Otvaranje ormara robotskom rukom pomoću računalnog vida i upravljanja silom

Vulić, Lukrecia

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

Diplomski sveučilišni studij Računarstvo
Izborni blok Robotika i umjetna inteligencija

Otvaranje ormara robotskom rukom pomoću računalnog vida i upravljanja silom

Diplomski rad

Lukrecia Vulić

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:	Lukrecia Vulić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	D-1257R, 07.10.2021.
OIB studenta:	76672914644
Mentor:	prof. dr. sc. Robert Cupec
Sumentor:	Valentin Šimundić, mag. ing. comp.
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	izv. prof. dr. sc. Emmanuel-Karlo Nyarko
Član Povjerenstva 1:	prof. dr. sc. Robert Cupec
Član Povjerenstva 2:	doc. dr. sc. Petra Pejić
Naslov diplomskog rada:	Otvaranje ormara 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 vrata ormara i ormarića. Na temelju snimke čovjeka kako otvara vrata ormara, program treba odrediti veličinu, položaj i smjer otvaranja vrata ormara te na temelju tih podataka isplanirati trajektoriju za robotsku ruku. Nakon toga, robot treba otvoriti i zatvoriti vrata ormara koristeći senzor sile radi korekcije planirane trajektorije na način da se minimiziraju sile kojima robot djeluje na ormar. Tema rezervirana za: Jana Dukić ili Lukrecia Vulić Sumentor s FERIT-a: Valentin Šimundić
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, 14.07.2023.

Ime i prezime studenta:	Lukrecia Vulić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1257R, 07.10.2021.
Turnitin podudaranje [%]:	6

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

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

i sumentora Valentin Šimundić, mag. ing. comp.

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

Stalan napredak i razvoj područja automatizacije i robotike doveo je do olakšanja radnih poslova, posebice onih opasnih po ljudski život. U početku je fokus bio na poslovima u kojima su robotski manipulatori obavljali opasne zadatke u tvornicama, ali u današnjici se automatiziraju poslovi u ljudskoj svakodnevici primjenom robota koji svojim radnjama ne mogu ozlijediti čovjeka i okolinu. Time se olakšava svakodnevni život u kućanstvima i institucijama poput bolnica.

Jedan od takvih svakodnevnih poslova je i otvaranje vrata radi prolaska u drugu prostoriju ili dohvaćanja predmeta iz ormara. U posljednjih se nekoliko godina ubrzano razvijaju algoritmi za detekciju vrata i automatiziranu robotsku manipulaciju njihova otvaranja. Razvoj efikasnih algoritama za otvaranje vrata ima veliki značaj u napretku i daljnjem razvoju industrije, povećanju pristupačnosti osobama s invaliditetom, pomoći u kućanstvima i općenitom unaprjeđenju robotike. Algoritmi za otvaranje vrata trebaju biti sigurni za osobe i predmete u okolini robota, za namještaj na kojem se obavlja akcija otvaranja te za sam robotski manipulator. Do ozljeda i oštećenja može doći uslijed djelovanja prekomjerne sile robotskog manipulatora tijekom manipulacije vratima. Stoga, ovakav robotski sustav treba prepoznati vrata koja treba otvoriti, ustanoviti os rotacije i smjer njihova otvaranja, izračunati trajektoriju otvaranja te je provesti uz istovremenu kontrolu djelovanja sile na vrata.

U ovom radu razmatra se tema otvaranja vrata robotskom rukom primjenom računalnog vida i upravljanja silom. Sustav razvijen u ovom radu izgrađuje model koji određuje položaj i smjer otvaranja vrata na temelju sekvence slika koje prikazuju njihovo otvaranje. Nadalje, robotski se manipulator pozicionira ispred ručke ormara te se računa trajektorija otvaranja vrata. Zbog moguće pogreške prilikom izrade modela, zbog šuma kamere, ili nedovoljno točne informacije o poziciji robota, izračunata trajektorija ponekad je neprecizna. Iz tog se razloga tijekom gibanja robotske ruke zadanom trajektorijom primjenjuje upravljanje po sili koje ispravlja nepreciznost trajektorije i osigurava glatko i neometano otvaranje ormara.

Rad je strukturiran na sljedeći način. U drugom poglavlju dan je pregled razvijenih metoda za otvaranje vrata. U trećem poglavlju daje se pregled metoda računalnog vida za detekciju vrata i metode izračuna putanje otvaranja vrata korištenih u ovom diplomskom radu. Tehnologije i razvojni sustavi korišteni u testiranju ispravnosti razvijenog rješenja opisani su u četvrtom poglavlju. U petom poglavlju opisuje se implementacija programskih rješenja za metode opisane u trećem poglavlju. Sve komponente rješenja obuhvaćene su šestim poglavljem, u kojem su predstavljene eksperimentalna evaluacija i analiza rezultata. Zaključak je dan u sedmom poglavlju.

2. PREGLED PODRUČJA METODA OTVARANJA VRATA

U ovom poglavlju daje se pregled literature o metodama otvaranja vrata u robotici. Analiziraju se postojeća rješenja te se ističu potencijalni problemi u metodama i načini primjene.

U radu *Robust and adaptive door operation with a mobile robot* [1], predstavljena je metoda za robusno i adaptivno otvaranje vrata, korištenjem autonomnog mobilnog manipulatora. Spomenuti rad sadržava i metodu za procjenu položaja i točke hvatanja za ručke na vratima na temelju RGB-D slika. Kinematički model vrata izgrađuje se pomoću Bayesovog okvira na temelju promatranog gibanja vrata te učenju iz prethodnih iskustava i ljudske demonstracije. Planiranje kretanja, odnosno izvršavanja putanje, obavljeno je pomoću Task Space Region (TSR) algoritma koji objedinjuje prikaz strategije zadovoljavanja zadanih ograničenja i planiranja putanje temeljene na uzorkovanju. Zadatak je podijeljen na pet podzadataka i to su: detektiranje ručke na vratima, hvatanje ručke, otključavanje ručke, estimiranje kinematičkog modela vrata i planiranje i izvršavanje putanje otvaranja vrata. Eksperimentalna evaluacija obavila se korištenjem mobilnog manipulatora Toyota HSR koji ima četiri stupnja slobode te kao alat koristi hvataljku s dva prsta. Također su korišteni 6-osni senzor sile i RGB-D kamera. Kod testiranja detekcije ručke na vratima i hvatanja ručke, fokus je stavljen na brzinu i točnost. Točnost se ocjenjuje računanjem srednje prosječne preciznosti (engl. *Mean Average Precision*, mAP) na testnom skupu koji se sastoji od različitih vrsta vrata i ručki. Performansa modela za prepoznavanje ručki, s uspjehom od 45%, pokazao se 10% lošijim od modela dobivenih najsuvremenijim detektorima objekata u skupovima podataka visoke kvalitete. Nisu poznate verzije ovih detektora, ali s njima su postignute mAP vrijednosti od 55% i 58%. Informacije vezane uz brzinu izvođenja ove akcije nisu dostupne u radu. Testiranje modela u laboratoriju također je bilo uspješno. U svakom pokusu u kojem je bio generiran točan model ručke, robotski manipulator bi uspio uhvatiti ručku ako se ručka nalazila u radnom prostoru robota. Za testiranje estimacije kinematičkog modela vrata bez ikakovog prethodnog znanja, korištena su tri tipa vrata. Zadatak se izvodio tako da robot uhvati ručku i otvori vrata dok istovremeno uči kinematički model vrata. Manipulacija je uspješno izvedena u 87% slučajeva. Prilikom neuspješnih slučajeva, greška nije bila u modelu, već u konstrukciji alata robota, korištena je hvataljka s dva prsta, koja nije prikladna za ovu vrstu manipulacije.

Cilj istraživanja rada *Learning Force Control Policies for Compliant Manipulation* [2] svodi se na razvijanje robota za obavljanje osjetljivih i preciznih zadatka u nestrukturiranoj okolini na ispravan i siguran način. Predstavljeni pristup uči manipulaciju za zadatke koje obavljaju kolaborativni roboti kroz podržano učenje. Ovime se omogućava ostvarenje robusne strategije za obavljanje zadataka manipulacije uz upravljanje silom. Korišten je PI^2 (engl. *Policy Improvement with Path Integrals*) algoritam za učenje principa upravljanja silom kroz optimizaciju kriterijske funkcije koja mjeri uspješnost obavljenog zadatka. Predloženi pristup uči o silama i zakretnim momentima koje treba kontrolirati u vrhu alata u kombinaciji s kinematičkom putanjom. Ovo omogućava izravnu kontrolu fizičkih veličina potrebnih za zadatke manipulacije i rukovanje pogreškama u procjeni položaja i stanja. Postupak uključuje pružanje početne kinestetičke demonstracije zadatka. Kinestetička demonstracija predstavlja postupak u kojem korisnik fizički demonstrira ili vodi robota kroz pokrete zadatka tako da izravno mijenja položaj robota. Početna kinestetička demonstracija pruža željenu poziciju i orijentaciju alata, a početne vrijednosti sile i zakretnog momenta su postavljene na nulu. Optimizacijski algoritam ovisi o postojanju početne demonstracije na temelju koje bi mogao optimizirati vrijednosti sila i zakretnih momenata. Testiranje ovog pristupa upravljanja silom obavljeno je na zadacima otvaranja vrata i podizanja olovke. U pokusima je korišten Barrett WAM manipulator koji ima sedam stupnjeva slobode, Barrettova hvataljka s tri prsta i 6-osni senzor sile i momenta. Za mjerenje uspješnosti, na ručku je postavljen senzor za mjerenje inercije (engl. *Inertial Me-*

asurement Unit, IMU), iz čijeg mjerenja se može izračunati kut otključavanja ručke u odnosu na vrata. Snimljena je željena trajektorija za pritiskanje kvake i otvaranje vrata te se iz njih izvlači inicijalna kriterijska funkcija. PI^2 algoritam koristi se za učenje potrebnih dimenzija sila i zakretnih momenata. Obavljeno je 110 pokusa za slučaj s vratima prilikom kojih je model učio, te su postignute vrijednosti pri kojima se, nakon učenja, zadatak obavlja s točnosti od 100%. U slučaju pokusa podizanja olovke, također je postignuta točnost od 100%, no za ovaj postupak učenja bilo je potrebno obaviti 21 pokus. Tijekom procesa učenja i testiranja nije provjerena robusnost algoritma na položaj objekata. Koristio se isti položaj objekta za manipulaciju tijekom cijelog procesa učenja i testiranja.

Autori rada *Door Opening by joining Reinforcement Learning and Intelligent Control* [3] predlažu način poboljšanja brzine učenja i adaptacije kod podržanog učenja korištenjem manjeg opsega pretraživanja. Predložena je metoda pretraživanja vođene inteligentnim algoritmom PI^2. U slučaju da prilikom primjene ove metode tražene vrijednosti ne konvergiraju tj. divergiraju, sustav se vraća na klasičnu metodu pretraživanja. Korištenjem klasičnih metoda pretraživanja prostor rješenja pretražuje se nasumično, ne uzimajući u obzir strukturu ili svojstava problema. Metodom predloženom u radu koristi se podržano učenje u cilju autonomnog učenje principa otvaranja vrata. Inicijalne pretpostavke su da je poznat položaj ručke na vratima i način na koji se hvata, a načini pritiska ručke i otvaranja vrata su riješeni metodom opisanom u radu. Predložena metoda testirana je i u simulaciji i korištenjem stvarnog robota. Kao simulacijsko okruženje korišten je MuJoCo HAPTIX i robotski manipulator KUKA LWR 4. Učinkovitost opisane metode ocjenjivala se na nekoliko različitih pokusa: otvaranje vrata sa standardnom kvakom, otvaranje vrata gdje se ručka otključava podizanjem gore, otvaranje ladice koja se otključava podizanjem ručke prema gore i otvaranje vrata čija je os otvaranja horizontalna. Razlika kod otvaranja vrata i ladica je u varijabli koja se zadaje: kod vrata se zadaje kut otvaranja, a kod ladice duljina otvaranja. Na sve slučajeve je primijenjen isti algoritam i u svim slučajevima zadatak je uspješno obavljen primjenjujući samo silu. U prosjeku je za izvlačenje ladice bilo potrebno da robot 10 puta provede izvlačenje ladice da nauči princip primjene sila, a za ostale slučajeve bilo je potrebno šest otvaranja. Metoda je testirana u laboratoriju korištenjem KUKA LWR 4 robotskog manipulatora, otvaranjem vrata s desne strane, te je za ovaj slučaj za učenje trebalo biti izvedeno devet otvaranja vrata.

Radom *Adaptive Force/Velocity control for opening unknown doors* [4] predlaže se dinamičko upravljanje silom i brzinom koji koristi adaptivnu procjenu radijalnog smjera na temelju adaptivnih procjena položaja osi rotacije vrata. Upravljačko djelovanje se rastavlja na radijalne i tangencijalne smjerove, koji su se pokazali u skladu s odgovarajućim stvarnim vrijednostima. Regulator sile koristi reaktivnu kompenzaciju tangencijalnih sila, regulirajući radijalnu silu na željenu vrijednost. Pokazano je da adaptivne procjene radijalnog smjera konvergiraju prema stvarnom radijalnom vektoru, a konvergencija radijalne sile i tangencijalne brzine prema željenim vrijednostima je analitički dokazana. Regulator brzine osigurava da se vrh alata robota pomiče željenom tangencijalnom brzinom. Eksperimentalna evaluacija metode obavljena je u simulaciji. Korišten je robotski manipulator CAS/KTH sa sedam stupnjeva slobode, u kojem su se zakočili određeni zglobovi tako da ima dva stupnja slobode. Također je korišten 6-osni senzor sile i momenta. Pogreška sile i procjene konvergiraju u nulu u prvih 0.5 sekundi. Uspješno su estimirani potrebni zakretni momenti koji se zadaju zglobovima robota.

