikazuje slika 3.13. Razvoj je potekal na operacijskem sistemu Ubuntu 16,04 in brez težav deluje tudi na drugih distribucijah Linux. Knjižnica je napisana v programskih jezikih C in C++, pri čemer je za C++ zahtevana različica 11. Za razvoj lastne knjižnice smo se odločili zaradi slabe implementacije knjižnice projekta OpenServo.

Komunikacija preko vodila I^2C

Najnižjo plast predstavlja komunikacija preko vodila I^2C. Operacijski sistemi ne dovoljujejo neposrednega dostopa do strojne opreme, ampak moramo uporabiti ustrezne sistemske knjižnice, ki tudi poenostavijo delo s strojno opremo, a se med operacijskimi sistemi lahko razlikujejo. Najnižjo plast zato predstavlja ločeni del (minimalistične) knjižnice in je edini del, ki je povezan z operacijskim sistemom. To omogoča lažji prehod na drug operacijski sistem, kar je lažje tudi v primeru zamenjave vrste vodila. Je tudi edini del knjižnice, napisan v programskem jeziku C.



Slika 3.13: Zgradba komunikacije.

Razred OpenServo

Na drugi plasti se nahaja implementacija razreda OpenServo in je prva abstrakcija komunikacije. Razred je poimenovan OpenServo zaradi komunikacije s prvo plastjo in ima implementirane vse potrebne metode za komunikacijo z mikrokmilnikom, na katerem se nahaja programska oprema OpenServo. Omogoča branje in pisanje vseh parametrov mikrokrmilnika in ga lahko uporabljamo tudi samostojno. Vsako spremembo registrov na strani mikrokrmilnka je treba spremeniti tudi v tem razredu, sicer lahko pride do nestabilnega delovanja. Poleg implementacije, ki omogoča posamezno branje vseh registrov, vsebuje tudi metode, ki v bloku preberejo del ali vse registre. To zmanjša nepotrebno dodatno komunikacijo, ki se pojavi pri samostojnem

naslavljanju vsakega registra. Pri zaporednem branju več registrov mikrokrmilnik namreč samodejno povečuje naslov registra in podatke pošilja, dokler ne dobi ukaza za njegovo prekinitev. Razred zato vsebuje metodo, ki v bloku prebere vse registre, in metodo, ki v bloku prebere le registre, ki se stalno posodabljajo.

Razred OpenServoRobot

Na tretjem nivoju se nahaja implementacija razreda OpenServoRobot, ki je tudi najvišja abstrakcija komunikacije in je neposredno odvisna od razreda OpenServo. Razred še dodatno poenostavi komunikacijo z robotom in tudi skrije vse nižje nivoje. Na tem nivoju podamo model robota, v našem primeru DH–parametre manipulatorja. Model robota je lahko poljuben, s čimer dosežemo veliko prilagodljivost, kar je tudi naš cilj in ena izmed pomembnejših prednosti projekta OpenServo. Glede na podan opis razred poskrbi za ustrezno pretvarjanje med vrednostmi mikrokrmilnika in višjimi nivoji. Vsa nadaljnja komunikacija z višjimi nivoji tako prejema ustrezno pretvorjene vrednosti. Poleg tega vsebuje tudi niti (ang. thread), ki skrbijo za nemoteno in neblokirano komunikacijo. Podrobneje, vodilo I^2C deluje veliko počasneje kot procesor, in da pri pošiljanju in branju podatkov z mikrokrmilnika ne pride do blokiranja glavnega procesa, so vse zahteve za branje in pisanje dane v čakalno vrsto. Dodatna nit nato izvede operacije, ki se nahajajo v čakalni vrsti. Implementirano je tudi samodejno branje vseh parametrov mikrokrmilnika, ki se med delovanjem stalno spreminjajo. Branje teh vrednosti poteka s frekvenco 30 Hz.

3.5.2 Fizična implementacija

Obstaja veliko možnosti priklopa robotskega manipulatorja na osebni računalnik. Ena od teh je z uporabo namenskega čipa, ki je vmesnik, kot na primer čip FT232H [10] znamke FTDI. Vmesnik ima tako na eni strani priklop

[10] http://www.ftdichip.com/Products/ICs/FT232H.htm

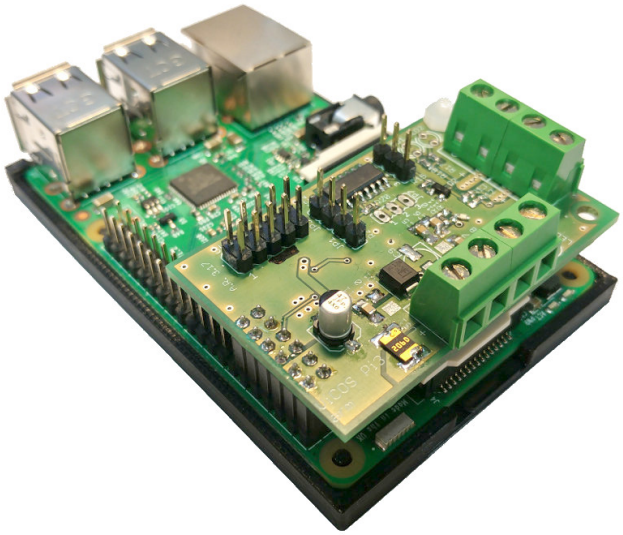
za vodilo I^2C in na drugi strani USB-priklop.

Primarno pa je bil naš sistem razvit z namenom delovanja tudi na manj zmogljivih računalnikih. Zaradi svoje sistemske nezahtevnosti torej brez težav deluje tudi na Raspberry Pi 3 [11]. To je majhen, a polno opremljen računalnik, ki temelji na procesorju ARM, s priklopi USB in HDMI ter splošno namenskimi vhodno-izhodnimi konektorji (ang. GPIO). Poleg vseh omenjenih priklopov ima na razpolago tudi vodilo I^2C, na katerega lahko priklopimo le naprave, ki delujejo na napetosti 3,3 V. Ker krmilno vezje deluje na napetosti 5 V, potrebujemo čip, ki ustrezno pretvori napetostne nivoje. Ravno tako je treba pred uporabo v sistemu omogočiti še delovanje [12] vodila I^2C. Izdelali smo lastno tiskano vezje, na katerem se nahajajo vse potrebne dodatne komponente za ustrezno komunikacijo, priklop zunanjega napajanja za Raspberry Pi 3 in servomotorje ter ojačevalec za zvočne efekte. Raspberry Pi 3 z dodatnim vezjem prikazuje slika 3.14, shema vezja pa je priložena v prilogi B.

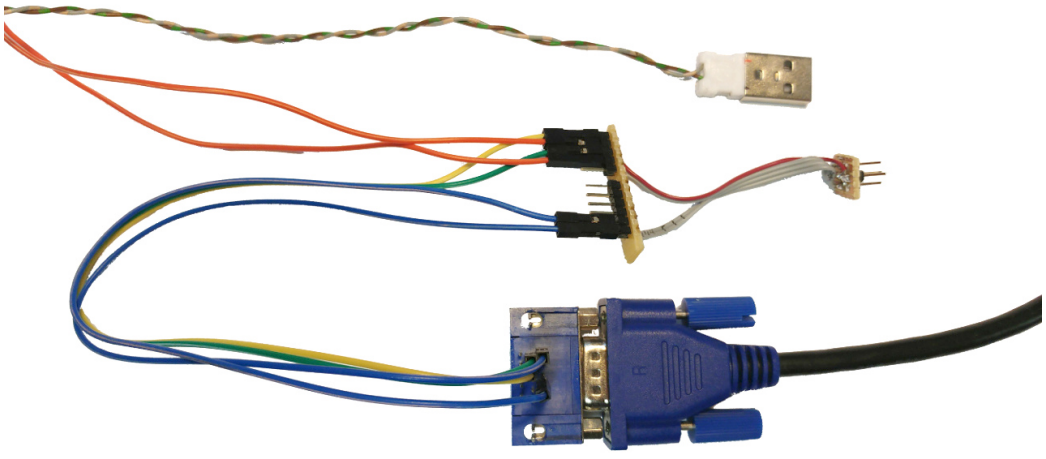
Obstaja pa še cenejša in lažje dostopna rešitev. Vodilo I^2C se v napravah pogosto uporablja za nezahtevno komunikacijo s posameznimi moduli. Tako se v računalnikih uporablja za komunikacijo z zunanjim monitorjem [13] [14]. Operacijski sistem s pomočjo komunikacije tako izve ločljivost monitorja in še nekaj dodatnih informacij (ang. DDC – display data channel [15]). To lahko izkristimo in tako za komunikacijo uporabimo kar priklop z VGA-konektorjem. Zaradi majhne tokovne zmogljivosti $+5\ V$ izhoda na konektorju je pri priklopu več kot dveh krmilnih vezij obvezno treba zagotoviti ustrezno napajanje. Primer uporabe priklopa VGA za komunikacijo z robotskim manipulatorjem prikazuje slika 3.15.

Takšen način uporabe sicer poceni in poenostavi krmiljenje robotskega manipulatorja, a ni idealen. Pri uporabi v računalnik vgrajene opreme mo-

[11] https://www.raspberrypi.org/
[12] https://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i2c
[13] https://en.wikipedia.org/wiki/VGA_connector
[14] https://en.wikipedia.org/wiki/HDMI
[15] https://en.wikipedia.org/wiki/Display_Data_Channel



Slika 3.14: Slika prikazuje Raspberry Pi 3 z vezjem za komunikacijo z robotskim manipulatorjem.



Slika 3.15: Primer uporabe kabla VGA za namen komunikacije I^2C z dodatnim napajanjem preko konektorja USB.

ramo biti previdni, saj lahko v nasprotnem primeru poškodujemo komponente in onemogočimo pravilno delovanje računalnika.

3.6 Integracija

Integracija zgoraj opisane knjižnice v različne sisteme je preprosta in nezahtevna ter jo lahko uporabimo tudi pri razvoju lastnega sistema. Knjižnico smo uspešno integrirali v dva sistema in jo uporabljali tudi v preprostem programu (deluje v terminalu), s katerim smo imeli poln dostop in nadzor nad delovanjem krmilnih vezij.

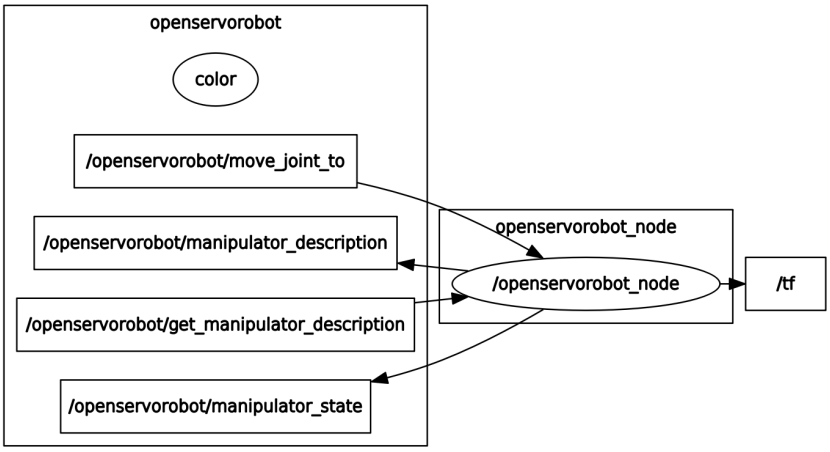
3.6.1 Integracija sistema v ROS

ROS (Robot Operating System) [20] [16] je pogosto uporabljen sistem za delo z različnimi roboti. Omogoča preprosto komunikacijo med različnimi moduli (ang. node) preko pošiljanja sporočil. Njegovo jedro skrbi za prejemanje in distribucijo sporočil vsem modulom, ki so na to sporočilo naročeni. Na voljo je tudi veliko orodij in drugih pripomočkov ter gonilnikov, ki zelo poenostavijo razvoj in uporabo sistema za želenega robota.

Zaradi preproste in modularne zasnove gonilnika, opisanega v poglavju 3.5, je njegova integracija v okolje ROS preprosta. Napisali smo nov modul in vanj integrirali gonilnik. Poleg modula smo ustvarili še vsa sporočila za sporočanje parametrov, ki se pošiljajo med posameznimi moduli v okolju ROS, saj vsa obstoječa sporočila za nas niso bila ustrezna. Primer ustvarjenih sporočil prikazuje slika 3.16.

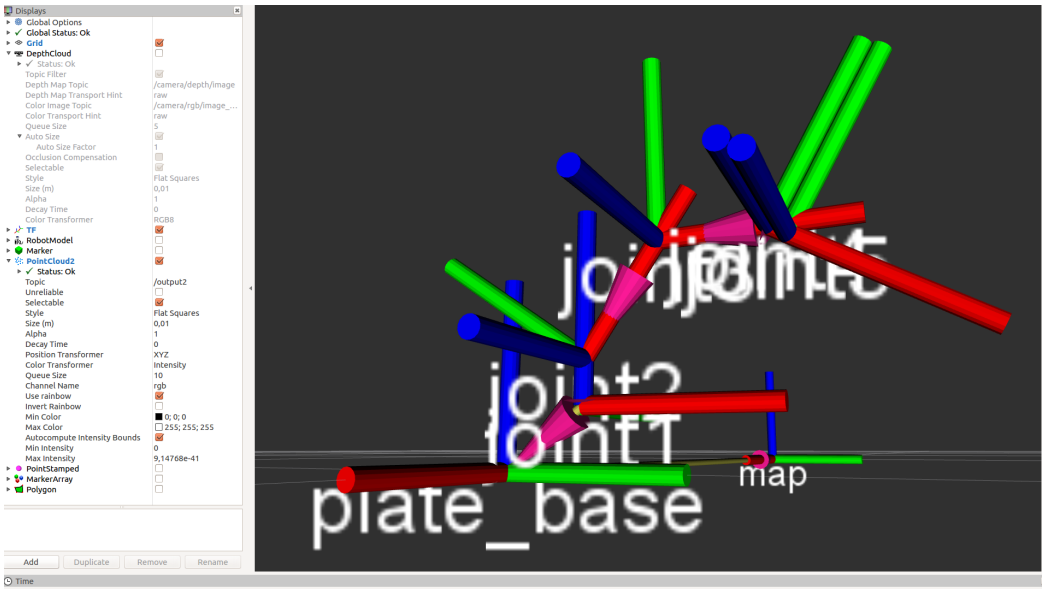
Ena od že definiranih vrst sporočil je tudi sporočanje lege v prostoru (ang. transformer oziroma tf v okolju ROS). Sporočilo smo tudi uporabili, saj omogoča preprosto pošiljanje lege, pri čemer podamo le izhodišče, premik in zasuk. Trenutno lego segmentov robotskega manipulatorja lahko spremljamo v 3D prikazovalniku RViz, kar prikazuje slika 3.17. Ta se lahko uporablja za

[16] http://www.ros.org/



Slika 3.16: Zgradba modula v okolju ROS.

vizualizacijo različnih 3D objektov, kot so roboti, objekti ali oblak točk s senzorja globine.



