Praćenje ljudske anatomije robotom koristeći medicinski stereovizijski sustav

Akrap, Nikola

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Nikola Akrap

Zagreb, 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentori:

Doc. dr. sc. Marko Švaco, mag. ing.

Student:

Nikola Akrap

Zagreb, 2024.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se doc. dr. sc. Marku Švaci na pruženoj pomoći i svim savjetima tijekom pisanja ovog rada. Također se zahvaljujem Branimiru Ćaranu, mag. ing., na izdvojenom vremenu i korisnim savjetima.

Zahvaljujem se i svim svojim kolegama i najbližim prijateljima na podršci i podupiranju.

Na kraju, zahvaljujem se i svojoj obitelji, a posebice roditeljima na konstantnom bodrenju, podršci i bezuvjetnoj ljubavi koju su mi pružili tijekom studiranja i pisanja ovog rada.

Nikola Akrap

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne i diplomske ispite studija mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa: **602 – 04 / 24 – 06 / 1**	
Ur.broj: **15 – 24 –**	

ZAVRŠNI ZADATAK

Student: **Nikola Akrap** JMBAG: **0035239991**

Naslov rada na hrvatskom jeziku: **Praćenje ljudske anatomije robotom koristeći medicinski stereovizijski sustav**

Naslov rada na engleskom jeziku: **Robot tracking of human anatomy using a medical stereovision system**

Opis zadatka:

Primjena robota u različitim medicinskim postupcima u današnje vrijeme bilježi sve veći rast. Roboti se koriste od područja rehabilitacije pa sve do kirurgije. Jedan od važnih aspekata primjene robota u medicini je brzina prilagodbe na promjene u položaju (poziciji i orijentaciji) pacijenta u odnosu na kojeg robotska ruka treba održavati zadani relativni položaj. Planirani relativni položaj robotske ruke u odnosu na pacijenta moguće je koristeći ostvariti medicinske stereovizijske sustave za praćenje krutih markera.

U ovom radu potrebno je ostvariti laboratorijsku primjenu praćenja dijela ljudskog tijela kao što su glava ili noga koristeći robotsku ruku i medicinski stereovizijski sustav za praćenje krutih markera. Anatomske strukture moguće je 3D printati ili koristiti dostupne imitacije (fantome) u laboratoriju.

U sklopu rada potrebno je:

- upoznati se s radom s medicinskog stereovizijskog sustava za praćenje krutih (engl. *rigid*) markera
- povezati računalo i medicinski stereovizijski sustav za praćenje krutih markera te ostvariti komunikaciju putem TCP/IP ili serijskog protokola
- napraviti simulaciju praćenja objekta ili anatomske strukture na računalu koristeći simuliranu robotsku ruku
- izraditi i kalibrirati sve potrebne alate (markere) koji će biti vezani uz robotsku ruku
- koristeći povratne informacije o položaju robota i anatomske strukture u stvarnom prostoru omogućiti praćenje anatomske strukture u stvarnom vremenu
- ispitati funkcionalnost te mogućnosti održavanja zadanog relativnog položaja za rubne uvjete kao što su najveće brzine i akceleracije, nagle promjene smjera i sl.

Funkcionalnosti iz završnog rada potrebno je demonstrirati na računalnoj opremi te na odabranom robotu u Regionalnom centru izvrsnosti za robotske tehnologije.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan:	Datum predaje rada:	Predviđeni datumi obrane:
24. 4. 2024.	**2. rok (izvanredni):** 11. 7. 2024. **3. rok:** 19. i 20. 9. 2024.	**2. rok (izvanredni):** 15. 7. 2024. **3. rok:** 23. 9. – 27. 9. 2024.

Zadatak zadao: Predsjednik Povjerenstva:

doc. dr. sc. Marko Švaco izv. prof. dr. sc. Petar Ćurković

SADRŽAJ

SADRŽAJ I
POPIS SLIKA III
POPIS TABLICA V
POPIS OZNAKA VI
POPIS KRATICA VII
SAŽETAK VIII
SUMMARY IX
1. UVOD 1
1.1. Robotski sustavi u medicini 2
1.1.1. Mako Robotic-Arm Assisted Surgery 2
1.1.2. StealthStation 3
1.1.3. ExcelsiusGPS 4
2. STEREOVIZIJSKI SUSTAVI 5
2.1. NDI Polaris Vega ST 6
3. UNIVERSAL ROBOTS UR5e 9
3.1. Mehaničke karakteristike 9
3.2. Ispitivanje komunikacije robotske ruke i stereovizijskog sustava u *RoboDK* okolišu 12
3.2.1. Računanje potrebnih transformacija 12
3.2.2. Simulacija u RoboDK 13
3.3. Problematika integracije robotske ruke sa stereovizijskim sustavom 14
3.4. Podešavanje robota i integracija u ROS2 okruženje 14
4. PREDUVJETI ZA IZRADU 17
4.1. ROS2 i MoveIt2 17
4.1.1. Principi rada ROS2 17
4.2. Namještanje radnog okruženja 19
4.2.1. Knjižnica TF2 19
4.2.2. Paket „ur_robot_driver“ 20
4.2.3. Paket „pymoveit2“ 21
4.2.4. Paket „scikit-surgerynditracker“ 21
5. POSTAVLJANJE EKSPREIMENTALNOG OKRUŽENJA 22
5.1. Namještanje koordinatnog sustava *stylus*-a 23
5.2. Povezivanje TCP/IP protokolom 26
5.3. Implementacija ROS2 26
5.3.1. Prikaz u RViz-u 26
5.3.2. Planiranje gibanja 27
5.3.3. Provedba gibanja 28
5.3.3.1. Izrada algoritma lokalizacije i registracije 29
5.3.3.2. ROS2 Servisi 33

6. ISPITIVANJE I ANALIZA ALGORITMA 35
6.1. Prva skupina ispitivanja 36
6.2. Druga skupina ispitivanja 37
6.3. Treća skupina ispitivanja 38
6.4. Analiza ispitivanja 38
6.4.1. Analiza prve skupine ispitivanja 39
6.4.2. Analiza druge skupine ispitivanja 40
6.4.3. Analiza treće skupine ispitivanja 42
7. ZAKLJUČAK 44
LITERATURA 45
PRILOZI 46

POPIS SLIKA

Slika 1. Mako robotska ruka[2] 2
Slika 2. StealthStation neuronavigacijski sustav[4] 3
Slika 3. ExcelsiusGPS platforma[5] 4
Slika 4. Princip rada stereovizijskog sustava[6] 5
Slika 5. NDI Polaris Vega ST[8] 6
Slika 6. Pasivni markeri[8] 7
Slika 7. Polaris Vega ST - mjerni volumen[8] 8
Slika 8. UR5e[9] 9
Slika 9. UR5e - radni prostor[9] 10
Slika 10. UR5e - gabaritne dimenzije[9] 10
Slika 11. Prikaz odnosa koordinatnih sustava alata za navođenje robota 12
Slika 12. UR5e - RoboDK 13
Slika 13. UR5e - upravljačka ploča 14
Slika 14. UR5e - prikaz u *RViz*-u 16
Slika 15. ROS2 – komunikacija[11] 17
Slika 16. Radno okruženje "zavrsni" 18
Slika 17. UR5e - prikaz u MoveIt2 20
Slika 18. Eksperimentalno okruženje 22
Slika 19. Alati eksperimentalnog okruženja 23
Slika 20. Mjerenje *stylus*-a mikrometrom 24
Slika 21. Prvotni koordinatni sustav *stylus*-a 25
Slika 22. Transformirani koordinatni sustav *stylus*-a 25
Slika 23. Lokacija statičnog markera u odnosu na bazu robota 27
Slika 24. Dijagram toka lokalizacije alata 27
Slika 25. Pokretanje čvorova 31
Slika 26. Slanje brzina zglobovima robota 31
Slika 27. Dijagram toka algoritma lokalizacije 32
Slika 28. Dijagram toka pozivanja servisa 32
Slika 29. ROS2 – princip komunikacije servisa[14] 33
Slika 30. Poziv prvog servisa 34
Slika 31. Poziv drugog servisa 34
Slika 32. Postav za provedbu testiranja 35
Slika 33. Ispitivanje kompenzacije malih pomaka: prihvatnica u početnom položaju (a), prihvatnica uz *stylus* (b), prihvatnica na položaju *stylus*-a (c), prihvatnica zadržava relativan položaj nakon zakreta kosti (d) 36
Slika 34. Ispitivanje kompenzacije velikih pomaka: prihvatnica u početnom položaju (a), veliki pomak kosti (b), prihvatnica kompenzira pomak kosti (c) 37
Slika 35. Ispitivanje graničnih slučajeva: početni položaj (a), nagla rotacija(b), visoka brzina translacije (c) 38
Slika 36. Dijagrami promjene translacije (lijevo) i rotacije (desno) prihvatnice i kosti za prvu skupinu ispitivanja 39
Slika 37. Dijagrami ukupne Euklidske pogreške (lijevo) i promjene kuta između z-osi prihvatnice i kosti (desno) za prvu skupinu ispitivanja 40
Slika 38. Dijagrami promjene translacije (lijevo) i rotacije (desno) prihvatnice i kosti za drugu skupinu ispitivanja 41
Slika 39. Dijagrami ukupne Euklidske pogreške (lijevo) i promjene kuta između z-osi prihvatnice i kosti (desno) za drugu skupinu ispitivanja 41

Slika 40. Dijagrami promjene translacije (lijevo) i rotacije (desno) prihvatnice i kosti za treću skupinu ispitivanja..... 42
Slika 41. Dijagrami ukupne Euklidske pogreške (lijevo) i promjene kuta između z-osi prihvatnice i kosti (desno) za treću skupinu ispitivanja 43

POPIS TABLICA

Tablica 1. Tehničke specifikacije sustava Polaris Vega ST[8] 7
Tablica 2. UR5e - mehaničke karakteristike[10]..... 11

POPIS OZNAKA

Oznaka	Jedinica	Opis
K_{pom}		Matrica transformacije između pomičnog krutog tijela i stereovizijskog sustava
K_{stat}		Matrica transformacije između statičnog krutog tijela i stereovizijskog sustava
K_{sty}		Matrica transformacije između *stylus*-a i stereovizijskog sustava
T_{pom}		Matrica transformacije između pomičnog krutog tijela i statičnog krutog tijela
T_{sty}		Matrica transformacije između *stylus*-a i statičnog krutog tijela
T		Matrica translacije koordinatnog sustava stylusa
R_y		Matrica rotacije koordinatnog sustava stylusa oko y osi
R_z		Matrica rotacije koordinatnog sustava stylusa oko z osi
T_{kon}		Matrica transformacije koordinatnog sustava stylusa
q_{stat}		Kvaternion transformacije između statičnog alata i baze robota
ΔP	mm	Srednja vrijednost pogreške
Δx	mm	Translacija po x-osi
Δy	mm	Translacija po y-osi
Δz	mm	Translacija po z-osi
ΔQ_x	rad	Rotacija oko x-osi
ΔQ_y	rad	Rotacija oko y-osi
ΔQ_z	rad	Rotacija oko z-osi

POPIS KRATICA

Oznaka	Opis
IP	Engl. Internet Protocol
TCP	Engl. Transmission Control Protocol
RMS	Engl. Root mean square
MRI	Engl. Magnetic resonance imaging
CT	Engl. Computed tomography

SAŽETAK

Cilj ovog rada je integracija kolaborativne robotske ruke UR5e i stereovizijskog optičkog sustava za praćenje Polaris Vega ST u ROS2 operativni sustav, njihovo povezivanje i osposobljavanje za praćenje dijela tijela tijekom izvođenja medicinskog zahvata. Stereovizijski optički sustavi omogućuju kontinuirano praćenje položaja i orijentacije objekata u prostoru koristeći povratne informacije markera od kojih se odbija infracrveno svjetlo iz izvora sa samog stereovizijskog sustava. Istražit će se kako kolaborativna robotska ruka u kombinaciji s jednim takvim sustavom može precizno pratiti anatomske strukture poput cjevanice (lat. *tibia*) i bedrene kosti (lat. *femur*) za vrijeme operacije koljena. U slučaju operacije navedenih kostiju, koljenu niti u jednom trenutku nije u potpunosti onemogućeno gibanje te je potrebno kompenzirati sve pomake pa čak i one najmanje nevidljive ljudskom oku. Navedene su glavne karakteristike robotske ruke i stereovizijskog optičkog sustava. U RoboDK okolišu napravljena je simulacija robotske ruke s ciljem testiranja optičkog sustava za praćenje. Napravljen je uvid u način povezivanja robotske ruke, optičkog sustava i računala. Pokazano je eksperimentalno okruženje u kojem se provelo testiranje u Regionalnom centru izvrsnosti za robotske tehnologije. Nadalje, pokazana je integracija sustava u ROS2 i MoveIt2 kao i pregled svih potrebnih paketa za nesmetano izvođenje rada. Predočene su sve skripte i algoritmi koji pogone sustav i omogućeno je pokretanje robota pomoću serva. Pokazane su mogućnosti stereovizijskih sustava za praćenje koji su nova pojava u polju medicinske robotike.

Ključne riječi: robotika, medicina, stereovizija, ROS2

SUMMARY

The aim of this paper is to integrate of the collaborative robotic arm UR5e and the Polaris Vega ST stereovision optical tracking system into the ROS2 operating system, establish their connection, and prepare them for tracking a body part during a medical procedure. Stereovision optical systems enable continuous tracking of the position and orientation of objects in space by using feedback from markers that reflect infrared light from the stereovision system itself. The study will explore how a collaborative robotic arm, in combination with such a system, can precisely track anatomical structures such as the tibia and femur during knee surgery. During surgery on these bones, the knee is never fully immobilized, so it is necessary to compensate for all movements, even those that are invisible to the human eye. The main characteristics of the robotic arm and the stereovision optical system are described. A simulation of the robotic arm was created in the RoboDK environment to test the optical tracking system. An overview of the connection between the robotic arm, the optical system, and the computer was made. The experimental environment used for testing at the Regional Center of Excellence for Robotic Technologies is presented. Furthermore, the system integration in ROS2 and MoveIt2 is shown, as well as a review of all the necessary packages for smooth operation. All scripts and algorithms that drive the system are presented, enabling the robot to be operated via a servomechanism. The capabilities of stereovision tracking systems, which are still a relatively new development in the field of medical robotics, are demonstrated.

Key words: robotics, medicine, stereovision, ROS2

1. UVOD

U današnjem modernom i dinamičnom svijetu sve je veći fokus stavljen na ljudsko zdravlje. Zbog brzog tehnološkog napretka i zbog sve većih potreba za poboljšanjem kvalitete života, primjena novih tehnologija u području medicine postaje neizbježna. To uključuje i robotiku koja je sama po sebi još uvijek relativno nova tehnologija u medicinskoj praksi no možemo uočiti kako počinje zauzimati prvo mjesto u raznim postupcima i zahvatima. Dugo je robotika svoju jedinu ulogu imala u industriji gdje se primarno koristila za ubrzavanje proizvodnih procesa te povećavanje i održivost kvalitete. Tek nedavno krenula je poprimati temelje u medicini i počela se seliti iz industrijskih pogona u operacijske sale.

Primjeri poput *da Vinci* kirurškog sustava, koji omogućava minimalno invazivne zahvate u postupcima kao što je laparoskopija ili *Senhance Surgical System* kirurškog sustava koji svoju primjenu ima u laparoskopskim zahvatima ginekologije, urologije te općoj kirurgiji kao i u operacijama toraksa, pokazuju kako robotika prelazi u svijet zdravstvene skrbi.

Unatoč navedenim napredcima, izazovi ostaju, a posebno u kontekstu potrebe za prilagodbom robotskih sustava na promjene položaja pacijenta. Prethodno navedeni medicinski sustavi uvelike su razvijeni i pružaju dobru pomoć u izvođenju zahvata, ali ne i u čuvanju položaja zbog konstantne potrebe primanja povratnih informacija od kirurga. Uvođenjem medicinskih stereovizijskih sustava otvaraju se nove mogućnosti u izvođenju medicinskih zahvata s povećanjem faktora sigurnosti.

Često korišteni primjerni optički navigiranih medicinskih sustava uključuju *Mako Robotic-Arm Assisted Surgery* robotsku ruku koja se koristi u ortopedskim zahvatima, *StealthStation* sustav za neurokirurgiju i zahvate leđne moždine, *ROSA Robotic System* sustav za neurokirurgiju i ortopediju i *ExcelsiusGPS* za operacije kralježnice.

U ovom radu bit će potrebno omogućiti međusobnu komunikaciju kolaborativne robotske ruke UR5e i optičkog stereovizijskog sustava Polaris Vega ST u slučaju operacije bedrene kosti (lat. *femur*) pomoću njihove integracije u ROS2 operativni sustav. Biti će potrebno izraditi sve potrebne *Python* skripte kako bi se omogućio nesmetan rad algoritma u kojemu se robotskoj ruci mora omogućiti održavanje relativnog položaja i orijentacije u odnosu na kost za vrijeme operacije pomoću podataka dobivenih od optičkog stereovizijskog sustava. Istražit će se točnost održavanja relativnog položaja u stvarnom vremenu, a ispitat će se i mogućnost održavanja relativnog položaja za slučajeve najvećih brzina i akceleracija te naglih promjena smjera kretanja.

1.1. Robotski sustavi u medicini

Robotika je već jednom nogom zagazila u područje medicine što pokazuju već navedeni robotski sustavi. Njihova glavna svrha je pomoći u raznim medicinskim zahvatima koji se izvode svakodnevno diljem svijeta. Kako bi se svrha ovog rada bolje približila potrebno je dati detaljniji uvod u dostupne sustave i načine na koje funkcioniraju.

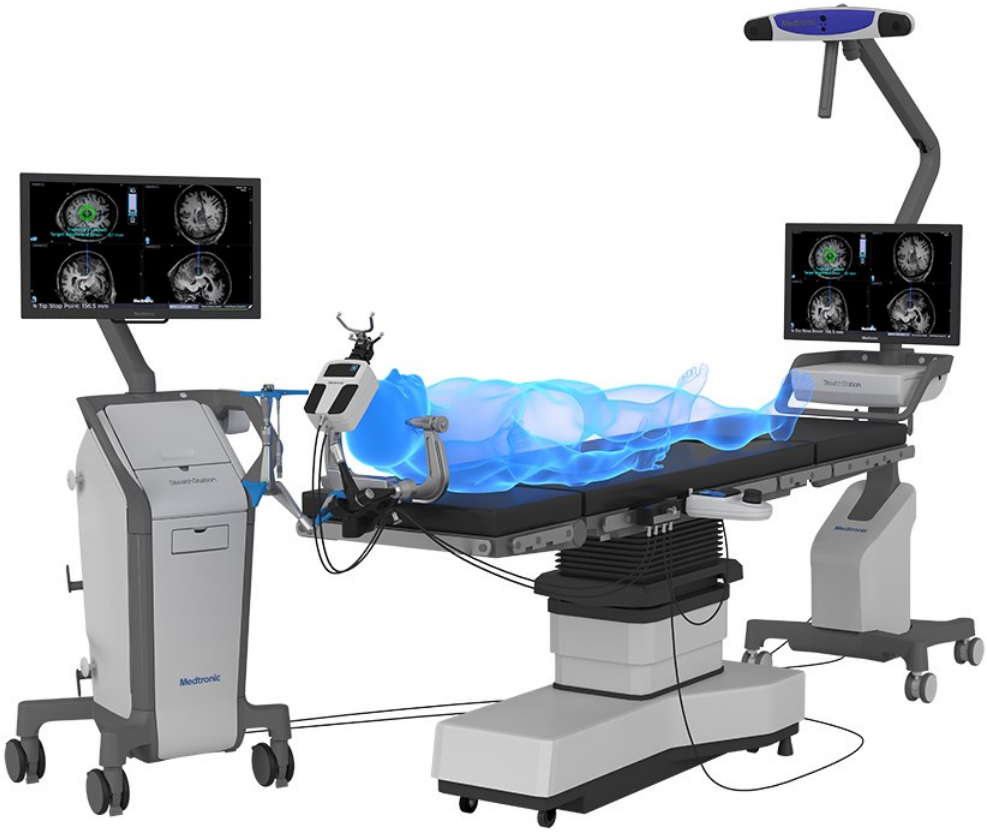
1.1.1. Mako Robotic-Arm Assisted Surgery

Mako robotski sustav za pomoć pri operaciji je tehnologija razvijena za operacije zamjene koljena. Omogućuje ortopedskom kirurgu da unaprijed isplanira operaciju koristeći *Mako* softver, a zatim izvede zahvat vođenjem robotske ruke kako bi precizno uklonio kost i hrskavicu. Prije operacije, putem MRI snimke dobiva se trodimenzionalni virtualni model zglobova nakon čega se pristupa preciznom planiranju zahvata i izradi implantata. Nakon što operacija započne usklađuje se trodimenzionalni model zglobova s anatomijom zglobova pacijenta u operacijskoj sali. Nakon usklađivanja modela *Mako* sustav putem optičkog reflektirajućeg sustava za praćenje markera počinje promatrati zglob pacijenta analizirajući biomehaniku zglobova i istegnuće ligamenata. Nakon optičke obrade i planiranja operativnog zahvata, kirurg pomoću robotske ruke izvodi isplanirane zahvate.