Chen i drugi u radu *Robust adaptive position and force tracking control strategy for door-opening behaviour* [5] predlažu adaptivni algoritam za upravljanje silom i pozicijom kako bi se potisnule prekomjerne unutarnje sile i osigurao pravilan položaj mobilnog manipulatora pod holonomnim i neholonomnim ograničenjima. Algoritam je dizajniran kako bi se pratili željeni položaj i sila prilikom otvaranja vrata, izbjegavajući kompleksnost fleksibilnih mehanizama i nepredvidljivosti kontaktne krutosti koja se javlja u tradicionalnom upravljanju impedancijom. U radu su predstavljeni rezultati dvije simulacije kako bi se potvrdila učinkovitost metode.

Prvo je odrađen numerički primjer u kojem je naglasak na teoretskoj analizi dinamike mobilnog manipulatora. Ovo je odrađeno u Matlab/Simulink okruženju. Predložena metoda upravljanja uspoređena je s modelom koji se oslanja na matematički model vrata. Odabrane su željene trajektorije u radnom prostoru mobilnog manipulatora, unutar sigurnosnih granica robota, kako bi se izbjegli sudari s vratima. Adaptivno upravljanje kod izvršavanja zadane trajektorije imalo je manje greške, nego upravljanje modelom. Adaptivnim upravljanjem također se osigurava da rotacijska brzina zgloba prati rotacijsku brzinu kvake te se ovom metodom postiže glatko otvaranje vrata. Pokus se odradio i na konkretnom mobilnom robotu s manipulatorom s pet stupnjeva slobode. U pokusu su primijećene velike oscilacije koje se mogu pripisati lošem položaju za hvatanje kvake. Dobiveni rezultati se razlikuju od onih dobivenih u simulaciji zbog pogrešaka u modeliranju i nepreciznih senzora. Prilikom izvođenja pokusa u trenutku pritiska kvake prisutne su velike oscilacije vrijednosti sile uzrokovane promjenom kontaktne sile. Najveće razlike između simulacije i pokusa događaju se prilikom hvatanja kvake. Prilikom postupka otvaranja vrata su također prisutne pogreške uzrokovane manjkom stupnjeva slobode manipulatora što rezultira lošim položajem manipulatora prilikom postupka otvaranja vrata. Rezultati simulacije i pokusa pokazuju da je predložena metoda robusna u modeliranju grešaka, trenja zglobova, vanjskih smetnji i ispunjava potrebne uvjete za otvaranje vrata s ručkom i suzbijanja prekomjernih unutarnjih sila.

3. METODOLOGIJA

U ovom poglavlju detaljno su opisane metode i algoritmi koji su korišteni za rješavanje problema otvaranja i zatvaranja vrata u okviru ovog diplomskog rada. Prvi korak u sustavu je primjena algoritma računalnog vida za detekciju i izgradnju modela vrata. Ovaj algoritam analizira sekvencu slika ljudske demonstracije otvaranja vrata i na temelju njih konstruira model vrata. Nakon izgradnje modela, koristeći poznate odnose dijelova sustava, izračunavaju se točke trajektorije. Ove točke određuju putanju kojom će se robotski manipulator kretati prilikom otvaranja vrata. Konačno, trajektorija se izvodi primjenom principa upravljanja po sili. Upravljanje po sili omogućuje kontrolu sile koja djeluje na vrata tijekom otvaranja kako bi se osiguralo sigurno i pravilno otvaranje vrata. Ovaj princip smanjuje rizik od oštećenja sustava i omogućuje precizno izvršavanje otvaranja vrata.

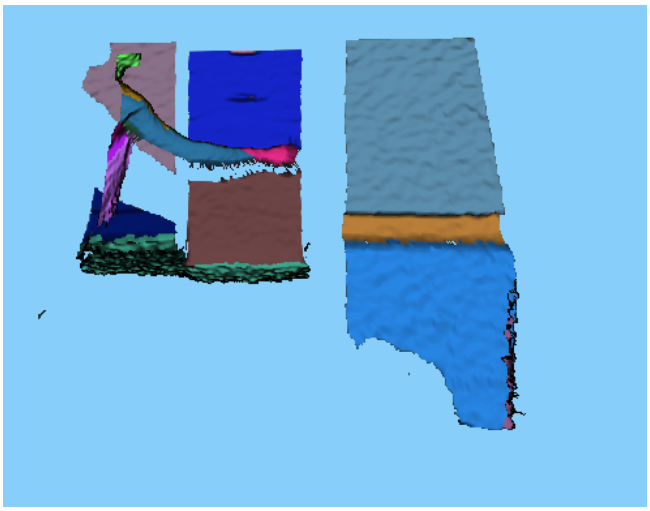
3.1. Algoritam za detekciju i izgradnju modela vrata

Ovo potpoglavlje opisuje ključne komponente metode za detekciju i izgradnju modela vrata, koje uključuju obradu slike, ekstrakciju značajki i izgradnju modela. Točno izgrađen model vrata ključan je za uspješnost obavljanja robotske akcije otvaranja i zatvaranja vrata.

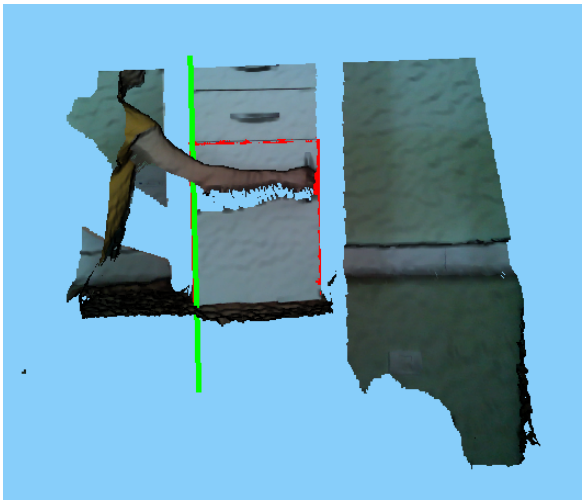
Korišten je pristup opisan u radu *Teaching a Robot Where Doors and Drawers Are and How To Handle Them*. Pristup se sastoji od dva algoritma, ali u ovom radu je korišten samo prvi algoritam. Prvi algoritam, DDD-THD (engl. *Teaching by Human Demonstration*) [6], služi za detekciju vrata na temelju sekvence RGB-D slika na kojima je prikazana ljudska demonstracija otvaranja vrata. Sekvenca je snimana statičnom dubinskom kamerom kako bi jedini pomični dijelovi na slikama bili čovjek i pomični dio vrata, a ostatak scene ne mijenja položaj u odnosu na kameru. U idućem koraku izvornog algoritma, uklanjaju se točke na slici koje pripadaju čovjeku. Na ovaj način se osigurava da, u idealnom slučaju, sve točke na sceni koje mijenjaju svoj položaj u odnosu na kameru pripadaju vratima. U ovom je diplomskom radu osigurana minimalna vidljivost čovjeka na sceni tijekom demonstracije te je ovaj korak preskočen, jer program za detekciju čovjeka zahtjeva računalne resurse koje računalo korišteno za izvođenje pokusa ne podržava. Algoritam generira jednu ili više hipoteza o pokretnim dijelovima (engl. *moving part hypothesis*, MPH) za svaku sliku i integrira te hipoteze kroz sekvencu kako bi formirao konačnu hipotezu o vratima, odnosno ladicama (engl. *door/drawer hypothesis*, DH). Svakoj hipotezi se dodjeljuje vrijednost pouzdanosti, a detektirani model se stvara na temelju hipoteze s najvišom vrijednosti.

Hipoteze pomičnih dijelova generirane su za svaku sliku u sekvenci slika tako da se scena prvo dijeli na ravninske segmente (engl. *planar patch*) prikazane na slici 3.1a. Usporedba slike sa svim slikama u unaprijed definiranoom vremenskom okviru u sekvenci u domeni vektora pomaka rezultira detekcijom pokretnih ravninskih segmenata te, u konačnici, pokretnih dijelova. Vektori pomaka dobiveni su usporedbom slika u obliku oblaka točaka, tako da razlika između međusobno najbližih točaka tih dvaju oblaka točaka predstavlja vektor pomaka. Središte grupiranih vektora pomaka naziva se dominantni vektor pomaka. Ravninski segment s najvećim brojem točaka čiji su vektori pomaka dovoljno slični dominantnom vektoru pomaka smatra se prednjim dijelom pokretnog dijela. Taj ravninski segment se zatim se koristi za generiranje hipoteze o pokretnom dijelu. Prikaz detekcije pokretnog djela prikazan je slikom 3.1b.

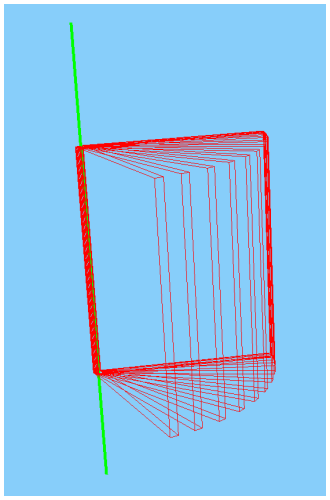
Na temelju hipoteze pokretnih dijelova računaju se hipoteze vrata i ladica. Model vrata, prikazan slikom 3.2, sastoji se od pokretnog dijela vrata koji je povezan s osi oko koje se vrata otvaraju i zatvaraju, prikazane isprekidanom plavom linijom na slici 3.2. Hipoteza vrata predstavljena je uređenim skupom $H = (\eta, {}^{C}T_A, s, r, o, \theta)$, gdje η predstavlja tip detektiranog objekta, što mogu biti vrata ili ladica. Položaj koordinatnog sustava vrata, S_A, koji se nalazi



(a) *Podjela scene na ravninske segmente*



(b) *Detektirana vrata (uokvirena crveno)*



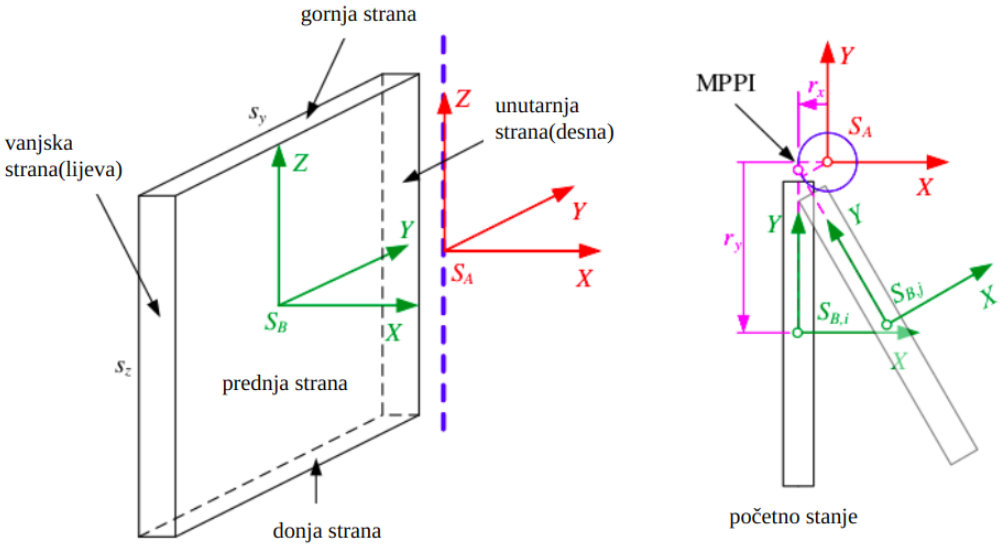
(c) *Pomični dio tijekom sekvence otvaranja vrata*

Slika 3.1: *Postupak detekcije vrata na temelju sekvence slika*

u središtu osi rotacije otvaranja vrata, u odnosu na koordinatni sustav kamere, S_C, opisan je matricom homogene transformacije ${}^{C}T_A$. Nadalje, dimenzije pomičnih dijelova definirane su vektorom s koji ima dvije komponente, s_y i s_z, koje predstavljaju dimenzije u y i z smjerovima S_A koordinatnog sustava. Relativna udaljenost između koordinatnih sustava S_A i S_B opisana je vektorom r koji se sastoji od pomaka r_x i r_y. Parametar o predstavlja smjer otvaranja, a θ stanje, koje se u slučaju vrata odnosni na kut otvorenosti na danoj slici. Koordinatni sustav S_B ima ishodište u središtu pomičnog dijela, a osi su mu paralelne sa stranicama tog dijela. Ishodišta koordinatnih sustava S_A i S_B leže u istoj horizontalnoj ravnini. Integracija hipoteza pomičnih dijelova u hipotezu vrata provedena je hijerarhijskim postupkom grupiranja [6]. Hipoteze pomičnog dijela prilikom sekvence otvaranja vrata prikazane su na slici 3.1c.

3.2. Izračun putanje otvaranja/zatvaranja vrata

Putanja vrata računa se na temelju matrica homogene transformacije. Matricama homogene transformacije opisani su prostorni odnosi između pojedinih dijelova robotskog manipulatora i vrata. U narednim potpoglavljima opisuju se matematički temelji matrice homogene tran-



Slika 3.2: *Model vrata [6]*

sformacije i njihova primjena u izračunu putanje robotske ruke tijekom otvaranja i zatvaranja vrata.

