Otvaranje ladice robotskom rukom pomoću računalnog vida i upravljanja silom

Dukić, Jana

Master's thesis / Diplomski rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:672671

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I
INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Diplomski sveučilišni studij Računarstvo
Izborni blok Robotika i umjetna inteligencija

Otvaranje ladice robotskom rukom pomoću računalnog vida i upravljanja silom

Diplomski rad

Jana Dukić

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 10.07.2023.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	Jana Dukić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	D-1198R, 05.10.2021.
OIB studenta:	47967888925
Mentor:	prof. dr. sc. Robert Cupec
Sumentor:	doc. dr. sc. Petra Pejić
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	izv. prof. dr. sc. Emmanuel-Karlo Nyarko
Član Povjerenstva 1:	prof. dr. sc. Robert Cupec
Član Povjerenstva 2:	Valentin Šimundić, mag. ing. comp.
Naslov diplomskog rada:	Otvaranje ladice robotskom rukom pomoću računalnog vida i upravljanja silom
Znanstvena grana diplomskog rada:	**Umjetna inteligencija (zn. polje računarstvo)**
Zadatak diplomskog rada:	Izraditi računalni program za sustav koji se sastoji od robotske ruke s ugrađenom 3D kamerom i senzorom sile, koji omogućava otvaranje ladica. Na temelju snimke čovjeka kako otvara ladicu, program treba odrediti veličinu, položaj i smjer otvaranja ladice te na temelju tih podataka isplanirati trajektoriju za robotsku ruku. Nakon toga, robot treba otvoriti i zatvoriti ladicu koristeći senzor sile radi korekcije planirane trajektorije na način da se minimiziraju sile kojima robot djeluje na ladičar. Tema rezervirana za: Lukrecia Vulić ili Jana Dukić Sumentor s FERIT-a: Petra Pejić
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	10.07.2023.

Potvrda mentora o predaji konačne verzije rada:	*Mentor elektronički potpisao predaju konačne verzije.*
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 19.07.2023.

Ime i prezime studenta:	Jana Dukić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1198R, 05.10.2021.
Turnitin podudaranje [%]:	6

Ovom izjavom izjavljujem da je rad pod nazivom**: Otvaranje ladice robotskom rukom pomoću računalnog vida i upravljanja silom**

izrađen pod vodstvom mentora prof. dr. sc. Robert Cupec

i sumentora doc. dr. sc. Petra Pejić

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD

Robotika kao znanstveno inženjerska grana svakodnevno se ubrzano razvija. Osnovna motivacija za razvoj robotike leži u želji da se čovjeku olakšaju svakodnevni zadaci. Prva upotreba robota javila se u industriji primjenom industrijskih alata koji su obavljali jednostavne zadatke u tvornicama, koji su bili zamorni ili opasni za ljude. S vremenom, robotika se počela primjenjivati i u kućanstvima pa su tako izumljeni prvi roboti usisavači i roboti za miješanje tijesta, a sve s ciljem olakšavanja svakodnevice krajnjim korisnicima.

Danas se od robota očekuje visoka razina autonomnosti koja podrazumijeva samostalno djelovanje u nestrukturiranim okolinama. Jedan od takvih primjera, koji se razmatra i u ovom diplomskom radu, je robotski manipulator sposoban za samostalno otvaranje i zatvaranje ladica. Takav robot mogao bi pomoći u dohvaćanju predmeta ili njihovom pospremanju u ladice u kućanstvima i institucijama poput bolnica i škola.

Kako bi robot mogao otvoriti ili zatvoriti ladicu, mora je detektirati u svojoj okolini, procijeniti njezinu veličinu, detektirati i lokalizirati ručku ladice u 3D prostoru i pomaknuti se u položaj prihvatljiv za hvatanje ručke. Nadalje, robotski sustav mora izračunati trajektoriju otvaranja ladice te tijekom provođenja akcije otvaranja ili zatvaranja po izračunatoj trajektoriji potrebno je regulirati silu kojom robot povlači ili gura ladicu. Djelovanje nekontroliranom silom može uzrokovati neželjena oštećenja, bilo na samom manipulatoru, ladici ili okolini. Moguće je kombinirano upravljanje po sili i poziciji, na način da se u smjeru otvaranja i zatvaranja ladice upravlja po poziciji, dok se u lateralnim smjerovima minimizira sila na vrijednost jednaku nuli.

U ovom radu, predstavljena su rješenja za probleme detekcije i izgradnje modela ladice, izračuna međusobnih položaja dijelova robotskog sustava i ladice predstavljenih transformacijskim matricama te upravljanja silom prilikom otvaranja i zatvaranja ladica robotskom rukom. Korištene su metode računalnog vida iz programske biblioteke RVL koje provode izgradnju modela ladice na temelju sekvence dubinskih slika u boji koje prikazuju ljudsku demonstraciju otvaranja vrata. Za regulaciju sile tijekom robotske manipulacije, koristi se ugrađena funkcija senzora sile i momenta. Rješenja su implementirana i ispitana na robotskom sustavu kojeg čini *Universal Robots* UR5 6-osni robotski manipulator s troprstom hvataljkom *Robotiq 3-Finger Gripper*, senzorom sile i momenta *Robotiq FT-300S* i dubinskom kamerom *Intel RealSense LiDAR L515*. Konačni je rezultat ovog diplomskog rada program za otvaranje i zatvaranje ladice robotskom rukom.

Rad je strukturiran kako slijedi. U drugom poglavlju dan je pregled postojećih rješenja za problem otvaranja i zatvaranja ladica s mobilnim i fiksnim robotskim manipulatorima. Trećim poglavljem prikazan je matematički opis odnosa koordinatnih sustava robotskog sustava i ladice te opis metoda za detekciju i izgradnju modela i upravljanje po sili i poziciji. U četvrtom poglavlju opisane su tehnologije i razvojni alati korišteni za razvoj i implementaciju predloženih rješenja. U petom poglavlju opisana je implementacija algoritama računalnog vida, algoritama za računanje matrica homogene transformacije i implementacija upravljanja silom i pozicijom robotskog manipulatora. Šesto poglavlje sastoji se od opisa provedenih pokusa i analize njihovih rezultata. Zaključak je dan u sedmom poglavlju.

2. PREGLED PODRUČJA AUTOMATIZIRANOG OTVARANJA I ZATVARANJA LADICA

U ovom poglavlju nalazi se pregled literature iz područja automatiziranog otvaranja i zatvaranja ladica. Prikazana su postojeća rješenja te njihovi rezultati.

Jain i drugi u radu *Pulling open doors and drawers: Coordinating an omni-directional base and a compliant arm with Equilibrium Point control* [1], predložili su implementaciju upravljanja impedancijom inspiriranu hipotezom o točki ravnoteže koju nazivaju *kontrolna točka ravnoteže* (engl. *Equilibrium Point Control*, EPC). Pokazali su da EPC može omogućiti robusno povlačenje različitih ladica i vrata u fiksnom položaju robota te zaključiti o njihovoj kinematici bez detaljnih prethodnih modela. Ovakvim rješenjem robot autonomno otvara vrata i ladice zadavanjem položaja ručke u grafičkom sučelju. Manipulacija s niskom impedancijom smanjuje sile i momente koje nastaju kao posljedica kontakta i tako smanjuje rizik od oštećenja robota, ljudi u blizini i okoline. EPC se koristi za koordinaciju pokretne robotske baze i ruke. Robotskom rukom upravlja se podešavanjem položaja kontrolne točke u Kartezijevom prostoru koja označava gdje bi se krajnji alat robota zaustavio u odsutnosti vanjskih sila izuzev gravitacije. Spomenuto su omogućili korištenjem virtualne viskoelastične opruge na zglobovima robota zajedno s kompenzacijom gravitacije, gdje robot može samostalno prilagoditi krutost tih virtualnih opruga. Na taj način ne modeliraju izričito dinamiku ruke ni impedanciju krajnjeg alata te ne koriste inverznu dinamiku. Zbog iznesenog navode kako je EPC jednostavan za korištenje. Temeljem demonstracije pokazuju da robot uspješno otvara različita vrata i ladice u 37 od 40 pokusa. U pokusima, robotu se daje informacija o lokaciji i orijentaciji ručke te, dok povlače ručku, određuju kinematiku mehanizma kako bi prilagodili kretnje robotske baze i ruke. Pokretna baza robota omogućava poboljšane performanse simultanog otvaranja. Međutim, u ovom diplomskom radu razvija se metoda koja je pogodna za otvaranje ladica bez korištenja grafičkog sučelja. Razvijeni sustav, na temelju modela dobivenog iz sekvence otvaranja ladice, računa lokaciju ručke za specifičnu ladicu.

Kim i drugi u radu *Operating an unknown drawer using an aerial manipulator* [2], predložili su rješenje za otvaranje i zatvaranje nepoznatih ladica korištenjem bespilotnih letjelica. Korištena bespilotna letjelica je multirotor u kombinaciji s robotskim manipulatorom s 3 stupnja slobode. Također, na vrhu manipulatora nalazi se kuka koju bespilotna letjelica zakači za ručku ladice. Za razliku od manipulatora koji otvaraju ladice dok su pričvršćeni za zemlju, bespilotni manipulatori mogu vrlo lako izgubiti ravnotežu. Iz tog razloga je osiguranje njegove stabilizacije osnovni fokus. Kod ovakvih manipulatora senzori sile i momenta ne pružaju dovoljno korisnih informacija jer manipulatori u zraku lako podliježu utjecaju vanjskih sila. Stoga se radi modeliranje dinamičkih karakteristika bespilotnog manipulatora i analizira interakcija s ladicom. Računa se i konfiguracija zračnog manipulatora koja je potrebna za izvršavanje željene sile koja se primjenjuje na ladicu. Kako bi se riješile nesigurnosti povezane s mehanizmom ladice, koristi se strategija prilagodbe brzine krajnjeg alata manipulatora. Iako u radu nisu navedeni detaljni rezultati provedenih pokusa, autori ističu uspješnu regulaciju translacijske i kutne brzine, omogućavajući uspješno otvaranje i zatvaranje ladice. Za razliku od predložene bespilotne letjelice, u ovom radu se ladica otvara korištenjem fiksiranog robotskog manipulatora upravljanjem po sili i poziciji.

Rühr i drugi u radu *A generalized framework for opening doors and drawers in kitchen environments* [3], predstavljaju rješenje za robusno manipuliranje nepoznatim ormarićima unutar kuhinje. Njihovo rješenje sastoji se od 4 dijela: detekcije ručki na temelju oblaka točaka, otvaranja/zatvaranja ormarića pomoću upravljanja impedancijom i učenja njihovih kinematičkih modela, spremanja i dohvaćanja informacija o objektima u prostornoj mapi te pouzdanog rukovanja ormarićima za koje je kinematički model poznat. Prvi dio detekcije ručki

odnosi se na prikupljanje informacija iz 3D oblaka točaka. Točke koje pripadaju ručkama se dobivaju izvlačenjem točaka do zadane udaljenosti od segmentirane ravnine vrata. Ako su ručke reflektivne, njihove točke se ne vide na oblaku točaka. Autori iskorištavaju spomenutu činjenicu na način da traže ručke u područjima gdje nedostaju vrijednosti. Demonstriraju rukovanje vratima i ladicama kombinacijom upravljanja impedancijom i učenjem kinematičkog modela. Koriste KnowRob (engl. *Knowledge Processing for Autonomous Personal Robots*) [4] sustav za obradu podataka uz pomoć kojeg se odvija spremanje artikuliranih modela i položaja ručke kao dio semantičke mape robotske okoline. Prilikom manipulacije objektom, omogućava se određivanje kinematičkog modela pokretnog dijela objekta korištenjem kontrolne točke ravnoteže i zabilježavanjem trajektorije. Eksperimentalna analiza je pokazala kako su dva PR2 robota uspješno rukovala u 51.9% od ukupno 104 slučajeva otvaranja kuhinjskih elemenata. Kao obrazloženje neuspješnog manipuliranja ormarićima i ladicama, navode kako su u velikom broju slučajeva roboti bili preslabi za rukovanje objektima. U ovom radu, naglasak je stavljen na učenje kinematičkog modela objekta iz snimljene sekvence otvaranja ladice te upravljanja po poziciji i sili kako bi se osiguralo uspješno rukovanje.

Y. Karayiannidis i drugi, u radu *Model-free robot manipulation of doors and drawers by means of fixed-grasps* [5], predstavili su metodu za manipulaciju objektima s prizmatičnim i rotacijskim zglobovima koja ne zahtijeva prethodno znanje kinematike objekta. Metoda se sastoji od upravljanja brzinom manipulatora koja se oslanja na mjerenja dobivena senzorom sile i momenta te estimacije smjera otvaranja, rotacijske osi i udaljenosti od centra rotacije. Tvrde kako je njihova metoda prikladna za bilo koji brzinom upravljani manipulator s ugrađenim senzorom sile i momenta na kraju vrha alata. Upravljanje silom i momentom regulira primijenjenu silu i moment u zadanim ograničenjima, dok regulator brzine osigurava kretnje vrha alata u skladu s tangencijalnom brzinom pojedinog zadatka koji robot obavlja. Navode kako dizajn cjelokupne metode osigurava konvergenciju estimiranih vrijednosti prema stvarnim vrijednostima. Predstavljana metoda implementirana je na robotsku platformu s upravljanjem brzinom čiji je cilj bio otvaranje ormarića. Rezultat pokusa uspoređivan je sa simuliranom vrijednosti. U stvarnom pokusu, za početnu estimaciju slobodnog smjera, metoda je pogriješila oko 30°, što je rezultiralo manjom privremenom pogreškom koja se ispravila te je robot uspješno otvorio vrata. Za razliku od opisanog rada, u ovom diplomskom radu nije potrebno zadavati hvat robota na ručki, već se generira kinematički model ladice iz vizije.

Prats i drugi u članku *Compliant interaction in household environments by the Armar-III humanoid robot* [6], predstavljaju humanoidnog robota koji može rukovati namještajem koji se obično pronalazi u kućanskim prostorima. Na robotovom zglobu nalazi se senzor sile i momenta, koji upravlja pokretima robota bez prethodnog znanja o mehanizmu koji se nastoji otvoriti. Spomenuti robot ima 8 stupnjeva slobode koji omogućavaju redundanciju zadatka. Redundancija zadatka omogućava povećanje prostora sekundarnog kretanja, što omogućuje zadržavanje položaja ruke u udobnom položaju tijekom obavljanja glavnog zadatka. Njihov pristup se temelji na *Task Frame Formalism*-u kojeg je osmislio M.T. Mason [7], zbog njegove prikladnosti za sve vrste radnji koje su upravljane silom. *Task frame* definiran je kao kartezijski koordinatni sustav zadan u koordinatama objekta, gdje se zadatak definira u smislu referenci brzine i sile, prema prirodnim ograničenjima koje nameće mehanizam koji se otvara. Korišteni robot je bio u interakciji s četiri različita elementa koji se mogu pronaći u kuhinji: hladnjak, ladica, perilica i ormarić. Svaki od navedenih mehanizama je različit te prethodno nepoznat robotu. U zaključku navode kako robot u stvarnom vremenu prilagođava svoje pokrete za pojedini zadatak te uspješno otvara navedene elemente. Kao manu ovog pristupa, navode kako procjena radnog okvira sadrži pogreške, što može rezultirati pogreškama u trajektoriji gdje se generiraju vrlo velike sile na robotsku ruku te može doći do oštećenja ako se robot ne zaustavi na vrijeme. Spomenuto rješavaju lokalnom modifikacijom trajektorije slijeđenjem pristupa impedancije. U rješenju predloženom u ovom diplomskom radu, računaju se globalne

točke trajektorije, gdje se temeljem upravljanja po sili i poziciji osigurava prilagodba trajektorije i upotreba manje sile za otvaranje i zatvaranje ladice.