Slika 1. Mako robotska ruka[2]

1.1.2. StealthStation

Neuronavigacijski sustav *StealthStation* namijenjen je kao pomoć pri preciznom lociranju anatomskih struktura u otvorenim ili perkutanim zahvatima. Sustav *StealthStation* namijenjen je za bilo koje medicinsko stanje u kojem bi stereotaktička kirurgija mogla biti prikladna, a gdje se može identificirati referenca na krutu anatomsku strukturu, poput lubanje, duge kosti ili kralješka, u odnosu na CT ili MR model, fluoroskopske slike ili digitalizirane anatomske orijentire. Sustav nudi optičke i elektromagnetske mogućnosti praćenja, integraciju s vanjskim uređajima poput mikroskopa i ultrazvuka, širok izbor instrumenata kao i softverske aplikacije za neurokirurgiju i zahvate na kralježnici. Sve mogućnosti sustavu omogućuje optička stereovizijska kamera koja prima informacije od markera koji se nalaze u operacijskoj sali.



Slika 2. StealthStation neuronavigacijski sustav[4]

1.1.3. ExcelsiusGPS

ExcelsiusGPS je prva i trenutno jedina robotska platforma na tržištu koja omogućava detaljan pristup kralježnim i kranijalnim procedurama, od početka do kraja. Platforma u stvarnom vremenu pruža vizualizaciju položaja instrumenata i implantata u odnosu na anatomiju pacijenta, što može pomoći u smanjenju količine potrebnog zračenja tijekom zahvata. Navigacija osigurava kontinuirane povratne informacije i vizualizaciju. Kompatibilna je s preoperativnim CT, intraoperativnim CT i fluoroskopskim sustavima za snimanje. Ovakva prilagodljivost omogućuje sustavu da se uklopi u bilo koji kirurški tijek rada te omogućuje planiranje i navigaciju u 2D ili 3D. Robotska ruka automatski se miče po planiranoj trajektoriji, a svaki alat sadrži markere pomoću kojih optički sustav dobiva točne informacije o njegovom položaju i orijentaciji u prostoru.

Slika 3. ExcelsiusGPS platforma[5]

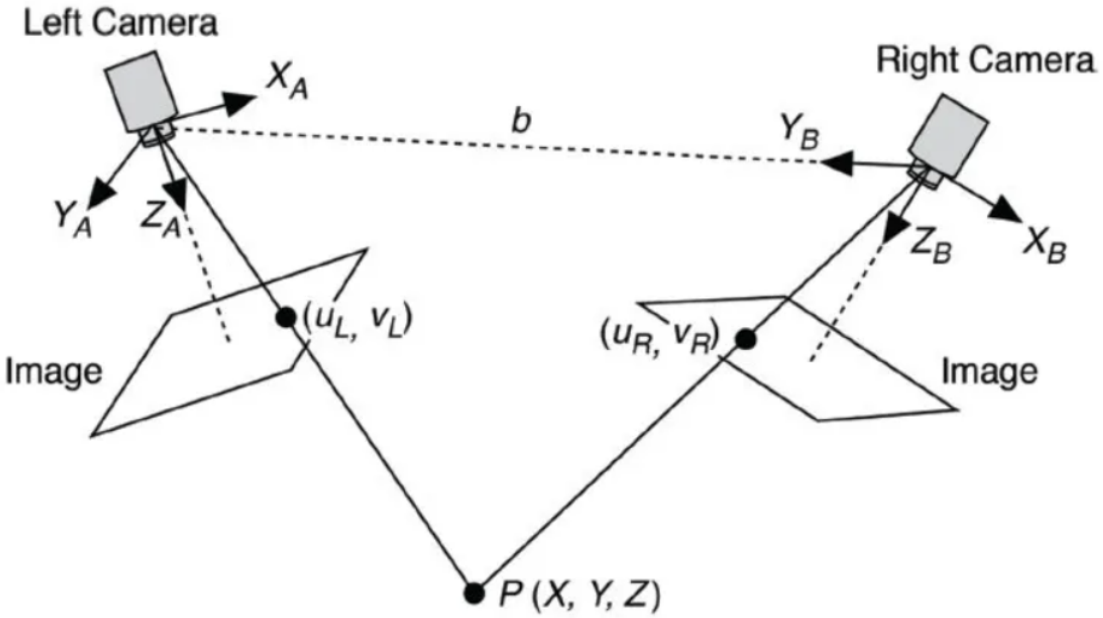
2. STEREOVIZIJSKI SUSTAVI

Tehnologija 3D obrade slika prošla je dug put od svojih početaka u akademskim istraživačkim laboratorijima. Zahvaljujući modernim inovacijama u senzorima, smanjenju troškova komponenti i razvoju 3D funkcija u software-u, 3D vizija se danas može pronaći u raznim primjenama strojne automatizacije. Od robotskog prikupljanja objekata vođenog vizijom pa sve do visoko precizne metrologije, najnovije generacije vizijskih sustava mogu obrađivati goleme količine podataka s pomoću najmodernijih algoritama.

Neke od uobičajenih tehnika 3D vizije su:

- Stereovizija (engl. *Stereovision*)
- Laserska triangulacija (engl. *Laser Triangulation*)
- Strukturirano svjetlo (engl. *Structured Light*)
- Vrijeme leta (engl. *Time of Flight*)

Stereovizijski sustavi, između ostalog, pronašli su svoju primjenu u medicini. Njihov princip rada funkcionira na oponašanju ljudskog vida. Jednako kao što ljudi informacije primaju iz dvije perspektive, odnosno dva oka, tako i stereovizijski sustavi svoje informacije primaju iz dvije horizontalno razmaknute kamere koje objekt gledaju iz dva različita kuta. Poznavanjem relativnog položaja kamera, software može usporediti točke na dvije dvodimenzionalne slike koje kamere proizvode, uočiti razlike i stvoriti trodimenzionalni oblak točaka.



Slika 4. Princip rada stereovizijskog sustava[6]

2.1. NDI Polaris Vega ST

U sklopu ovog rada korišten je optički sustav Polaris Vega ST kanadskog proizvođača Northern Digital Incorporated s pomoću kojeg je moguće pratiti 3D poziciju i orijentaciju markera uz visoku pouzdanost. Optički sustav emitira infracrvenu svjetlost koja se odbija od pasivne markere prekrivene reflektivnim materijalom. Postoji i mogućnost uporabe aktivnih markera koji sami emitiraju infracrvenu svjetlost no to u ovom radu nije obrađeno. Pasivni markeri se postavljaju na alate po točno određenim pravilima:

- Svaki alat mora sadržavati minimalno tri markera
- Udaljenosti između markera moraju biti različite

Pasivne markere prikazuje Slika 6.

Kamere optičkog sustava sadrže senzore koji su osposobljeni za primanje reflektirane i emitirane infracrvene svjetlosti od markera te na principu stereovizije, koji je objašnjen u poglavlju 2, određuju poziciju markera u 3D prostoru.

Slika 5. NDI Polaris Vega ST[8]

Ovaj optički sustav razvijen je kako bi svoju primjenu imao u medicini, a neka od područja u kojima se koristi su:

- Transkranijalna kirurgija
- Ortopedija
- Neurokirurgija
- Radioterapija
- Dentalna medicina

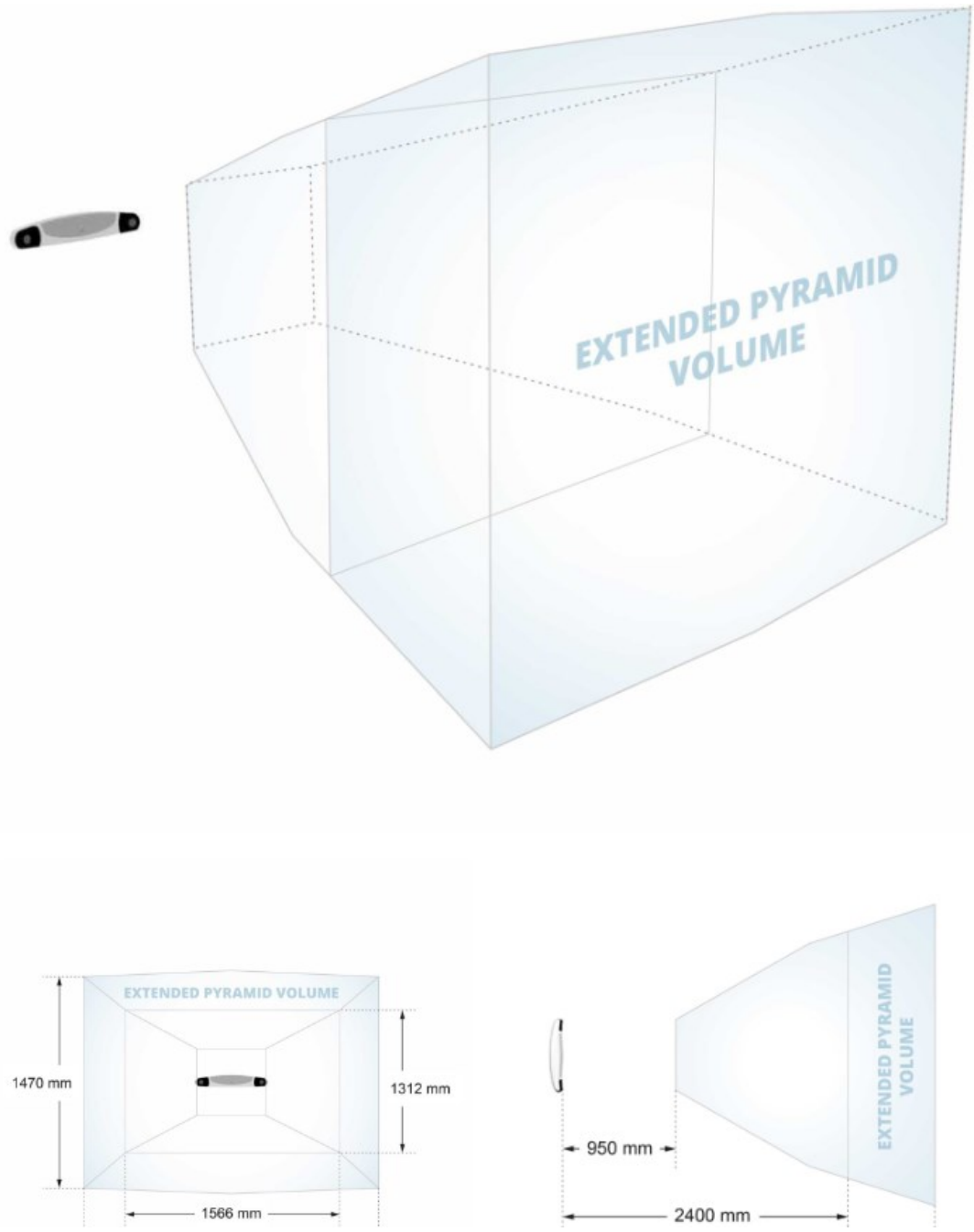
Slika 6. Pasivni markeri[8]

U sljedećoj tablici prikazane su detaljne karakteristike optičkog sustava Polaris Vega ST.

Tablica 1. Tehničke specifikacije sustava Polaris Vega ST[8]

Uređaj	Polaris Vega ST
Brzine osvježavanja	20 Hz, 30 Hz, 60 Hz
Latencija	17 ms pri 60 Hz
Mjerni volumen	Slika 7
RMS točnost	0,12 mm
Maksimalan broj alata	25 (najviše 6 bežičnih s aktivnim markerima)
Maksimalan broj markera po alatu	6 na jednoplanarnom / 20 na višeplanarnom
Komunikacijski protokol	TCP
Dimenzije	591 mm × 103 mm × 106mm
Masa	1,7 kg (2,5 kg) [1]

[1] U slučaju rada u prostorima s visokom koncentracijom ionizirajućeg zračenja elektroničke komponente moraju biti otpornije na radijaciju pa se im se iz tog razloga dodatno ojačava otpor na zračenje



Slika 7. Polaris Vega ST - mjerni volumen[8]

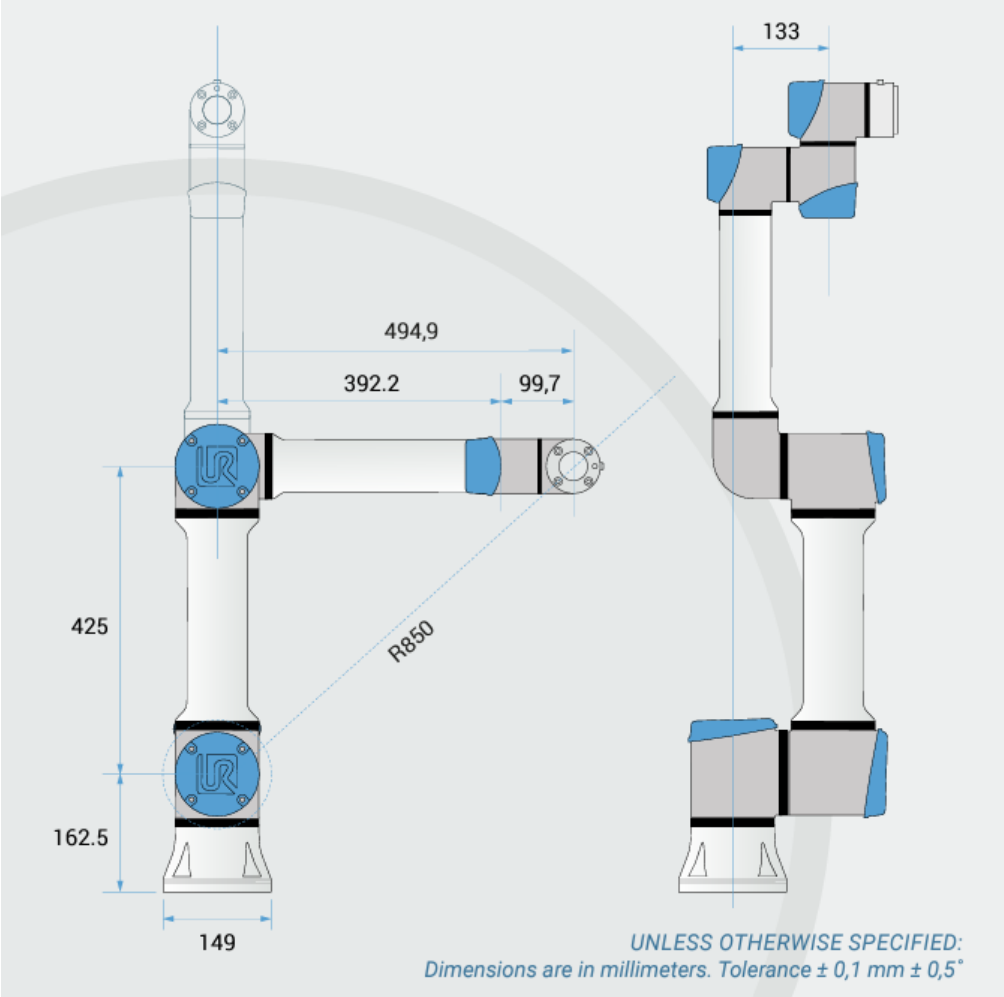
3. UNIVERSAL ROBOTS UR5e

3.1. Mehaničke karakteristike

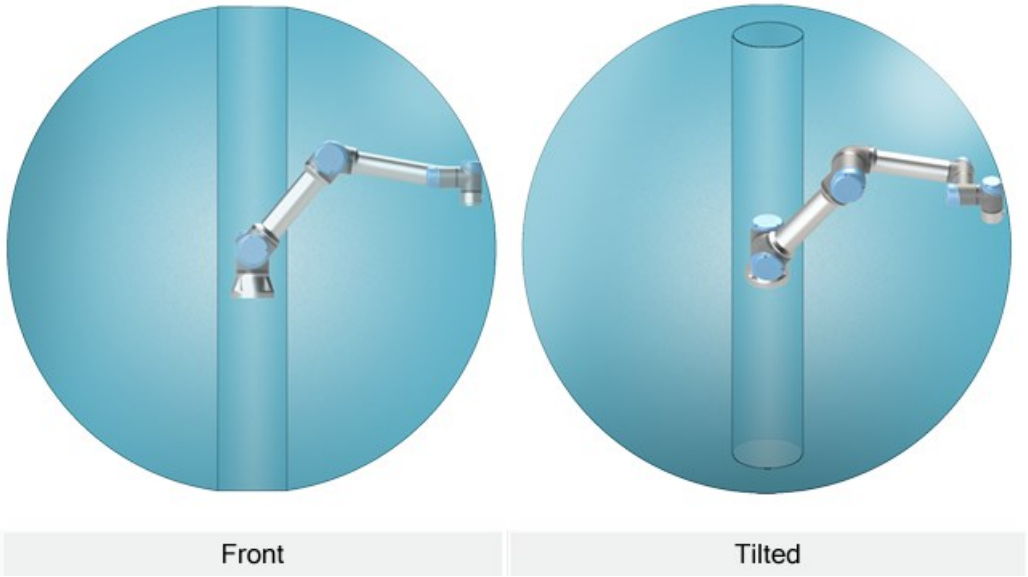
UR5e kolaborativna je robotska ruka Danskog proizvođača Universal Robots koja je 2018. godine nastala kao nadogradnja na već poznatu UR5 kolaborativnu ruku. Poznata je kao jako fleksibilna kolaborativna robotska ruka s ugrađenim senzorima sile i momenta, povećanom preciznošću i još mnogim drugim sustavima koji omogućuju sigurnu automatizaciju ponovljivih i rizičnih operacija. Zbog izvrsnih kolaborativnih karakteristika i jednostavnosti korištenja ova robotska ruka idealna je za primjenu u medicini. Upravo iz tog razloga odabrana je za uporabu u ovom radu.

Slika 8. UR5e[9]

UR5e ima šest stupnjeva slobode gibanja s maksimalnom nosivošću od 5 kg i dosegom od 850 mm.