3.2.1 Matrice homogene transformacije

Matrica homogene transformacije T dimenzija 4x4, dana izrazom 3-1, predstavlja prikaz orijentacije i pozicije tijela u prostoru, gdje je orijentacija predstavljena rotacijskom matricom R dimenzija 3x3, a pozicija translacijskim vektorom t dimenzija 3x1 [7].

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{3-1}$$

Matrica homogene transformacije koristi se, osim za predstavljanje orijentacije i pozicije tijela u prostoru, za promjenu referentnog koordinatnog sustava u kojem se neki vektor nalazi. Ako je poznata međusobna rotacija i translacija dvaju koordinatnih sustava te je poznat položaj vektora u jednom od tih koordinatnih sustava, množenjem vektora s matricom homogene transformacije taj vektor može se prikazati u drugom koordinatnom sustavu.

Neka svojstva matrica homogene transformacije, bitna za daljni rad, su sljedeća:

1. Inverz matrice homogene transformacije iz prvog koordinatnog sustava u drugi je matrica homogene transformacije iz drugog koordinatnog sustava u prvi

$$ {}^{B}T_{A}{}^{-1} = {}^{A}T_{B} \tag{3-2}$$

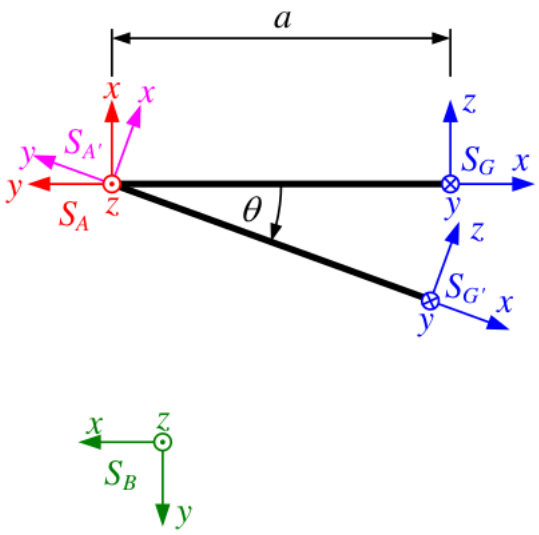
2. Produkt dvije matrice homogene transformacije je ulančana matrica homogene transformacije

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

3.2.2 Računanje putanje

U ovom potpoglavlju opisana je metoda za izračun putanje vrata temeljena na zadanom sustavu koordinatnih sustava i matricama homogene transformacije. Sustav koordinatnih sustava postavljen je na slici 3.3. Koordinatni sustav vrata S_A, postavljen je tako da njegova z-os

predstavlja os rotacije oko koje se vrata otvaraju i zatvaraju. S_B predstavlja koordinatni sustav baze robota, S_G predstavlja koordinatni sustav vrha robotske hvataljke i S_T predstavlja koordinatni sustav šestog zgloba robotskog manipulatora (engl. *tool center point*, TCP). Kut θ predstavlja kut otvaranja, odnosno zatvaranja vrata, dok varijabla a predstavlja udaljenost između osi rotacije vrata i pozicije ručke.



Slika 3.3: *Međusobni odnos koordinatnih sustava vrata i dijelova robotskog manipulatora*

Pretpostavlja se da je početna pozicija manipulatora prilikom ovog izračuna točno ispred ručke na ormaru, te je hvataljka okomita na vrata. Matricom homogene transformacije ${}^{B}T_{T}$ predstavljen je položaj koordinatnog sustava alata robota S_T, u odnosu na koordinatni sustav baze robota S_B. Međusobni položaj ova dva koordinatna sustava moguće je zadati kotroleru robota te kontroler robota na zahtijev korisnika može dati trenutno važeću informaciju o tom položaju.

Rezultat izračuna putanje je sljedeći položaj u koji robotska hvataljka treba doći kako bi vrata bila otvorena za kut θ. Ovo je predstavljeno matricom ${}^{B}T_{T'}$, te iz slike 3.3 vrijedi izraz 3-4.

$$ {}^{B}T_{T'} = {}^{B}T_{T}\,{}^{T}T_{G}\,{}^{A}T_{G}{}^{-1}\,{}^{A}T_{A'}\,{}^{A'}T_{G'}\,{}^{T'}T_{G'}{}^{-1} \tag{3-4} $$

Međusobni položaj koordinatnih sustava S_A i S_G tijekom otvaranja ili zatvaranja vrata ostaje nepromijenjen pa vrijedi izraz

$$ {}^{A'}T_{G'} = {}^{A}T_{G} \tag{3-5} $$

Slično vrijedi i za međusobni položaj koordinatnog sustava TCP-a S_T i robotske hvataljke S_G jer je hvataljka pričvršćena na TCP

$$ {}^{T'}T_{G'} = {}^{T}T_{G} \tag{3-6} $$

Ustanovivši prethodno navedene odnose, izraz 3-4 može se zapisati na način prikazan sljedećom jednadžbom

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

Matricu homogene transformacije ${}^{A}T_{G}$ moguće je zapisati na temelju poznatih međusobnih rotacija osi koordinatnih sustava S_A i S_G i njihove međusobne udaljenosti na temelju slike 3.3 te je njihov odnos zapisan jednadžbom 3-8.

$$ {}^{A}T_{G} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & -a \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3-8} $$

Međusobni odnos koordinatnih sustava S_A i $S_{A'}$ je takav da je koordinatni sustav $S_{A'}$ rotiran za kut θ po z-osi koordinatnog sustava S_A, kako je prikazano izrazom 3-9.

$$ {}^{A}T_{A'} = \left[\begin{array}{ccc|c} & R_z(\theta) & & 0 \\ & & & 0 \\ & & & b \\ \hline & 0 & & 1 \end{array}\right] \tag{3-9} $$

Matrica ${}^{T}T_{G}$ opisuje međusobni položaj koordinatnih sustava S_T i S_G, prikazanih slikom 3.3 i opisana je izrazom 3-10. Ovaj odnos je konstantan te je računicu potrebno provest samo jednom. Međusobnu poziciju prethodno spomenutih koordinatnih sustava jednostavno je odrediti. Koordinatni sustav S_G je u odnosu na koordinatni sustav S_T pomaknut po z-osi koordinatnog sustava S_T za iznos b koji je moguće ručno izmjeriti. Pošto orijentaciju ${}^{T}R_{G}$ nije moguće mjeriti, računa se na temelju poznatih rotacijskih matrica ${}^{B}R_{T}$ i ${}^{B}R_{G}$, na način prikazan izrazom 3-11. Hvataljka je dovedena u orijentaciju opisanu izrazom 3-12 koja opisuje međusobnu orijentaciju S_G i S_B koordinatnih sustava. Nakon što je manipulator postavljen u ovu orijentaciju, očitan je trenutni položaj manipulatora na temelju kojeg je formirana matrica ${}^{B}T_{T}$ iz koje je moguće očitati rotacijsku matricu ${}^{B}R_{T}$.

$$ {}^{T}T_{G} = \left[\begin{array}{ccc|c} & {}^{T}R_{G} & & 0 \\ & & & 0 \\ & & & b \\ \hline & 0 & & 1 \end{array}\right] \tag{3-10} $$

$$ {}^{T}R_{G} = {}^{B}R_{T}{}^{-1}{}^{B}R_{G} \tag{3-11} $$

$$ {}^{B}R_{G} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{3-12} $$

3.2.3 Računanje početnog položaja putanje

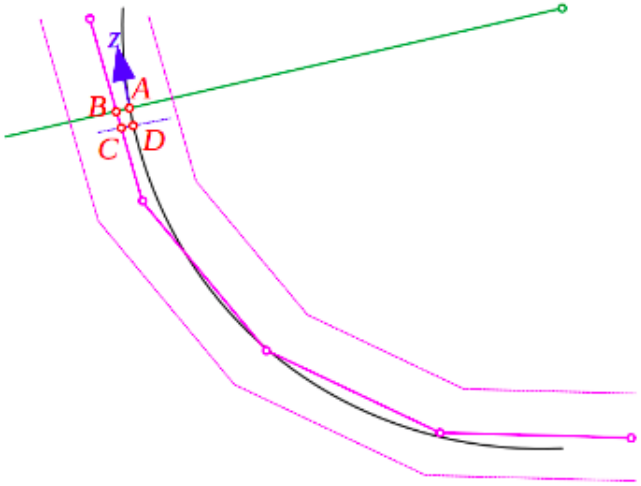
Računanje početnog položaja manipulatora ima ključnu ulogu u daljnjem izvođenju zadatka otvaranja i zatvaranja vrata. Početnim položajem se smatra položaj za hvatanje ručke, opisan matricom ${}^{B}T_{T'}$. Matrica ${}^{B}T_{T'}$ odnosi se na matricu ${}^{B}T_{T}$ u izračunu putanje opisanom u poglavlju 3.2.2. Početni položaj putanje računa se izrazom

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

Podrazumijeva se da je poznat model vrata te da je poznata matrica homogene transformacije ${}^{T}T_{C}$ koja opisuje položaje koordinatnih sustava kamere S_C i TCP-a S_T dobivena kalibracijom kamere. Pretpostavlja se da je kamera pričvršćena pri vrhu alata robotskog manipulatora, u takozvanoj *eye-in-hand* konfiguraciji. Matrica ${}^{C}T_{A}$ dobivena je iz modela vrata dobivenog pomoću računalnog vida, a matrica ${}^{B}T_{T}$ dobivena je očitavanjem položaja TCP-a u odnosu na bazu robota u položaju u kojemu je snimljen model vrata. Postupak za izračun matrica ${}^{T}T_{G}$ i ${}^{A}T_{G}$ detaljno je opisan u poglavlju 3.2.2.

3.3. Upravljanje silom i pozicijom

Postizanje precizne kontrole pozicije i sile u robotici od iznimne je važnosti kod interakcije s nepredvidivom okolinom i obavljanja osjetljivih zadataka. Upravljanje koja kombinira istovremenu sposobnost upravljanja silom i pozicijom još se naziva i hibridna kontrola. Hibridni pristup upravljanju primjenjuje se u zadacima u kojima je kretanje robota djelomično ograničen u nekom smjeru, te su jedni stupnjevi slobode sustava upravljani po poziciji, a drugi po sili [8]. Ovakav princip mora riješiti sljedeće probleme: kontrola pozicije manipulatora duž smjerova u kojima postoji prirodno ograničenje sile, kontrola sile manipulatora duž smjerova u kojima postoji prirodna ograničenost položaja, implementacija proizvoljnog miješanja ovih načina kontroliranja duž ortogonalnih stupnjeva slobode proizvoljnog okvira [8]. Drugim riječima, ova strategija kontrolira poziciju u smjeru slobodne kretnje, a silu u smjeru fizičkih ograničenja zadatka.



Slika 3.4: *Ispravak trajektorije otvaranje vrata hibridnim upravljanjem*

U slučaju otvaranja i zatvaranja vrata, smjer bez fizičkih ograničenje bila bi os u smjeru otvaranja i zatvaranja vrata, tj. z-os koordinatnog sustava S_G prikazanog na slici 3.3. Smjerovi kod kojih postoji ograničenje leže na osima x i y koordinatnog sustava S_G te se u tim smjerovima pokušava minimizirati sila kojom djeluje manipulator. Ciljana vrijednost sile u smjerovima koji imaju ograničenje je 0 N . U smjeru z-osi upravlja se pozicijom, na temelju točaka trajektorije izračunatih kako je opisano u poglavlju 3.2. Na ovaj način moguće je ispravljati trajektoriju koja sadrži nepreciznosti uzrokovane pogreškama mjerenja senzora ili zaokruživanja kod izračuna.

Slika 3.4 prikazuje princip kojim je moguće izvršiti idealnu putanju otvaranja i zatvaranja vrata upravljanjem po sili i poziciji. Crna linija prikazuje idealnu putanju, onu pri kojoj su sile u fizički ograničenim smjerovima 0 N. Čistim upravljanjem pozicijom, ova putanja nikada se ne bi mogla postići zbog nepreciznosti mjerenja senzora i numeričkih pogrešaka tijekom računanja, ali korištenjem upravljanja po sili nepreciznu putanju, prikazanu ružičastom linijom, moguće je ispraviti. Točka A predstavlja jedan uzorak idealne putanje, a točka B pripadajući uzorak na izračunatoj putanji. Obje točke predstavljaju poziciju vrha alata, odnosno koordinatnog sustava S_G. Između ove dvije točke postoji određeno odstupanje uzrokovano prethodno navedenim razlozima te bi upravljanjem samo po poziciji u smjerovima x i y koordinatnog sustava S_G potencijalno došlo do primjene prekomjernih sila tijekom izvođenju izračunate putanje. Uvođenjem upravljanja po sili u smjerovima ograničenog kretanja te upravljanja po poziciji

samo u smjeru slobodnog kretanja moguće je ispraviti ovu razlike. Naime, upravljanjem po poziciji u smjeru kretanja osigurava se pomak potreban za otvaranje i zatvaranje vrata, a upravljanjem po sili u ograničenom smjerovima osigurava se da se pozicija u tom smjeru ispravi na onu idealne putanje jer je jedino na idealnoj putanji sila u tim smjerovima jednaka 0 N. Preostale dvije ružičaste linije na slici predstavljaju dopušteno odstupanje od izračunate putanje.

4. EKSPERIMENTALNI POSTAV I RAZVOJNI ALATI

U ovom poglavlju cilj je pružiti uvid u stvarni robotski sustav na kojem su implementirane i testirane opisane metode detekcije i otvaranja, odnosno zatvaranja vrata te u pritom korištene razvojne alate, ističući njihove funkcionalnosti, prednosti i potencijalna ograničenja. Robotski sustav sastoji se od robotskog manipulatora UR5 CB3, robotske hvataljke Robotiq 3-Finger Adaptive Robot Gripper, senzora sile i moment FT 300-S Sensor i dubinske kamere Intel RealSense LiDAR Camera L515. Alati korišteni pri implementaciji programskog rješenja su ROS i Docker.

