# Detekcija i praćenje ljudi u svrhu sigurnog kretanja čovjeka u radnom prostoru robota

**Mihelčić, Dario**



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

**Diplomski sveučilišni studij**

# DETEKCIJA I PRAĆENJE LJUDI U SVRHU SIGURNOG KRETANJA ČOVJEKA U RADNOM PROSTORU ROBOTA

**Diplomski rad**

**Dario Mihelčić**

**Osijek, 2021.**

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

**Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit**

**Osijek, 12.07.2021.**

**Odboru za završne i diplomske ispite**

# Imenovanje Povjerenstva za diplomski ispit

| | |
|---|---|
| **Ime i prezime studenta:** | Dario Mihelčić |
| **Studij, smjer:** | Diplomski sveučilišni studij Računarstvo |
| **Mat. br. studenta, godina upisa:** | D-1073R, 06.10.2019. |
| **OIB studenta:** | 89366144523 |
| **Mentor:** | Prof.dr.sc. Robert Cupec |
| **Sumentor:** | Dr. sc. Petra Đurović |
| **Sumentor iz tvrtke:** | |
| **Predsjednik Povjerenstva:** | Izv. prof. dr. sc. . Damir Filko |
| **Član Povjerenstva 1:** | Prof.dr.sc. Robert Cupec |
| **Član Povjerenstva 2:** | Izv. prof. dr. sc. Emmanuel-Karlo Nyarko |
| **Naslov diplomskog rada:** | Detekcija i praćenje ljudi u svrhu sigurnog kretanja čovjeka u radnom prostoru robota |
| **Znanstvena grana rada:** | **Automatizacija i robotika (zn. polje elektrotehnika)** |
| **Zadatak diplomskog rada:** | Zadatak rada je implementacija algoritama za detekciju i praćenje ljudi u sustav koji se sastoji od stvarnog robota i RGB-D kamere integriranih u ROS-u. Potrebno je definirati kretnje robota kako bi se zajamčila sigurnost čovjeka u radnom prostoru robota. Tema rezervirana za: Dario Mihelčić Sumentor s FERIT-a: Petra Đurović |
| **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<br>Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda<br>Jasnoća pismenog izražavanja: 2 bod/boda<br>Razina samostalnosti: 3 razina |
| **Datum prijedloga ocjene mentora:** | 12.07.2021. |

| | |
|---|---|
| Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija: | Potpis: |
| | Datum: |

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

# IZJAVA O ORIGINALNOSTI RADA

**Osijek, 20.07.2021.**

| | |
|---|---|
| **Ime i prezime studenta:** | Dario Mihelčić |
| **Studij**: | Diplomski sveučilišni studij Računarstvo |
| **Mat. br. studenta, godina upisa:** | D-1073R, 06.10.2019. |
| **Turnitin podudaranje [%]:** | 3 |

Ovom izjavom izjavljujem da je rad pod nazivom**: Detekcija i praćenje ljudi u svrhu sigurnog kretanja čovjeka u radnom prostoru robota**

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

i sumentora Dr. sc. Petra Đurović

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

Suradnja ljudi i robota važan je čimbenik budućnosti automatizacije. Robotski manipulatori i pokretni roboti integriraju se u mnoge procese proizvodnje zbog sposobnosti obavljanja zahtjevnih poslova, uz visoku preciznost i rijetke pogreške. Obavljaju poslove poput prenošenja materijala, varenja, bojenja, sastavljanja i drugih. Kako procesi industrijske proizvodnje i dalje zahtijevaju fleksibilan način ljudskog rada, tako se povećava broj slučajeva gdje ljudi i roboti moraju dijeliti zajednički radni prostor. Iako sigurnosne mjere obično uključuju fizičko odvajanje robota pomoću kaveza ili ograda, nije uvijek moguće potpuno izbjeći interakciju robota i ljudi. Dolazi do potrebe razvoja sigurnosnih sustava kojima je omogućena detekcija ljudi u okolini robota u svrhu osiguranja sigurnost ljudi na radu. Ovakvi sustavi moraju ograničiti kretnje robota ovisno o samom stanju radne okoline i ljudima koji se nalaze u neposrednoj blizini. Glavni elementi takvih sustava su postupci prikupljanja i obrade podataka iz okoline robota.

U ovome radu predstavljen je oblik sigurnosnog sustava zasnovan na detekciji i praćenju ljudi na slikama dobivenim RGB-D senzorom, koji se koristi za upravljanje radom robotskog manipulatora kako bi se izbjegla kolizija s čovjekom. Realiziran je sustav koji prikuplja informacije iz okoline robota, reagira na kretnju ljudi u stvarnom vremenu, ispunjava definirane uvjete ograničenja kretnje robota uz minimalni trošak vremena. Sustav je napravljen u sklopu projekta Humans Detected by Robots (HDR) u suradnji s tvrtkom Danieli Systec, a rezultati projekta objavljeni su u [1].

Rad je podijeljen na sedam poglavlja. U poglavlju nakon uvodnog dan je uvid u temu rada te pregled radova u području teme. U trećem poglavlju opisani su algoritmi detekcije i praćenja ljudi, a u četvrtom poglavlju su objašnjeni glavni elementi sustava. Petim poglavljem predstavljena je implementacija opisanog sustava i njegovih komponenti, a u šestom poglavlju opisan je način i rezultati testiranja stvorenog sustava. U posljednjem poglavlju dan je zaključak rada.

## 1.1. Zadatak diplomskog rada

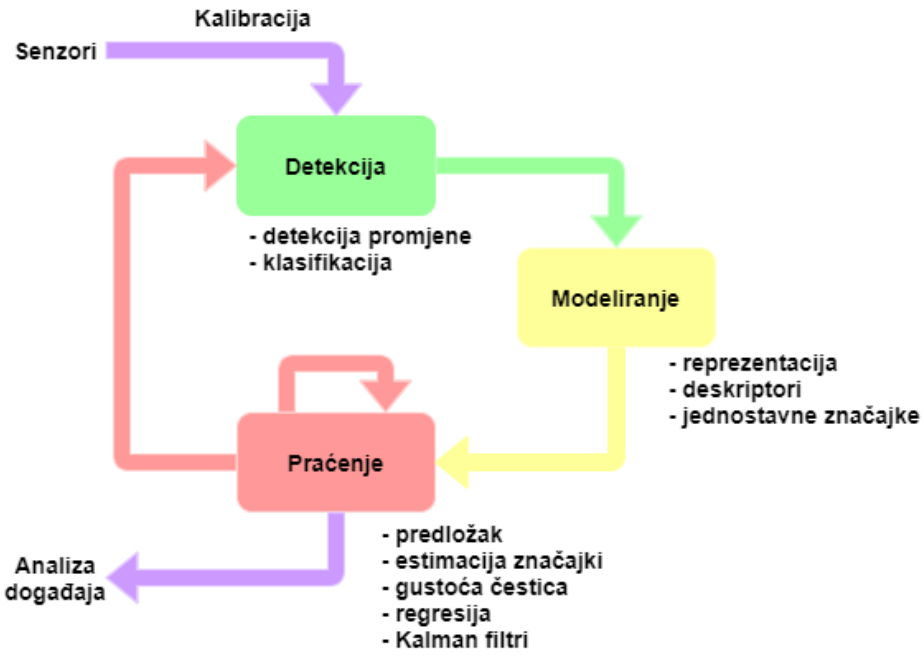
Zadatak rada je implementacija algoritama za detekciju i praćenje ljudi u sustav koji se sastoji od stvarnog robota i RGB-D kamere integriranih u ROS-u. Potrebno je definirati kretnje robota kako bi se zajamčila sigurnost čovjeka u radnom prostoru robota.

# 2. PREGLED PODRUČJA TEME

U ovom poglavlju dan je pregled teme rada. Opisani su postupci koji se koriste, te je dan pregled radova koji se bave sličnim temama.

## 2.1. Detekcija i praćenje

Vizualni sustavi u dinamičnim okolinama pokušavaju detektirati, pratiti i prepoznati značajne objekte ili složenije akcije i ponašanja. Kako je opisano u [2], postoje tri glavna koraka u vizualnoj analitici: detekcija objekata, praćenje objekata i indikatora kroz vrijeme, i evaluacija rezultata praćenja kako bi se opisali događaji i izvele posljedice. Glavni problem ovakvih sustava je varijabilnost ulaznih podataka. Vizualni sustavi detekcije i praćenja moraju imati svojstvo generaliziranja zbog varijacija u izgledu objekata koje su posljedica gledišta, položaja, osvjetljenja, kvalitete slike ili zaklonjenosti objekta. Iako ovakvi sustavi moraju imati jako svojstvo generalizacije, bitno je da ne označavaju sve u vidnom polju kao objekt od interesa. Kako bi se riješio ovaj problem, definiraju se uvjeti detekcije i praćenja. Kako bi se praćenje objekata izvelo na što točniji način, rezultati dobiveni detekcijom mogu se filtrirati i preoblikovati u prigodniji oblik. Dodatno, poželjno je da se ovi zadatci obavljaju u stvarnom vremenu i na standardnom računalu, jer se obično implementiraju u vremenski osjetljive sustave upravljane industrijskim računalima. Osnovni zadaci sustava za detekciju i praćenje prikazani su na slici 2.1.




Sl. 2.1. Osnovni zadatci sustava za detekciju i praćenje [2]

Detekcija objekata je osnova za inicijalizaciju procesa praćenja, koristi se u svakom trenutku rada praćenja. Postoje brojni pristupi detekciji objekata. Jedan pristup detekciji objekata je usporedba trenutne scene s modelom statične pozadine, koji je izrađen obradom niza uzastopnih slika scene. Drugačiji pristup detekcije objekata koristi pomični prozor koji se prevlači preko cijele slike, te se estimira točnost usporedbe dijela slike i sadržaja prozora i klasificira se objekt od interesa. Varijacija pristupa traži točke od interesa na slici te se klasificiraju samo dijelovi slike oko pronađenih točaka. Proces klasifikacije dijelova slike može se obaviti na različite načine, a najpopularnije metode su: usporedba s definiranim predloškom, evaluacija jednostavnih značajki i dijelova objekta ili korištenje neuronskih mreža. Na kraju detekcije se modeliraju podatci u oblik koji na odgovarajući način prikazuje tražene odnose predmeta i događaja, detekcije se opisuju deskriptorima ili jednostavnim značajkama.

Zadatak algoritma praćenja je, na temelju dobivenih podataka detekcije, slijediti objekt po uzastopnim slikama uz računanje vremenski ovisnih karakteristika praćenog objekta. Praćenje objekta se može obavljati uzastopnim korištenjem detekcije ili podešavanjem lokacije i položaja praćenog objekta na temelju predviđanja iz prošlih stanja. Robusno praćenje objekta koji konstantno mijenja poziciju, orijentaciju i brzinu kretanja predstavlja velik izazov. Praćenje se može pojednostaviti postavljanjem uvjeta za kretnju i izgled objekta. Naglo promjenjive i nepredvidljive kretnje mogu se predstaviti jednostavnim modelima poput modela konstantne brzine ili modela konstantne akceleracije. Razvijeno je mnogo algoritma za praćenje objekata kroz slijed uzastopnih scena, od kojih se najčešće koriste: algoritam podudaranja predložaka, estimacija gustoće čestica, regresija, estimacija kretnje, Kalmanovi filtri.

## 2.2. Estimacije 3D položaja čovjeka za podučavanje robota

U radu [3] predstavljen je pristup estimaciji 3D položaja čovjeka u stvarnom svijetu iz samo jedne RGB-D slike. Ovaj pristup koristi robusne detektore ključnih točaka čovjeka za slike u boji, koji se korištenjem podataka o dubini preslikavaju u 3D prostor. Estimacija 3D položaja čovjeka se koristi u kombinaciji sa sustavom za podučavanje robota putem demonstracije kako bi se upravljalo robotom bez potrebe posebnih markera.

Čovjek se snima Kinect RGB-D senzorom, te se u prvom koraku predviđaju ključni dijelovi čovjeka na slici u boji. Koristi se Open Pose biblioteka [4] kako bi se detektirala područja na 2D slici gdje se najvjerojatnije nalaze ključni dijelovi čovjeka. Razvijene su dvije nove mreže za ovaj pristup estimacije položaja čovjeka. Dobivene mape ključnih dijelova se proširuju po z-dimenziji.

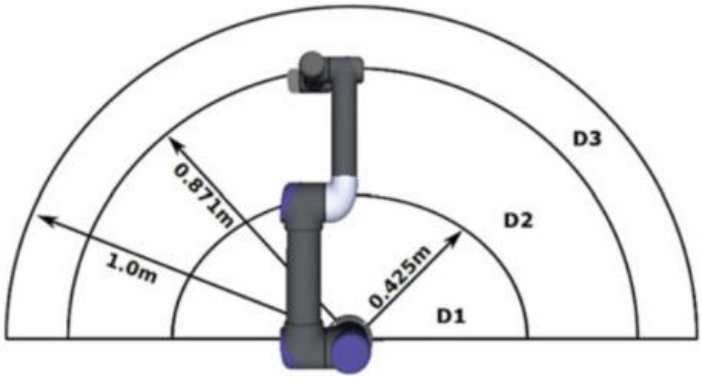
Voksel mreže zauzetosti dobivene iz dubinske slike i mape ključnih dijelova služe kao ulaz za VoxelPoseNet mrežu, koja predviđa ključne točke u 3D prostoru. Isječci slike oko ruku čovjeka predaju se zasebnoj HandNormalNet mreži, koja predviđa vektor normale svake ruke. Robot snima pokrete čovjeka koje mora ponoviti. Robot je dizajniran da slijedi naučene pokrete što točnije, uz očekivana odstupanja uzrokovana geometrijom robota. Rješava se problem optimizacije grafa, kojem ulaz čine trajektorije manipuliranog objekta i čovjekovih ruku i tijela.

