SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Borna Zbodulja, Jakov Vulama, Petar Ljubotina

Humanoidni robot Pepper kao menadžer dobrodošlice na FER-u

Zagreb, kolovoz 2020.

Ovaj rad izrađen je u Laboratoriju za robotiku i inteligentne sustave upravljanja (LARICS) na Zavodu za automatiku i računalno inženjerstvo na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu pod vodstvom prof. dr. sc. Zdenka Kovačića i predan je na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2019./2020.

SADRŽAJ

POPIS SLIKA

1. Uvod

U današnje vrijeme ljudi sve više razmišljaju kako iskoristiti robote koje imaju na raspolaganju u svakodnevnom životu. U 2005. godini 90% robota korišteno je u autoindustriji kao manipulatori za slaganje [1]. Danas roboti dostižu sve veću popularnost i neki od ljudima najzanimljivijih su humanoidni roboti, autonomne letjelice i medicinski roboti [2]. Roboti u potpunosti zamjenjuju ljude u poslovima koji su opasni ili se obavljaju u okruženjima nepristupačnim za ljude. U knjizi [3] opisana je uloga robota za razminiranje minskih polja. Njegovim korištenjem rizik za ljudske živote sveden je na minimum. U okruženjima u kojima ljudi svakodnevno funkcioniraju, roboti se koriste u suživotu s njima. Oni obavljaju jednostavnije poslove te tako olakšavaju ljudima i daju im mogućnost da oni rješavaju probleme za koje roboti još uvijek nisu sposobni.

Potreba za korištenjem robota u takvim interakcijama najbolje je vidljiva u novonastaloj globalnoj krizi uslijed pandemije COVID-19 virusa. U svijetu su pojmovi, kao što su nošenje zaštitnih maski i socijalna distanca, postali svakodnevica. Dolaskom osobe u bliski kontakt sa zaraženim može imati velike posljedice. U Brislu, glavnom gradu Belgije, humanoidni robot Pepper koristi se u dočekivanju pacijenata u bolnici [4]. S njima obavlja početne konzultacije te ih upozorava na nošenje zaštitnih maski za lice i držanje socijalne distance. To je monoton posao koji čovjeka može umarati, ali ne i robota. Tako se oslobađaju ljudski resursi za bolju brigu o pacijentima i smanjuje mogućnost zaraze osoblja. Također, na FER-u je razvijen softver kojim humanoidni robot NAO pomaže ljudima tijekom rehabilitacije nakon prijeloma ruku [5]. Robot izvodi koreografije osamnaest različitih vježbi spremljenih u njegovoj memoriji koje prema programu rehabilitacije pacijent treba napraviti. Robot izvodi vježbe paralelno s čovjekom i broji koliko je bilo ponavljanja.

Zahvaljujući tome što LARICS laboratorij, u kojem je ovaj rad napravljen, raspolaže s dva humanoidna robota Pepper dizajnirana za kompleksnu interakciju s ljudima, rodila se ideja da se pokuša izraditi programsko rješenje koje će omogućiti da Pepper postane menadžer dobrodošlice (engl. *welcome manager*) na glavnom ulazu Fakulteta

elektrotehnike i računarstva (FER-a) te po potrebi bude i vodič za posjetitelje, studente i djelatnike jer se može kretati po ravnom podu zahvaljujući mobilnoj bazi na kotačima.

Detaljniji scenarij zamišljen za vođenje po Fakultetu elektrotehnike i računarstva opisan je u drugom poglavlju. Treće poglavlje opisuje korištene materijale i metode, podijeljene na hardverski, softverski dio i metodu stabla ponašanja pomoću koje je cijelo programsko rješenje izvedeno. Hardverski dio detaljno opisuje sve komponente korištenog Pepper robota. U softverskom djelu opisana su razvojna okruženja ROS (engl. *Robot Operating System*) i NAOqi neophodna za upravljanje robotom kao i sve metode potrebne za cjelokupno programsko rješenje problema. Mapiranjem prostora dobivaju se različite 2D i 3D mape potrebne za navigaciju u prostoru. Audio vizualna komunikacija sa sugovornikom odvija se usmeno ili preko tableta. Unaprijed se određuje što će se prikazivati na tabletu i što robot mora čuti ili izgovoriti, ovisno o određenoj situaciji. U radu je opisana i aplikacija za tablet koja je izrađena za komunikaciju s posjetiteljem. U četvrtom poglavlju prikazani su rezultati testiranja u simulatoru, laboratoriju i u stvarnom okruženju na FER-u.

2. Pepper kao menadžer dobrodošlice

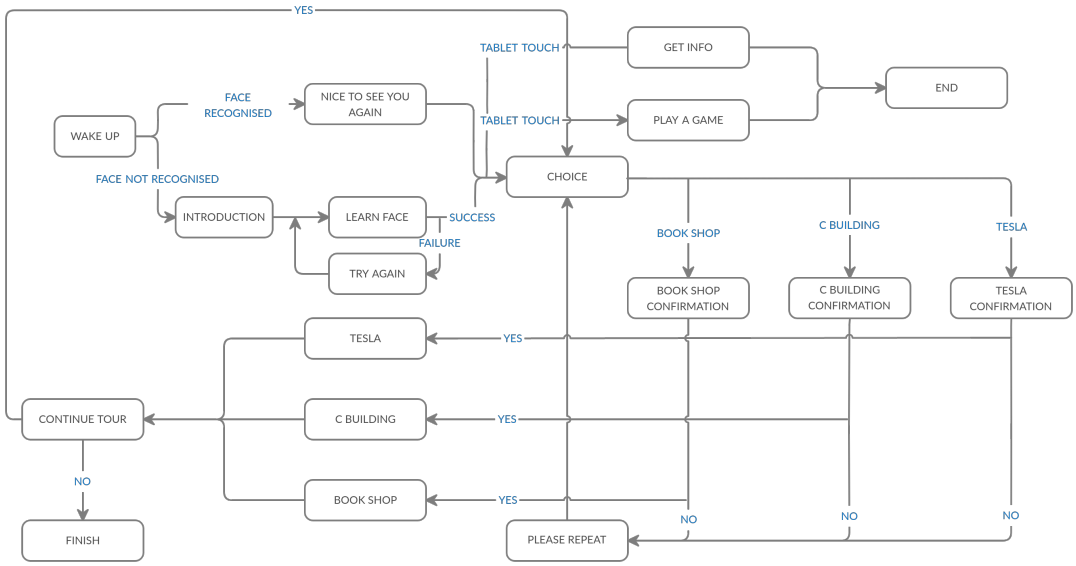
Kada osoba prvi puta dolazi na nepoznato mjesto, mora upitati nekoga da ga uputi na željenu lokaciju. Problem nastaje ako sugovornik ne zna, ili nije siguran gdje se ta lokacija nalazi ili kako najbrže doći do nje. Takav problem rješava robot koji u svojoj memoriji ima spremljene sve potrebne lokacije i najbrže putove do njih. Ideja ovog rada je da Pepper bude robot koji će imati ulogu menadžera dobrodošlice koji bi u prostorima fakulteta dočekivao goste, studente i djelatnike te s njima ostvarivao komunikaciju. Kako izgleda zamišljeni tijek događaja, prikazano je na slici 2.1.

2.1. Dočekivanje gostiju

Pepper bi se nalazio na glavnom ulazu FER-a (crvena točka na slici 2.2) te bi prilazio ljudima i dočekivao ih. Kako bi komunikacija s ljudima bila što prirodnija, implementirani su već postojeći algoritmi za prepoznavanje i učenje lica čovjeka. Kada Pepper priđe osobi koju je već prije upoznao, pozdravi je imenom i upita kako se osjeća. Kada uoči nepoznatu osobu na ulazu, priđe joj te se krene predstavljati. Potom ju gleda u lice i pokušava ga naučiti i zapamtiti. Pokušaj traje šest sekundi te se u slučaju neuspjeha ponavlja. Kada je lice čovjeka naučeno ili prepoznato, na tabletu se pokreće stranica za odabir nastavka komunikacije s Pepperom.

2.2. Odabir vrste interakcije s Pepperom

Interakcija Peppera i sugovornika nije ista ako je sugovornik odrasla osoba ili dijete te se zato pokreće stranica na tabletu koja sadrži tri različita gumba za odabir vrste željene interakcije. Odrasla osoba kojoj Pepper pristupi, a želi otići do pojedine lokacije na FER-u, odabrat će gumb da mu Pepper omogući odabir lokacije na koju će ga odvesti. Dijete koje dođe na fakultet s roditeljima želi se zabavljati te pritiskom na drugi gumb Pepper pokreće igricu "*križić kružić*" (engl. *tic tac toe*) koju dijete može



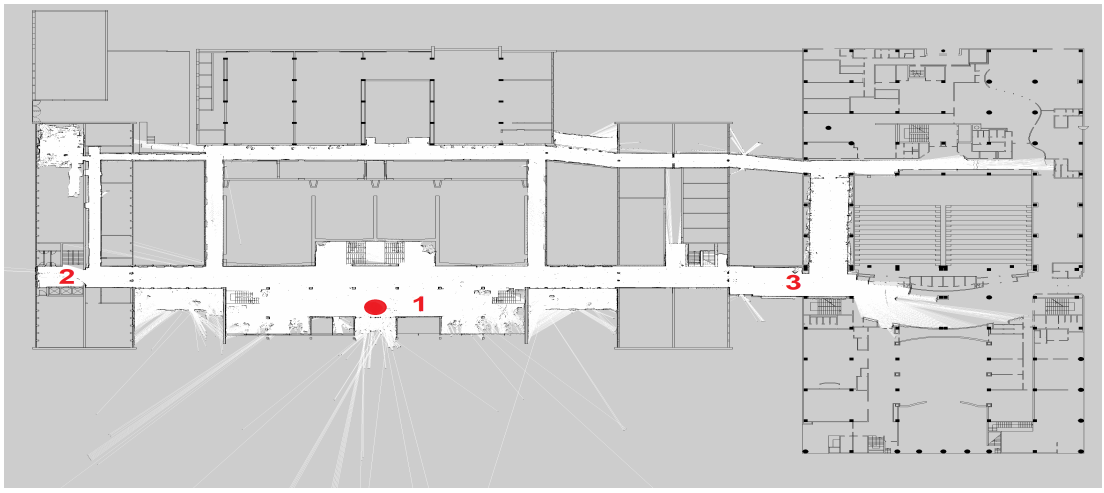
Slika 2.1: Tijek događaja Peppera kao menadžera dobrodošlice

igrati protiv robota ili protiv druge osobe. Ako je Pepperov sugovornik osoba koja poznaje lokacije na fakultetu te pomoću njega želi saznati pojedine informacije, pritiskom na treći gumb pokreće se web stranica LARICS laboratorija koja sadrži sve potrebne informacije o njemu. Prema potrebi i ovisno o scenariju u kojem Pepper funkcionira kao menadžer dobrodošlice, na trećem gumbu može se pokretati bilo koja javna web stranica s informacijama. Čovjek može igrati igricu ili pregledavati web stranicu koliko god puta želi te, kada završi, interakcija između robota i njega prestaje te se Pepper vraća u svoje početno stanje u kojem dočekuje druge posjetitelje.

2.3. Vođenje po fakultetu

Kada čovjek odabere vođenje po fakultetu, Pepper mu ponudi tri različite lokacije. Skriptarnica (broj 1, slika 2.2) odabrana je kao najbliža lokacija do koje robot treba doći jer se nalazi vrlo blizu glavnog ulaza. Liftovi C zgrade (broj 2, slika 2.2) odabrani su kao srednje udaljena lokacija. U C zgradi nalazi se većina zavoda s mnogo različitih djelatnika FER-a te Pepper kao vodič do liftova može biti koristan posjetiteljima koji tamo prvi puta dolaze na sastanke. Kao najudaljenija lokacija odabrana je bista Nikole Tesle (broj 3, slika 2.2). Nikola Tesla veliki je um dvadesetog stoljeća koji je u svijet elektrotehnike unio mnoge inovacije i njegov spomenik važna je lokacija na fakultetu. Čovjek odabire na koju lokaciju želi ići usmeno ili pritiskom na gumb na tabletu. Kada je lokacija odabrana, Pepper vrši provjeru je li dobro razumio kamo

treba ići. Provjera se vrši zato što zbog buke Pepper može krivo razumjeti koja mu je riječ izrečena te tako krivo odvesti sugovornika. Potvrdu je li čuo dobru riječ Pepper očekuje usmeno slušajući riječi da (engl. *yes*) ili ne (engl. *no*). Pogreška se može dogoditi i prilikom prepoznavanja tih riječi, ali u poglavlju 3.2.5 je detaljno objašnjeno zašto je ovakav način komunikacije valjan. Ako je odgovor negativan, Pepper miruje na mjestu te ponovno pokušava čuti novu riječ. Kada je odgovor pozitivan, robot se počinje kretati prema odabranoj lokaciji. Prilikom kretnje do lokacije izgovara tekst o njoj. Dolaskom do skriptarnice govori što se sve u njoj može kupiti, a do C zgrade izgovara informacije o zavodima i laboratorijima koji se u njoj nalaze. Putem do biste Nikole Tesle izgovara informacije o njegovom životu te popularnosti koju ima i danas. Najveći tvorac električnih automobila na svijetu nosi njegovo ime te je Tesla poznat širom svijeta. Po dolasku na lokaciju robot staje te nakon nekoliko sekundi upita sugovornika želi li otići do neke druge moguće lokacije. Ako je odgovor pozitivan, robot miruje i ponovno očekuje da čovjek kaže kamo želi ići usmeno ili pritiskom gumba na tabletu. Ako je odgovor negativan, Pepper se zahvaljuje sugovorniku na interakciji te se vraća na svoju početnu poziciju. Pepperov ciklus interakcije s čovjekom završava kada se uspješno vrati na početnu poziciju te je nakon toga spreman za dočekivanje novog sugovornika.

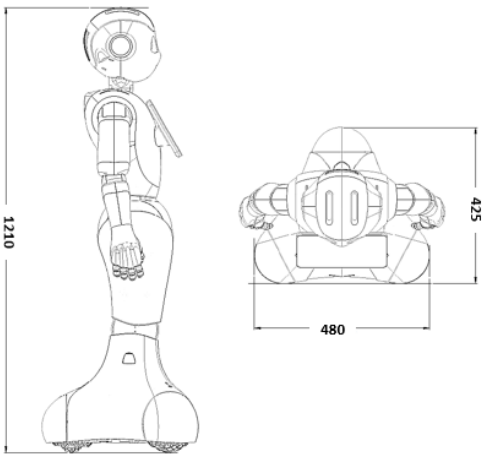


Slika 2.2: Mapa fakulteta s označenim lokacijama

3. Metode i materijali

3.1. Humanoidni robot Pepper

Pepper robot je humanoidni robot kojeg je proizvela francuska tvrtka SoftBank Robotics, a predstavljen je 2014. godine [6]. Projektiran je za što bolju komunikaciju s ljudima te ima sposobnost prepoznavanja govora, ljudskih emocija i ugrađene standardne reakcije na detektiranu emociju. Visok je 1.2 metara i težak 28 kilograma, a većina mase smještena je na donjem dijelu robota kako bi se omogućilo fluidnije izvođenje mimike pokretima torza i ruku. Na tržištu postoje tri verzije robota Pepper, a za izradu ovog rada korištena je verzija robota 1.8a dostupna na FER-u. Kretanje robota izvodi se korištenjem triju višesmjernih (engl. *omnidirectional*) kotača i broj upravljivih stupnjeva slobode prilikom kretanja robota jednak je ukupnom broju stupnjeva slobode robota što Peppera svrstava u skupinu holonomskih robota.