Diankov i drugi predstavljaju *Manipulation Planning with Caging Grasps* [8], odnosno novi algoritam za planiranje kretanja za obavljanje ograničenih zadataka kao što su otvaranje vrata i ladica. Njihova metoda izračunava hvat za pojedini objekt i koristi učinkovite algoritme pretraživanja za izradu planova kretanja koji zadovoljavaju ograničenja zadatka. Obavijajući hvat (engl. *caging grasp*) predstavlja vrstu hvata u kojoj prsti robota obavijaju prste oko ručke, čineći taj hvat čvrstim. Glavne prednosti njihove metode su značajno povećani raspon mogućnosti kretnje robota bez potrebe za provođenjem strogih ograničenja između vrha alata i ciljanog objekta. Kako bi testirali performanse algoritma, vrijeme planiranja i područja izvedivosti izračunate su za tri različita robota u simulaciji. Svaki robot je bio postavljen nasumično na podlozi i tada se pokrenulo planiranje. Tisuće različitih nasumičnih položaja se testiraju u svakoj sceni kako bi se izračunalo srednje vrijeme izvođenja. U stvarnosti su provedena dva eksperimenta s dva različita objekta koji su uspješno odradili zadani zadatak. Autori koriste algoritme pretrage kako bi odabrali plan kretnje kojim bi se otvorila ladica. Međutim, razlika predloženog rješenja u ovom diplomskom radu oslanja se na činjenicu kako se sve ladice otvaraju u jednom smjeru te se za sve ladice jednako računaju globalne točke trajektorije kada se odredi koordinatni sustav ladice.

3. METODOLOGIJA

U ovom poglavlju opisane su metode za računanje trajektorije otvaranja ladice i položaja robota prikladnog za hvatanje ručke ladice, metode za detekciju i izgradnju modela ladice te metoda za upravljanje po sili i poziciji.

3.1. Trajektorija otvaranja ladice

Ladica predstavlja pokretni objekt s jednim stupnjem slobode, što znači da se može otvoriti i zatvoriti duž jedne osi. Primjer predloženog koordinatnog sustava ladice, S_A, prikazan je slikom 3.1. Trajektorija ladice svodi se na translaciju po jednoj osi u pozitivnom ili negativnom smjeru x-osi koordinatnog sustava ladice, S_A.

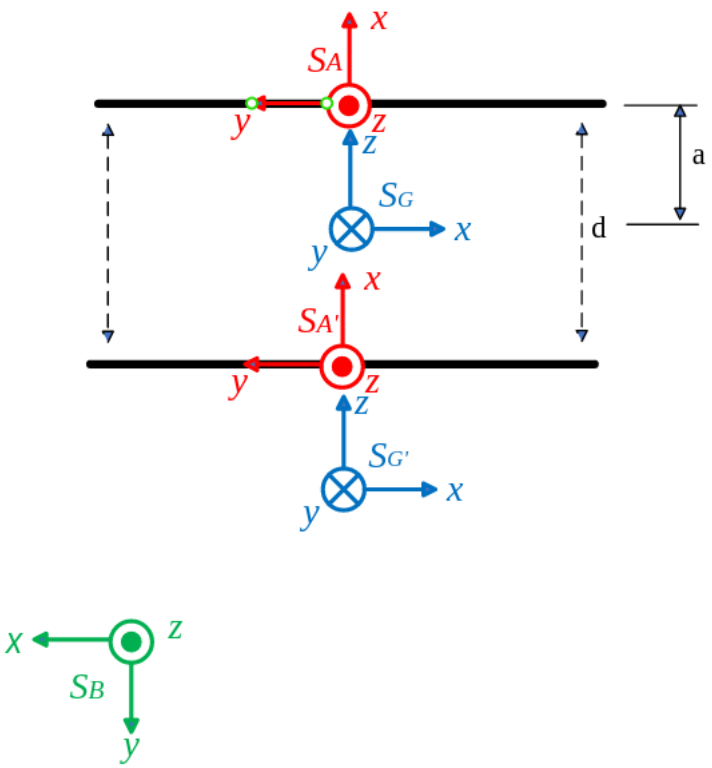
Slika 3.1: *Primjer postavljenih osi koordinatnog sustava ladice.*

Pretpostavlja se da je robotski manipulator postavljen ispred ormarića s ladicom. Potrebno je postaviti koordinatne sustave svih dijelova robotskog sustava i ladice te na temelju njihovih ovisnosti izračunati trajektoriju u 3D prostoru kako bi se ladica uspješno otvorila za zadanu duljinu otvorenosti. Koordinatni sustavi prikazani su na shemi 3.2 ladice koja se otvara iz ptičje perspektive.

Koordinatnim sustavom (KS) S_A definiran je koordinatni sustav ladice koju je potrebno otvoriti. S_B predstavlja koordinatni sustav baze robota, dok S_G predstavlja KS hvataljke s ishodištem u TCP-u robotskog manipulatora. TCP (engl. *Tool Center Point*) predstavlja točku pri samom vrhu zadnjeg zgloba robotskog manipulatora te ima zaseban koordinatni sustav, koji nije prikazan na slici. Kada se robotska ruka kreće linearno, TCP se kreće po ravnoj liniji. KS $S'_A i S'_G$ predstavljaju pomaknute koordinatne sustave nakon otvaranja ladice za duljinu d. Kao što je prikazano slikom 3.2 KS S_A i S_G se ne nalaze u istoj točki, već su udaljeni za a. Oznakom S_T označen je KS TCP-a. Odnos KS S_G i S_T prikazan je matricom homogene transformacije, dane izrazom 3-1, gdje ${}^T R_G$ predstavlja rotacijsku matricu hvataljke u odnosu na TCP, a ${}^T t_G$ predstavlja translacijski vektor hvataljke u odnosu na TCP. Odnos KS S_G i S_T je konstantan, te se izračun treba provesti jedanput za željenu rotaciju hvataljke.

$$ {}^T T_G = \left[\begin{array}{ccc:c} & {}^T R_G & & {}^T t_G \\ \hdashline 0 & 0 & 0 & 1 \end{array}\right] \tag{3-1} $$

Rotacijsku matricu ${}^T R_G$ nije moguće dobiti mjerenjem, već se računa na temelju poznatih rotacijskih matrica ${}^B R_T$ i ${}^B R_G$ prema izrazu 3-2. Najprije se hvataljka robotskog manipulatora postavlja u željenu orijentaciju, odnosno postavlja se željeni odnos KS S_G i S_B prikazan izrazom



Slika 3.2: *Koordinatni sustavi ladice i dijelova robotskog sustava*

3-3. Potom se očitava trenutni položaj manipulatora iz kojeg se formulira matrica ${}^{B}T_{T}$ iz koje se očita tražena rotacijska matrica ${}^{B}R_{T}$.

$$ {}^{T}R_{G} = {}^{B}R_{T}{}^{-1}{}^{B}R_{G} \tag{3-2} $$

$$ {}^{B}R_{G} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{3-3} $$

Iz dane matrice vidljivo je kako je x-os KS S_G suprotne orijentacije od x-osi KS S_B, y-os KS S_G je suprotno orijentirana od z-osi KS S_B i z-os KS S_G je suprotno orijentirana od y-osi KS S_B. Translacijski vektor ${}^{T}t_{G}$, prikazan izrazom 3-4, opisuje udaljenost TCP-a od KS hvataljke koji je smješten pri samom vrhu prstiju alata i to za iznos b.

$$ {}^{T}t_{G} = \begin{bmatrix} 0 \\ 0 \\ b \end{bmatrix} \tag{3-4} $$

Odnos KS S_A i S_G prikazan je matricom 3-5.

$$ {}^{A}T_{G} = \begin{bmatrix} 0 & 0 & 1 & -a \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-5} $$

Također, definiran je odnos KS S_A i S'_A, te je prikazan izrazom 3-6. Zbog pravocrtne trajektorije otvaranja ladice ova su dva KS-a jednake orijentacije, dok postoji pomak u negativnom x smjeru za vrijednost d.

$$ {}^{A}T_{A'} = \begin{bmatrix} 1 & 0 & 0 & -d \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-6} $$

Položaj hvatanja ručke robotskom hvataljkom opisan je matricom homogene transformacije ${}^{B}T_{T}$ koja prikazuje položaj KS S_T u donosu na KS S_B. Matrica ${}^{B}T_{T'}$ koja označava položaj vrha alata u koji se robot treba postaviti u odnosu na bazu, da bi otvorio ladicu za željeni iznos d, opisana je izrazom 3-7.

$$ {}^{B}T_{T'} = {}^{B}T_{T}{}^{T}T_{G}{}^{A}T_{G}{}^{-1}{}^{A}T_{A'}{}^{A'}T_{G'}{}^{T'}T_{G'}{}^{-1} \tag{3-7} $$

Odnos KS S_A i S_G nakon pomicanja ladice u nove koordinatne sustave $S_{A'}$ i $S_{G'}$, opisan izrazom 3-8, ostaje nepromijenjen jer se ladica i hvataljka translatiraju zajedno za isti iznos.

$$ {}^{A'}T_{G'} = {}^{A}T_{G} \tag{3-8} $$

Također, nepromijenjeni odnos zadržavaju i KS TCP-a i hvataljke zbog fiksne ugradnje hvataljke na TCP, što je prikazano izrazom 3-9.

$$ {}^{T'}T_{G'} = {}^{T}T_{G} \tag{3-9} $$

Točke između položaja opisanog matricom ${}^{T}T_{G}$ i položaja opisan matricom ${}^{T'}T_{G'}$ predstavljaju trajektoriju otvaranja ladice.

3.2. Hvatanje ručke robotskom hvataljkom

Prije samog hvatanja ručke i otvaranja ladice, robot se nalazi u položaju iz kojeg snima sekvencu slika na temelju koje stvara model ladice. U nastavku je opisano računanje prikladnog položaja u koji se robot mora postaviti kako bi uspješno uhvatio ručku prstima hvataljke.

Položaj hvataljke u odnosu na bazu robota, ${}^{B}T_{G}$, opisan je sljedećim izrazom.

$$ {}^{B}T_{G} = {}^{B}T_{T}{}^{T}T_{C}{}^{C}T_{A}{}^{A}T_{G} \tag{3-10} $$

Pri tome je matrica ${}^{A}T_{G}$ opisana izrazom 3-5, matrica ${}^{C}T_{A}$ dobivena je programom za detekciju vrata, matrica ${}^{T}T_{C}$ dobivena je kalibracijom sustava robot-kamera, a matrica ${}^{B}T_{T}$ opisuje položaj alata u trenutku snimanja slike te se može dobiti kinematikom robota.

Uzevši u obzir izraz 3-9, matrica ${}^{B}T_{G}$ može se opisati i izrazom 3-11, gdje ${}^{B}T_{T'}$ predstavlja položaj alata prikladan za hvatanje ručke ladice u odnosu na bazu.

$$ {}^{B}T_{G} = {}^{B}T_{T'}{}^{T}T_{G} \tag{3-11} $$

Izjednačavanjem desnih strana jednadžbi 3-10 i 3-11, dobiva se izraz 3-12,

$$ {}^{B}T_{T'}{}^{T}T_{G} = {}^{B}T_{T}{}^{T}T_{C}{}^{C}T_{A}{}^{A}T_{G} \tag{3-12} $$

iz čega konačno slijedi izraz 3-13, koji opisuje položaj alata prikladan za hvatanje ručke ladice.

$$ {}^{B}T_{T'} = {}^{B}T_{T}{}^{T}T_{C}{}^{C}T_{A}{}^{A}T_{G}{}^{T}T_{G}{}^{-1} \tag{3-13} $$

3.3. Algoritam za detekciju i izgradnju modela ladice

U ovom poglavlju opisan je algoritam koji se koristi za detekciju ladica i vrata, implementiran u programskoj biblioteci *Robot Vision Library*, RVL [9]. Algoritam proširuje kartu okoline koju je kreirao navigacijski sustav mobilnog robota umetanjem modela vrata i ladica u tu kartu. Svaki kreirani model vrata i ladica sadrži informaciju o veličini vrata ili ladice te poziciji i orijentaciji osi rotacije ili translacije u odnosu na referentnu kartu okoline. U nastavku su opisani model ladice i algoritmi za detekciju.

3.3.1 Model ladice

Prednja strana ladice predstavljena je kvadrom kojem su dodijeljeni koordinatni sustav S_B i vektor s koji opisuje veličinu ladice. Ostatak ladice nije modeliran.

Model ladice se može prikazati skupom $H = (\eta, {}^{C}T_A, s, \Theta)$, gdje η predstavlja tip detektiranog objekta: ladica ili vrata. Matrica ${}^{C}T_A$ predstavlja položaj koordinatnog sustava ladice, S_A, u odnosu na koordinatni sustav kamere, S_C. Vektor s sadrži vrijednosti procijenjene širine i dužine ladice. Parametar Θ predstavlja stanje otvorenosti, gdje je nulom označena zatvorena ladica. Ladica se otvara u negativnom smjeru x-osi KS S_A.

3.3.2 Metoda za detekciju ladice

Algoritam *Door and Drawer Detector* (DDD), iz biblioteke RVL, sastoji se od dva algoritma. Prvi algoritam DDD-THD (engl. *Teaching by Human Demonstration*), detektira vrata ili ladicu na temelju sekvence RGB-D slika snimljenih 3D kamerom. Drugi algoritam naziva DDD-SE, (engl. *State Estimator*), određuje trenutno stanje otvorenosti vrata ili ladice na RGB-D slici. Oba su algoritma opisana u nastavku, dok je za potrebe ovog diplomskog rada korišten samo DDD-THD.