Ovaj pristup detekcije i praćenja kretnji čovjeka mogao bi se primijeniti u izradi sigurnosnog sustava za robota, s obzirom da daje potrebne informacije o položaju čovjeka u prostoru. Snaga ovog pristupa je točna estimacija položaja zglobova i ekstremiteta čovjeka, što bi sigurnosnom sustavu dalo mogućnost bolje reakcije na prisutnost ruke čovjeka u radnom području robota. No, u pitanje dolazi brzina odziva sustava, jer pristup nije namijenjen za rad u stvarnom vremenu.

## 2.3. Sigurnosni koncept za suradnju s robotima

Autori [5] predstavili su sigurnosni sustav za suradnju s robotom koji se sastoji od RGB-D kamere za nadzor dijeljenog radnog prostora, niza optičkih senzora daljine i lasera za detekciju radnika koji ulazi u dijeljeni prostor. Programski dio, koji uključuje algoritme detekcije kolizije, planiranja putanje, upravljanja robotom i predviđanja ponašanja radnika u okolini zasnovan je na ROS-u.

Sustav se sastoji od nekoliko interaktivnih modula. Koristi se vanjski program za upravljanje robotom koji generira naredbe za upravljanje položaja zglobova. Koristi se javno dostupna KDL biblioteka za rješavanje inverzne kinematike UR5 industrijskog robota. Podatci prikupljeni senzorima se dijele između modula preko ROS-a ili jednostavnih sučelja. Koristi se javno dostupna FCL biblioteka za detekciju kolizije, koja omogućuje računanje vjerojatnosti kolizije statičnih i dinamičnih objekata u 3D prostoru. Za planiranje putanje robota koristi se OMPL biblioteka, koja koristi lokalni i globalni planer kako bi generirala moguću 3D putanju i provjerila okolinu robota. Planiranje putanje obavljeno je u virtualnom okruženju uz pomoć biblioteke MoveIt [6]. Prostor oko robota podijeljen je na tri polja (D1, D2, D3), kao što je prikazano na slici 2.2.



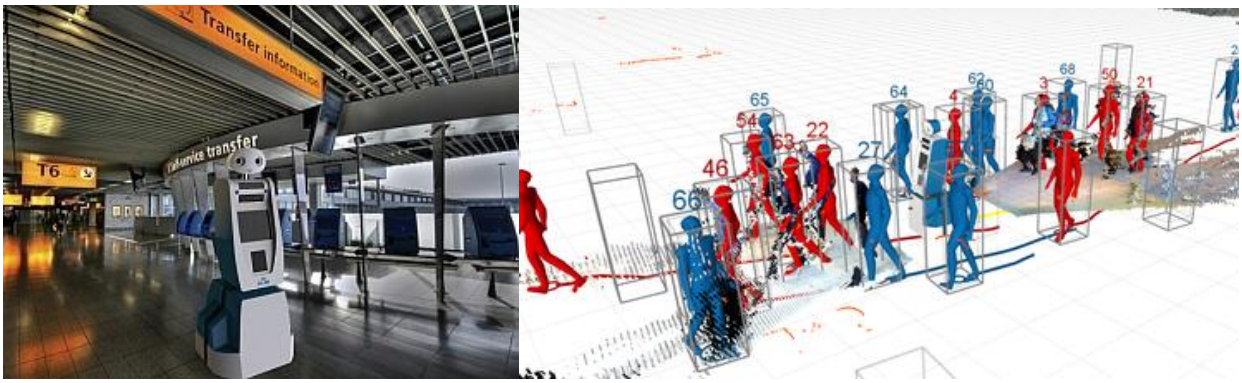

Sl. 2.2. Zone opasnosti oko robotske ruke (Izvor: [5])

Zadatak sigurnosnog kontrolera je prikupiti podatke iz okoline robota te pratiti robota i ljude u dijeljenom prostoru. Sigurnosne funkcije su implementirane u obliku automata konačnih stanja. Kada se detektira čovjek u D3 polju, sustav usporava robota na 250 mm/s. Ako se čovjek nalazi u D2 i udaljenost od robota mu je manja od 200 mm, robot se zaustavlja. Također, robot se zaustavlja ako čovjek ulazi u D1 zonu.

Testirano je vrijeme odaziva robota na zadane uvjete, gdje su autori zaključili da sustav nije dovoljno brz da detektira nagle pokrete ljudi. Ova implementacija sigurnosnog sustava je pogodan samo za lagane industrijske robote kod kojih se može koristiti vanjski planer i kontroler, a nije pogodan za veće industrijske robote koji nemaju mogućnost korištenja vanjskih kontrolera.

## 2.4. SPENCER sustav za detekciju i praćenje ljudi

U radu [7], opisan je sustav za praćenje koji se sastoji od više modula za pokretnog robota opremljenog RGB-D i laser senzorima. Sustav je namijenjen za pokretnog robota SPENCER, koji se mora kretati u prostorima s mnogo ljudi, poput aerodroma. U sustavu su implementirane različite javno dostupne metode praćenja ljudi, koje su testirane u zahtjevnim i dinamičnim scenarijima s puno ljudi. Baza cijelog sustava je ROS, a uključuje module poput: alati za sistematičnu evaluaciju algoritama praćenja ljudi u jednakim uvjetima, tri algoritma detekcije ljudi, četiri algoritma praćenja ljudi, algoritam estimacije socijalnih odnosa između ljudi, alati za obradu podataka i drugo. Na slici 2.3 je prikazan izgled robota te primjer vizualizacije rezultata detekcije i praćenja (preuzeto iz [8]).




Sl. 2.3. Izgled SPENCER mobilnog robota (lijevo), primjer rezultata detekcije i praćenja (desno) [8]

Za detekciju ljudi u 2D laserskim podatcima korištena je metoda za klasifikaciju zasnovana na algoritmu nasumičnih šuma. Sustav omogućava i korištenje metoda klasifikacije Adaboost ili SVM, no algoritam nasumične šume imao je najbolje rezultate na testnom skupu podataka. Kao ulaz u algoritam klasifikacije koriste se geometrijske 2D značajke, koje generira detektor na temelju grupiranih laserskih podataka. Za detekciju ljudi u RGB-D podatcima implementirani su detektor zasnovan na predlošku dubine gornjeg dijela tijela čovjeka opisan u [9], groundHOG detektor iz [10], i modificirani detektor iz [11]. Razvijeni su algoritmi fuzije podataka iz više izvora, gdje svaka detekcija ima oblik *DetectedPerson* ROS poruke. Rezultati detekcije se koriste kao ulazni podatci algoritma za praćenje ljudi. Implementirana su četiri algoritma praćenja: praćenje najbližih susjeda (engl. *nearest-neighbor tracker*), prošireni algoritam praćenja najbližih susjeda (engl. *extended nearest-neighbour tracker*), praćenje s više pretpostavki (engl. *Multi-hypothesis tracker*) i vizualno fokusirano MDL praćenje. U svim navedenim algoritmima praćenja primljene detekcije dolaze iz različitih koordinatnih sustava, specifičnih za pojedini senzor, te se transformiraju u jedinstveni, fiksni koordinatni sustav postavljen u bazi robota.

Testiranjem se pokazalo da kombinacija detektora zajedno s proširenim algoritmom praćenja najbližih susjeda daje najbolje rezultate od implementiranih metoda. Sustav je u većoj mjeri javno dostupan te autori motiviraju druge da implementiraju vlastite metode u sustav. Sustav je odabran kao prikladan za rješavanje postavljenog zadatka stvaranja sigurnosnog sustava za industrijske robote. U nastavku je opisana implementacija i modifikacija opisnog sustava,

# 3. DETEKCIJA I PRAĆENJE LJUDI

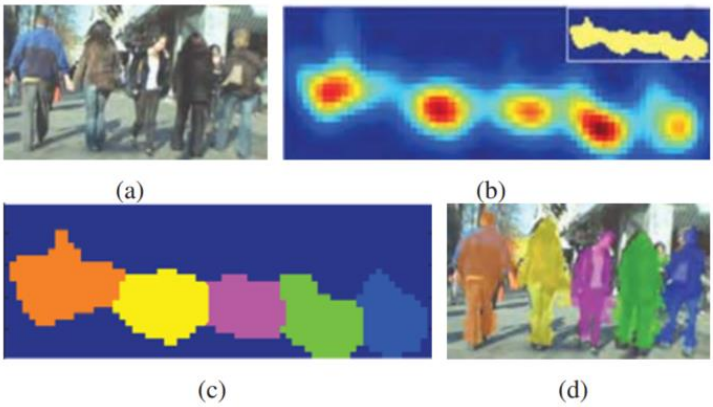
U ovom poglavlju objašnjeni su korišteni algoritmi detekcije i prepoznavanje ljudi. Opisan je način rada glavnih algoritama i dodataka.

## 3.1. Algoritam detekcije

Detektor objekta važan je dio svakog sustava za praćenje. Pristup predložen u [9] koristi detektor na bazi RGB-D podataka za detekciju ljudi u stvarnom vremenu, namijenjen za mobilne robote i kamere nošene na glavi. Cilj rada je u potpunosti iskoristiti informacije o dubini dobivene RGB-D senzorom, kako bi se izradio sustav za detekciju i praćenje koji može raditi u stvarnom vremenu i postavlja što manje zahtjeve za računalo. Bitan dio sustava, koji je opisan u nastavku, je detektor ljudi za Kinect kameru namijenjen za detekciju ljudi na malim udaljenostima u zatvorenim prostorima. Ideja iza detektora je iskoristiti informacije o dubini za pronalaženje područja od interesa za brzu detekciju ljudi s ciljem ograničenja pretrage prostora. Detektor može generirati rezultate frekvencijom većom od 20 fps, pri tome koristeći jednu jezgru CPU-a standardnog osobnog računala. Za razliku od drugih javno dostupnih metoda praćenja, metoda ne koristi GPU i može se izvoditi na slabijim računalima.

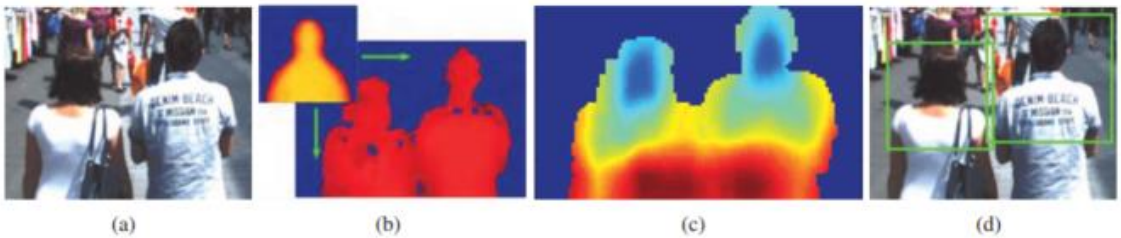
Detektor prima informacije s Kinect RGB-D senzora. U prvom koraku se označavaju primljene 3D točke tako da se dijele na temelju visine u 3 skupine: objekti, ravnina poda, fiksne strukture. Točke koje pripadaju skupini fiksnih struktura nisu važni za daljnji proces detekcije, stoga se odvajaju od ostatka točaka. Estimacija ravnine poda obavlja se na točkama koje pripadaju skupini plohe poda. Traži se ploha koja najbolje pristaje skupini 3D točaka korištenjem RANSAC algoritma. Točke u skupini za objekte se prosljeđuju modulu za pronalaženje područja od interesa (engl. *Region of interest, ROI*). Modul pronalazi ROI-e tako da projicira 3D točke na plohu poda. Projicirane točke se zapisuju u histogram, gdje su točke otežane kvadratom udaljenosti od kamere, kako bi se omogućio nastavak procesiranje objekata koji se nalaze dalje od kamere i sastavljeni su od manje točaka u usporedbi s objektima bližim kameri. Histogrami se obrađuju Gauss filtrom, kako bi rubovi objekata poprimili zaobljeniji oblik, te je postavljen prag vrijednosti pripadnosti točaka da bi se otklonile greške izazvane mjernim šumom. Odvajaju se nakupine točaka u zasebne objekte korištenjem Quick Shift algoritma [12]. U centrima nastalih zasebnih nakupina točaka postavlja se središte baze kvadra koji predstavlja granični okvir ROI-a, čija je visine jednake visini najviše točke u skupini. Za svaki tako pronađeni ROI u oblaku točaka pronalazi se odgovarajući

ROI na 2D slici projiciranjem pravokutnika graničnog okvira okomitog na kameru. Opisani proces prikazan je na slici 3.1. Prva slika je originalna slika koja se želi obraditi, zatim su prikazane točke projicirane na plohu poda, segmentirane nakupine točaka te područja od interesa na originalnoj slici dobivena projekcijom ROI-a.