Slika 3.1: Prikaz i dimenzije robota Pepper

3.1.1. Senzori

Postoji više mogućih podjela senzora robota s obzirom na njihovu funkciju, lokaciju na robotu, vidljivost itd. Za potrebe ovog rada senzori se mogu podijeliti na one potrebne za navigaciju i mapiranje prostora i na one potrebne za audio-vizualnu interakciju s ljudima. Za navigaciju i mapiranje koriste se laseri, sonari i inercijalna mjerna jedinica (engl. *IMU - inertial measurement unit*) smješteni na bazi robota i dubinska kamera smještena na glavi. Za audio-vizualnu komunikaciju s ljudima koriste se zvučnici, mikrofoni i 2D kamere smještene na glavi.

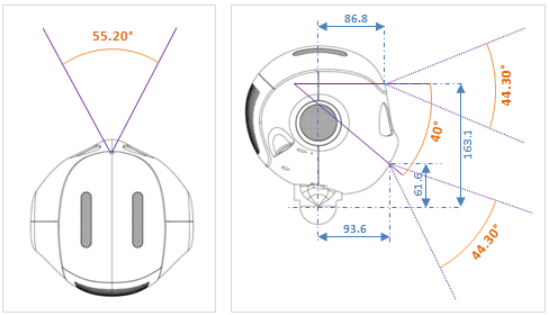
Laseri

Pepper dolazi opremljen sa šest laserskih senzora. Oni se mogu podijeliti u dvije skupine: tri laserska senzora usmjerena su na tlo ispred robota, a preostala tri su usmjerena na okruženje oko samog robota (prednji, desni i lijevi laser). Lokacija i vidno polje skupine laserskih senzora usmjerenih na okruženje oko robota prikazani su na slici 3.2. Frekvencija ažuriranja svakog lasera iznosi 6.25 Hz, a valna duljina 808 nanometara. Maksimalna udaljenost na kojoj laseri usmjereni na okruženje oko robota mogu detektirati prepreku je 10 metara.

Slika 3.2: Lokacija i vidno polje lasera usmjerenih na okruženje oko robota (prednji, lijevi i desni laser numerirani redom brojevima 1-3)

2D kamere

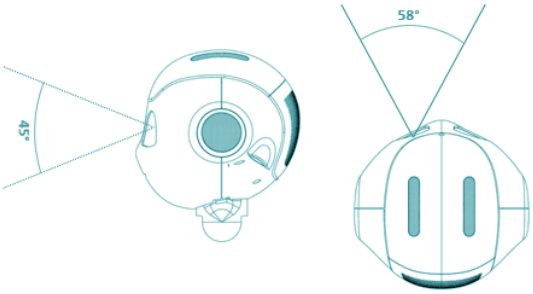
Pepper ima dvije jednake 2D kamere od kojih se jedna nalazi na njegovom čelu, a druga u ustima. Iz svake kamere može se dobivati slika rezolucije 2560 x 1920 piksela uz frekvenciju osvježavanja od 1 slike po sekundi ili 640 x 480 piksela uz frekvenciju osvježavanja od 30 slika po sekundi. Vidna polja kamera prikazana su na slici 3.3. Budući da se vertikalna vidna polja kamera ne preklapaju, kamere nije moguće koristiti kako bi se dobila 3D (stereo) slika.



Slika 3.3: Horizontalno i vertikalno vidno polje 2D kamera

Dubinska kamera

U lijevom oku robota nalazi se ASUS Xtion 3D kamera koja daje sliku rezolucije 320 x 240 piksela uz frekvenciju osvježavanja od 20 slika po sekundi. Raspon fokusa dubinske kamere je između 40 centimetara i 8 metara, a to ujedno predstavlja i raspon udaljenosti unutar kojih dubinska kamera može detektirati prepreke. Vidno polje dubinske kamere prikazano je na slici 3.4.



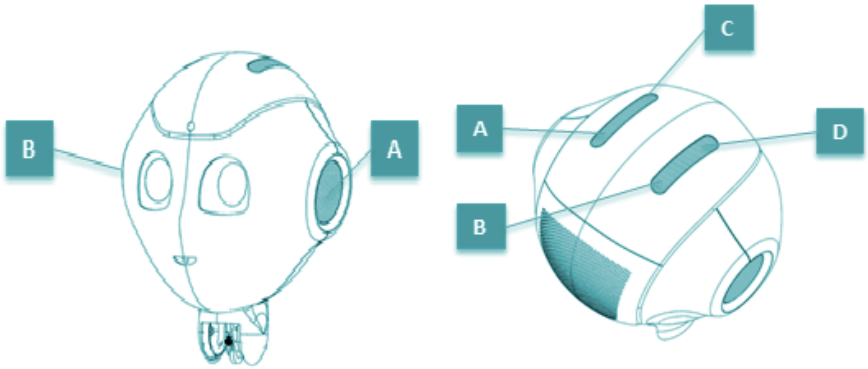
Slika 3.4: Vertikalno i horizontalno vidno polje dubinske kamere

Zvučnici

Dva zvučnika nalaze se u ušima robota, prikazani na slici 3.5a. Osjetljivost zvučnika je 78 dB 1W/1m na frekvenciji 1 kHz. Frekvencijski raspon zvučnika je od 220 Hz do 20 kHz.

Mikrofoni

Pepper posjeduje četiri mikrofona koji su smješteni na glavi robota, a njihova lokacija prikazana je na slici 3.5b. Frekvencijski opseg mikrofona je od 100 Hz do 10 kHz, pri čemu je pri takvom opsegu 10 dB referentno kao 1kHz. Osjetljivost mikrofona na frekvenciji od 1kHz je 300 mv/Pa +/- 3 dB.



(a) Lokacija zvučnika **(b)** Lokacija mikrofona

Slika 3.5: Prikaz lokacija zvučnika i mikrofona na glavi robota

Tablica 3.1: Tehničke specifikacije tableta

Model procesora	ATOM E3845
Radna memorija	4 GB DDR3
Interna memorija	32 GB eMMC (od kojih 24 GB dostupno za korištenje)
Veličina ekrana	10.1 inča
Rezolucija ekrana	1280 x 800 piksela

3.1.2. Tablet

Na Pepperovim prsima nalazi se tablet koji omogućava još jedan način komunikacije između Peppera i čovjeka. Model tableta je LG CNS Tablet, a njegove tehničke specifikacije prikazane su u tablici 3.1. Tablet koristi Android operacijski sustav što omogućava izradu jednostavnih ili kompleksnih korisničkih sučelja, čime se omogućava komunikacija robota s ljudima koji imaju poteškoća sa sluhom. Također, to omogućava dvostruku komunikaciju sa sugovornikom. Pepper na tabletu može ispisati što je rekao, u slučaju da ga sugovornik nije razumio. Prikaz robotovog tableta s primjerom korisničkog sučelja prikazan je na slici 3.6.

3.2. Programska podrška

Programska podrška bitna je sastavnica svakog robotskog sustava. Potrebno je ukomponirati različite programske jezike i programske alate u jednu funkcionalnu cjelinu. U ovom radu korišteni su ROS i NAOqi razvojna okruženja, programski jezici Python i C++, aplikacije za upravljanje robotom Choregraphe i RViz te Gazebo simulator za simulaciju virtualnog robota.

Slika 3.6: Prikaz robotovog tableta s korisničkim sučeljem za odabir lokacije

3.2.1. ROS

ROS (engl. *Robot Operating System*) je razvojna okolina koja obuhvaća skup biblioteka, alata i konvencija s ciljem olakšavanja razvoja i rješavanja kompleksnih problema koji se postavljaju robotu [7]. Postoji velika baza implementacija otvorenog koda (engl. *open source*) za česte funkcionalnosti u robotici, implementiranih za korištenje u ROS-u, od kojih će neke biti korištene u ovom radu. Programski kod u ROS-u organiziran je u čvorove (engl. *nodes*), pri čemu svaki čvor obavlja određenu funkciju, npr. primanje i obrada podataka iz senzora ili upravljanje zglobom robotske ruke. Skup čvorova koji zajedno obavljaju složeniju funkciju grupiraju se u pakete (engl. *packages*). Čvorovi se mogu pisati u Python i C++ programskim jezicima i međusobno komuniciranju pomoću izdavač - pretplatnik (engl. *publisher - subscriber*) koncepta, putem komunikacijskih kanala koji se u terminologiji ROS-a nazivaju teme (engl. *topics*). Informacije se između čvorova razmjenjuju u obliku poruka (engl.

messages). Takav način komunikacije omogućava da čvorovi budu pisani u različitim programskim jezicima jer nisu međusobno direktno povezani. Također, omogućava da se svaki čvor može zamijeniti drugim čvorom iste funkcionalnosti bez potrebe za dorađivanjem ostalih čvorova. Do danas su izašle brojne verzije ROS-a, a u ovom radu korištena je verzija *melodic*.

3.2.2. NAOqi

NAOqi OS operacijski je sustav koji se nalazi na Pepper robotu, a u ovom radu korištena je verzija 2.5. Za njegovo programiranje koristi se NAOqi razvojno okruženje (engl. *framework*) koji je proizvod tvrtke SoftBank Robotics, kao i roboti Pepper i NAO. Sadrži u sebi API (engl. *application programming interface*) koji sadrži mnoge različite module napravljene za programiranje robota NAO i Pepper. NAOqi razvojno okruženje nalazi se u kompletu za razvoj softvera PythonSDK (engl. *software development kit*) koji se instalira na lokalnom računalu i tako pokreće iz programskog jezika Python. Za izradu ovog rada korištena je Python verzija 2.7 i PythonSDK verzija 2.5.5.5, a sve se pokreće na lokalnom računalu s operacijskim sustavom *Ubuntu* 16.04. Pepper i lokalno računalo međusobno se povezuju preko wi-fi veze preko Pepperove IP (engl. *internet protocol*) adrese. IP adresa specifična je za svaki robot posebno. Prilikom izrade ovog rada korišteni su mnogi moduli za upravljanje robotom, a oni su: *ALSpeechRecognition*, *ALMemory*, *ALTextToSpeech*, *ALTabletService*, *ALFaceDetection*, *ALMotion* i *ALBehaviorManager*. Službena dokumentacija svih modula nalazi se u [8]. Pomoću svakog modula stvara se posebna *proxy* varijabla koja ima sve funkcije kao i modul, a kako izgleda njezino stvaranje prikazano je u kodu 3.1.

```
from naoqi import ALProxy

nameProxy = ALProxy("ALModuleName", variables.IP,
↪  variables.PORT)
```

Izvorni kod 3.1: Stvaranje proxy varijable

Koristi se funkcija ALProxy koja ima tri argumenta. Prvi je ime modula, drugi IP adresa robota i treći PORT robota koji je za Peppera najčešće 9559. *AlTextToSpeech* modul koristi se za robotsko izgovaranje teksta. *ALMotion* modul koristi se za upravljanje robotskim pokretima. U kodu 3.2 prikazani su primjeri korištenja navedenih modula.

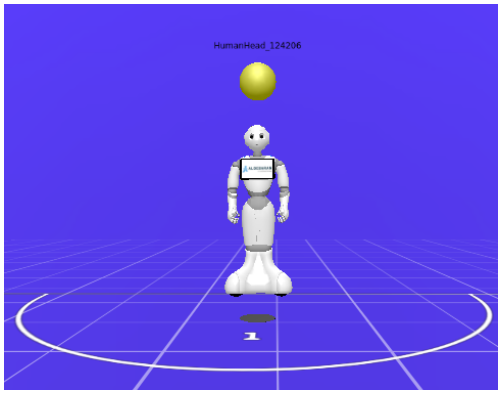
```
from naoqi import ALProxy

wakeProxy = ALProxy("ALMotion", variables.IP, variables.PORT)
sayProxy = ALProxy("ALTextToSpeech", variables.IP,
↪  variables.PORT)

wakeProxy.wakeUp() #buđenje robota
sayProxy.say("text") #izgovaranje nekog teksta u navodnicima
```

Izvorni kod 3.2: Izvođenje pojedinih funkcija

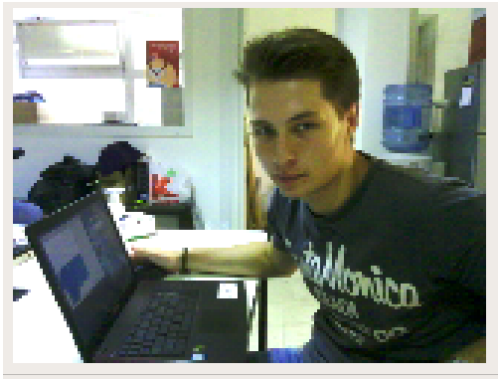
ALMemory modul služi za pristupanje memorijskim lokacijama na robotu. U memoriji se nalaze podaci o aktuatorima, senzorima i vlastiti događaji (engl. *events*) potrebni za funkcioniranje robota. Kada se pozove specifični događaj, poduzimaju se predefinirane akcije za svaki od njih posebno. Čitanje i pisanje nije karakteristično samo za *ALMemory* modul već svi ostali moduli mogu također pristupati memoriji. *ALSpeechRecognition* je modul kojim Pepper prepoznaje i sluša riječi koje čovjek izgovara. Kako bi mogao prepoznati riječi, potrebno ih je spremiti u vokabular pomoću kojeg Pepper prati koje riječi može čuti i prepoznati. *ALTabletService* je modul koji služi za upravljanje funkcijama na tabletu. Pomoću njega se kontrolira je li i gdje je tablet pritisnut, pokreću se web stranice i upravlja svim njegovim funkcijama. *ALBehaviorManager* je modul pomoću kojeg se iz memorije pokreću različita ponašanja, a detaljnije je opisan u poglavlju 3.2.6. *ALFaceDetection* je modul koji služi da Pepper prepoznaje lica čovjeka. Koristi funkcije *learnFace("name")* i *getLearnedFacesList()*. Funkcija *learnFace("name")* služi za učenje lica čovjeka, kojeg ne poznaje, pod imenom upisanom u zagradama. Funkcija *getLearnedFacesList()* dohvaća listu naučenih lica. Svako lice je spremljeno pod različitim imenom te tako Pepper jednoznačno može prepoznati osobu koju vidi. Kako Pepper vidi osobu sa svojim 2D kamerama te kako izgleda percepcija čovjeka u simulatoru prikazano je na slikama 3.7 i 3.8.