DDD-THD algoritam:

Ulaz u DDD-THD algoritam predstavljaju trenutni položaj robota u odnosu na koordinatni sustav okoline, dobiven metodom robotske lokalizacije, i sekvenca snimljenih RGB-D slika koje prikazuju čovjeka kako otvara i zatvara vrata ili ladice. Važno je spomenutu sekvencu slika snimiti iz stacionarnog položaja kamere, kako bi jedini pomični objekti na snimljenoj sceni bili čovjek i pokretni dio artikuliranog objekta (vrata ili ladica). Korištenjem metode TensorMask [10] algoritam detektira i segmentira čovjeka u danim slikama. Zatim, algoritam uklanja čovjeka iz spremljenih slika kako bi jedini pomični objekt na sceni ostao pokretni dio artikuliranog objekta. Treba napomenuti da je u ovom diplomskom radu preskočen korak detekcije, segmentacije i uklanjanja čovjeka sa slike. Umjesto toga, na snimljenim scenama vidi se samo dio ljudske ruke koji zanemarivo utječe na daljnji rad algoritma. Potom algoritam generira jednu ili više hipoteza pokretnih dijelova za svaku sliku i integrira kreirane hipoteze snimljene sekvence kako bi u konačnici kreirao hipoteze vrata/ladica. Svakoj hipotezi se dodjeljuje vrijednost pouzdanosti (engl. *confidence score*). Konačni model vrata/ladice se kreira na temelju hipoteze s najvećom pouzdanosti te se on postavlja u kartu okoline. Hipoteze pokretnih dijelova se generiraju segmentiranjem scene na približno ravne segmente korištenjem prikladne segmentacijske metode i određivanjem koje od tih područja mijenja položaj u odnosu na kameru. Za detekciju ravninskih segmenata (engl. *planar patch*) korištena je metoda predložena u [11]. Pokretna područja se detektiraju usporedbom trenutne slike s nekom drugom slikom iz iste sekvence, unutar zadanog vremenskog okvira. Dvije usporedne slike su predstavljene kao 3D oblaci točaka i za svaku točku jednog oblaka se određuje najbliža točka u drugom oblaku točaka. Razlika između te dvije točke predstavlja vektor pomaka. Svi pronađeni vektori pomaka iz tog

para slika su klasterirani te se na temelju dominantnog klastera određuje dominantni vektor pomaka. Ravna površina s najvećim brojem točaka čiji vektori pomaka su dovoljno slični dominantnom vektoru pomaka se smatra prednjom stranom pomičnog dijela. Zatim se ta površina koristi za generiranje hipoteza pomičnih dijelova.

DDD-SE algoritam:

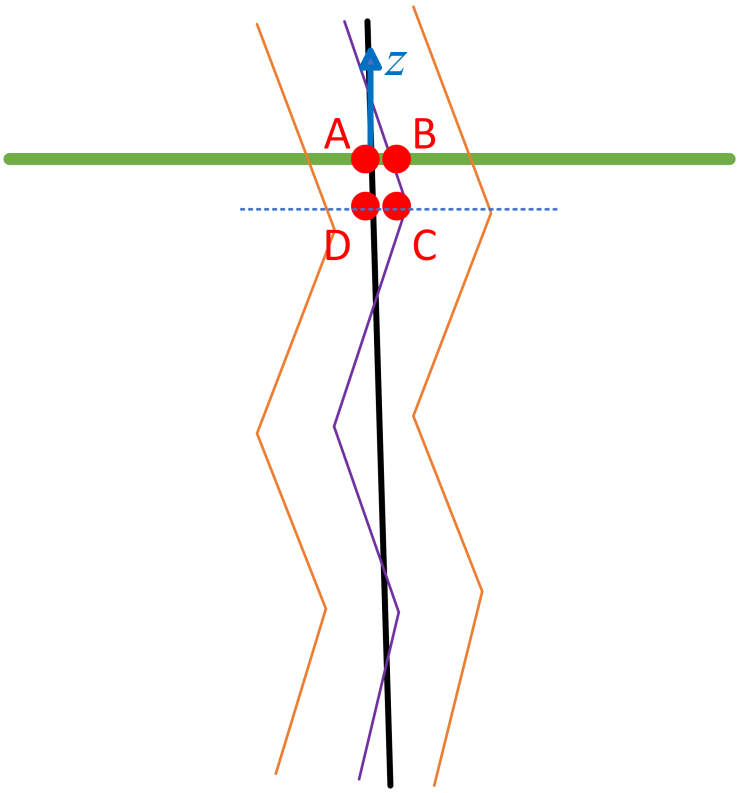
Određivanje stanja objekta, a to su kut panela vrata ili otvorenost ladice u odnosu na zatvoreno, nulto stanje, provodi se na sljedeći način. Temeljem danog položaja modela vrata ili ladice, područje interesa (engl. *region of interest*, ROI) je opisano kao 3D koordinatni sustav koji je usklađen s osima KS S_A. Veličina ROI-a u y i z-smjeru je jednaka drugom i trećem elementu vektora s, koji je proširen za 10%, dok je njegova veličina u x-smjeru konstantna vrijednost dovoljno velika da ROI sadrži vrata ili ladicu u bilo kojem stanju. Za svaki dio ravnine koji sadrži minimalno 1000 točaka unutar ROI-a, model se generira na način da je x-os KS S_G paralelna s normalom ravnine. Stanje vrata se predstavlja kao kut između x-osi KS S_G i S_A, dok se stanje ladice predstavlja kao udaljenost između ravnine i prednje strane ladice u nultom stanju.

Više o spomenutom DDD algoritmu može se pronaći u radu *Teaching a Robot Where Doors and Drawers Are and How To Handle Them* [9].

3.4. Upravljanje po sili i poziciji

Pri rješavanju problema otvaranja i zatvaranja ladica koristi se hibridno upravljanje po sili i poziciji [12]. Upravljanje silom (engl. *force control*) predstavlja tehniku reguliranja sile koja se primjenjuje pri specifičnom zadatku. Informacije o sili djelovanja između robotskog manipulatora i okoline dobivaju se uporabom senzora sile i momenta [12]. Upravljanje po sili i poziciji provodi se na način da se pozicijom upravlja u smjeru neograničenog gibanja (engl. *unconstrained task*), dok se upravljanje silom odvija u smjeru ograničenog gibanja (engl. *constrained task*). To znači da u slučaju otvaranja ladica, neograničeni je smjer otvaranja, dok bi ograničeni smjer bio duž širine i visine ladice. Povlačenje ladice u ograničenim smjerovima prevelikom silom može dovesti do njezina oštećenja, stoga je ciljana vrijednost sile u ograničenim smjerovima jednaka nuli. Točke trajektorije pružaju informaciju o neograničenom smjeru, ali one mogu biti neprecizne zbog šuma kamere ili pogreške zaokruživanja pri njihovu izračunu. Svako odstupanje idealne od izračunate trajektorije u ograničenim smjerovima ispravlja se pomoću upravljanja silom. Gibanje robota u neograničenom smjeru je kruto i slijedi poziciju prema izračunatoj trajektoriji, dok je istovremeno prilagodljivo u ograničenom smjeru gibanja i ispravlja svoj položaj kako bi se osigurala vrijednost sile približno nuli u tim smjerovima.

Slikom 3.3 prikazana je trajektorija ladice. Zelenom linijom prikazana je ladica u nekom trenutku k izvođenja trajektorije otvaranja. Crnom linijom prikazana je idealna trajektorija ladice. Ljubičastom bojom prikazana je trajektorija planirana na temelju podataka dobivenih računalnim vidom. Točka A je točka vrha alata (TCP-a) u trenutku k. Pretpostavlja se hibridno upravljanje po poziciji u smjeru z-osi alata, a po sili u lateralnim smjerovima alata x i y - pri čemu je zadana sila 0 N. Tada točka A u kojoj se stvarno nalazi TCP odgovara točki B na planiranoj trajektoriji u kojoj bi se trebao nalaziti TCP u trenutku k. Međutim, postoji dozvoljeni raspon odstupanja (engl. *deviation range*), koji omogućava upravljanje po sili u smjerovima x i y. Raspon odstupanja prikazan je narančastim linijama. Točka C je točka na planiranoj trajektoriji u kojoj bi se trebao nalaziti TCP u trenutku $k+1$. Plavi vektor z, predstavlja z-os alata u trenutku k. Točka D, ima istu z-koordinatu kao točka C (prikazano plavom linijom), ali su u njoj lateralne sile jednake nuli. U stvarnosti bi TCP trebao doći u tu točku u trenutku $k+1$. Na opisani način, TCP bi se trebao kretati po crnoj putanji uz lateralne sile jednake nuli.



Slika 3.3: *Skica trajektorije otvaranja ladice.*

4. ROBOTSKI SUSTAV I KORIŠTENI ALATI

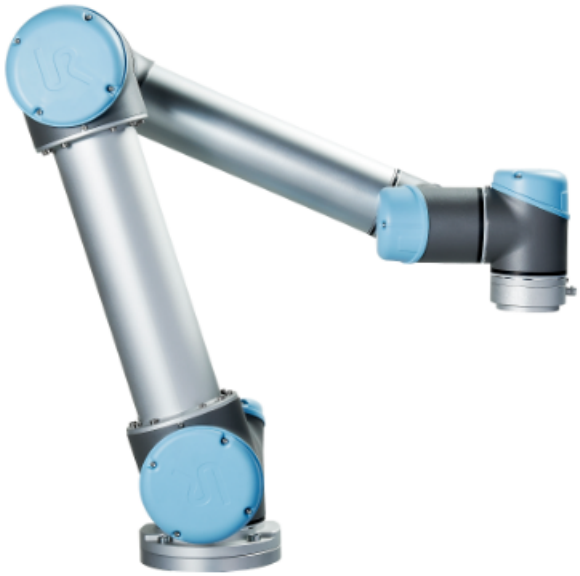
4.1. Eskperimentalni postav

Robotski sustav koji se u ovom radu koristi za zadatak otvaranja ladice sastoji se od: 6-osnog robotskog manipulatora *Universal Robots* UR5 s hvataljkom *Robotiq 3-Finger Gripper*, senzorom sile i momenta *Robotiq FT 300-S* i dubinskom kamerom u boji *Intel RealSense LiDAR L515*, koji su opisani u nastavku.

4.1.1 Universal Robots UR5

Universal Robots UR5 verzije 3.0 jest 6-osni robotski manipulator koji se koristi za obavljanje automatiziranih repetativnih radnji. Zbog jednostavnosti upravljanja, često se koristi za obavljanje radnji poput sastavljanja, pakiranja, testiranja i uzimanje predmeta pobacanih na hrpu jednog-po-jednog [13]. Izgled robotskog manipulatora prikazan je slikom 4.4. Ovakav manipulator teži 18.4 kg, ima kapacitet nosivosti do 5 kg i radijus dohvata 850 mm od centra baze [13]. Sastoji se od 6 rotacijskih zglobova koji omogućavaju pomicanje alata u cijelom radnom prostoru, osim točno iznad i ispod baze robota. Baza robota predstavlja donji dio manipulatora kojim je manipulator pričvršćen za podlogu. Sami vrh alata robota (ili *Wrist 3*) predstavlja dio na koji se mogu pričvrstiti razni senzori i aktuatori poput senzora sile, kamere i hvataljke.

Za upravljanje manipulatorom, koristi se grafičko korisničko sučelje *PolyScope* koje se pokreće na interaktivnom ekranu za upravljanje (engl. *teach pendant*). Unutar ovog sučelja nalaze se kontrole za pokretanje robota, izvršavanje predefiniranih programa te stvaranje novih. Za pomicanje robotskog manipulatora, mogu se koristiti naredbe unutar sučelja *PolyScope*, ali



Slika 4.4: *Universal Robots UR5 manipulator. [13]*

i zasebni gumb na poleđini ekrana čijim pritiskom se aktivira *freeDrive mode*, koji otključava zglobove robotskog manipulatora te ga čovjek ručno može dovesti u željeni položaj. Također, ovakav tip robota podržava *socket* komunikaciju korištenjem skriptnog jezika, *UR Script*, opisanog u poglavlju 5.4. Kako je riječ o kolaborativnom manipulatoru s ISO 10218-1 i ISO 10218-2 standardima, robot ne mora biti u kavezu, već se ljudi mogu slobodno kretati u njegovoj okolini. Također, ima ugrađenu funkciju *Protective Stop* koja se aktivira u slučaju kolizije robota s predmetom ili čovjekom u radnom prostoru robota. Ova funkcija automatski zaustavlja robota te je potrebna ponovna aktivacija od strane čovjeka. U slučaju prisilnoga zaustavljanja pritiskom na sigurnosni gumb STOP, robotski manipulator se trenutno zaustavlja te se može ponovno pokrenuti rotiranjem sigurnosnog gumba. Zbog navedenih karakteristika, ovaj manipulator prikladan je za obavljanje radnje otvaranja ladice.

4.1.2 Robotiq 3-Finger Gripper

Robotska hvataljka *Robotiq 3-Finger Gripper*, prikazana slikom 4.5, sastoji se od tri prsta koja se mogu zatvarati i otvarati kako bi se mogli hvatati i ispuštati razni objekti[14]. Na vrhu svakog prsta nalazi se gumirana podloga koja osigurava veće trenje prilikom manipuliranja predmetima. Postoje četiri stanja u koja se robotska hvataljka može dovesti: *basic, wide, pinch* i *scissor mode*. Za rad s ručkama ladica, korišten je *pinch mode* u kojem se dva prsta s iste strane stisnu jedan uz drugi, a treći je pozicioniran nasuprot njih te njihov oblik podsjeća izgledom na hvat pincete. Ovaj način hvatanja omogućava precizno i kontrolirano držanje malih objekata. Također, postoje dva oblika hvata: *fignertip grip* i *encompassing grip*. *Fingertip grip* odnosi se na hvat kada se objekt drži samo vrhovima prstiju te se u ovom slučaju stabilnost hvata održava zbog trenja između prstiju i objekta. S druge strane, *encompassing grip* predstavlja hvat u kojemu prsti hvataljke obuhvate predmet te stabilnost nije povezana s trenjem nego s nemogućnošću prolaska predmeta kroz zatvorene prste hvataljke. Kako je veličina ručke na ladicama značajno manja od veličine prstiju ove robotske hvataljke, prikladno je koristiti *pinch mode* s *fingertip* hvatom.

Slika 4.5: *Robotiq 3-Finger Gripper. [14]*

4.1.3 FT 300-S Force Torque Sensor

Senzor sile i momenta *FT 300-S Force Torque* tvrtke Robotiq, prikazan slikom 4.6, predstavlja periferni uređaj za prikupljanje podataka o sili i momentu [15]. Pričvršćuje se pri vrhu alata UR5 manipulatora, prije postavljanja hvataljke. Koristi se prilikom obavljanja radnji tijekom kojih je potrebno precizno mjerenje i upravljanje silom kako bi se osigurala zaštita od oštećenja uslijed djelovanja prevelike sile, ali i pravilno izvođenje zadanih zadataka, poput sastavljanja, testiranja kvalitete ili poliranja. Preciznost senzora iznosi +- 1.45 N i +-0.05 Nm. U ovom radu, korišten je za implementaciju upravljanja po sili i poziciji.

Slika 4.6: *Robotiq FT 300-S Force Torque Sensor. [15]*

4.1.4 Intel RealSense LiDAR L515 kamera

Kamera *Intel RealSense LiDAR L515*, prikazana slikom 4.7, predstavlja ujedno LiDAR, dubinsku i RGB kameru. Zahvaljujući LiDAR-u, omogućava dubinsku preciznost na udaljenostima od 25 cm do 9 m [16]. Također, opremljena je žiroskopom, akcelerometrom i inercijalnom mjernom jedinicom (engl. *Inertial Measurement Unit*). RBG kamera i LiDAR mogu snimati do 30 slika u sekundi u različitim rezolucijama. U ovom diplomskom radu, ova je kamera korištena za prikupljanje dubinskih i RGB slika na kojima su detektirane ladice.