Sl. 3.1. Vizualizacija pronalaska ROI-a [9]. (a) Izvorna slika (b) Projekcija točaka na plohu poda (c) Odvajanje zasebnih ROI-a (d) ROI na slici nastali projekcijom unazad

Dobivena 2D područja od interesa se prosljeđuju detektoru gornjeg dijela tijela zasnovanom na dubini. Predložak gornjeg dijela tijela dobiven je iz vrijednosti sadržaja anotiranih dubinskih slika ljudi. Detektor uspoređuje sadržaj ROI-a s naučenim predloškom gornjeg dijela tijela čovjeka, s time da je predložak skaliran u ovisnosti o pronađenom lokalnom maksimumu histograma točaka ROI-a, jer visina ljudi na slici ovisi o stvarnoj visini i o udaljenosti čovjeka od kamere. Predložak se uspoređuje sa slikom metodom pomičnog prozora, jer točan položaj čovjeka u ROI-u nije poznat. Detektor računa matricu udaljenosti koja se sastoji od Euklidskih udaljenosti između predloška i svakog uspoređenog segmenta ROI-a dubinske slike. Na slici 3.2. prikazan je opisani proces detekcije ljudi unutar područja od interesa slike. Udaljenost do koje se čovjek može detektirati je ograničena udaljenošću do koje senzor može precizno snimati dubinsku sliku, što je za Kinect kameru oko 5 metara.




Sl. 3.2. Proces detekcije ljudi predloškom gornjeg tijela [9]. (a) Generirani ROI (b) Primjena dubinskog predloška (c) Matrica udaljenosti (d) rezultat detekcije

## 3.2. Algoritam praćenja

Metoda praćenja predstavljena u [13] bazirana je na algoritmu pridruživanja podataka najbližih susjeda (engl. *nearest-neighbor dana association*). Kod praćenja objekata, pridruživanje podataka se odnosi na povezivanje informacija iz novih slika s postojećim objektima praćenja. U radu se želi pokazati da je odabir algoritma pridruživanja podataka manje bitan u usporedbi s dodatcima algoritma kao što su postupci pokretanja i prekidanja praćenja, kompenzacija zaklanjanja i predviđanje kretnje čovjeka. Metoda je izrađena s ciljem postizanja robusne i efikasne metode za praćenje.

Korištenjem relativno jednostavnih dodataka na glavni algoritam praćenja pokušavaju se postići bolji rezultati bez oslanjanja na praćenje s više hipoteza koje je znatno zahtjevnije za izvesti. U ovoj metodi praćenja se sve primljene detekcije ljudi transformiraju u zajednički, fiksni koordinatni sustav. Ovim se stvara zaseban proces za praćenje koji nije ovisan o kretnji robota, ako je metoda implementirana na pokretnog robota. Novi pristigli podatci detekcija se pridružuju postojećim objektima praćenja proširenim algoritmom najbližih susjeda, koji računa Mahalanobisovu udaljenost kao kriterij usporedbe. Kretnja ljudi se predviđa pomoću proširenog Kalman filtra primijenjenog za svakog čovjeka, što se obično može predstaviti modelom gotovo konstantne brzine (engl. *Nearly Constant Velocity, CV*) uz dodavanje određene greške. Uz dodatke za algoritma praćenja, razvijeni su i filtri koji izdvajaju željenu skupinu objekta praćenja. Razvijeni su filtri za odvajanje samo onih objekata praćenja koji su potvrđeni detektorom, odvajanje ljudi koji se ne gibaju i odvajanje željenog broja ljudi koji su najbliži robotu.

Autori posebno ističu važnost razvijenih dodatka osnovnom algoritmu praćenja. Algoritma za predviđanje brzine čovjeka poboljšan je dodavanjem modela brzine gibanja čovjeka. Osim osnovnog modela gotovo konstante brzine, dodani su modeli Brownovog gibanja, gotovo konstantne akceleracije (engl. *Nearly Constant Acceleration, CA*) i gotovo konstantnog koordiniranog zakretanja (engl. *Nearly Coordinated Turn, CT*). Varijacije u konstantnim promjenama se kompenziraju dodavanjem bijelog Gaussovog šuma. Kako bi se uspješno predvidjele varijacije ljudske kretnje, modeli se mogu kombinirati filtrom višestrukih interaktivnih modela (engl. *Interacting Multiple Models filter, IMM*). Drugi dodatak algoritmu praćenja je logika pokretanja praćenja. Stvaranje objekta praćenja se filtrira postavljanjem praga vrijednosti za Euklidsku udaljenost detekcije i predviđenih vrijednosti praćenja za određeni broj isječaka. Dodan je i filter koji potvrđuje samo one objekte praćenja čija je brzina unutar definiranog intervala, kako bi se izbjeglo praćenje mirnih predmeta oblika sličnih čovjeku.

# 4. ELEMENTI SUSTAVA

U ovom poglavlju opisane su metode i alati koji su korišteni u procesu stvaranja sigurnosnog sustava. Opisani su glavni radni okviri, programski paketi, korišteni senzori i opreme.

## 4.1. ROS

Robotski operacijski sustav (engl *Robot Operation System*, ROS [14]) je fleksibilni radni okvir za izradu programa za robote. ROS je kolekcija alata, biblioteka, i konvencija čiji je cilj olakšati proces izrade složenih i robusnih ponašanja robota. Nudi razne usluge značajne za operacijske sustave poput apstrakcije sklopovlja, upravljanja uređajima na niskoj razini, implementacije često korištenih funkcija, razmjene poruka između procesa, i upravljanja paketima. Nudi alate i programske pakete za preuzimanje, građenje, pisanje, i pokretanja programskih kodova na više računala. Glavni cilj ROS-a je stvoriti zajednicu koja razvija i razmjenjuje programe u području robotike.

Komunikacija ovog sustava se zasniva na mreži procesa koji su međusobno povezani ROS komunikacijskom infrastrukturom. ROS sadrži implementaciju raznih vrsta komunikacije, uključujući komunikaciju u obliku zahtjeva i odgovora putem servisa (engl. *services*), asinkronu razmjenu podataka putem tema (engl. *topic*), i pohranu podataka na parametarski server (engl. *Paremeter Server*). Upravo ovi oblici komunikacije omogućuju stvaranje sustava čiji su procesi gotovo u potpunosti odvojeni, koji rade neovisno o drugim procesima, što ROS korisnicima daje mogućnost stvaranja modularnih programa, te preuzimanje i modificiranje raznih dijelova programa otvorenog koda i implementiranje dijelova u vlastiti sustav.

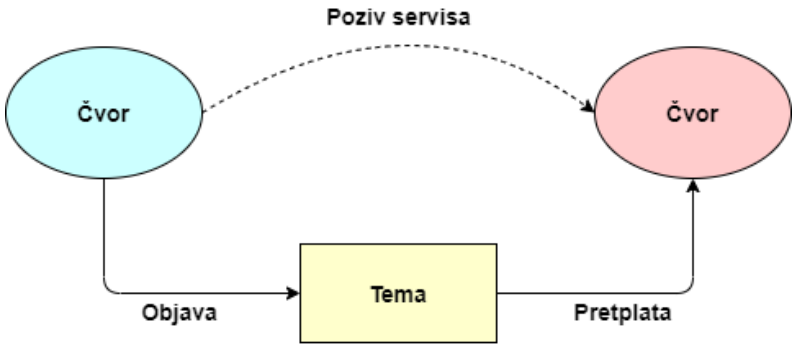
ROS je distribuirani radni okvir procesa koji omogućuje raspodjelu izvršnih programa koji mogu biti zasebno dizajnirani i slabo povezani tijekom rada sustava. Sustav je namijenjen za korištenje na platformama s Unix bazom. Programi za ROS se najviše testiraju na Ubuntu i Mac OS X sustavima, iako je razvijena potpora i za druge Linux platforme. U nastavku su objašnjeni glavni koncepti koji se koriste u ROS-u i važniji programski paketi.

### 4.1.1. Osnovni koncepti

ROS programi se sastoje od međusobno povezanih čvorava (engl. *nodes*). Čvorovi su procesi koji obavljaju određene zadatke, čine osnovni element kojim se grade ROS programi. Čvorovi međusobno komuniciraju putem poruka, složenih struktura podataka posebno definiranih za različite tipove komunikacija. Poruke mogu sadržati standardne tipove podataka, kao što su cjelobrojni, decimalni brojevi, polja brojeva, i složene strukture podataka. Razmjena poruka između čvorova odvija se preko glavnog procesa zvanog *Master*. Ovaj proces je zaslužan za registraciju imena komponenti, povezivanje čvorova i prenošenje poruka.

Najčešći način komunikacije u ROS-u je preko tema. Teme predstavljaju tok podataka kojim se prenose poruke unutar ROS transportnog sustava, odnosno tema je ime kojim je obilježen sadržaj poruke koja se prenosi. Ovaj sistem radi na temelju izdavača (engl. *publisher*) i pretplatnika (engl. *subscriber*). Izdavač objavljuje poruke na temu, a pretplatnik prikuplja podatke s teme, pri tome čvorovi rade neovisno jedan o drugom. Čvor koji želi primiti poruku određenog tipa mora se pretplatiti na odgovarajuću temu, no taj izdavač nema direktnu komunikaciju s čvorom koji objavljuje na temu. Ideja iza ovog oblika komunikacije je razdvojiti procese stvaranja podataka od procesa prikupljanja i obrade podataka.

Drugi često korišteni oblik komunikacije, važan kod stvaranja distribuiranih sustava, su servisi (engl. *services*). Servisi se zasnivaju na direktnom obliku komunikacije zahtjeva i odgovora. Servis definiraju dva tipa poruka, jedan za zahtjev i jedan za odgovor, kojima se obavlja željena radnja. Čvor koji obavlja željenu radnju nudi servis pod određenim imenom, a klijentski čvor koristi servis slanjem poruke zahtjeva i čeka odgovor. Ovim oblikom komunikacije stvara se ovisnost između čvorova, jer čvor koji zahtjeva obavljanje radnje preko servisa ovisi o čvoru koja nudi odgovarajući servis. Na slici 4.1. prikazan je jednostavni grafički prikaz ROS komunikacijskih oblika.




Sl. 4.1. Oblici komunikacije između ROS čvorova [14]

### 4.1.2. ROS alati

Velika prednost ROS-a je u moćnim alatima za prikupljanje, obradu, simulaciju i vizualizaciju podataka. Velik broj alata olakšava stvaranje složenih programa, korisnicima daje jednostavan i brz način rješavanja učestalih problema, pogodnih za pronalaženje grešaka u kodu. ROS sadrži brojne alate, od službeno razvijenih za ROS do korisnički razvijenih pomoćnih alata, a najvažniji alati su objašnjeni u nastavku.

Za razvoj robota često je važno zapisati informacije o simuliranju i testiranja koje se kasnije mogu reproducirati u svrhu usporedbe i analize. ROS nudi efektivno rješenje ovog problema u obliku *.bag* datoteka korištenjem *rosbag* naredbe. Ovaj format datoteke jednostavni je način spremanja ROS poruka, omogućuje i spremanje vremenski ovisnih podataka koji se mogu koristiti za reprodukciju događaja. Alat *rosbag* sadrži skup naredbi za snimanje, reprodukciju, istraživanje, modificiranje ovog formata datoteka, obično korištenjem unutar terminala, ali razvijen je i API (engl. *Application Programming Interface*) putem kojeg se naredbe mogu koristiti unutar programa.

Moćan alat za vizualizaciju i simulaciju podataka u ROS razvojnom okviru je RViz (engl. *ROS Visualization* [15]). RViz daje mogućnost simulacije robotskih modela, prikupljanje informacija sa senzora robota, reproduciranje prikupljenih informacija. Ovo je složen alat kojim se mogu vizualizirati brojni tipovi podataka, što korisnicima daje olakšan način otklanjanja neželjenih ponašanja i grešaka. Korištenjem RViz alata može se prikazati 3D podatke senzora kao što su stereo kamere, laseri, Kinect kamere i drugi uređaji u obliku oblaka točaka ili dubinskih slika. Podatci prikupljeni 2D senzorima poput web kamera, RGB kamera i 2D lasera mogu se prikazati u obliku slika. Ovim alatom se mogu vizualizirati lokalne i globalne putanje, koordinatne osi, karte prostora, granični okviri, koji su značajni korisniku u procesu otklanjanja grešaka.

Pored RViza, koristan razvojni okvir za razvoj grafičkih alata za vizualizaciju i upravljanje podataka u ROS-u je *rqt* [16]. Ova skupina alata grupirana je u jednu, lako modularnu aplikaciju koja se pokreće naredbom *rqt*, a željeni alati se koriste kao dodaci. Korisna je za prikaz informacija o ROS programu kao što su: imena aktivnih tema, oblici poruka koje se razmjenjuju, aktivni čvorovi, obavijesti o greškama i upozorenja, parametri programa, grafovi povezanosti, sadržaj *.bag* datoteka i drugo. Kao i RViz, znatno olakšava praćenje rada programa i otklanjanja neželjenih ponašanja i grešaka.

Modularni pristup izgradnje programa omogućuje zaseban razvoj dijelova programa, no nastaje problem organizacije programa. ROS nudi posebne datoteke za pokretanje programa, *launch*

datoteke. Ove datoteke imaju ulogu organiziranja stvorenog programa tako da su u njima definirani parametri programa te pozivi čvorova. Izvođenje ovih datoteka obavlja se naredbom *roslaunch*. Ovaj alata je zaslužan i za pokretanje glavnih programa i čvorova koji moraju biti pokrenuti za svaki sustav baziran na ROS-u. Ovi programi i čvorovi se kolektivno zovu *roscore*, a sastoje se od ROS *Mastera*, ROS parametarskog servera i čvora za zapis podataka aktivnih procesa.