4.1. Robotski manipulator UR5 CB3

UR5 robot je industrijski robot kojeg proizvodi tvrtka Universal Robots, dio CB3 serije. Korišten je za automatiziranje raznih zadataka poput rukovanja alatom i nepomičnim objektima te za manipulaciju proizvodima manjih dimenzija. UR5 također spada u kategoriju kolaborativnih robota, što znači da se robot može pokretati u blizini ljudi, bez ikakvih fizičkih pregrada, bez straha od ozljeda i oštećenja. Sastoji se od šest robotskih zglobova i dvije aluminijske cijevi koje su spojene na bazu, kao što je vidljivo na slici 4.5 [9].

Slika 4.5: *Robotski manipulator UR5 [9]*

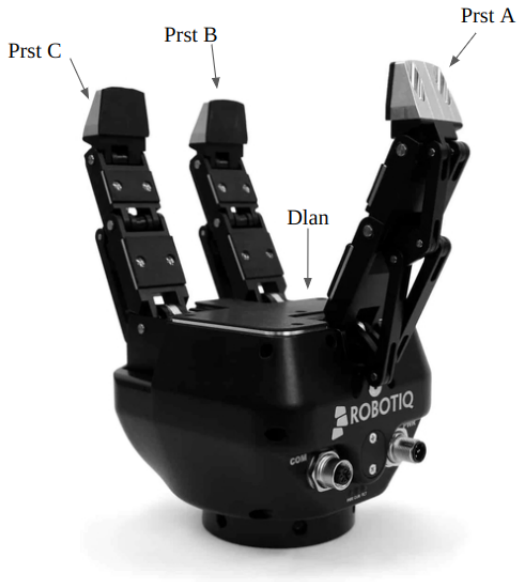
Robot UR5 jednostavno se pri bazi pričvršćuje na radnu površinu. Ovaj robot ima šest rotirajućih osi, što mu omogućuje fleksibilno kretanje u različitim smjerovima. UR5 robot ima prosječne dimenzije u usporedbi s drugim robotima iste kategorije. Visina robota iznosi 950 mm, a doseg mu je 850 mm. U pogledu mase, UR5 se svrstava u lakšu kategoriju s težinom od 18.4 kg.

Za programirenje i upravljanje UR5 robotom koristi se PolyScope. PolyScope je grafičko korisničko sučelje koje omogućava upravljanje robotom i kontrolerom te izvršavanje programa. Polyscope omogućava upravljanje robotom u *freedrive* načinu rada, kojim korisnik može ručno postaviti robota u željeni položaj [10]. Upravljanje je moguće na razini Polyscopa i na skriptnoj

razini korištenjem programskog jezika URscript [11]. Programiranje na skriptnoj razini obavlja se slanjem korisničkog programa s osobnog računala, korištenjem TCP/IP protokola, na UR-Control. URControl je niskorazinski robotski kontroler koji se izvršava na *Mini-ITX* računalu u upravljačkoj kutiji robota [11]. URsript uključuje gotove funkcije, a jedna od njih je *force mode*. Force mode je ugrađena funkcija UR robota koja omogućava upravljanje po sili prilikom obavljanja raznih zadataka. Moguće je regulirati silu i zakretne momente u x, y i z smjerovima, u granicama zadanih vrijednosti [11].

4.2. Robotska hvataljka Robotiq 3-Finger Adaptive Robot Gripper

Robotiq 3-Finger Adaptive Robot Gripper je robotska hvataljka namijenjena za industrijsku primjenu, ali koristi se i u istraživanju i razvoju. Sastoji se od tri prsta A, B i C, od kojih svaki ima tri zgloba kao što je vidljivo na slici 4.6. Svaki prst moguće je zasebno kontrolirati što pruža sposobnost dohvaćanja objekata pri različitim kutevima i orijentacijama. Također, svaki prst ima ugrađen senzor sile, kojim se regulira pritisak prilikom dohvaćanja objekta. Prilagodba oblicima različitih objekata moguća je zbog manjeg broja motora u odnosu na broj zglobova, [12]. Ova hvataljka ima mogućnost prilagodbe hvata objektima različitih oblika, dimenzija i materijala [12]. *Operation mode* je vrsta omogućenih pokreta hvataljke, kojima se definira orijentacija prstiju B i C, te je ovaj pokret određen od strane korisnika na temelju procjene oblika i dimenzije objekta kojeg treba uhvatiti. Postoje četiri vrste hvata i to su: *basic mode*, *wide mode*, *pinch mode* i *scissor mode*. Drugi način upravljanja hvataljkom odnosi se na zatvaranje hvataljke i postoje dvije vrste: *fingerprints grip* i *encompassing grip*. U ovom načinu rada nije moguće kontrolirati svaki prst zasebno. Hvataljka se zatvara dok ne postigne stabilno stanje, oko objekta ili oko "dlana" hvataljke kada je hvataljka potpuno zatvorena. Korisnički se može upravljati relativnom brzinom kojom se zatvara i zadavati relativna sila koja se primjenjuje na objekt prilikom zatvaranja [12].



Slika 4.6: *Robotska hvataljka Robotiq 3-Finger Adaptive Robot Gripper [12]*

4.3. Senzor sile i momenta FT 300-S

FT 300-S je senzor sile i momenta tvrtke Robotiq, prikazan slikom 4.7. Namijenjen za mjerenje sile i zakretnog momenta u robotskim sustavima. Senzor mjeri sile pri vrhu alata robota i postavlja se između TCP-a robota i alata. FT 300-S je industrijski senzor primjenjiv u poslovima poput montaže, rukovanja materijalom i poliranja. Senzor očitava vrijednosti sila u stvarnom vremenu što omogućava preciznije obavljanje osjetljivih zadataka te mogućnost adaptacije robotskih manipulacija u zadanim granicama. Očitavaju se sile u x, y i z smjeru te je opseg mjerenja na svakoj osi ± 300 [N], a zakretnih momenata ± 30 [Nm]. Softver *Force copilot* uključuje već gotove funkcije za lakše upravljanje silom i zakretnim momentom.

Slika 4.7: *Senzor sile i momenta FT 300-S [13]*

4.4. Dubinska kamera Intel RealSense LiDAR L515

Intel RealSense LiDAR L515 je kamera za dubinsko snimanje u boji koja koristi LiDAR tehnologiju za snimanje dubine. Koristi laserske zrake za mjerenje dubine tako što ispusti lasersku zraku te mjeri vrijeme do povratka spomenute zrake koju detektira fotodioda. Pogodna je za integraciju u razne sustave zbog svojih visokih performansi i kompaktne dimenzije. Ima raspon od 9 [m] i točnost mjerenja dubine manju od 5 [mm] na udaljenosti 1 [m] i manju od 14 [mm] na udaljenosti 9 [m] [14].

Slika 4.8: *Dubinska kamera Intel RealSense LiDAR L515 [15]*

4.5. ROS

ROS (*Robot Operating System*) je metaoperacijski sustav otvorenog koda koji se koristi za razvoj robotskih sustava. Kao i standardni operacijski sustav uključuje apstrakciju hardvera, upravljanje uređajima na niskoj razini, implementaciju često korištenih funkcionalnosti, komunikaciju između procesa razmjenom poruka i upravljanje paketima. Također pruža alate i programske biblioteke [16]. ROS prati filozofiju Unix operacijskih sustava. Jedan od filozofskih aspekata ROS-a je *peer to peer* način komunikacije. ROS sustav čini veći broj manjih programa koji su međusobno povezani razmjenom poruka, direktno jedan drugome, bez centralne jedinice koja ih preusmjerava. Prednosti ovakvog pristupa vide se u sustavima gdje postoji velika količina razmjene podataka. ROS nema standardno integrirano razvojno i izvršno okruženje. Zadaci poput navigacije, vizualizacije, generiranja dokumentacije ili bilježenja podataka obavljaju se pomoću zasebnih programa. Kod pisanja korisničkih programa moguće je koristiti različite programske jezike poput C++, Python, LISP, Java, JavaScript, MATLAB, Ruby, Haskell, R i Julia [17].

Povezivanje čvorova (engl. *node*) u ROS-u obavlja se pomoću servisa (engl. *service*) naziva *roscore* koji pruža informacije potrebne za povezivanje s čvorovima, kako bi se mogle dogoditi razmjene poruka (engl. *message*). *Roscore* pruža informacije za međusobni pronalazak čvorova. Prilikom pokretanja svih potrebnih čvorova, svaki od njih se registrira u *rosecore*-u pružajući informacije o tome koji tip poruke pruža i koji tip poruke želi primati [17].

Za pisanje i izvršavanje programskog koda u ROS-u potrebno je osigurati radno okruženje (engl. *workspace*), koje se definira kao sustav direktorija koji sadrži potrebne dijelove za izvršavanje korisničkog koda te se kreiraju korištenjem alata *catkin*. *Workspace* direktorij organiziran je u pakete (engl. *package*), gdje se svaki paket sastoji od programskog koda, podataka i dokumentacije. Svaki paket sadrži dokumente koji opisuju sadržaj paketa i način na koji *catkin* treba rukovati tim paketom. Za izvršavanje korisničkih programa dostupan je alat *rosrun* kojim se pretražuje *workspace* za odgovarajući paket i izvršni program zadan naredbom te je obavezno prethodno pokretanje čvora *roscore*. *Roslaunch* je također alat korišten za pokretanje programa, a razlika je u tome što se pomoću *roslaunch*-a može pokrenuti veći broj čvorova odjednom, a čvor *roscore* se automatski pokreće. Najčešći način razmjene podataka između ROS čvorova je preko tema (engl. *topic*), koje se definiraju kao slijed poruka određenog tipa. Ovaj tip komunikacije implementira razmjenu poruka na temelju mehanizama objavljivanja (engl. *publish*) i pretplaćivanja (engl. *subscribe*). Čvorovi mogu objavljivati na temu, što znači da odašilju podatke određenog tipa na temu određenog imena. Prije objavljivanja mora se definirati ime teme na koju će objavljivati i tip podatka poruka koji se objavljuje. Na drugoj strani ove razmjene su čvorovi koji se pretplaćuju na teme, koji također definiraju ime teme s koje žele čitati podatke i tip podatka koji se čita.

4.6. Docker

Docker je platforma otvorenog koda za razvoj, prijenos i pokretanje aplikacija [18]. Korištenjem Dockera omogućava se virtualizacija aplikacija u odnosu na infrastrukturu računala te je moguće upakirati aplikaciju u jednu cjelinu zajedno s operacijskim sustavom. Jedan takav objekt naziva se slika (engl. *image*) i predstavlja predložak s instrukcijama za stvaranje Docker kontejnera (engl. *container*). U dokumentu naziva *Dockerfile* sadržane su upute za izgradnju i pokretanje slika. Slika se sastoji od slojeva koji se stvaraju izvršavanjem linija u Dockerfile-u, te su memorijski nezahtjevne. Docker kontejner je instanca slike koja se može izvršavati. Korištenjem ovog mehanizma osigurava se izolacija aplikacije od *host* računala te omogućava pokretanje aplikacije sa svim potrebnim instalacijama na drugom računalu.

5. IMPLEMENTACIJA ALGORITAMA

U ovom poglavlju opisuje se implementacija metoda opisanih u poglavlju 3. Algoritmi su implementirani u programskim jezicima Python i URscript.

5.1. Algoritam za detektiranje i izgradnju modela vrata

5.1.1 Algoritam za snimanje slika

Za izgradnju modela potrebna je sekvence slika ljudske demonstracije otvaranja vrata. Algoritam za snimanje slika implementiran je u obliku ROS čvora, prikazanog kodom 1, koji je opisan u nastavku, a za što je korištena *rospy* biblioteka. Snimanje slika omogućeno je *OpenCV* bibliotekom. Potrebno je snimiti sekvencu istovremeno uslikanih RGB i dubinskih slika. RGB i dubinske slike dobivene su pretplaćivanjem na teme ROS čvora kamere koje objavljuju poruke sadržavajući ove tipove slika. To su teme `/camera/color/image_raw` i `/camera/aligned_depth_to_color/image_raw`. Za istovremeno čitanje oba tipa poruka korištena je biblioteka `message_ filters` uz sinkronizaciju ulaznih kanala po vremenskim oznakama uporabom *Time Synchronizer*. Ulazni kanali su pretplatnici na teme za RGB i dubinske slike. Snimanje slika u razvijenom programu započinje i završava pritiskom tipke *space* i slike se spremaju u zasebne direktorije za kasniju upotrebu. Prilikom snimanja slika, također se sprema položaj robotskog manipulatora iz kojeg je snimljena slika scene na kojoj se nalaze vrata ormara, korištenjem `ur_rtde` biblioteke.

Kod 1: *Postavljanje ROS čvora za snimanje slika*

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

5.1.2 Algoritam za izgradnju modela vrata

Kako bi bilo moguće izgraditi model vrata, najprije se moraju generirati PLY datoteke. PLY datoteka (eng. *Polygon File Format*) predstavlja 3D format datoteke koji pohranjuje grafičke objekte opisane kao skupine poligona. Ove datoteke stvorene su na temelju RGB i dubinskih slika. Model se izgrađuje na temelju stvorenih PLY datoteka i RGB slika. Detektor, opisan u poglavlju 3.1 i radu *Teaching a Robot Where Doors and Drawers Are and How To Handle Them*, implementiran je metodom iz RVL biblioteke, `RVLPYDDDetector`, s predefiniranom konfiguracijom. Ulazni podaci za detektor su RGB slike ljudske demonstracije otvaranja vrata i PLY datoteke. Detektor u poseban dokument zapisuje sve generirane hipoteze i vraća onu koja se kriterijskom funkcijom pokazala najboljom. Krajnji rezultat je opis modela, ako je pronađen, opisan u poglavlju 3.1. Model se sprema u tekstualnu datoteku koja se koristi za kasniju upotrebu kod računanja putanje otvaranja vrata.