Slika 4.7: *Intel RealSense LiDAR L515 kamera. [16]*

4.2. Razvojni alati

Operacijski sustav na kojem je razvijano rješenje predloženo u ovom diplomskom radu je Ubuntu 20.04.05. Za povezivanje upravljačkih programa robotskog sustava i razvijenih funkcija za obradu slike i upravljanje robotom, korišten je otvoreni softverski razvojni alat *Robot Operating System*, ROS, distribucije ROS Noetic. ROS je skup softverskih biblioteka i alata koji pomažu u razvijanju robotskih aplikacija [17]. Pod glavne komponente ROS-a ubrajamo: ROS Core, čvorove (engl. *nodes*), teme (engl. *topics*), poruke (engl. *messages*), servise (engl. *services*), akcije (engl. *actions*) i pakete (engl. *packages*). ROS Core je centralna komponenta koja omogućuje komunikaciju između različitih čvorova. Uključuje ujedno i Master čvor koji upravlja čvorovima te je ujedno i poslužitelj parametara koji omogućuje konfiguraciju i dijeljenje parametara između čvorova. Čvorovi su nezavisne aplikacije koje se individualno kompajliraju i pokreću te su organizirani u pakete. Tema ili tok poruke je osnovni način komunikacije unutar ROS-a. Komunikacija se odvija na način da jedan čvor objavljuje (engl. *publish*), a ostali se pretplaćuju (engl. *subscribe*) na primanje poruka. Poruke definiraju strukturu podataka koja služi za razmjenu informacije između čvorova i ujedno definiraju tip teme. Servisi omogućavaju dvosmjernu komunikaciju na principu zahtjeva i odgovora. S druge strane, akcije se razlikuju od servisa tako što ne blokiraju izvršenje programa klijenta dok čekaju odgovor poslužitelja. Akcije također pružaju dodatne mogućnosti za prekidanje zadatka i dobivanja statusa o stanju zadatka u nekim intervalima. Općenito, akcije su pogodne za dužu komunikaciju. Tijekom izrade ovog rada korišteni su ROS čvorovi i pretplaćivanje na teme kako bi se razmjenjivale željene informacije između čvorova.

Nadalje, korištena je platforma *Docker* koja služi za kreiranje kontejnera unutar kojih je omogućeno izvođenje aplikacija i njihovih ovisnosti u izoliranom okruženju. Na ovaj način moguće je pokretati više procesa i aplikacija odvojenih jednih od drugih radi boljeg iskorištenja infrastrukture uz sigurnost koja bi se inače dobila radom u odvojenom sustavu. Docker se koristi kao softversko okruženje za razvoj aplikacija i kasniju distribuciju uz prednost kompatibilnosti aplikacija [18]. To znači da će bilo koja aplikacija koja je razvijena u kontejneru raditi na bilo kojem Dockeru bez problema s kompatibilnošću [18]. U sklopu ovog diplomskog rada, korišten je Docker paket naziva: `rvl_ur5_detectron2` te je unutar spomenutog paketa izvedena instalacija svih potrebnih alata i biblioteka za daljnji rad. Primjer Docker datoteke prikazan je izlistanjem koda 1, gdje su vidljive potrebne instalacije za rad s kamerom i ROS-om. Cijela Docker datoteka nalazi se u Prilogu diplomskog rada.

Kod 1: *Primjer Dockerfile datoteke.*

```
RUN apt install python3-rosdep
RUN apt-get install -y ros-noetic-realsense2-camera
RUN apt-get install -y ros-noetic-openni-launch
RUN apt-get install -y ros-noetic-openni2-launch
RUN apt-get install -y ros-noetic-ros-numpy
RUN apt-get install -y ros-noetic-rosbash
RUN apt-get install -y ros-noetic-ros-control
RUN apt-get install -y ros-noetic-soem
RUN apt-get install -y ros-noetic-moveit
RUN apt-get install -y ros-noetic-trac-ik

RUN pip3 install pymodbus --upgrade
RUN pip3 install ur-rtde
RUN pip3 install pyyaml

RUN mkdir -p /home/RVLuser/ur5_ws/src
COPY ur5_ws/src/ /home/RVLuser/ur5_ws/src/
RUN ls /opt/ros/noetic
RUN rosdep init
```

Za detekciju ladica na slici i izgradnju modela ladice, korištena je programska biblioteka *Robot Vision Library*, RVL. Ova je biblioteka razvijena u sklopu rada Istraživačke grupe za inteligentne sustave i robotiku na FERIT-u. Korištene funkcije iz RVL biblioteke opisane su u poglavlju 3.3.2.

Za razvijanje programa i pokretanje Docker kontejnera, korišteno je razvojno okruženje Visual Studio Code.

5. IMPLEMENTACIJA ALGORITMA

Ovo poglavlje opisuje implementacju algoritama opisanih u poglavlju 3. Implementirani algoritam otvaranja i zatvaranja ladice provodi se na sljedeći način. Robot se ručno postavlja u položaj u kojem kamera vidi ladicu koja se treba otvoriti. Snima se sekvenca slika ljudske demonstracije otvaranja ladice te se kreiraju PLY (engl. *Polygon File Format*) datoteke na temelju kojih se izgrađuje model. PLY format opisan je u poglavlju 5.2. Nakon izgradnje modela otvara se hvataljka. Zatim se robot, ukoliko se u međuvremenu pomaknuo, postavlja u položaj iz kojeg je snimljena sekvenca slika. Iz tog položaja, računa se početni položaj vrha alata prikladan za hvatanje ručke ladice. Robot se zatim postavlja u taj položaj. Kada se postavi ispred ručke, vrijednosti senzora sile i momenta postavljaju se na nulu te se zatvara hvataljka koja hvata ručku. Prilikom hvatanja ručke, pokreće se upravljanje po sili i momentu. U tom trenutku nastupa računanje trajektorije otvaranja ladice, tj. proizvoljnog broja točaka u prostoru kroz koje TCP robota mora proći kako bi uspješno otvorio ladicu. TCP robota prolazi redom kroz točke tijekom otvaranja, a obrnutim redom tijekom zatvaranja ladice. Nakon što je ladica zatvorena, upravljanje silom i momentom se gasi, otvara se hvataljka te je robot završio sa svojim zadatkom.

Svi opisani algoritmi implementirani su u Python programskom jeziku.

5.1. Snimanje i pohrana sekvence slika

Algoritam je osmišljen na način da se robot postavi u položaj ispred ormarića s ladicom te na pritisak tipke `space` započinje snimanje i spremanje oba tipa slike, a ponovnim pritiskom tipke se snimanje zaustavlja. U tom se trenutku zapisuju položaj TCP-a i kutovi svih zglobova robotskog manipulatora u odnosu na bazu kako bi se zapamtio položaj robota iz kojega su snimljene slike. Kreiranje novog ROS čvora te pretplata na teme `/camera/color/image_raw` za RGB sliku i `/camera/aligned_depth_to_color/image_raw` za dubinsku sliku prikazani su kodom 2. Funkcija `make_files()` kreira datoteke u koje će se spremati snimljene slike. Također, kako bi se istovremeno spremala oba tipa slike korišten je `ApproximateTimeSynchronizer` mehanizam. Funkcija `image_callback()` pretvara *ROS Image* poruku u *OpenCV Image* poruku, kako bi se moglo uz pomoć OpenCV biblioteke prikazati trenutno snimljena slika s kamere na zaslonu računala. U ovom slučaju, prilikom pretvaranja koristi se *CvBridge()* paket uz predavanje vrste kodiranja: *bgr8* za RGB sliku i *passthrough* za dubinsku sliku. U spomenutoj *callback* funkciji očekuje se korisnikov pritisak na tipku `space` za spremanje snimljenih slika te zaustavljanje snimanja, kako je već objašnjeno.

Kod 2: *Isječak funkcije take_image.py.*

```
1  rospy.init_node('rgbAndDepth_image_subscriber')
2  make_files()
3  rgb_image_sub = Subscriber('/camera/color/image_raw', Image)
4  depth_image_sub = Subscriber('/camera/aligned_depth_to_color/image_raw', Image)
5
6  synchronizer = ApproximateTimeSynchronizer([rgb_image_sub, depth_image_sub],
       ↪ queue_size = 5, slop=0.1)
7  synchronizer.registerCallback(image_callback)
```

5.2. Kreiranje PLY datoteke i stvaranje modela ladice

Sljedeći korak predstavlja kreiranje PLY datoteka na temelju snimljenih RGB i dubinskih slika. PLY format predstavlja 3D format datoteke koja pohranjuje grafičke objekte kao skup poligona. Objekti u PLY formatu su opisani skupom vrhova ili drugih elemenata sa svojstvima poput boje i smjera normale za pojedini element [19]. PLY datoteke se generiraju pomoću biblioteke RVL. Nakon generiranja PLY datoteka na temelju svake snimljene slike, započinje generiranje modela. Generiranje modela se radi u funkciji `DDDetector` biblioteke RVL. Najprije se učitavaju PLY datoteke i zatim se pokreće postupak generiranja hipoteza koji rezultira detektiranjem ladice. U slučaju uspješnog detektiranja ladice, dobiva se klasa `drawer`, a u slučaju da se ladica nije uspjela detektirati, parametar η postavlja se na `unknown`. Nakon generiranja, model se pohranjuje u datoteku.

5.3. Otvaranje i zatvaranje ladice

U nastavku su opisani ključni koraci algoritma otvaranja i zatvaranja ladice.

5.3.1 Računanje matrice homogene transformacije ${}^{B}T_{T}$

Ciljni položaj alata robota za hvatanje ručke ladice podrazumijeva položaj u kojem alat robotskog manipulatora može uspješno uhvatiti ručku prilikom zatvaranja hvataljke. Položaj robota definira se s vektorom položaja koji se sastoji od šest elemenata čije prve tri komponente opisuje pomak TCP-a u x, y i z-smjeru u odnosu na bazu robotskog manipulatora, a zadnje tri komponente predstavljaju kutove rotacije, odnosno rotacijski vektor. Spomenuti položaj predstavljen je transformacijskom matricom ${}^{B}T_{T}$. Prvi korak podrazumijeva izračun rotacijske matrice ${}^{B}R_{T}$ koja daje odnos rotacije TCP-a i baze robotskog manipulatora. U tu svrhu implementirana je `get_transformation_matrix_from_pose_vector` funkcija koja računa matricu homogene transformacije ${}^{B}T_{T}$ iz predanog trenutno očitanog položaja robotskog manipulatora. Rotacijski dio se računa prema matričnom prikazu 5-14.

$$\begin{bmatrix} k_x^2(1-c\theta)+c\theta & k_yk_x(1-c\theta)-k_zs\theta & k_zk_x(1-c\theta)+k_ys\theta \\ k_xk_y(1-c\theta)+k_zs\theta & k_y^2(1-c\theta)+c\theta & k_zk_y(1-c\theta)-k_xs\theta \\ k_xk_z(1-c\theta)-k_ys\theta & k_yk_z(1-c\theta)+k_xs\theta & k_z^2(1-c\theta)+c\theta \end{bmatrix} \tag{5-14}$$

Rotacijski vektor, koji se predaje kao zadnje tri komponente vektora položaja robota očitanog iz programa, normira se na jedinični vektor da bi se dobile k_x, k_y, k_z komponente vektora k, gdje je duljina tog vektora označena s θ. Uvrštavanjem tih vrijednosti u 5-14, dobiva se rotacijska matrica, a posljedično transformacijska matrica. Drugim riječima, oznakama k_x, k_y, k_z predstavljene su komponente jediničnog vektora k koji predstavlja os rotacije, a duljina tog rotacijskog vektora je označena s θ, izražen u radijanima. Oznaka c predstavlja funkciju kosinus, a oznaka s sinus.

Nakon izračuna matrice, njezinom trećem stupcu potrebno je promijeniti vrijednosti na trenutne rotacijske kutove. Tada, matrica dana izrazom 5-14 predstavlja rotacijsku matricu ${}^{B}R_{T}$. Konačno, matrica ${}^{B}T_{T}$ se dobiva na način da se unutar jedinične matrice dimenzije $4x4$, kopira rotacijska matrica na mjesto prva tri retka i prva tri stupca, gdje se u translacijski dio (četvrti stupac) prepisuju prve tri vrijednosti iz predanog vektora položaja.

5.3.2 Računanje matrice homogene transformacije ${}^{T}T_{G}$

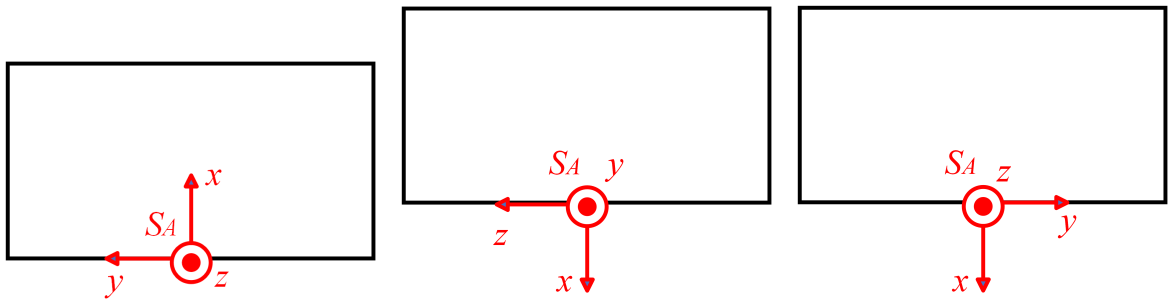
Kako je zahtjevno ručno izmjeriti rotaciju hvataljke u odnosu na TCP, u nastavku je opisan postupak za računanje. Najprije se ormarić s ladicom poravnava s KS baze robotskog manipulatora na način da je y-os KS S_A paralelna s x-osi KS S_B 3.2. Zatim se robotski manipulator dovodi u položaj gdje su prsti hvataljke orijentirani tako da robotski manipulator može uhvatiti ručku ladice, na način da je z-os hvataljke poravnata s $-y$-osi baze. Jedan prst hvataljke se nalazi na gornjem dijelu ručke, dok se dva prsta nalaze ispod ručke. Matricu homogene transformacije hvataljke u odnosu na TCP, ${}^{T}T_{G}$, potrebno je izračunati samo jednom te ona vrijedi za sva daljnja računanja. Postoji nekoliko načina prikaza orijentacije, te je u ovom slučaju odabran RPY tip. RPY kutovi predstavljaju *roll, pitch* i *yaw* kutove, gdje se kutovi zadaju u stupnjevima u odnosu na TCP robota [13]. *Roll* predstavlja orijentaciju oko x-osi robota, *pitch* predstavlja nagib ili rotaciju oko y-osi robota te *yaw* predstavlja zakretanje tj. rotaciju oko z-osi robota. Željena orijentacija hvataljke za otvaranje ladice postavljena je na RPY(90°, 45°, 0°). Potom se pokrene funkcija naziva `calculate_TGT()`, koja najprije računa ${}^{B}T_{T}$ iz trenutno očitanog položaja u kojem se robot nalazi. Potom se temeljem izraza 3-2 i 3-3 dobiva ${}^{T}T_{G}$ matrica homogene transformacije.

5.3.3 Računanje početnog položaja prikladnog za hvatanje ručke ladice

Nakon gore opisanih izračuna matrica homogenih transformacija, potrebno je izračunati položaj u koji se robotski manipulator treba postaviti kako bi uhvatio ručku ladice. U poglavlju 3.2 dan je opis matematičkog problema s općenitim rješenjem. Međutim, kako u stvarnosti rad s algoritmom za detekciju daje drugačiju rotaciju KS S_A od postavljenog referentnog KS S_A prikazanog slikom 3.2, u nastavku će biti posebno objašnjen svaki matematički korak za računanje početnog položaja.