### 4.1.3. ROS-Industrial

ROS-Industrial [17] je projekt otvorenog koda koji povezuje napredne sposobnosti ROS programa s industrijskim sklopovljem i aplikacijama. ROS-Industrial repozitorij sadrži sučelja za industrijske manipulatore, alate, senzore i mreže uređaja. Dostupne su i biblioteke za automatsku kalibraciju 2D/3D kalibraciju, obradu planiranja putanji ili pokreta, aplikacije poput Scan-N-Plan, koja omogućuje planiranje putanji u stvarnom vremenu ovisno o skeniranim 3D podatcima, alate za razvoj programa kao što je Qt Creator ROS Plugin, program koji pojednostavljuje zadatke i objedinjuje razne ROS alate, i programe za vježbu koji su posebno izrađeni za pojedine proizvođače.

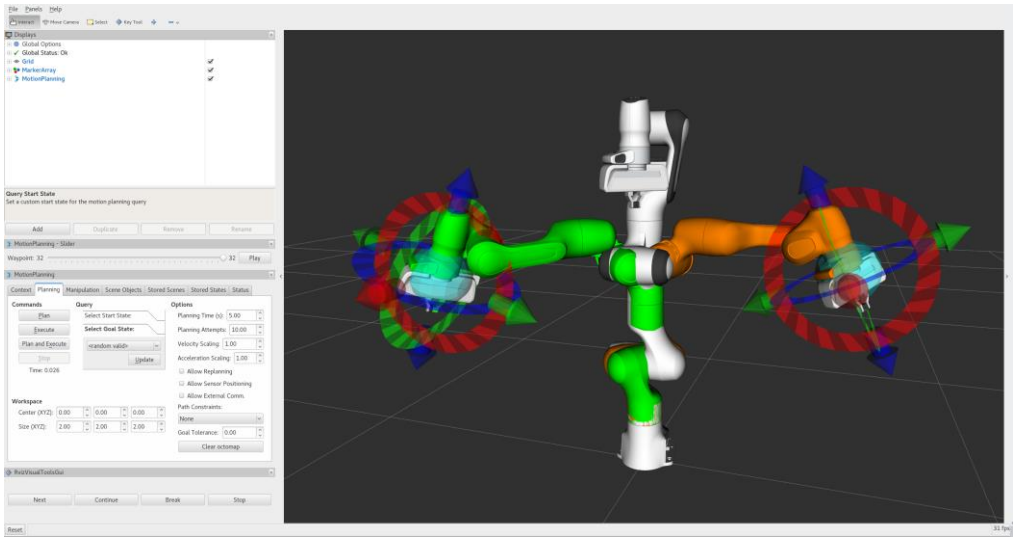
Ovaj projekt se razlikuje od samog ROS-a po tome što dodaje URDF (engl. *Unified Robot Description Formats*) datoteke, koje se koriste za predstavljanje robotskih modela, biblioteke sučelja za razne industrijske aplikacije i opremu, te druge mogućnosti korisne za automatizaciju u proizvodnji, uz naglasak na pouzdanost i sigurnost. ROS-Industrial proširuje mogućnosti ROS-a na automatizaciju proizvodnje, objedinjuje različite industrijske uređaje i programe u jedinstven komunikacijski sustav. Nudi biblioteke koje daju potporu za korištenje ABB manipulatora, dostupni su čvorovi za komunikaciju s ABB kontrolerima industrijskih robota, URDF modeli za određene modele, uz odgovarajuće MoveIt pakete.

Neke od mogućnosti ROS-Industriala su: daje objedinjeni okvir za razvoj ROS programa za potrebe proizvodnje, usmjerenost prema robusnim i pouzdanim aplikacijama za potrebe industrije, daje kombinaciju snage ROS-a i postojećih tehnologija, daje vezu između ROS funkcionalnosti visoke razine i pouzdanih i sigurnih kontrolera industrijskih robota niske razine, potiče razvoj programa koji nisu ovisni o sklopovlju standardizacijom sučelja, daje jednostavan način za primjenu suvremenih istraživanja u industrijskim aplikacijama.

### 4.1.4. MoveIt

MoveIt [6] je javno dostupna platforma za razvoj programa za robotske manipulatore i mobilne robote. MoveIt nudi alate potrebne za rješavanje planiranja trajektorije, računanje inverzne kinematike, prenošenje složenih kretnji na kontrolere sklopovlja, prikupljanje i vizualizaciju podataka 3D senzora, izbjegavanje sudara i drugo.

Za korištenje MoveIta s robotskim manipulatorom potrebno je definirati model robota u obliku URDF datoteke. Ove datoteke su izrađene i javno dostupne za velik broj robota zahvaljujući velikoj zajednici korisnika koji su MoveIt koristili s više od 126 robota. Proces postavljanja ostalih parametra potrebnih za rad s robotom može se obaviti korištenjem MoveIt assistant aplikacije. Simulacija i upravljanje robota može se obaviti korištenjem RViz alata tako da se uključi MoveIt dodatak. Na slici 4.2. prikazana je vizualizacija robotskog manipulatora u RVizu s MoveIt dodatkom.



Sl. 4.2. Vizualizacija robotskog manipulatora u RVizu (Izvor: [6])

MoveIt omogućuje korištenje različitih algoritma planiranja trajektorije, koji se mogu mijenjati i pokretati korištenjem RViz dodatka ili u programima putem C++ ili Python API-a. Središnji čvor move_group objedinjuje sve funkcionalnosti potrebne za upravljanje robotom, te nudi pojedine funkcije preko ROS-a u obliku akcija ili servisa. Ovaj čvor zaslužan je za prikupljanje informacija o trenutnom stanju robota, prikupljanje podataka senzora robota i za komunikaciju s kontrolerom robota.

## 4.2. Kinect

Kinect [18] je naziv za seriju Microsoftovih senzora namijenjenih za prikupljanje informacija o kretnji iz okoline. Iako su uređaji namijenjeni za korištenje kao oprema za video igre, često se koristi i u raznim istraživačkim radovima, uglavnom zbog povoljne cijene senzora. Danas se senzor koristi u razvoju robota, medicine, brige o zdravlju i slično.

Kako se Kinect kamera sve više koristila za razvoj programa robotskog vida, tako su proizvođači odlučili razviti senzor koji se može koristiti i na Windows operacijskom sustavu, a kasnije su razvijeni i potrebni programi za pokretanje kamere na Linux operacijskim sustavima. Danas se Kinect namijenjen za konzolu Xbox 360 može koristiti na većini operacijskih sustava. Na slici 4.3. prikazan je Kinect senzor za Xbox 360.

Kinect senzor sadrži RGB kameru, dubinski senzor i mikrofon. Dubinski senzor se sastoji od projektora infracrvenih lasera, koji zajedno s jednobojnim CMOS senzorom prikupljaju podatke o 3D prostoru. Frekvencija prikupljenih podataka može biti između 9 i 30 Hz. RGB i dubinska slika su rezolucije 640 x 480, iako sklopovlje kamere ima mogućnost snimanja RGB slike rezolucije 1280 x 1024. Dubina se može snimati na udaljenostima od 1.2 do 5 metara.


Sl. 4.3. Kinect senzor [18]

## 4.3. ABB IRB 2400L

IRB 2400 označava skupinu robota tvrtke ABB optimiranih za zavarivanje, primjenu u procesu proizvodnje ili utovaru i istovaru robe. Ova skupina robota pokazuje velike sposobnosti u procesnim zadatcima gdje je preciznost vrlo bitna. Naglasak je stavljen na jednostavnu instalaciju robota, visoku pouzdanosti i laku održivosti.

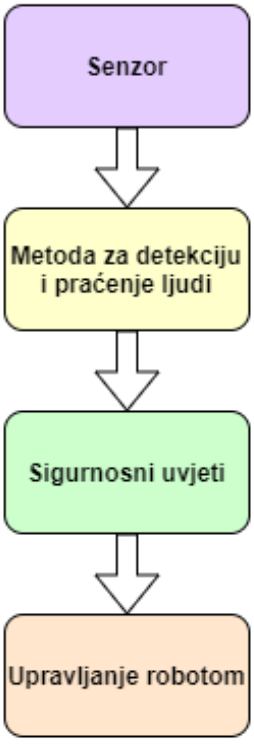
IRB 2400L [19] robot je sa šest stupnjeva slobode, poseban po širokom dohvatu od 1.8 metara, može prenositi predmete težine do 7 kilograma. Robot se može upravljati S4C, S4C+ ili IRC5 kontrolerom. Na slici 4.4. prikazan je izgled ABB IRB 2400L robota.

Kontroler robota IRC5 omogućuje brzo i efikasno upravljanje robotom. Kontroler korisnicima pruža fleksibilno, sigurno, modularno, prilagodljivo upravljanje i potporu za PC alate. Dodatak na kontroler je zaslon na dodir i 3D upravljač, *FlexPendant*, koji nudi intuitivno upravljanje. Kontroler koristi RAPID programski jezik, jednostavan i fleksibilan način za implementaciju programa i procesnih aplikacija. IRC5 nudi mogućnost spajanja preko GSM ili Ethernet veze u svrhu nadziranja rada robota.

Sl. 4.4.. Robot ABB IRB 2400L [19]

# 5. IMPLEMENTACIJA

Sustav za sigurno kretanje ljudi u radnom prostoru robota baziran je na ROS-u (ROS Kinetic Kame, Ubuntu 16.04). Programski jezici koji su korišteni za realizaciju sustava su C++ i Python. Sustav se sastoji od 4 cjeline koje uzastopno prosljeđuju informacije: senzor, program za detekciju i praćenje ljudi, implementirana sigurnosna ograničenja i robotski manipulator, kako je prikazano na slici 5.1. Korišteni senzor je opisana RGB-D Kinect kamera, koja prikuplja vizualne informacije o radnom prostoru robota i ljudi koji se mogu nalaziti u njemu. Metoda za detekciju i praćenje koja se koristi u ovom sustavu je modificirana SPENCER metoda. Metoda daje precizne informacije o poziciji, orijentaciji i brzini praćenih ljudi u radnoj okolini robota, koje se koriste u osiguravanju sigurnosnih uvjeta. Sigurnosni uvjeti su skup pravila kojima se ograničava kretnje robota ovisno o prisutnosti ljudi u radnom prostoru robota, a implementirani su u obliku ROS čvora. Čvor prima informacije od metode za detekciju i praćenje ljudi, provjerava definirane uvjete, te šalje odgovarajući upravljački signal robotu. Na kraju sustava se upravlja IRB 2400L robotom korištenjem MoveIt biblioteke i *Ethernet* veze s kontrolerom robota. U nastavku ovog poglavlja su detaljno opisane komponente sustava.




Sl. 5.1. Komponente sigurnosnog sustava

## 5.1. Senzor

Protok podataka u sigurnosnom sustavu započinje prikupljanjem podataka o okolini RGB-D senzorom. U ovom sustavu se koristi Kinect RGD-B kamera, koja se pokreće korištenjem programskog paketa OpenNI [20] . Kada se pokrenu potrebni programi, na ROS *Master* čvoru je dostupan popis tema na koje Kinect senzora objavljuje podatke snimanog prostora. Najbitnije teme su:

- camera/rgb/image_color (sensor_msgs/Image)
- camera/rgb/camera_info (sensor_msgs/CameraInfo)
- camera/depth_registered/image (sensor_msgs/Image)
- camera/depth_registered/camera_info (sensor_msgs/CameraInfo)
- camera/depth_registered/points (sensor_msgs/PointCloud2)

Prve dvije teme predstavljaju sliku u boji i parametre kalibracije kamere, na druge dvije teme se objavljuju dubinske slike snimljene scene i parametri kalibracije za dubinske slike, dok se na posljednju temu objavljuju 3D točke snimljenog prostora. Ove teme se koriste u metodi za detekciju i praćenje. Čvorovi metode se pretplaćuju na određenu temu te tako preuzimaju željene informacije.

U sustavu je važno poznavati prostorni odnos robota i ljudi. Kako bi se odradila pozicija detektiranog čovjeka u odnosu na robota, potrebno je obaviti kalibraciju kamere prema robotu. Ovim procesom želi se pronaći transformacija između koordinatnog sustava kamere i koordinatnog sustava smještenog u bazi robotskog manipulatora. Kalibracija se obavlja uz pomoć ArUco markera postavljenog na alat robotskog manipulatora. Kamera se postavlja u blizini robota tako da je vidljiv prostor u kojem se ljudi mogu kretati i gdje se robot može vidjeti. Program za kalibraciju kamere prema robotu postavlja alat robota u različite pozicije ispred kamere. Kamera detektira marker te računa položaj markera u prostoru. Dobivaju se koordinate alata robota u koordinatnom sustavu robota i u koordinatnom sustavu kamere, iz čega se može izračunati transformacijska matrica koordinatnih sustava $T_{RC}$. Ako se zanemari 'z' komponenta i rotacija koordinatnih sustava, problem položaja baze robota u odnosu na kameru svodi se na pomak u dvodimenzionalnom prostoru. Tada je samo potrebno pronaći translaciju baze robota po x-osi i y-osi koordinatnog sustava kamere. Informacije o poziciji baze robota se koriste u implementaciji sigurnosnih uvjeta.