Slika 3.17: V RVizu prikazane pozicije posameznih sklepov.

Robotski manipulator lahko tako preprosto krmilimo z računalnikom in

ga uporabljamo samostojno ali pritrdimo na mobilno platformo. Z integracijo v okolje ROS namreč pridobimo veliko dodatne funkcionalnosti, saj ima ROS na voljo veliko različnih modulov. Primer modulov so detekcija in prepoznavanje predmetov s pomočjo barvne in globinske kamere, iskanje treaktorij, kako naj se robotski manipulator približa zaznanemu objektu in kako ga naj zagrabi, in veliko več.

3.6.2 Integracija sistema v Manus

Manus [17] je odprtokodno ogrodje za manipulacijo z robotskimi manipulatorji in je namenjen predvsem za učne namene. Avtor dr. Luka Čehovin Zajc je sistem zasnoval kot podporo, ogrodje, za poučevanje dela z robotskimi manipulatorji.

Podrobneje, je ogrodje, ki je primarno namenjeno za delo z robotskimi manipulatorji. Sestavljajo ga različni moduli, ki med seboj komunicirajo preko sporočil. [18] Komunikacija je podobna kot v okolju ROS, vendar s pomembno razliko. Razvita je bila s poudarkom na učinkovitosti in minimalističnosti z namenom optimalnega delovanja tudi na manj zmogljivih računalnikih, kot je Raspberry Pi 3.

Osnovni moduli, ki jih prikazuje slika 3.18, so:

- *zajem slike*: Modul v intervalih s kamere zajema sliko in jo nato pošlje vsem modulom, ki so naročeni na prejemanje slik;
- *komunikacija z robotom*: Modul skrbi za komunikacijo s fizičnim robotskim manipulatorjem in ostalimi moduli;
- *spletni vmesnik*: Modul predstavlja manjši strežnik, na katerega lahko dostopamo preko lokalne mreže. Vsebuje spletni vmesnik z vizualnim prikazom roke (izris roke glede na prebrane lastnosti segmentov in v primeru zajemanja slik, ustrezno umesti na sliko). Z uporabo drsnikov

[17] https://github.com/lukacu/manus
[18] https://github.com/vicoslab/echolib

omogoča tudi preprosto premikanje sklepov robotskega manipulatorja, pozicije lahko shranimo in tako sestavimo zaporedje premikov. Da bi lahko robotsko platformo uporabljali tudi najmlajši uporabniki, ima na razpolago tudi programski vmesnik za vizualno programiranje, ki temelji na okolju Blockly [19];

- *registracija kamere*: Na zajeti sliki išče oznake podlage in umesti kamero v prostor. To se uporablja za določanje delovne ravnine, pravilno postavitev in izris robotskega manipulatorja ter zaznanih objektov v prostor z uporabo tehnike obogatene resničnosti (ang. AR–augmented reality).

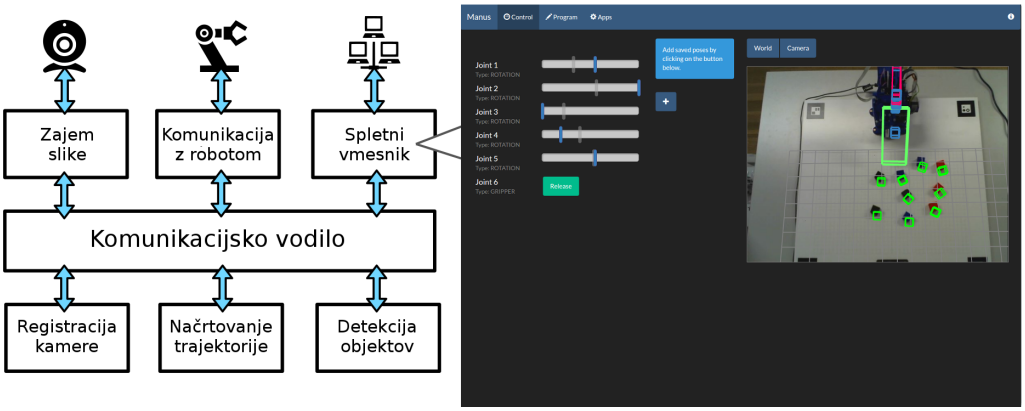
- *načrtovanje trejaktorije*: Modul vsebuje odprto kodno knjižnico *Orocos KDL* [21] [20], ki se uporablja za direktno in inverzno kinematiko. Modulu podamo trenutno in želeno ciljno pozicijo ter dobimo zaporedje premikov sklepov;

- *zaznava objetkov*: Modul s pomočjo računalniškega vida na sliki poišče objekte in vrne njihovo pozicijo in orientacijo. Slika 3.18 prikazuje pozicijo in orientacijo zaznanih kock različnih barv.

Omenili smo, da Manus vsebuje modul *komunikacija z robotom*, ki skrbi za komunikacijo z robotskim manipulatorjem in je tudi modul, v katerega smo integrirali knjižnico za komunikacijo. Podobno kot pri integraciji gonilnika v okolje ROS, je tudi integracija v okolje Manus sorazmerno preprosta. V datoteki za ustvarjanje oblik sporočil opišemo zgradbo sporočil, ki se pošiljajo med moduli, in modul povežemo s knjižnico za komunikacijo.

Poleg krmiljenja in programiranja robotskega manipulatorja preko *spletnega vmesnika* ogrodje omogoča tudi druge pristope. Preko lokalne mreže lahko do robotskega manipulatorja dostopamo tudi iz programskega okolja Matlab. Za najzahtevnejše scenarije pa je priporočljivo pisanje modulov, ki neposredno komunicirajo s sistemom Manus.

[19] https://developers.google.com/blockly/
[20] http://www.orocos.org/kdl



Slika 3.18: Programska zgradba ogrodja Manus in zaslonska slika spletnega vmesnika.

Poglavje 4

Evalvacija

V tem poglavju se bomo osredotočili na evalvacijo tiskanega vezja, servomotorjev in izdelanega robotskega manipulatorja. Pri evalvaciji tiskanega vezja se bomo posvetili izračunom izgub na tokovno kritičnem delu in vrednosti primerjali z originalnim vezjem projekta OpenServo. Evalvacija servomotorjev zajema meritve napak AD-pretvornika in krmiljenja. Pri evalvaciji robotskega manipulatorja bomo določili delovni prostor, glede na parametre manipulatorja in servomotorjev izračunali teoretično natančnost manipulatorja ter nato ovrednotili še ponovljivost.

4.1 Evalvacija tiskanega vezja

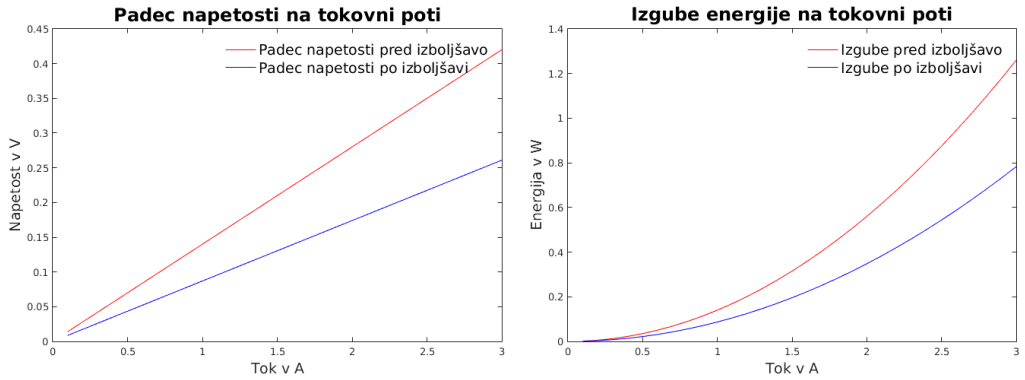
Evalvacija tiskanega vezja zajema evalvacijo tokovno najbolj obremenjenega dela. To je del vezja, ki skrbi za krmiljenje elektromotorja.

Poleg že omenjenih sprememb na krmilnem vezju imajo določene spremembe vpliv tudi na učinkovitost delovanja. Ta se pokaže predvsem kot manjše izgube in posledično manjše segrevanje komponent. Podrobneje, izdelano vezje uporablja boljše močnostne tranzistorje, z manjšo notranjo upornostjo. Ravno tako je nižja upornost upora, ki se ga uporablja pri merjenju toka. Zaradi nižje upornosti je padec napetosti na komponenti nižji (enačba (4.1)) in posledično je energija (enačba (4.2)), ki se prevede v toploto, manjša

$$U = I * R \tag{4.1}$$

$$P = I^2 * R. \tag{4.2}$$

Manjše izgube na komponentah pripomorejo k boljši zmogljivosti servomotorja kot celote. Grafa na sliki 4.1 primerjata vrednosti vsot padcev napetosti in energije izgub komponent na originalnem in posodobljenem krmilnem vezju. Vrednosti, uporabljene pri izračunih, so bile vzete iz uradne dokumentacije komponent [1] [2], pri čemer smo upoštevali pričakovane vrednosti, podane pri napetosti delovanja 4,5 V. Vrednost upora za merjenje toka je pred nadgradnjo znašala $0,05\ \Omega$, po njej pa $0,022\ \Omega$.



Slika 4.1: Izračunane vrednosti padcev napetosti in toplotnih izgub pred izboljšavo in po njej.

Kot vidimo na sliki 4.1, pri 3 A znaša padec napetosti 0,42 V pred izboljšavo in 0,26 V po njej. Zaradi padcev napetosti se pojavi tudi izguba energije, ki pri 3 A pred nadgradnjo znaša 1,26 W in po njej 0,78 W. Izguba po nadgradnji je tako manjša za tretjino, kar pripomore k malo boljši zmogljivosti servomotorja, malo manjši porabi energije in predvsem manjšemu segrevanju komponent, kar zmanjša verjetnost njihovega pregrevanja.

[1] https://www.infineon.com/dgdl/irf7317pbf.pdf?fileId=5546d462533600a4015355f5b8de1b3d
[2] https://www.infineon.com/dgdl/irf7307pbf.pdf?fileId=5546d462533600a4015355f20d211b0e

4.2 Evalvacija servomotorjev

V tem poglavju se bomo posvetili evalvaciji servomotorjev. S pomočjo procesiranja slik smo na preprost način določili razpon oziroma največji kot, ki ga lahko doseže servomotor. Nato smo se posvetili še meritvi natančnosti in ponovljivosti. Meritve bomo izvedli za vse uporabljene servomotorje in tako pridobili manjkajoče podatke, potrebne za določanje specifikacije robotskega manipulatorja.

4.2.1 Merilni sistem

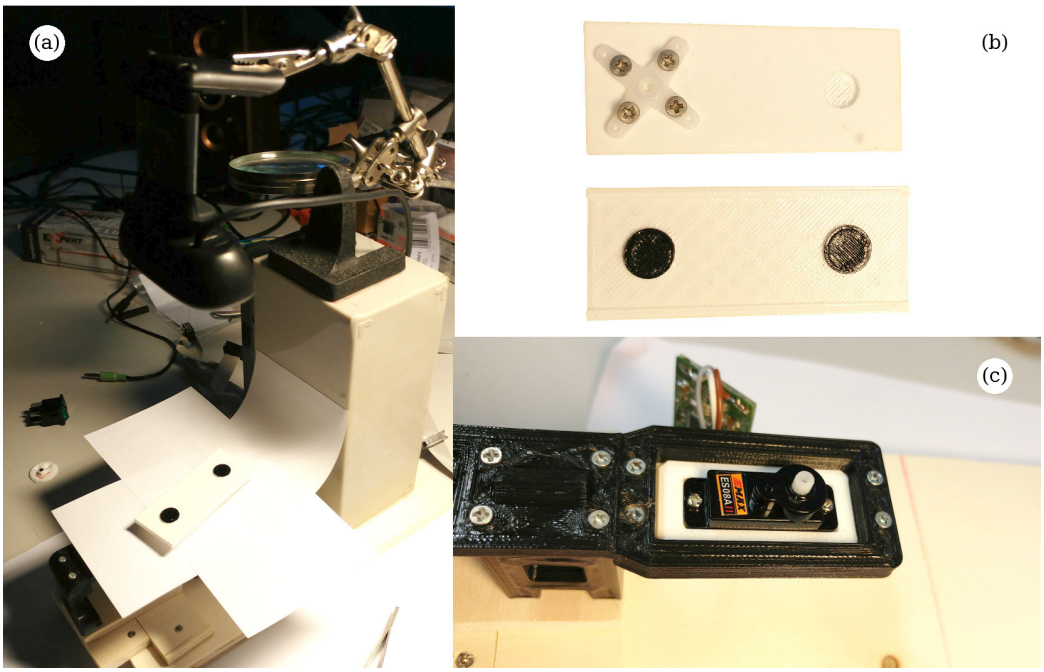
V poglavju 3.2.2 smo se seznanili s podrobnostmi RC-servomotorjev in njihovimi prednostmi in pomanjkljivostmi. Ena izmed pomanjkljivosti je, da ne poznamo celotnega območja delovanja ter ustrezne preslikave med vrednostjo, ki jo vrne AD-pretvornik, in kotom zasuka.

Vsi servomotorji imajo vgrajene rotacijske potenciometre z omejenim fizičnim in električnim zasukom. Fizični zasuk, kot že ime pove, omejuje fizični premik oziroma kot zasuka gredi. Električni kot zasuka omejuje območje, pri katerem se na izhodnem konektorju še pojavi sprememba upornosti in je (običajno) manjša od fizičnega kota zasuka. Vse lastnosti potenciometrov proizvajalci navedejo v dokumentaciji, a za potenciometre, vgrajene v servomotorje, nismo uspeli najti ustrezne dokumentacije.

K problemu pretvorbe med kotom zasuka in vrednostjo, ki jo vrne AD-pretvornik, smo pristopili na sledeč način. S pomočjo računalnika smo izdelali 3D model z dvema diskoma, ki smo ga natisnili s 3D tiskalnikom in pritrdili na servomotor. S kamero smo nato zajeli sliko in na njej z uporabo računalniškega vida poiskali dva kroga. Merilni sistem prikazuje slika 4.2. Glede na zaznani središči obeh krogov smo nato izračunali naklon premice s pomočjo formule

$$angle = \tan^{-1}(\frac{y_1 - y_2}{x_1 - x_2}), \tag{4.3}$$

pri čemer točka (x_1, y_1) predstavlja sredino prvega kroga, točka (x_2, y_2) pa sredino drugega kroga. Rezultat je podan v radianih.



Slika 4.2: (a) Merilni sistem s kamero in (b) ploščica s cilindroma, ki jima s pomočjo računalniškega vida določimo sredino ter (c) držalo servomotorja.