3.2.3. Mapiranje prostora

Mnoge robotske aplikacije poput leta drona ili kretanja robota opremljenim manipulatorom, zahtijevaju korištenje 3D modela radnog okruženja [9]. U ovom radu ispitana je mogućnost dobivanja 3D mapa i modela prostora pomoću Pepperovih senzora. Paket u ROS-u koji je korišten za dobivanje 3D mapa je *octomap* koji koristi *octree* strukturu podataka za reprezentaciju trodimenzionalnog prostora. Svaki čvor (engl. *node*) u *octree* strukturi podataka predstavlja prostor sadržan u vokselu i mini-



Slika 3.7: Prikaz robota - Choregraphe



Slika 3.8: Izlaz Pepperovih 2D kamera

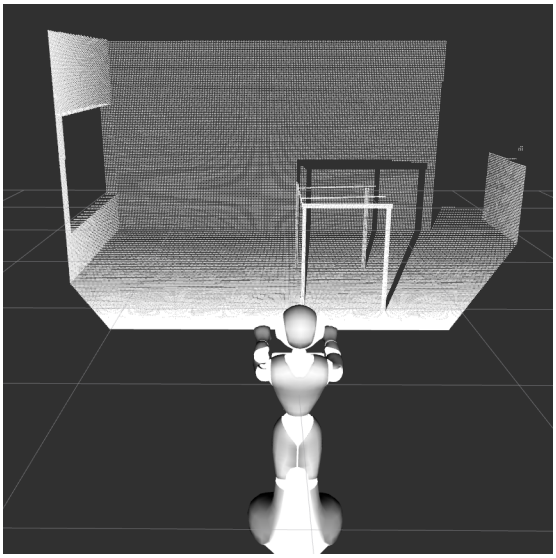
malna veličina voksela definira rezoluciju *octree-a*. Za dobivanje 3D mapa robotom Pepper koristi se njegova dubinska kamera. Mapa se izrađuje na temelju oblaka točaka (engl. *pointcloud*) iz robotove dubinske kamere (slika 3.9), a paket u ROS-u koji služi za dobivanje oblaka točaka iz slike dubinske kamere je *depth_image_proc* [10]. Prije mapiranja, može se podesiti minimalna veličina voksela, kao i uključiti filtriranje ravnine podloge (ako je ova opcija uključena, podloga se izostavlja iz mape) i podešavati parametre tog filtra. Tijekom procesa mapiranja kretanjem robota upravlja se pomoću tipkovnice na računalu, korištenjem paketa *teleop_twist_keyboard* [11]. Mapu je moguće vizualizirati korištenjem *octomap_rviz* dodatka (engl. *plugin*) u RVizu. Dobivena mapa sprema se pomoću naredbe

```
$ rosrun octomap_server octomap_saver -f mapname.ot
```

i može se naknadno uređivati (primjerice ukloniti strop iz mape prostora) korištenjem alata *octovis* koji je sadržan u *octomap* paketu. Dobivanje 2D mape senzorima robota Pepper analizirano je u radu [12]. Na temelju tih zaključaka korištena je 2D mapa FER-a prikazana na slici 2.2 dobivena s mobilnim robotom Pioneer 3-DX.

3.2.4. Navigacija u prostoru

Navigacija robota u ROS-u izvodi se implementacijom 2D navigacijskog stoga (engl. *stack*). Paket u ROS-u koji se koristi za navigaciju i upravljanje kretanjem robota je *move_base* paket [13]. Dvije osnovne komponente navigacijskog stoga u ROS-u čine lokalizacijski algoritam i planeri. Kao izvor informacija o okruženju oko robota korištena je slika iz dubinske kamere. Kako bi se ta slika mogla koristiti za navigaciju i lokalizaciju, potrebno je tu sliku pretvoriti u oblak točaka i potom u senzorsko očitanje (engl. *scan*). To se u ROS-u izvodi korištenjem paketa *depth_image_proc*

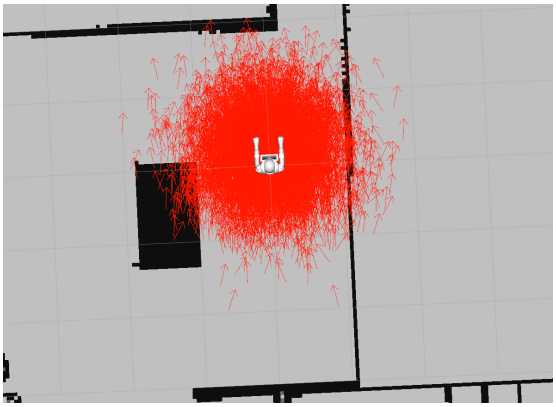


Slika 3.9: Primjer oblaka točaka iz robotove dubinske kamere

koji sliku pretvara u oblak točaka, i paketa *pointcloud_to_laserscan* koji potom oblak točaka projicira na ravninu podloge i pretvara u senzorsko očitanje [10][14]. Parametri navigacijskog stoga mogu se prilagođavati u odgovarajućim *yaml* datotekama, a u ovom radu korišten je najbolji set parametara dobiven u radu [15].

Lokalizacijski algoritam nastoji precizno odrediti trenutnu poziciju robota u mapi prostora. U ROS-u postoji nekoliko različitih lokalizacijskih paketa, a onaj korišten u ovom radu je amcl (*AMCL - Adaptive Monte Carlo Localization*) [16]. On učitava 2D mapu prostora podijeljenu na ćelije (engl. *grid based map*) i pokušava senzorska očitanja koja primi iz dubinske kamere uskladiti s dobivenom mapom te na temelju toga procijeniti postoji li pomak u odnosu na procjenu pozicije na temelju robotove odometrije (odometrija na robotu Pepper temelji se na analizi podataka iz enkodera na robotovim kotačima). Ukoliko algoritam procjeni da je došlo do pomaka, on se kompenzira tako da algoritam objavljuje transformaciju između koordinatnih sustava odometrije i baze robota (engl. *base frame*). Korišteni algoritam verzija je čestičnog filtra (engl. *particle filter*), pri čemu čestice (engl. *particles*) predstavljaju skup mogućih pozicija robota i one se mogu vizualizirati u obliku oblaka čestica (engl. *particlecloud*) kao što je prikazano na slici 3.10. Poziciju one čestice, za koju algoritam smatra da je najvjerojatnija, odnosno ona za koju je vjerojatnost da se robot nalazi na toj poziciji najveća, objavljuje se na temi *amcl_pose*.

Planeri u navigaciji služe za pronalazak optimalne trajektorije do zadanog cilja. U ROS-u postoje dvije vrste planera: globalni i lokalni planer. Globalni planer za pronalazak optimalne trajektorije do cilja koristi isključivo početnu mapu prostora, što



Slika 3.10: Vizualizacija oblaka čestica u RVizu

znači da može izbjegavati samo statičke prepreke koje se već nalaze u mapi. On na temelju početne mape prostora stvara globalnu cjenovnu mapu (engl. *costmap*) gdje se svakoj ćeliji mape pridružuje cijena prolaska kroz tu ćeliju i potom koristi neki od algoritama pretraživanja prostora stanja (npr. Dijsktrin ili A* algoritam) za pronalazak optimalnog puta do cilja. U ROS-u postoji nekoliko različitih globalnih planera, a u ovom radu korišten je Navfn-ROS globalni planer.

Lokalni planer, za razliku od globalnog planera, za pronalazak optimalnog puta do cilja koristi lokalnu cjenovnu mapu, koja se generira u svakoj iteraciji planera, na temelju senzorskih očitanja dobivenih iz dubinske kamere. To mu daje sposobnost izbjegavanja dinamičkih prepreka i on korigira trajektoriju globalnog planera i daje robotu konačne naredbe za brzinu. U ROS-u postoje dva osnovna lokalna planera: TrajectoryPlannerROS i DWA (Dynamic Window Approach) planer, a u ovom radu je korišten DWA lokalni planer.

Zadavanje navigacijskih ciljeva može se izvršiti pomoću funkcije *2DNavGoal* u RVizu ili izradom akcijskog servera unutar ROS čvora i pozivom metode *sendGoal*. Primjer takvog ROS čvora objašnjen je na [17].

3.2.5. Audio vizualna komunikacija

Audio vizualna komunikacija koristi se za komunikaciju čovjeka i robota. Kao što je već rečeno, za programiranje audio komunikacije na robotu koriste se moduli *ALTextTospeech* i *ALSpeechRecognition*. Za određeni scenarij unaprijed se napiše tekst kojeg robot mora izgovarati u pojedinim slučajevima. U ovom radu robot najviše teksta izgovara prilikom kretnje do određene lokacije na engleskom jeziku. U kodu 3.3 prikazano je kako izgleda kada Pepper izgovara nekoliko rečenica o Nikoli Tesli. Pep-

per ne izgovara cijeli tekst odjednom, nego svaku rečenicu posebno s pauzama između njih 0.3 ili 0.5 sekundi kako bi komunikacija bila prirodnija.

```
sayProxy = ALProxy("ALTextToSpeech", variables.IP,
↪  variables.PORT)

sayProxy.say("I believe you have already heard about Nikola
↪  Tesla.")
time.sleep(0.5)
sayProxy.say("He is one of the greatest minds of the 20th
↪  century. ")
time.sleep(0.5)
sayProxy.say("He was born in Croatia in 1856, but most of
↪  his life he lived in the USA. ")
time.sleep(0.3)
sayProxy.say("In his early days he was an assistant to
↪  Thomas Alva Edison, but they got into a fight.")
```

Izvorni kod 3.3: Izgovaranje teksta o Nikoli Tesli

Pepper ima mogućnost slušanja i prepoznavanja riječi, ali one moraju biti prije spremljene u listi koja se naziva vokabular (engl. *vocabulary*). Kada se napravi *proxy* varijabla modula *ALSpeechRecognition*, potrebno je varijablu pretplatiti (engl. *subscribe*) na modul Test_ASR. Na kraju, kada je riječ prepoznata, *proxy* varijablu obavezno je odjaviti (engl. *unsubscribe*) s modula Test_ASR. U protivnom, ako se varijabla ne odjavi, pokretanje procesa prepoznavanja riječi na robotu više neće biti moguće. U radu [18] analizirano je s kolikom pouzdanošću Pepper prepoznaje riječi te ona iznosi oko 60%. Prilikom izgovaranja riječi treba uzeti u obzir navedenu pouzdanost jer se lako može dogoditi da Pepper čuje krivu riječ. Iz tog razloga postoji dodatna provjera koju je riječ Pepper čuo. Nakon njegovog upita je li čuo ispravnu riječ, čovjek odgovara s jednostavnim odgovorima da ili ne. Kod prepoznavanja tih riječi također može doći do pogreške, ali prilikom testiranja pokazano je kako njihova pouzdanost iznosi oko 70% uz stopostotno ispravno raspoznavanje te se tako smatra da bi Pepper uvijek trebao razumjeti riječi da ili ne. Mogućnost poboljšanja pouzdanosti prepoznavanja riječi je korištenje Google Cloud usluge pomoću koje pouzdanost prelazi i 90% te nije potreban vokabular nego se odmah prepoznaje koja je riječ izgovorena.

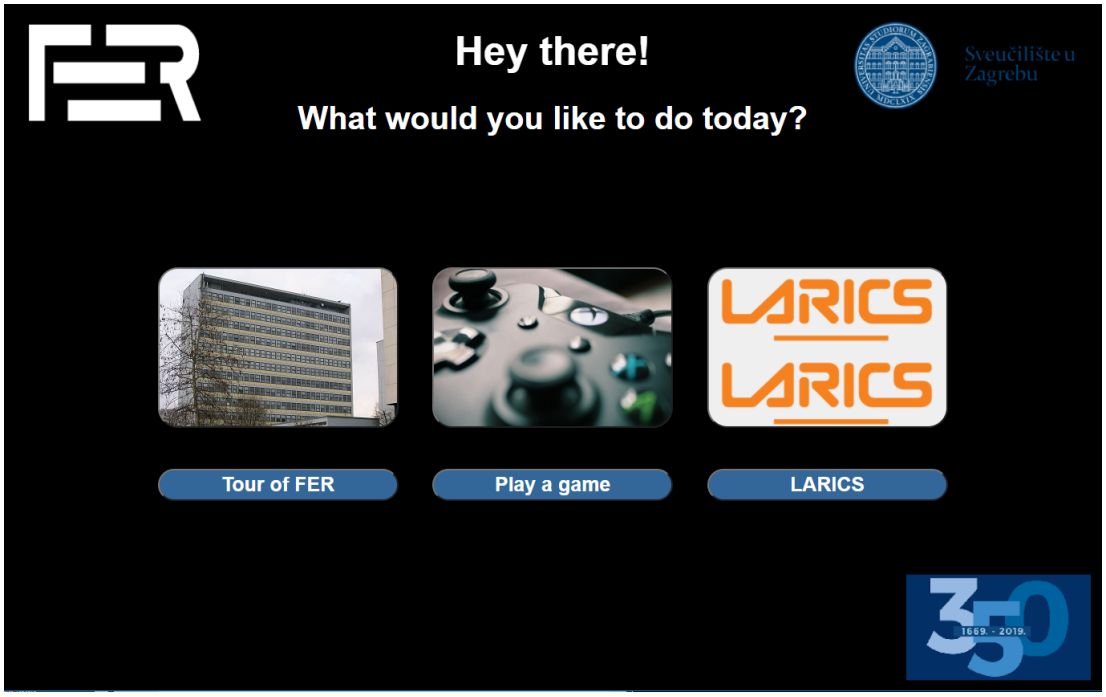
Kako bi sugovornik što više imao osjećaj da razgovara s čovjekom, korišten je modul za prepoznavanje lica. Tako prvi puta kada vidi čovjeka Pepper zapamti njegovo lice, a kada ga ugleda sljedeći put pozdravi ga po imenu i rukuje se s njim. U

radu [19] napravljene su i testirane funkcije za prepoznavanje lica čovjeka. Kako bi raspoznavanje dobro funkcioniralo potrebno je osigurati dovoljno svjetla oko robota i čovjek ga mora gledati ravno u lice. Pogreške se događaju ako čovjek robota gleda pod drugačijim kutom nego što ga je zapamtio. Ako Pepper dobro vidi čovjeka u svojem vidnom polju, točno ga prepoznaje po imenu. Ime čovjeka unosi se preko računala, a ne usmeno, kako bi se izbjegla mogućnost pamćenja osobe pod krivim imenom.

3.2.6. Tablet aplikacija

U sklopu ovog rada razvijene su dvije multimedijske web stranice. Stranice služe za višenamjensku interakciju s korisnikom preko tableta na prsima Peppera.