## 5.2. Metoda za detekciju i praćenje ljudi

Za potrebe detekcije i praćenja ljudi odabrana je metoda implementirana u SPENCER razvojnom okviru. SPENCER metoda javno je dostupna na GitHubu [8]. Razvojni okvir može se instalirati naredbama u terminalu prikazanima na slici 5.2.:

**Linija Kod**

```
1:      mkdir -p spencer-people-tracking-ws/src
2:      cd spencer-people-tracking-ws/src
3:      git clone https://github.com/spencer-
        project/spencer_people_tracking.git
4:      git checkout $ROS_DISTRO
5:      rosdep update
6:      rosdep install -r --from-paths . --ignore-src
7:      cd ..
8:      catkin config --init --cmake-args -DCMAKE_BUILD_TYPE=RelWithDebInfo
9:      catkin build -c -s
10:     source devel/setup.bas
```

Sl. 5.2. Terminal naredbe za instalaciju SPENCER razvojnog okvira

SPENCER okvir je podijeljen na 7 skupina paketa: *detection*, *launch*, *messages*, *simulation*, *tracking*, *utils* i *visualization*. Programi, alati i podatci potrebni za pokretanje kompletnog programa za detekciju i praćenje raspodijeljeni su u ove skupine paketa zbog veće modularnosti i održivosti komponenti. U *detection* skupini nalaze se implementirani detektori ljudi, a u skupini *tracking* nalaze se implementirane metode praćenja ljudi. Svi nestandardni oblici ROS poruka koje se koriste u razvojnom okviru definirani su u *messages* skupini. Paketi skupina *simulation, utils* i *visualization* sadrže alate za simulaciju, obradu i vizualizaciju podataka za snimljene skupove podataka ili snimanje u stvarnom vremenu. U *launch* skupini nalaze se skripte potrebne za pokretanje komponenti programa, od kojih su najvažnije *tracking_on_bagfile.launch*, skripta koja pokreće program koji koristi podatke sačuvane u *.bag* formatu, i *tracking_single_rgbd_sensor.launch*, koja pokreće program za stvarni senzor.

Datoteka za pokretanje programa *tracking_on_bagfile.launch* koristi se za testiranje metode korištenjem snimljenih podataka. Glavne promjene potrebne za adaptaciju metode napravljene su u ovoj datoteci. U početnim testiranjima metode korišten je KTP skup podataka [11]. Skup podataka sastoji se od nekoliko snimki ljudi snimljenih Kinect RGB-D kamerom. U nastavku su opisani važni dijelovi datoteke, te promjene potrebne za adaptaciju KTP skupa podataka. Na slici 5.3. prikazani su argumenti *tracking_on_bagfile.launch* datoteke. Prvim argumentom datoteke, *bagfile_folder*, određena je putanja do snimljenih podataka. Datoteka za pokretanje programa namijenjena je za korištenje sa četiri različita senzora, no za potrebe ovog projekta potreban je

samo jedan. Zato je samo *front_rgbd* argumentu postavljena *true* vrijednost, a korištenje druga tri senzora je onemogućeno. Na linijama 6-9 definirani su argumenti za odabir detektora koji se želi koristiti. Detektor koji se koristi u realizaciji sigurnosnog sustava je detektor gornjeg dijela tijela čovjeka, označen argumentom *use_upper_body_detector*. Argumentom *static_map* se pokazuje je li dostupna statična karata prostora, koja služi za potencijalno izbjegavanje krivih detekcija na mjestima statičnih objekata. Argumenti na linijama 11-15 služe za upravljanje vizualizacijom programa u RViz-u. Uz osnovne argumente, dodana su dva nova. Argument *height_above_ground* postavlja visinu na koju je postavljen RGB-D senzor. Visina senzora koristi se kod estimacije ravnine poda i kod definiranja graničnih okvira praćenih ljudi. Pretpostavljena vrijednost visine je 1.6 metara, visina koja je korištena i u izvornom programu. Argumentom *ds* je označeno korištenje dodatnih izmjena datoteke za pokretanje, u slučaju da se koristi skup podataka različit izvornom skupu za testiranje. Izvorni skup za testiranje, javno dostupan na GitHub stranici razvojnog okvira, ima posebnu strukturu koja omogućuje lako pokretanje s izvornom *launch* datotekom, dok se drugi skupovi podataka moraju dodatno izmijeniti.

**Linija Kod**

```
1:      <arg name="bagfile_folder" default="$(env
        HOME)/.ros/spencer_tracking_rgbd_2dlaser_example_bagfiles/"/>

2:      <arg name="front_laser" default="false"/>
3:      <arg name="front_rgbd" default="true"/>
4:      <arg name="rear_rgbd" default="false"/>
5:      <arg name="rear_laser" default="false"/>

6:      <arg name="use_pcl_detector" default="false"/>
7:      <arg name="use_hog_detector" default="false"/>
8:      <arg name="use_upper_body_detector" default="true" unless="$(arg
        use_pcl_detector)"/>
9:      <arg name="use_upper_body_detector" default="false"    if="$(arg
        use_pcl_detector)"/>

10:     <arg name="static_map" default="false"/>

11:     <arg name="loop" default="false"/>
12:     <arg name="rate" default="0.3"/>
13:     <arg name="visualization" default="true"/>
14:     <arg name="rviz_config_file" default="$(find
        spencer_people_tracking_launch)/rviz/tracking-rgbd-laser.rviz"
        unless="$(arg ds)"/>
15:     <arg name="rviz_config_file" default="$(find
        spencer_people_tracking_launch)/rviz/tracking-rgbd-robotarm-sim.rviz"
        if="$(arg ds)"/>

16:     <arg name="height_above_ground" default="1.6"/>
17:     <arg name="ds" default="false"/>
```

Sl. 5.3. Argumenti *tracking_on_bagfile.launch* datoteke

Kako bi se KTP skup podataka uspješno koristio u SPENCER sustavu programa za detekciju i praćenje ljudi, prvo je potrebno promijeniti imena tema snimljenih podataka. Promjenom imena tema će se omogućiti da detektor ljudi pronađe odgovarajuće podatke preko glavnog ROS čvora. Imena tema u snimljenom skupu podatak mijenja se korištenjem *remap* naredbe i korištenjem *relay* čvora iz *image_transport* paketa. Naredba *remap* prima dva argumenta: početno ime teme i željeno ime teme. Izvođenjem ove naredbe, tema početnog imena će se proslijediti svakom čvoru koji se pretplati na temu definiranu argumentom željenog imena. Tako se obavlja prividna promjena imena teme, gdje je ime zamijenjeno za vrijeme obavljanja programa, ali u izvornom skupu podatka ostaje nepromijenjeno. Čvor *relay* ima sličnu funkciju kao naredba *remap*. Ovim čvorom se podatci jedne teme mogu ponovno objaviti na temu drugačijeg imena. Za razliku od naredbe *remap,* ovaj čvor omogućuje korištenje i stare teme i nove teme tako da sve poruke koje stignu na određenu temu ponovo objavljuje na novoj temi. U nekim slučajevima je potrebno prilagoditi i imena tema i format podataka u ulaznom skupu podataka, obično jer se podatci sažimaju zbog smanjivanja veličine datoteke. U tim slučajevima može se koristiti *republish* čvor *topic_tools* ROS paketa, koji se pretplati na određenu temu te objavljuje iste podatke ali u drugom formatu na drugu temu. Primjer korištenja *remap* naredbe i *relay* i *republish* čvora prikazan je na slici 5.4..

**Linija** **Kod**

```
1:  <remap if="$(arg ds)" from="/spencer/sensors/camera/rgb/camera_info"
    to="/spencer/sensors/rgbd_front_top/rgb/camera_info"/>
2:  <node name="relay_odom" type="relay" pkg="topic_tools" args="/odom
    /spencer/sensors/odom"/>
3:  <node name="republish_rgb" type="republish" pkg="image_transport"
    args="compressed in:=/spencer/sensors/camera/rgb/image_color raw
    out:=/spencer/sensors/rgbd_front_top/rgb/image_rect_color"/>
```

Sl. 5.4. Primjer korištenja naredbe *remap* i *relay* i *republish* čvora

Osim preimenovanja tema skupa podataka, potrebno je definirati koordinatne sustave koji nedostaju, a to su koordinatni sustavi baze robota, karte prostora i koordinatni sustavi kamere. Ovi sustavi mogu se postaviti transformiranjem poznatih koordinatnih sustava korištenjem *static_transform_publisher* čvora iz *tf* ROS programskog paketa. Na slici 5.5. je prikazan primjer programskog koda kojim se postavljaju potrebne koordinatne osi.

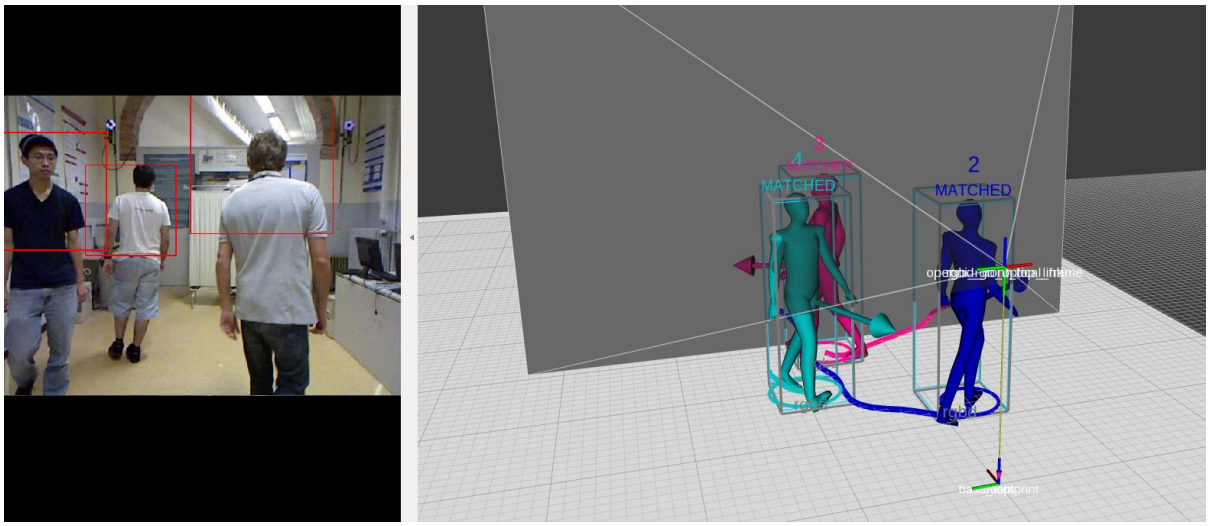
**Linija** **Kod**

```
1:  <node name="tf_openni" pkg="tf" type="static_transform_publisher"
    args="0 0 0 -1.570796 0 -1.570796 rgbd_front_top_link
    openni_rgb_optical_frame 10"/>
```

Sl. 5.5. Postavljanje koordinatnih sustava koji nedostaju

U ostatku *tracking_on_bagfile.launch* datoteke pozivaju se druge *launch* datoteke zaslužne za pokretanje reprodukcije skupa podataka iz *.bag* datoteke, pokretanje detektora i programa za praćenje. Uz navedene promjene, podatci iz KTP skupa mogu se koristiti kao ulazni podatci programa za detekciju i praćenje. Logika adaptacije programa može se primijeniti i na druge skupove podataka. U završnom testiranju sigurnosnog sustava koristi se novi skup podataka snimljen Kinect kamerom koji je prilagođen SPENCER metodi opisanim postupkom.

Na slici 5.6. prikazan je isječak KTP skupa podataka nakon generiranja rezultata detekcije i praćenja SPENCER metodom, vizualiziran u RViz alatu. Na RGB slici je vidljiv granični okvir koji generira detektor gornjeg dijela tijela, a na desnom dijelu slike prikazana je 3D vizualizacija praćenih ljudi te 3D granični okvir koji generira program praćenja. Rezultat detektora je poruka formata *DetectedPersons*, koja sadrži listu *DetectedPerson* poruka i zaglavlje u kojem su definirane informacije o vremenu stvaranja poruke, inkrementalni identifikacijski broj i identifikacijska oznaka referentnog sustava. Poruka detekcije sadrži identifikacijski broj čovjeka, pouzdanost i položaj čovjeka. Poruke preuzima program za praćenja, koji na temelju podatka o detektiranim ljudima obnavlja odgovarajuće objekte praćenja, odnosno obnavlja pretpostavljenu poziciju, orijentaciju i brzinu praćenih ljudi. Program praćenja objavljuje poruke tipa *TrackedPerson* u listi unutar *TrackedPersons* poruke, koja sadrži i ostale podatke u zaglavlju kao i *DetectedPersons* tip poruke.



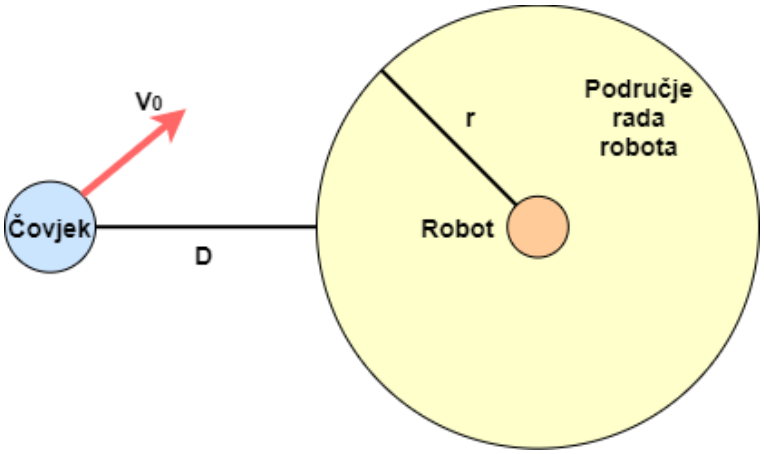

Sl. 5.6. Rezultati SPENCER metode za isječak KTP skupa podataka

Datoteka *tracking_on_robot.launch* koristi se za pokretanje SPENCER metode za snimanje u stvarnom vremenu. Za korištenje ove datoteke nisu potrebne veće promjene, može se pokrenuti s Kinect kamerom uz postavljanje vrijednosti argumenata za visinu senzora i za pokretanje određenog detektora. Koristi se OpenNi program za pokretanje Kinect kamere, a podatci senzora se objavljuju na *rgbd_front_top* skupini tema. Program koji se pokreće ovom datotekom daje iste rezultate kao i program pokrenut *tracking_on_bagfile.launch* datotekom. Program objavljuje *DetectedPersons* i *TrackedPersons* poruke.