Ker pa običajno meritve niso odporne na napake, smo najprej ocenili napako merilnega sistema. Postopek smo izvedli v petih ponovitvah in pri vsaki zajeli 100 meritev. Za napako smo vzeli primer z največjim standardnim odklonom. Dobljene rezultate prikazuje tabela 4.1.

	Min	Max	Povprečje	Mediana	Standardni odkon
°	29,9498	29,9811	29,9665	29,9666	0,0063

Tabela 4.1: Izmerjene vrednosti napake meritve, zaokrožene so na štiri decimalna mesta.

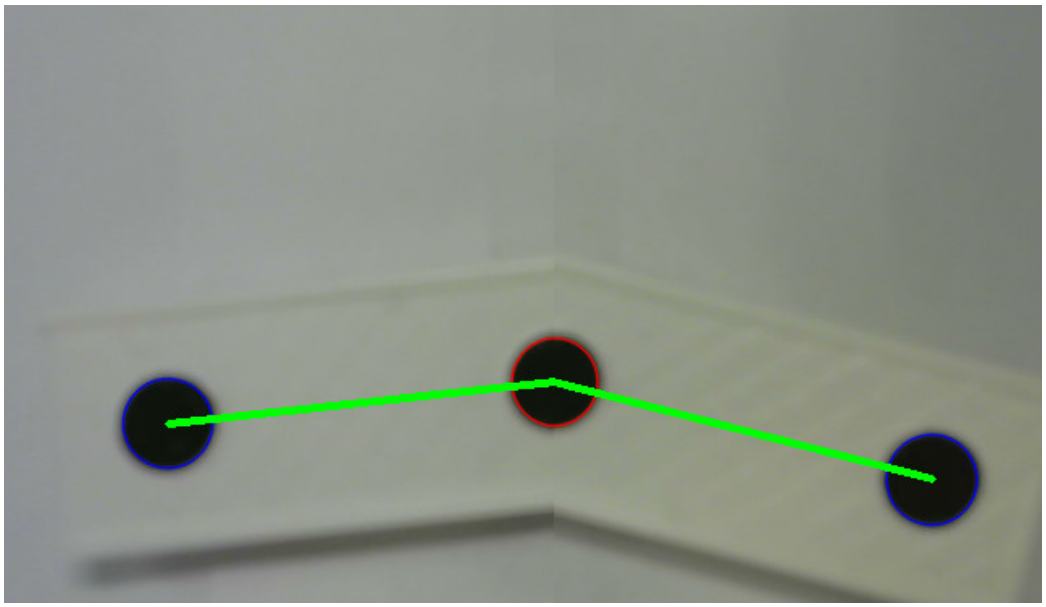
Izmerjeni standardni odklon napake je sorazmerno majhen in znaša 0,0063°. Največja pričakovana napaka meritve, ocenjena s standardnim odklonom napake 3σ, torej znaša 0,02° (vrednost je zaokrožena na drugo decimalno mesto)

in je še vedno dovolj majhna za oceno kota. Predlagana metoda meritev je torej primerna za merjenje kotov.

4.2.2 Območje delovanja

Območje delovanja servomotorja podaja območje oziroma kot zasuka, v katerem se gred servomotorja nemoteno vrti in se hkrati na potenciometru spreminja upornost. Oba podatka sta potrebna za izračun vrednosti, ki pretvarja vrednost AD-pretvornika v stopinje.

Območje delovanja smo izmerili s pomočjo predhodno opisanega merilnega sistema. Gred smo desetkrat zasukali v obe skrajni meji in v vsaki zajeli 100 meritev. Primer meritve prikazuje slika 4.3. Razlike povprečnih vrednosti opravljenih meritev in izračun vrednosti za pretvarjanje prikazuje tabela 4.2.



Slika 4.3: Slika prikazuje največji kot zasuka gredi servomotorja HS-645. Zasuk poteka z leve proti desni v smeri urinega kazalca.

Zanimiv podatek tabele 4.2 je stolpec *Razpon*, ki prikazuje največji zasuk gredi. Vsi uporabljeni servomotorji in tudi večina RC-servomotorjev imajo

Servomotor	Razpon	ΔAD delcev	$°/AD$	rad/AD
HS-645MG	200,62°	3745,03	0,05357	0,000935
HS-485HB	201,10°	3772,09	0,05331	0,000931
HS-311	202,01°	3758,45	0,05375	0,000938
ES08AII	201,71°	4095,00	0,04926	0,000860

Tabela 4.2: Izmerjena območja delovanja in vrednosti za pretvarjanje pri uporabi 12-bitnega AD-pretvornika.

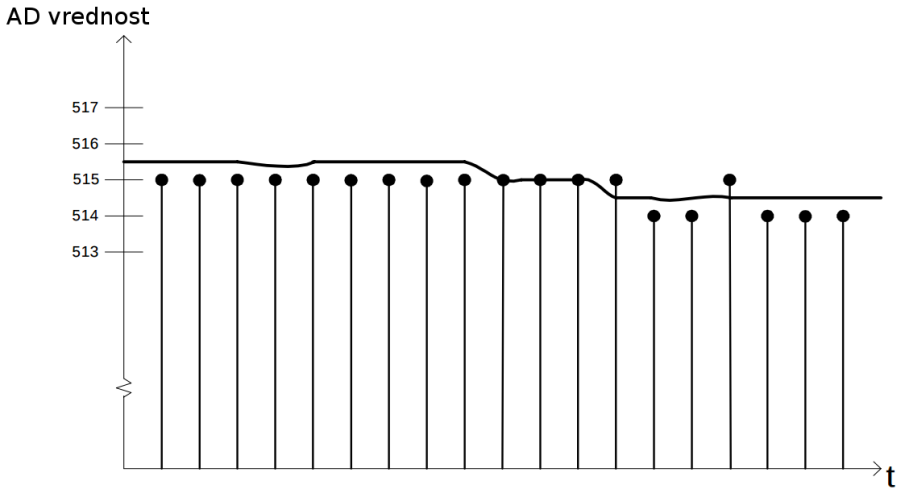
zasuk omejen na 180°, v tabeli 4.2 pa opazimo, da so predelani servomotorji dosegli približno 200°. Z uporabo lastnega krmilnega vezja torej dosežemo večji zasuk gredi, kar predstavlja dodatno prednost.

Poznavanje obeh skrajnih meja na potenciometru je pomembno tudi za pravilno krmiljenje. Vrednosti sta shranjeni na mikrokrmilniku, ki skrbi, da se zasuk izvaja le znotraj območja, in v primeru napačno podane pozicije, gred zasuče le do shranjene skrajne meje. V mikrokrmilniku shranjeni skrajni meji se lahko tudi razlikujeta od vrednosti skrajnih meja, a morata biti znotraj intervala. Pri kalibraciji končnega sistema, na primer robotskega manipulatorja, imamo lahko segment, ki se zasuče le za 100° in pri katerem se meji razlikujeta od mej servomotorja. Na podlagi dobljenih vrednosti lahko zasnujemo končni sistem in ocenimo območje delovanja.

4.2.3 Natančnost meritev AD-pretvornika

Napake pri meritvah se pojavljajo tudi pri pretvarjanju analogne vrednosti v digitalno. Tu se pojavi napaka kvantizacije in je posledica končno mnogega števila vrednosti pri pretvarjanju. Napako nazorno prikazuje slika 4.4, na kateri se analogna vrednost nahaja med obema stopnicama. Po specifikacijah mikrokrmilnika napaka zaradi kvantizacije znaša ± 0,5 [16]. Zaradi nepopolnosti sistema na meritev vplivajo tudi šum v napetosti, kakovost izdelave potenciometra in mnogi drugi dejavniki, za katere bomo predpostavljali, da je njihov vpliv zanemarljiv, in jih ne bomo obravnavali. Merili smo

tako, da smo vsak servomotor premaknili na neko pozicijo, izklopili napajanje motorjev (kar zagotovi, da se gred res ne premika) in opravili 100 meritev AD-pretvornika.



Slika 4.4: Vizualizacija vpliva decimacije. Vrednost AD-pretvornika (pike) in realna vrednost (neprekinjena črta) [18].

Dobljeni rezultati, ki jih prikazuje tabela 4.3, so zelo zanimivi in jih lahko povežemo v dve skupini. V prvi skupini sta servomotorja HS-645 in ES08AII s povprečno vrednostjo na sredini dveh celih vrednosti in napako povprečja približno ± 0,50. V drugi skupini sta servomotorja HS-485 in HS-311, pri katerih je povprečna vrednost skoraj celo število in napaka povprečja skoraj ± 0,55. Oba primera prikazujeta, da pri kvantizaciji napaka znaša približno ± 0,5. Razlog, da v drugem primeru dobimo ocenjeni standardni odklon malenkost večji je, da je prava vrednost zelo blizu celemu številu, recimo 3021. Zaradi šuma nato vrednost naraste na približno 3021,5 ali pade na približno 3020,5 in zaradi kvantizacijske napake obstaja večja verjetnost, da dobimo vrednosti 3022 ali 3020. Kvantizacijske napake ne moremo odpraviti, lahko le zmanjšamo njen vpliv s povečanjem ločljivosti AD-pretvornika, kar smo storili v poglavju 3.4.3. Z upoštevanjem podatkov iz tabele 4.2 vpliv standardnega odklona decimacije v stopinjah znaša ± 0,0268° za HS-645MG,

± 0,0282° za HS-485, ± 0,0301° za HS-311 in ± 0,0246° za ES08AII. Vrednosti so sorazmerno majhne in pri zasnovanem robotskem manipulatorju ne predstavljajo večje pomanjkljivosti.

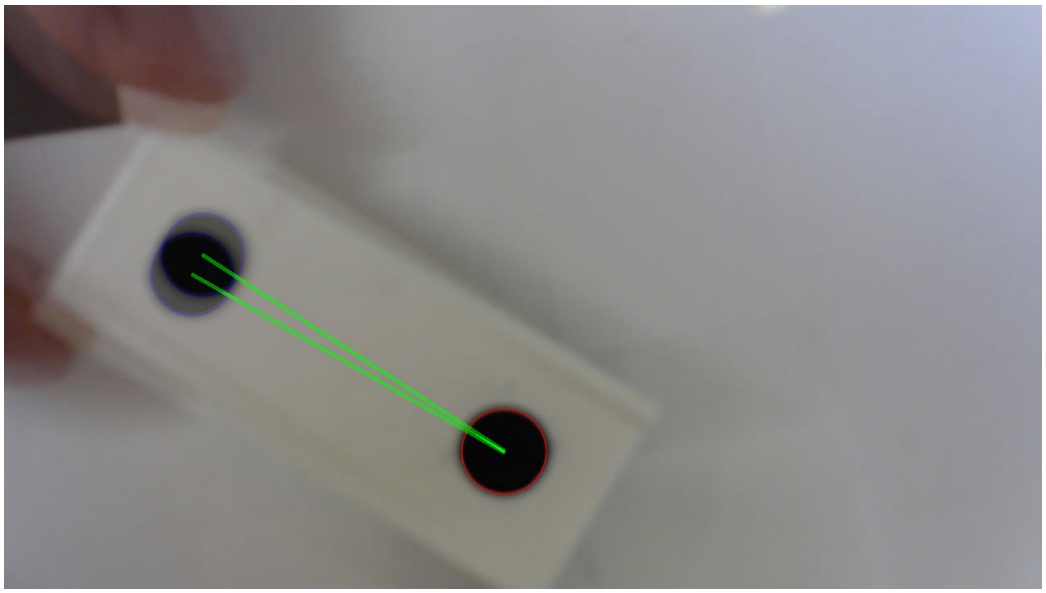
Servomotor	Min	Max	Povprečje	Stand. odklon	Stand. odklon
HS-645MG	1218	1219	1218,46	0,50	0,0268°
HS-485HB	3020	3022	3020,98	0,53	0,0282°
HS-311	1217	1219	1217,92	0,56	0,0301°
ES08AII	815	816	815,54	0,50	0,0246°

Tabela 4.3: Vrednosti meritev in izračunana standardni odklon, ki je za lažje razumevanje pretvorjena tudi v stopinje.

4.2.4 Napaka zaradi izdelave

Zaradi zgradbe pri servomotorjih naletimo na dodatno odstopanje. Povezano je z zobniškimi prenosi, njihovo kakovostjo izdelave in kakovostjo izdelave servomotorja kot celote. Zračnost (ang. backlash) je pri zobniških prenosih pogost pojav [22] in je povezana z obliko zobnika in kakovostjo izdelave ter se povečuje z obrabljenostjo. Primer zračnosti na gredi servomotorja prikazuje slika 4.5.

Zračnost pri zobniških prenosih vpliva na odzivnost in natančnost krmiljenja. Več kot je prenosov, večji je vpliv zračnosti in posledično je večja napaka krmiljenja, kar lahko v določenih primerih privede do neželenih oscilacij. Vzrok je, da elektromotor potrebuje več časa, da dovolj zavrti vse zobnike, pri tem pa pridobi večji vrtilni moment, zaradi česar ponovno potrebuje več časa za zaustavitev in lahko posledično prekorači želeno pozicijo. Tabela 4.4 prikazuje meritve zračnosti uporabljenih servomotorjev tako v delcih AD-pretvornika kot tudi v stopinjah. Za meritev zračnosti smo z majhno silo premaknili gred v eno in nato drugo stran ter zajeli 50 meritev, pri čemer smo pazili, da se gred motorja ni premaknila. Meritev je le ocena, saj bi za



Slika 4.5: Vpliv zračnosti servomotorja EW08AII, prikazuje, za največ koliko se lahko gred servomotorja zasuka, ne da bi se zasukala gred elektromotorja.

pravilno in točnejšo meritev morali pritrditi gred elektromotorja in na gred servomotorja delovati s točno določeno silo.

Servomotor	Min	Max	Standardni odklon	Standardni odklon
HS-645MG	3045	3048	0,75	0,04°
HS-485HB	3022	3027	1,49	0,06°
HS-311	1215	1224	4,29	0,15°
ES08AII	1195	1224	14,06	0,87°

Tabela 4.4: Vrednosti meritev in izračunane napake tako v delcih AD-pretvornika kot tudi v stopinjah.