Za računanje položaja alata za hvatanje ručke, napisana je funkcija `calculate_starting_position()` unutar koje je riješena jednadžba 3-13.

Homogena transformacijska matrica ${}^{B}T_{T}$ računa se kako je opisano u poglavlju 5.3.1. Homogena matrica transformacije ${}^{T}T_{C}$ se dobiva kalibracijom sustava robot-kamera. Homogena matrica transformacije ${}^{A}T_{G}$ odgovara matrici opisanoj izrazom 3-5. Inverzna homogena transformacijska matrica ${}^{T}T_{G}{}^{-1}$ računa se kako je opisano u poglavlju 5.3.2. Međutim, matrica ${}^{C}T_{A}$ dobivena algoritmom detekcije je drugačije orijentirana nego što je pretpostavljeno u 3.2. Tijekom izvođenja algoritma za detekciju ladice i kreiranje modela, može doći do drugačije pretpostavke orijentiranja KS S_A te su dva primjera KS S_A prikazana slikom 5.8.



(a) *Orijentacija referentnog S_A.* **(b)** *Orijentacija S_A kada je s = [visina, dužina].* **(c)** *Orijentacija S_A kada je s = [dužina, visina].*

Slika 5.8: *Primjeri orijentacija S_A.*

Naime, program detektira pravokutnu površinu koja se pomiče u smjeru normale na tu površinu, a to je upravo smjer x-osi KS S_A. Druge dvije osi, y i z, program orijentira u smjeru stranica (visina i dužina) pravokutnika, te nije određeno koja će stranica biti paralelan s y, a koji sa z-osi. Stoga se nekada dogodi da vektor s iz modela ladice bude zapisan kao: [dužina, visina], a nekad kao [visina, dužina] ladice. Programski se radi dodatna rotacija estimirane matrice ${}^{C}T_{A}$ tako da KS S_A iz izgrađenog modela nakon rotacije odgovara referentnom položaju prikazanog slikom 5.8a. Ovaj korak je nužan kako bi se mogla koristiti postavljena jednadžba 3-13. U prvom slučaju, KS S_A se detektira kao na slici 5.8b te se rotacijsku matricu ${}^{C}R_{A}$ rotira oko y-osi za 180° i oko x-osi za -90°. U drugom slučaju, KS S_A se detektira kao na slici 5.8c te se rotacijsku matricu ${}^{C}R_{A}$ rotira oko y-osi za 180° i oko x-osi za -180°. Na spomenuti način nastavljaju se koristiti prethodno objašnjene matematičke relacije te se izlazom iz ove funkcije dobiva matrica homogene transformacije ${}^{B}T_{T'}$ iz koje se, kako je opisano u 5.3.1, dobiva novi vektor položaja od 6 elemenata koji definiraju pomake u x, y i z-smjeru, te rotacije R_x,R_y i R_z TCP-a u odnosu na bazu robotskog manipulatora.

5.3.4 Računanje trajektorije otvaranja i zatvaranja ladice

Trajektorija otvaranja i zatvaranja ladice sastoji se od niza jednako razmaknutih točaka koje leže na istom pravcu. Te točke se računaju funkcijom `sample_drawer_opening_point`, kojoj se predaju parametri: vektor trenutnog položaja robota, `startPoseVector`, duljina od stanja zatvorenosti do stanja otvorenosti ladice u prvoj točki, `d`, i broj točaka trajektorije otvaranja ladice, `numberOfPoints`. Pozivom ove funkcije s parametrima: `d=0.05, numberOfPoints=4`, algoritam definira četiri točke u prostoru, gdje je svaka nova točka udaljena od prethodne točke za 0.05 metara. Na kraju izvođenja algoritma, ladica će se otvoriti za duljinu od $0.05 \cdot 4 = 0.2m$. Programski kod ove funkcije prikazan je kodom 3. Izlaz iz funkcije je lista položaja u koje se TCP robotskog manipulatora mora postaviti kako bi uspješno izvršio trajektoriju otvaranja ladice. Za zatvaranje ladice, koriste se iste točke, samo suprotnim redoslijedom. Funkcija `get_next_drawer_pose` prima dva parametra: transformacijsku matricu trenutnog položaja ${}^{B}T_{T}$, označenu s `prev_poseTB` i parametar duljine otvaranja ladice, `d`. Ova funkcija kao rezultat vraća transformacijsku matricu ${}^{B}T_{T'}$ koju će robotski manipulator imati u sljedećoj izračunatoj točki putanje temeljem formule 3-7, pomaknutoj u odnosu na trenutnu točku za iznos d u smjeru $-z$-osi KS S_T.

Kod 3: *Isječak iz funkcije sample_drawer_opening_point.*

```
1  def sample_drawer_opening_point(startPoseVector, d, numberOfPoints):
2
3      TTB = get_transformation_matrix_from_pose_vector(startPoseVector)
4      poses = []
5      prev_poseTB = TTB
6      for i in range(0, numberOfPoints):
7          TT_B = get_next_drawer_pose(prev_poseTB, d)
8          pose_vecT_B = get_pose_vector_from_transformation_matrix(TT_B)
9          poses.append(pose_vecT_B)
10         prev_poseTB = get_transformation_matrix_from_pose_vector(pose_vecT_B).copy
               ↪ ()
11
12     return poses
```

5.4. Upravljanje silom i momentom

Kako bi se dinamički upravljalo silom i pozicijom, potrebno je komunicirati s robotskim manipulatorom preko *socket* komunikacije slanjem UR skripti. *URScript* je specijalni nižerazinski skriptni jezik koji se koristi za upravljanje robotima Universal Robots, poput UR5 robotskog manipulatora, korištenog u ovom radu. *URControl* predstavlja nižerazinski robotski kontroler koji se pokreće na Mini-ITX matičnoj ploči u kontrolnoj kutiji [20]. Kada se spomenuta matična ploča podigne, URControl pokreće demona (npr. servis) te se PolyScope ili GUI mogu povezati na njega korištenjem lokalne TCP/IP konekcije [20]. Programiranje robota na skriptnom nivou napravljeno je na način da se napiše aplikacija za klijenta (koja se pokreće na drugom računalu) i spaja na URControl korištenjem TCP/IP soketa [20]. Postavljaju se host i port, gdje je u ovom slučaju host IP adresa manipulatora: `192.168.22.14`, a port: `30002`. Kada se konekcija uspostavi, UR Script programi ili naredbe se šalju kao čisti tekst preko soketa. Potrebno je naglasiti kako svaka linija u skriptnom jeziku mora završavati sa znakom `\n`. Kako bi URControl prepoznao skriptni kod, mora zadovoljavati sljedeća 3 uvjeta:

1. skripta mora počinjati definicijom funkcije,
2. sve skriptne linije moraju biti uvučene za barem jedan "white" razmak i
3. zadnja linija skripte mora završavati s ključnom riječi "end" [20].

Kako bi se robotskom manipulatoru zadala željena trajektorija, potrebno je koristiti sljedeće postojeće funkcije: `zero_ftsensor()`, `force_mode(task_frame, selection_vector, wrench, type, limits)`, `tool_pose()`, `movej(q, a = 1.4, v = 1.05, t = 0, r = 0)` i `get_inverse_kin(x, qnear, maxPositionError = 1e-10, maxOrientationError = 1e-10, tcp = 'active_tcp')`.

Funkcija `zero_ftsensor()` resetira mjerenja očitana senzorom sile i momenta, koji je ugrađen na TCP robotskog manipulatora, na način da oduzme trenutna mjerenja u idućem trenutku.

Funckija `force_mode(task_frame, selection_vector, wrench, type, limits)`, postavlja robota u stanje prikladno za upravljanje u `force mode` načinu [20]. Sastoji se od četiri parametra. Parametar `task_frame` predstavlja vektor pozicije koji definira KS sile u odnosu na KS baze robota. Parametar `selection_vector` predstavlja binarni vektor od šest elemenata, gdje 1 znači da će robot biti tolerantan (engl. *compliant*) duž odgovarajuće osi zadanog KS. Parametar `wrench` definira sile i momente koje će robot primijeniti na svoju okolinu. Robot prilagođava svoju poziciju oko/duž tolerantne osi kako bi postigao zadanu silu ili moment. Ako su postavljene vrijednosti za tolerantne osi jednake nuli, robot ih zanemaruje te ne upravlja po njima. Stvarna sila ili moment mogu u stvarnosti biti značajno manji od zadanog zbog sigurnosnih ograničenja zglobova. Stvarne vrijednosti sile i momenta se mogu pročitati pomoću funkcije `get_tcp_force()` u zasebnoj niti. Parametar `type` odnosi se na cjelobrojni (engl. *integer*) broj od 1 do 3 koji definira kako robot interpretira KS sile:

1. KS sile je transformiran na način da je njegova y-os usporedna s vektorom koji pokazuje iz robotskog TCP-a prema ishodištu KS sile.
2. KS sile nije transformiran.
3. KS sile je transformiran na način da je njegova x-os projekcija vektora brzine robotskog TCP-a na xy-ravninu tog KS sile[20].

Zadnji parametar, `limits`, predstavlja decimalni (engl. *float*) šest-dimenzionalni vektor, gdje za tolerantnu os vrijednosti predstavljaju maksimalnu dopuštenu brzinu TCP-a duž/oko te

osi. Za netolerantnu os vrijednosti predstavljaju maksimum dozvoljene devijacije duž/oko osi između stvarne TCP pozicije i one koja je dobivena programom.

Funkcija `tool_pose()` vraća informacije o trenutnom položaju alata.

Funkcija `movej(q, a=1.4, v=1.05, t=0, r=0)` pomiče robotski manipulator po linearnoj putanji u prostora zglobova od trenutne do ciljne točke[20]. Kada se koristi ova funkcija, robot mora biti u stanju mirovanja ili se prethodno kretati `movej` ili `movel` funkcijom. Funkcija prima pet parametara: vrijednosti zglobova, akceleraciju zglobova u vodećoj osi u rad/s^2, brzinu zglobova u vodećoj osi u rad/s, vrijeme u sekundama i polumjer spoja (engl. *blend radius*). Polumjer spoja predstavlja duljinu glatkog prijelaza između dvije putanje kada robot mijenja smjer kretanja. Ako je zadan polumjer spoja, trajektorija robotske ruke će biti prilagođena za izbjegavanje zaustavljanja robota u svakoj postignutoj točki.

Funkcija `get_inverse_kin(x, qnear, maxPositionError = 1e-10, maxOrientationError = 1e-10, tcp = 'active_tcp')` računa inverznu kinematiku iz prostora alata u prostor zglobova [20]. Parametar `x` predstavlja položaj TCP-a. Parametar `qnear` predstavlja listu vrijednosti zglobova (neobavezno za postavljanje). Ako je parametar `qnear` definiran, dobiva se rješenje najbliže tom parametru, suprotno, dobiva se rješenje najbliže trenutnim vrijednostima zglobova. Parametar `maxPositionError` predstavlja maksimalnu dopuštenu pogrešku u poziciji od zadane (neobavezno za postavljanje). Parametar `maxOrientationError` predstavlja maksimalnu dopuštenu pogrešku orijentacije od zadane (neobavezno za postavljanje). Parametar `tcp` predstavlja odstupanje ciljnog položaja TCP-a od trenutnog (neobavezno za postavljanje), kada nije postavljen parametar koristi se trenutni položaj TCP u odnosu na bazu za daljnje računanje. Kao izlaznu vrijednost, ova funckija vraća vrijednosti zglobova dobivenih inverznom kinematikom.

UR Script-ni kod koji se šalje robotskom manipulatoru generira se automatski korištenjem funkcije `write_drawer_opening_closing_UR_script`, s parametrima: `poses`, koje predstavljaju listu vektora položaja za zadani broj točaka i parametrom `start_pose` koji predstavlja vektor položaja robotskog manipulatora za hvatanje ručke. Funkcija je vidljiva kodom 4. Ovom funkcijom kreira se tekstualna datoteka koja predstavlja UR Script-ni kod vidljiv kodom 5, koji se potom šalje kontroleru robotskog manipulatora pomoću *socket* komunikacije. Na ovaj način osigurava se upravljanje robotskim manipulatorom putem računala.

Kod 4: *Primjer funkcije za generiranje skriptnog koda.*

```
1  def write_drawer_opening_closing_UR_script(poses,start_pose):
2  n=len(poses)
3  with open("/home/RVLuser/ur5_ws/src/ao_manipulation/scripts/drawer_files/
       ↪ URscript_%d.txt"%model_number, "w") as f:
4      f.write("def zero():\n")
5      f.write("zero_ftsensor()\n")
6      f.write("end\n")
7      f.write("def door_open():\n")
8      f.write("force_mode(tool_pose(), [1, 1, 0, 0, 0, 0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
           ↪ 2, [0.2, 0.2, 0.1, 0.60, 0.60, 0.35])\n")
9      for i in range(0,n):
10         f.write("movej(get_inverse_kin(p"+ str(poses[i]) +"),a=1.4,v=0.2)\n")
11     f.write("end\n")
12     f.write("def door_close():\n")
13     f.write("force_mode(tool_pose(), [1, 1, 0, 0, 0, 0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0],
           ↪ 2, [0.2, 0.2, 0.1, 0.60, 0.60, 0.35])\n")
14     for i in range(0,n):
15         f.write("movej(get_inverse_kin(p"+ str(poses[n-1-i]) +"),a=1.4,v=0.2)\n")
```

```
16          f.write("movej(get_inverse_kin(p"+ str(start_pose) +"),a=1.4,v=0.2)\n")
17          f.write("end\n")
18          f.write("zero()\n")
19          f.write("door_open()\n")
20          f.write("door_close()\n")
21          f.close()
```

Kod 5: *Primjer UR Scripte za otvaranje i zatvaranje ladice.*

```
1  def zero():
2  zero_ftsensor()
3  end
4  def drawer_open():
5  force_mode(tool_pose(), [1, 1, 0, 0, 0, 0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 2, [0.2, 0.2, 0.1, 0.60,
       ↪  0.60, 0.35])
6  movej(get_inverse_kin(p[−0.2798493223683374, −0.6751136746193462,
       ↪ 0.5014374104069097, 1.5514956133506401, 0.4158412372703615,
       ↪ −0.5672168454032708]),a=1.4,v=0.2)
7  movej(get_inverse_kin(p[−0.27477699464966876, −0.6253727467414995,
       ↪ 0.5014374104069097, 1.5514956133506403, 0.41584123727036154,
       ↪ −0.5672168454032707]),a=1.4,v=0.2)
8  movej(get_inverse_kin(p[−0.2697046669310001, −0.5756318188636527,
       ↪ 0.5014374104069097, 1.5514956133506403, 0.41584123727036154,
       ↪ −0.5672168454032707]),a=1.4,v=0.2)
9  movej(get_inverse_kin(p[−0.2646323392123314, −0.5258908909858058,
       ↪ 0.5014374104069097, 1.5514956133506403, 0.41584123727036154,
       ↪ −0.5672168454032707]),a=1.4,v=0.2)
10 end
11 def drawer_close():
12 force_mode(tool_pose(), [1, 1, 0, 0, 0, 0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 2, [0.2, 0.2, 0.1, 0.60,
       ↪  0.60, 0.35])
13 movej(get_inverse_kin(p[−0.2646323392123314, −0.5258908909858058,
       ↪ 0.5014374104069097, 1.5514956133506403, 0.41584123727036154,
       ↪ −0.5672168454032707]),a=1.4,v=0.2)
14 movej(get_inverse_kin(p[−0.2697046669310001, −0.5756318188636527,
       ↪ 0.5014374104069097, 1.5514956133506403, 0.41584123727036154,
       ↪ −0.5672168454032707]),a=1.4,v=0.2)
15 movej(get_inverse_kin(p[−0.27477699464966876, −0.6253727467414995,
       ↪ 0.5014374104069097, 1.5514956133506403, 0.41584123727036154,
       ↪ −0.5672168454032707]),a=1.4,v=0.2)
16 movej(get_inverse_kin(p[−0.2798493223683374, −0.6751136746193462,
       ↪ 0.5014374104069097, 1.5514956133506401, 0.4158412372703615,
       ↪ −0.5672168454032708]),a=1.4,v=0.2)
17 movej(get_inverse_kin(p[−0.2849216500870061, −0.7248546024971931,
       ↪ 0.5014374104069097, 1.5514956133506397, 0.4158412372703614,
       ↪ −0.5672168454032708]),a=1.4,v=0.2)
18 end
19 zero()
20 drawer_open()
21 drawer_close()
```