Slika 10. UR5e - gabaritne dimenzije[9]



Slika 9. UR5e - radni prostor[9]

U sljedećoj su tablici prikazane neke mehaničke karakteristike robota:

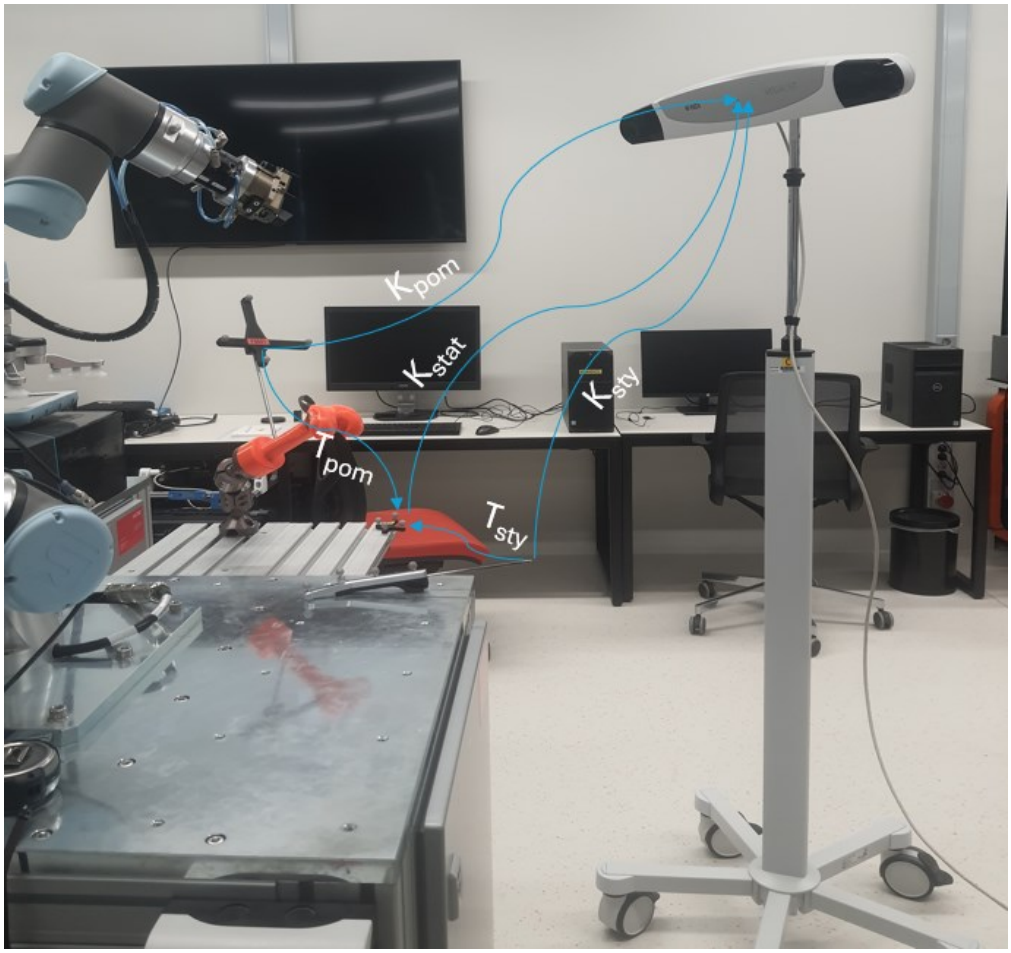
Tablica 2. UR5e - mehaničke karakteristike[9]

	UR5e
Broj stupnjeva slobode gibanja	6
Masa	20,6 kg
Nosivost	5 kg
Doseg	850 mm
Radni opseg zglobova	±360º
Brzina zglobova	±180º /s
Brzina prihvatnice	1 m/s
Ponovljivost	±0,03 mm
Komunikacija	TCP USB 2.0, USB 3.0 Ethernet/IP
Napajanje	100-240 VAC, 47-440 Hz
Raspon temperatura	0-50°C

3.2. Ispitivanje komunikacije robotske ruke i stereovizijskog sustava u *RoboDK* okolišu

3.2.1. Računanje potrebnih transformacija

Kako bi se svi alati mogli uspješno prikazati u *RoboDK* i *RViz*-u potrebno je izračunati transformacije koje prikazuje Slika 11.



Slika 11. Prikaz odnosa koordinatnih sustava alata za navođenje robota

Stereovizijski sustav kao informaciju daje matrice transformacija K_{pom}, K_{stat} i K_{sty} koje redom predstavljaju položaj pomičnog krutog alata, statičnog krutog alata i *stylus*-a u koordinatnom sustavu stereovizijskog sustava. Kako bi uspješno prikazali navedene alate u željenim okruženjima potrebno je izračunati položaje pomičnog krutog tijela i *stylus*-a u koordinatnom sustavu statičnog krutog tijela. Potrebne matrice T_{pom} i T_{sty} mogu se izračunati pomoću izraza (1) i (2).

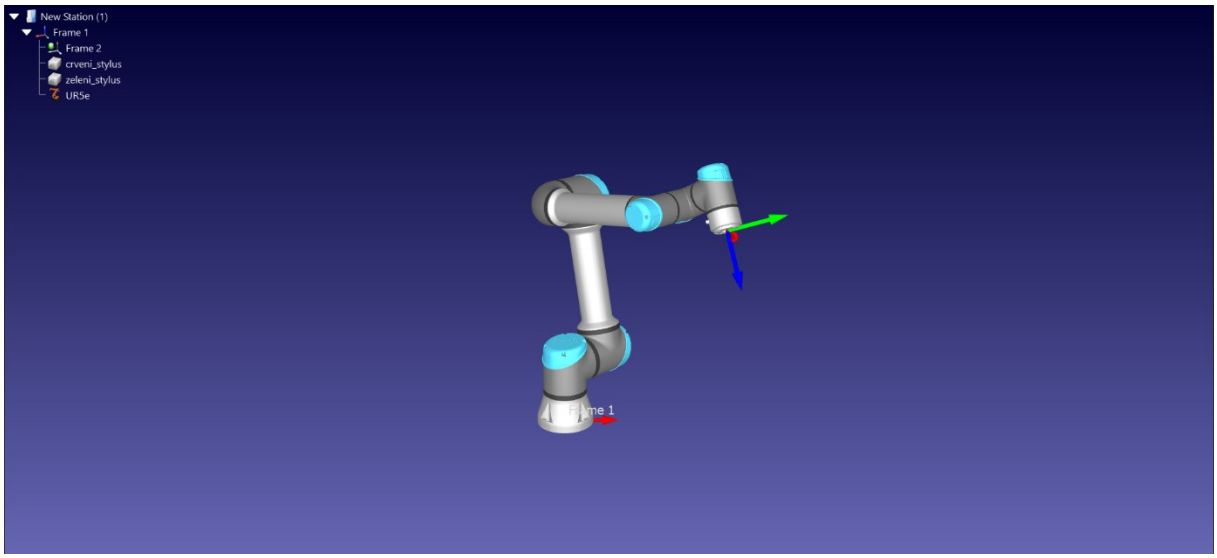
$$T_{pom} = {K_{stat}}^{-1} \times K_{pom} \quad \textbf{(1)}$$

$$T_{sty} = {K_{stat}}^{-1} \times K_{sty} \quad \textbf{(2)}$$

3.2.2. Simulacija u RoboDK

Prije nego se krene na integraciju robota i stereovizijskog optičkog sustava potrebno je na simuliranoj robotskoj ruci provjeriti rad optičkog sustava. *RoboDK* je najrašireniji izvanmrežni program za programiranje i simulaciju robota. Simulacija treba provjeriti domet stereovizijskog sustava, rad svih alata kao i planirane pokrete robotske ruke u pojedinim položajima alata. Simulacija je neizbježan dio rada kako bi se uspješno izbjegli neočekivani pokreti robotske ruke.

Kako bi simulacija bila izvediva napravljena je *Python* skripta *test_vega.py* unutar koje je izrađena veoma jednostavna kontrola tijeka programa. Ako korisnik pritisne tipku „space“ transformacija između *stylus*-a i statičnog krutog tijela trebala bi odgovarati transformaciji prihvatnice robota i njegove baze. Ovaj korak bio je zaslužan kako bi se potvrdile dobre rotacije koordinatnog sustava *stylus*-a kao i mogućnost komunikacije optičkog sustava i računala.



Slika 12. UR5e - RoboDK

Unutar *RoboDK* okruženja napravljena je uspješna i jednostavna simulacija robotske ruke kako bi se testirala mogućnost integracije Polaris Vega ST stereovizijskog optičkog sustava i računala.

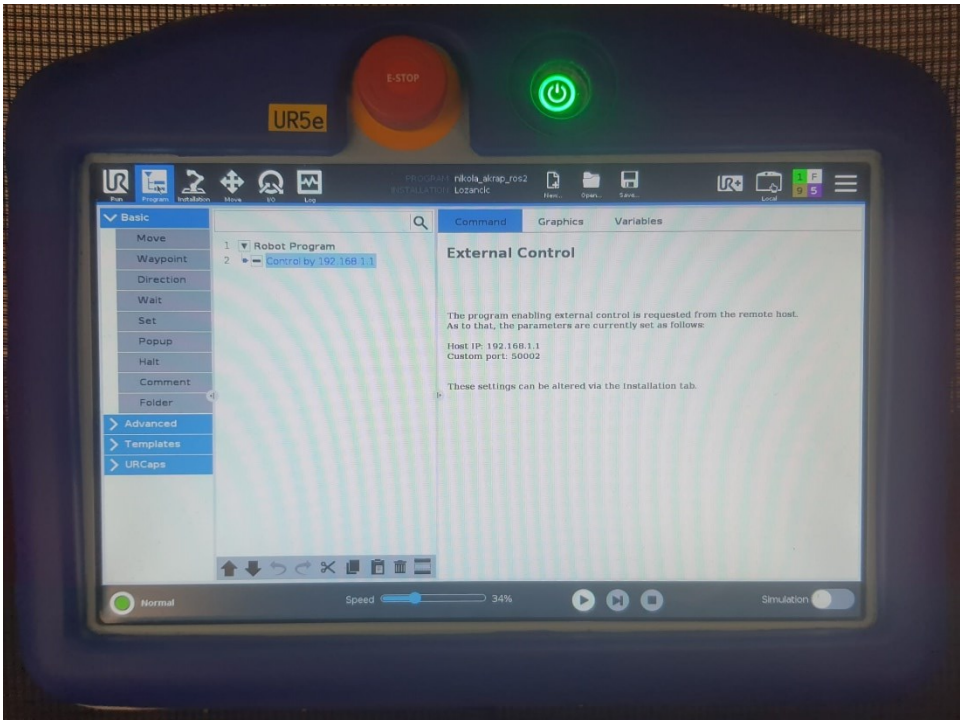
3.3. Problematika integracije robotske ruke sa stereovizijskim sustavom

Idući korak je integracija robotske ruke i stereovizijskog sustava. Pošto robot mora imati nesmetanu komunikaciju sa stereovizijskim optičkim sustavom za praćenje mora se odabrat najbolji način za njihovo povezivanje i komunikaciju.

Za izvođenje rada odabran je okoliš Ubuntu 22.04 kernela koji podržava mogućnost rada u stvarnom vremenu, a na kojega je instaliran ROS2 operativni sustav. ROS2 je logično rješenje kod pristupanja ovakvom radu zbog njegove odlične podrške za rad u stvarnom vremenu, velike sigurnosti i pouzdanosti i jako razvijene arhitekture koja uključuje mnogobrojne korisne pakete. Sve *Python* skripte, kao i svi algoritmi koji se budu spominjali u radu bit će primijenjeni unutar ROS2 operativnog sustava.

3.4. Podešavanje robota i integracija u ROS2 okruženje

Kako bi se omogućila integracija robota u ROS2 okruženje potrebno je povezati robota s računalom i napraviti novi program za vanjsko upravljanje. Robota se isto kao i kameru s računalom povezuje preko TCP/IP protokola čiji je postupak detaljnije opisan u poglavlju 5.2. Na robotskoj upravljačkoj ploči tada je potrebno izraditi novi program, a sama upravljačka ploča mora biti na minimalnoj PolyScope[2] verziji 5.1. U novom programu potrebno je instalirati



Slika 13. UR5e - upravljačka ploča

[2] PolyScope je vodeći software za upravljanje kolaborativnim robotima

„External control URCap“ kojega tada povezujemo s IP adresom računala i dajemo mu „port“ preko kojega će komunicirati s računalom.

Računalo mora pokretati „real-time“ kernel za Ubuntu 22.04 radi najbrže komunikacije s robotom. Taj je detalj neizbježan jer će se cijeli sustav primjenjivati u medicini gdje poruke moraju teći što brže radi sprječavanja eventualnih nezgoda.

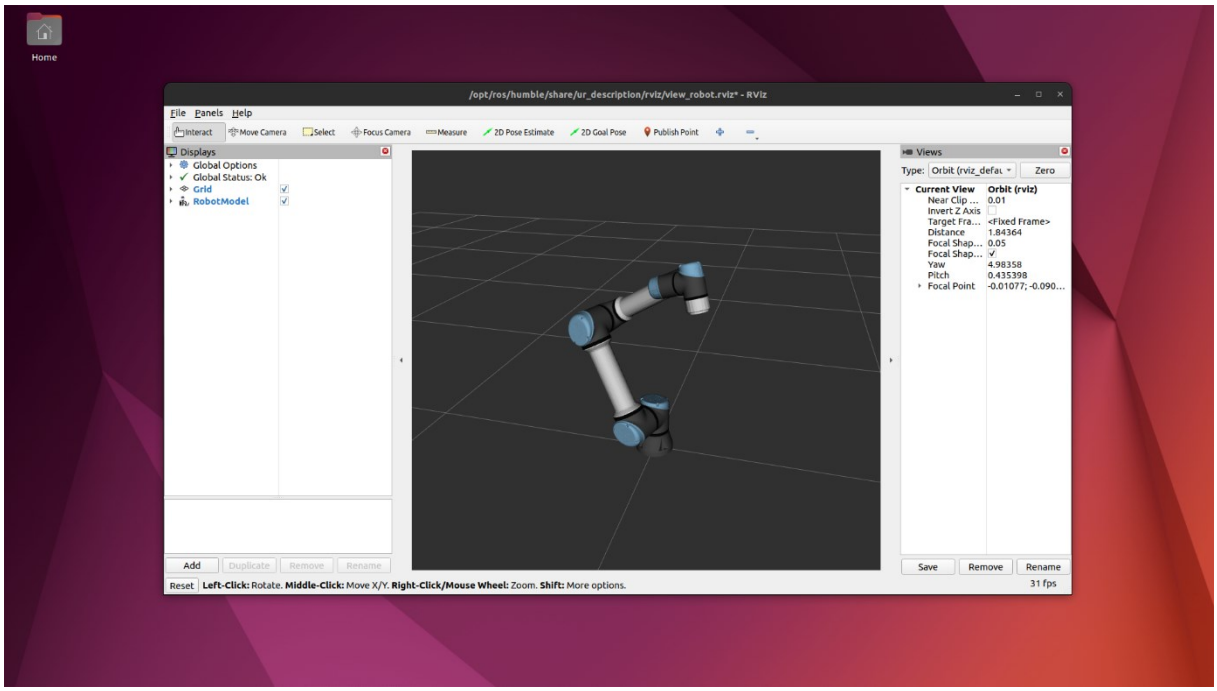
Za izvođenje ovog rada odabran je ROS2 operativni sustav zbog njegove sve veće uporabe u industriji. Sama integracija UR5e u ROS2 okruženje u ovom radu može poslužiti kao olakšanje u budućim primjenama u ovom području istraživanja. Kako bi se robotu omogućilo primanje naredbi s računala potrebno je pokrenuti učitani program na upravljačkoj ploči. Aktivacijom programa robot može primati naredbe samo sa strane računala dok se svi ostali načini upravljanja deaktiviraju čime se povećava faktor sigurnosti od vanjskih utjecaja na aktivnosti što je bitan dio bilo kakvog rada u medicinskim okruženjima.

Za slanje naredbi robotu potrebno je preuzeti službene UR drivere *„ur_robot_driver“* i pripadajuće pakete s pomoću kojih ROS2 može slati naredbe robotu. [10]

Driver podržava neke ključne funkcionalnosti kolaborativnih robota, kao što su:

- Pauza pri zaustavljanju u nuždi
- Zaštitno zaustavljanje
- Automatsko smanjenje brzine kako bi se izbjeglo kršenje sigurnosnih postavki
- Ručno smanjenje brzine putem upravljačke ploče

Kako bi paketi bili dostupni za uporabu potrebno ih je instalirati u radni prostor s pomoću naredbi *„colcon build“* i *„source“*. Nakon uspješne instalacije paketa mogu se pokrenuti sve „launch“ datoteke, *Python* skripte i ROS2 čvorovi koji omogućuju upravljanje robotom. Pokretanjem naredbe *„ros2 launch ur_robot_driver ur_control.launch.py ur_type:=ur5e robot_ip:=192.168.1.10“* robot će se prikazati u *„RViz2“* gdje se uživo može pratiti konfiguracija robota kao i svi njegovi pokreti. Detaljniji opis ROS2, njegovih paketa, mogućnosti kao i cijelog radnog okruženja bit će obrađen u poglavlju 4.



Slika 14. UR5e - prikaz u *RViz*-u

4. PREDUVJETI ZA IZRADU

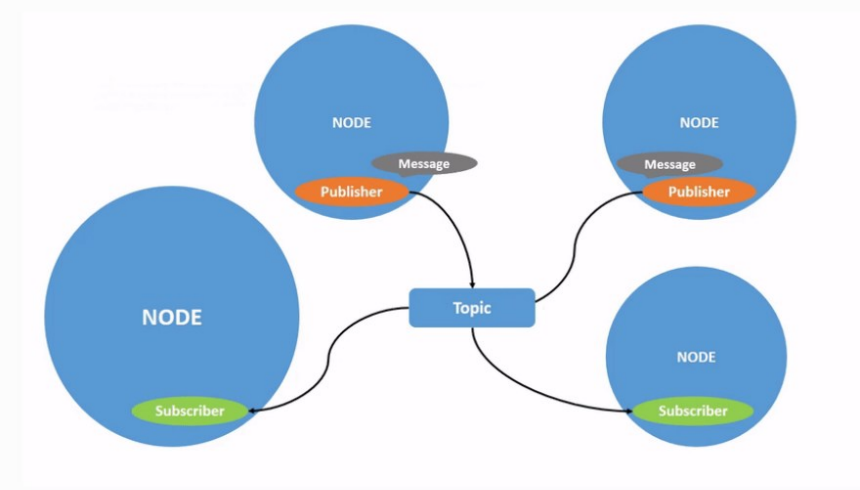
4.1. ROS2 i MoveIt2

The Robot Operating System (ROS2) je *open-source* skup softverskih knjižnica i alata za izradu robotskih aplikacija. Zbog njegove velike proširenosti u svim krugovima, od stručnjaka do studenata, idealan je alat za izradu robotičkih rješenja koja se mogu konstantno nadograđivati i poboljšavati. U ovom radu koristi se izdanje ROS2 Humble jer je to trenutno najrazvijenije i najstabilnije izdanje za uporabu svih potrebnih paketa koji se koriste u izradi. ROS2 pokreće se na Ubuntu 22.04 pošto je to jedina verzija koja podržava navedeno izdanje ROS2.

MoveIt2 je platforma za robotsku manipulaciju za ROS 2 i uključuje najnovija dostignuća u planiranju gibanja, manipulaciji, 3D percepciji, kinematici, kontroli i navigaciji. S pomoću ove platforme u radu će biti omogućeno upravljanje robotom. Zbog potrebe za najboljim mogućim „real-time“ pokretima robota koristit će se *servo* koji omogućuje izravno slanje brzina prihvatnici robota uz najmanje kašnjenje podataka koji se šalju robotu.