5.2. Algoritam za otvaranje/zatvaranje vrata robotskom rukom UR5

U sljedećim potpoglavljima su opisani dijelovi algoritma otvaranja i zatvaranja vrata.

5.2.1 Izračun transformacijske matrice

Položaj robota predstavljen je 6-dimenzionalnim vektorom, čije prve tri vrijednosti predstavljaju poziciju TCP-a u radnom prostoru robota, a posljednje tri orijentaciju u obliku rotacijskog vektora. Rotacijski vektor predstavljaja rotaciju objekta oko tri osi Kartezijevog koordinatnog sustava. Ovakav prikaz položaja naziva se *prikaz u prostoru alata*, i koristi se u narednim izračunima. Funkcija `get_transformation_matrix_from_pose_vector`, prikazana kodom 2, kao parametar prima vektor položaja i kao povratnu vrijednost računa matricu homogene transformacije. Duljina rotacijskog vektora predstavlja kut rotacije u radijanima i označena je kao θ . U slučaju da je normirana vrijednost zadanog praga ($1e^{-6}$), matrica rotacija računa se koristeći Rodriguesovu formulu za rotaciju. Vektor k predstavlja jedinični vektor koji predstavlja os rotacije, izračunat kao omjer zadanog rotacijskog vektora i njegove normirane vrijednosti.

Kod 2: *Izračun transformacijske matrice*

```
def get_transformation_matrix_from_pose_vector(vec):
    mat = np.identity(4)
    rot_vector = vec[3:].copy()
    th = np.linalg.norm(rot_vector)
    if np.abs(th) > 1e-6:
        k = rot_vector / th
        cs = np.cos(th)
        sn = np.sin(th)
        a = 1.0 - cs
        rot_obj_temp = np.array([[k[0]*k[0]*a + cs, k[1]*k[0]*a-k[2]*sn, k[2]*k[0]*a
            ↪ +k[1]*sn],
                                 [k[0]*k[1]*a+k[2]*sn, k[1]*k[1]*a + cs, k[2]*k[1]*a-k[0]*sn],
                                 [k[0]*k[2]*a-k[1]*sn, k[1]*k[2]*a+k[0]*sn, k[2]*k[2]*a + cs]])
    else:
        rot_obj_temp = np.identity(3)
    mat[0:3, 0:3] = rot_obj_temp.copy()
    mat[0:3, 3] = vec[0:3].copy()
    return mat
```

U slučaju da je vrijednost kuta rotacije ispod zadanog praga, odnosno da nema značajne promjene, za rotacijsku matricu uzima se jedinična matrica. Naposljetku se izračunata rotacijska matrica i položaj, očitan kao prve tri vrijednosti vektora položaja, slažu u rotacijsku matricu opisanu u poglavlju 3.2.1. Funkcijom `get_pose_vector_from_transformation_matrix` računa se vektor položaja u prostoru alata na temelju matrice homogene transformacije.

5.2.2 Izračun matrice ${}^{T}T_{G}$

Matrica homogene transformacije ${}^{T}T_{G}$ predstavlja odnos koordinatnog sustava šake, S_G, u odnosu na koordinatni sustav TCP-a, S_T, kako je prikazano na slici 3.3. Kako bi se u daljnjem računanju dobili ispravni rezultati, ovu matricu potrebno je računati na temelju specifičnog položaja šake, pri kojemu je moguće uhvatiti ručku vrata. Ovaj položaj vidljiv je na slici

6.10. Hvataljka mora biti okomita na vrata te se prsti moraju nalaziti sa svake strane ručke. Konkretne vrijednosti ove orijentacije za Eulerove kuteve u stupnjevima su (90°, -45°, 0°). Taj trenutni položaj ujedno predstavlja željenu orijentaciju alata, te se ova matrica s ovim vrijednostima, koristi u daljnjem izračunu točaka putanje. Implementiran je postupak izračuna matrice ${}^{T}T_{G}$ opisan u poglavlju 3.2.2.

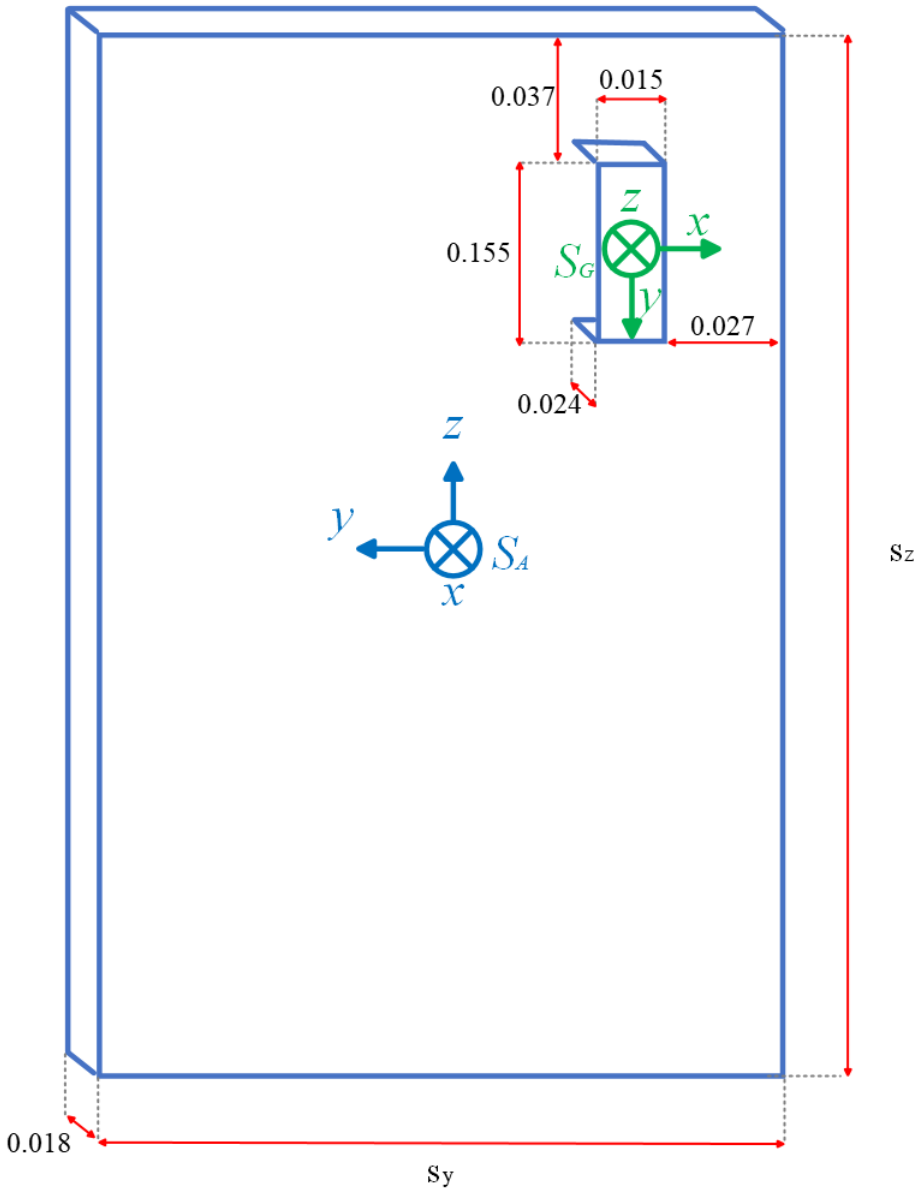
5.2.3 Izračun početnog položaja

Izračun početnog položaja implementiran je funkcijom `calculate_starting_pose` prikazana kodom 3, na temelju postupka opisanog u poglavlju 3.2.3 i izraza 3-13. Matrica ${}^{C}T_{A}$ te vektori s i r, dobiveni su iz modela vrata, izračunatog postupkom opisanim u poglavlju 3.1, te učitani iz tekstualne datoteke pomoću funkcije `extract_door_model`. Matrica ${}^{T}T_{G}$ računa se kako je opisano u poglavlju 5.2.2 te je također učitana iz tekstualne datoteke. Kalibracijom kamere dobivena je matrica ${}^{T}T_{C}$. Matrica ${}^{A}T_{G}$ se implementira na način prikazan kodom 3 linijama 11 do 17. Kod računanja pozicije koordinatnog sustava S_G u odnosu na koordinatni sustav S_A u obzir se mora uzeti pozicija ručke na vratima i pozicija koordinatnog sustava S_A u odnosu na središte vrata koju opisuje vektor r u modelu vrata. Vrijednosti u kodu 3 odnose se na specifična vrata korištena u pokusima, budući da detekcija ručke nije obuhvaćena u ovom radu. Širina i visina vrata dobivene su iz modela i označene su vektorom s, dok je debljina vrata izmjerena. Također su izmjerene širina i visina ručke, kao i njen položaj unutar vrata. Sve ove dimenzije prikazane su na slici 5.9, te su sve vrijednosti na slici izražene u metrima. Koordinatni sustav S_G postavljen je u središte ručke u y i z smjerovima koordinatnog sustava S_A, dok je u x smjeru postavljen na površinu vrata kako bi se osiguralo da hvataljka može čvrsto držati ručku. Vektor položaja robota dobiven je očitanjem trenutnog položaja robota `ur_rtde` bibliotekom funkcijom `getActualTCPPose` te se matrica ${}^{B}T_{T}$ računa na temelju očitanog položaja robota korištenjem funkcije `get_transformation_matrix_from_pose_vector`.

Kod 3: *Implementacija algoritma za izračun početnog položaja*

```
1  def calculate_starting_pose():
2      b = 0.28
3      text_file=open("/home/RVLuser/ur5_ws/src/door_opening/models/doorModel_%d.
           ↪ txt"%model_number,"r")
4      ao=text_file.read()
5      text_file.close()
6      TAC, s, r= extract_door_model(ao)
7      TCT = np.array([[−0.9786901906426425, 0.05770529828634073,
           ↪ 0.03676758168288247, −0.01078821015709552 ],
8                      [−0.0593574405847809, −0.9782635664950653,
                           ↪ −0.044646774674482546, 0.18152850107216661 ],
9                      [ 0.034036021511733676, −0.04676257896325065,
                           ↪ 0.9793727871559502, −0.001666384428069767],
10                     [ 0. , 0., 0. , 1. ]])
11     x_GA = r[0] − 0.018/2. − 0.024 +0.024
12     y_GA = r[1] − s[0]/2. + 0.027 + 0.015/2.
13     z_GA = s[1]/2. − 0.037 − 0.155/2.
14     TGA = np.array([ [0, 0, 1, x_GA],
15                      [−1, 0, 0,y_GA],
16                      [0, −1, 0, z_GA],
17                      [0, 0, 0, 1]])
```

```
    rtde_r = rtde_receive.RTDEReceiveInterface("192.168.22.14")
    tcp_position = rtde_r.getActualTCPPose()
    TTB=get_transformation_matrix_from_pose_vector(tcp_position)
    hf = h5py.File('/home/RVLuser/ur5_ws/src/door_opening/src/TGT.h5', 'r')
    n1 = hf.get('dataset_1')
    TGT = np.array(n1)
    TT_B = TTB @ TCT @ TAC @ TGA @ np.linalg.inv(TGT)
    start_pose=get_pose_vector_from_transformation_matrix(TT_B))
    return start_pose
```



Slika 5.9: *Konstantne vrijednosti dimenzija vrata i ručke*

5.2.4 Izračun točaka putanje

Funkcija `sample_door_opening_points`, prikazana kodom 4, provodi uzorkovanje točaka putanje otvaranja i zatvaranja vrata. Temelji se na poznavanju ${}^{T}T_{G}$ matrice te se pretpostavlja

da je u trenutku početka računanja putanje, robotski manipulator u početnom položaju za otvaranje vrata. Argumenti funkcije su početni položaj robotskog manipulatora, kut otvaranja i broj točaka otvaranja. Kut otvaranja predstavlja kut između točaka putanje između dvije uzastopne scene u sekvenci, a ne cijeli kut otvaranja vrata. Broj točaka otvaranja utječe na to koliko je manipulacija vrata glatka, te predstavljaju sekvencu točaka u koje se robot mora pomaknuti da bi otvorio vrata. Izračun putanje obavlja se rekurzivno, odnosno računa se na temelju prethodno izračunate točke putanje postupkom opisanim u poglavlju 3.2.2 pozivom funkcije `get_next_door_pose`. Kao rezultat funkcije `sample_door_opening_points` dobije se lista točaka putanje otvaranja vrata. Ove točke predstavljaju pozicije vrha alata robotskog manipulatora tijekom otvaranja vrata. Dobivene točke korištene su i za zatvaranje vrata, u obrnutom redoslijedu.