Prva linija UR Script-nog koda predstavlja definiranje funkcije `zero()`, za resetiranje vrijednosti senzora. Od četvrte do desete linije koda prikazana je funkcija za otvaranje ladice. Od jedanaeste do osamnaeste linije prikazana je funkcija za zatvaranje ladice. U oba slučaja, i za otvaranje i za zatvaranje, korištena je ista konfiguracija parametara `force_mode` funkcije te ona glasi: `force_mode(tool_pose(), [1, 1, 0, 0, 0, 0], [0.0, 0.0, 0.0, 0.0, 0.0, 0.0], 2, [0.2, 0.2, 0.1, 0.60, 0.60, 0.35])`. Prvi parametar definiran je kao `tool_pose()`, što označava koordinatni sustav TCP-a. Drugi parametar `[1, 1, 0, 0, 0, 0]` pokazuje kako je postavljeno ograničenje sile samo za x i y-os, te je zadana sila od 0.0 N za x i y os: `[0.0, 0.0, 0.0, 0.0, 0.0, 0.0]`. Parametar tipa je postavljen na `type=2` čime se postavlja da KS nije transformiran. Zadnjim parametrom: `[0.2, 0.2, 0.1, 0.60, 0.60, 0.35]`, definirane su maksimalne brzine za x i y-os te maksimalne devijacije za netolerantne osi. Maksimalna brzina u x i y-smjeru iznosi 100 mm/s, maksimalno odstupanje z-osi iznosi 100 mm, maksimalna R_x i R_y rotacija je 0.6 radijana ili približno 35°, dok je za R_z maksimalno odstupanje rotacije zadano približno 20°.

Kada se poziva funkcija `force_mode`, ona djeluje na izvođenje svih drugih funkcija koje su naknadno pozvane u istoj niti. Primjer funkcije koja se izvodi u istoj niti je `movej` što znači da će se robot pomicati uz regulaciju sile i pozicije. Primjer prve točke u koju se robot mora postaviti da bi započeo otvaranje ladice je opisan sljedećim pozivom funkcije: `movej(get_inverse_kin(p[-0.279, -0.675, 0.501, 1.551, 0.415, -0.567]), a=1.4, v=0.2)`. Prvi parametar odnosi se na dobivanje inverzne kinematike za predanu 3D točku u koju se treba postaviti TCP robota. Vidljivo je da je tražena točka u odnosu na bazu pomaknuta u $-x$-smjeru za 0.28 m (desno od ishodišta baznog KS), u $-y$-smjeru za 0.68 m (ispred KS baze) i u z-smjeru za 0.5 m (iznad KS baze). Nakon izračunate inverzne kinematike, `movej` funkcija pomiče robota u nove kutove zglobova na način da pomak bude linearan u prostoru zglobova s akceleracijom od 1.4 rad/s^2 i brzinom od 0.2 rad/s. Kako je vidljivo u primjeru koda 5, definirane su četiri `movej` funkcije za otvaranje - što upućuje na četiri točke u koje se robot treba postaviti da bi otvorio ladicu za duljinu $4 \cdot d$. Kod zatvaranja se definira pet točaka, navedene četiri za otvaranje, ali u suprotnom redoslijedu i peta točka koja predstavlja položaj u kojem je robot uhvatio ručku - ona se postavlja kako bi robot doveo ladicu u isti položaj iz kojeg je započeta kretnja.

6. EKSPERIMENTALNA EVALUACIJA

U ovom poglavlju opisani su pokusi provedeni kako bi se mogla ocijeniti uspješnost razvijenog programa za otvaranje i zatvaranje ladice robotskom rukom pomoću računalnog vida i upravljanja silom.

6.1. Postav i izvođenje pokusa

Za uspješno izvođenje razvijenog programa potrebno je računalo snažnih performansi te UR5 robotski manipulator, senzor sile i momenta FT 300-S, robotska hvataljka Robotiq 3-F Gripper, dubinska kamera Intel RealSense LiDAR L515 te ormarić s barem jednom ladicom. U ovim je pokusima korišten ormarić s dvije ladice i vratima, a za otvaranje se koristila srednja ladica. Opisani postav prikazan je slikom 6.9. Kako bi pokus mogao započeti, ručka ladice mora se nalaziti u radnom prostoru UR5 robota te je u ovom slučaju ishodište ručke bilo postavljeno na udaljenosti `(x,y,z) = (-0.19, -0.52, 0.59)` u odnosu na bazni KS robotskog manipulatora. Na ovaj način dobiva se postav kao na slici 3.2.

Slika 6.9: *Ekserimentalni postav - robotski sustav i ormarić s ladicama.*

Kako bi se moglo upravljati robotskim manipulatorom, potrebno ga je najprije ručno upaliti te postaviti ukupni iznos mase (engl. *payload*) koja je dodana na zadnji članak, a čine je hvataljka, senzor sile i momenta i kamera, ukupne mase 3.2 kg. Prije svakog upravljanja

robotskim manipulatorom potrebno je i uspostaviti TCP/IP konekciju, ručno unijeti IP adresu računala unutar PolyScopa u kartici *External Control* te ručno pokrenuti aktivaciju hvataljke.

Sljedeći korak podrazumijeva povezivanje računala s robotom i pokretanje razvijenih funkcija za njegovo upravljanje, što je ostvareno preko sljedećih naredbi koje se unose u zasebne terminale na računalu povezanom na istu TCP/IP konekciju:

1. $ ./run_docker.sh
2. $ hostname -I
3. $ roslaunch ur_robot_driver ur5_bringup.launch robot_ip:=192.168.22.14 kinematics_config:=/home/RVLuser/ur5_calibration.yaml
4. $ roslaunch realsense2_camera rs_camera.launch enable_pointcloud:=true depth_width:=640 depth_height:=480 depth_fps:=15 color_width:=640 color_height:=480 color_fps:=15 align_depth:=true
5. $ rosrun robotiq_3f_gripper_control Robotiq3FGripperTcpNode.py 192.168.22.11
6. $ rosrun ao_manipulation take_image.py
7. $ rosrun ao_manipulation build_model.py
8. $ rosrun ao_manipulation openClose_drawer.py

Naredbom `./run_docker.sh` otvara se kontejner unutar Dockera gdje se zatim pokreću sve ostale naredbe. Naredbom `hostname -I` provjerava se IP adresa računala koja se unosi u PolyScope. Naredbom `roslaunch ur_robot_driver ur5_bringup.launch robot_ip:= 192.168.22.14 kinematics_config:=/home/RVLuser/ur5_calibration.yaml` uspostavlja se konekcija s robotom za njegovo upravljanje. Naredbom `roslaunch realsense2_camera rs_camera.launch enable_pointcloud:=true depth_width:=640 depth_height:=480 depth_fps:=15 color_width:=640 color_height:=480 color_fps:=15 align_depth:=true` pokreće se čvor za dobivanje vrijednosti s kamere Intel RealSense LiDAR L515.
Naredbom `rosrun robotiq_3f_gripper_control Robotiq3FGripperTcpNode.py 192.168.22.11` računalo se povezuje na 3-F Gripper i omogućava njegovo upravljanje. Nakon osnovnog povezivanja, postav je spreman za testiranje cjelokupnog programa. Međutim, zbog nedostatka dovoljne procesorske snage u računalu koje je korišteno za odrađivanje diplomskog rada, cjelokupni algoritam se morao rastaviti na manje dijelove kako bi se izbjeglo smrzavanje i padanje sustava na računalu. Zbog toga su navedene tri naredbe za pokretanje tri različita algoritma nazvana: `take_image.py`, `build_model.py` i `openClose_drawer.py`.

Najprije se pokreće naredba `rosrun ao_manipulation take_image.py`, kojom se otvara prozor sa slikom dobivenom postavljenom Intel RealSense LiDar kamerom. U ovom trenutku potrebno je namjestiti robotski manipulator u položaj iz kojeg je jasno vidljiva demonstracija otvaranja ladice. Primjer postavljanja u takav položaj prikazan je slikom 6.10a. Za potrebe snimanja slike, zbog osjetljivosti kamere, ugašeno je svjetlo u laboratoriju te je slikom 6.10b prikazano kakva se RGB slika dobiva Intel RealSense LiDAR L515 kamerom iz tog položaja.

Nakon dovođenja robotskog manipulatora u željeni položaj upotrebom *freeDrive* načina, `take_image.py` funkcija očekuje pritisak "space" tipke kako bi započelo snimanje sekvence slika frekvencijom od 10 slika u sekundi, gdje se istovremeno snimaju RGB i dubinske slike osobe koja otvara ladicu. Ladica se može otvoriti na način kako je prikazano slikom 6.10b - hvatanjem za rub ili povlačenjem za ručku. U slučaju povlačenja za ručku može doći do manjeg odstupanja u procjeni širine i visine ladice, zbog toga što ruka prekrije određeni dio ladice, koji onda nije vidljiv na slici. Snimljene su sekvence od 30 slika za otvaranje ladice. Ponovnim pritiskom na

(a) *Položaj robotskog manipulatora prikladan za snimanje scene.*

(b) *Snimljena RGB slika iz položaja prikazanog na slici 6.10a.*

Slika 6.10: *Primjer položaja robotskog manipulatora i RGB slike snimljene iz tog položaja.*

"space" tipku, program završava. Snimljena sekvenca RGB slika prikazana je slikom 6.11, dok je pripadajuća sekvenca snimljenih dubinskih slika prikazana slikom 6.12.

(a) *Prva snimljena slika.*

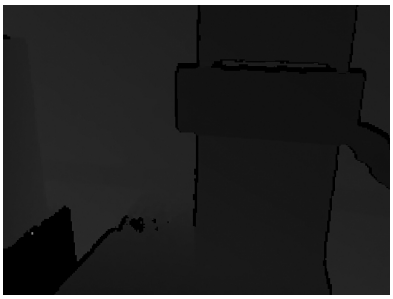
(b) *Snimljena slika u sredini sekvence.*

(c) *Zadnja snimljena slika.*

Slika 6.11: *Sekvenca snimanja RGB slika.*

(a) *Prva snimljena slika.*

(b) *Snimljena slika u sredini sekvence.*

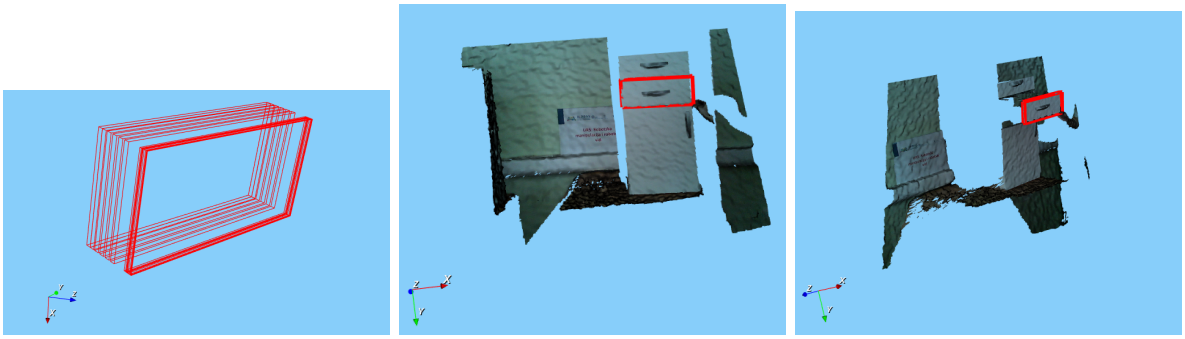


(c) *Zadnja snimljena slika.*

Slika 6.12: *Sekvenca snimanja dubinskih slika.*

Nakon snimljenih i spremljenih RGB i dubinskih slika, pokreće se naredba: `rosrun`

`ao_manipulation build_model.py`, kojom se prvo kreiraju PLY datoteke unutar nove mape nazvane "ply", a potom se na temelju tih PLY datoteka izgrađuje model ladice. Primjer generirane hipoteze otvaranja ladice vidljiv je na slici 6.13a. Dva primjera detekcije ladice na prvoj i zadnjoj slici u sekvenci otvaranja ladice prikazana su na slikama 6.13b i 6.13c. Izlaz iz danog programa je izgrađeni model, prikazan kodom 6.