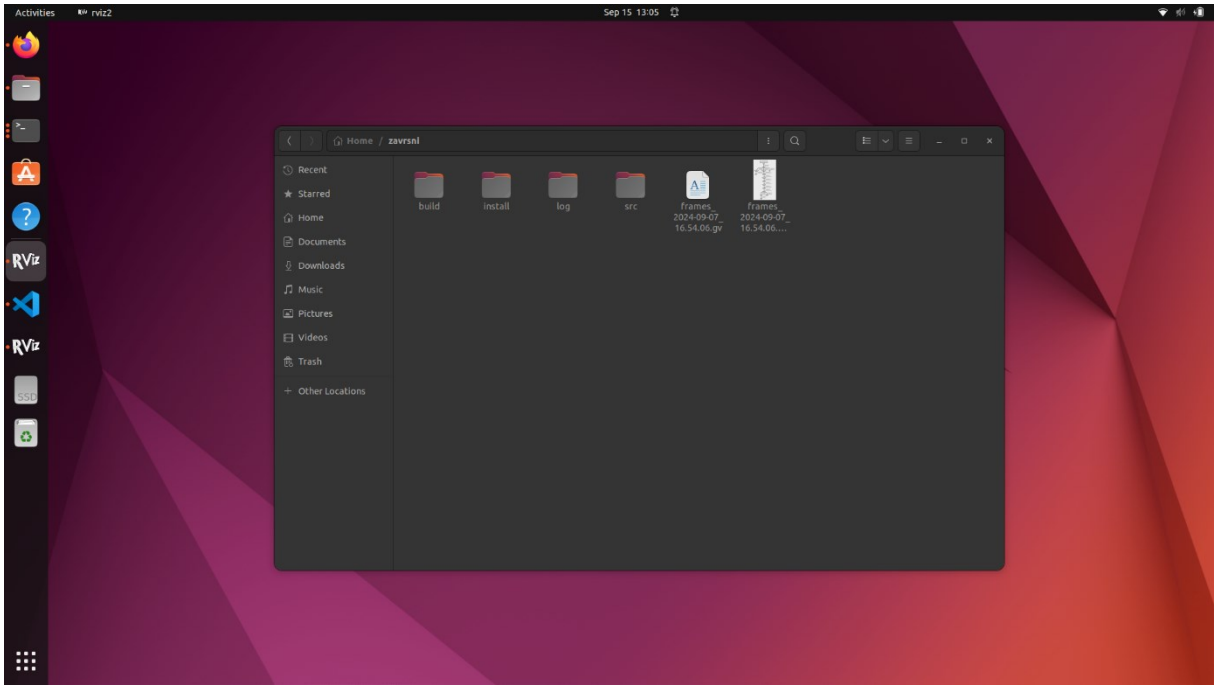
4.1.1. Principi rada ROS2

U svojoj osnovi, ROS2 sastoji se od mnoštva zasebnih čvorova (engl. *Node*) koji kontinuirano međusobno komuniciraju. Komunikacija radi na principu pošiljatelja (engl. *Publisher*) i pretplatnika (engl. *Subscriber*). Pošiljatelj prima podatak od čvora i objavljuje ga na neku temu (engl. *Topic*) nakon čega tema prenosi taj podatak pretplatniku koji ga šalje u drugi čvor. Čvor može objavljivati podatke na neograničen broj tema i istovremeno biti pretplaćen na neograničen broj tema.



Slika 15. ROS2 – komunikacija[11]

Cijeli proces komunikacije odvija se u stvarnom vremenu, a kako bi različiti paketi mogli međusobno komunicirati potrebno je napraviti radno okruženje. Za ovaj rad izrađeno je radno okruženje „zavrsni“.



Slika 16. Radno okruženje "zavrsni"

4.2. Namještanje radnog okruženja

Radno okruženje može se generirati iz terminala s pomoću naredbe *„mkdir -p /zavrsni/src“*. Unutar direktorija pozivanjem ove naredbe stvara se poddirektorij „src“ u kojemu će biti instalirani svi potrebni paketi koji će omogućiti gibanje robota u stvarnom vremenu. Uz sve postojeće pakete bit će potrebno generirati novi paket proizvoljnog imena *„kamera_tf2_py“* unutar kojega će se nalaziti glavne *Python* skripte i glavna „launch“ skripta pomoću koje će se pozivati sve izrađene skripte. Novi paket generira se naredbom *„ros2 pkg create –build-type ament_python –license Apache-2.0 kamera_tf2_py“*. Pokretanjem naredbe generiraju se i sve potrebne datoteke. Tada se poziva naredba *„colcon build“* koja izgrađuje radno okruženje tako da instancira svaki kod i pregleda koji paketi ovise o drugima. Prije nego što se pokrenu čvorovi unutar paketa potrebno je izvršiti naredbu *„source install/setup.bash“* koja će sve što je naredba *„colcon build“* izgradila dodati u putanju (engl. *Path*). Ove naredbe moraju se izvršiti nakon svake promjene koja se napravi unutar bilo kojeg od paketa koji se nalaze u radnom okruženju. Za pravilnu izvedbu ovog rada potrebno je instalirati još neke pakete i knjižnice koji neće biti instalirani u radno okruženje nego izravno u izvorni direktorij ROS2 Humblea.

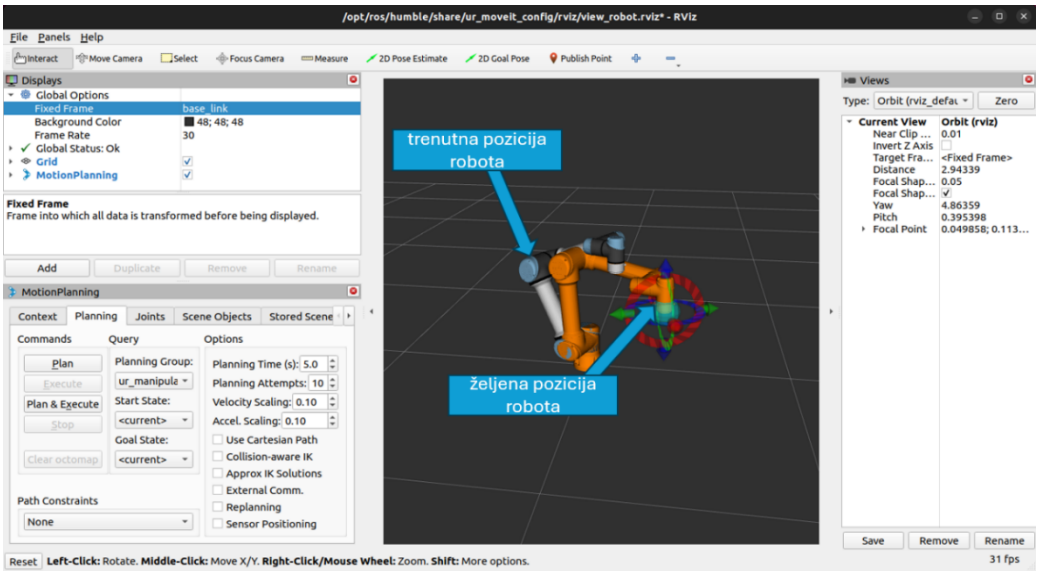
4.2.1. Knjižnica TF2

TF2 je druga generacija poznate knjižnice koja je prvotno izašla kao dodatak prvoj verziji ROS operativnog sustava, a predstavlja knjižnicu transformacija u ROS2. Održava odnos između koordinatnih sustava u strukturi stabla koja je vremenski međuspremana i omogućuje korisniku da transformira točke između bilo koja dva koordinatna sustava u bilo kojem željenom trenutku. Knjižnica se može instalirati pokretanjem naredbe *„sudo apt-get install ros-humble-tf2-ros“*. Kako bi se omogućilo upravljanje s pomoću *Python* skripti potrebno je instalirati dodatne alate koje *Python* skripte koriste što se može obaviti pokretanjem naredbe *„sudo apt-get ros-humble-tf2-tools“*. Koristeći instalirane alate u *Python* skriptama možemo u stvarnom vremenu dobivati odnose koordinatnih sustava alata koje koristimo i vidjeti njihov prikaz u *RViz2*. Koristeći ovu knjižnicu možemo na jako jednostavan način dobiti uvid u načine ponašanja pojedinih koordinatnih sustava u stvarnom vremenu. Na slici prikazani su samo najbitniji koordinatni sustavi robota i markera na kojima će se bazirati ovaj rad a to su „base_link“, „tool0“, „base_link_marker“, „stylus“ i „kost“.

4.2.2. Paket „ur_robot_driver“

Ur_robot_driver službeni je driver za sve Universal Robots kolaborativne robote za ROS2. Neke od novih značajki omogućene su s pomoću ROS2 i uključuju smanjenu latenciju, poboljšanu sigurnost i veću fleksibilnost u pogledu konfiguracije middlewarea. Paket sadrži „launch“ datoteke koje omogućuju brzo pokretanje i korištenje upravljačkog programa kao samostalne verzije ili u kombinaciji s MoveIt2 platformom. Paket se može instalirati kloniranjem GitHub repozitorija u kojem slučaju je potrebno odabrati stablo koje odgovara korištenoj ROS2 verziji. Jednostavniji način kojim se smanjuje rizik od greške nastale u kloniranju repozitorija je da se paket instalira pokretanjem naredbe *„sudo apt-get install ros-humble-ur“*. U ovom radu paket je instaliran izravno u ROS2 središnji direktorij, a ne u radno okruženje „zavrsni“ zbog moguće potrebe uporabe paketa za neke druge projekte. Nakon što je paket instaliran potrebno je pripremiti robota za rad. Više o namještanju programa na upravljačkoj ploči opisano je u poglavlju 3.4.

Podrška za integraciju robota u MoveIt2 platformu je ugrađena u drivere. Pokretanjem naredbe *„ros2 launch ur_moveit_config ur_moveit.launch.py ur_type:=ur5e launch_rviz:=true”* pokrenuti će se novi prozor u *RViz2* u kojem možemo vidjet dva preklopljena prikaza robota. Prvi prikaz prati robota u stvarnom vremenu dok drugi prikaz možemo sami pomicati pritiskom na kuglicu koja se nalazi na prihvatnici i njenim pomicanjem namjestiti poziciju u koju želimo da se robot preseli. Ovakvo planiranje neće se koristit u radu, ali MoveIt2 je potreban zbog svojih mogućnosti *servo* upravljanja robotom.



Slika 17. UR5e - prikaz u MoveIt2

MoveIt2 robotom upravlja tako da šalje naredbe u obliku brzina i pozicija na čvorove na koje su pretplaćeni kontroleri robota. Ovisno o vrsti poslane informacije, aktivira se odgovarajući kontroler. Kontroleri su dio drivera i nije ih potrebno instalirati odvojeno. Više o kontrolerima govorit će se u poglavlju 5.3.3.

4.2.3. Paket „pymoveit2“

Već spomenuti detalj je da MoveIt2 platforma ima mogućnost *servo* upravljanja robotom. Kako bi se upravljanje pojednostavilo u radu je bilo potrebno instalirati *„pymoveit2“ Python* paket. Paket sadrži funkcije s pomoću kojih robota možemo upravljati slanjem željene konačne poze prihvatnice robota ili slanjem brzina kojima želimo da se zglobovi pomiču. U prvom slučaju robot samostalno računa inverznu kinematiku i odabire najbolju putanju za dolazak u željenu pozu. U ovom radu fokus je stavljen na slanje brzina robotu takozvanim *servo* upravljanjem. U skriptama je potrebno pozvati klasu MoveIt2Servo koja omogućuje uporabu svih potrebnih funkcija.

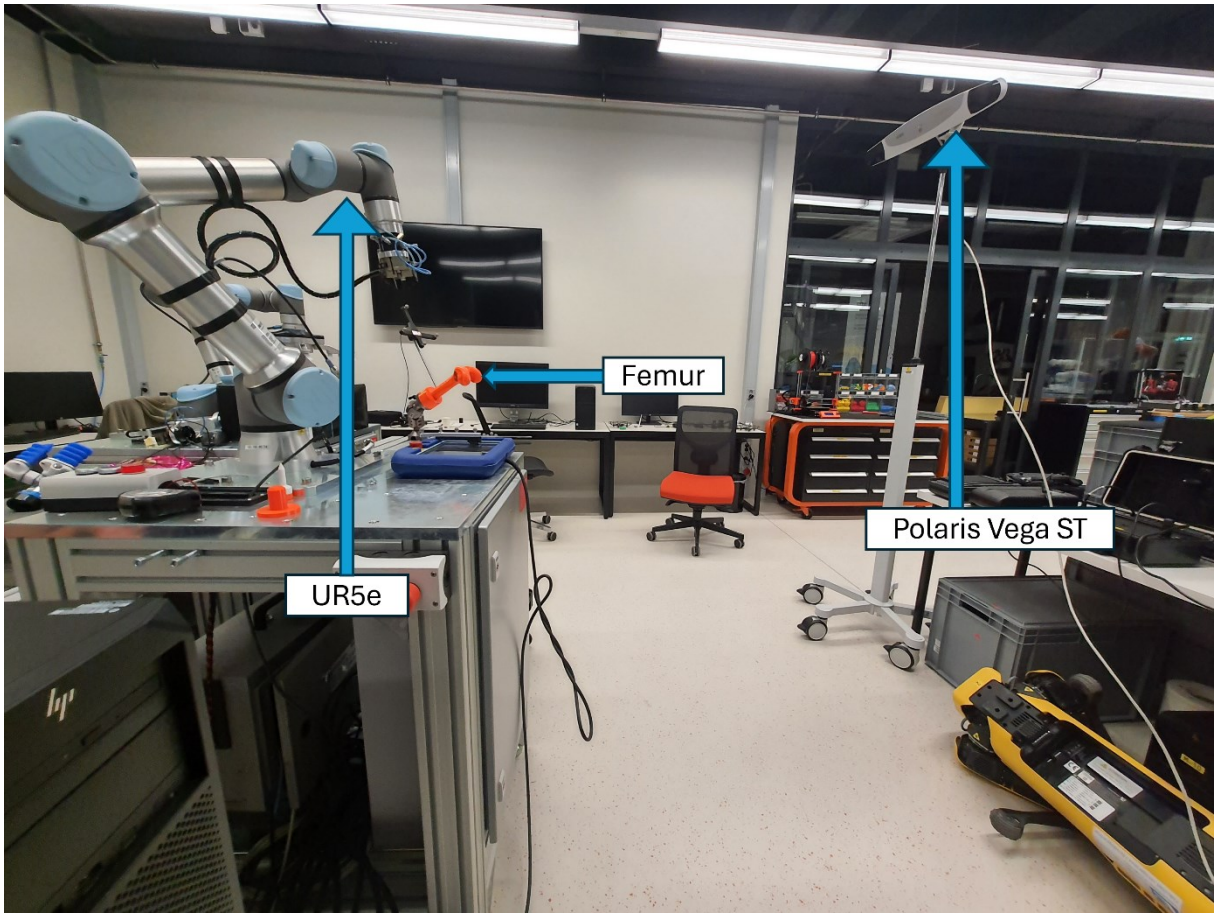
4.2.4. Paket „scikit-surgerynditracker“

Kako bi omogućili primanje informacija od strane Polaris optičkog sustava potrebno je instalirati *Python* paket *„scikit-surgerynditracker“*. Ovaj paket omogućava izravnu komunikaciju s optičkim sustavom putem *Pythona*. U skripti je potrebno unijeti postavke sustava koje uključuju točan sustav koji koristimo, sve .rom datoteke koje opisuju alate i markere i IP adresu sustava. Nakon toga pozivom funkcije *„start_tracking“* započinje prikupljanje informacija, a kontinuiranim pozivanjem funkcije *„get_frame“* dobivamo uvid u trenutačne matrice transformacija između alata i optičkog sustava.

5. POSTAVLJANJE EKSPREIMENTALNOG OKRUŽENJA

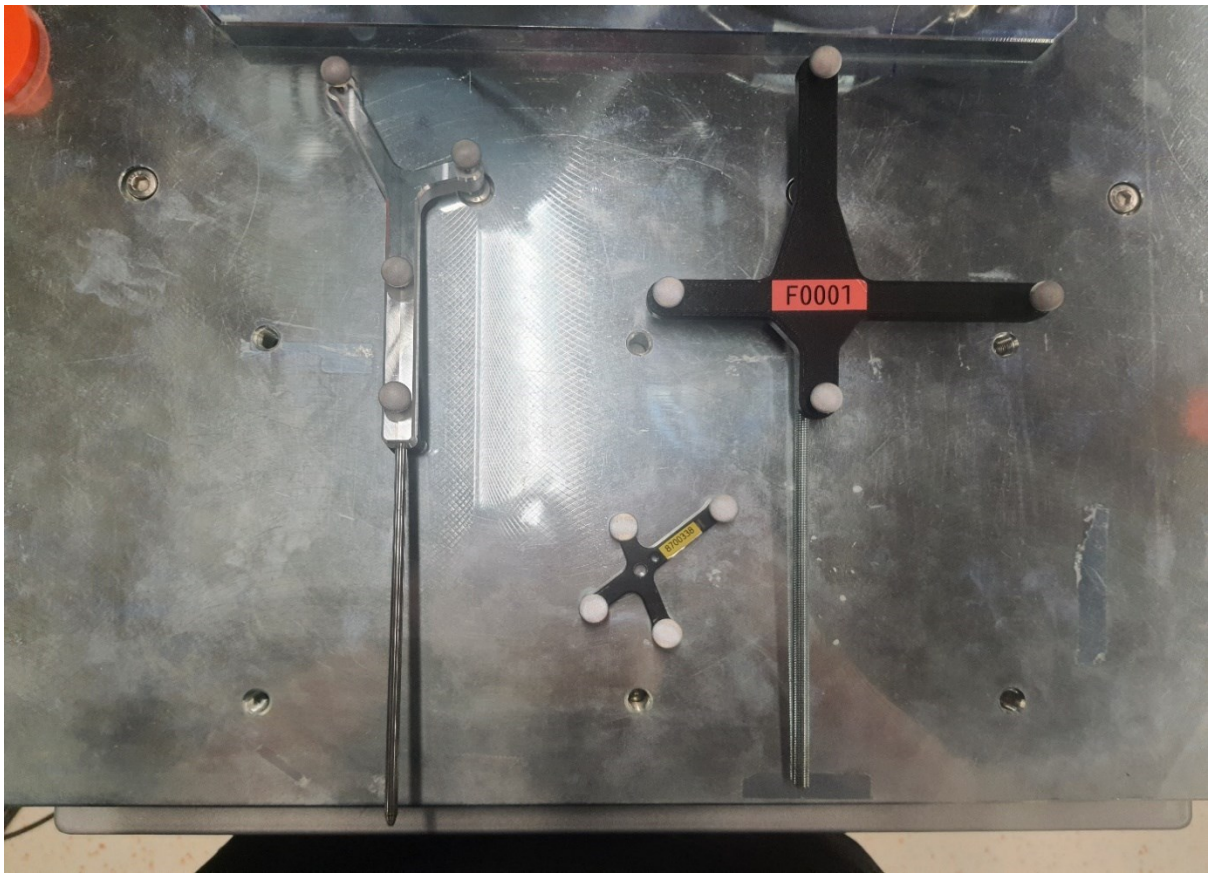
Nakon uspješne integracije paketa u ROS2 potrebno je postaviti eksperimentalno okruženje. Ovaj proces uključuje postavljanje opreme na odgovarajuće pozicije unutar laboratorija, potrebne matematičke proračune, mjerenja i pisanje svih potrebnih *Python* skripti.

Oprema koja se koristila u ovom radu uključuje Polaris Vega ST optički stereovizijski sustav, tri alata s pasivnim markerima, UR5e kolaborativnu robotsku ruku i 3D printani model femura.



Slika 18. Eksperimentalno okruženje

Korišteni alati uključuju *stylus* koji služi za dovođenje robota na željenu poziciju gdje će se vršiti operativni zahvat te dva kruta tijela, jedno statično koje se koristi kao referentni koordinatni sustav i drugo pomično koje je cijelo vrijeme pričvršćeno za *femur* koji ima mogućnost gibanja. Slika 19 prikazuje navedene alate.