Kod 4: *Implementacija algoritma za izračun točaka putanje*

```
def sample_door_opening_points(start_pose_vec, angle_deg, num_points):
    TTB = get_transformation_matrix_from_pose_vector(start_pose_vec)
    poses = []
    prev_poseTB = TTB
    for i in range(0, num_points):
        TT_B = get_next_door_pose(prev_poseTB, door_width, angle_deg)
        pose_vecT_B = get_pose_vector_from_transformation_matrix(TT_B)
        poses.append(pose_vecT_B)
        prev_poseTB = get_transformation_matrix_from_pose_vector(pose_vecT_B).
            ↪ copy()
    return poses
```

5.2.5 Upravljanje silom i pozicijom

Upravljanje silom i pozicijom implementirano je u URscript programskom jeziku te se koriste ugrađene funkcije za upravljanje silom. Pokretanjem čvora `force_mode` upravljanje silom djeluje na čvorove koji sadrže naredbe za pomicanje robota po točkama putanje otvaranja, odnosno zatvaranja vrata. Funkcija `force_mode` definirana je s pet parametra, kako je prikazano primjerom koda 5, kojima se opisuje način upravljanja silom.

Kod 5: *Funkcija force_mode*

```
force_mode(task_frame, selection_vector, wrench, type, limits)
```

Parametar `task_frame` predstavlja vektor položaja koji definira koordinatni sustav sile u odnosu na bazni koordinatni sustav. Nadalje, `selection_vector` je vektor od šest dimenzija kojim se definira po kojim osima koordinatnog sustava se upravlja po sili i zakretnom momentu. Prve tri vrijednosti opisuju upravljanje silom, a zadnje tri upravljanje zakretnim momentom. Vrijednosti se postavljaju na 0 ili 1, gdje 1 označava da se po toj osi upravlja silom, odnosno zakretnim momentom. Parametar `wrench` sadržava konkretne vrijednosti sila i zakretnih momenata koje se primjenjuju na okolinu, poslagane u skladu sa `selection_vector` parametrom. Parametar `type` opisuje način na koji se interpretira koordinatni sustav. Može poprimiti vrijednosti u rasponu [1,3]. Vrijednost 1 označava transformaciju koordinatnog sustava sile tako da mu je y-os usklađena s vektorom koji pokazuje od TCP-a robota prema ishodištu koordinatnog sustava sile. Vrijednost 2 označava da se koordinatni sustav sile ne transformira, a vrijednost 3 označava da se koordinatni sustav sile transformira tako da mu je x-os projekcija vektora brzine TCP-a robota na x-y ravninu koordinatnog sustava sile. Posljednji parametar, `limits`, je vektor

od šest elemenata čije vrijednosti predstavljaju maksimalnu dozvoljenu brzinu TCP-a duž osi kojima se upravlja. Za preostale osi, vrijednosti vektora predstavljaju maksimalno dozvoljeno odstupanje udaljenosti duž tih osi između stvarne pozicije TCP-a i pozicije postavljene od strane korisnika.

Jedna od ugrađenih funkcija za pomicanje robota je `movej`, prikazana primjerom koda 6, koja opisuje linearni pomak u prostoru zglobova.

Kod 6: *URscript funkcija movej*

```
movej(q, a=1.4, v=1.05, t=0, r =0)
```

Sadrži pet parametara, od kojih su tri obavezna. Parametar q predstavlja vektor vrijednosti zglobova. Vrijednosti zglobova mogu se izračunati na temelju položaja robota u prostoru alata tako da se koristi funkcija za inverznu kinematiku, `get_inverse_kin`. Parametar a predstavlja akceleraciju vodećih osi zglobova zadanu u [rad/s^2], a v brzinu vodećih osi zglobova izraženu u [rad/s]. U slučaju da parametar t nije zadan, pridružena mu je vrijednost 0, a u slučaju da je zadan ima prioritet nad parametrima a i v. Njime se zadaje vrijeme u kojem je potrebno izvršiti naredbu i zadaje se u sekundama. Parametar r opisuje radijus spajanja kojim se osigurava glatki prijelaz između uzastopnih točaka putanje. Vrijednost ovog parametra određuje mjeru u kojoj se primjenjuje efekt spajanja, te ako vrijednost nije zadana, postavlja se u nulu.

Izgled URscript programskog koda za otvaranje vrata dan je primjerom koda 7. `Force mode` čvor mora biti sadržan u funkciji ili niti (engl. *thread*) kako bi se dinamičko upravljanje silom primijenilo u potrebnim trenutcima, kada se otvaraju i zatvaraju vrata. Također, u ovom čvoru sadržane su `movej` funkcije kojima su zadane izračunate točke putanje otvaranja i zatvaranja vrata. Tijekom provedbe testiranja sustava, parametar a postavljen je na vrijednost 1.4 [rad/s^2], a parametar v na 0.2 [rad/s] kako bi operater stigao reagirati na pokrete robota pri kojima bi moglo doći do oštećenja robota, njegovih perifernih dijelova ili namještaja. U slučaju implementacije funkcije za zatvaranje vrata, točke putanje poslagane su obrnutim redoslijedom.

Kod 7: *URscript funkcija za otvaranje vrata*

```
1 def door_open():
2     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])
3         movej(get_inverse_kin(p[−0.2956346618292891, −0.4714747426028585,
          ↪ 0.39932375871706466, 1.3440291838019371, −0.7148264084943261,
          ↪ 0.45162848257653354]),a=1.4,v=0.2)
4         movej(get_inverse_kin(p[−0.23126148357653997, −0.4274231471205734,
          ↪ 0.39932375871706466, 1.2633473563985522, −0.8313740390558433,
          ↪ 0.32116622042369314]),a=1.4,v=0.2)
5         movej(get_inverse_kin(p[−0.16018674594886936, −0.39521594050393727,
          ↪ 0.39932375871706466, 1.1764780014008775, −0.9422857199355544,
          ↪ 0.19014473222456374]),a=1.4,v=0.2)
6 end
7 door_open()
```

Primjer koda 7 predstavlja generalni izgled URscript programskog koda koji se mijenja u ovisnosti o izračunatim točkama putanje. Generiranje ovog koda obavljeno je funkcijom `write_UR_script` koja na temelju izračunate početne pozicije i točaka putanje u tekstualnu datoteku zapisuje programski kod u URscript programskom jeziku. Zapisuju se funkcije za otvaranje i zatvaranje vrata te njihovi pozivi.

URscript programi mogu se slati na upravljači sustav robota URControl, opisan u poglavlju 4.1, liniju po liniju ili cjelovito u obliku definicije funkcije i poziva funkcije. URControl

i osobno računalo ostvaruju komunikaciju preko TCP/IP soketa na portu 30002. Naredbe se šalju korištenjem Python biblioteke `socket`, u kojoj je potrebno definirati port za slanje i IP adresu odredišta slanja, što je u ovom slučaju robot. Opisana komunikacija prilikom poziva funkcije `zero_ftsensor()` prikazana je kodom 8. Svrha funkcije `zero_ftsensor()` je postavljanje vrijednosti sile i zakretnih momenata FT senzora na vrijednost 0 N. Ovom funkcijom trebalo bi se osigurati preciznije upravljanje.

Kod 8: *Slanje naredbe zero_ftsensor*

```
def zeroSensor():
    HOST = "192.168.22.14"
    PORT = 30002
    s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    s.connect((HOST,PORT))
    time.sleep(1)
    s.sendall(("zero_ftsensor()" + "\n").encode('utf−8'))
    time.sleep(2)
    data=s.recv(1024)
    s.close()
```

5.2.6 Upravljanje hvataljkom

Robotskom hvataljkom upravlja se na temelju principa objavljivanja na temu u ROS-u. Koristeći ROS poruke specifične za robotsku hvataljku, opisanu u poglavlju 4.2, moguće je definirati način i silu zatvaranja šake potrebne u svrhu zadatka otvaranja vrata ormara. Funkcije za otvaranje i zatvaranje hvataljke sadržane su u objektu `RobotControl`, pri čijoj inicijalizaciji se definira objavljivanje poruka tipa `Robotiq3FGripperRobotOutput` na temu robotske hvataljke. Spomenutim tipom poruke postavljeni su parametri za aktivaciju robotske hvataljke i za obavljanje zadanog pokreta (engl. *request position*). Parametri brzine, sile i stupnja zatvaranja opisuju se u rasponu [0,255], gdje 0 predstavlja najmanju moguću vrijednost sile i brzine, te potpuno otvorenu poziciju šake. Također se mogu definirati željeni `Operation mode` opisan u poglavlju 4.2. Funkcija za zatvaranje šake, `closeGripper`, prikazana je kodom 9.

Kod 9: *Funkcija za zatvaranje šake*

```
def closeGripper(self):
    command = Robotiq3FGripperRobotOutput()
    command.rACT = 1 # activation
    command.rGTO = 1 # request position
    command.rSPA = 255 # closing speed
    command.rFRA = 255 # force
    command.rATR = 0 # automatic opening if the force is reached (normal = 0)
    command.rMOD = 1 # gripper mode −> 0 = normal, 1 = pinch, 2 = wide, 3
        ↪ = scissors (needs checking)
    command.rPRA = 255 # 0 = open, 255 = close
    # 3 second delay to let the gripper close
    start_time = time.time()
    while True:
        self.pub.publish(command)
        rospy.sleep(0.1)
```

```
15              end_time = time.time()
16              if float(end_time - start_time) >= 3.0: # 3s
17                  break
```

5.2.7 Upravljački program

Povezivanjem prethodno opisanih dijelova dobiven je konačni program, sadržan u prilogu diplomskog rada, koji se izvršava s ciljem obavljanja zadatka otvaranja i zatvaranja vrata. Prvi je korak, dakle, postavljanje manipulatora u položaj snimanja modela. Položaj se učitava iz tekstualne datoteke funkcijom `read_camera_position`, spremljene tijekom snimanja i obrade slika, a robot se pomiče u taj položaj slanjem URscript naredbe `movej`. Nadalje, šalje se naredba za otvaranje robotske šake. Tada započinje izračun početnog položaja manipulatora opisan poglavljem 3.2.3 funkcijom `calculate_starting_position`. Manipulator se u početni položaj postavlja korištenjem naredbe `movej`. Prilikom dolaska u početni položaj, poziva se URscript funkcija `zero_ftsensor` i zatvara se šaka koja pritom hvata ručku vrata ormara. Obavlja se izračun točaka putanje na temelju zadanih parametara te se konačno poziva URscript funkcija za otvaranje ili zatvaranje vrata, prikazana primjerom koda 7. Po završetku sekvence otvaranja ili zatvaranja vrata, otvara se robotska šaka. Cijeli je postupak prikazan u kodu 10.

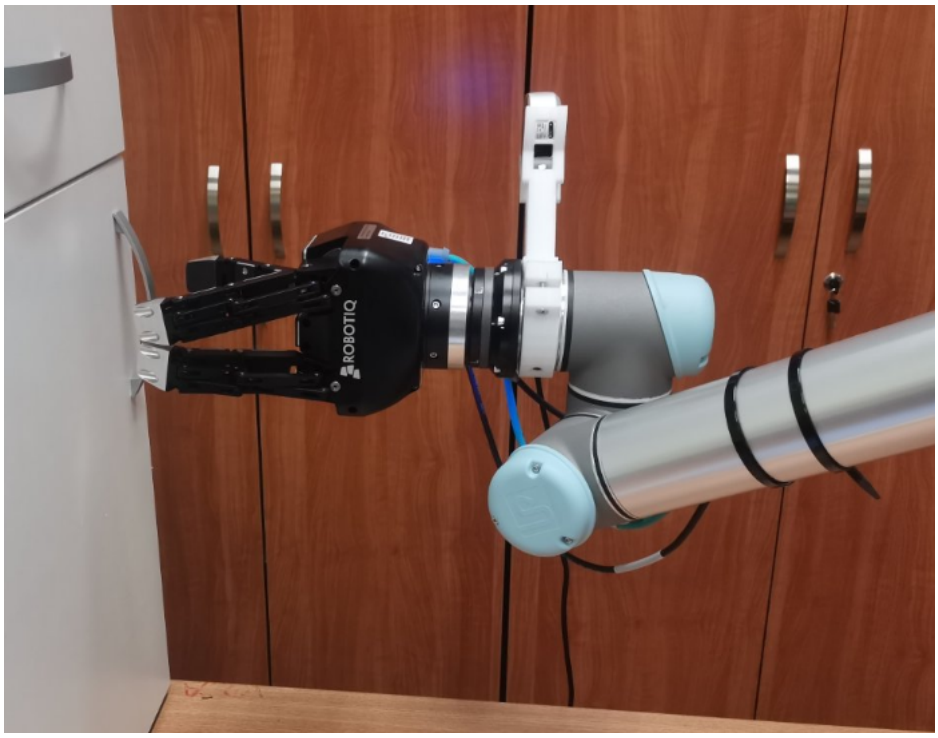
Kod 10: *Cjelovita implementacija postupka otvaranja/zatvaranja vrta*

```
1  if __name__ == '__main__':
2      start_pose,joints=read_camera_position()
3      send_movej(start_pose)
4      wait_to_reach_pose()
5      RC = RobotControl()
6      RC.openGripper()
7      time.sleep(1)
8      start_pose=calculate_starting_position()
9      send_movej(start_pose)
10     wait_to_reach_pose()
11     zeroSensor()
12     time.sleep(1)
13     RC.closeGripper()
14     time.sleep(2)
15     poses = sample_door_opening_points_and_joints(start_pose,angle_deg=-10,
           ↪ num_points=3)
16     write_UR_script(poses, start_pose)
17     sendURscript()
18     wait_to_reach_pose()
19     RC.openGripper()
```

6. EKSPERIMENTALNA ANALIZA

6.1. Implementacija algoritama na stvarnom robotskom sustavu

Hardverski postav eksperimenta sastoji se od komponenti opisanih u poglavlju 4: UR5 CB3 robotski manipulator, robotska hvataljka Robotiq 3-finger Adaptive Robot Griper, senzor sile i momenta FT 300-S senzor i dubinska kamera Intel RealSense LiDAR Camera L515. Baza robotskog manipulatora čvrsto je pričvršćena na radni stol. FT 300-S senzor postavljen je na TCP robota, dok je hvataljka Robotiq 3-finger Adaptive Robot Griper montirana na njega, kao što je prikazano na slici 6.10. Intel RealSense LiDAR Camera L515 kamera postavljena je na stalak pri vrhu alata, vidljiv na slici 6.10.