Izmerjene napake, prikazane v tabeli 4.4, sovpadajo s pričakovanji in rangom servomotorjev. Najmanjšo napako ima HS-645, ki je izmed vseh najboljše kakovosti in uporablja kovinske zobniške prenose. Nato mu z rangom

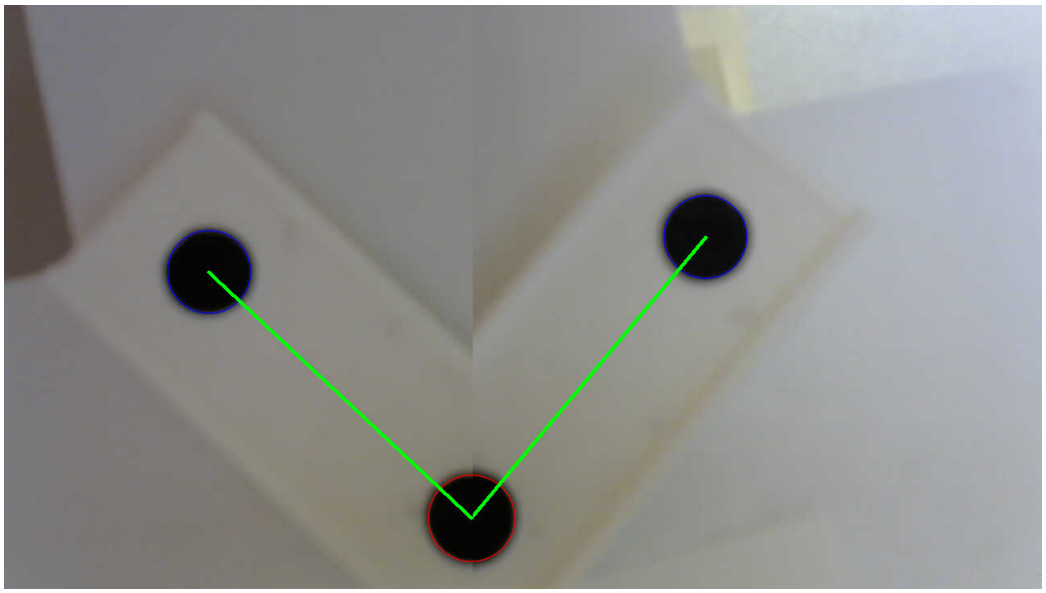
nižje sledi HS-485 in še nižje HS-311, oba s plastičnimi zobniškimi prenosi iz različne umetne mase. Najslabše se je obnesel ES08AII z največjo zračnostjo in vgrajenimi plastičnimi zobniškimi prenosi iz najlona.

4.2.5 Ponovljivost

Pri krmiljenih sistemih je zelo pomemben podatek o ponovljivosti sistema. Zanimata nas odstopanje in natančnost sistema, ki opišeta porazdelitev napak meritve oziroma povesta pričakovano napako. Odstopanje izhodišča dobimo kot srednjo vrednost napak oziroma absolutno razliko do referenčne vrednosti. Za natančnost pa vzamemo razpršenost okoli povprečne vrednosti.

Za meritev smo programsko krmilili kot zasuka gredi servomotorja. Opravili smo 25 zasukov v izhodišče in pri vsakem zajeli 10 vrednosti AD-pretvornika. Primer izvajanja meritve prikazuje slika 4.6. Dobljeni rezultati so prikazani v tabeli 4.5 in so med seboj zelo raznoliki. Najnižje odstopanje in najboljšo natančnost doseže servomotor HS-645MG, najslabšo pa HS-485HB. Vzrokov za toliko slabše dobljene vrednosti pri HS-485HB je lahko več: slaba izdelava elektromotorja, kar vpliva na slabšo odzivnost elektromotorja in posledično večjo prekoračitev zasuka gredi, šum pri pretvarjanju analogne vrednosti potenciometra v digitalno in podobno. Odstopanje in natančnost HS-311 in ES08AII sta zadovoljiva, še posebej ker sta oba veliko cenejša kot HS-645MG in HS-485HB.

Kljub odstopanjem so dobljeni rezultati zadovoljivi, še posebej zaradi nizke cene izdelave. Glede na dobljene vrednosti lahko zdaj izračunamo in ocenimo natančnost robotskega manipulatorja. Za doseg najboljših rezultatov pri krmiljenju robotskega manipulatorja bi bilo treba opraviti analizo več različnih servomotorjev in nato izbrati najprimernejšega.



Slika 4.6: Slika prikazuje samodejno premikanje gredi servomotorja med dvema pozicijama za merjenje točnosti in natančnosti.

Servo-motor	Iskana pozi-cija	Min	Max	Povp-rečje	Odsto-panje	σ	Odsto-panje	σ
HS-645MG	1221	1219	1222	1220,38	0,62	0,79	0,033°	0,043°
HS-485HB	1221	1216	1222	1219,30	1,70	1,64	0,091°	0,087°
HS-311	1221	1219	1222	1219,76	1,24	0,77	0,066°	0,041°
ES08AII	1221	1219	1226	1220,65	0,35	1,77	0,017°	0,087°

Tabela 4.5: Vrednosti meritev in izračunane vrednosti odstopanj in standardnega odklona tako v delcih AD-pretvornika kot tudi v preračunanih stopinjah.

4.3 Evalvacija robotskega manipulatorja

V poglavju 4.2 smo se posvetili analizi uporabljenih servomotorjev, pridobljene meritve pa bomo zdaj uporabili za določanje parametrov manipulatorja. Posvetili se bomo delovnemu prostoru robotskega manipulatorja, ter ocenili njegovo teoretično natančnost in ponovljivost robotskega manipulatorja.

4.3.1 Delovni prostor

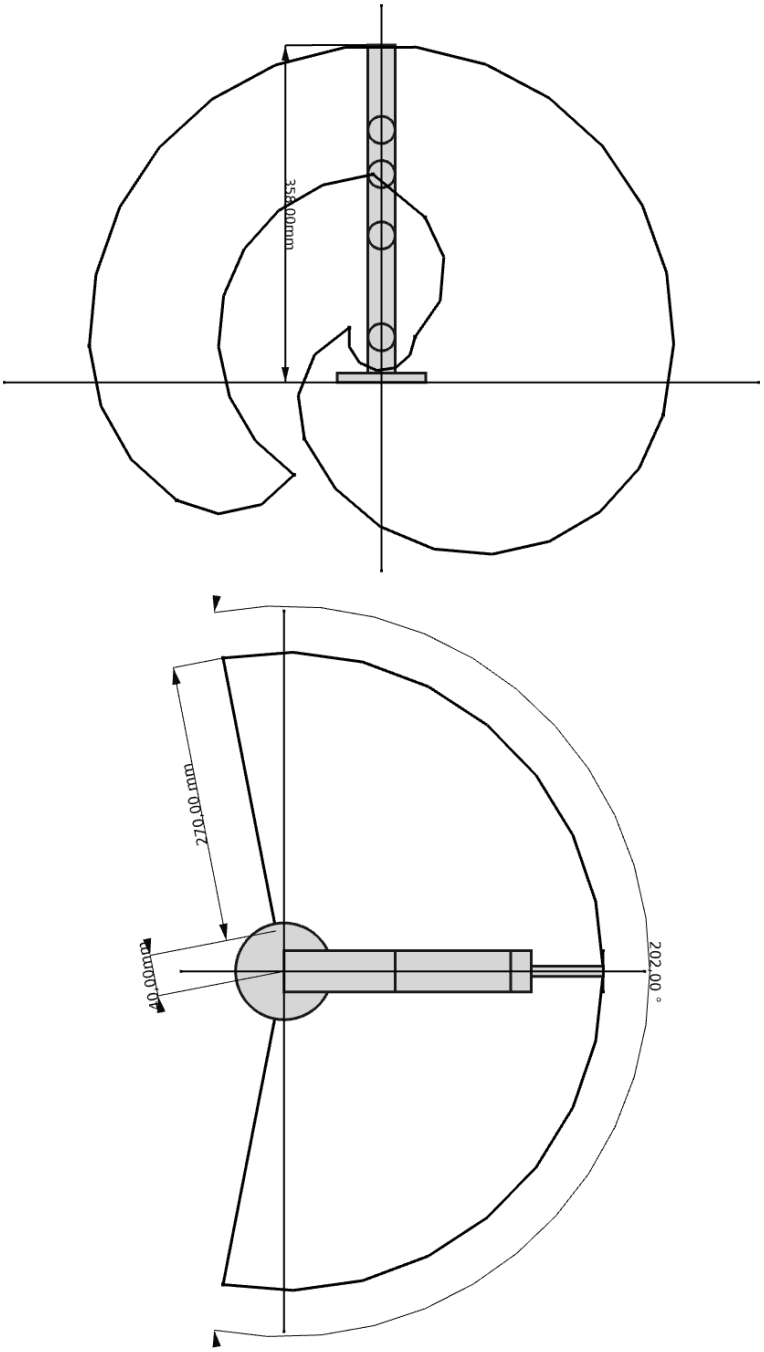
Delovni oziroma dosegljivi prostor robota predstavlja prostornino, v kateri lahko vrh manipulatorja doseže vsako točko [8]. Poznamo tudi priročni delovni prostor, ki predstavlja prostornina, v kateri lahko manipulator doseže vsako točko ob poljubni orientaciji prijemala. Zaradi končnih dimenzij zapestja in prijemala je ta prostor vedno manjši od dosegljivega prostora in je tem večji, čim krajše je prijemalo manipulatorja.

Delovni prostor je odvisen od modela robotskega manipulatorja in implementacije pogonskega dela. V poglavju 4.2.2 smo izmerili območje delovanja servomotorjev. Pove nam, koliko se lahko zasuka os servomotorja in posledično ustreznega segmenta, kar posledično tudi omeji končni delovni prostor in obliko. Slika 4.7 tako prikazuje območje, ki ga vrh našega manipulatorja lahko doseže.

4.3.2 Teoretična natančnost

Teoretična natančnost nam pove, kolikšno odstopanje ocenjene pozicije od resnične pozicije robotskega manipulatorja lahko pričakujemo, če upoštevamo lastnosti pogonskega dela in robotskega manipulatorja. Hkrati predstavlja tudi teoretično največjo natančnost robotskega manipulatorja.

V programskem okolju Matlab smo vzorčili vrednosti kotov zasukov posameznih sklepov v skladu s porazdelitvami iz tabele 4.5, pozicijo pa smo izračunali z enačbo (2.7). Za vsak nabor kotov smo ob vzorčenju geometrijskega modela manipulatorja izračunali pozicijo brez napak in nato za isto



Slika 4.7: Delovni prostor robotskega manipulatorja. Prikazuje tudi prostor, ki ga v trenutni konfiguraciji ne doseže, saj del prostora sega vanj.

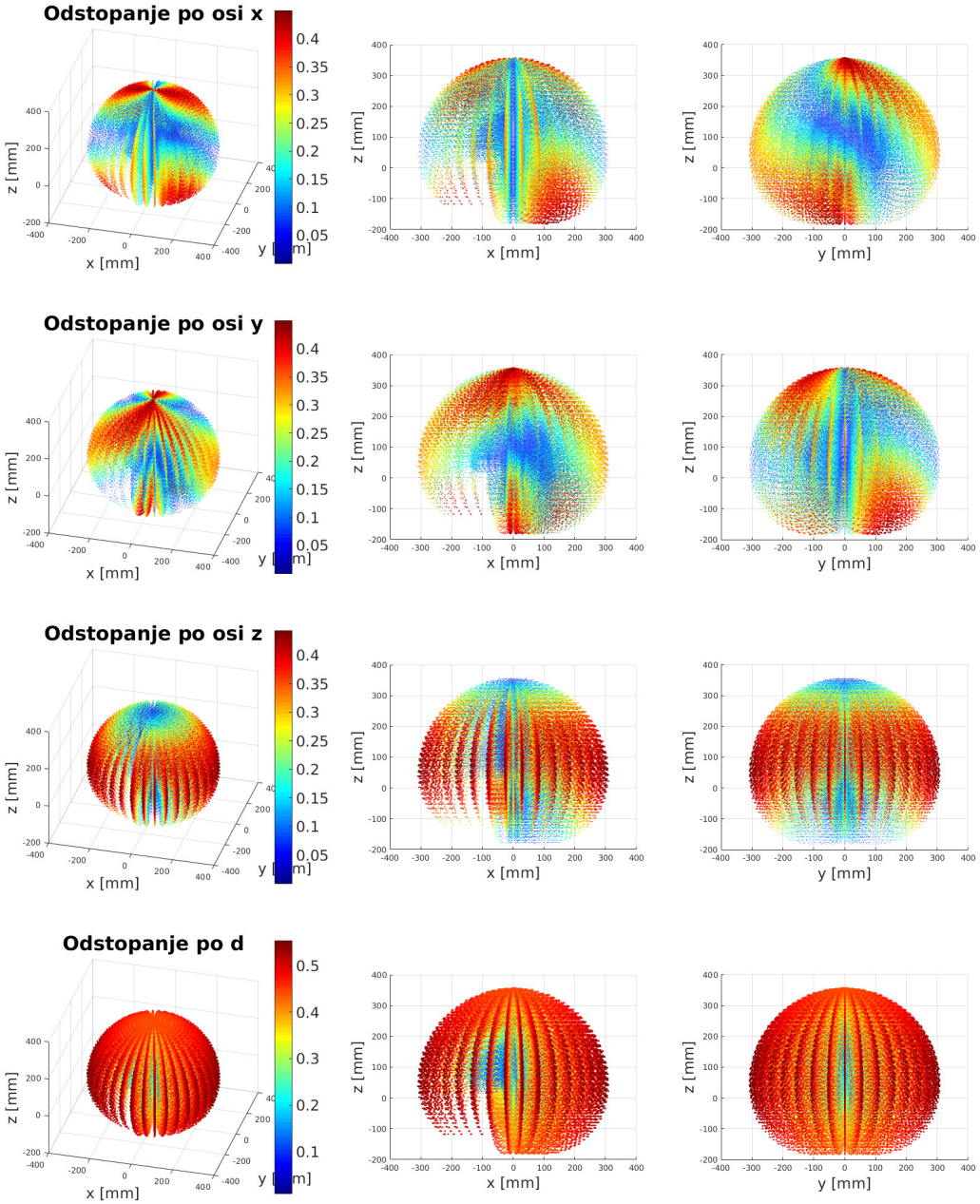
izhodišče vzorčili še 21 odstopanj vsakega segmenta. Dobljene pozicije smo nato odšteli od pozicije brez šuma ter izračunali odstopanje in standardni odklon. Zaradi kombinatorične zahtevnosti računanja smo omejili število pozicij manipulatorja za vzorčenje, in sicer smo pri segmentih 1, 2, 3 in 4 (slika 3.8) število korakov iz ene v drugo skrajno lego servomotorja omejili na 20 in pri segmentu 5 na 1 (segment ne vpliva na pozicijo vrha manipulatorja, ampak le na orientacijo). Povzetek dobljenih rezultatov prikazuje tabela 4.6, slika 4.8 pa prikazuje 3D razporeditev odstopanj v območju delovanja robotskega manipulatorja. Za lažjo vizualizacijo slika 4.9 prikazuje še največjo napako v preseku dveh ravnin v območju delovanja robotskega manipulatorja.

Osi	Odstopanje (mm)			Standardni odklon (mm)		
	Najmanj	Največ	Povprečno	Najmanj	Največ	Povprečno
x	0,00	0,45	0,14	0,01	0,61	0,20
y	0,00	0,45	0,14	0,00	0,61	0,21
z	0,00	0,44	0,16	0,03	0,56	0,66
d	0,10	0,55	0,29	0,14	0,73	0,41

Tabela 4.6: Glede na lastnosti servomotorjev in DH–parametre robotskega manipulatorja izračunana odstopanja in standardni odklon po oseh (neodvisno) in skupni evklidski razdalji (d).