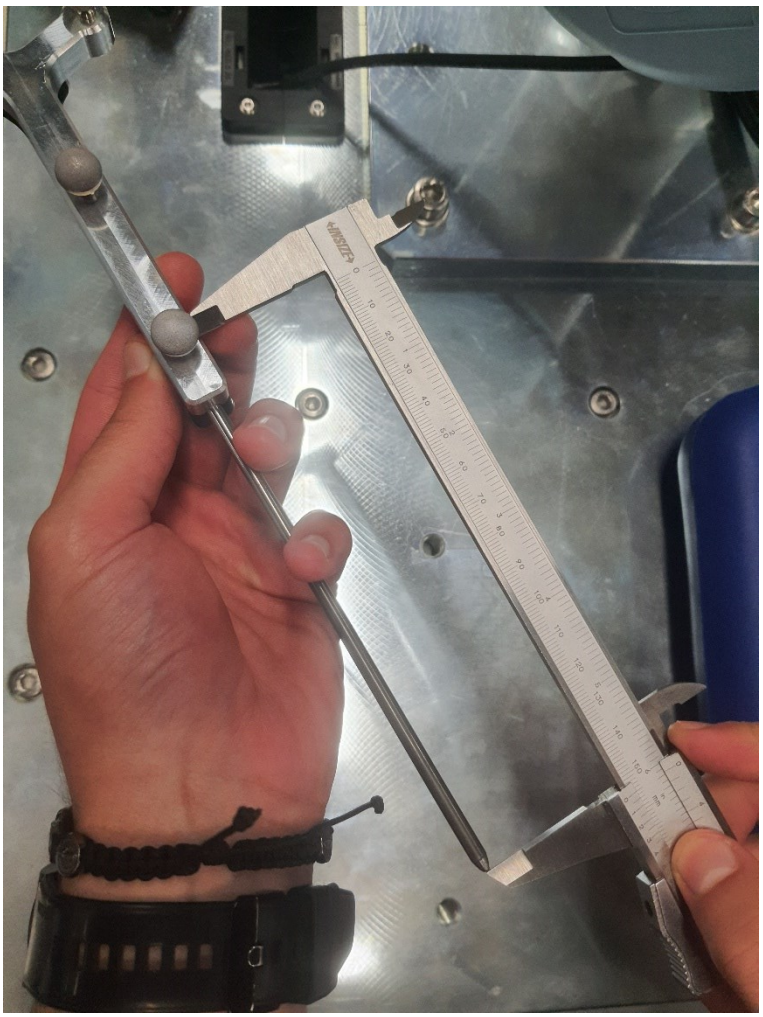
Slika 19. Alati eksperimentalnog okruženja

5.1. Namještanje koordinatnog sustava *stylus*-a

Prije početka rada s alatima najprije je potrebno namjestiti koordinatni sustav stylus-a.

Prvotno ishodište njegovog koordinatnog sustava nalazi se u markeru koji je najbliže oštrici *stylus*-a, a za svrhe ovog rada potrebno ga je premjestiti u vrh oštrice. Kako bi to bilo izvedivo potrebno je izračunati potrebne rotacije i transformacije kako bi koordinatni sustav uspješno doveli na željeno mjesto. Iz CAD modela alata može se doći do udaljenosti između centra kuglice i središta presjeka oštrice, ali budući da je oštrica izrađivana u laboratoriju njezinu duljinu potrebno je izmjeriti ručno. Mjerenje je obavljeno pomičnim mjerilom čime je dobivena matrica translacije T. Postupak mjerenja prikazan je na Slika 20.

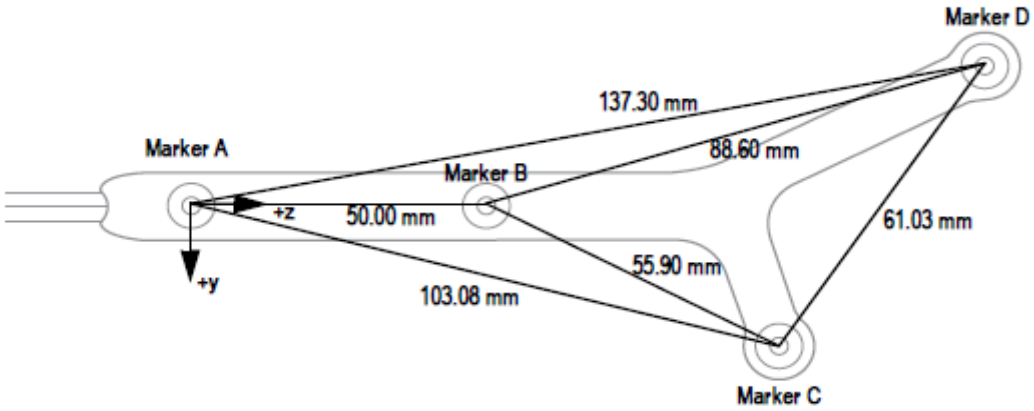
$$T = \begin{bmatrix} 1 & 0 & 0 & -19 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -154{,}717 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Slika 20. Mjerenje *stylus*-a mikrometrom

Potrebno je još izvršiti dodatne rotacije kako bi sustav bio ispravno orijentiran. Slika 21 pokazuje kako pozitivan smjer z-osi gleda u smjeru suprotnom od oštrice što za ovu primjenu nije prihvatljivo, a radi boljeg poklapanja *stylus*-a s robotom osima x i y bi također trebalo zamijeniti mjesta. Bitno je napomenuti kako x-os izlazi iz ravnine papira. Kako bi se ostvarila željena rotacija potrebno je zarotirati koordinatni sustav oko potrebno je zarotirati koordinatni sustav oko y-osi za 180 stupnjeva i naknadno oko z-osi za 90 stupnjeva. Navedene rotacije redom pokazuju matrice R_y i R_z.

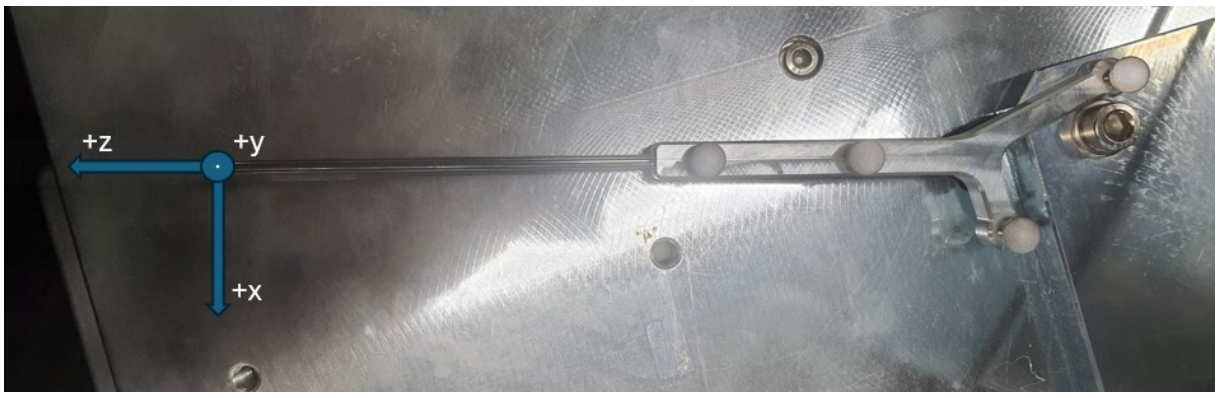
$$R_y = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad R_z = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Slika 21. Prvotni koordinatni sustav *stylus*-a

Kako bi se koordinatni sustav premjestio na željenu poziciju potrebno je redom pomnožiti matrice R_y, R_z i T. Postupak množenja pokazuje jednadžba (3), gdje je T_{kon} konačna matrica transformacije.

$$T_{kon} = R_y \times R_z \times T = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & -19 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -154{,}717 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & -19 \\ 0 & 0 & -1 & 154{,}717 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$



Slika 22. Transformirani koordinatni sustav *stylus*-a

5.2. Povezivanje TCP/IP protokolom

TCP i IP su odvojeni protokoli koji surađuju kako bi osigurali da se podaci isporuče na predviđeno odredište unutar mreže. IP dobiva i definira adresu aplikacije ili uređaja kojem se podaci moraju poslati. TCP je tada odgovoran za transport i usmjeravanje podataka kroz mrežnu arhitekturu te osigurava da podaci budu dostavljeni na odredišnu aplikaciju ili uređaj koji je IP definirao. Za nesmetanu komunikaciju robot, stereovizijski optički sustav i računalo spajaju se putem mrežnog preklopnika (engl. *network switch*).

Odabrane su proizvoljne IP adrese za svaki uređaj pa tako računalo ima adresu *192.168.1.1*, stereovizijski optički sustav *192.168.1.5*, a robot *192.168.1.10*.

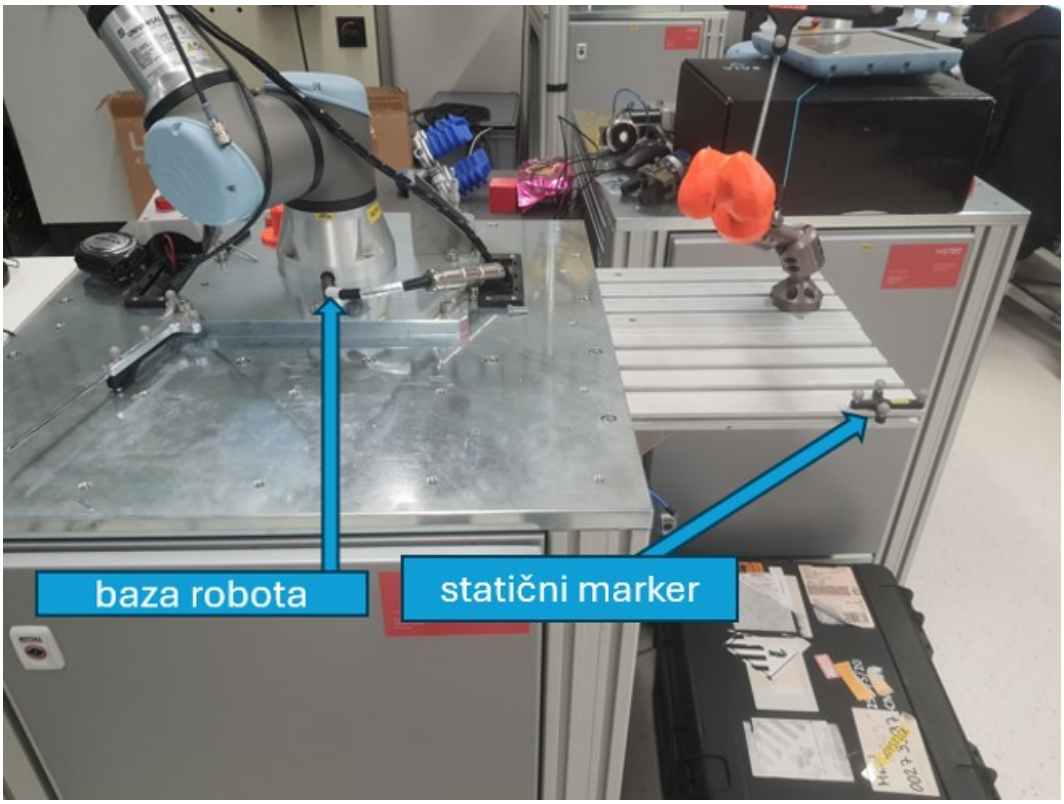
5.3. Implementacija ROS2

U poglavlju 4 navedeni su svi potrebni preduvjeti kako bi se zadatak mogao izvršiti. U ovom poglavlju u detalje će se objasniti svi provedeni postupci unutar ROS2 operativnog sustava koji omogućavaju provedbu zadatka, a središte interesa bit će *Python* skripte unutar kojih se vrše sve radnje. Unutar radnog okruženja „zavrsni“ izrađen je paket „kamera_tf2_py“ koji sadrži četiri *Python* skripte i jednu „launch.py“ datoteku.

5.3.1. Prikaz u RViz-u

Pokretanjem optičkog sustava u *Rviz*-u će se pokazati koordinatni sustavi alata. Njihove međusobne relacije dobro su prikazane no javlja se problem pozicije koordinatnih sustava u odnosu na robota. Pošto je glavni koordinatni sustav onaj od kamere svi koordinatni sustavi alata nalazit će se ispod mreže *RViz*-a dok će robot biti iznad nje što može izazvati probleme kod računanja transformacija prihvatnice robota koji mogu dovesti do nesreća. Problem se može riješiti objavljivanjem statičnog koordinatnog sustava „base_link_marker“ koji će u *RViz*-u predstavljati statično kruto tijelo. Kako bi se to izvelo potrebno je napraviti skriptu *static_marker.py* u kojoj se s pomoću objekta *StaticTransformBroadcaster RViz*-u šalje transformacija statičnog koordinatnog sustava u odnosu na bazu robota, a čija transformacija odgovara stvarnoj transformaciji između baze robota i statičnog alata. Također, potrebno je osigurati da se osi koordinatnog sustava u *RViz*-u i onoga u laboratoriju preklapaju. Transformacija je dobivena jednostavnim mjerenima postolja robota te je dobiven sljedeći kvaternion:

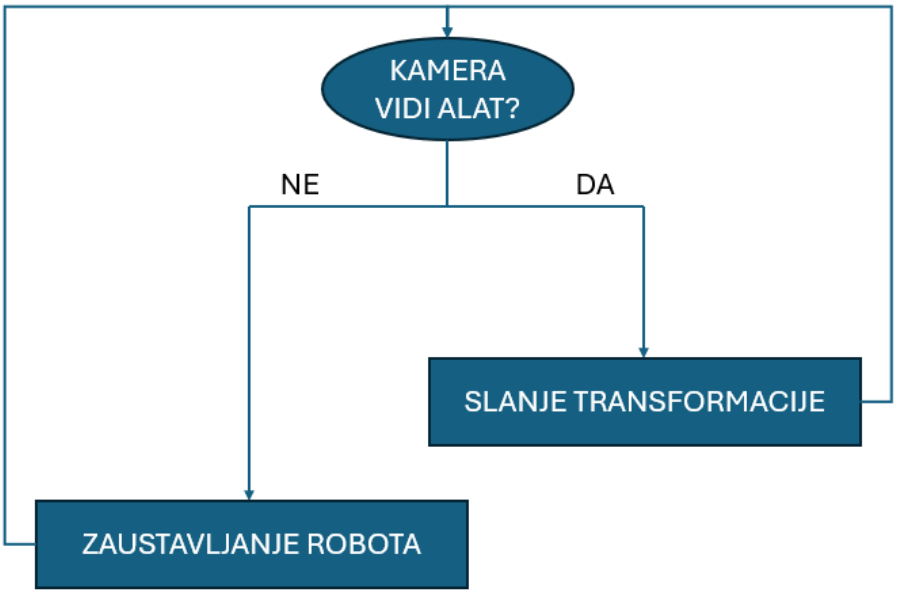
$$q_{stat} = [0.575 \quad -0.34 \quad -0.02 \quad 0.707 \quad 0 \quad 0.707 \quad 0]$$



Slika 23. Lokacija statičnog markera u odnosu na bazu robota

5.3.2. Planiranje gibanja

Planiranje gibanja robota omogućit će skripta *kamera_tf2.py* u kojoj se provodi inicijalizacija postavki kamere kao i potrebnih paketa. Potrebno je pozvati objekt *TransformBroadcaster* iz paketa *TF2*. Njegova uloga je slanje transformacije između dva koordinatna sustava koja se nalaze unutar istog stabla.



Slika 24. Dijagram toka lokalizacije alata

Nakon što je transformacija definirana i objavljena njoj može pristupiti bilo koji čvor koji je u tom trenutku pokrenut. U slučaju ovog rada to će biti čvor */move_robot* koji će tu transformaciju iskoristiti kako bi robotu slao ispravne brzine zglobova. Skripta objavljuje pozicije stylusa i pomičnog krutog tijela u odnosu na statično kruto tijelo. Uvođenjem potrebnih transformacija iz poglavlja 5.1 omogućit će se pravilan prikaz koordinatnih sustava u *RViz*-u. Objavljivanje transformacija vrši se u petlji kako bi se omogućilo *real-time* praćenje promjena poza svih alata i robota. U slučaju da kamera ne vidi alat javit će se problem gdje *RViz* pamti posljednju transformaciju i drži ju u memoriji sve dok kamera opet ne primi informaciju o lokaciji alata. U toj situaciji može doći do neželjenog micanja robota što može dovesti do nesreće. Kako bi se izbjegao taj problem jednostavnom uvjetnom *if* petljom može se narediti *RViz*-u da u tom slučaju objavljuje da se alat nalazi u prihvatnici robota. Time se postiže da je robot automatski na željenoj lokaciji, zglobovima se ne šalju brzine i robot se zaustavlja u trenutku. To je poželjna osobina programa jer se u medicinskom okruženju želi eliminirati bilo kakav nagli pokret robota ili nemogućnost zaustavljanja.

5.3.3. Provedba gibanja

Za ispravnu izvedbu gibanja nadređena je MoveIt2 platforma. Gibanje izravno prati podatke koje objavljuje *TransformPublisher* u skripti *kamera_tf2.py*, a omogućeno je u skripti *move_robot.py* koja se bazira na paketu *pymoveit2* i njegovom objektu *MoveIt2Servo*. *Servo* omogućuje istovremeno praćenje pozicije prihvatnice i slanje pravocrtne i kutne brzine. Za ispravno pozivanje *MoveIt2Servo* objekta potrebno je unijeti određene podatke. Oni uključuju referentni koordinatni sustav za *servo*, translacijsku i kutnu brzinu te ime grupe zglobova u *RViz*-u. Te vrijednosti su „tool0“, 2.0 m/s, 2.0 rad/s te „ur_manipulator“. U dijelu skripte gdje se vrši inicijalizacija čvora također je potrebno aktivirati *servo* pozivom funkcije *„servo.enable()“*.

Robot naredbe za micanje dobiva preko svojih kontrolera. Kako bi slanje brzina imalo utjecaj na stvarno micanje robota potrebno je podesiti konfiguracijske datoteke robota i aktivirati *forward_velocity_controller* koji je po zadanim postavkama isključen. Kako bi se kontroler ispravno aktivirao potrebno je otići u *launch* datoteku paketa *ur_robot_driver* naziva *ur_control.launch.py* gdje treba pronaći postavku *„initial_joint_controller“* u kojoj je potrebno zamijeniti kontrolere. Po zadanim postavkama to je *scaled_joint_trajectory_controller*, a treba biti *forward_velocity_controller*. Slijedeći isječak pokazuje navedenu promjenu controllera unutar *ur_control.launch.py* skritpte:

```
declared_arguments.append(
    DeclareLaunchArgument(
        "initial_joint_controller",
        default_value="forward_velocity_controller",
        description="Initially loaded robot controller.",
    )
)
```

Prije nego je robot uspješno pripremljen za primanje naredbi za gibanje potrebno je urediti njegovu *servo* konfiguracijsku datoteku naziva *ur_servo.yaml*. U njoj je između ostaloga definirano prima li robot naredbe pozicije, brzine ili akceleracije kao i koji čvor kontroler upravlja tim naredbama i na koje čvorove dolaze naredbe pozicije i brzine. Po zadanim postavkama robot prima samo naredbe pozicije preko čvora kontrolera */forward_position_controller/commands* koje se šalju na čvor */delta_joint_cmds*.

Potrebno je izvršiti izmjene prema kojima robot prima samo naredbe brzine preko čvora kontrolera */forward_velocity_controller/commands* koje se šalju na čvor */delta_twist_cmds*.