Za potrebe ovog projekta isključena je opcija za fuziju različitih senzora, jer se koristi samo jedan RGB-D Kinect senzor. Ovim se znatno ubrzava rad programa. Isključena je i opcija za estimaciju društvenih odnosa, jer te informacije nisu važne za funkcionalnost ovog projekta. Kao što je opisano u poglavljima 3.2 i 3.3, komponente programa za detekciju i praćenje ne zahtijevaju mnogo resursa računala, nema potrebne za GPU-om, mogu se pokretati na standardnim osobnim računalima. Program objavljuje rezultate frekvencijom 20-30 Hz, ovisno o dodatnim opterećenjima na procesoru računala.

## 5.3. Sigurnosni uvjeti

Sigurnosni sustav za upravljanje robotom mora ograničiti kretnju robota ako se u radnom prostoru nalaze ljudi. No, kako ljudska kretnja nije uvijek jednolična, potrebno je definirati takve uvjete upravljanja robotom ovisne o položaju i brzini ljudi da robot nikad ne stupi u kontakt sa čovjekom tijekom rada. Kretanje robota može se ograničiti smanjenjem brzine gibanja ili potpunim zaustavljanjem robota. Informacije o ljudima u radnome prostoru robota mogu se preuzeti od SPENCER metode za detekciju i praćenje ljudi, koja daje podatke o poziciji, orijentaciji i brzini ljudi. Na temelju informacija algoritma praćenja ljudi, moguće je implementirati sigurnosne uvjete za robotski manipulator. U svrhu implementacije sigurnosnih uvjeta, problem prostornog odnosa robotskog manipulatora i ljudi može se svesti na 2D prostor. Područje u kojem robot obavlja radnje može se predstaviti kružnicom sa središtem u bazi robota, gdje je radijus kružnice doseg robotske ruke. Čovjek u promatranom 2D prostoru je označen položajem i vektorom brzine. Opisani postav problema prikazan je na slici 5.7., gdje je s $V_o$ označen vektor brzine čovjeka, D je udaljenost između čovjeka i radnog područja robota, a r je radijus radnog područja robota.




Sl. 5.7. 2D prikaz čovjeka i područja rada robota

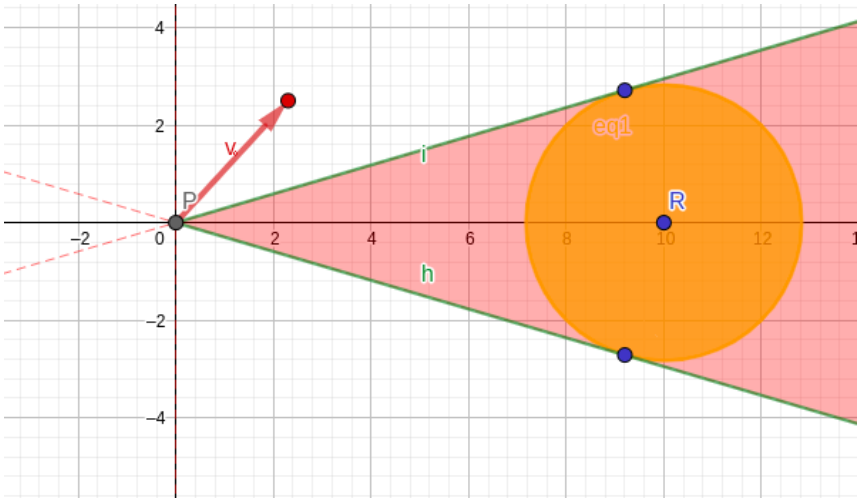
Zaustavljanje robota nakon što je čovjek ušao u radno područje može biti prekasno, ako se robot točno u tom trenutku giba u istom području. Stoga su definirana proširenja za radni prostor robota, odnosno za radijus r, ovisna o brzini gibanja čovjeka, tako da sigurnosni sustav stigne zaustaviti robota prije kolizije. Proširenja radnog prostora robota ovisna o brzini čovjeka su definirana jednadžbom 5.1., gdje je s r označena vrijednost radijusa radnog područja robota u nekom trenutku, a s $r_0$ označava minimalnu vrijednost radijusa, odnosno domet robotske ruke. Radijus radnog područja robota može poprimiti tri vrijednosti: normalni doseg robotske ruke, doseg uvećan za 10% i doseg uvećan za 50%.

$$r = \begin{cases} r_0, & ||V_o|| < 0.1 \left[\frac{m}{s}\right] \\ 1.1r_0, & 0{,}1 \left[\frac{m}{s}\right] \leq ||V_o|| < 1 \left[\frac{m}{s}\right] \\ 1.5r_0, & ||V_o|| \geq 1 \left[\frac{m}{s}\right] \end{cases} \quad (5.1.)$$

Robot se mora zaustaviti ako je detektiran čovjek u radnom području. Ako se čovjek giba velikom brzinom prema robotu, promjena radijusa radnog područja omogućuje bržu reakciju sigurnosnog sustava. Uz ovaj uvjet, implementirani su uvjeti koji će ograničiti kretnju robota za posebne slučajeve definirane jednadžbom 5.2. U jednadžbi je s $V_{ee}$ označen koeficijent brzine alata robotskog manipulatora čija se brzina ograničava, $V_o$ označava brzinu čovjeka koji se prati. $|\delta|$ je apsolutna vrijednost kuta između vektora brzine čovjeka $V_o$ i tangente na kružnicu radnog područja robota koja prolazi kroz ishodište koordinatnog sustava čovjeka, a $\gamma$ je kut između pravca koji prolazi kroz ishodišta koordinatnih sustava čovjeka i robota i vektora brzine čovjeka $V_o$.

$$V_{ee} = \begin{cases} 50\% & V_o < 0.1\ \left[\frac{m}{s}\right] \\ 25\% & 0.1\ \left[\frac{m}{s}\right] \le V_o < 2\ \left[\frac{m}{s}\right] \\ 0\% & \begin{cases} V_o \ge 2\ \left[\frac{m}{s}\right] \\ |\delta| \le \gamma \\ D \le 0 \end{cases} \\ 100\% & inače \end{cases} \tag{5.2.}$$

Prvi uvjet usporava brzinu gibanja alata robota na 50% dopuštene brzine kada je u promatranom prostoru prisutan čovjek čija je brzina kretanja manja od 0.1 [m/s]. Drugi uvjet smanjuje originalnu brzinu na 25% kada je prisutan čovjek koji se giba brzinom između 0.1 i 2 [m/s]. Posljednji uvjet definira događaje koji će prouzročiti zaustavljanje robota. Robot će se zaustaviti ako je u bilo kojem trenutku u promatranom području detektiran čovjek koji se giba brzinom većom od 2 [m/s], ako je smjer gibanja čovjeka ravno prema robotu ili ako se čovjek nalazi unutar radnog prostora robota. Uvjet za zaustavljanje robota za slučaj ovisan o orijentaciji čovjeka definiran je tako da će se robot zaustaviti u trenutku kada je položaj vektora brzine čovjeka unutar prostora ograničenog tangentama na kružnicu radnog prostora robota koje prolaze kroz ishodište koordinatnog sustava čovjeka. Vizualni prikaz ovog uvjeta prikazan je na slici 5.8. , gdje je crvenom bojom označen dio koordinatnog sustava, postavljenog u ishodište pozicije čovjeka, u kojem se vektor $V_0$ mora nalaziti da bi se uvjet aktivirao, odnosno robot zaustavio. Uvjet se uvijek provjerava tako da se sve komponente prenesu u koordinatni sustav čije je ishodište na mjestu detektiranog čovjeka, a x-os prolazi kroz središte robota.




Sl. 5.8. 2D prikaz uvjeta za zaustavljanje $|\delta| \le \gamma$

Uvjeti su implementirani u obliku ROS čvora, napisanog C++ programskim jezikom. U nastavku su opisani važni dijelovi koda. Na slici 5.9. prikazan je konstruktor klase implementiranih uvjeta, nazvan *HumanRobotInteraction*. Konstruktor stvara instancu klase uz postavljanje osnovnih parametara, postavlja se ime teme s koje će čvor preuzimati podatke i stvara se objekt *move_group* iz MoveIt biblioteke. U tijelu konstruktora nalazi se samo poziv *init* metode, koja je opisana u nastavku.

***Linija Kod***

```
1:   namespace RobotControl {
2:       // constructor()
         HumanRobotInteraction::HumanRobotInteraction(const ros::NodeHandle
3:           &node_handle, const ros::NodeHandle &private_node_handle):
4:       nh(node_handle),
5:       pnh(private_node_handle),
         input_topic(
6:           "/spencer/perception/tracked_persons_orientation_fixed"),
7:       PLANING_GROUP("manipulator"),
         move_group(moveit::planning_interface::MoveGroupInterface(
8:           PLANING_GROUP)){
9:           this->init();
10:      }
```

Sl. 5.9. Konstruktor klase *HumanRobotInteraction*

Na slici 5.10. prikazan je kod *init* metode klase *HumanRobotInteraction*. Metoda se poziva pri stvaranju instance klase. Unutar ove metode se inicijaliziraju varijable i postavljaju osnovne vrijednosti. Na liniji koda 3 i 4 inicijaliziraju se mjerači vremena, koji će se kasnije koristiti za osiguranje izlaska iz određenih stanja. Na liniji 5 stvara se pretplatnik na temu na koju SPENCER metoda objavljuje rezultate praćenja. Na linijama 8-12 definirani su parametri koji određuju poziciju robota i varijable upravljanja kretanje robota. Varijabla *stopMovement* koristi se kao znak za zaustavljanje, odnosno pokretanje robota. Varijabla *velocityFactor* koristi se za postavljanje željene brzine kretnja robota. Pozicija robota može se učitati i iz datoteke koja sadrži transformaciju koordinatnih sustava kamere i robota, što je definirano na liniji 15. Naredba se koristi ako je korištena nova pozicija kamere za koju je potrebno obaviti kalibraciju kamere prema robotu. Na linijama 19 i 20 definirana su dva položaja robotskog manipulatora, koji će se koristiti za izvođenje jednostavne kretnje robota. Posljednjim linijama koda, linije 23 i 24 metode, postavljaju se konstante važne za upravljanje robotom. Metodom *setPlanningTime* postavlja se maksimalno dozvoljeno vrijeme za planiranje putanje, a metoda *allowReplanning* omogućuje ponovno planiranje putanje.

***Linija Kod***

```
1:   // init()
2:   void HumanRobotInteraction::init() {
         stop_timer = pnh.createTimer(ros::Duration(4.0),
3:           &HumanRobotInteraction::stopTimer, this);
         speed_timer = pnh.createTimer(ros::Duration(2.0),
4:           &HumanRobotInteraction::speedTimer, this);
         tracksSubscriber = pnh.subscribe(input_topic, 3,
5:           &HumanRobotInteraction::newTrackedReceived, this);
6:
7:       // default robot position, mainly for testing
8:       robotPosition_x = 2.95;
9:       robotPosition_y = 2.0;
10:      robotRadius = 1.81;
11:      stopMovement = false;
12:      velocityFactor = 1.0;
13:
14:      // real robot position
15:      tcrFileName = "/home/dario/Code/calib_ws/src/abbcmd/TCR.txt";
16:      //setCalibRobotPosition(tcrFileName);
17:
18:      // two robot poses
         setPose(first_pose, 0.94768, 0.03007, 1.7031, 0.0, 0.99831,
19:          0.0, 0.05813);
         setPose(second_pose, 0.82601, -0.3745, 1.624, 0.0, 0.99789,
20:          0.0, -0.064918);
21:
22:      move_group.setPlanningTime(10.0);
23:      move_group.allowReplanning(true);
24:      //move_group.setPlannerId();
25:  } // init()
```

Sl. 5.10. Metoda *init*

Na slici 5.11. prikazan je kod metode *newTrackedReceived*, koja prima podatke o ljudima od SPENCER metode. Ova metoda se poziva svaki put kada je dostupna nova poruka na pretplaćenoj temi */spencer/perception/tracked_persons_orientation_fixed*. Na liniji koda 6 definirana je *for* petlja kojom se prolazi kroz sve poruke oblika *TrackedPerson* sadržane unutar poruke preuzete na pretplaćenoj temi, koja ima oblik *TrackedPersons*. Za svaki primljen objekt praćenja provjerava se je li praćena osoba u tom trenutku potvrđena detekcijom, u kojem slučaju se nastavlja s radom. Za svaki takav objekt bilježi se pozicija i brzina praćenog čovjeka, te se poziva metoda *robotControlStrategy* u kojoj se provjeravaju uvjeti za upravljanje robotom.