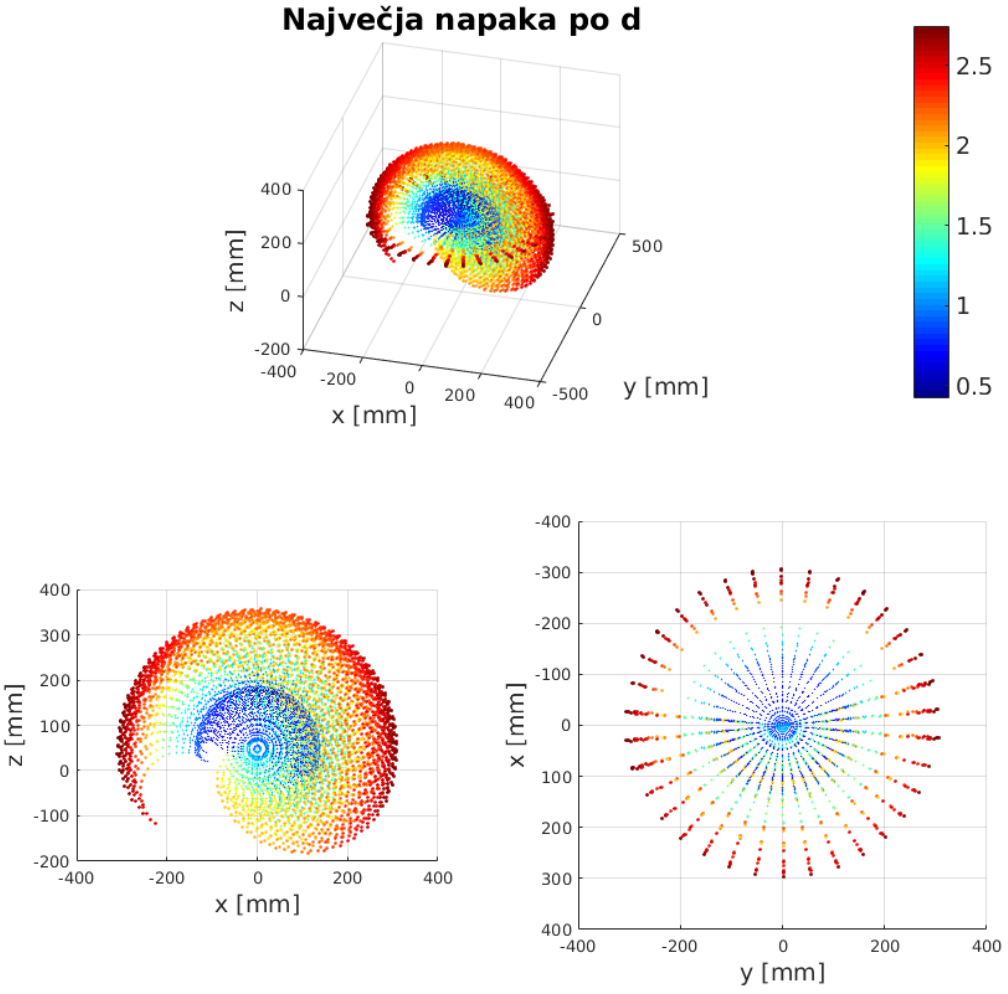
Rezultati v tabeli 4.6 prikazujejo odstopanje in standardni odklon pozicije vrha robotskega manipulatorja, pri čemer so podane vrednosti po oseh x, y in z neodvisne, d pa predstavlja skupno evklidsko razdaljo v vsaki poziciji vzorčenja. Zasledimo, da so vrednosti *odstopanja* manjše kot vrednosti *standardnega odklona*. Večji standardni odklon nakazuje občuten vpliv šuma.

Slika 4.8 nazorno prikazuje območja, v katerem pride do največjih odstopanj. Opazimo, da je odstopanje po skupni evklidski razdalji največje na robu delovnega prostora robotskega manipulatorja. To je pričakovano, saj se vpliv majhne spremembe kota z razdaljo povečuje. Pri odstopanjih po



Slika 4.8: Vrednosti odstopanj po vseh oseh (x, y in z) in skupni evklidski razdalji (d), z istim koordinatnim izhodiščem kot na sliki 2.3.

posameznih oseh pa opazimo, da je porazdelitev odstopanj različna. Tako pri x, y in z na robu delovnega prostora obstajajo tako območja s sorazmerno velikim odstopanjem kot območja z majhnim odstopanjem, pri vseh pa se z bližanjem izhodišču sistema napaka odstopanja zmanjšuje. Zanimivo je tudi, da se največja odstopanja po oseh nikoli ne prekrivajo, kar je razvidno tudi iz tabele 4.6, saj je največja skupna evklidska razdalja od največjih vrednosti po oseh večja le za 0,10 mm.



Slika 4.9: Največja napaka po skupni evklidski razdalji na ravnini z–x in x–y.

Poleg odstopanj in standardnega odklona smo izračunali tudi največjo skupno evklidsko napako, katere porazdelitev v prostoru prikazuje slika 4.9. V najslabšem primeru znaša 2,74 mm in v povprečju 1,52 mm. Presek ravnine z–x nazorno prikazuje porazdelitev napak robotskega manipulatorja, ravno tako presek ravnine x–y. Opazimo pa tudi, da lahko zaradi prikazovanja skupne evklidske razdalje ravnino z–x zavrtimo okoli osi z in tako dobimo porazdelitev napake za celoten delovni prostor.

Zaključimo lahko, da največja teoretična napaka robotskega manipulatorja znaša 3 mm, ki pa je za nizkocenovni robotski manipulator dovolj majhna in primerna glede na namen uporabe. Pri tem moramo upoštevati, da je to le teoretična ocena napake in je v resnici napaka poziciji robotskega manipulatorja lahko večja. Hkrati pa teoretična ocena nazorno prikazuje porazdelitev napake v prostoru.

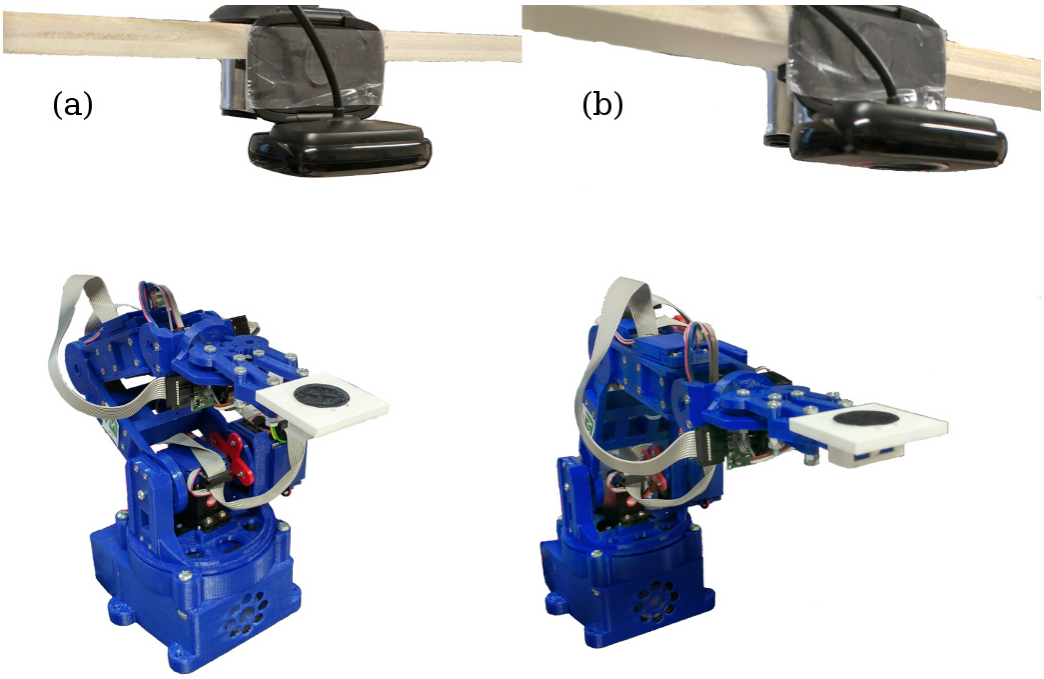
4.3.3 Ponovljivost

Da bi tudi empirično izmerili natančnost lastnega sistema, bi morali zelo natančno izmeriti pozicijo oz. lego vrha robotskega manipulatorja v 3D prostoru. Ker ustreznega merilnega sistema nismo imeli, smo meritev poenostavili in jo izvedli v 2D s premiki po ravnini (x_0, y_0) izhodiščnega koordinatnega sistema, prikazanega na sliki 2.3.

Avtomatizirano merjenje ponovljivosti robotskega manipulatorja smo tudi tokrat izvedli s pomočjo računalniškega vida. Merilni sistem je preprost in ga predstavljata na robotski manipulator pritrjeni oznaka in kamera (slika 4.10). Tudi ta sistem ni odporen na napake in zaradi preprostosti izdelave natančnost meritev znaša $0,2\ mm$, kar pa je še vedno primerno za oceno ponovljivosti.

Ponovljivost smo izmerili za dve poziciji, prikazana na sliki 4.10, pri čemer smo določili izhodiščno pozicijo in 7 dodatnih pozicij iz katerih smo manipulator premikali v izhodiščno pozicijo. Povzetek rezultatov prikazuje tabela 4.7, slika 4.11 pa vrednosti 30 zajetih meritev.

Iz tabele 4.7 lahko povzamemo, da je ponovljivost roke boljša bližje iz-

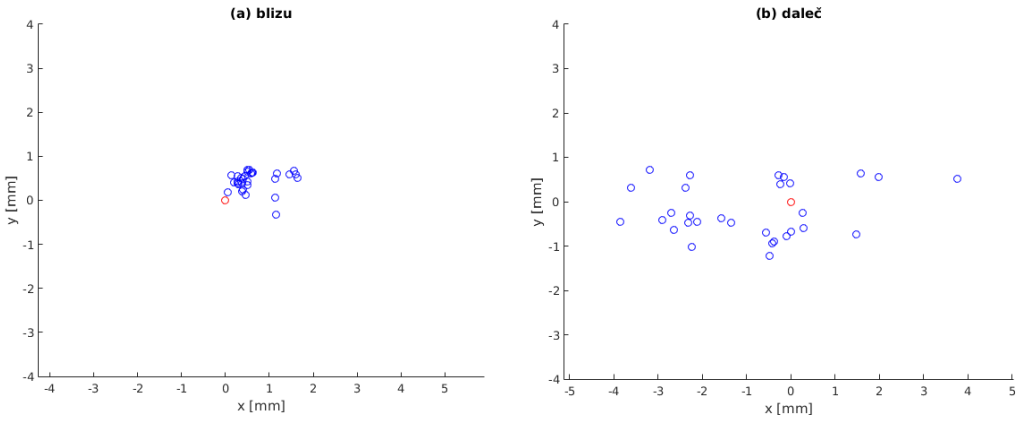


Slika 4.10: Robotski manipulator v obeh izhodiščnih pozicijah: (a) predstavlja blizu in (b) predstavlja daleč.

Meritev	Os x	Os y
Blizu	1,4 mm	0,6 mm
Daleč	3,8 mm	0,7 mm

Tabela 4.7: S pomočjo merilnega sistema izmerjena največja odstopanja po oseh x in y v primeru malo (*Blizu*) in skoraj v celoti iztegnjenega robotskega manipulatorja (*Daleč*).

hodišču in je proti zunanjemu robu delovnega prostora vse slabša. Zaradi poravnanosti robotskega manipulatorja z osjo x je ponovljivost po osi y veliko manjša kot po osi x, saj v tem primeru na napako po osi y vpliva le napaka prvega segmenta. Iz zajetih meritev in napake merilnega sistema lahko povzamemo, da znaša ponovljivost izdelanega robotskega manipula-



Slika 4.11: Odstopanja (v milimetrih) pozicije po oseh x in y: (a) malo, *Blizu*, in (b) bolj iztegnjenega, *Daleč*, robotskega manipulatorja. Rdeč krog predstavlja izhodišče.

torja po oseh x in y približno 4 mm.

Meritve ponovljivosti smo izvedli tudi ročno, in sicer smo v prostoru izbrali dve točki, robotski manipulator premikali med njima in merili pozicijo premika. Za lažje izvajanje meritev smo postopek merjenja ponovljivosti razdelili na dva dela. Prvi del se osredotoči na premikanje segmentov 2, 3 in 4, pri čemer se segment 1 (zasuk baze) ne premika. Pri drugem delu se osredotočimo le na prvi segment, ki rotira bazo, ostali sklepi pa so fiksni.

Pri prvem delu sta bili točki med seboj oddaljeni 50 mm in na višini 70 mm. V koordinatnem sistemu prvega segmenta (S1 s slike 3.8) se je vrh manipulatorja premikal po osi x, osi y in z pa sta nespreminjajoči. Točki sta bili izbrani tako, da so se premikali vsi trije segmenti. Meritve so bile izvedene s pomočjo ravnila. Rezultati meritev prvega dela so zelo spodbudni. Prepotovana razdalja je vedno večja od 48 mm in manjša od 52 mm. Pri obeh točkah so bila odstopanja znotraj milimetra, tako po osi x kot tudi po osi z.

Drugi del meritev zajema le zasuk baze pri fiksnih preostalih sklopih. Tudi tokrat sta točki med seboj oddaljeni 50 mm in na višini 70 mm, vendar

približno na dveh tretjinah radija delovnega prostora. Rezultati so podobni prej omenjenim. Tudi tokrat je celotna prepotovana razdalja med 48 mm in 52 mm in se v večini primerov točkama približa na manj kot 1 mm. Rezultat sovpada z oceno napake zasuka gredi servomotorja na isti oddaljenosti prijemala.

Vzrok za različne rezultate pri ročnem in avtomatiziranem merjenju je, da se je pri ročnem merjenju robotski manipulator premikal le med dvema pozicijama, ki sta bila sorazmerno blizu in tudi v območju, kjer je glede na sliko 4.11 največja napaka manjša. Zaradi bližine točk je bila tudi hitrost premikanja robotskega manipulatorja nižja, za razliko od avtomatiziranega postopka, ki se je premikal med zelo raznolikimi in oddaljenimi pozicijami (kar predstavlja tudi realnejše delovanje), zato je bila tudi hitrost premikanja večja. Kljub temu je ponovljivost za manipulator zadovoljiva in jo v najslabšem primeru ocenjujemo na 4 mm.