Kada se naprave svi potrebni ispravci može se krenuti na izradu samog algoritma unutar *move_robot.py* skripte. Slijedeći isječci prikazuju potrebne promjene unutar *ur_servo.yaml* datoteke:

```
cartesian_command_in_topic: ~/delta_twist_cmds
joint_command_in_topic: ~/delta_joint_cmds
joint_topic: /joint_states
status_topic: ~/status
command_out_topic: /forward_velocity_controller/commands
```

```
publish_joint_positions: false
publish_joint_velocities: true
publish_joint_accelerations: false
```

5.3.3.1. Izrada algoritma lokalizacije i registracije

Algoritam ima dva načina rada koji se aktiviraju i deaktiviraju pozivanjem ROS2 servisa (engl. *service*). Prvi način rada aktivira se automatski pokretanjem skripte i inicijalizacijom potrebnih objekata a on omogućava *real-time* praćenje *stylus*-a. Koristeći *TF2* funkciju *„lookup_transform()“* omogućena je povratna informacija o kontinuiranoj promjeni koordinatnog sustava *stylus*-a u odnosu na prihvatnicu robota. Pošto funkcija vraća kvaternione prva tri člana predstavljaju translaciju, a zadnja četiri rotaciju. Primjenom *Python*-ove knjižnice

scipy i njene funkcije *„as_euler()“* možemo lagano dobiti potrebne kutne brzine u obliku Eulerove notacije. Dobivene vrijednosti šalju se u funkciju *„servo()“* koja za argumente prima translacijsku i kutnu brzinu. Kako ne bi došlo do kolizije robota i *stylus*-a, a i radi očuvanja sigurne distance radi obavljanja potrebnih provjera prije izvođenja zahvata robot će se zaustaviti kada se nalazi na udaljenosti 10 cm od *stylus*-a. Cijelo vrijeme izvršava se i skripta *kamera_tf2.py* koja upravlja situacijama kada *stylus* nije vidljiv, a čiji je način rada opisan u poglavlju 5.3.2.

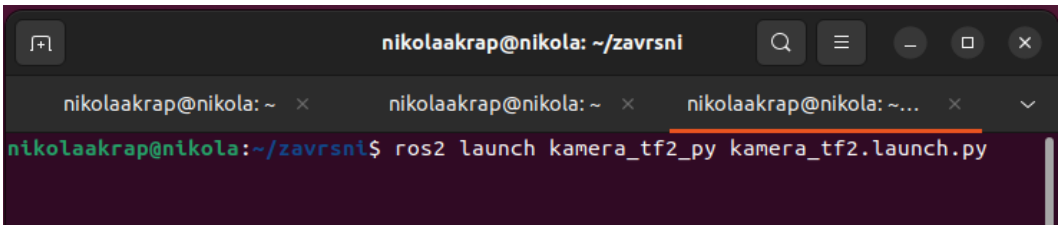
Pozivanjem ROS2 servisa koji su napravljeni u skripti može mijenjati vrijednosti parametara *self.stylus_tracking* i *self.bone_tracking*.

Ovi su parametri u inicijalizaciji skripte pozvani redom kao *True* i *False* vrijednosti. Pozivanjem prvog servisa parametar *self.stylus_tracking* prelazi u *False* i stvara se nova varijabla *self.last_known_transform* u koju se sprema transformacija *stylus*-a u odnosu na pomično kruto tijelo u trenutku pozivanja servisa. Spremanje transformacije omogućeno je izradom funkcije „*get_last_transform()“* koja u sebi poziva funkciju *„lookup_transform()“* o kojoj je bilo više govora u opisu prvog načina rada algoritma. Pozivanjem drugog servisa parametar *self.bone_tracking* prelazi u *True* čime započinje drugi način rada algoritma.

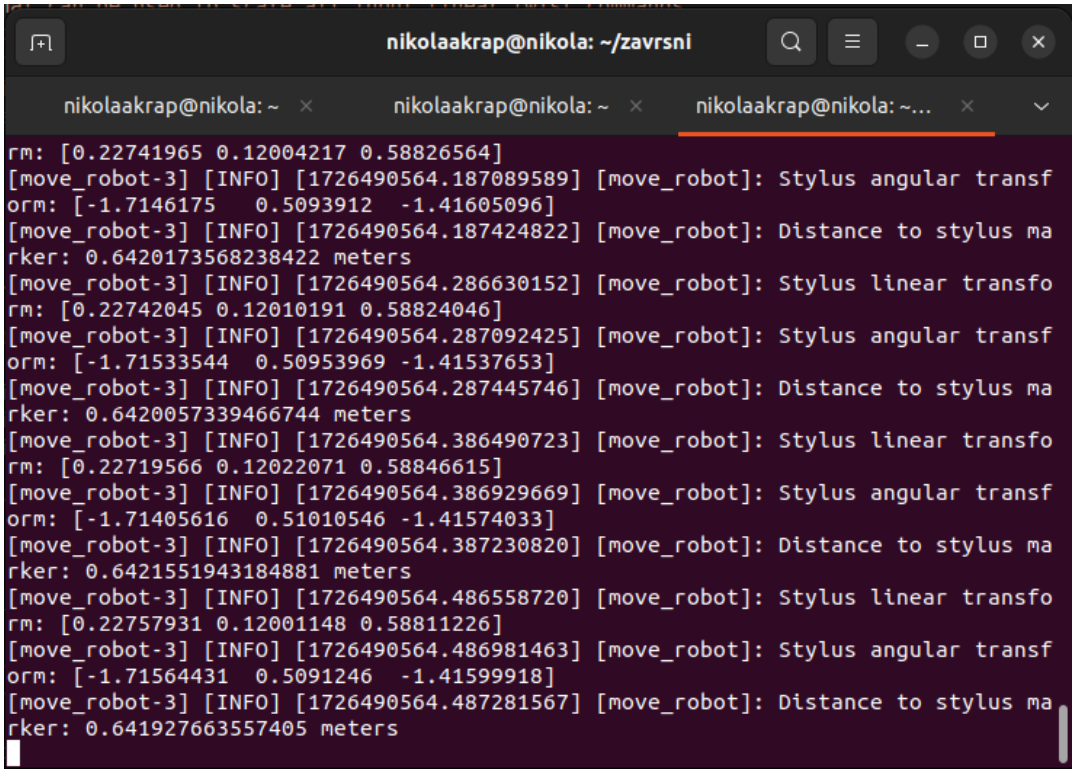
Drugi način rada moguć je samo u slučaju da postoji memorirana posljednja transformacija između *stylus*-a i pomičnog krutog tijela. Za vrijeme njegovog pokretanja konstantno se poziva funkcija *„lookup_transform()“* između pomičnog krutog tijela i prihvatnice robota. Od svake translacije koju funkcija da kao rezultat izvođenja oduzimaju se translacije po svakoj osi od posljednje umemorirane transformacije između *stylus*-a i aktivnog krutog tijela koja se umemorirala pozivanjem prvog servisa. Uz translacije omogućeno je i prilagođavanje rotacije kako bi ukupna poza prihvatnice u odnosu na pomično kruto tijelo u svakom trenutku bila ista. Kada prihvatnica dođe na željenu lokaciju brzine će se i dalje slati pa se moraju odrediti tolerancije unutar kojih pozicija robota može varirati. Te su tolerancije određene proizvoljno i one iznose 5 mm. U stvarnoj primjeni one mogu biti i manje od milimetra no u radu je radi jednostavnosti uzeta veća vrijednost. To ne znači da će robot uvijek biti udaljen 5 mm od željene lokacije nego da toliko smije odstupati. Razlog ovim odstupanjima su stalne promjene matrica transformacija koje kamera daje algoritmu što je dobro radi točnosti pozicioniranja. Također u ovom načinu rada omogućen je i pomak robota unazad ako se kost primakne robotu te on uđe unutar zadane udaljenosti. Svi potrebni čvorovi pokreću se pozivanjem naredbe *„ros2 launch kamera_tf2_py kamera_tf2.launch.py“* u terminalu.

U skripti je omogućena izvedba procesa navođenja robota bez uporabe stylusa. U tom slučaju zanemaruju se promjene koordinatnog sustava *stylus*-a, a robota je moguće rukom dovesti u željeni položaj. U tom slučaju ne sprema se zadnja transformacija između *stylus*-a i prihvatnice već između kosti i prihvatnice.

Izvođenjem ovog algoritma omogućena je implementacija glavnog cilja ovog rada, a to je održavanje relativnog položaja robota i dijela tijela nad kojim se vrši operativni zahvat. Prikaz dijagrama toka algoritma prikazuje Slika 27.

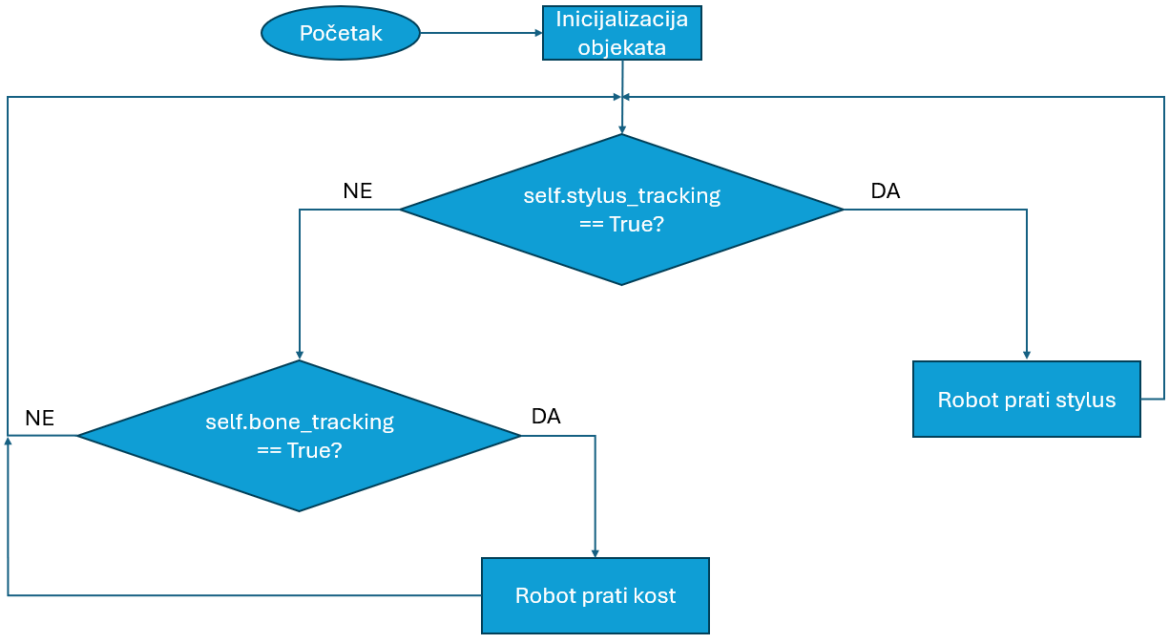


Slika 25. Pokretanje čvorova

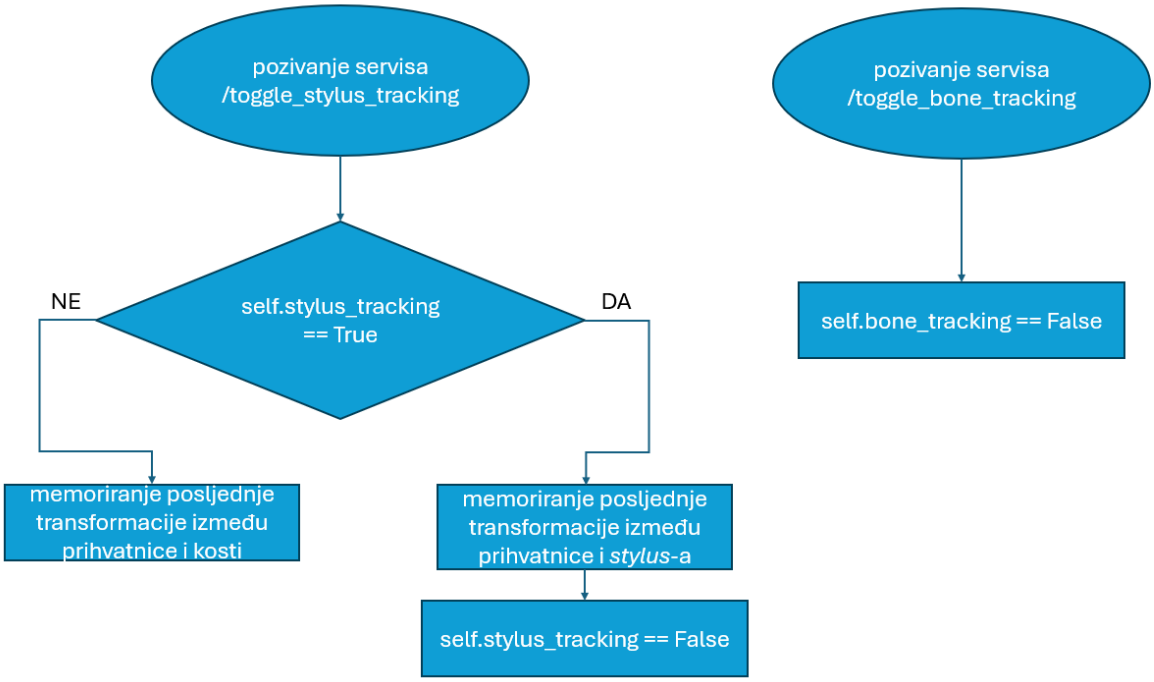


Slika 26. Slanje brzina zglobovima robota

Na slijedećim slikama prikazani su dijagram toka lokalizacije kao i dijagrami toka pozivanja kreiranih ROS2 servisa.



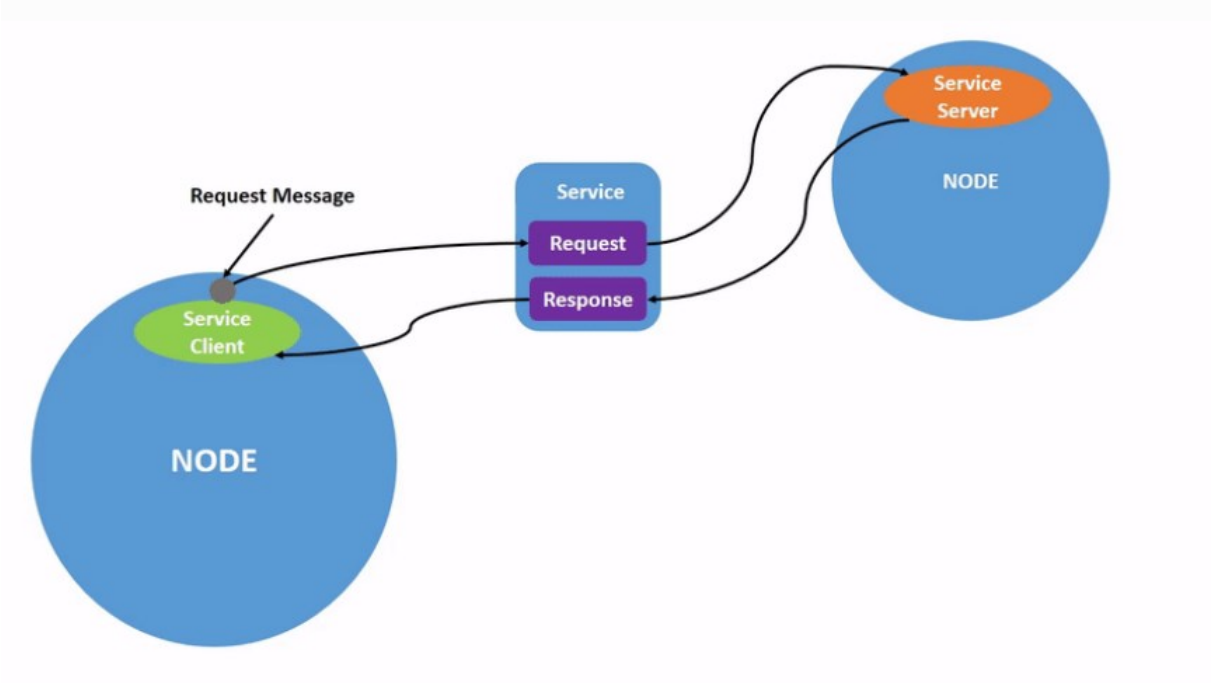
Slika 27. Dijagram toka algoritma lokalizacije



Slika 28. Dijagram toka pozivanja servisa

5.3.3.2. ROS2 Servisi

Servisi su još jedna metoda komunikacije između čvorova u ROS grafu. Servisi se temelje na modelu poziva i odgovora, za razliku od modela objavljivača i pretplatnika kod tema. Dok teme omogućuju čvorovima da se pretplate na tokove podataka i dobivaju kontinuirana ažuriranja, servisi pružaju podatke samo kada ih posebno zatraži klijent. Može postojati mnogo klijenata servisa koji koriste isti servis. Međutim, za jedan servis može postojati samo jedan poslužitelj servisa.

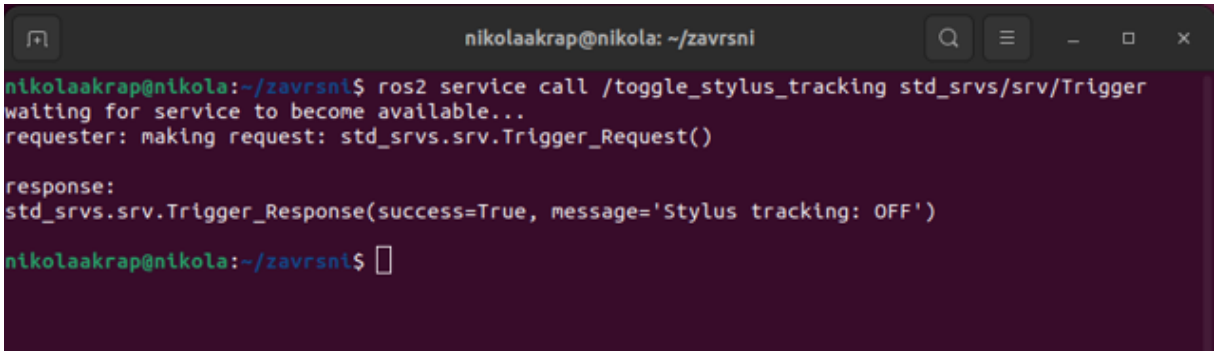


Slika 29. ROS2 – princip komunikacije servisa[14]

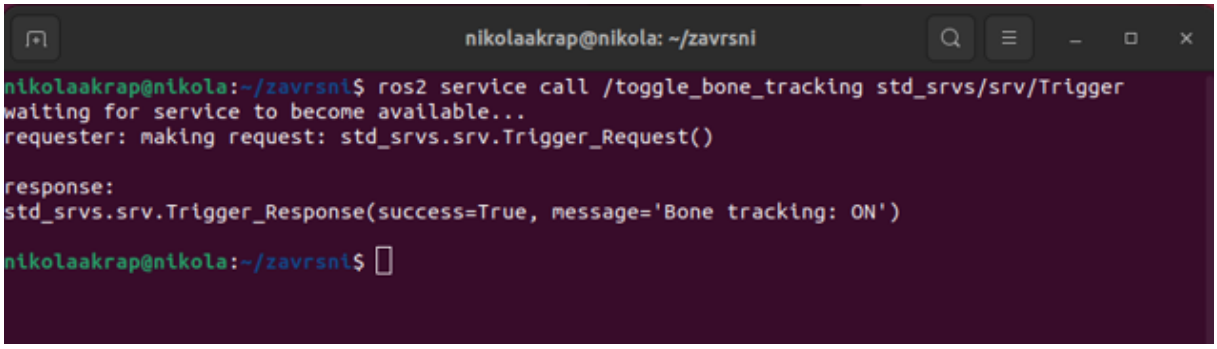
Pokretanjem naredbe *„ros2 service list“* u terminalnu, prikazat će se popis svih servisa koji su trenutno aktivni.

Za potrebe ovog rada izrađena su dva servisa koja pružaju uslugu *Trigger*. Oni ne šalju nikakve kompleksne podatke nego rade na principu „da/ne“, odnosno „*True/False“* signala. Pošto se ne koriste nikakvi kompleksni podaci ovdje je ovakva vrsta servisa dovoljna. Svaki servis poziva svoju funkciju. Prvi poziva funkciju *toggle_stylus_switch*, a drugi funkciju *toggle_bone_switch*. Aktivnosti ovih funkcija opisane su u poglavlju 5.3.3.2.

Servisi se redom mogu pozvati iz terminala unošenjem naredbi *„ros2 service call /toggle_stylus_tracking std_srvs/srv/Trigger“* i *„ros2 service call /toggle_bone_tracking bstd_srvs/srv/Trigger“*. Nakon poziva servisa povratno se dobiva poruka o tome je li zahtjev uspješno izvršen.