Slika 6.10: *Vrh alata robotskog manipulatora s montiranom hvataljkom, FT senzorom i kamerom*

Algoritmi opisani u poglavlju 5 implementirani su na sljedeći način. Algoritam za snimanje slika implementiran je u ROS Noetic radnom okruženju (engl. *workspace*), izvan Docker kontejnera zbog poteškoća s povezivanjem RealSense kamere u Docker kontejner. Skripta je sadržana u ROS paketu `taking_images`. Kamerom se upravlja pozivanjem *launch* datoteke za povezivanje kamere s osobnim računalom i postavljanjem potrebnih parametara kamere, kako je prikazano primjerom koda 11. Ova se naredba pokreće u jednoj instanci terminala. U drugoj instanci terminala pokreće se skripta `taking_images.py` koja provodi snimanje i spremanje slika ljudske demonstracije otvaranja vrata, kako je prikazano kodom 12. Ovo je ujedno i prvi korak izgradnje modela vrata.

Kod 11: *Pokretanje čvora kamere*

```
$ source devel/setup.bash
$ 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
```

Kod 12: *Pokretanje čvora za snimanje slika*

```
$ source devel/setup.bash
$ rosrun taking_images take_image.py
```

Nakon pokretanja čvorova, na ekranu računala vidljiv je prikaz slike koju snima kamera te je potrebno namjestiti kameru tako da je vidljiv objekt koji se želi detektirati, u ovom slučaju vrata. Uz to, također mora biti vidljiv dio horizontalne ravnine na kojoj se nalazi ormar, kako bi se vrata ormara ispravno detektirala. S obzirom da je kamera učvršćena na robotski manipulator u *eye in hand* konfiguraciji, položaj kamere se namješta tako da se robot postavi u *freedrive* način rada pritiskom tipke na Polyscopu. Nakon postavljanja robotskog manipulatora, a samim time i kamere u željeni položaj, pokreće se postupak snimanja slika ljudske demonstracije otvaranja vrata pritiskom tipke `space` na računalu. Tijekom ljudske demonstracije, vrata je potrebno otvoriti za kut od minimalno 45°kako bi detekcija bila uspješna. U prosjeku je snimljena sekvenca od 30 slika tijekom otvaranja vrata, na temelju kojih se izgrađuje model. Primjer dijelova sekvence ljudske demonstracije otvaranja vrata prikazan je slikama 6.11. U zasebnu je datoteku spremljen položaj robota iz kojeg su snimljene slike, u prostoru alata i prostoru zglobova, tim redoslijedom, kako je prikazano primjerom koda 13.

(a) *Slika 17* **(b)** *Slika 25* **(c)** *Slika 29*

Slika 6.11: *Jedan uzorak sekvence ljudske demonstracije otvaranja vrata*

Kod 13: *Primjer zapisa položaja robotskog manipulatora*

```
[−0.045464126409942164, 0.18162867337022484, 0.5340019649878112,
↪ 1.8593661945408198, −0.09303016258033707, −0.21005998562837688]
[−1.8921335379229944, −0.046073261891500294, −1.9484718481646937,
↪ −1.4354155699359339, 1.686860203742981, 3.0072922706604004]
```

Algoritam za izgradnju modela, izračun i izvršavanje putanje implementirani su u ROS radnom okruženju unutar Docker kontejnera naziva `rvl_ur5_detectron2`. Docker kontejner je generiran i pokrenut prije pokretanja ostalih čvorova, na način prikazan kodom 14.

Kod 14: *Pokretanje docker kontejnera*

```
$ ./build_docker.sh
$ ./run_docker.sh
```

Pokretanje skripte za izgradnju modela vrata izvršava se na način prikazan kodom 15. Algoritmom je dobiven model vrata prikazan primjerom koda 16 koji se sastoji od dijelova opisanih u poglavlju 3.1.

Kod 15: *Pokretanje čvora za izgradnju modela*

```
$ source devel/setup.bash
$ rosrun door_opening build_model.py
```

Kod 16: *Primjer zapisa modela vrata*

```
{'object_class': 'door', 'R': array([[−0.1301617 , −0.9913551 , 0.01652608],
   [−0.36828604, 0.03286595, −0.92913145],
   [ 0.92055607, −0.12702365, −0.3693801 ]], dtype=float32), 't': array
      ↪ ([−0.20006205, 0.11687329, 1.0525303 ], dtype=float32), 's': array
      ↪ ([0.38924247, 0.5329771 ], dtype=float32), 'r': array([ 2.5357547e−05,
      ↪ −1.9462124e−01], dtype=float32), 'openingDirection': −1.0}
```

Zbog opterećenja koje uzrokuje postupak izgradnje modela, algoritam za računanje i izvršavanje putanje otvaranja vrata izvodi se u zasebnoj skripti. Za izvršavanje ovog algoritma potrebno je spajanje na robotsku upravljačku kutiju, prikazano primjerom koda 17. U drugom terminalu obavlja se spajanje na robotsku hvataljku Robotiq 3-finger Adaptive Robot Griper, prikazano primjerom koda 18. Konačno, pokrenut je čvor za računanje putanje i izvođenje manipulacije otvaranja i zatvaranja vrata, kako je prikazano kodom 19. Prilikom zadavanja broja točaka i kuta između svake točke u funkciji `sample_door_opening_points` nije moguće zadavati kumulativni kut otvaranja veći od 45°zbog međusobnog položaja robotskog manipulatora i ormarića na kojem se izvodi otvaranje vrata. Naime, zbog ograničenog prostora u kojem su postavljena ova dva objekta, izvođenje otvaranja vrata za veće kuteve neizvedivo je zbog zaštitnih mjera ugrađenih u kontrolu robota. Aktiviranje zaštitnih mjera posljedica je fizičkog ograničenja manipulatora da se pozicionira u točke putanje za veće kuteve otvaranja. UR5 CB3 ulazi u stanje *protective stop* ako detektira sudar svojih dijelova te je u svrhu izbjegavanja ovakve situacije kut otvaranja vrata prilikom svih pokusa postavljen na 30°. Za izračuna matrice ${}^{T}T_{G}$, opisane poglavljem 5.2.2, izmjerena je udaljenost između TCP-a i vrha alata iznosa 0.28 m.

Kod 17: *Pokretanje čvora za spajanje s robotom*

```
$ source devel/setup.bash
$ roslaunch ur_robot_driver ur5_bringup.launch robot_ip:=192.168.22.14
   ↪ kinematics_config:=/home/RVLuser/UR5_calibration.yaml
```

Kod 18: *Pokretanje čvora za spajanje s robotskom hvataljkom*

```
$ source devel/setup.bash
$ rosrun robotiq_3f_gripper_control Robotiq3FGripperTcpNode.py 192.168.22.11
```

Kod 19: *Pokretanje čvora za računanje putanje i izvođenje manipulacije otvaranja i zatvaranja vrata*

```
$ source devel/setup.bash
$ rosrun door_opening door_opening_final_version.py
```

6.2. Rezultati i analiza pokusa

Rezultati pokusa prikazani su u tablici 6.1. Testirane su uspješnost detekcije i izgradnje modela vrata (oznaka `Detekcija` u tablici 6.1) te uspješnost otvaranja i zatvaranja vrata upravljanjem po sili i poziciji (oznaka `Manipulacija` u tablici 6.1). Uspješnost ovih zadataka vrednovana je s 0 ako je zadatak neispravno izvršen ili s 1 za uspješno izvršen zadatak. Kako je za uspješnu manipulaciju nužna ispravna detekcija, u slučajevima kada je model krivo prepoznat oznakom - označena je nemogućnost obavljanja manipulacije. Rezultati eksperimenta mogu se podijeliti u tri kategorije: 1. ispravno obavljena i detekcija i manipulacija, 2. neispravno detektiran model vrata i nemogućnost provođenja manupilacije te 3. nemogućnost obavljanja zadatka zbog ograničenja robotskog manipulatora. U trećem je slučaju nemoguće ocijeniti uspješnost detekcije i izgradnje modela, kao i obavljanja zadatka otvaranja i zatvaranja vrata, pa su za oba zadatka u tablici 6.1 dodijeljene oznake -.

Ukupno su provedena 23 pokusa. Ispravna detekcija vrata, izgradnja modela i uspješno obavljen zadatak otvaranja i zatvaranja vrata dogodio se u 14 od 23 slučaja, tj. u 60.86% slučajeva, kao što je prikazano u tablici 6.2. Eksperiment je postavljen na način da su koordinatni sustavi baze robota, S_B, i osi rotacije vrata, S_A, poravnati po y osi koordinatnog sustava baze. Kamera kojom se snima scena postavljena je na TCP robotskog manipulatora. Ispravni modeli dobiveni su u slučajevima kada je kamera postavljena iznad baze robota. Također, prikladni su bili položaji translatirani za manju vrijednost, lijevo ili desno, po x osi koordinatnog sustava S_B. Kamera je u većini slučajeva bila u ravnini gornje strane vrata, te za manji kut, u odnosu na y os S_B koordinatnog sustava, usmjerena prema dolje. Prilikom snimanja ovih modela u potpunosti je prikazan prednji dio vrata i podloga na kojoj se nalaze vrata. Zbog robusnosti opreme, manje pogreške prilikom pozicioniranja manipulatora u početni položaj nisu negativno utjecale na izvođenje pokusa. Primjeri položaja kamere u odnosu na ormarić za slike kojima je uspješno izgrađen model, prikazani su slikama 6.12, ako hvataljka nije u potpunosti centrirana s obzirom na ručku, već je pomaknuta za nekoliko centimetara lijevo ili desno, u trenutku hvatanja na ručku se pritom vrši veća sila, kao što je vidljivo na slici 6.13. U prvih nekoliko uzoraka uočene se povećane vrijednosti očitane sile te je moguće uočiti trenutak u kojemu se sila počinje regulirati, u trećoj sekundi. Na ovim grafovima također je moguće uočiti regulaciju sila s obzirom na to otvara li robot vrata ili zatvara. Pokret otvaranja odvija se do petnaeste sekunde te je nakon toga vidljiva promjena smjera sile i do kraja prikazanog vremenskog okvira odvija se pokret zatvaranja vrata. Na grafovima je između pokreta otvaranja i zatvaranja kod sila na x i y osi uočena simetrija, ovo su ujedno i smjerovi u kojima je regulirana sila. U x smjeru sila raste prilikom otvaranja, od vrijednosti 5 [N] do 9 [N], a prilikom zatvaranja opada sa vrijednosti 7 [N] do 0 [N]. Vrijednosti u y smjeru prilikom pokreta otvaranja padaju od 10 [N] do -20 [N], a kod zatvaranja rastu u obrnutom rasponu. Opisano ponašanje ukazuje na prilagodbu manipulatora da u ovim smjerovima primjenjuje minimalnu moguću silu. Sile u z smjeru, u kojem se ne regulira sila, vidljiva je promjena smjera u petnaestoj sekundi, prilikom otvaranja sile se kreću u rasponu od -2 [N] do 2 [N], a prilikom zatvaranja u rasponu od -6 [N] do -2 [N].

Pokusi pri kojima je netočno određen model vrata rezultiraju pogrešnim početnim položajem robota u trenutku otvaranja vrata u kojem robot nije u mogućnosti uhvatiti ručku. Pogreške ovog tipa dogodile su se u 21.74% slučajeva. Zadatak otvaranja i zatvaranja vrata u tim slučajevima nije izvršen. Detektirani model u dva je slučaja imao neispravne parametre dimenzija vrata te su se vrata detektirala na način prikazan slikom 6.14. Osim vrata, detektirane su i ladice kao pomični objekt. U preostala tri slučaja, pogreška je uočena tek u trenutku kada se robotski manipulator pozicionirao predaleko od ručke. Slikama 6.15 prikazan je položaj kamere u odnosu na vrata prilikom snimanja slika za izgradnju modela pri kojima se dogodila ova pogreška.