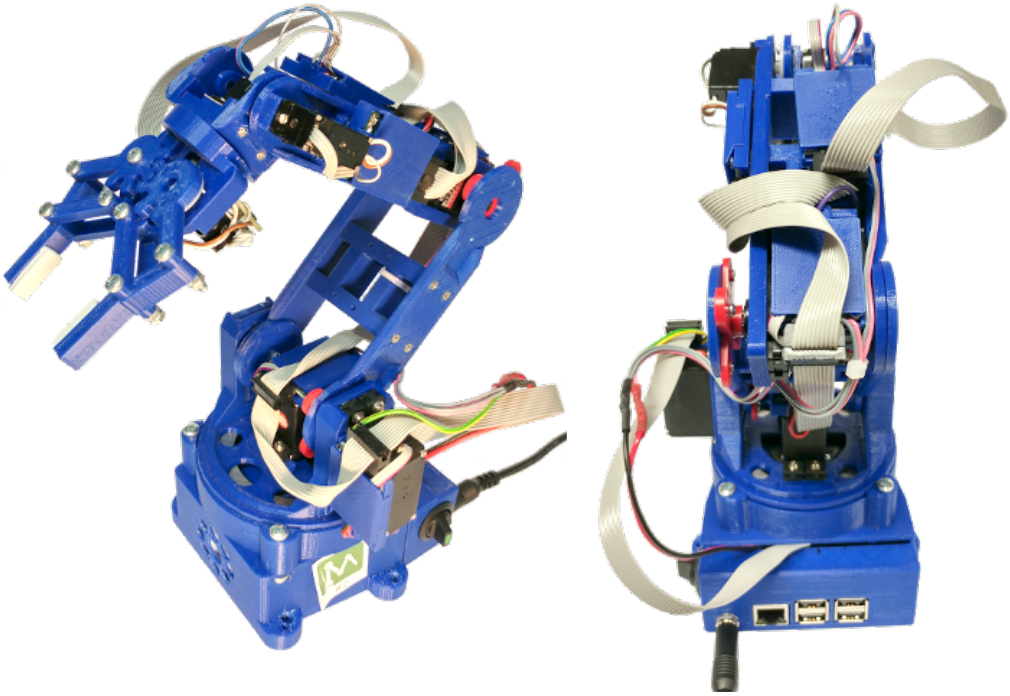
Poglavje 5

Zaključek

Izdelan robotski manipulator s podanimi specifikacijami v tabeli 5.1 in prikazan na sliki 5.1 je dosegel praktično vsa naša pričakovanja. Robotski manipulator je kompakten in ravno dovolj velik za uporabo na mizi ter je kljub svoji majhnosti dovolj zmogljiv za poučevanje. Nosilnostjo znaša 80 g, dejanska nosilnost pa bi lahko bila večja, a smo zaradi manj zmogljivega napajalnika omejili tokove na servomotorjih. Zelo zadovoljni smo tudi z doseženima natančnostjo in prilagodljivostjo. Uspeli smo ohraniti tudi nizko ceno izdelave, ki je s ceno komponent približno 300 €. Kosovnica s popisom uporabljenega materiala se nahaja v dodatku C.

Za dosego cilja smo mikrokrmilnik pripeljali skoraj do njegovih skrajnih meja. Z uporabo večkratnega vzorčenja smo povečali resolucijo AD-pretvornika, ki zdaj znaša 12 bitov namesto 10 bitov. Z večjo resolucijo natančneje ocenimo pozicijo in izboljšamo točnost robotskega manipulatorja. Do skrajnih meja smo pripeljali tudi odzivnost sistema, natančneje hitrost posodabljanja. Zdaj se parametri posodabljajo s frekvenco 256 Hz namesto prvotnih 100 Hz, s čimer dosežemo hitrejše in natančnejše krmiljenje.

Zadovoljni smo tudi s široko uporabnostjo krmilnega vezja in zmogljivostjo. Zaradi modularne zasnove imamo namen sistem uporabljati tudi pri drugih aplikacijah, na primer pri izdelavi sistema z dvema prostostnima stopnjama, na katerega namestimo kamero in jo lahko ustrezno rotiramo. Z



Slika 5.1: Izdelan robotski manipulator, pripravljen za uporabo.

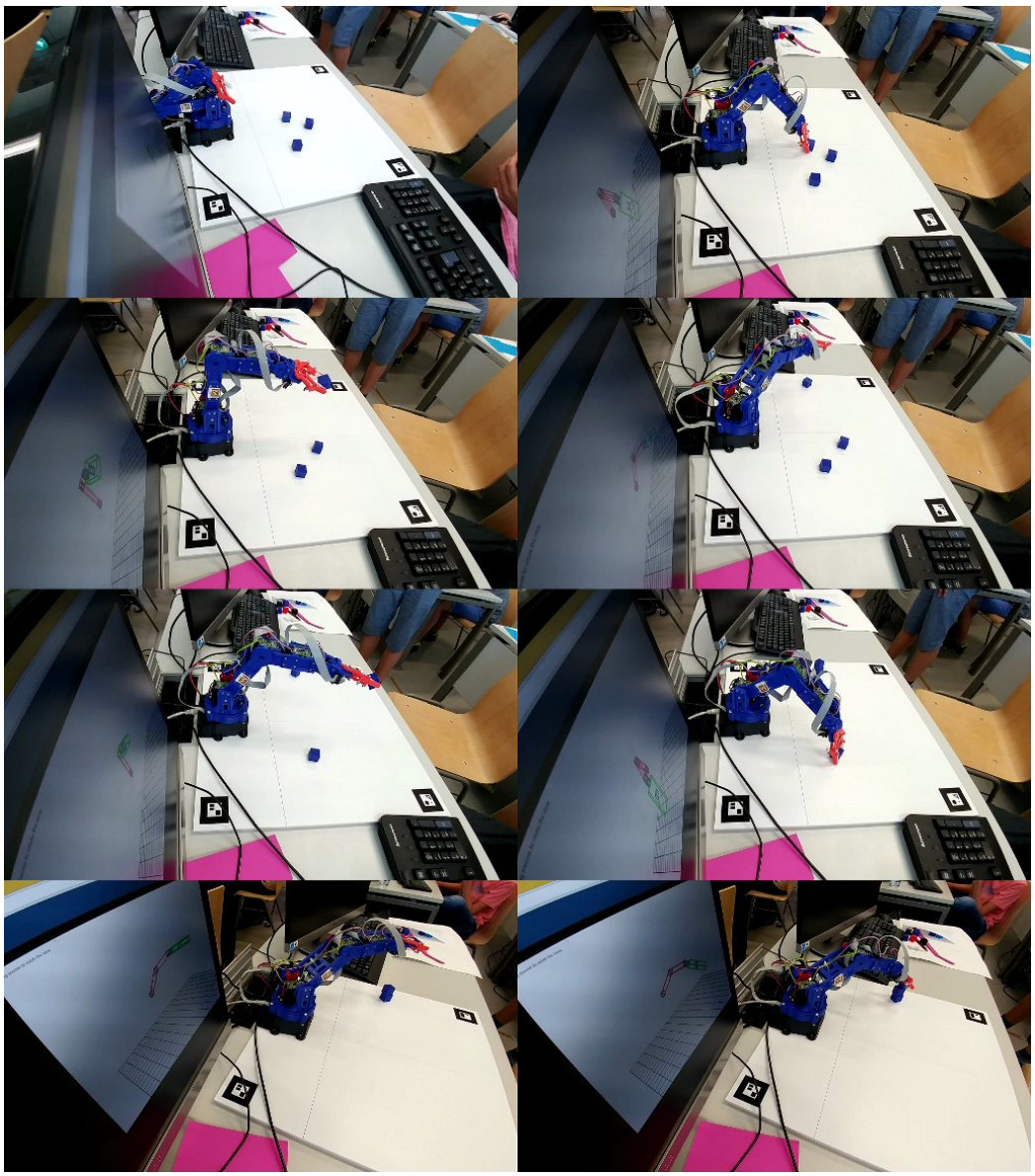
Lastnost	Vrednost
Višina	358 mm
Radij	310 mm
Ponovljivost	4 mm
Teža	842 g
Nosilnost	80 g
Napajanje	$5,0$ V
Tok	6000 mA

Tabela 5.1: Specifikacije robotskega manipulatorja.

nekaj nadgradnje bi bilo sistem mogoče uporabiti tudi za izdelavo robotskega vozila.

Uporabo robotskih manipulatorjev prikazuje slika 5.2. Na poletni šoli FRI

2017 so se namreč osnovnošolci podučili o osnovah robotike in programirali robotske manipulatorje.



Slika 5.2: Primer zlaganja kock v stolp.

5.1 Nadaljnje delo

Prostora za nadgradnjo je še veliko. Poleg zaznave tokovne prekoračitve bi z računanjem naklona tokovne spremembe hitreje zaznali zunanje vplive na robotski manipulator ter tako povečali funkcionalnost in varnost ravnanja z robotskim manipulatorjem. To lahko uporabimo tudi za učenje premikanja robotskega manipulatorja, pri čemer z roko premikamo manipulator na želeno pozicijo. Zelo uporabna možnost je tudi krmiljenje z uporabo krivulj. Pri tem je pri računalniku treba omogočiti, da inverzna in direktna kinematika vračata ustrezne parametre, potrebne za krmiljenje s pomočjo krivulj, s čimer dosežemo bolj gladko in neprekinjeno premikanje manipulatorja. Z implementacijo stanj (pospešuje, se premika, zavira, drži) bi omogočili lažje spremljanje notranjega stanja krmilnega vezja. Za dodatno varnost lahko spremljamo temperaturo mikrokrmilnika in vezja ter v primeru pregrevanja zmanjšujemo moč elektromotorjev ali celo izklopimo krmiljenje.

Nadgraditi bi bilo primerno tudi shemo vezja. Če se vezje uporablja v predvidenih specifikacijah, ni težav, pojavijo pa se, ko se približujemo predvidenim mejam uporabe. Vezje nima vgrajenih prenapetostnih zaščit in zato se pri prekoračitvi meja mikrokrmilnik poškoduje ali celo v celoti uniči. Ker se tiskano vezje lahko uporablja kot samostojni modul, se lahko zgodi, da po nesreči priklopimo napajanje s previsoko napetostjo ali pri krmilnem delu naredimo kratek stik in tako nenamerno poškodujemo komponente na tiskanem vezju.

Zelo uporabna funkcionalnost bi bila dinamično računanje sil oziroma navora za posamezne servomotorje glede na lastnosti robotskega manipulatorja in trenutno pozicijo. S pomočjo takega dinamičnega modela lahko nato dinamično nastavljamo tokovne meje in dosežemo, da z roko poljubno in z malo sile premikamo robotski manipulator. To predstavlja močno orodje za ročno učenje robotskega manipulatorja.

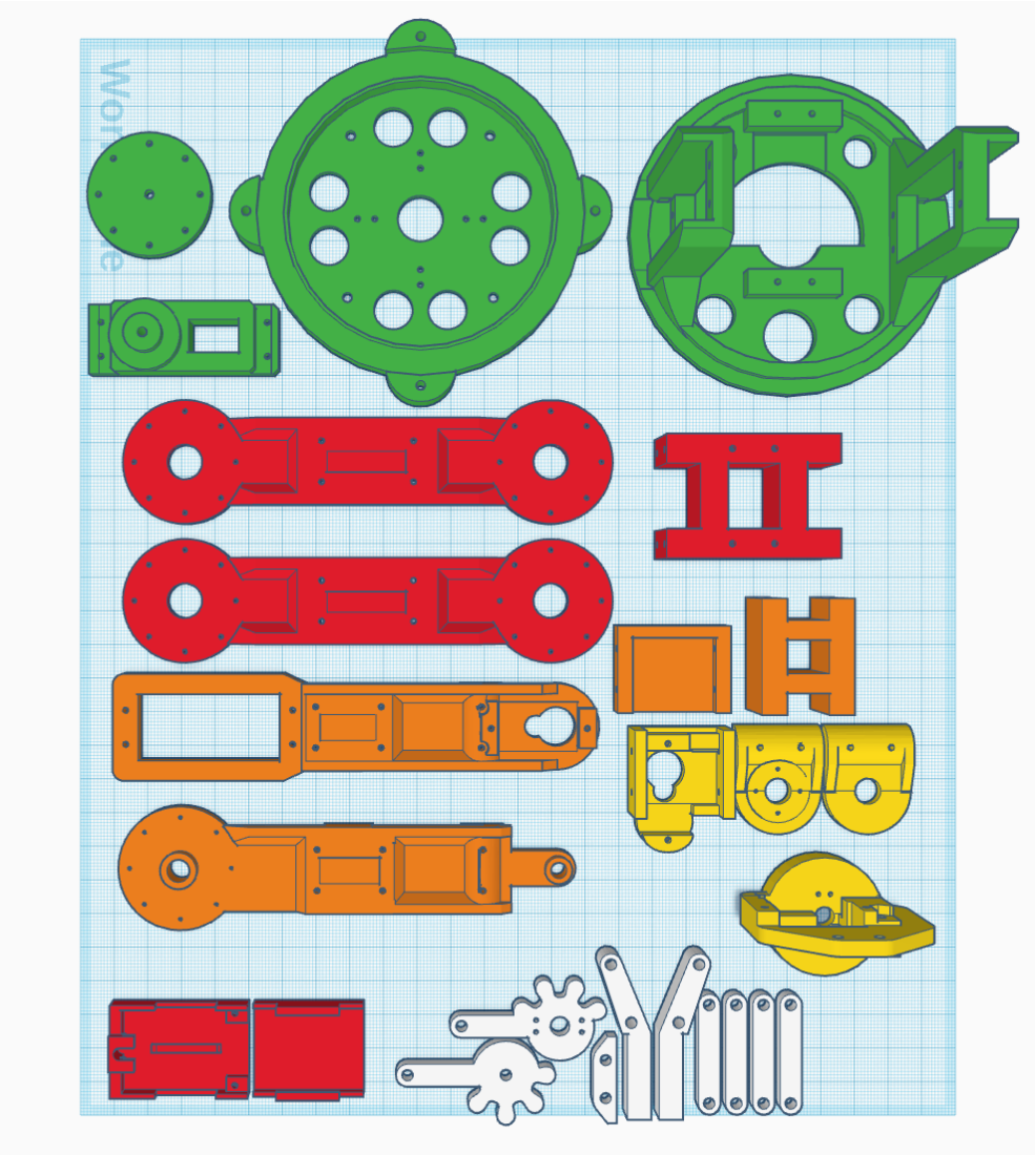
Razmišljamo o izdelavi novega modela roke, pri katerem bi izboljšali pomanjkljivosti trenutnega modela. Z vsemi napotki za izdelavo ga bomo tudi javno objavili.