Slika 30. Poziv prvog servisa



Slika 31. Poziv drugog servisa

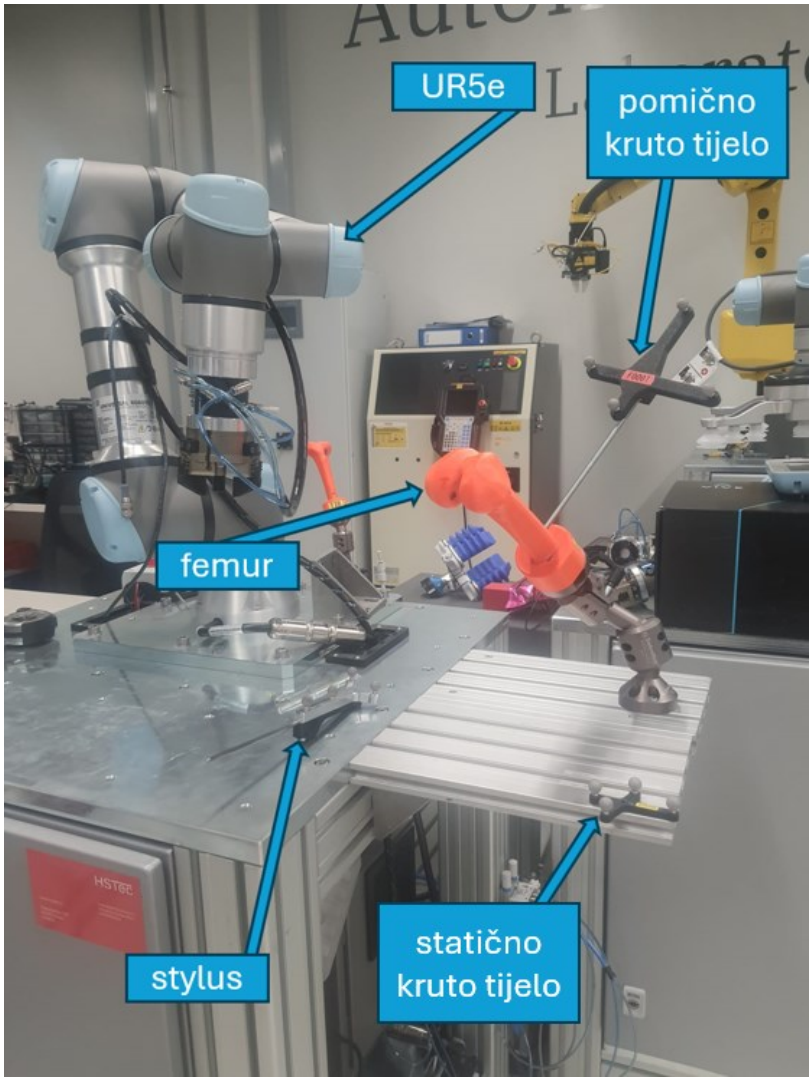
6. ISPITIVANJE I ANALIZA ALGORITMA

Na samom kraju potrebno je provesti testiranje i analizirati dobivene rezultate kako bi se utvrdile točnost i ponovljivost izvođenja procesa. Analiza se provodila na 3D printanom *femur*-u, a sadrži se od tri skupine ispitivanja.

U prvoj skupini ispitivanja, koja se izvode navođenjem uporabom *stylus*-a, promatraju se mali pomaci kosti na razini nekoliko milimetara.

Druga skupina ispitivanja izvodi se ručnim navođenjem robota na željeni položaj, a promatraju se veći pomaci kosti na razini nekoliko centimetara.

Treća skupina ispitivanja obuhvaća rubne slučajeve poput visokih brzina i akceleracija kao i naglih pomaka i promjena smjera gibanja. Ova se ispitivanja također provode ručnim navođenjem robota na željeni položaj.

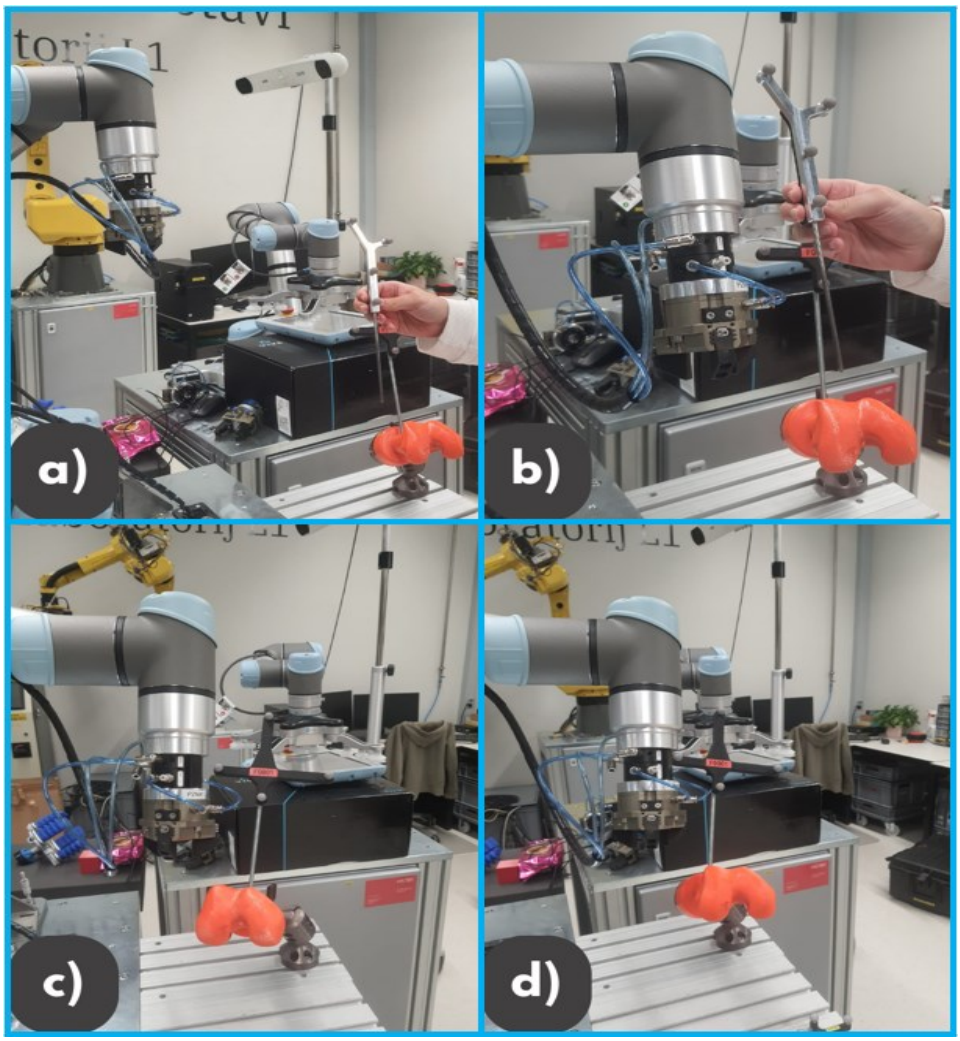


Slika 32. Postav za provedbu testiranja

6.1. Prva skupina ispitivanja

Kao što je ranije navedeno u ovoj skupini provode se ispitivanja milimetarskih pomaka kosti uz navođenje robota *stylus*-om. Pretpostavka je da će robot uspješno kompenzirati pomak kosti s malom greškom jer se radi o malom pomaku i zakretu.

Odabire se proizvoljan željeni položaj postavljanjem *stylus*-a u blizinu kosti. Robot prati *stylus* i zaustavlja se na 10 cm udaljenosti. Pozivanjem prvog servisa pamti se položaj prihvatnice u odnosu na *stylus*, a pozivanjem drugog servisa prihvatnica dolazi u položaj *stylus*-a i počinje održavati relativan položaj u odnosu na kost. Brzina je podešena na način da robot uspori kako se približi željenom položaju. Opadanje brzine je poželjno pogotovo u primjenama gdje se iziskuju visoka preciznost i sigurnost. Za vrijeme pomicanja prihvatnice *servo* upravlja brzinama zglobova na način da robot usporava gibanje s približavanjem konačnom položaju.

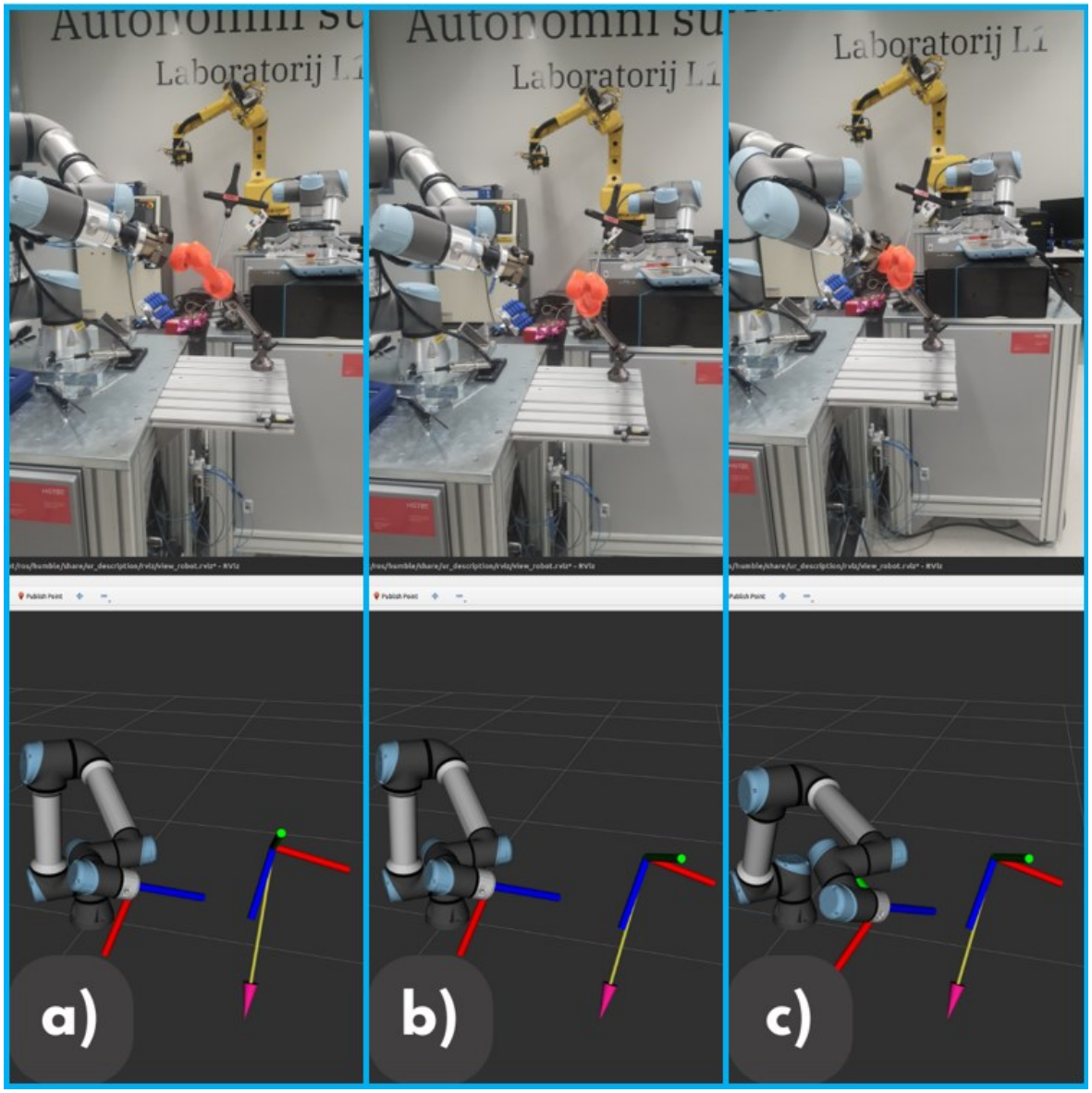


Slika 33. Ispitivanje kompenzacije malih pomaka: prihvatnica u početnom položaju (a), prihvatnica uz *stylus* (b), prihvatnica na položaju *stylus*-a (c), prihvatnica zadržava relativan položaj nakon zakreta kosti (d)

6.2. Druga skupina ispitivanja

U ovoj skupini promatramo pomake veličine nekoliko centimetara. Pretpostavka, isto kao i u prvoj skupini, jest da će robot uspješno kompenzirati svaki pomak uz minimalnu pogrešku.

Robot se navigira rukom pri čemu se u petlji *Python* skripte ne uzima u obzir praćenje *stylus*-a. Prihvatnica se postavlja na proizvoljan položaj uz kost. Zatim se poziva prvi servis koji pamti položaj prihvatnice u odnosu na kost, a nakon toga i drugi koji pokreće kompenzaciju pomaka.

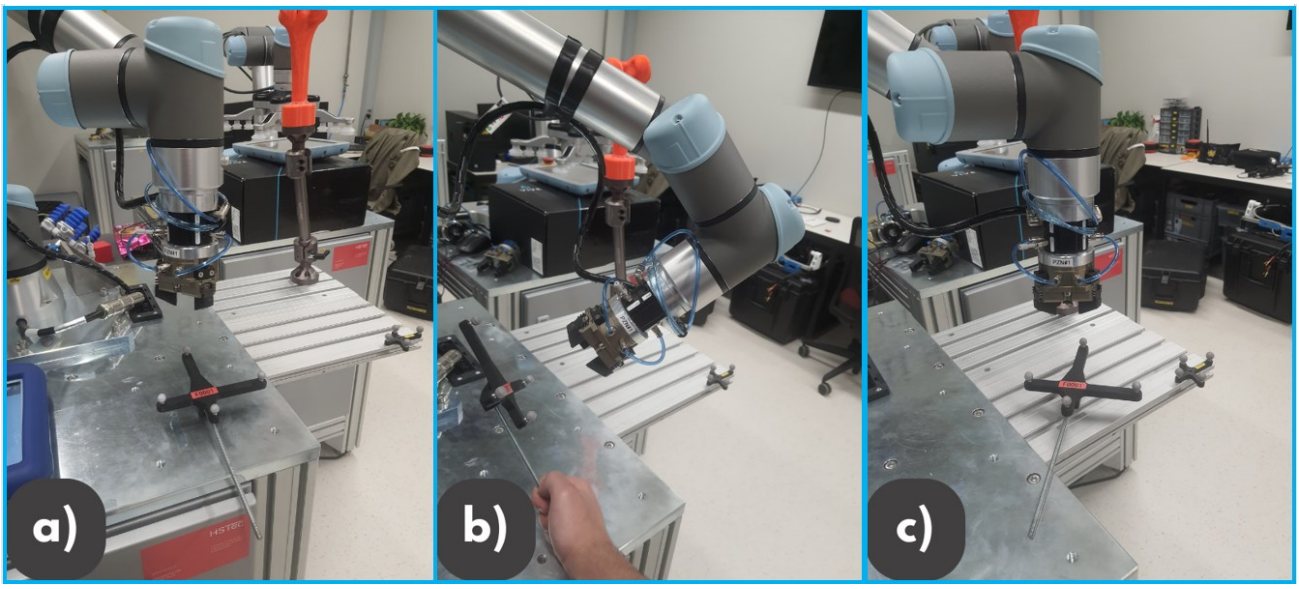


Slika 34. Ispitivanje kompenzacije velikih pomaka: prihvatnica u početnom položaju (a), veliki pomak kosti (b), prihvatnica kompenzira pomak kosti (c)

Slika 34 prikazuje položaj robota u stvarnosti kao i u simuliranom okruženju *RViz*-a.

6.3. Treća skupina ispitivanja

Treća skupina ispitivanja podrazumijeva ispitivanje graničnih slučajeva naglih pomaka. Ovakve se situacije rijetko događaju u medicinskim okruženjima, ali je korisno znati kako će robot reagirati u takvim trenutcima. Pretpostavka je da će se ovdje vidjeti najveća greška, ali i dalje dovoljno mala, na razini jednog milimetra, da ne izaziva probleme za vrijeme izvođenja zahvata.



Slika 35. Ispitivanje graničnih slučajeva: početni položaj (a), nagla rotacija(b), visoka brzina translacije (c)

6.4. Analiza ispitivanja

Kako bi se provela uspješna analiza potrebno je grafički prikazati promjenu translacija po x, y i z-osi, rotacija oko x, y i z-osi te pogreške translacije i rotacije. Također, pratila se srednja vrijednost pogreške koja se može izračunati pomoću izraza za Euklidsku udaljenost (4). Za ove potrebe unutar *Python* skripte *move_robot.py* računaju se navedene promjene položaja nakon čega se mogu grafički prikazati uporabom *Python* knjižnice *matplotlib*.

$$P = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad \textbf{(4)}$$

Kako bi se izračunali $\Delta P, \Delta x, \Delta y, \Delta z, \Delta q_x, \Delta q_y$ i Δq_z računata je aritmetička sredina razlike posljednje i prve izmjerene vrijednosti. Postupak je omogućen uporabom programa *PlotJuggler*. Svaka točka za vrijeme micanja robota sprema se u *.csv* datoteku koja se tada

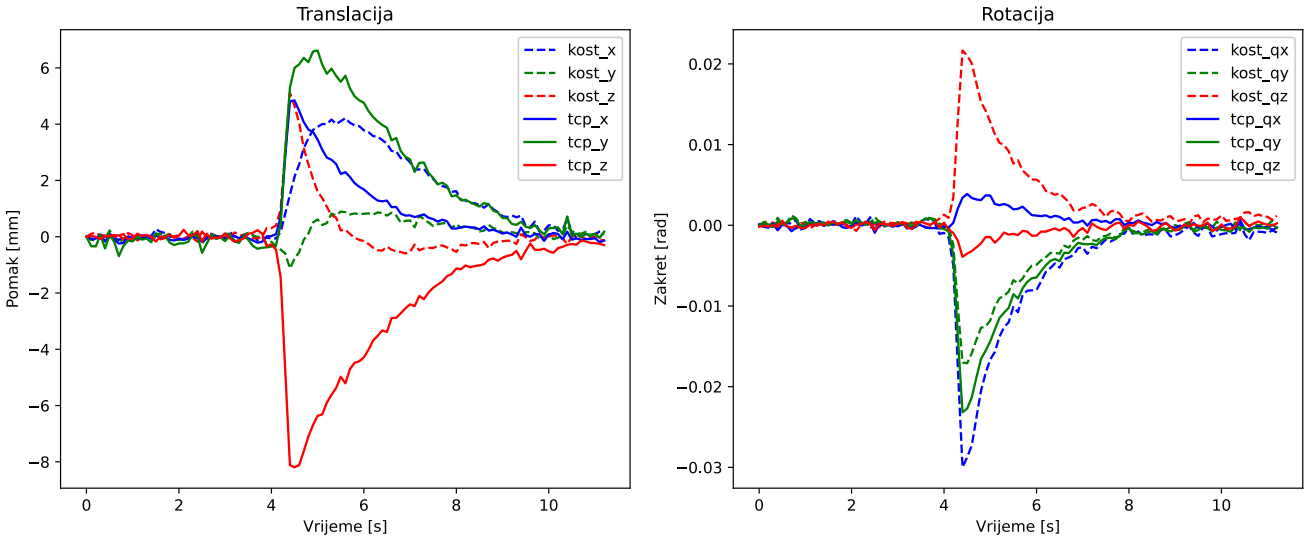
ubacuje u *PlotJuggler*, a na temelju koje se dobivaju dijagrami promjena translacija i rotacija. Iz dijagrama je moguće iščitati tražene vrijednosti u svakom trenutku micanja robota. One najbitnije su uzete na početku i na kraju micanja robota te je izračunata aritmetička sredina njihovih razlika.