Prva web stranica

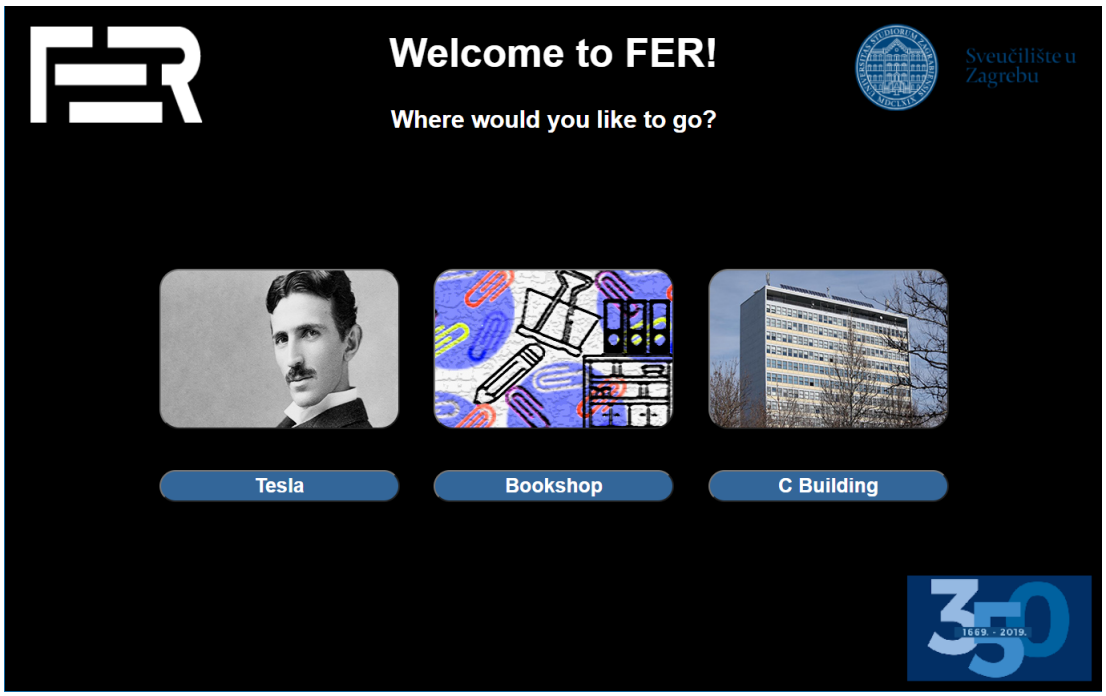


Slika 3.11: Prva web stranica

Prva web stranica (slika 3.11) prikazuje se nakon upoznavanja ili prepoznavanja čovjeka. Na stranici su prikazane tri različite opcije, a korisnik može odabrati između obilaska nekih od lokacija unutar zgrade fakulteta (tada aplikacija prelazi na sljedeću web stranicu gdje se nalazi odabir), igranja igrice "*križić kružić*" te posjećivanja LARICS-ove web stranice gdje se mogu pronaći razne zanimljivosti vezane uz laboratorij. Na ovaj način proširena je multimedijska funkcionalnost Peppera kao

menadžera dobrodošlice. Igrica je namijenjena za mlađe uzraste kojima nije toliko zanimljiv obilazak fakulteta. Moguće je igrati protiv robota protiv druge osobe, na taj način je omogućeno uključenje više djece u igru u isto vrijeme i ona postaju zainteresirana za interakciju s robotom.

Druga web stranica



Slika 3.12: Druga web stranica

Druga web stranica (slika 3.12) služi kao glavni izbornik za odabir lokacija na koje Pepper može voditi posjetitelje fakulteta. Moguće je odabrati između spomenika Nikole Tesle, skriptarnice i C zgrade fakulteta. Na njoj su sadržani slikovni prikazi lokacija, apstraktno ili realno. Prilikom odabira očitava se koja je lokacija pritisnuta te se prema tome aktivira odgovarajući algoritam za navigaciju do nje.

Aplikaciju na tabletu robota nije moguće pokretati lokalno s računala, ona se mora pospremiti u Pepperovu memoriju te iz nje pokretati. Spremanje u memoriju vrši se preko programa Choregraphe. Mapa (engl. *folder*) imena *html* sprema se u ponašanje (engl. *behavior*), a stranica koja se prikazuje na tabletu mora biti imena *index.html*. Cijelo ponašanje sprema se u memoriju pod određenim imenom te se aplikacija iz Pythona pokreće s modulom *ALBehaviorManager* metodom *runBehavior*.

3.3. Stablo ponašanja

Metoda stabla ponašanja model je kojim se opisuje autonomno ponašanje robota. Glavna odlika metode stabla ponašanja (engl. *Behavior Tree method*) je da pruža veću robusnost i modularnost prilikom projektiranja. Stablo se sastoji od više malih podstabala koje se svako zasebno modulira i testira. Veliko stablo ponašanja dobiva se spajanjem više malih podstabala u jednu funkcionalnu cjelinu te je zato lagano vršiti popravke na pojedinim podstablima. Također, pojedino podstablo može se ukloniti ili zamijeniti s drugim bez da se mijenja izvedba stabla.

3.3.1. Izrada stabla ponašanja

Kao i svako stablo, sastoji se od čvorova (engl. *nodes*) koji se nazivaju korijen (engl. *root*) i listovi (engl. *leaves*). Svi čvorovi međusobno odnose se kao roditelji (engl. *parents*) i djeca (engl. *children*). Korijen je jedini čvor koji ima samo djecu, nema roditelja, a na najnižoj razini stabla su listovi koji imaju samo roditelje, nemaju djecu. Svi preostali čvorovi u stablu su roditelji i djeca ostalim čvorovima. Stablo ponašanja ima šest vrsta čvorova, a oni su redom: slijedni (engl. *sequence node*), povratni (engl. *fallback node*), paralelni (engl. *parallel node*), dekorativni (engl. *decorator node*), akcijski (engl. *action node*) i uvjetni (engl. *condition node*). Njihovi simboli prikazani su na slici 3.13. Karakteristika stabla ponašanja je međusobna komunikacija svih navedenih čvorova odašiljući poruke uspjeh (engl. *success*), pogreška (engl. *failure*) ili u izvođenju (engl. *running*) [20]. Tako je moguće u potpunosti promijeniti strukturu nekog čvora bez potrebe za promjenom njegovog roditelja. Bitna stvar su poruke koje se odašilju između njih, a ne funkcionalnost svakog čvora.

Slika 3.13: Čvorovi u stablu ponašanja

Slijedni čvor djecu pokreće s lijeva na desno, jedno po jedno. Svojem roditelju vraća uspjeh ako sva njegova djeca njemu vrate uspjeh. Ako samo jedno dijete vrati pogrešku ili da je u izvođenju, on svojem roditelju vraća pogrešku ili da je u izvođenju. Funkcioniranje slijednog čvora može se usporediti s funkcioniranjem logičkog I (engl. *AND gate*) u logičkim operacijama. Pseudokod slijednog čvora prikazan je u kodu 3.1.

```
for i <-- 1 to N do
    childStatus <-- Tick(child(i))
    if childStatus == running then
        return running
    else if childStatus == failure then
        return failure
return success
```

Kod 3.1: Pseudokod za slijedni čvor

Povratni čvor pokreće djecu s lijeva na desno, jedno po jedno. Svojem roditelju vraća uspjeh ako barem jedno njegovo dijete vrati uspjeh. Kako bi povratni čvor svojem roditelju vratio pogrešku, sva njegova djeca njemu moraju vratiti pogrešku. Kada bilo koje dijete vrati da je u izvođenju, on također vraća da je u izvođenju. Funkcioniranje povratnog čvora može se usporediti s funkcioniranjem logičkog ILI (engl. *OR gate*) u logičkim operacijama. Pseudokod povratnog čvora prikazan je u kodu 3.2.

```
for i <-- 1 to N do
    childStatus <-- Tick(child(i))
    if childStatus == runnin} then
        return running
    else if childStatus == success then
        return success
return failure
```

Kod 3.2: Pseudokod za povratni čvor

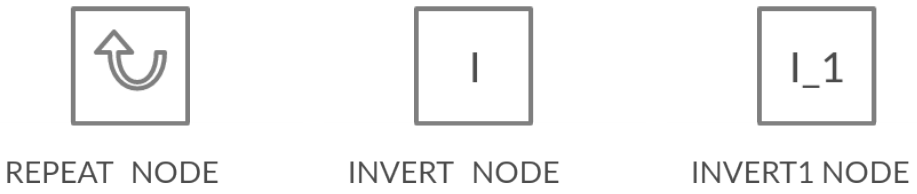
Paralelni čvor ima N djece i sve ih pokreće, s lijeva na desno. Broj M, $M \leq N$, definira korisnik, a određuje koliko djece mora vratiti uspjeh kako bi i paralelni čvor vratio uspjeh. Paralelni čvor vraća pogrešku ako $N - M + 1$ djece vrati pogrešku. U preostalim slučajevima vraća da je u izvođenju. Pseudokod paralelnog čvora prikazan je u kodu 3.3.

```
for i <-- 1 to N do
    childStatus(i) <-- Tick(child(i))
if number of(childStatus(i) == success) >= M then
    return success
else if number of(childStatus(i) == failure) >= (N - M) then
    return failure
return running
```

Kod 3.3: Pseudokod za *parallel node*

Dekorativni čvor je posebna vrsta čvora koji korisniku daje slobodu u slaganju stabla ponašanja. Nema točno definirano kada vraća uspjeh, pogrešku ili da je u izvođenju, nego te odnose definira korisnik, kao i simbol čvora. U radu su korištena tri dekorativna čvora prikazana na slici 3.14. Ponavljajući čvor (engl. *repeat node*) beskonačno se vrti dok njegovo dijete ne vrati uspjeh te tada i on sam vraća uspjeh. Ako dođe do pogreške u čvoru djeteta, on isto vraća pogrešku. Za sve vrijeme izvođenja petlje vraća da je u izvođenju. Napravljena su dva invertirajuća čvora (engl. *invert node*) čija je funkcija međusobno različita. Prvi invertirajući čvor, na slici 3.14 *invert node*, služi za invertiranje izlaza djeteta. Ako je dijete vratilo uspjeh, on to pretvori u pogrešku i obrnuto. Kada je konverzija uspješno obavljena, invertirajući čvor vraća uspjeh, inače pogrešku. Drugi invertirajući čvor, na slici 3.14 *invert1 node*, radi kao polukonverter izlaza djeteta. Ako dijete vrati uspjeh, on to pretvara u pogrešku, inače vrijednost koju je vratilo dijete ostaje nepromijenjena. Invertirajući čvor vraća uspjeh ako je konverzija uspješno obavljena ili za njom nije bilo ni potrebe.

Akcijski i uvjetni čvorovi nemaju svoju djecu, imaju samo roditelje. Akcijski čvor izvodi zadanu akciju, a uvjetni provjerava je li zadani uvjet ispunjen. Vraćaju uspjeh kada su akcija ili uvjet napravljeni ili ispunjeni, inače pogrešku. Akcijski čvor vraća da je u izvođenju za vrijeme trajanja akcije, a uvjetni čvor ne može vratiti da je u izvođenju. U radu [19] detaljno je opisan način stvaranja stabla ponašanja i matematički prikaz cijelog modela.

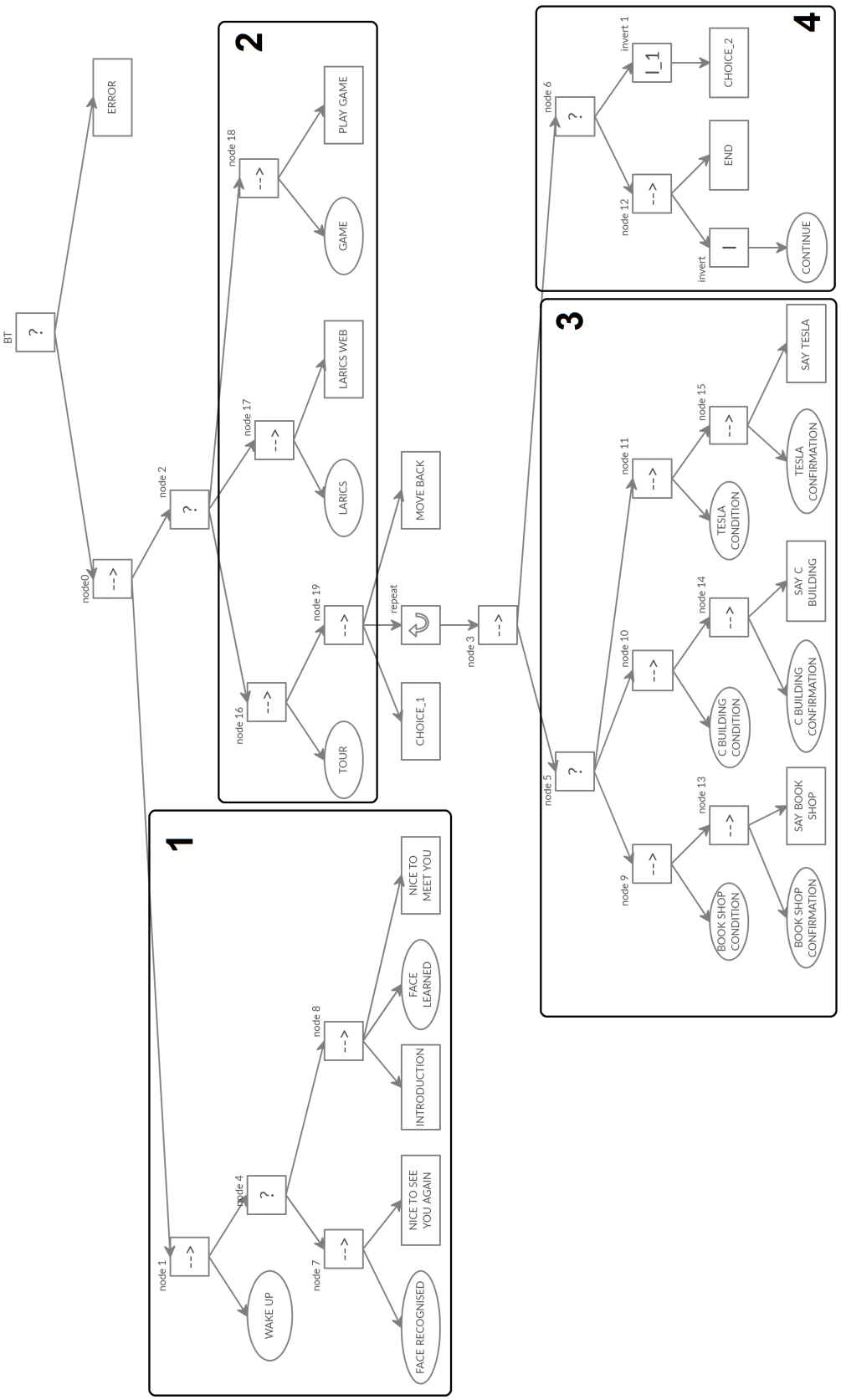


Slika 3.14: Posebni dekorativni čvorovi

3.3.2. Pepper kao menadžer dobrodošlice opisan metodom stabla ponašanja

Scenarij kojim je Pepper kao menadžer dobrodošlice opisan stablom ponašanja sadrži 16 slijednih, 5 povratnih, 3 dekorativna te 13 akcijskih i 13 uvjetnih čvorova. Svaki čvor je programiran kao zasebna Python skripta. Sve skripte međusobno su po-