Dodatek A

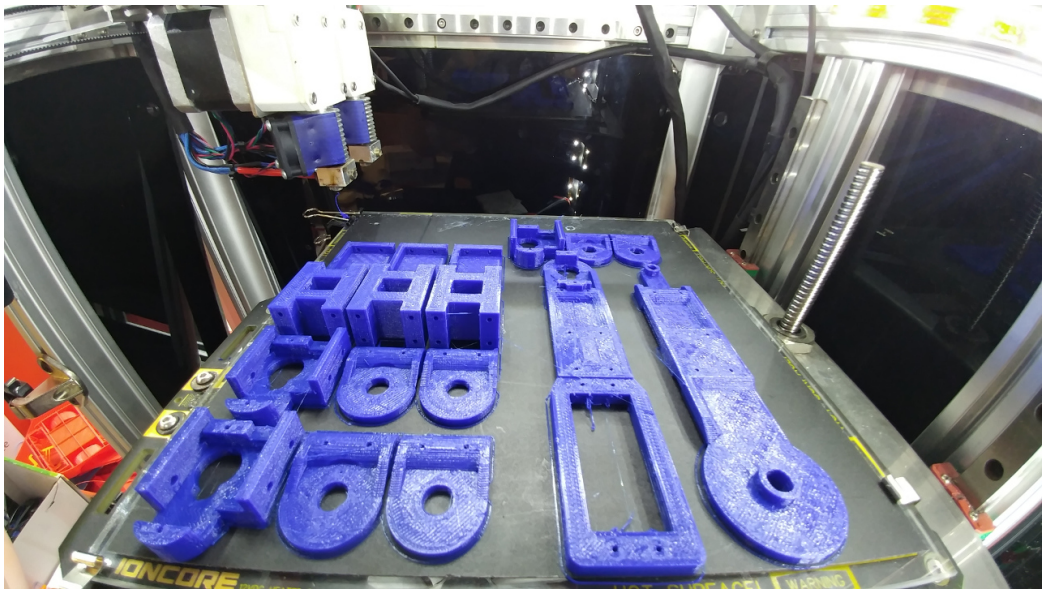
Ogrodje manipulatorja

Slika A.1 prikazuje vse dele, ki so potrebni za sestavo robotskega manipulatorja in jih je treba natisniti s 3D tiskalnikom. Na sliki pa ni prikazan namenski podstavek, v katerem je vgrajen Raspberry Pi z dodatni vezjem. Tiskanje s 3D tiskalnikom Zinter Pro [1] (slika A.2) pri srednji kakovosti traja približno 24 ur.

[1] http://www.zinter.com/



Slika A.1: Deli robotskega manipulatorja, pripravljeni za tiskanje s 3D tiskalnikom. Temno modre črte predstavljajo mrežo z razmikom 10 mm.



Slika A.2: Tiskanje delov za izdelavo robotskega manipulatorja.

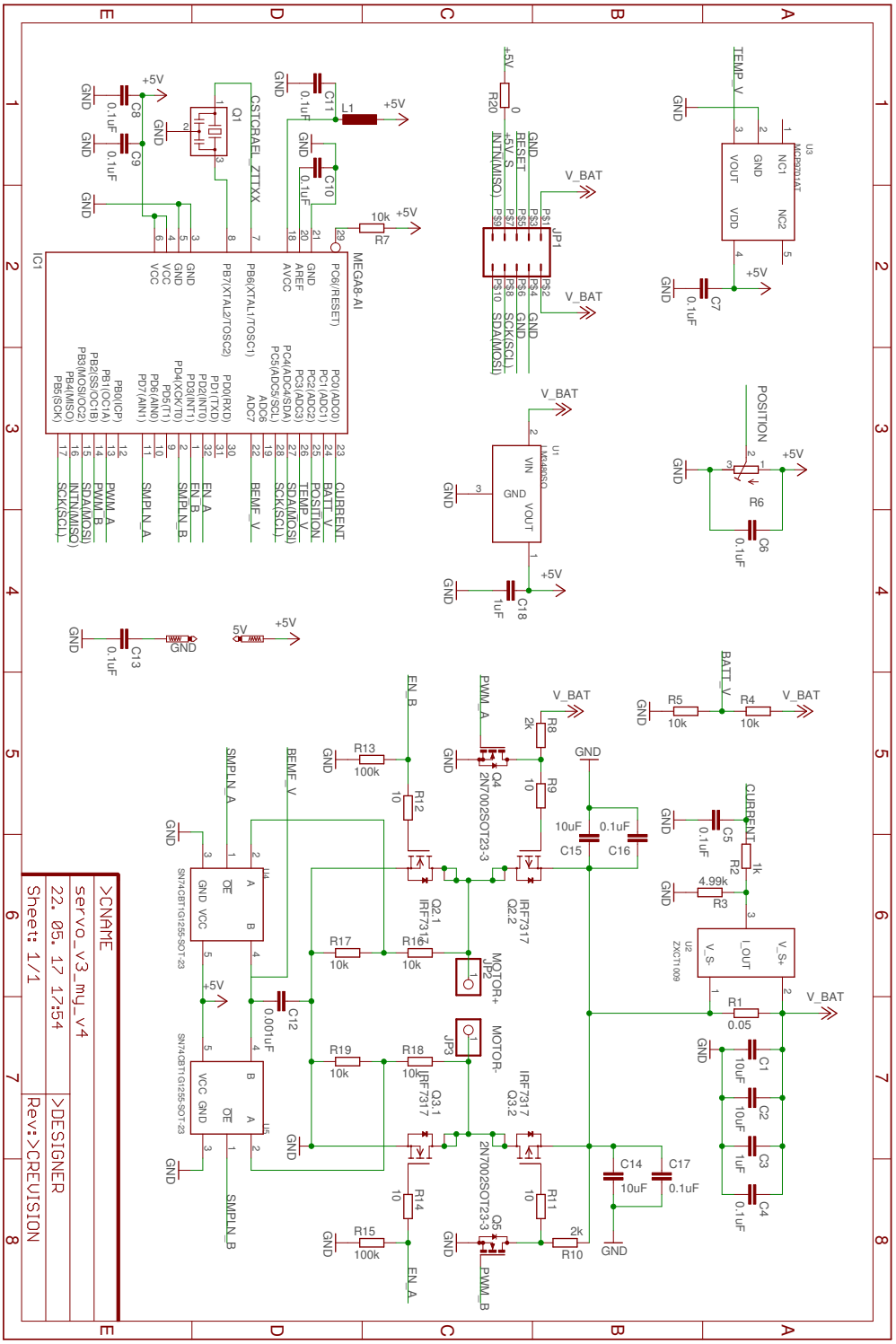
Dodatek B

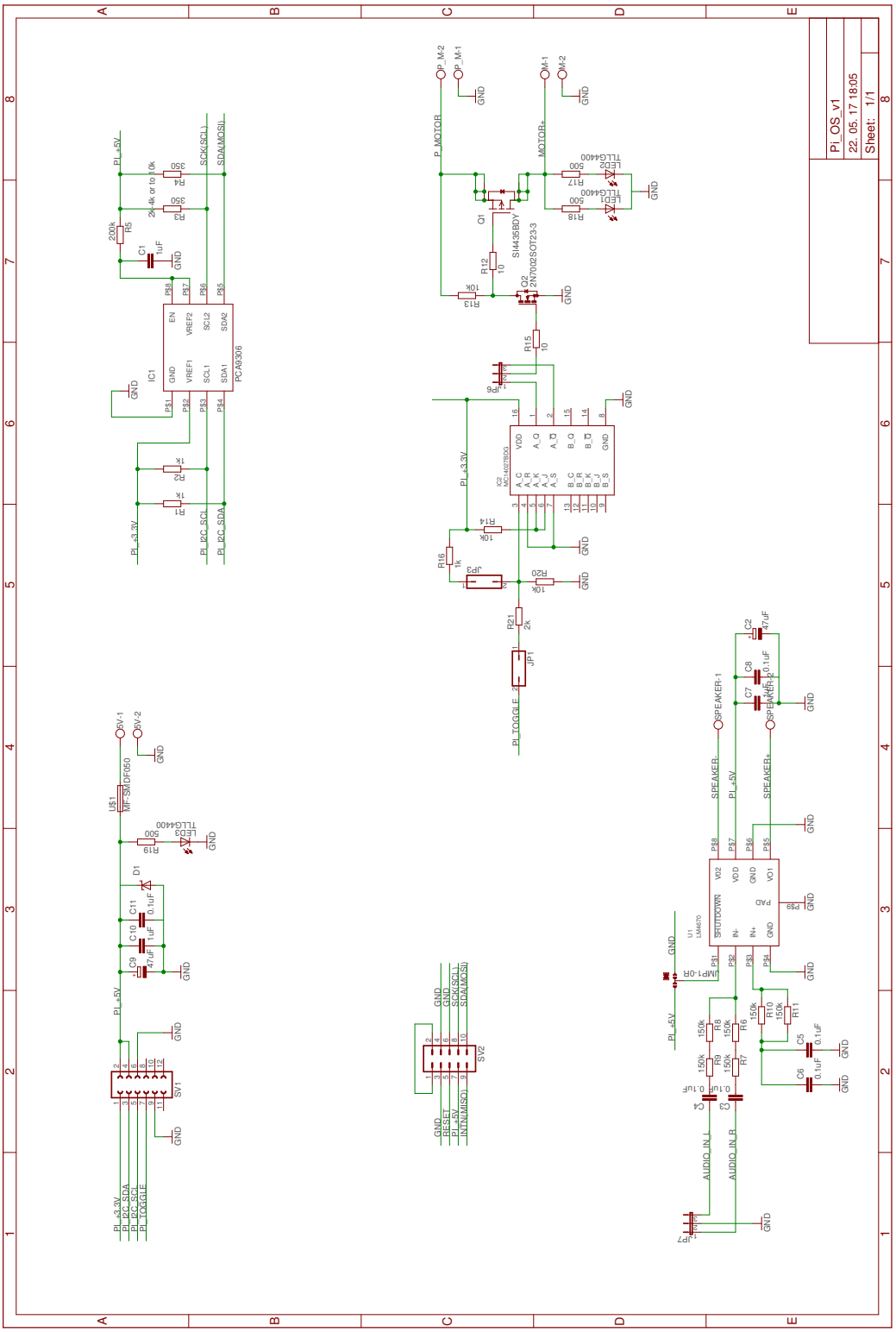
Sheme tiskanih vezij

Priložili smo shemo tiskanega vezja projekta OpenServo z vsemi opravljenimi spremembami. Sledi tudi shema vmesnega vezja, ki je namenjeno priklopu robotskega manipulatorja na Raspberry Pi 3. Obe shemi in tiskani vezji so bili narejeni z uporabo brezplačne različice programa Eagle [1] (različica 7,6). Tiskano vezje lahko damo v manjših količinah (na primer za domačo uporabo) izdelati tudi lokalnim posrednikom. [2]

[1] Dostopno na: `https://www.autodesk.com/products/eagle/overview`

[2] `https://www.svet-el.si/proizvodi-in-storitve/tiskana-vezja`


>CNAME
servo_v3_my_v4
22. 05. 17 17:54
>DESIGNER
Sheet: 1/1
Rev:>CREVISION
MEGA8-AI
IC1
CSTCRAEL_ZTTXX
Q1
L1
JP1
MOTOR+
MOTOR-
JP2
JP3
IRF7317
Q2.1
Q2.2
Q3.1
Q3.2
2N7002SOT23-3
Q4
Q5
SN74CBT1G125-SOT-23
U4
U5
U1
LM4040SO
U2
ZXCT1009
U3
MCP9701AT
TEMP_V
POSITION
CURRENT
BATT_V
V_BAT
+5V
GND
EN_A
EN_B
PWM_A
PWM_B
SMPLN_A
SMPLN_B
BEMF_V
SDA(MOSI)
SCK(SCL)
INTN(MISO)
RESET
R1 0.05
R2 1k
R3 4.99k
R4 10k
R5 10k
R6
R7 10k
R8 2k
R9 10
R10 2k
R11 10
R12 10
R13 100k
R14 10
R15 100k
R17 10k
R18 10k
R19 10k
R20 0
C1 10uF
C2 10uF
C3 1uF
C4 0.1uF
C5 0.1uF
C6 0.1uF
C7 0.1uF
C8 0.1uF
C9 0.1uF
C10 0.1uF
C11 0.1uF
C12 0.001uF
C13 0.1uF
C14 10uF
C15 10uF
C16 0.1uF
C17 0.1uF
C18 1uF


Pi_OS_v1
22. 05. 17 18:05
Sheet: 1/1
P_M-2
P_M-1
GND
P_MOTOR
M-1
M-2
MOTOR+
TLG4400
Si4435BDY
2N7002SOT23-3
PI_+5V
SCK(SCL)
SDA(MOSI)
200k
4k or to 10k
1uF
IC1
PCA9306
PI_I2C_SCL
PI_I2C_SDA
PI_+3.3V
BV-1
BV-2
MF-SMDFD50
PI_TOGGLE
SPEAKER-
SPEAKER+
LM4670
SHUTDOWN
PAD
AUDIO_IN_L
AUDIO_IN_R
RESET
INT(MISO)
SV1
SV2

Dodatek C

Kosovnica

Na naslednjih straneh je naveden popis komponent s cenami. Tabela C.1 navaja cene vseh posameznih delov, ki se uporabljajo za pripravo robotskega manipulatorja v učne namene. Za domačo uporabo bi lahko ceno izdelave še dodatno znižali. Namesto Raspberry Pi 3 se za krmiljenje lahko uporabi kar klasičen osebni računalnik ali prenosnik. Za napajanje krmilnih vezij in servomotorjev se lahko uporabi star računalniški napajalnik, pri izdelavi pa lahko uporabimo kar kable iz starega računalnika. V tem primeru se cena izdelave z 298 € zmanjša na približno 200 €.

Količina	Komponenta	Cena na enoto	Cena
6	OpenServo	14,47 €	86,82 €
1	Raspberry Pi OpenServo vmesno vezje	17,51 €	17,51 €
1	Material za sestavo robotskega manipulatorja	192,82 €	192,82 €
		Skupaj z DDV	297,15 €

Tabela C.1: Cena delov, potrebnih za izdelavo robotskega manipulatorja. Vse cene vsebujejo tudi DDV in veljajo na datum 22. 8. 2017.