***Linija Kod***

```
1:   // newTrackedReceived()
     void HumanRobotInteraction::newTrackedReceived(const
2:       abb_control::TrackedPersons::ConstPtr& trackedPersons) {
3:       float trackPosition_x, trackPosition_y = 0;
4:       float trackSpeed_x, trackSpeed_y = 0;
5:
6:       for (int i = 0; i < trackedPersons->tracks.size(); i++) {
             const abb_control::TrackedPerson& track =
7:               trackedPersons->tracks[i];
8:
9:           // if the track is currently matched by a detector
10:          if (track.is_matched){
11:              trackPosition_x = track.pose.pose.position.x;
12:              trackPosition_y = track.pose.pose.position.y;
13:              trackSpeed_x = track.twist.twist.linear.x;
14:              trackSpeed_y = track.twist.twist.linear.y;
15:
                 this->robotControlStrategy(trackPosition_x,
16:                  trackPosition_y, trackSpeed_x, trackSpeed_y);
17:          }
18:      }
19:  } // newTrackedReceived()
```

Sl. 5.11. Metoda *newTrackedReceived*

U prvom dijelu metode *robotControlStrategy* računa se ukupna brzina čovjeka iz x i y komponenti brzine. Na temelju brzine postavlja se radijus radnog područja robota promjenom vrijednosti varijable *robotEnvelope*. Nakon toga, računa se udaljenost čovjeka od robota. Dio metode *robotControlStrategy* prikazan je na slici 5.12.. U ovom dijelu provjeravaju se uvjeti za zaustavljanje. Uvjeti su poredani tako da se kritični uvjeti provjeravaju prvi. Prvo se provjerava uvjet za udaljenost čovjeka od radnog područja robota, zatim uvjet zaustavljanja zbog velike brzine gibanja, i na kraju uvjet za orijentaciju gibanja čovjeka. Ako je bilo koji od uvjeta za zaustavljanje ispunjen, varijabla *stopMovement* postavlja se na vrijednost *true* i prelazi se na sljedeću grupu provjere. Ako je aktiviran signal za zaustavljanje, robot se zaustavlja te brojač vremena *stop_timer* započinje odbrojavati. Ovaj brojač osigurava da se robot neće nastaviti gibati dok ne prođe minimalni period vremena, a odbrojavanje se resetira pri svakom ispunjenju uvjeta za zaustavljanje. Ako je aktiviran signal zaustavljanja, metoda prestaje s provjerama uvjeta nakon obnavljanja brojača vremena. U suprotnom, prelazi se u blok provjera uvjeta ograničenja brzine robota.

***Linija Kod***

```
1:    // stop conditions
2:    if (trackDistance <= robotEnvelope){
3:        stopMovement = true;
4:    }
5:    else {
6:        if (trackSpeed >= 2){
7:            stopMovement = true;
8:        }
9:        else {
10:           alpha = atan2f(trackSpeed_y, trackSpeed_x);
11:           beta = atan2f((robotPosition_y-trackPosition_y),
12:                       (robotPosition_x-trackPosition_x));
13:           gamma = asinf(robotEnvelope/trackDistance);
14:
15:           if (fabsf(alpha-beta) <= gamma){
16:               stopMovement = true;
17:           }
18:       }
19:   }
20:   if (stopMovement){
21:       if (!stop_timer.hasStarted()){
22:           move_group.stop();
23:           resumePlaning = false;
24:           ROS_WARN("Stop condition met, robot motion disabled!");
25:       }
26:
27:       stop_timer.stop();
28:       stop_timer.start();
29:       return;
30:   }
```

Sl. 5.12. Prvi dio *robotControlStrategy* metode

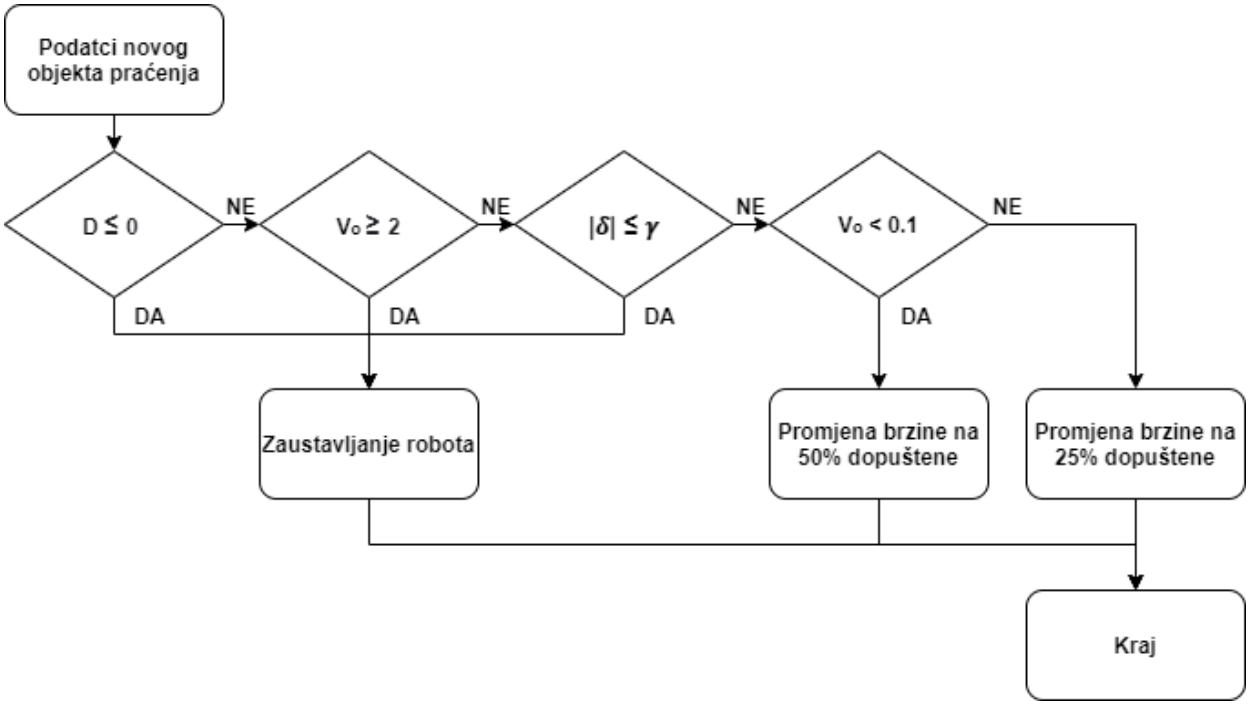
Ako nije aktiviran signal za zaustavljanje robota, i mjerač vremena nije pokrenut, postavlja se ograničenje na brzinu gibanja robota. Na slici 5.13. prikazan je drugi dio koda *robotControlStrategy* metode. U varijabli *velocityFactor* zapisana je trenutna vrijednost brzine gibanja robota, a varijablom *newVelocityFactor* postavljena je vrijednost brzine robota koja je rezultat provjere uvjeta za trenutni objekt detekcije. Ako su vrijednosti ovih varijabli različite, putanja robota se ponovno računa s novom brzinom gibanja. Mjerač vremena obnavlja se pri svakoj detekciji unutar prostora od interesa, ukoliko nije aktiviran uvjet zaustavljanja robota.

***Linija Kod***

```
1:    else if (!stop_timer.hasStarted()){
2:        if (trackSpeed < 0.1){
3:            newVelocityFactor = 0.05;
4:        }
5:        else {
6:            newVelocityFactor = 0.025;
7:        }
8:        if (newVelocityFactor != velocityFactor){
9:            if (!speed_timer.hasStarted()){
10:               ROS_WARN("Max velocity changed, replaning movement!");
11:           }
12:           velocityFactor = newVelocityFactor;
13:           move_group.stop();
14:       }
15:       speed_timer.stop();
16:       speed_timer.start();
17:   }
```

Sl. 5.13. Drugi dio *robotControlStrategy* metode

Opisani slijed provjere uvjeta u *robotControlStrategy* metodi prikazan je dijagramom toka na slici 5.14. Provjera uvjeta je poredana tako da se kritični uvjeti provjeravaju prvi. Ako uvjeti za ograničenje kretnje robota nisu ispunjeni, robot se giba maksimalnom dozvoljenom brzinom.




Sl. 5.14. Slijed provjere uvjeta za upravljanje robotom

## 5.4. Upravljanje robotom

Razvijeni su ROS alati za spajanje i upravljanje ABB robotom, no za pojedine modele robota potrebno je definirati dodatne alate. Za upravljanje robotom ABB IRB 2400L korišten je konfiguracijski paket opisanu u [21]. Konfiguracijski paket omogućava upravljanje robotom pomoću osobnog računala, a može se koristiti i robot simuliran u RVizu. ABB robota spojenog na IRC5 kontroler moguće je povezati s osobnim računalom korištenjem *Ethernet* veze. Preko *move_group* čvora MoveIt ROS paketa moguće je primati informacije o stanju robota ili slati zahtjeve kontroleru robota. Implementiran je program za upravljanje robotskim manipulatorom koji izvodi jednostavnu kretnju između dva položaja. Na slici 5.15. je prikazan dio koda klase *HumanRobotInteraction*, koji se izvodi u beskonačnoj petlji.

***Linija Kod***

```
1:    while (ros::ok()){
2:        if (resumePlaning){
3:            current_pose = move_group.getCurrentPose().pose;
4:            move_group.setPoseTarget(current_pose);
5:            if (poseFlag==0){
6:                if (comparePosition(current_pose, first_pose) == 0) {
7:                    move_group.setPoseTarget(first_pose);
8:                }
9:                else {
10:                   poseFlag = 1;
11:                   continue;
12:               }
13:           }
14:           else if (poseFlag==1){
15:               if (comparePosition(current_pose, second_pose) == 0) {
16:                   move_group.setPoseTarget(second_pose);
17:               }
18:               else {
19:                   poseFlag = 0;
20:                   continue;
21:               }
22:           }
23:           move_group.setMaxVelocityScalingFactor(velocityFactor);
24:           move_group.plan(move_plan);
25:           move_group.execute(move_plan);
26:       }
27:   }
```

Sl. 5.15. Glavna petlja programa za upravljanje robotom

Petlja se izvodi dok je ROS aktivan, konstantno se računa putanja koja se šalje robotu. Varijabla *resumePlaning* služi za zaustavljanje ponovnog planiranja robota ako je aktiviran uvjet za zaustavljanje robota, odnosno pokrenut je mjerač vremena. Korištenjem *move_group* sučelja, od kontrolera se preuzima trenutni položaj robota. U petlji se naizmjenično postavljaju ciljevi položaja robota tako da se trenutni položaj robota uspoređuje s ciljnim i u slučaju da su položaji jednaki postavlja se sljedeći ciljni položaj. Na liniji 23 slike 5.15. postavlja se maksimalna brzina koju robot može imati pri izvođenju gibanja. Na linijama 24 i 25 koda napisane su naredbe za pokretanje planiranja i izvođenje isplanirane kretnje.

U ovoj implementaciji, ograničenja na kretnju robota se postavljaju tako da se prvo prekida trenutna radnja robota. Nakon toga, putanja robota se ponovno planira s postavljenim ograničenjima, te se plan šalje kontroleru na izvođenje. Ovakva implementacija ograničena je mogućnostima MoveIt alata i kontrolere robota, nije optimalna za upravljanje u stvarnom vremenu. Bolji način upravljanja robotom može se ostvariti korištenjem novog ROS programa za upravljanje ABB robotom [22], koji pruža opciju direktnog upravljanja robotom, no za njegovo korištenje je potrebno napredno znanje o ROS-u.

# 6. TESTIRANJE

Izrađen je skup podataka snimljen Kinect kamerom nazvan HDR skup podataka. Skup se sastoji od niza video snimki ljudi u različitim scenama i uvjetima, kojima se žele simulirati stvarni događaji u industrijskim okolinama u kojima roboti i ljudi dijele prostor. Skup je izrađen s ciljem testiranja komponenata sigurnosnog sustava kao što je detektor ljudi, algoritam praćenja, procjena pozicije, brzine i orijentacije čovjeka, te aktivacija sigurnosnih uvjeta. Skup je snimljen u *.bag* formatu za lako korištenje u ROS-u. Na slici 6.1. prikazana je slika iz snimke HDR skupa u kojem se više ljudi kreće ispred kamere. Na slici je vidljiv ArUco marker koji predstavlja robotski manipulator, a na podu su trakom označene tri kružnice radnih prostora robota opisane jednadžbom 5.1.

Sl. 6.1. Slika HDR skupa podataka

Skup podataka podijeljen je u 3 skupine: test detektora, test predviđanja brzine i test uvjeta za zaustavljanje robota. Test detektora sadrži tri snimke kretanja čovjeka koji je djelomično zaklonjen predmetom i snimke više ljudi koji se kreću ispred kamere. Ovom skupinom snimki želi se testirati kako će metoda detekcije raditi u slučajevima kada čovjek nije u potpunosti vidljiv na snimci. Test predviđanja brzine sastoji se od 10 snimki na kojima ljudi hodaju ispred kamere različitim brzinama. Snimljena su četiri intervala brzina hodanja: $v \geq 2.0$ [m/s], $1.0$ [m/s] $\leq v < 2.0$ [m/s], $0.1$ [m/s] $\leq v < 1.0$ [m/s] i $v \leq 1.0$ [m/s]. Uz promjenu brzine, u snimkama su i drugačiji smjerovi hodanja: paralelno, okomito i dijagonalno u odnosu na z os koordinatnog sustava kamere koji se proteže od središta kamere prema lijevoj strani prostora. Za brzinu $v \geq 2.0$ [m/s] snimljen je jedan

smjer. U posljednjoj testnoj skupini nalaze se tri snimke: u prvoj se čovjek giba od izvan radnog prostora robota prema robotu, a u druge dvije snimke čovjek se kreće pokraj robota te mijenja smjer kretanja direktno prema robotu. Prvom se snimkom testira uvjet zaustavljanja ovisan o udaljenosti čovjeka od kružnice radnog područja robota, a sa posljednje dvije snimke se testira uvjet ovisan o smjeru gibanja čovjeka u odnosu na radno područje robota.