(a) *Primjer generiranih hipoteza otvaranja ladice.* **(b)** *Detekcija ladice kada je zatvorena.* **(c)** *Detekcija ladice kada je otvorena.*

Slika 6.13: *Prikaz generirane hipoteze i detekcije ladice build_model.py algoritmom.*

Kod 6: *Dobiveni model ladice.*

```
{'object_class': 'drawer', 'R': array([[−0.1919654 , 0.02972439, −0.98095155],
        [ 0.32749203, −0.9403073 , −0.09258074],
        [−0.9251477 , −0.3390261 , 0.17077197]], dtype=float32), 't': array([ 0.1979409 ,
            ↪ −0.15877956, 0.989609 ], dtype=float32), 's': array([0.15697597,
            ↪ 0.39924842], dtype=float32), 'r': array([ 0.01180495, −0.34409824], dtype
            ↪ =float32), 'openingDirection': 1.0}
```

U estimiranom modelu vidljiva je rotacijska matrica označena s R koja predstavlja rotacijski dio homogene matrice transformacije ${}^{C}T_{A}$ i translacijski vektor t koji predstavlja zadnji stupac te matrice. Iz vektora s, vidljiva je procijenjena visine ladice od 0.156 m, dok je procijenjena širina ladice 0.399 m. Ukoliko dobivene vrijednosti usporedimo sa stvarnim vrijednostima ladice: visina=0.149 m i širina=0.397 m, zaključujemo kako je ovakav model uspio dovoljno dobro estimirati visinu i širinu ladice. Ovakvom usporedbom, odlučuje se je li model točan ili ne. U ovom trenutku, robotski manipulator ima sve informacije potrebne za pokušaj otvaranja i zatvaranja ladice koje se pokreće naredbom `rosrun ao_manipulation openClose_drawer.py`

Izračunata je i matrica ${}^{G}T_{T}$ na način opisan u poglavlju 5.3.2 te ona iznosi:

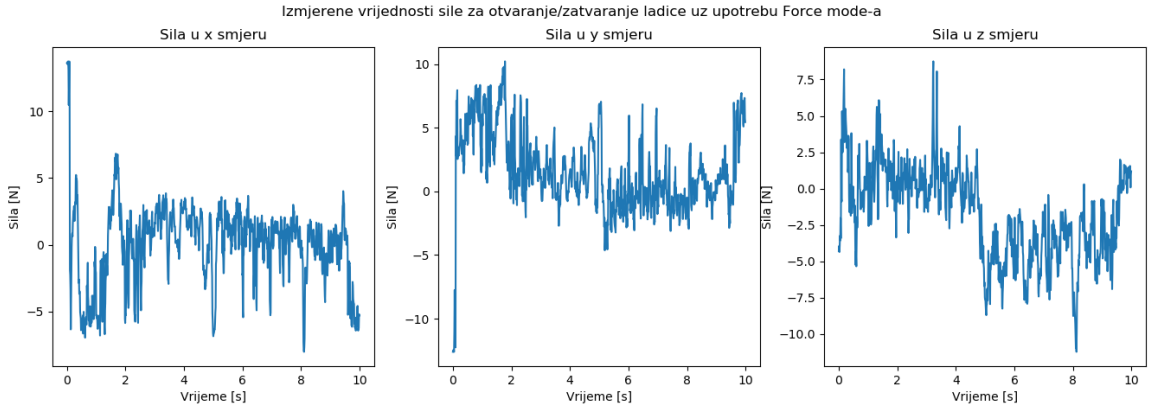
Kod 7: ${}^{T}T_{G}$ *matrica za rotaciju alata: (90°,45°,0°).*

```
TTG:
[[−7.07145644e−01, 7.07067914e−01, −5.64536543e−05, 0.00000000e+00]
[−7.07067916e−01, −7.07145643e−01, 3.24043285e−05, 0.00000000e+00]
[−1.70088947e−05, 6.28311474e−05, 9.99999998e−01, 2.80000000e−01]
[ 0.00000000e+00, 0.00000000e+00, 0.00000000e+00, 1.00000000e+00]]
```

Iz zadnjeg stupca ove matrice vidljiva je udaljenost duž z-osi S_G KS u odnosu na S_T KS u iznosu od 0.28m (stvarna izmjerena udaljenost). Također, iz rotacijske matrice, vidljivo je kako su z-osi KS TCP-a i hvataljke paralelne, dok je x-os TCP-a zarotirana za približno -45°u odnosu na x-os hvataljke i približno 45°u odnosu na y-os hvataljke; te je y-os TCP-a zarotirana

za približno -45°u odnosu na x i y-os hvataljke. Drugim riječima, KS TCP-a je zarotiran oko z-osi KS S_G za 45 stupnjeva. Nakon izračuna matrice, robotski manipulator se najprije postavlja u položaj iz kojeg je detektirana ladica, zatim iz tog položaja računa prvi položaj u koji se treba postaviti kako bi mogao zatvaranjem hvataljke obujmiti ručku ladice u *pinch* načinu.

Potom se upotrebom FT senzora upravlja po sili i poziciji i robot izvodi trajektoriju otvaranja, a potom zatvaranja. Slikom 6.14 prikazan je graf sila očitanih s FT senzora u ovisnosti o vremenu. Na danom grafu, primijenjena sila otvaranja ladice je prikazana u vremenu od 0 s do 5 s, a od 5 s do 10 s prikazana je sila prilikom zatvaranje ladice. Ukoliko pogledamo iznos sila, vidljivo je kako sila u x-smjeru najprije iznosi -5 N (početak otvaranja kada se uključuje upravljanje silom), a zatim je njena vrijednost manja od početne sile te varira oko nule. Može se primijetiti značajnije odstupanje iznosa sile oko osme sekunde, što govori kako je robot tada više gurao u smjeru tolerantne osi te se trenutak kasnije uočava smanjenje sile na vrijednost približno 0 N. Spomenuti trenutak ukazuje kako je robotski manipulator krenio u smjeru zadane točke trajektorije koja zbog nepreciznosti izračuna odstupa od idealne točke trajektorije te robotski manipulator gura većom silom (velika odstupanja u x i y smjeru oko osme sekunde). Međutim, trenutak kasnije se upravlja po sili u ta dva smjera gdje se sila potom vidljivo smanjuje. U y-smjeru, tijekom otvaranja ladice, primjećuje se u prvih nekoliko sekundi veća sila što uzrokuje povlačenje ladice u tom smjeru zbog neprecizno izračunate točke trajektorije, no već u sljedećim sekundama vidljiva je vrijednost sile približno nula. U z-smjeru, vidljivo je otvaranje ladice silom otprilike 2.5 N te zatvaranje ladice s -5 N. Analizom sile prije prve sekunde, može se primijetiti upravljanje po sili prilikom otvaranja ladice. U spomenutom trenutku sila u z-smjeru ima negativni predznak, što upućuje da je robotski manipulator vrlo kratko djelovao u suprotnom smjeru od otvaranja, kako bi smanjio silu u x i y smjeru.



Slika 6.14: *Prikaz djelovanja sila za vrijeme otvaranja i zatvaranja ladice. [21]*

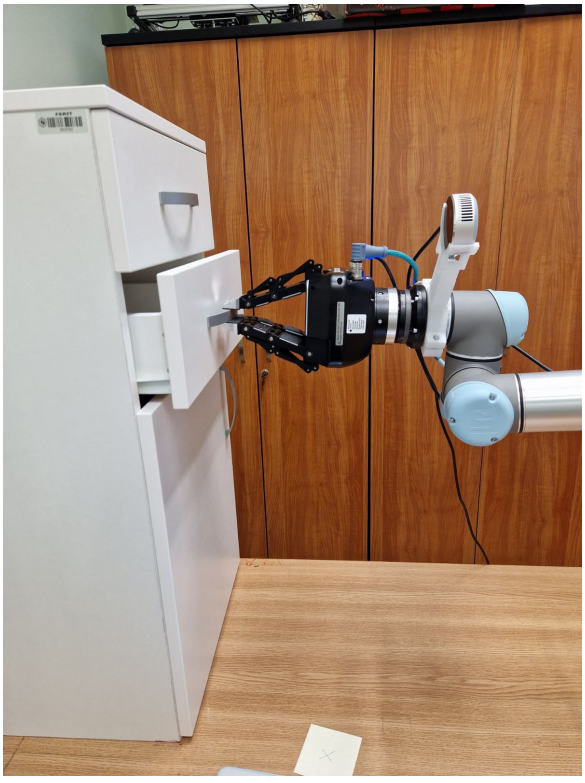
Slikom 6.15, prikazano je uspješno otvaranje ladice.

6.2. Rezultati pokusa

Provedena su ukupno 24 pokusa. Rezultati pokusa prikazani su tablicom 6.1. Pokusi su podijeljeni u tri faze i pratila se uspješnost svake od njih: 1. uspješnost sustava za detekciju ladice, označena kao "Detekcija", 2. uspješnost određivanja ciljnog položaja alata bez javljanja samokolizije, označeno kao "Zaštita robota" te 3. uspješnost otvaranja i zatvaranja ladice, označena kao "Otvaranje/zatvaranje" u tablici 6.1. Uspješnost detekcije odnosi se na procjenu matrice homogene transformacije ${}^{C}T_{A}$ i vektora dimenzije ladice, uspješnost ugrađenog sustava podrazumijeva izračun inverzne kinematike, čija pogreška koja može dovesti do sudara robota sa okolinom ili samim sobom te, konačno, uspješnost otvaranja i zatvaranja ladice opisuje je

(a) *Dolazak do ladice.*

(b) *Hvatanje za ručku.*



(c) *Prva razina otvaranja ladice.*

(d) *Druga razina otvaranja ladice.*

Slika 6.15: *Otvaranje ladice robotskim manipulatorom.*

li robotski manipulator uspio otvoriti i zatvoriti ladicu nakon uspješne prve dvije faze pokusa. Pokus se smatra potpuno uspješnim ako su uspješne sve tri faze.

Moguće su sljedeće kombinacije:

1. ✓✓✓= Oznaka potpuno uspješnog pokusa.
2. ✗- - = Pokus je prekinut u fazi detekcije.
3. ✗✓✗= Pokus se odvijao uz pogrešnu detekciju te nije uspješno završio.
4. ✓✗- = Robotski manipulator je zaustavljen prije faze otvaranja zbog pogreške određivanja ciljnog položaja bez samokolizije.

Pokus	Detekcija	Zaštita robota	Otvaranje/zatvaranje	Komentar
0	✓	✓	✓	
1	✗	-	-	Pogreška detekcije: class "unknown".
2	✓	✓	✓	
3	✗	✓	✗	Pogrešno izračunata visina ladice, hvataljka se nije uspjela zatvoriti i uhvatiti ručku.
4	✓	✓	✓	
5	✗	✓	✗	Detektiran cijeli ormarić kao ladica.
6	✓	✗	-	Zaustavljen robot zbog kolizije robota i kamere.
7	✓	✓	✓	
8	✓	✓	✓	Hvataljka uhvatila ručku s 2 prsta umjesto s 3.
9	✓	✓	✓	
10	✓	✓	✓	
11	✗	✓	✗	Krivo izračunata udaljenost od ladice, sudar s ladicom.
12	✓	✓	✓	
13	✓	✓	✓	
14	✓	✓	✓	
15	✓	✓	✓	
16	✓	✓	✓	
17	✗	✓	✗	Krivo izračunata udaljenost od ladice, sudar s ladicom.
18	✗	✓	✗	Krivo izračunata visina ručke, sudar s ladicom.
19	✗	✓	✗	Krivo izračunata udaljenost i visina do ladice, sudar s ladicom.
20	✗	✓	✗	Krivo izračunata visina ladice, hvataljka se nije uspjela zatvoriti i uhvatiti ručku.
21	✓	✓	✓	
22	✓	✓	✓	
23	✓	✓	✓	

Tablica 6.1: *Rezultati provedenih pokusa.*

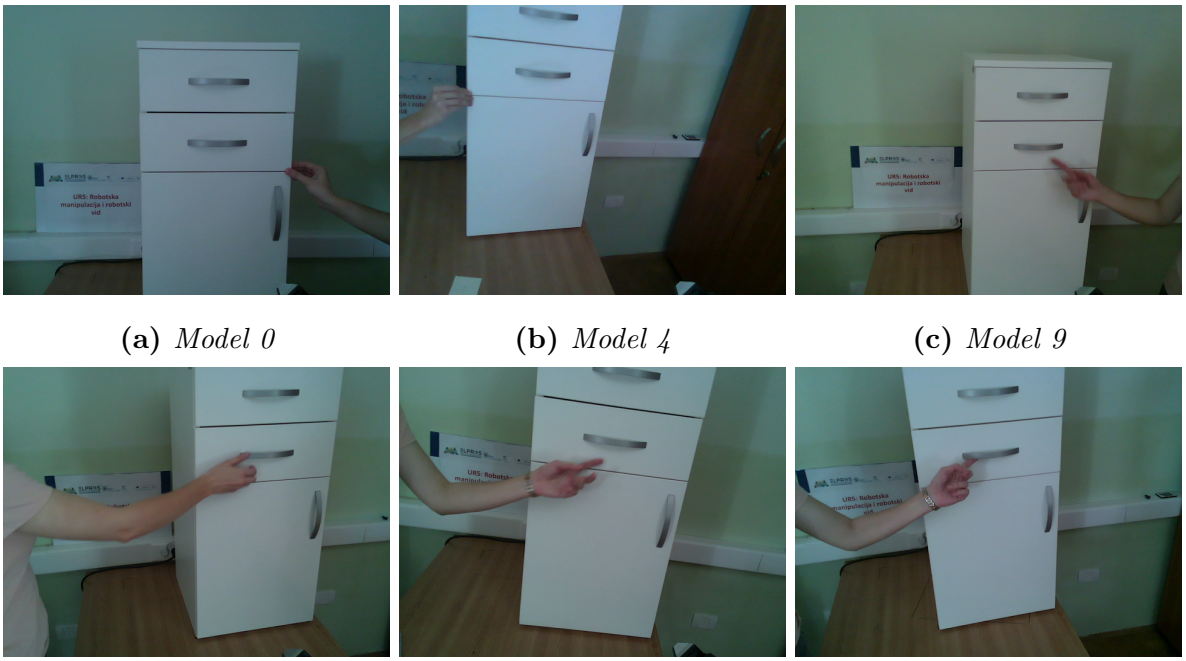
Rezultati provedenih pokusa ukazuju na nekoliko tipova pogreške. Najčešće je do problema dolazilo u fazi detekcije te je ladica pogrešno prepoznata ili je neprecizno određena njezina visina ili udaljenost ručke ladice od trenutnog položaja robotskog manipulatora. Do pogreške je došlo i zbog javljanja samokolizije, kada se držač kamere i predzadnjeg zgloba robotskog manipulatora sudario za vrijeme dolaska u položaj za hvatanje ručke. U jednom slučaju je pokus izveden uspješno iako je robotski manipulator uhvatio ručku ladice sa samo dva prsta, umjesto sa sva tri.

Slikom 6.16 prikazane su RGB slike modela 0, 4, 9, 15, 16 i 23 na kojima je program uspješno detektirao model. Slikom 6.17 prikazane su RGB slike modela 1, 3, 17, 18, 19 i 20 na kojima program nije uspješno detektirao model. Iz priloženih slika može se zaključiti kako sustav dobro detektira model kada se nalazi ispred ili lijevo od ladice, uz položaj kamere u visini ladice koja se otvara. S druge strane, kada je robot bio postavljen ispred ladice ali u ptičjoj perspektivi ili s desne strane ladice, detekcija modela je radila pogrešno.

Tablicom 6.2 prikazana je uspješnost izvođenja pokusa od 62.5%. Može se zaključiti kako dobivena uspješnost nije zadovoljavajuća u kontekstu stvarnog korištenja jer bi ovakav sustav u stvarnoj primjeni imao veliku vjerojatnost pogreške. Međutim, uspješnost se može smatrati zadovoljavajućom u kontekstu razvijanja robotskog sustava koji gotovo samostalno provodi postupak stvaranja modela i izvođenja trajektorije otvaranja ladice.