OpenServo tiskano vezje					Cena z DDV	14,47 €			
Količina	Oznaka	Naziv artikla	Opis	Velikost	Cena na kos	Skupj	Proizvajalec	Dobavitelj	Kataloška
1	IC1	ATMEGA328P-AU	8 Bit Microcontroller, Low Power High Performance, ATmega, 20 MHz, 32 KB, 2 KB, 32 Pins, TQFP	TQFP	3,2	3,2	Microchip	Farnell	1715486
11	C4, C5, C6, C7, C8, C9, C10, C11, C13, C16, C17	MCB0603R104KCT	smd ceramic capacitor 0.1uF, 16V	V 0603	0,0078	0,0858	Multicomp	Farnell	9406140
2	R8, R10	MC0063W060312K	smd, resistro, 2k, 1%	V 0603	0,0037	0,0074	Multicomp	Farnell	9330763
1	U2	ZXCT1009F	current sensor, high side	V 0603	0,876	0,876	Diodes inc.	Farnell	1132757
1	R2	MC0063W060311K	smd resistor 1k 1%, 0603	V 0603	0,0075	0,0075	Multicomp	Farnell	9330380
2	R13, R15	MC0063W06031100K	smd resistor, 100k 1%	V 0603	0,002	0,004	Multicomp	Farnell	9330402
1	R1	MCCSM06FTDUR022	smd current sense resistor, 0.022Ohm +-1%	V 1206	0,432	0,432	Multicomp	Farnell	2294085
2	U4, U5	SN74CBT1G125DBVT	logic, switch bus fet sgl	SOT-23-5	0,324	0,648	Texas Instruments	Farnell	1741495
2	C3, C18	C1608X7R1C105K080AC	smd ceramic capacitor 1uF, 16V +-10%, X7R	V 0603	0,0588	0,1176	TDK	Farnell	1907343
4	C1, C2, C14, C15	C3216X7R1C106K160AC	smd ceramic capacitor 10uF, 16V +-10%, X7R	V 1206	0,3	1,2	TDK	Farnell	1907353
1	C12	MCSH18B102K160CT	smd ceramic capacitor 1000pF, 16V +-10%, X7R	V 0603	0,0425	0,0425	Multicomp	Farnell	1856321
1	U3	MCP9701AT-E/LT	temperature sensor IC	SC-70, 5pins	0,387	0,387	Microchip	Farnell	1627203
1	Q1	CSTCE20M0V53	resonator, ceramic, 20MHz, smd		0,432	0,432	MURATA	Farnell	2443266
1	L1	MLF1608E100K	inductor, high frequency, 10uH +-10	V 0603	0,211	0,211	TDK	Farnell	1669541
2	Q2, Q3	IRF7317PBF	dual MOSFET, N and P channel, 6.6A, 20V	SOIC	0,835	1,67	INFINEON	Farnell	9102159
2	Q4, Q5	NX7002BK	N Channel, 270 mA, 60 V, 2.2 ohm, 10 V, 1.6 V	SOT-23	0,0414	0,0828	NEXPERIA	Farnell	2469640
7	R4. R5, R7, R16, R17, R18, R19	MC0063W0603110K	10k 1%	V 0603	0,0036	0,0252	Multicomp	Farnell	9330399
4	R9, R11, R12, R14	MC0063W0603110R	10 O 1%	V 0603	0,0036	0,0144	Multicomp	Farnell	9330429
1	R3	MC0063W060317K5	7.5k 1%	V 0603	0,0012	0,0012	Multicomp	Farnell	9331514
0,125		Letvica moška	2x40 pinov		0,3361	0,0420125	Amphenol	Ic-Elektronika	410112360100
1		Konektor	konektor speedy 6 ženski		0,2705	0,2705		Ic-Elektronika	400010060101
1		Tiskano vezje			2,1	2,1		Svet elektronike	

Raspberry Pi OpenServo vmesno vezje				Skupaj z DDV:	17,51 €			
Količina	Oznaka	Naziv	Cena kos	Cena	Proizvajalec	Dobavitelj	Kataloška	
1	D1	LITTELFUSE SMBJ5.0A TVS Diode, Transil SMBJ Series, Unidirectional, 5 V	0,323	0,323	LITTELFUSE	Farnell	1456972	
1	IC1	ON SEMICONDUCTOR PCA9306DTR2G Voltage Level Translator, Bidirectio	0,724	0,724	ON SEMICONDUCTOR	Farnell	2464616	
1	U1	TEXAS INSTRUMENTS LM4670SD Audio Power Amplifier, D, 1 Channel, 3 W	1,02	1,02	TEXAS INSTRUMENTS	Farnell	1312668	
2	C2, C9	PANASONIC ELECTRONIC COMPONENTS EEEFPJ470UAR SMD Aluminiu	0,319	0,638	PANASONIC ELECTRON	Farnell	1539448	
6	C3, C4, C5, C6, C8, C11	KEMET C0805C104K4RACTU SMD Multilayer Ceramic Capacitor, 0805 [201	0,061	0,366	KEMET	Farnell	2429355	
3	C1, C7, C10	KEMET C0805C105K4RACTU SMD Multilayer Ceramic Capacitor, 0805 [201	0,067	0,201	KEMET	Farnell	9227792	
1	U$1	BOURNS MF-MSMF150-2 PPTC Resettable Fuse, SMD, MF-MSMF Series,	0,324	0,324	BOURNS	Farnell	9350357	
1		PRO SIGNAL ABS-215-RC Speaker, Mini, Mylar Cone, 1.5 W, 8 ohm, 550 Hz	2,07	2,07	PRO SIGNAL	Farnell	1300023	
1	Q1	VISHAY SI4477DY-T1-GE3 MOSFET Transistor, P Channel, -26.6 A, -20 V, 0	1,3	1,3	VISHAY	Farnell	1852575	
1	Q2	FAIRCHILD SEMICONDUCTOR 2N7002 MOSFET Transistor, N Channel, 11	0,186	0,186	FAIRCHILD SEMICONDU	Farnell	9845313	
1	IC2	ON SEMICONDUCTOR MC14027BDG Flip-Flop, Master/Slave,Complement	0,422	0,422	ON SEMICONDUCTOR	Farnell	9666354	
1	SV1	LETVICA 2x6 Ž PROF.VISOKA R=2,54	0,46	0,46	Neltron	IC-Elektronika	4102120151001	
4	SV, SPEAKER, P_M, M	VRSTNA SPONKA ZA TIV 90° 2P - NIZKA (ZELENA)	0,35	1,4	TE Connectivity	IC-Elektronika	2610103401001	
2	R12, R15	UPOR 10R 0603 1% SMD	0,025	0,05	Vishay	IC-Elektronika	2096110001021	
2	R3, R4	UPOR 340R 0603 1% SMD	0,025	0,05	Yageo	IC-Elektronika	2096234001001	
3	R17, R18, R19	UPOR 511R 0603 1% SMD	0,025	0,075	Yageo	IC-Elektronika	2096251101001	
3	R16	UPOR 1K 0603 1% SMD	0,025	0,075	Vishay	IC-Elektronika	2096310001011	
1	R21	UPOR 2K 0603 1% SMD 7''	0,025	0,025	Vishay	IC-Elektronika	2096320001021	
3	R13, R14, R20	UPOR 10K 0603 1% SMD	0,025	0,075	Vishay	IC-Elektronika	2096410001011	
6	R5, R6, R7, R8, R9, R10	UPOR 150K 0603 1% SMD	0,025	0,15	Vishay	IC-Elektronika	2096515001011	
0,125	SV2	LETVICA 2X40 M R=2,54	0,3361	0,042012	5 Neltron	IC-Elektronika	4101123601001	
1		Tiskano vezje	4,38	4,38		Svet elektronike		

Material za sestavo robotskega manipulatorja		Cena z DDV	192,82 €			
Količina	Naziv	Cena na kos	Cena	Proizvajalec	Dobavitelj	Kataloška
1	HS-645MG	28,28	28,28	HITEC	Mibo Modeli	H112645
1	HS-485HB	16,4	16,4	HITEC	Mibo Modeli	H112485
1	HS-311	8,94	8,94	HITEC	Mibo Modeli	H112311
3	ES08A	5,5	16,5	EMAX	Mibo Modeli	E-ES08A
0,02	Vgrezni vijaki za pločevino 2.2 mm 6.5 mm križni Philips DIN 7982 nerjaveče jeklo A2	13,08	0,2616	TOOLCRAFT	Conrad	1068218
0	100-DELNI SET PLOČEVINSKIH VIJJAKOV Z LEČASTO GLAVO DIN7981C2,2X6,5	1,88	0	TOOLCRAFT	Conrad	827354
0,032	Vgrezni vijaki za pločevino 2.2 mm 13 mm križni Philips DIN 7982 nerjaveče jeklo A2	15,54	0,49728	TOOLCRAFT	Conrad	1068220
0	Cilindrični vijak, M2, 25mm, zareza, DIN 84, pocinkano jeklo, 1 kos	0,123	0	TOOLCRAFT	Conrad	888677
0,01	PAN vijak TOOLCRAFT, križni, M3, 25mm, DIN 7985, pocinkano jeklo, 100 kosov	2,37	0,0237	TOOLCRAFT	Conrad	827136
0,07	PAN vijak TOOLCRAFT, križni, M3, 20mm, DIN 7985, pocinkano jeklo, 100 kosov	2,95	0,2065	TOOLCRAFT	Conrad	815411
0	PAN vijak TOOLCRAFT, križni, M2, 20mm, DIN 7985, pocinkano jeklo, 100 kosov	3,52	0	TOOLCRAFT	Conrad	830422
0	Šesterokotna matica TOOLCRAFT, M2, DIN 934, medenina, 100 kosov	1,96	0	TOOLCRAFT	Conrad	812819
0,14	Šestroba matica M3 DIN 934 jeklo, galvansko pocinkano 100 kosov TOOLCRAFT 13	0,82	0,1148	TOOLCRAFT	Conrad	131823
0	Pločevinasti vijaki z lečasto glavo 2.9 mm 16 mm križni Pozidriv ISO 7049 jeklo g	2,46	0	TOOLCRAFT	Conrad	147588
0	Podložka notranji premer: 2.2 mm DIN 433 jeklo 500 kosov TOOLCRAFT 106677	2,78	0	TOOLCRAFT	Conrad	106677
0,24	Podložka notranji premer: 3.2 mm DIN 433 jeklo, galvansko pocinkano 500 kosov	2,78	0,6672	TOOLCRAFT	Conrad	106698
1	Vtični napajalnik, stalna napetost Mean Well SGA60E05-P1J 5 V/DC 6000 mA 30 W	30,29	30,29	Mean Well	Conrad	1439210
1	SCI Stikalo s prevesno ročico,6 A R13-112LP-02 LED GREEN250V/AC 6 A	3,36	3,36	SCI	Conrad	700320
1	Nizkonapetostni konektor, preklopni kontakt: odprta vtičnica, pokončna vgradnja 5.7	1,96	1,96	BKL	Conrad	1460763
0,3	Filament PrimaValue PLA 1kg	22,13	6,639	PRIMA	iTehLab	
1,1	FLAT KABEL 10 X 0,14 AWG28 105°C	0,42	0,462	Neltron	IC Elektronika	358000100200
1	KABEL 2x0,75 ČRNO-RDEČ ZA ZVOČNIKE	0,43	0,43	Velleman	IC Elektronika	357010000100
1	RASPBERRYPI3-MODB-1GB	32,79	32,79	RASPBERRY-PI	Farnell	2525225
1	SanDisk SDHC 16GB Ultra UHS-I 80MB/s	7,41	7,41	SanDisk	Nakupovanje	44625
1	Aksialni ležaj 51111	2,82	2,82	Codex	Codex	51111

Literatura

[1] M. Quigley, A. Asbeck, A. Ng, A low-cost compliant 7-dof robotic manipulator, 2011 IEEE International Conference on Robotics and Automation (May 2011) pp. 6051–6058.

[2] J. A. N. Cocota, H. S. Fujita, I. J. da Silva, A low-cost robot manipulator for education, 2012 Technologies Applied to Electronics Teaching (TAEE).

[3] S. Karmoker, M. M. H. Polash, K. M. Z. Hossan, Design of a low cost pc interface six dof robotic arm utilizing recycled materials, 2014 International Conference on Electrical Engineering and Information and Communication Technology.

[4] B. Pronadeep, N. Vishwajit, Low cost shadow function based articulated robotic arm, 2015 International Conference on Energy, Power and Environment: Towards Sustainable Growth (ICEPE).

[5] M. R. S. Pol, M. S. Giri, M. A. Ravishankar, M. V. Ghode, Labview based four dof robotic arm, 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (Sep. 2016) pp. 1791–1798.

[6] H. Zhang, J. Gonzalez-Gomez, Z. Xie, S. Cheng, J. Zhang, Development of a low-cost flexible modular robot gz-i, IEEE/ASME International Conference on Advanced Intelligent Mechatronics vol. (no.) (July 2008) pp. 223–228.

[7] Z. Fang, Y. Fu, T. Chai, A low-cost modular robot for research and education of control systems, mechatronics and robotics, IEEE vol. (no.) (2009) pp. 2828–2833.

[8] T. Bajd, Osnove robotike, Univerza v Ljubljani, Fakulteta za elektrotehniko, 2006.

[9] R. S. H. J. Denevit, A Kinematic Notation for Lower-Pair Mechanisem Based on Matrices, Journal of Applied Mechanics (1955) pp. 215–221.

[10] A. C. Gómez, Diseño y puesta en funcionamineto de un brazo robótico imprimible, Universidad Carlos III de Madrid, 2012.

[11] B. Carter, OpenServo, accessed: 2016-12-3 (2008).
URL http://www.openservo.com/

[12] ATMEL, Atmel 8-bit AVR Microcontroller with 2/4/8K Bytes In-System Programmable Flash, accessed: 2017-12-7 (2013).
URL http://www.atmel.com/images/atmel-2586-avr-8-bit-microcontroller-attiny25-a
datasheet.pdf

[13] ATMEL, 8-bit Atmel with 8KBytes In-System Programmable Flash, ATmega8, ATmega8L, accessed: 2017-12-7 (2013).
URL http://www.atmel.com/Images/Atmel-2486-8-bit-AVR-microcontroller-ATmega8_
L_datasheet.pdf

[14] ATMEL, 8-bit Microcontroller with 16K Bytes In-System Programmable Flash, ATmega16, ATmega16L, accessed: 2017-12-7 (2010).
URL http://www.atmel.com/images/doc2466.pdf

[15] NXP, I^{2}C-bus specification and user manual, accessed: 2017-12-7 (2014).
URL http://www.nxp.com/docs/en/user-guide/UM10204.pdf

[16] ATMEL, ATmega328/P DATASHEET COMPLETE, accessed: 2017-12-7 (2016).
URL http://ww1.microchip.com/downloads/en/DeviceDoc/

Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf

[17] ATMEL, AVR Instruction Set Manual, accessed: 2017-12-7 (2016).
URL http://www.atmel.com/images/Atmel-0856-AVR-Instruction-Set-Manual.pdf

[18] ATMEL, AVR121: Enhancing ADC resolution by oversampling, accessed: 2017-12-7 (2005).
URL http://www.atmel.com/images/doc8003.pdf

[19] A. Blad, P. Löwenborg, H. Johansson, Design trade-offs for linear-phase FIR decimation filters and $\Sigma\Delta$-modulators, in: 2006 14th European Signal Processing Conference, 2006, pp. 1–5.

[20] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, ROS: an open-source Robot Operating System, in: ICRA Work. Open Source Softw., 2009.

[21] P. Beeson, B. Ames, Trac-ik: An open-source library for improved solving of generic inverse kinematics, in: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), 2015, pp. 928–935. doi:10.1109/HUMANOIDS.2015.7363472.

[22] D. K. Prasanga, T. Mizoguchi, K. Tanida, K. Ohnishi, Compensation of backlash for teleoperated geared motor drive systems, in: IECON 2013–39th Annual Conference of the IEEE Industrial Electronics Society, 2013, pp. 4067–4072. doi:10.1109/IECON.2013.6699787.