vezane preko posebne zasebne skripte naziva *variables.py* u kojoj je kreirana struktura *singleton* pomoću koje sve ostale skripte mogu zapisivati i čitati vrijednosti varijabli zapisanih u njoj [21]. Na slici 3.15 prikazano je stablo ponašanja za scenarij opisan u poglavlju 2. Pod brojem 1 prikazano je podstablo koje služi za prepoznavanje ili upoznavanje čovjeka. Čvor *node1* pokreće čvor *wake up* za buđenje robota te nakon što on vrati uspjeh pokreće se čvor *node4*. On prvo pokreće podstablo za raspoznavanje lica i pozdravljanje osobe imenom. Ako ono vrati pogrešku, pokreće drugo podstablo za predstavljanje i učenje čovjekovog lica. Kada bilo koji od ta dva stabla vrati uspjeh, čvor *node4* vraća uspjeh te čvor *node1* također vraća uspjeh. Čvor *node4* vraća pogrešku ako oba stabla za raspoznavanje ili učenje lica čovjeka vrate pogrešku. Nakon uspješnog izvođenja čvora *node1*, pokreće se čvor *node2* koji služi za očitanje koji gumb je pritisnut na tabletu robota, vidljivo pod brojem 2 na slici 3.15. Nakon očitanja, čvorovi *node16*, *node17* i *node18* pokreću uvjetne čvorove za provjeru koji je gumb pritisnut. Ako su pritisnuti gumbi za pokretanje igrice ili web stranice LARICS-a, čvorovi *node17* i *node18* pokreću potrebne akcije. U slučaju odabira vođenja po fakultetu, čvor *node16* pokreće čvor *repeat* koji pokreće podstablo za odabir lokacije, prikazane pod brojem 3 na slici 3.15. Ono se sastoji od tri manja idejno jednaka podstabla i čvora *node5* u kojem se pokreće algoritam za prepoznavanje riječi. Kada je prepoznata bilo koja ključna riječ, on prestaje te pokreće jedno od svoja tri podstabla, ovisno o tome je li prepoznata riječ *Tesla monument*, *bookshop* ili *C building*. Ako je, na primjer, robot čuo riječ *C building*, pokreće čvor *node10* koji pokreće uvjetni čvor za provjeru riječi. Nakon što je prepoznata dobra riječ, čvor *node14* pokreće čvor za potvrdu da je robot čuo točnu riječ te se i nakon njegovog uspješnog obavljanja pokreće čvor *say Tesla* u kojem se pokreće algoritam za navigaciju do Tesline biste i paralelno izgovaranje teksta o njemu. Nakon uspješnog dolaska na lokaciju, pokreće se podstablo pod brojem 4 na slici 3.15 koje provjerava želi li sugovornik nastaviti vođenje po fakultetu. Ako je odgovor negativan, izvodi se blok *end* koji, kada vrati uspjeh, pokreće reakciju da svi čvorovi do čvora *repeat* vrate uspjeh. Potom se pokreće čvor *move back* koji robota vraća na početnu poziciju te kada se uspješno završi svi preostali čvorovi vraćaju uspjeh i interakcija je završena. Ako je odgovor pozitivan, čvor *invert* svome roditelju šalje pogrešku, on pokreće čvor *invert1* i *choice_2* za izgovaranje teksta o nastavku komunikacije. Nakon tog izvršenja, ponovno se počinje vrtjeti petlja u čvoru *node5* za slušanje riječi te tako sve dok sugovornik ne odluči prekinuti komunikaciju. U slučaju pogreške bilo kojeg čvora u stablu, pokreće se čvor *error* koji javlja u kojem čvoru je došlo do pogreške te tako olakšava provjeru njezinog uzroka. Interakcija je uspješna kada čvor *node0* vrati uspjeh.



Slika 3.15: Scenarij opisan metodom stabla ponašanja

4. Rezultati

4.1. Rezultati u simulatoru

Sva testiranja provedena su u Gazebo simulatoru u kojem je moguće testirati sposobnost zaobilaženja dinamičkih prepreka različitoga oblika i veličina.

4.1.1. Rezultati navigacije u simulatoru

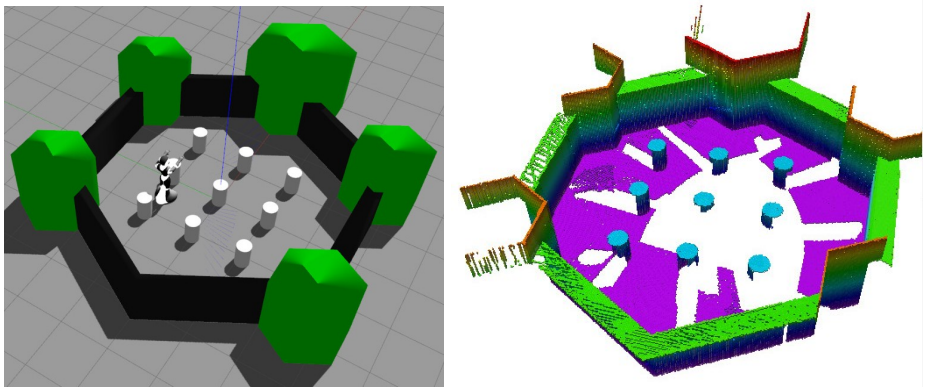
Navigacijski stog korišten za navigaciju uspješno je testiran u simulatoru. Na zadani cilj robot dolazi optimalnom trajektorijom planera, krećući se prosječnom linearnom brzinom od 25 cm/s. Robot se kreće po već gotovim, unaprijed napravljenim, mapama zatvorenog prostora. Testiranje je uspješno provedeno u napravljenim poligonima na slikama 4.2a i 4.3a. Prije same navigacije potrebno je napraviti 2D mape prikazane na slikama 4.2c i 4.3c. Na slici 4.1 prikazan je ispis koji se prikazuje na ekranu za vrijeme kretanja robota. Porukom "cilj postignut" (engl. *goal reached*) označava se kako je robot uspješno stigao na zadanu poziciju. U radu [15] detaljno je obrađena navigacija do nekoliko različitih ciljeva u statičkim uvjetima i sposobnost izbjegavanja dinamičkih prepreka.

```
[ INFO] [1588852562.424881972, 39.382000000]: Got new plan
[ INFO] [1588852563.011286579, 39.582000000]: Got new plan
[ INFO] [1588852563.664585858, 39.782000000]: Got new plan
[ INFO] [1588852564.232137332, 39.982000000]: Got new plan
[ INFO] [1588852564.811087189, 40.182000000]: Got new plan
[ INFO] [1588852565.391705832, 40.382000000]: Got new plan
[ INFO] [1588852565.391796954, 40.382000000]: Goal reached
```

Slika 4.1: Ispis po uspješnoj navigaciji u simulatoru

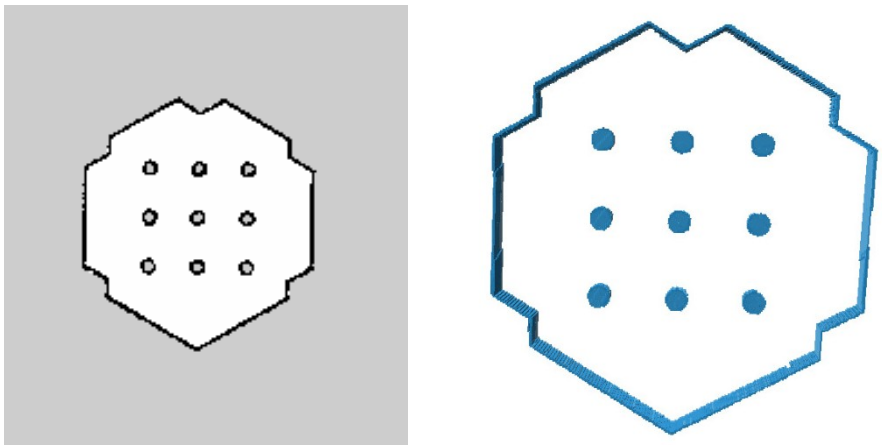
4.1.2. Rezultati mapiranja u simulatoru

Mapiranje u simulatoru je provedeno uz rezoluciju od dva centimetra po vokselu. Napravljena su dva različita poligona za kretnju robota u simulatoru. Prvi na slici 4.2a je približnog oblika šesterokuta, a drugi na slici 4.3a je pravokutnog oblika. Mapiranjem šesterokutnog poligona dobije se vrlo dobra mapa prostora prikazana na slici 4.2b. Vidljivo je kako su zidovi u potpunosti pravilno mapirani, a pojedini valjci u sredini nisu. Razlog tomu je što robot kada jednom prođe pored ravnog zida jasno očitava prepreke, a kružne oblike ne može očitati u potpunosti. Rješenje tog problema je da robot nekoliko puta prođe oko svakog valjka te tako pokuša mapirati što veći njegov dio. Ipak, usporedbom 2D mapa prostora na slici 4.2c i projekcija u 2D ravninu na slici 4.2d, vidljivo je kako za time nema potrebe. Mape su gotovo identične te se zaključuje kako robot u oba slučaja može znati gdje se nalazi i koje su prepreke oko njega.



(a) Šesterokutni 3D prostor

(b) 3D mapa šesterokutnog prostora

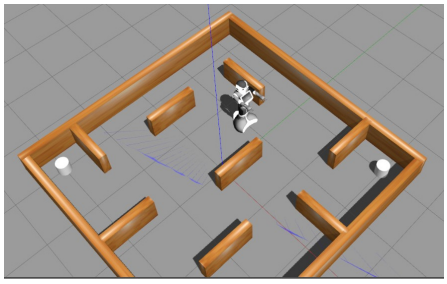


(c) 2D mapa šesterokutnog prostora

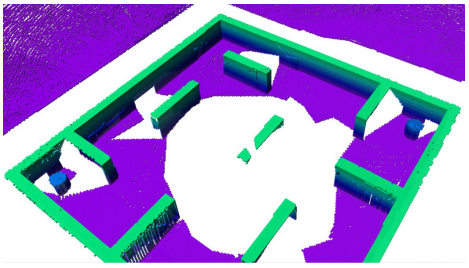
(d) Projekcija 3D mape u 2D ravninu

Slika 4.2: Šesterokutni prostor u simulatoru i dobivene mape

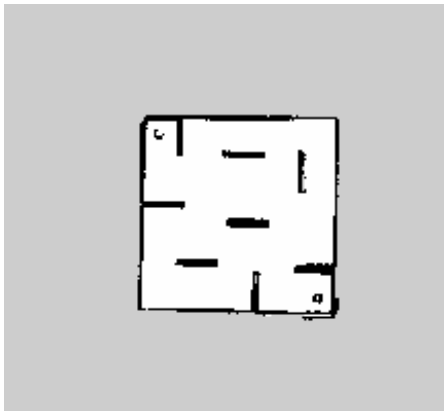
Trodimenzionalna mapa pravokutnog poligona prikazana je na slici 4.3b. Vidljivo je kako su svi vanjski zidovi pravilno mapirani, ali pojedine unutarnje prepreke oblika kvadra nisu. Ako je robot slučajno u trenutku prolaska pored prepreke imao okrenutu glavu, prepreka mu se mogla naći izvan vidokruga. Takva pogreška se ne događa sa zidovima jer su oni veći te ih je moguće mapirati iz više različitih kutova. Moguće rješenje ovog problema je da robot mapom prođe nekoliko puta te tako bude siguran da je u vidokrugu uhvatio sve potrebne plohe zidova i prepreka. Također, kao i kod šesterokutnog poligona, usporedbom slika 2D mape prostora na slici 4.3c i projekcije mape u 2D ravninu na slici 4.3d vidljivo je kako za time nema potrebe. Mape su gotovo identične osim u prepreci koja se nalazi u sredini. Ona u sebi sadrži određenu rupu nastalu prilikom 3D mapiranja, ali ona ne utječe na samu projekciju mape u 2D prostor. I dalje je poznato koji su rubovi te prepreke i gdje se ona točno nalazi. Robot je u oba slučaja svjestan svoje okoline, zna gdje se nalazi te koje su prepreke oko njega.



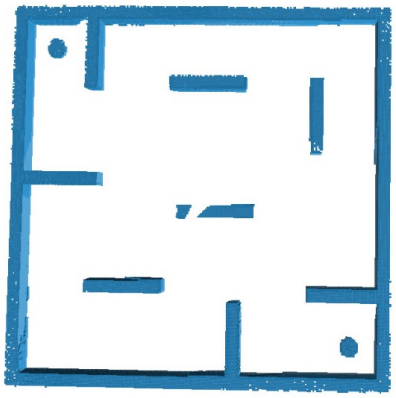
(a) Pravokutni prostor u simulatoru



(b) 3D mapa pravokutnog prostora



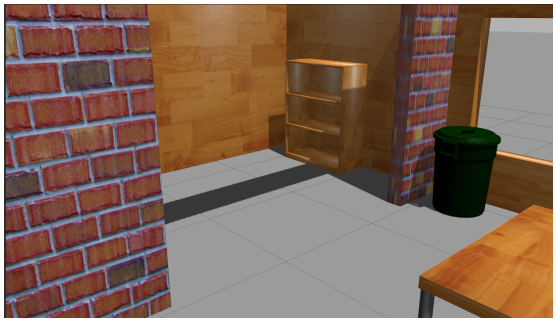
(c) 2D mapa pravokutnog prostora



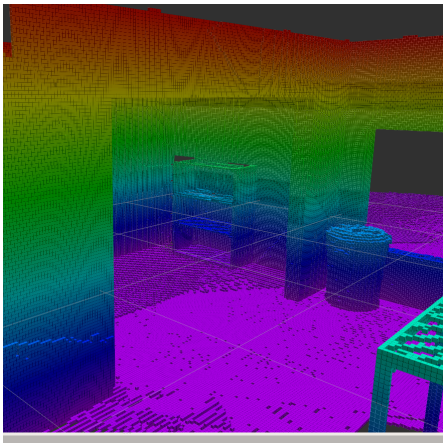
(d) Projekcija 3D mape u 2D ravninu

Slika 4.3: Pravokutni prostor u simulatoru i dobivene mape

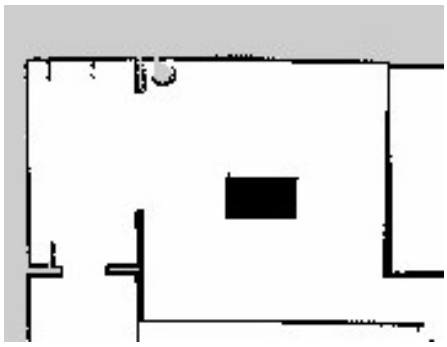
Na slici 4.4a prikazani su uvećani objekti u mapi koje je potrebno precizno mapirati. Na slici 4.4b vidljivo je kako je korištena rezolucija od dva centimetra po vokselu vrlo dobra. Iako su predmeti različitih oblika i veličina, svi su odlično mapirani i jasno ih se može vidjeti na mapi. Usporedbom 2D mape prostora na slici 4.4c s projekcijom u 2D prostor na slici 4.4d uočeno je kako postoje određena nepoklapanja. Na 2D mapi loše je mapirana polica, dok je na 3D mapi desni dio prolaza u hodniku nevidljiv. Iako mape nisu savršene, i dalje su vrlo dobre te robotu omogućavaju sigurno kretanje u njima.



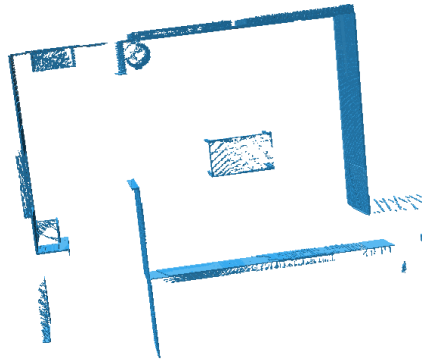
(a) Objekti u 3D prostoru



(b) 3D mapa objekata u prostoru



(c) 2D mapa objekata u prostoru



(d) Projekcija 3D mape u 2D ravninu

Slika 4.4: Objekti u prostoru u simulatoru i dobivene mape

Sva mapiranja provedena u simulatoru daju vrlo dobre rezultate bez većih problema koje bi bilo potrebno ukloniti.