HDR skup podataka korišten je za testiranje sigurnosnog sustava. Snimljene su ciljane vrijednosti svakog testa po vremenskim oznakama koje su uspoređene s ručno označenim rezultatima. Iz usporedbe dobivenih i označenih rezultata računa se odziv, preciznost i točnost. Ručno označeni podatci mogu biti nepouzdani zbog neodređenosti događaja kao što je točan trenutak prvog pojavljivanja čovjeka na slici, točan vremenski interval kada se čovjek giba nekom brzinom i drugi. U tablici 6.1. prikazani su rezultati dobiveni provođenjem testiranja sustava.

Tablica 6.1. Rezultati testiranja sustava.

| | **Odziv** | **Preciznost** | **Točnost** |
|---|---|---|---|
| **Test detektora** | | | |
| **Test 1** | 0.4211 | 1.0 | 0.4761 |
| **Test 2** | 0.0294 | 1.0 | 0.0294 |
| **Test 3** | 0.4953 | 1.0 | 0.4953 |
| **Test 4** | 0.9610 | 0.9867 | 0.9535 |
| **Test predviđanja brzine** | | | |
| **$v \geq 2$ [m/s]** | 0.8571 | 0.5455 | 0.7273 |
| **$1.0 \leq v < 2.0$ [m/s]** | | | |
| **Smjer 1** | 0.5 | 0.5 | 0.5556 |
| **Smjer 2** | 0.8235 | 0.7778 | 0.72 |
| **Smjer 3** | 0.8462 | 1.0 | 0.875 |
| **$0.1 \leq v < 1.0$ [m/s]** | | | |
| **Smjer 1** | 1.0 | 0.75 | 0.7872 |
| **Smjer 2** | 0.6875 | 1.0 | 0.7436 |
| **Smjer 3** | 0.4761 | 0.8333 | 0.4583 |
| **$v \leq 0.1$ [m/s]** | | | |
| **Smjer 1** | 0.4598 | 0.9756 | 0.4545 |
| **Smjer 2** | 0.292 | 0.9855 | 0.2936 |
| **Smjer 3** | 0.3588 | 0.9683 | 0.3547 |
| **Test zaustavljanja robota** | | | |
| **Test 1** | 1.0 | 1.0 | 1.0 |
| **Test 2** | 0.8929 | 1.0 | 0.9318 |
| **Test 3** | 0.8421 | 1.0 | 0.9063 |

Testiranjem se pokazalo da korišteni detektor ljudi slabo reagira na zaklonjenost čovjeka, te su tako najlošiji rezultati dobiveni u prvoj skupini testova. Najveći problem prouzrokovan je zaklonjenošću čovjeka pri ulasku na scenu, što onemogućuje pravilnu inicijalizaciju praćenja. Metoda detekcije i praćenja dobro radi kod kratkih zaklanjanja, gdje se izgubljene detekcije kompenziraju pravilnim predviđanjem kretnji čovjeka. Test brzine pokazuje da je predviđanje brzine najlošije za predviđanje brzina manje od 0.1 m/s. U ovom slučaju sustav teško određuje kojom se brzinom čovjek giba, u određenim trenutcima smatra da čovjek miruje zbog malih pomaka. Odziv testiranja ostalih intervala brzine je nizak zbog nedostatka vrijednosti pri inicijalizaciji praćenja. Mogli bi se dodati nepouzdani rezultati praćenja koji bi povećali odziv, no smanjili točnost sustava. Testiranjem zaustavljanja robota dobiveni su dobri rezultati, gdje se greške ponajviše očituju u neodređenosti promatranih događaja, odnosno u kojem trenutku su uvjeti za zaustavljanje ostvareni. Na rezultate testiranja detektora i predviđanja brzine ponajviše utječe detektor sustava. Glavna prednost detektora je nisko opterećenje procesora računala, no za uzvrat nije dovoljno robustan za detekciju ljudi u posebnim slučajevima. Detektoru je potrebna vizualna predodžba cijelog tijela čovjeka kako bi ga uspješno detektirao, što nije pogodno za detekcije ljudi koji izlaze ili ulaze u kadar kamere. Detektor također nije robustan na velike promjene izgleda tijela čovjeka, kao što je slučaj kod nošenja predmeta, saginjanja i drugo.

# 7. ZAKLJUČAK

U ovom radu opisana je izrada sustava za detekciju i praćenje ljudi u svrhu sigurnog kretanja u okolini robotskog manipulatora. Dan je pregled radova u području te opisi alata i metoda koje se koriste u procesu implementacije. Zatim je opisan rad sustava i eksperimentalno je utvrđena njegova učinkovitost.

Objašnjeno je korištenje komercijalne Kinect kamere u svrhu prikupljanja RGB-D podataka za algoritme detekcije i praćenja. Opisano je korištenje javno dostupnog SPENCER razvojnog okvira, koji sadrži potrebne metode i alate za detekciju i praćenje ljudi, te promjene napravljene za potrebe implementacije sigurnosnog sustava. Informacije dobivene metodom detekcije i praćenja ljudi primjenjuju se u svrhu provjere uvjeta kojima se ograničava brzina kretanja robotskog manipulatora. Opisan je način upravljanja industrijskim robotom ABB IRB 2400L korištenjem komunikacijske veze s IRC5 kontrolerom preko ROS-a. Definirani uvjeti ograničenja kretanja mogu se primijeniti i na druge robotske manipulatore.

Izrađen je novi skup podataka snimljen Kinect kamerom, koji se koristi za testiranje komponenti sustava. Sigurnosni sustav prouzrokuje nisko opterećenje na računao na kojem se izvod, no jednostavnost sustava dovodi do smanjenja odziva i točnosti rezultata. Metoda za detekciju i praćenje daje dobre rezultate za normalne scene bez smetnji, proizvodi dobra predviđanja za položaj i brzinu čovjeka. Metoda nije dovoljno robusna na zaklanjanje tijela promatranih ljudi i na nagle promjene u promatranom prostoru. Sustav bi se mogao poboljšati korištenjem snažnijeg računala i robusnije metode za detekciju ljudi, sposobne detektirati ljude pri zaklanjanju tijela stranim predmetom i promjene položaja tijela. Mogao bi se implementirati program za upravljanje robotom u stvarnom vremenu, koji bi smanjio vrijeme u kojem robot miruje.

# LITERATURA


[1] V. Šimundić, D. Mihelčić, D. Svirac, P. Đurović, i R. Cupec, „Safety System for Industrial Robots Based on Human Detection Using an RGB-D Camera“, Mipro 2021, (prihvaćeno za objavu)

[2] F. Porikli i A. Yilmaz, „Object Detection and Tracking“, u *Video Analytics for Business Intelligence*, sv. 409, C. Shan, F. Porikli, T. Xiang, i S. Gong, Ur. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, str. 3–41. doi: 10.1007/978-3-642-28598-1_1.

[3] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, i T. Brox, „3D Human Pose Estimation in RGBD Images for Robotic Task Learning“, u *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, svi. 2018, str. 1986–1992. doi: 10.1109/ICRA.2018.8462833.

[4] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, i Y. Sheikh, „OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields“, *ArXiv181208008 Cs*, svi. 2019, Pristupljeno: svi. 17, 2021. [Na internetu]. Dostupno na: http://arxiv.org/abs/1812.08008

[5] S. Kallweit, R. Walenta, i M. Gottschalk, „ROS Based Safety Concept for Collaborative Robots in Industrial Applications“, u *Advances in Robot Design and Intelligent Control*, sv. 371, T. Borangiu, Ur. Cham: Springer International Publishing, 2016, str. 27–35. doi: 10.1007/978-3-319-21290-6_3.

[6] „MoveIt Motion Planning Framework“. https://moveit.ros.org/ (pristupljeno lip. 12, 2021).

[7] T. Linder, S. Breuers, B. Leibe, i K. O. Arras, „On multi-modal people tracking from mobile platforms in very crowded and dynamic environments“, u *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, svi. 2016, str. 5512–5519. doi: 10.1109/ICRA.2016.7487766.

[8] „GitHub - spencer-project/spencer_people_tracking: Multi-modal ROS-based people detection and tracking framework for mobile robots developed within the context of the EU FP7 project SPENCER.“ https://github.com/spencer-project/spencer_people_tracking (pristupljeno lip. 12, 2021).

[9] O. H. Jafari, D. Mitzel, i B. Leibe, „Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras“, u *2014 IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, China, svi. 2014, str. 5636–5643. doi: 10.1109/ICRA.2014.6907688.

[10] P. Sudowe i B. Leibe, „Efficient Use of Geometric Constraints for Sliding-Window Object Detection in Video“, u *Computer Vision Systems*, sv. 6962, J. L. Crowley, B. A. Draper, i M. Thonnat, Ur. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, str. 11–20. doi: 10.1007/978-3-642-23968-7_2.

[11] M. Munaro, F. Basso, i E. Menegatti, „Tracking people within groups with RGB-D data“, u *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura-Algarve, Portugal, lis. 2012, str. 2101–2107. doi: 10.1109/IROS.2012.6385772.

[12] A. Vedaldi i S. Soatto, „Quick Shift and Kernel Methods for Mode Seeking“, u *Computer Vision – ECCV 2008*, sv. 5305, D. Forsyth, P. Torr, i A. Zisserman, Ur. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, str. 705–718. doi: 10.1007/978-3-540-88693-8_52.

[13] T. Linder i F. Girrbach, „Towards a Robust People Tracking Framework for Service Robots in Crowded, Dynamic Environments“

[14] „ROS.org | Powering the world’s robots“. https://www.ros.org/ (pristupljeno lip. 14, 2021).

[15] „rviz - ROS Wiki“. http://wiki.ros.org/rviz (pristupljeno lip. 16, 2021).

[16] „rqt - ROS Wiki“. http://wiki.ros.org/rqt (pristupljeno lip. 16, 2021).

[17] „ROS-Industrial“. https://rosindustrial.org/ (pristupljeno lip. 17, 2021).
[18] „Kinect - Wikipedia“. https://en.wikipedia.org/wiki/Kinect (pristupljeno lip. 18, 2021).
[19] „IRB2400_R3_US 02_05.pdf“. Pristupljeno: lip. 18, 2021. [Na internetu]. Dostupno na: https://library.e.abb.com/public/7ee62fc704b04002c1257b130056d125/IRB2400_R3_US%2002_05.pdf
[20] „GitHub - OpenNI/OpenNI: OpenNI“. https://github.com/OpenNI/OpenNI (pristupljeno lip. 21, 2021).
[21] M. Meisel, „SUSTAV UPRAVLJANJA ROBOTSKIM MANIPULATOROM POMOĆU 3D KAMERE U ROS OKVIRU“, diplomski rad
[22] „GitHub - ros-industrial/abb_robot_driver: The new ROS driver for ABB robots“. https://github.com/ros-industrial/abb_robot_driver (pristupljeno lip. 30, 2021).

# SAŽETAK

U ovom radu opisana je izrada sustava za detekciju i praćenje ljudi u svrhu sigurnog kretanja u okolini robotskog manipulatora. Realizirani sustav zasnovan je na ROS-u. Za prikupljanje podataka iz okoline koristi se Kinect RGB-D senzor. Korištenjem SPENCER razvojnog okvira, implementirana je metoda detekcije i praćenja ljudi koja računa opisne informacije o promatranim ljudima na temelju podataka snimljenih Kinect kamerom. Definirana su pravila za ograničenje kretanja robotskog manipulatora koji obavlja radnje u prostoru dijeljenom s ljudima. Informacije dobivene metodom detekcije i praćenja ljudi koriste se za provjeru uvjeta koji određuju ograničenja postavljena na kretanje robota. Ostvareno je upravljanje ABB IRB 2400L robotskim manipulatorom povezivanjem osobnog računala s IRC5 kontrolerom robota. Rad izrađenog sustava testiran je novim skupom podataka snimljenim Kinect kamerom.

**Ključne riječi:** detekcija, praćenje, RGB-D senzor, robotski manipulator, ROS

## ABSTRACT

In this thesis, development of a human detection and tracking system for the purpose of safe movement in the robot environment is described. The developed system is based on ROS. Environment data is gathered using a Kinect RGB-D sensor. Using the SPENCER framework, a human detection and tracking method was implemented, which calculates descriptive information about observed humans using the data recorded with the Kinect camera. Rules for movement restriction of a robotic manipulator, operating in a workspace shared with humans, were defined. Conditions, that define the restrictions imposed to the manipulator movement, are checked using data from the human detection and tracking method. Control of the ABB IRB 2400L robotic manipulator was realized using a connection between a personal computer and an IRC5 robot controller. The developed system is tested using a new dataset recorded with a Kinect sensor.



**Key words:** detection, tracking, RGB-D sensor, robotic manipulator, ROS

# ŽIVOTOPIS

Dario Mihelčić rođen je 4. prosinca 1997. godine u Slavonskom Brodu. Pohađanje osnovne škole započinje u Starim Perkovcima, a kasnije završava osnovnu školu u Vrpolju. Upisao je srednju školu Gimnaziju „Matija Mesić“ u Slavonskom Brodu gdje je prošao sve četiri godine. 2016. godine upisuje preddiplomski sveučilišni studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Nakon završetka preddiplomskog studija, 2019. godine na istom fakultetu upisuje diplomski studij računarstva, izborni blok Robotika i umjetna inteligencija. Za rad na HDR projektu 2021. godine osvaja Rektorovu nagradu.

_____

Potpis autora