Broj modela	Detekcija	Manipulacija	Razlog neuspjeha
1	1	1	
2	1	1	
3	1	1	
4	1	1	
5	1	1	
6	1	1	
7	0	-	pogrešno izgrađen model
8	1	1	
9	0	-	pogrešno izgrađen model
10	-	-	nije izvršeno
11	-	-	nije izvršeno
12	0	-	pogrešno izgrađen model
13	1	1	
14	1	1	
15	0	-	pogrešno izgrađen model
16	1	1	
17	1	1	
18	-	-	nije izvršeno
19	0	-	promašena početna pozicija
20	1	-	pogrešno izgrađen model
21	1	1	
22	1	1	
23	1	1	

Tablica 6.1: *Prikaz rezultata eksperimenta*

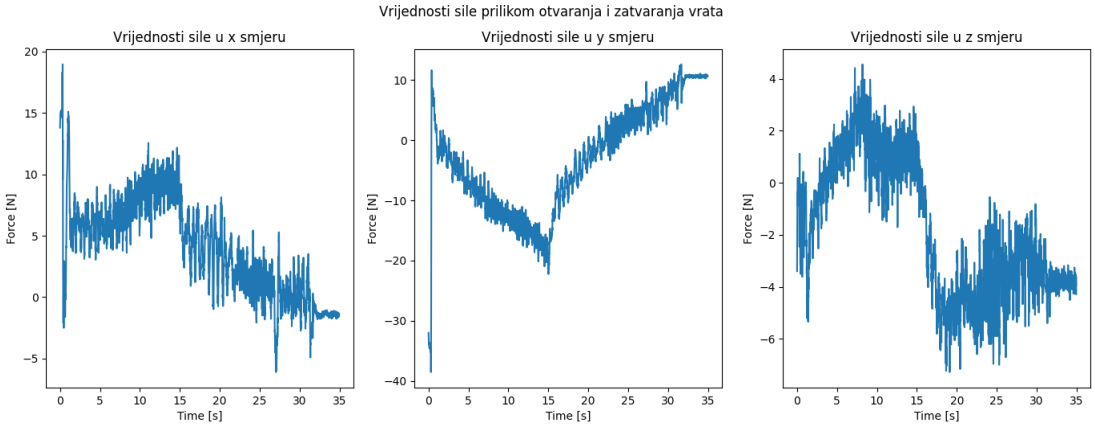
Ukupno	Uspješno	Neuspješno	Neodređeno
23	14	5	4
	60.86%	21.74%	17.39%

Tablica 6.2: *Uspješnost eksperimenata*

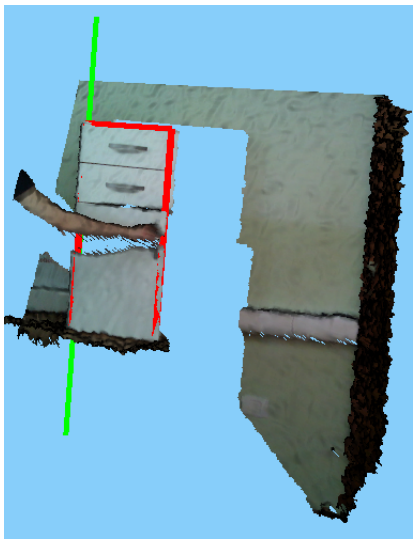
U 4 od 23 slučaja pokus se nije izveo zbog ograničenja robotskog manipulatora. Neki položaji robota pri kojima se obavljala detekcija vrata kao posljedicu su imali nemogućnost

(a) *Model 1* **(b)** *Model 2* **(c)** *Model 3*

Slika 6.12: *Prikaz snimanja uspješno detektiranih modela*



Slika 6.13: *Izmjerene sile u x, y i z smjerovima tijekom otvaranja i zatvaranja vrata*

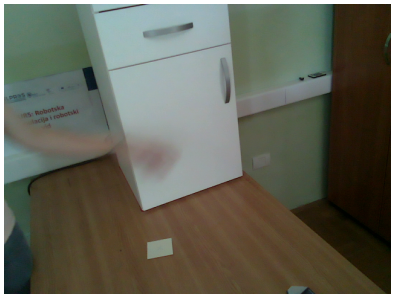


Slika 6.14: *Vizualizacija modela 15 s neispravno izgrađenim modelom vrata*

pozicioniranja u početni položaj. Ovo se događalo u slučajevima u kojima je drugi članak robota paralelan sa stolom na kojem se nalazi robot, kao što je vidljivo na slici 6.16. Prilikom pozicioniranja robota u početni položaj, došlo bi do kolizije robota i stola, što bi pokrenulo sigurnosne mjere robota te bi se robot zaustavio.

Zbog položaja robota u nekim slučajevima nije bilo moguće izračunati kutove zglobova inverznom kinematikom za zadani početni položaj.

(a) *Model 7*



(b) *Model 9*

(c) *Model 12*

(d) *Model 15*

(e) *Model 19*

Slika 6.15: *Prikaz snimanja netočno izgrađenih modela*

Slika 6.16: *Primjer lošeg položaja robotskog manipulatora u trenutku snimanja scene*

7. ZAKLJUČAK

U ovom diplomskom radu predstavljen je sustav za otvaranje i zatvaranje vrata robotskim manipulatorom pomoću računalnog vida i upravljanja silom. Postojećom metodom *Door and Drawer Detector* temeljenom na računalnom vidom provedena je detekcija i izgradnja modela vrata. Primjena senzora sile pri vrhu alata robotskog manipulatora i upravljanja po sili osigurava adaptivnu manipulaciju otvaranja i zatvaranja vrata, koja osigurava veću preciznost izvršenja zadatka i smanjuje rizik oštećenja manipulatora ili dodatne opreme. Sustav je implementiran na UR5 CB3 robotski manipulator, s hvataljkom Robotiq 3-Finger Adaptive Robot Gripper, senzorom sile FT 300-S i Intel RealSense LiDAR Camera L515 kamerom. Cjelokupna programska implementacija ispitana je u ROS Noetic distribuciji i u Docker sustavu.

Tijekom eksperimentalne analize, vrednovane su uspješnost ispravne detekcije vrata te uspješnost obavljanja zadatka otvaranja i zatvaranja vrata. Provedena su 23 eksperimenta. Oba zadatka uspješno su obavljena u 60.86% slučajeva. U tim slučajevima, robotska manipulacija vratima prilikom postupka otvaranja i zatvaranja provedena je bez oštećenja na vratima i na robotskom manipulatoru, što govori u prilog učinkovitosti uporabe senzora sile i upravljanja po sili. U 21.74% pogrešno je izgrađen model vrata te se zadatak otvaranja vrata nije mogao izvesti. U preostalih 17.39% slučajeva nepoznata je učinkovitost algoritama zbog prekida u izvođenju uzrokovana aktivacijom sigurnosnih mjera robota.

Utjecaj na opisane rezultate imaju položaj kamere prilikom izgradnje modela vrata te međusobni položaj robotskog manipulatora i ormara u početnom trenutku provođenja pokusa. U slučaju eksperimenata u kojima je krivo izgrađen model, ormarić je sniman iz bočnih i ptičjih perspektiva. Kutovi snimanja koji su rezultirali ispravnom izgradnjom modela vrata bili su usmjereni na ormarić s prednje stane i iz bliže udaljenosti. U preostalim slučajevima, u kojima je pokus prekinut aktivacijom sigurnosnih mjera robota, došlo je kolizije robota sa stolom ili je pozicioniranje robota u zadanu točku bilo neizvedivo.

Zaključak je da postoji velika osjetljivost na kut pri kojem su snimana vrata ormarića pri čemu algoritam za detekciju i izgradnju modela pogrešno detektira i neprecizno izgrađuje model vrata. Rješenje ovog problema bilo bi povećanje robusnosti korištenog algoritma za različite kuteve i udaljenosti snimanja pri kojima se može ispravno izgraditi model. Nadalje, aktivaciju sigurnosnih mjera robota, u slučajevima u kojima je došlo do kolizije sa stolom, moguće je izbjeći postavljanjem robota na drugačije postolje, primjerice stup površine prikladne za montiranje baze robota. Prilikom hvatanja ručke, koje se ne provodi adaptivno, zadatak hvatanja ručke obavljen je grubo s mogućnosti oštećenja ručke ili samog namještaja. Stoga je potrebno razviti metodu upravljanja silom tijekom hvatanja, metodu vizualnog navođenja ili precizniju metodu pozicioniranja prilikom prihvaćanja ručke.

LITERATURA

[1] Miguel Arduengo, Carme Torras, and Luis Sentis. Robust and adaptive door operation with a mobile robot. *Intelligent Service Robotics*, 14(3):409–425, July 2021.

[2] Mrinal Kalakrishnan, Ludovic Righetti, Peter Pastor, and Stefan Schaal. Learning force control policies for compliant manipulation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4639–4644, September 2011.

[3] Bojan Nemec, Leon Žlajpah, and Aleš Ude. Door opening by joining reinforcement learning and intelligent control. In *2017 18th International Conference on Advanced Robotics (ICAR)*, pages 222–228, July 2017.

[4] Yiannis Karayiannidis, Christian Smith, Petter Ögren, and Danica Kragic. Adaptive Force/Velocity control for opening unknown doors1. *IFAC Proceedings Volumes*, 45(22):753–758, January 2012.

[5] C.-C Chen, J.-S Li, and J. Luo. Robust Adaptive Position and Force Tracking Control Strategy for Door-Opening Behaviour. *International Journal of Simulation Modelling*, 15:426–435, September 2016.

[6] 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)*, Busan, Korea, 2023. prihvaćen za objavljivanje.

[7] Kevin M. Lynch and Frank C. Park. *Modern robotics: mechanics, planning, and control*. Cambridge University Press, Cambridge, UK, 2017.

[8] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Pearson, Upper Saddle River, N.J, 3rd edition edition, August 2004.

[9] Universal Robots, User Manual, UR5/CB3. https://www.usna.edu/Users/weaprcon/kutzer/_files/documents/User%20Manual,%20UR5.pdf. Pristup: 21.6.2023.

[10] Universal Robots, User Manual, PolyScope Manual. https://www.usna.edu/Users/weaprcon/kutzer/_files/documents/Software%20Manual,%20UR.pdf. Pristup: 21.6.2023.

[11] Universal Robots, User Manual, The URScript Programming Language for e- Series. https://s3-eu-west-1.amazonaws.com/ur-support-site/115824/scriptManual_SW5.11.pdf. Pristup: 21.6.2023.

[12] Robotiq 3-Finger Adaptive Robot Gripper, Instruction Manual. https://assets.robotiq.com/website-assets/support_documents/document/3-Finger_PDF_20190221.pdf. Pristup: 16.6.2023.

[13] Robotiq FT 300-S Force Torqu Sensor for TM Series Robots, Instruction Manual. https://s3.amazonaws.com/com-robotiq-website-prod-assets/website-assets/support_documents/document/FT300-S_Sensor_Manual_TM_PDF_20210301.pdf. Pristup: 16.6.2023.

[14] Intel RealSense LiDAR Camera L515 Datasheet. https://dev.intelrealsense.com/docs/lidar-camera-l515-datasheet. Pristup: 16.6.2023.

[15] Intel® RealSense™ LiDAR Camera L515. https://www.intelrealsense.com/lidar-camera-l515/. Pristup: 16.6.2023.

[16] ROS/Introduction - ROS Wiki. https://wiki.ros.org/ROS/Introduction. Pristup: 16.6.2023.

[17] Morgan Quigley, Brian Gerkey, and William Smart. *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*. O'Reilly Media, Sebastopol, 1st edition edition, January 2016.

[18] Docker overview. https://docs.docker.com/get-started/overview/. Pristup: 18.6.2023.

SAŽETAK

U ovom diplomskom radu obrađen je problem detekcije i otvaranja vrata robotskim manipulatorom uz primjenu računalnog vida. Detekcija i izgradnja modela vrata obavljena je na temelju sekvence RGB-D slika kojima je prikazana ljudska demonstracija otvaranja vrata. Za postupak otvaranja vrata, na temelju prethodno izračunatih točaka putanje, koristilo se hibridno upravljanje silom i pozicijom. Robotski sustav sastojao se od UR5 robotskog manipulatora, robotskog alata Robotiq 3-Finger Adaptive Robot Gripper, Intel RealSense LiDAR Camera L515 kamere i senzora sile FT 300-S. Za programiranje robotskog sustava korišteni su razvojni alati ROS Noetic i Docker. Eksperimentalnom analizom utvrdila se uspješnost izvođenja algoritama za detekciju i izgradnju modela vrata i algoritma za otvaranje vrata robotskom rukom uz upravljanje silom. U 60.86% pokusa uspješno su izvedena oba algoritma, u 21.74% pogrešno je izgrađen model vrata te je u ovim slučajevima nemoguće odrediti uspješnost algoritma za otvaranje vrata. U preostalih 17.39% slučajeva nije bilo moguće utvrditi uspješnost niti jednog algoritma, zbog aktiviranja sigurnosnih mjera uzrokovanih lošim položajem manipulatora.

Ključne riječi: detekcija vrata na RGB-D slikama, robotska manipulacija, upravljanje silom, UR5, ROS

ABSTRACT

This master thesis addresses the problem of door detection and opening using a robotic manipulator with the application of computer vision. The detection and construction of the door model were based on a sequence of RGB-D images showing a human demonstration of door opening. Hybrid force-position control was used to open the door using the previously calculated path points. The robotic system consisted of a UR5 robotic manipulator, a Robotiq 3-finger adaptive robotic gripper, the L515 Intel RealSense LiDAR camera, and FT 300-S force sensor. The development tools ROS Noetic and Docker were used to programme the robotic system. Experimental analysis was performed to evaluate the performance of the door detection, model construction, and force control opening algorithms. Both algorithms were successfully executed in 60.86% of the experiments, while in 21.74% of the cases the door model was inaccurately constructe making it impossible to determine the success of the door opening algorithm. In the remaining 17.39% of cases, the success of neither algorithm could be determined because security measures were activated due to the poor positioning of the manipulator.

Keywords: door detection on RGB-D images, robotic manipulation, force control, UR5, ROS

ŽIVOTOPIS

Lukrecia Vulić rođena je u Našicama 4. prosinca 1999. Završila je Osnovnu školu kralja Tomislava Našice 2014. godine i iste godine upisuje Srednju školu Isidora Kršnjavoga u Našicama. Nastavlja školovanje na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku 2018. godine na preddiplomskom studiju računarstva. Na istom fakultetu 2021. godine upisuje diplomski studiji računarstva, smjer Robotika i umjetna inteligencija. Dobitnica je nagrada za sudjelovanje u izvannastavnim aktivnostima, za uspješnost u studiranju i dvije nagrade za izrađene studentske radove. Sudjelovala je u izradi rada *Ramifications of the greenhouse effect: a closed vehicle example* objavljenog na konferenciji SST 2022.

PRILOG/NA CD-U

1. Diplomski rad u PDF formatu
2. Programsko rješenje
3. Podaci dobiveni eksperimentalnom analizom