4.2. Rezultati u laboratoriju

Sva testiranja provedena su u Laboratoriju za robotiku i inteligentne sustave upravljanja (LARICS) koji djeluje na FER-u.

4.2.1. Rezultati testiranja komunikacije u laboratoriju

Komunikacija je u laboratoriju testirana pet puta te je svaki put uspješno izvedena do kraja. Komunikacija s čovjekom preko tableta funkcionira svaki puta, svakim pritiskom na gumb točno se očitava koja pozicija na tabletu je pritisnuta. Prilikom svakog testiranja usmene komunikacije ispitane su sve tri lokacije za vođenje po fakultetu, skriptarnica, liftovi C zgrade i Teslina bista. U tablicama 4.1 i 4.2 numerički su prikazani dobiveni rezultati. Prilikom svakog testiranja uočeno je koliko je puta potrebno izgovoriti određenu riječ da bi ju Pepper razumio i kolika je sigurnost prepoznate riječi. U izračun prosječne sigurnosti prilikom pojedinog testiranja uzete su sve točne i netočne riječi koje je Pepper prepoznao. Iz tablice 4.1 vidljivo je kako prosječna sigurnost prilikom pojedinog testiranja iznosi između 55 i 65%. Takvi rezultati su očekivani i u skladu s prijašnjim testiranjima provedenim u [18]. U laboratoriju, u kojem u pravilu vladaju tišina i mirni uvjeti, sve riječi je potrebno izgovoriti samo jedanput, osim u zadnjem slučaju prilikom prepoznavanja riječi *bookshop*. Prilikom tog testiranja vladala je određena buka koja je onemogućila Pepperu da besprijekorno čuje izgovorenu riječ. Prepoznavanje i učenje lica čovjeka uspješno je funkcioniralo u četiri od pet slučajeva. Razlog tomu je vrlo dobro osvjetljenje u laboratoriju. Moguća pogreška dogodila se zbog gledanja robota u lice pod drugim kutom nego što je robot zapamtio lice.

Tablica 4.1: Prosječna sigurnost prepoznatih i broj ponavljanja riječi pri pojedinom pokušaju

Broj testa	**Broj ponavljanja / prosječna sigurnost Tesla monument**	**Broj ponavljanja / prosječna sigurnost Bookshop**	**Broj ponavljanja / prosječna sigurost C building**	**Prosječna sigurnost po testiranju**
1	1 / 51.23%	1 / 46.91%	1 / 63.25%	58.68%
2	1 / 58.68%	1 / 44.07%	1 / 63.25%	60.55%
3	1 / 49.26%	1 / 44.55%	1 / 66.37%	61.17%
4	1 / 49.26%	1 / 43.83%	1 / 70.51%	64.26%
5	1 / 45.08%	4 / 46.52%	1 / 65.37%	64.75%

$$n = \frac{\sum_{i=1}^{N} x_i}{N} \tag{4.1}$$

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{4.2}$$

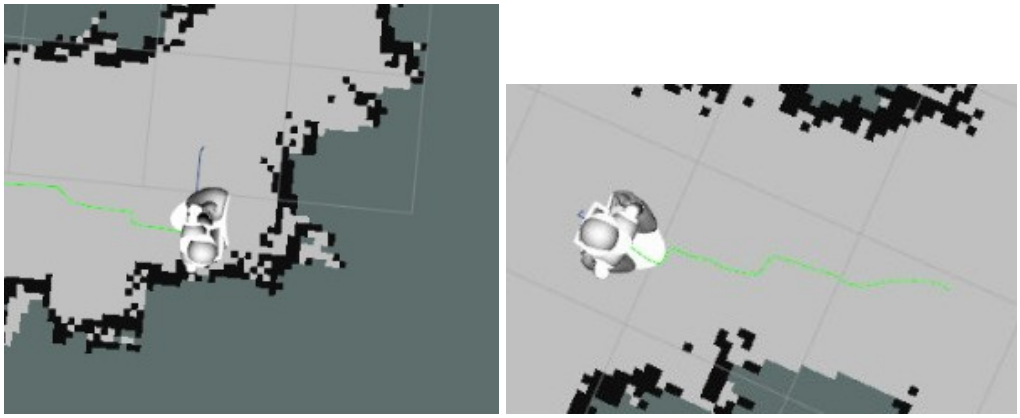
Tablica 4.2 predstavlja prosječnu sigurnost, računatu po formuli 4.1, po određenim riječima prilikom svih pet testiranja. Vidljivo je kako je najbolje prepoznata riječ *C building*, a najlošije *bookshop* sa sigurnošću malo većom od 40%. Prilikom računanja prosječne sigurnosti riječi u obzir su uzete sve točno i netočno prepoznate riječi. Standardna devijacija, računata po formuli 4.2, svih rezultata iznosi od 3 do 5% što prikazuje kako su sva mjerenja vrlo slična i dobro izvedena. Tri puta je neprepoznata riječ *bookshop* iz već navedenih razloga.

Tablica 4.2: Prosječna sigurnost prepoznavanja po pojedinoj riječi

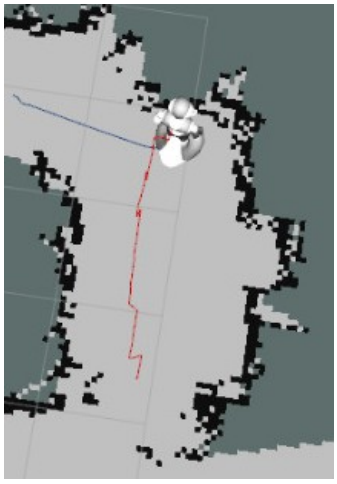
Riječ	Prosječna sigurnost	Standardna devijacija	Broj neprepoznatih riječi
Tesla monument	50.7	4.99	0
Bookshop	42.57	3.81	3
C building	65.75	2.99	0

4.2.2. Rezultati navigacije u laboratoriju

Za testiranje navigacije u laboratoriju, korištena je 2D mapa laboratorija prikazana na slici 4.7d. Pravocrtna navigacija u prostoru laboratorija funkcionirala je dobro i robot je uspješno dolazio do cilja. Gibao se vrlo sporo, minimalnom brzinom zadanom u lokalnom planeru, koja je bila 10 cm/s. Primjeri trajektorija za pravocrtnu navigaciju prikazani su na slici 4.5. Navigacija za ciljeve koji zahtijevaju trajektoriju s rotacijom robota nije bila uspješna. Robotova kutna brzina nikad nije bila veća od 0.03 rad/s i robot bi često došao vrlo blizu statičkih prepreka u mapi i aktivirao bi se sigurnosni sustav za zaustavljanje robota kako ne bi došlo do kolizije s preprekom. Primjer trajektorije za takav slučaj prikazan je na slici 4.6.



Slika 4.5: Primjeri trajektorija za pravocrtnu navigaciju u laboratoriju



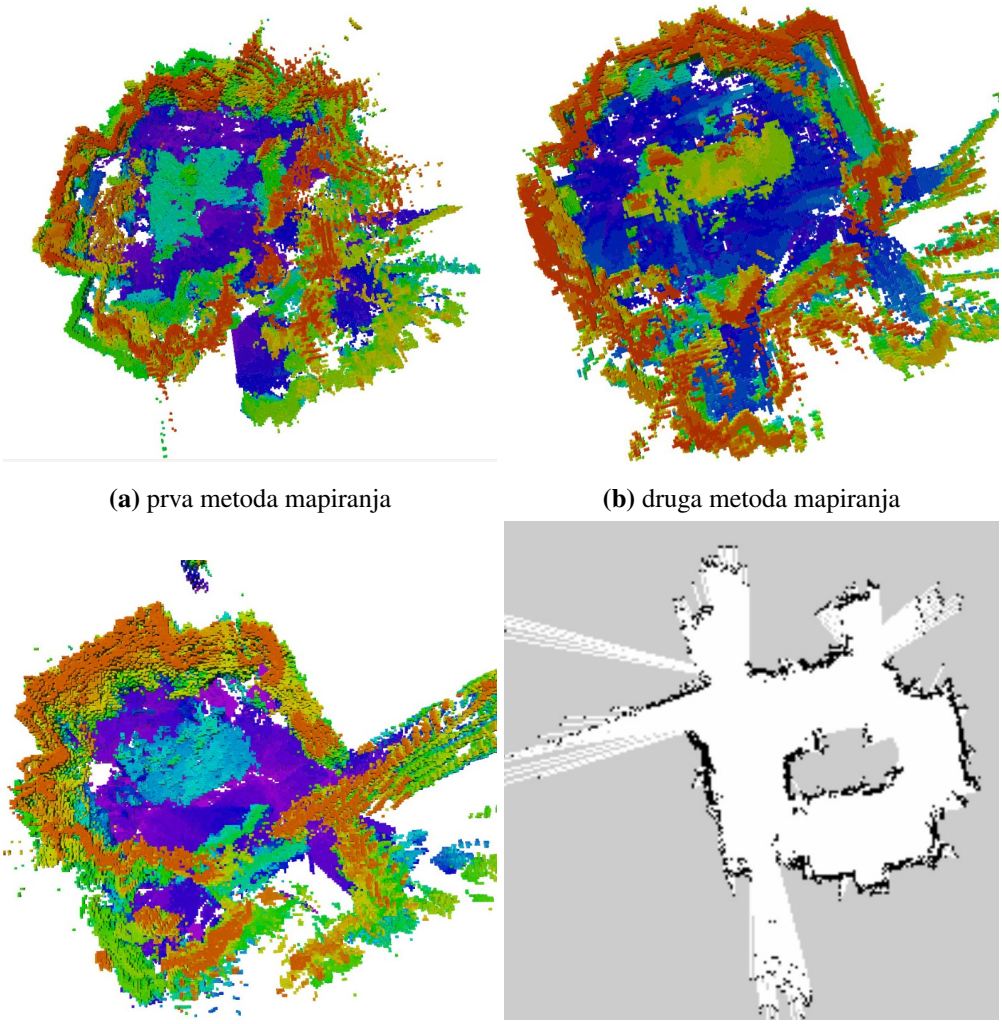
Slika 4.6: Primjer trajektorije za aktivaciju sustava za sigurnosno zaustavljanje robota (crvenom bojom označena je robotova trajektorija, a plavom bojom trajektorija lokalnog planera)

4.2.3. Rezultati mapiranja u laboratoriju

Za mapiranje prostora laboratorija korištena je standardna (engl. *default*) vrijednost rezolucije voksela od pet centimetara. Kako bi se odredio utjecaj načina mapiranja na kvalitetu mape, dobivene su mape laboratorija uz tri različite metode mapiranja. U prvoj metodi korištene brzine za upravljanje robota iznosile su: linearna brzina od 0.1 m/s u smjeru x osi i kutna brzina od 0.2 rad/s oko z osi. Kako bi se dobilo čim bolje očitanje prostora oko robota, izvedeno je puno rotacija robota oko njegove osi. Druga metoda istovjetna je prvoj, samo su brzine upravljanja kretanjem robota iznosile: linearna brzina od 0.2 m/s u smjeru x osi i kutna brzina od 0.4 rad/s oko z osi. Za treću metodu korištene su iste brzine gibanja robota kao i za drugu metodu, ali uz značajno manji broj rotacija robota u odnosu na prvu i drugu metodu. Pomoću alata Choregraphe rotirala se glava robota dok je tijelo mirovalo. Također, ispitan je utje-

caj podloge na kvalitetu konačne mape. Izrađene su mape prostorija koje imaju dvije različite podloge: mramor i parket.

Usporedba mapa dobivenih navedenim metodama mapiranja prikazana je na slici 4.7 i slici 4.8. Vidljivo je da se mapa najveće kvalitete dobije za treću metodu mapiranja. Objekti u 3D mapi jasno su vidljivi i usporedivi sa stvarnom slikom laboratorija na slici 4.8d. Prilikom rotacije robota oko vlastite osi dolazi do trzanja cijelog tijela što ima za posljedicu smanjenje kvalitete dobivene mape. Najveći efekt trzanja zabilježen je pri manjim brzinama rotacije pa je posljedično i mapa dobivena prvom metodom ona najmanje kvalitete. Isti se zaključci mogu primijeniti i na utjecaj podloge. Neravnija podloga, kao što je mramor, uzrokuje znatno više trzanja robota nego ravnija podloga, kao što je parket. Dobivena mapa prostorije popločene parketom prikazana je na slici 4.10. Vidljivo je kako se na slici 4.10a može jasno očitati da se određeni objekti nalaze uzduž zidova, što je vidljivo na slici 4.10c koja prikazuje stvari prostor. Mapiranje prostora ponovljeno je trećom metodom u uvjetima niskog vanjskog osvjetljenja, a dobivena mapa prikazana je na slici 4.9. Mapiranje u uvjetima niskog vanjskog osvjetljenja pokazalo se boljim zbog toga što je značajno smanjen utjecaj refleksije od podloge i time je smanjena vjerojatnost nepoželjnog detektiranja nepostojećih prepreka u prostoru. Analizom je utvrđeno kako se ovom metodom objekti najbolje ocrtavaju i kako su najpravilniji. Također, testirano je mapiranje uz rezoluciju voksela od dva centimetra, ali dobivene mape zauzimale su znatno veću količinu memorije i bile su dosta zahtjevne za vizualizaciju. Uzrokovale su nenadane prekide (engl. *crash*) procesa mapiranja i nije dobivena mapa cijelog laboratorija. Za mali dio prostora moguće je dobiti mapu veće rezolucije koja puno bolje ocrtava objekte. Usporedba je vidljiva na slici 4.11. Mapa stvarnog prostora prikazanog na slici 4.11c mapirana većom rezolucijom na slici 4.11a puno je preciznija i detaljnija od one niže rezolucije na slici 4.11b. Na slici 4.12 prikazano je kako je u realnim uvjetima na velikim mapama dovoljno koristiti rezoluciju od pet voksela jer se objekti dovoljno dobro ocrtavaju.