Ukupan broj pokusa	Uspješni	Neuspješni
24	15	9
	62.5%	37.5%

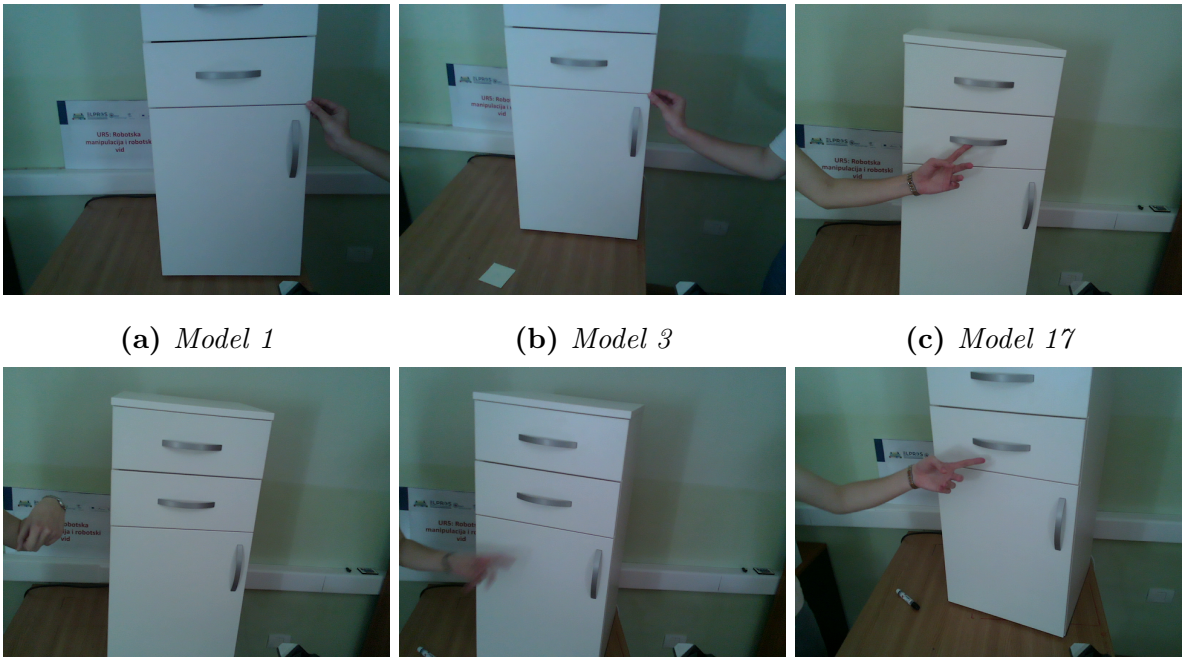
Tablica 6.2: *Ukupni rezultat provedenih eksperimenata.*



(a) *Model 0* **(b)** *Model 4* **(c)** *Model 9*

(d) *Model 15* **(e)** *Model 16* **(f)** *Model 23*

Slika 6.16: *Slike na kojima je uspješno detektiran model.*



(a) *Model 1* **(b)** *Model 3* **(c)** *Model 17*

(d) *Model 18* **(e)** *Model 19* **(f)** *Model 20*

Slika 6.17: *Slike na kojima je neuspješno detektiran model.*

7. ZAKLJUČAK

U ovom diplomskom radu razvijeno je programsko rješenje za izračun trajektorije otvaranja i zatvaranja ladice koja se nalazi unutar radnog prostora robotskog manipulatora, na temelju izgrađenog modela ladice. Za razvoj ovog rješenja i provođenje pokusa korišteni su Ubuntu operacijski sustav, ROS Noetic, Docker, programska bibilioteka RVL te programski jezik Python i UR Script. Robotski sustav na kojem je provedena implementacija i ispitivanje rješenja sastoji se od UR5 robotskog manipulatora na kojeg su ugrađeni senzor sile i momenta FT 300-S, hvataljka 3-F Gripper i kamera Intel RealSense LiDAR L515.

Postupak izvođenja trajektorije otvaranja i zatvaranja ladice proveden je uz adaptaciju trajektorije regulacijom sile. Hibridna regulacija sile i pozicije ostvarena je korištenjem funkcije *force_mode* na način da se u lateralnim smjerovima iznos sile minimizirao na nulu, dok se pozicijom upravljalo u smjeru otvaranja i zatvaranja ladice. Upravljanje silom prilikom izvođenja trajektorije je neophodno jer neke fizikalne komponente ladice, poput mase ili podmazanosti kotačića, nisu poznati. Robotski manipulator, stoga, mora prilagođavati svoje kretnje kako bi uspio izvršiti otvaranje ili zatvaranje ladice bez upotrebe prevelike sile koja može oštetiti namještaj ili samog robota. Izračun trajektorije otvaranja ladice ovisan je o modelu otvaranja ladice koji se izgrađuje na temelju prethodno snimljene sekvence slika čovjekove demonstracije otvaranja ladice. Model ladice sadrži sve potrebne informacije kako bi se estimirao položaj ladice u 3D prostoru u odnosu na trenutni položaj robota. Postupak otvaranja ladice po izračunatoj trajektoriji provodi se na sljedeći način. Prvo se računa prihvatljiv položaj za hvatanje ručke ladice. Nakon postavljanja robotskog manipulatora u položaj za hvatanje ručke, računa se globalna trajektorija i robotski manipulator izvodi zadanu trajektoriju koristeći upravljanje po sili i poziciji.

Eksperimentalnom evaluacijom vrednovala se uspješnost otvaranja i zatvaranja ladice, koja iznosi 62.5% za ukupnih 24 pokušaja. Zaključeno je kako najčešće pogreške proizlaze iz netočne detekcije modela i samim time pogrešne procjene položaja koordinatnog sustava ladice u odnosu na koordinatni sustav kamere. Time automatski dolazi do pogreške kod dovođenja robotskog manipulatora u položaj prikladan za hvatanje ručke. S obzirom da se za detekciju i izgradnju modela ladice koristila vanjska biblioteka RVL, nije bio moguć utjecaj na taj dio programa kako bi se osigurala veća točnost modela. Međutim, primijećeno je kako početni položaj snimanja sekvence slika utječe na točnost izgradnje modela te se najveća točnost pokazala kada se vidno polje kamere, koja je pričvršćena na robotski manipulator, nalazilo u visini razine otvaranja, smješteno ispred ili lijevo od ladice koja se želi otvoriti.

Može se zaključiti kako predloženo rješenje u svom trenutnom obliku ne zadovoljava zahtjeve manipulacije u stvarnosti, zbog velike vjerojatnosti pogreške modela. Prilikom eksperimentalne analize identificirano je nekoliko područja u programskom rješenju koja se mogu poboljšati. Nedovoljno precizna detekcija i izgradnja modela predstavlja osnovni problem na koji se može utjecati na nekoliko načina: snimanjem sekvence slika iz gore opisanog položaja kamere i unapređenjem postojeće RVL biblioteke ili korištenjem druge biblioteke koja bi dala veću točnost modela. Uz upotrebu računala s boljom procesorskom moći osigurala bi se veća brzina izgradnje modela te paralelizacija cijelog programa. U slučaju manje pogreške modela u procjeni visine ladice, bilo bi učinkovito regulirati silu prilikom hvatanja ručke, kako bi se izbjeglo naglo podizanje ladice. Uz navedena poboljšanja, razvijeni sustav za otvaranje ladice robotskom rukom bi u budućnosti mogao uspješno samostalno otvarati i zatvarati ladice.

LITERATURA

[1] Advait Jain and Charles C. Kemp. Pulling open doors and drawers: Coordinating an omni-directional base and a compliant arm with equilibrium point control. In *2010 IEEE International Conference on Robotics and Automation*, pages 1807–1814, 2010.

[2] Suseong Kim, Hoseong Seo, and H. Jin Kim. Operating an unknown drawer using an aerial manipulator. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5503–5508, 2015.

[3] Thomas Rühr, Jürgen Sturm, Dejan Pangercic, Michael Beetz, and Daniel Cremers. A generalized framework for opening doors and drawers in kitchen environments. In *2012 IEEE International Conference on Robotics and Automation*, pages 3852–3858, 2012.

[4] M. Tenorth and M. Beetz. Knowrob — knowledge processing for autonomous personal robots. In *EEE/RSJ Intl. Conf. on Intelligent Robots and Systems.*, pages 4261–4266, 2009.

[5] Yiannis Karayiannidis, Christian Smith, Francisco Vina, Petter Ogren, and Danica Kragic. Model-free robot manipulation of doors and drawers by means of fixed-grasps. pages 4485–4492, 05 2013.

[6] Mario Prats, Angel P. del Pobil, Steven Wieland, Tamim Asfour, and Rüdiger Dillmann. Compliant interaction in household environments by the armar-iii humanoid robot. *2008 8th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2008*, 12 2008.

[7] M. T. Mason. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 11:418–432, 1981.

[8] Rosen Diankov, Siddhartha Srinivasa, Dave Ferguson, and James Kuffner. Manipulation planning with caging grasps. 01 2008.

[9] Robert Cupec, Ivan Vidović, Valentin Šimundić, Petra Pejić, Sergi Foix, and Guillem Alenya. Teaching a robot where doors and drawers are and how to handle them. In *2023 32th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, 2023. Prihvaćen za objavljivanje.

[10] Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollar. Tensormask: A foundation for dense object segmentation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2061–2069, 2019.

[11] Robert Cupec, Damir Filko, and Emmanuel K. Nyarko. Segmentation of depth images into objects based on local and global convexity. In *2017 European Conference on Mobile Robots (ECMR)*, pages 1–7, 2017.

[12] BRUNO SICILIANO and LUIGI VILLANI. *Robot force control*. Kluwer Academic Publishers Boston/Dordrecht/London.

[13] Universal robots, user manual, ur5/cb3. https://www.usna.edu/Users/weaprcon/kutzer/_files/documents/User%20Manual,%20UR5.pdf. Pristup: 15.11.2023.

[14] Robotiq 3-finger adaptive robot gripper, instruction manual. https://assets.robotiq.com/website-assets/support_documents/document/3-Finger_PDF_20190221.pdf. Pristup: 16.11.2023.

[15] Robotiq ft 300-s force torque sensor. https://s3.amazonaws.com/com-robotiq-website-prod-assets/website-assets/support_documents/document/FT300-S_Sensor_Manual_TM_PDF_20210301.pdf. Pristup: 16.11.2022.

[16] Intel's new realsense camera packs a lidar sensor for enhanched depth perception. https://venturebeat.com/ai/intels-new-realsense-camera-packs-a-lidar-sensor-for-enhanced-depth-perception/. Pristup: 17.6.2023.

[17] Ros system. https://www.ros.org/. Pristup: 17.6.2023.

[18] Što je docker i za kojih se 6 glavnih stvari koristi? https://dir.hr/sto-je-docker/. Pristup: 18.6.2023.

[19] What is a ply file? https://docs.fileformat.com/3d/ply/. Pristup: 22.6.2023.

[20] The urscript programming language for e-series. https://s3-eu-west-1.amazonaws.com/ur-support-site/115824/scriptManual_SW5.11.pdf. Pristup: 24.6.2023.

[21] Jana Dukić, Lukrecia Vulić, Valentin Šimundić, Petra Pejić, and Robert Cupec. Opening of doors and drawers by a ur5 robot with force control. U postupku recenzija, 2023.

SAŽETAK

U ovom diplomskom radu predstavljena je metoda za otvaranje i zatvaranje ladica robotskom rukom koja se temelji na računanju globalne trajektorije pomoću računalnog vida te upravljanju silom i pozicijom tijekom robotske manipulacije. Metode su implementirane i ispitane na šesto-osnom robotskom manipulatoru UR5 s robotskom hvataljkom Robotiq 3-Finger Gripper. Za detekciju modela ladice koriste se metode računalnog vida iz programske biblioteke RVL koje procesiraju slike dobivene Intel RealSense LiDAR L515 kamerom pričvršćenom pri vrhu alata robotskog manipulatora. Na temelju modela ladice i međusobnog položaja dijelova robotskog manipulatora i ladice, računa se trajektorija otvaranja i zatvaranja ladice. Upravljanje silom i pozicijom ostvareno je čitanjem vrijednosti s ugrađenog senzora sile i momenta FT 300-S. Kako bi se osiguralo robusno manipuliranje ladicom, iznos sile se minimizira u lateralnim smjerovima, dok se upravljanje pozicijom odvija u smjeru otvaranja i zatvaranja ladica. Na ovaj način, osigurane su adaptivne korekcije lokalne trajektorije otvaranja i zatvaranja. Predloženo rješenje sastoji se od tri faze: 1. detekcija modela ladice temeljem snimljene sekvence RGB i dubinskih slika ljudske demonstracije otvaranja ladice, 2. računanje položaja za hvatanje i hvatanje ručke robotskom hvataljkom te 3. robotska akcija otvaranja i zatvaranja ladice upravljana po sili i poziciji po izračunatoj trajektoriji. Eksperimentalnom evaluacijom utvrđenja je uspješnost izvođenja akcije otvaranja i zatvaranja ladice od 62.5%. Ustanovljeno je da detekcija modela ovisi o položaju kamere u trenutku snimanja slike ladice na kojoj se provodi detekcija, a najčešće greške događale su se uslijed krive detekcije modela, odnosno pogreške u dijelu programa temeljenom na računalnom vidu. Sustav je implementiran i ispitan u Ubuntu operacijskom sustavu, ROS Noetic distribuciji i unutar Docker kontejnera.

Ključne riječi:: otvaranje/zatvaranje ladice, robotska manipulacija, upravljanje silom, robotski vid, ROS

ABSTRACT

This thesis presents a method for opening and closing drawers with a robotic arm based on the calculation of global trajectory using computer vision and the force and position control during robot manipulation. The methods were implemented and tested on a 6DoF UR5 robotic manipulator with a Robotiq 3-finger gripper. Computer vision methods from the RVL program library are used to detect the drawer model, processing images obtained from an Intel RealSense LiDAR L515 camera attached to the robot manipulator end effector. Based on the model of the drawer and the mutual position of the parts of the robot manipulator and the drawer, the trajectory for opening and closing the drawer is calculated. Force and position control is performed by reading the values of the built-in force and torque sensor FT 300-S. To ensure robust manipulation of the drawer, the force is minimized in the lateral directions, while position control is performed in the direction of opening and closing the drawer. In this way, adaptive corrections of the local opening and closing trajectory are ensured. The proposed solution consists of three phases: 1. recognition of the drawer model based on the recorded RGB-D image sequence of a human demonstration of opening the drawer, 2. calculation of the position for grasping the handle with a robot gripper, and 3. force and position controlled robot action of opening and closing the drawer, according to the calculated trajectory. The experimental evaluation showed that the success of the action of opening and closing the drawer was 62.5%. It was found that the recognition of the model depends on the position of the camera at the time of capturing the image of the drawer on which the recognition is performed, and that the most frequent errors occurred due to incorrect recognition of the model, i.e. due to an error in the part of the program based on computer vision. The system was implemented and tested in the Ubuntu operating system, the ROS Noetic distribution and within the Docker container.

Keywords: opening/closing of drawers, robotic manipulation, force control, robot vision, ROS

ŽIVOTOPIS

Jana Dukić rođena je 1.9.1999. godine u Osijeku. Osnovnoškolsko obrazovanje završava 2014. godine u Osnovnoj školi Ivana Filipovića u Osijeku. Iste godine upisuje III. Gimnaziju Osijek te 2018. godine obrazovanje nastavlja na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, gdje upisuje preddiplmski studij računarstva. Diplomski studij upisuje 2021. godine, smjer Robotika i umjetna inteligencija. Tijekom studija, dobitnica je nagrade za uspješnost u studiranju, nagrade za postignut uspjeh u izvannastavnim aktivnostima, dvije nagrade za nagrađeni studentski rad, Rektorove nagrade i Dekanove nagrade. Također, koautor je u pisanju radova na konferencijama: "Real Fluid Simulation for Determination of Engine Oil Viscosity" (OTO 2021), "Generating Artificial Biscuit Tile Images" (OTO 2021), te "Ramifications of the greenhouse effect: a closed vehicle example" (SST 2022).

PRILOG NA CD-U

1. Diplomski rad u PDF formatu
2. Programski kod
3. Prikupljeni podaci eksperimentalnom analizom