6.4.1. Analiza prve skupine ispitivanja

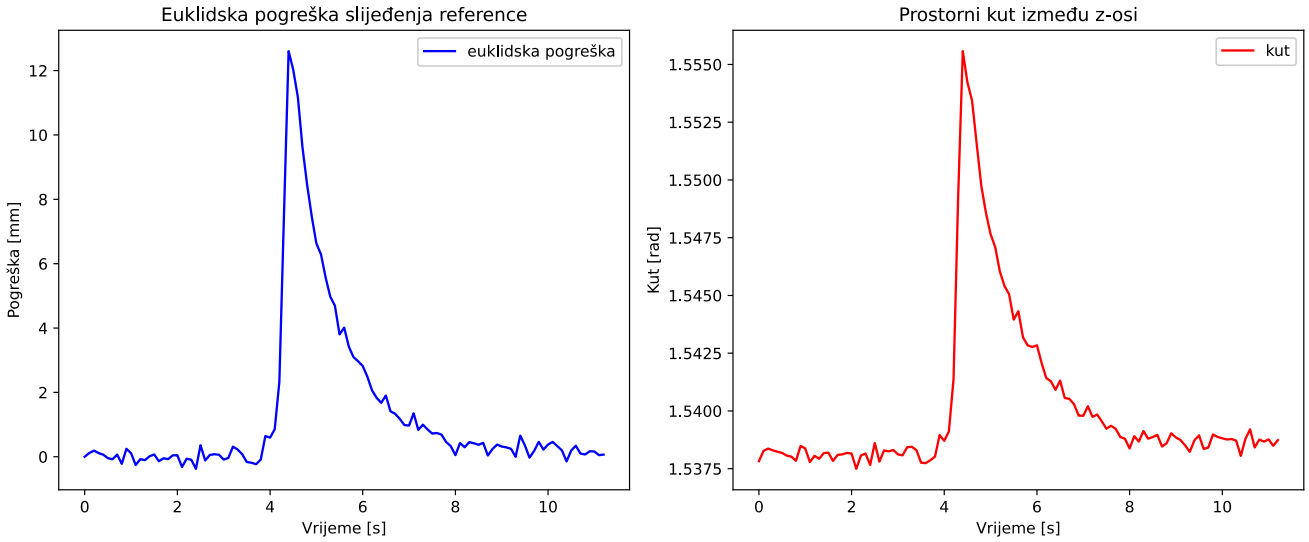
Provedeno je 5 ispitivanja, a pretpostavka o dobroj kompenzaciji promjene položaja je istinita. Srednje vrijednosti grešaka su kako slijedi:

- $\Delta P = 0{,}431901\ mm$
- $\Delta x = 0{,}157\ mm$
- $\Delta y = 0{,}626\ mm$
- $\Delta z = 0{,}124$ mm
- $\Delta Q_x = 0{,}00584\ rad$
- $\Delta Q_y = 0{,}00952\ rad$
- $\Delta Q_z = 0{,}00147\ rad$

Slika 36 i Slika 37 primjer su samo jednog bilježenja promjena u udaljenosti, translaciji i rotaciji prihvatnice robota u odnosu na kost. Promjena položaja događa se u petoj sekundi nakon čega kreće kompenzacija promjene položaja.



Slika 36. Dijagrami promjene translacije (lijevo) i rotacije (desno) prihvatnice i kosti za prvu skupinu ispitivanja



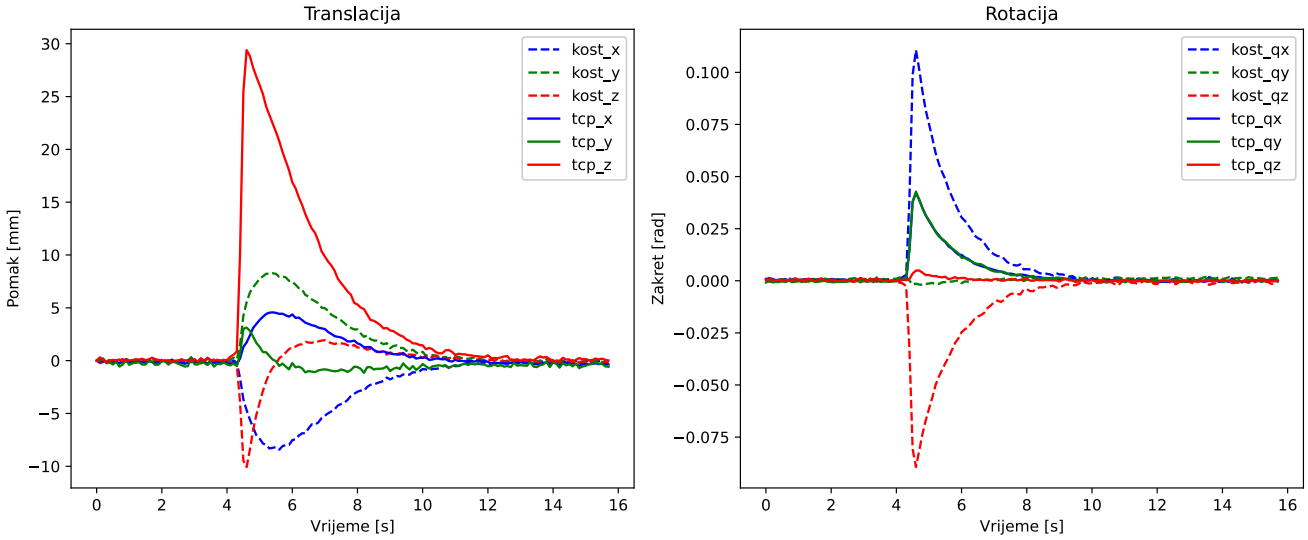
Slika 37. Dijagrami ukupne Euklidske pogreške (lijevo) i promjene kuta između z-osi prihvatnice i kosti (desno) za prvu skupinu ispitivanja

6.4.2. Analiza druge skupine ispitivanja

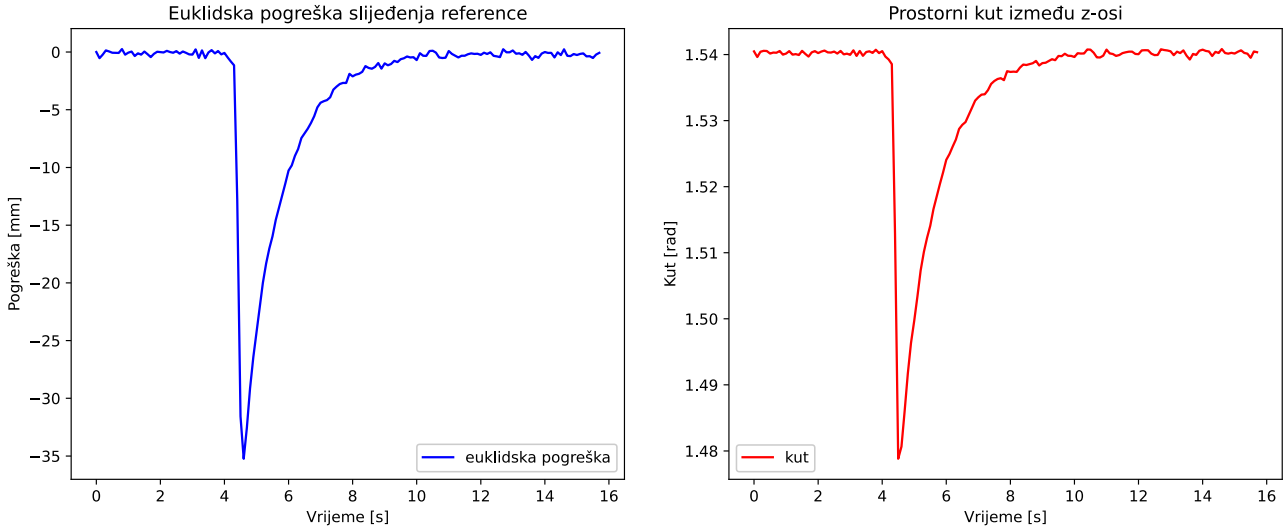
Za drugu skupinu ispitivanja također je provedeno 5 ispitivanja. Pretpostavka slična onoj u prvoj skupini, a to je da će promjena udaljenosti biti ispod jednog milimetra, se ispostavila istinitom. Srednje vrijednosti grešaka su kako slijedi:

- $\Delta P = 0{,}874481\ mm$
- $\Delta x = 0{,}241\ mm$
- $\Delta y = 0{,}892\ mm$
- $\Delta z = 0{,}144$ mm
- $\Delta Q_x = 0{,}00745\ rad$
- $\Delta Q_y = 0{,}001826\ rad$
- $\Delta Q_z = 0{,}00384\ rad$

Slika 38 i Slika 39 prikazuju mjerenja translacije, rotacije i pogreške položaja i kuta prihvatnice robota u odnosu na kost. Promjena položaja događa se u petoj sekundi i vidljiva je brza reakcija robota u pokušaju kompenzacije.



Slika 38. Dijagrami promjene translacije (lijevo) i rotacije (desno) prihvatnice i kosti za drugu skupinu ispitivanja



Slika 39. Dijagrami ukupne Euklidske pogreške (lijevo) i promjene kuta između z-osi prihvatnice i kosti (desno) za drugu skupinu ispitivanja

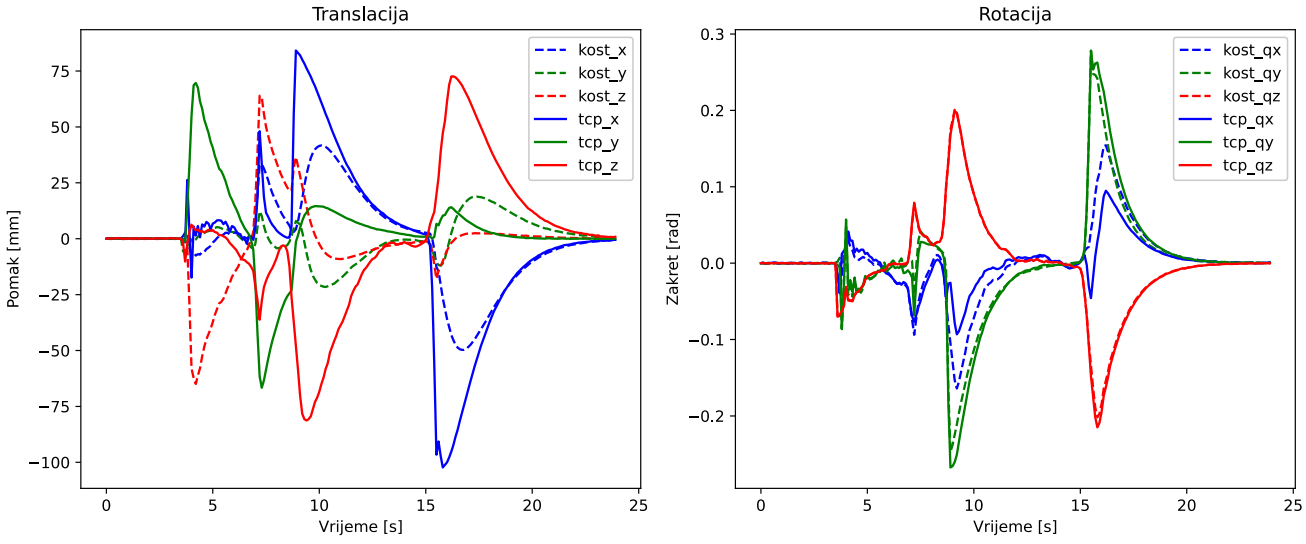
6.4.3. Analiza treće skupine ispitivanja

Treća skupina ispitivanja obuhvaćala je sve nagle promjene položaja. Robot se na sve zahtjeve dobro prilagodio no primjećuju se lošije kompenzacije promjene položaja. Ulogu u lošijim kompenzacijama ima i činjenica da se pomično kruto tijelo u ovim ispitivanjima pomicalo rukom pa niti u jednom trenutku tijelo nije bilo u potpunosti mirno.

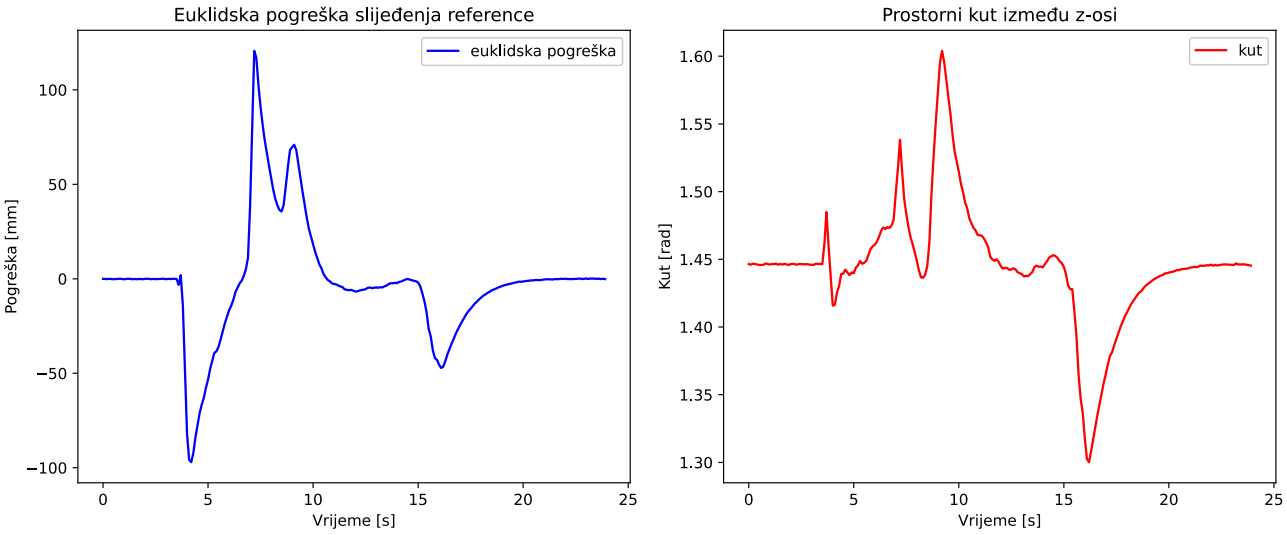
Srednje vrijednosti grešaka su kako slijedi:

- $\Delta P = 0{,}972526\ mm$
- $\Delta x = 0{,}358\ mm$
- $\Delta y = 0{,}861\ mm$
- $\Delta z = 0{,}321$ mm
- $\Delta Q_x = 0{,}001273\ rad$
- $\Delta Q_y = 0{,}002118\ rad$
- $\Delta Q_z = 0{,}001297\ rad$

Slika 40 i Slika 41 prikazuju izraženije promjene položaja kao i veće greške i promjene kuta. Promjene položaja događaju se u više navrata i vidljivo je da ih u svakoj situaciji robot uspješno kompenzira.



Slika 40. Dijagrami promjene translacije (lijevo) i rotacije (desno) prihvatnice i kosti za treću skupinu ispitivanja



Slika 41. Dijagrami ukupne Euklidske pogreške (lijevo) i promjene kuta između z-osi prihvatnice i kosti (desno) za treću skupinu ispitivanja

7. ZAKLJUČAK

U ovom završnom radu uspješno je provedena integracija kolaborativne robotske ruke UR5e u ROS2 operativni sustav. Također, uspješno je obavljeno povezivanje robota, računala i Polaris Vega ST stereovizijskog sustava putem TCP/IP protokola čime se omogućio nesmetan prohod informacija za rad u *real-time* okruženju. U *RoboDK* okolišu uspješno je napravljena simulacija pomicanja robotske ruke čime je omogućen lakši prelazak na rad u ROS2 operativnom sustavu. Svi neophodni paketi su također uspješno integrirani u ROS2 operativni sustav kao i u radno okruženje „zavrsni“. Zahvaljujući velikoj preciznosti Polaris Vega ST optičkog sustava omogućena je visoka ponovljivost dok je greška svedena na minimalnu. Uporabom vlastitih *Python* skripti uspješno je napravljen algoritam za navođenje robota za vrijeme medicinskih zahvata koji omogućava robotu da održava kontinuirano isti položaj u odnosu na dio tijela nad kojim se vrši zahvat. Provedeno je ispitivanje na tri načina koji uključuju male pomake kosti na razini milimetra, veće pomake na razini nekoliko centimetara te rubne slučajeve tijekom kojih dolazi do naglih promjena položaja, brzine, akceleracije i smjera gibanja. Algoritam uspješno prati sve navedene slučajeve s maksimalnim greškama malo iznad jednog milimetra.

Daljnjim istraživanjima može se još više poboljšati točnost, a samim time i osigurati veća sigurnost izvođenja svih medicinskih zahvata i procedura. *Servo* upravljanje gibanjem robota ponekad izaziva trzaje koji nikako nisu dobri u navedenim okruženjima. Također, poželjno bi bilo koristiti višeplanarne alate jer u slučaju uporabe jednoplanarnih, kao što je slučaj u ovom radu, može se desiti da robot svojim prisustvom omete pogled optičkog sustava na markere te dolazi do pogrešaka u gibanju koje mogu dovesti do nezgoda. Uz to, daljnja istraživanja bi mogla pronaći još bolje načine za kvantificiranje pogreške koristeći drukčije metode i pristupe kako bi se konačna pogreška svela na još manju nego što je postignuto u ovom radu.

LITERATURA

[1] Mako Robotic-Arm Assisted Surgery, Orthopaedic associates of the Lakelands, https://lakelandsorthopaedics.com/ortho-surgery/mako.html, posjećeno: 10.09.2024.

[2] StealthStation S8 Surgical Navigation System, Medtronic, https://europe.medtronic.com/xd-en/healthcare-professionals/products/neurological/surgical-navigation-systems/stealthstation.html, posjećeno: 10.09.2024.

[3] Phoenix Children's Hospital to deploy Medtronic neurosurgery robot, The Robot Report, https://www.therobotreport.com/phoenix-childrens-hospital-medtronic-robotic-neurosurgery/, posjećeno: 10.09.2024.

[4] ExcelsiusGPS, Globus Medical. https://www.globusmedical.com/international/solutions/excelsiusgps/, posjećeno: 10.09.2024.

[5] Basic 3D machine vision techniques and principles, Zivid, https://www.zivid.com/3d-vision-technology-principles, posjećeno: 11.09.2024.

[6] A guide to stereovision and 3D imaging, Tech Briefs, https://www.techbriefs.com/component/content/article/14925-a-guide-to-stereovision-and-3d-imaging, posjećeno: 11.09.2024.

[7] Polaris Vega ST, NDI, https://www.ndigital.com/optical-navigation-technology/polaris-vega-st/, posjećeno: 11.09.2024.

[8] Fattori, G, Lomax, AJ, Weber, DC, Safai, S. Technical assessment of the NDI Polaris Vega optical tracking system, BMC, 2021. , posjećeno: 11.09.2024.

[9] UR5e, Universal Robots, https://www.universal-robots.com/products/ur5-robot/, posjećeno: 13.09.2024.

[10] ROS2 Humble, https://docs.ros.org/en/humble/index.html, posjećeno: 14.09.2024.

[11] ROS2 Topics, https://docs.ros.org/en/humble/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Topics/Understanding-ROS2-Topics.html, posjećeno: 14.09.2024.

[12] TF2, https://docs.ros.org/en/foxy/Tutorials/Intermediate/Tf2/Introduction-To-Tf2.html, posjećeno: 14.09.2024.

[13] What is Transmission Control Protocol TCP/IP, https://www.fortinet.com/resources/cyberglossary/tcp-ip, posjećeno: 15.09.2024.

[14] ROS2 Services, https://docs.ros.org/en/foxy/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Services/Understanding-ROS2-Services.html, posjećeno: 15.09.2024.

PRILOZI

I. https://github.com/nikolaakrap/zavrsni_rad

II. https://github.com/SciKit-Surgery/scikit-surgerynditracker

III. https://github.com/UniversalRobots/Universal_Robots_ROS2_Driver/tree/humble

IV. https://github.com/AndrejOrsula/pymoveit2