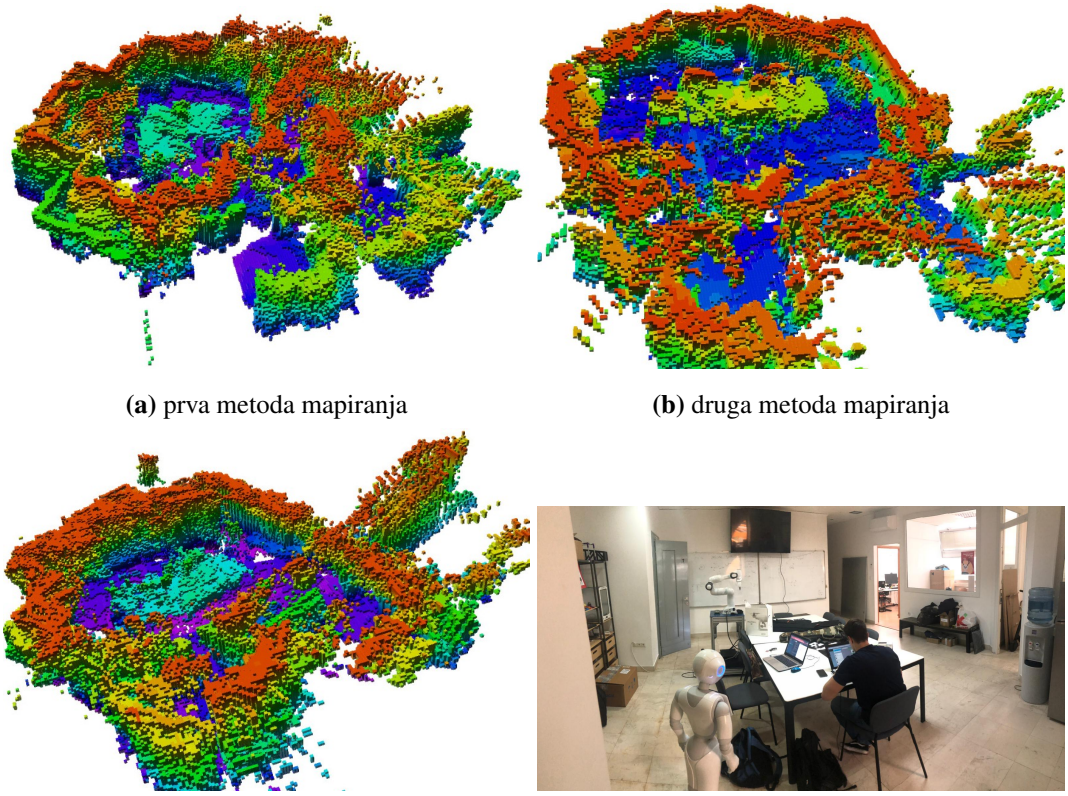
(a) prva metoda mapiranja

(b) druga metoda mapiranja

(c) treća metoda mapiranja

(d) 2D mapa prostora

Slika 4.7: Usporedba dobivenih mapa laboratorija za različite metode mapiranja (pogled odozgo)



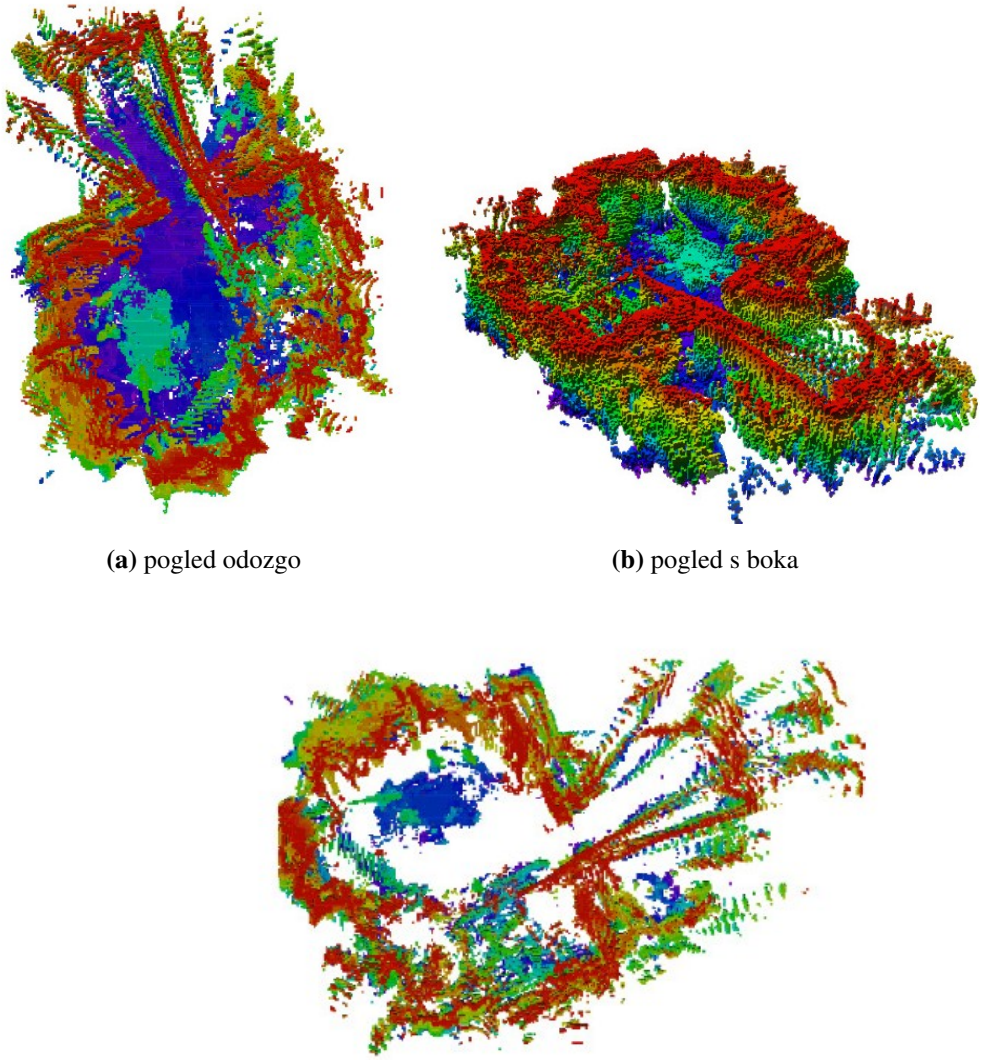
(a) prva metoda mapiranja

(b) druga metoda mapiranja

(c) treća metoda mapiranja

(d) panoramska slika laboratorija

Slika 4.8: Usporedba dobivenih mapa laboratorija uz različite metode mapiranja (pogled s boka)

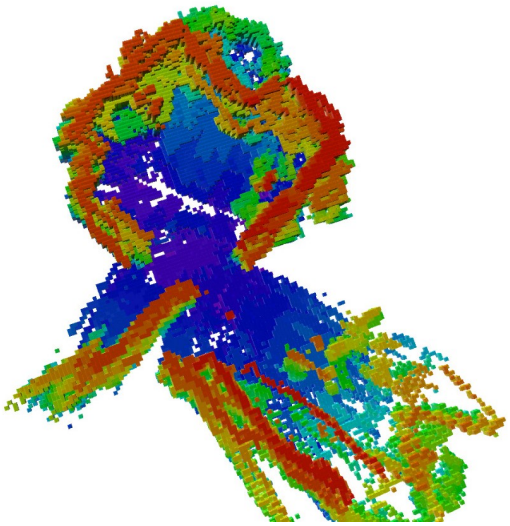


(a) pogled odozgo

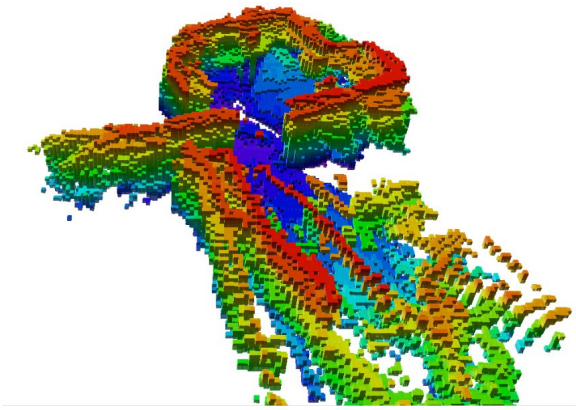
(b) pogled s boka

(c) projekcija 3D mape na 2D ravninu

Slika 4.9: Mapa laboratorija dobivena trećom metodom u uvjetima niskog vanjskog osvjetljenja



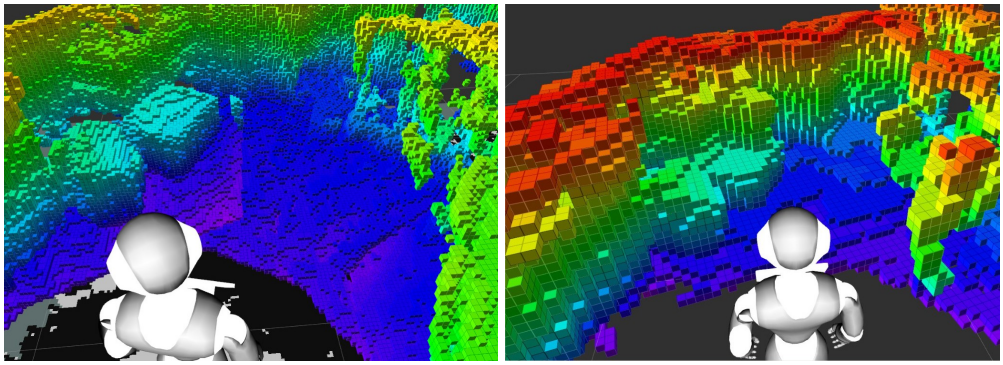
(a) pogled odozgo



(b) pogled s boka

(c) panoramska slika prostorije

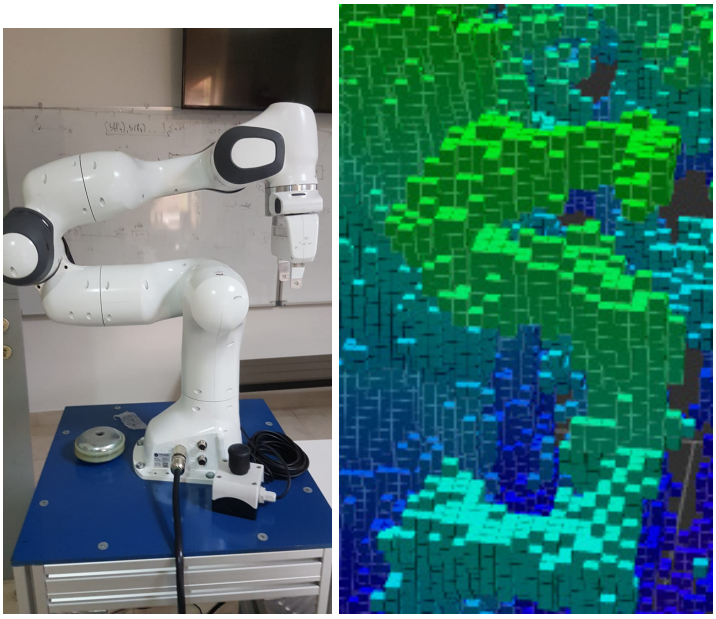
Slika 4.10: 3D mapa prostorije popločene parketom



(a) rezolucija voksela od 2 centimetra **(b)** rezolucija voksela od 5 centimetara

(c) prostor u stvarnosti

Slika 4.11: Usporedba stvarnog prostora i 3D mapa za različite rezolucije voksela



(a) objekt u stvarnosti **(b)** dobiveni 3D model

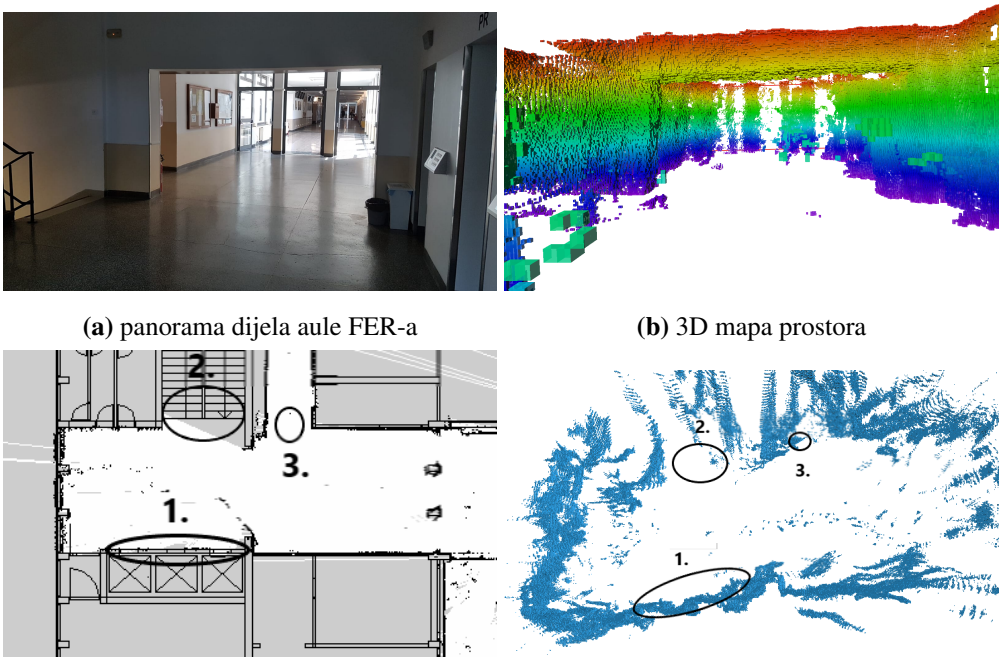
Slika 4.12: Usporedba stvarnog objekta i dobivenog 3D modela

4.3. Rezultati u prostorima FER-a

Sva testiranja provedena su u auli FER-a.

4.3.1. Mapiranje u prostoru FER-a

Za mapiranje aule FER-a korištena je treća metoda mapiranja opisana u rezultatima u laboratoriju u poglavlju 4.2.3. Korištena rezolucija voksela je pet centimetara. Na slikama 4.13 i 4.14 prikazane su usporedbe stvarnog prostora na fakultetu i dobivene 3D i 2D mape. Slika 4.14 prikazuje prostor koji se nalazi pored glavnog ulaza FER-a. Pod, stupovi u sredini i dijelovi zidova obloženi su mramorom. Mramor vrlo jako odbija svijetlo te se dubinskoj kameri prilikom mapiranja stvaraju problemi. Također, u auli ljudi stalno prolaze ispred robota te on gubi percepciju gdje se što nalazi. Usporedbom mapa na slikama 4.14c i 4.14d vidljivo je kako je mapiranje vrlo loše. Na 2D mapi pod brojem 1 vidljiv je stup oko kojeg nema ostalih predmeta, a projekcijom 3D mape vidljivo je kako se u trenutku mapiranja nešto nalazilo pored njega (ljudi u pokretu). Mapiranje vrlo dobro funkcionira u prostorima koji su statični, kao što je laboratorij.



(a) panorama dijela aule FER-a

(b) 3D mapa prostora

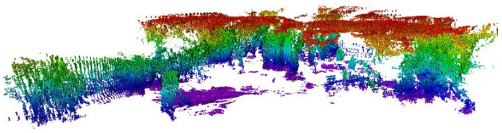
(c) 2D mapa prostora

(d) projekcija 3D mape na 2D ravninu

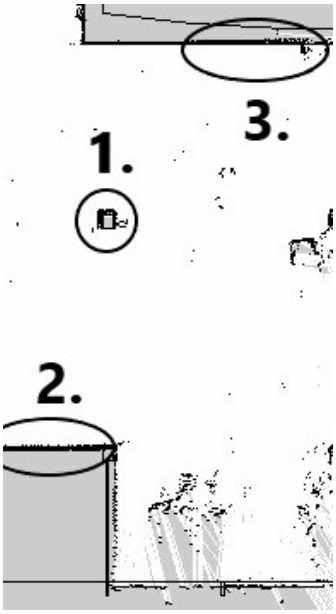
Slika 4.13: Usporedba stvarnog prostora i dobivene mape dijela prostora FER-a (brojem 1 označeni su liftovi, brojem 2 stepenice, a brojem 3 hodnik)

Bolja mapa, iako i dalje lošija nego u laboratoriju, dobivena je mapiranjem prostora u drugom dijelu aule, pored liftova C zgrade. U tom prostoru pod je od kamena, zidovi bijelo obojeni te je količina svijetla manja nego ispred glavnog ulaza. Refleksija od objekata u tom je slučaju smanjena te je zato mapa bolja. Također, frekvencija ljudi koji prolaze ispred robota manja je nego ispred glavnog ulaza. Usporedbom mapi na slikama 4.13c i 4.13d vidljivo je kako je moguće provesti određenu usporedbu lokacija označenim brojevima. Poboljšanje mapa na slikama 4.14d i 4.13d moguće je mapiranjem po mraku s minimalnim osvjetljenjem.

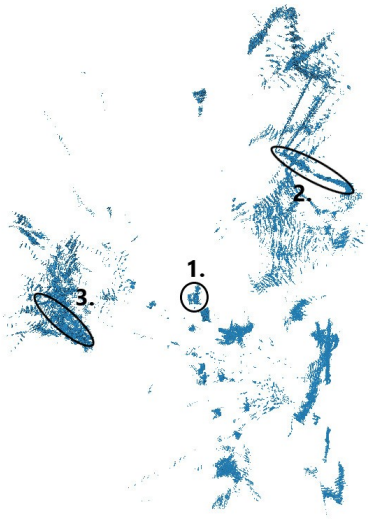
(a) panorama dijela aule FER-a



(b) 3D mapa prostora



(c) 2D mapa prostora



(d) projekcija 3D mape na 2D ravninu

Slika 4.14: Usporedba stvarnog prostora i dobivene mape dijela prostora FER-a (brojem 1 označen je stup, brojem 2 skriptarnica, a brojem 3 zid)

4.3.2. Testiranje komunikacije u prostoru FER-a

Kao i u laboratoriju, komunikacija je i u prostorima FER-a u realnom okruženju testirana pet puta. Komunikacija s čovjekom preko tableta i u realnom okruženju funkcionira besprijekorno. Razlika između uvjeta u laboratoriju i u prostorima FER-a je razina buke koja se stvara. U laboratoriju je tišina većinu vremena te Pepperu ništa ne smeta za slušanje, dok u auli fakulteta ljudi stalno prolaze te je razina buke u pojedinim trenucima neravnomjerna. U laboratoriju je riječi stalno izgovarao jedan čovjek dok je u prostorima fakulteta riječi izgovaralo više različitih ljudi. U tablici 4.3 prikazani su rezultati koliko je put bilo potrebno izgovoriti koju ključnu riječ te s kolikom ju je sigurnošću Pepper razumio. U skoro svakom pokušaju, osim jedanput, bilo je potrebno pojedine riječi izgovarati više puta. Vidljivo je kako nema pravila koju riječ Pepper ne čuje dobro. Razlog tomu može biti što druge osobe drugačije izgovaraju pojedine riječi, ali najčešće je razlog povremena buka koja se stvara prolaskom ljudi. Kao i u laboratoriju, prosječna sigurnost prepoznavanja svih riječi po pokušaju iznosi između očekivanih 55 i 65%. Komunikacija u realnim uvjetima funkcionira malo slabije, ali u prosjeku rezultati testiranja su slični kao i u laboratoriju. Testiranje prepoznavanja i učenja lica čovjeka uspješno je provedena u dva od pet slučaja. Razlog slabijem prepoznavanju lica u prostoru FER-a je slabija osvijetljenost prostora prilikom testiranja.

Tablica 4.3: Prosječna sigurnost prepoznatih i broj ponavljanja riječi pri pojedinom pokušaju

Broj testa	**Broj ponavljanja / prosječna sigurnost Tesla monument**	**Broj ponavljanja / prosječna sigurnost Bookshop**	**Broj ponavljanja / prosječna sigurnost C building**	**Prosječna sigurnost po testiranju**
1	2 / 48.49%	1 / 52.63%	2 / 67.58%	55.67%
2	1 / 48.4%	1 / 51.47%	1 / 59.68%	59.90%
3	1 / 53.59%	2 / 46.76%	1 / 58.35%	59.85%
4	2 / 55.08%	1 / 43.11%	1 / 66.66%	59.32%
5	1 / 52.96%	1 / 50.99%	2 / 66.52%	63.41%

Tablica 4.4 prikazuje prosječnu sigurnost, računatu po formuli 4.1, po određenim riječima prilikom svih pet testiranja. Vidljivo je kako je najbolje prepoznata riječ *C building*, a najlošije *Tesla monument*. Ovakvi rezultati razlikuju se od onih dobivenih u laboratoriju. Kao što je već rečeno, razlog tomu je povremena buka u prostoru fakulteta i više različitih ljudi koji su izgovarali riječi. Standardne devijacije, računate po formuli 4.2, svih riječi veće su nego one u tablici 4.2, što je u skladu s očekivanjima. Vidljivo je kako su one značajne za riječi *Tesla monument* i *C building*, ali to i dalje ne predstavlja

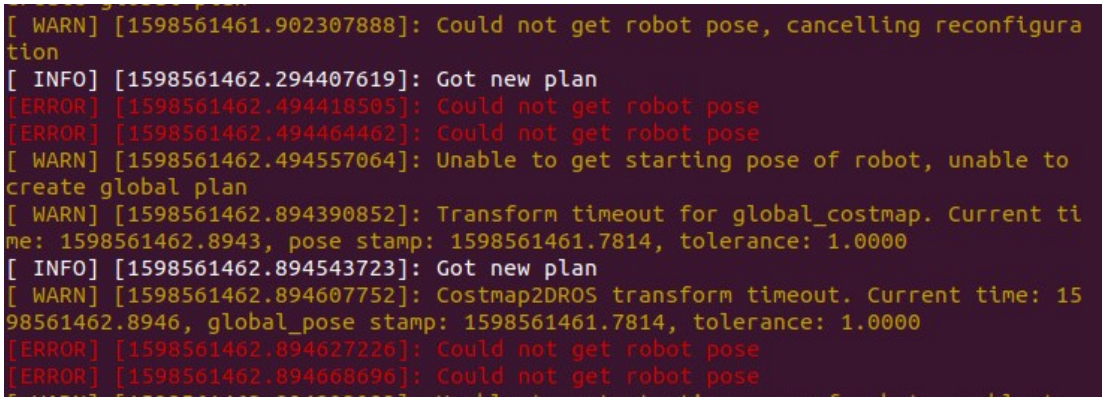
velike probleme u komunikaciji jer je broj neprepoznatih riječi u jednom i drugom slučaju samo dva.

Tablica 4.4: Prosječna sigurnost prepoznavanja po pojedinoj riječi

Riječ	Prosječna sigurnost	Standardna devijacija	Broj neprepoznatih riječi
Tesla monument	46.61	9.06	2
Bookshop	49.28	3.62	1
C building	57.97	12.33	2

4.3.3. Navigacija u prostoru FER-a

Korištenje ROS navigacijskog stoga u prostoru FER-a pokazalo se neuspješnim. Zbog nestabilne internetske veze, dolazilo je do kašnjenja od nekoliko sekundi u prijenosu podataka iz dubinske kamere i odometrije robota na računalo te lokalizacijski algoritam nije mogao odrediti trenutnu poziciju robota u prostoru. Na slici 4.15 prikazane su poruke koje označavaju neuspješnu lokalizaciju robota u prostoru. Kada bi se osigurala brža i stabilnija internetska veza, lokalizacija robota i njegova navigacija u prostorima FER-a bila bi moguća.



Slika 4.15: Ispis po neuspjeloj navigaciji zbog kašnjenja u prijenosu podataka

5. Zaključak

U ovom radu pokazani su rezultati i mogućnosti 2D i 3D mapiranja, navigacije u prostoru te audio vizualne komunikacije robota Pepper s ljudima. Prilikom 3D mapiranja uočene su očekivane razlike između rezultata mapiranja u simulatoru, laboratoriju te realnim uvjetima na fakultetu. Mapiranje u stvarnim uvjetima najbolje je kada tijelo robota miruje, okreće se samo glava, te je utjecaj vanjskog osvjetljenja sveden na minimum. Navigacija u simuliranim prostorima funkcionira u svim slučajevima dok u realnim uvjetima mora biti osigurana brza i stabilna internet veza kako bi komunikacija između robota i računala bila moguća. Ako je internet veza stabilna, kretnja robota moguća je po pravocrtnim trajektorijama. Prilikom promjene smjera kretanja robot nije u mogućnosti pratiti trajektoriju te završava svoje kretanje. U sklopu testiranja komunikacije robot Pepper u idealnim laboratorijskim uvjetima uspješno prepoznaje izgovorene riječi te se komunikacija provodi do kraja bez problema. U prostorima fakulteta, uz prisustvo buke, smanjena je sigurnost prepoznavanja riječi pa bi korisnik trebao stati bliže robotu. Tablet aplikacija je u potpunosti funkcionalna u svim testiranim uvjetima te tako može u lošijim uvjetima služiti kao zamjena za verbalnu komunikaciju s korisnikom i odabir željene lokacije za posjetu. Tablet pruža razne mogućnosti za interakciju s korisnikom poput odabira cilja navigacije, zabavljanja djece igranjem igrica s njima i pristup bilo kojim web stranicama kao izvor informacija.

ZAHVALA

Zahvaljujemo profesorima prof. dr. sc. Zdenku Kovačiću i doc. dr. sc. Tamari Petrović što su nam kao studentima treće godine preddiplomskog studija omogućili rad na stvarnom robotu i poticali nas na izvrsnost. Zahvala mag. ing. Ivanu Hrabaru i dr. sc. Frani Petricu što su svojim savjetima i materijalima pomogli prilikom izrade ovog rada.

LITERATURA

[1] *What Is Robotics?* URL: https://builtin.com/robotics.

[2] *Tipovi robota danas*. URL: https://robots.ieee.org/learn/types-of-robots/.

[3] James P. Trevelyan, Sung-Chul Kang i William R. Hamel. "Robotics in Hazardous Applications". *Springer Handbook of Robotics*. Ur. Bruno Siciliano i Oussama Khatib. Springer Berlin Heidelberg, 2008.

[4] *Pepper kao pomoćnik u bolnici u Brislu*. URL: https://www.brusselstimes.com/brussels/113643/robot-enforces-social-distancing-in-brussels-hospital/.

[5] D. Matić i Z. Kovačić. "NAO Robot as Demonstrator of Rehabilitation Exercises after Fractures of Hands". 2019.

[6] *Aldebaran Pepper dokumentacija*. URL: http://doc.aldebaran.com/2-4/home_pepper.html.

[7] *ROS dokumentacija*. URL: http://wiki.ros.org/.

[8] *Službena NAOqi Core dokumentacija*. URL: http://doc.aldebaran.com/2-5/naoqi/core/index.html.

[9] Armin Hornung i dr. "OctoMap: An efficient probabilistic 3D mapping framework based on octrees". (Travanj 2013).

[10] *Dokumentacija za paket depth_image_proc*. URL: http://wiki.ros.org/depth_image_proc.

[11] *Dokumentacija za paket teleop_twist_keyboard*. URL: http://wiki.ros.org/teleop_twist_keyboard.

[12] Karlo Valentin Cihlar. "Mapiranje prostora korištenjem senzora Pepper robota". (Lipanj 2020).

[13] *Dokumentacija za move_base paket*. URL: http://wiki.ros.org/move_base.

[14] *Dokumentacija za paket pointcloud_to_laserscan*. URL: http://wiki.ros.org/pointcloud_to_laserscan.

[15] Borna Zbodulja. "Navigacija Pepper robota u mapiranom zatvorenom prostoru". (Lipanj 2020).

[16] *Dokumentacija za paket amcl*. URL: http://wiki.ros.org/amcl.

[17] *Zadavanje navigacijskih ciljeva korištenjem ROS čvora*. URL: http://wiki.ros.org/navigation/Tutorials/SendingSimpleGoals.

[18] Matko Ferenca. "Upravljanje robotom Pepper putem govora". (Lipanj 2020).

[19] Petar Ljubotina. "Interakcija robota Pepper s ljudima opisana metodom stabla ponašanja". (Lipanj 2020).

[20] Michele Colledanchise i Petter Ogren. *Behavior Trees in Robotics and AI: An Introduction*. Srpanj 2018.

[21] Alexander Shvets. *Dive Into Design Patterns*. 2019.

Autori: Borna Zbodulja, Jakov Vulama, Petar Ljubotina
Humanoidni robot Pepper kao menadžer dobrodošlice na FER-u

Sažetak

Cilj ovog rada je pripremiti humanoidnog robota Pepper kao menadžera dobrodošlice na FER-u koji bi dočekivao goste i na temelju interakcije s njima ih vodio i pričao o odabranoj lokaciji. Za ulogu menadžera dobrodošlice potrebno je napraviti mapu prostora, implementirati sposobnost navigacije u prostoru i razviti model komunikacije robota s ljudima. Mapiranje i navigacija robota izvodi se u ROS razvojnoj okolini. U radu je analizirana sposobnost dobivanja 3D mapa prostora pomoću robotove dubinske kamere i analiziran je utjecaj načina mapiranja na kvalitetu mape. Testirana je sposobnost navigacije robotom korištenjem robotovih senzora u simulatoru, laboratorijskim uvjetima i realnom okruženju. Interakcija čovjeka i robota napravljena je u NAOqi Core razvojnoj okolini. Testirana je robotova sposobnost prepoznavanja pojedinih riječi prilikom takve interakcije u laboratorijskim i realnim uvjetima. Razvijena je aplikacija za robotov tablet čime se ostvaruje dodatan oblik komunikacije između robota i čovjeka. Opisan je i implementiran model stabla ponašanja kao metode koja služi za povezivanje svih gore navedenih potrebnih dijelova za funkcionalnog Peppera kao menadžera dobrodošlice.

Ključne riječi: stablo ponašanja, 3D mapiranje, menadžer dobrodošlice, Pepper robot, navigacija

Authors: Borna Zbodulja, Jakov Vulama, Petar Ljubotina

Pepper as a welcome manager at the Faculty of Electrical Engineering and Computing

Abstract

The aim of this paper is to prepare the humanoid robot Pepper as a welcome manager at the Faculty of Electrical Engineering and Computing who would welcome guests and, based on interaction with them, guide them and talk about the chosen location. For the role of welcome manager, it is necessary to make a map of space, implement the ability to navigate in space and develop a model of communication between robot and humans. Robot mapping and navigation is performed in the ROS development environment. The paper analyzes the ability to obtain 3D maps of space using a robotic depth sounder and analyzes the impact of mapping methods on the quality of the map. The ability to navigate the robot using his sensors was tested in the simulator, laboratory conditions and real environment. The human-robot interaction was made in the NAOqi Core development environment. The robot's ability to recognize individual words during such interaction in laboratory and real conditions was tested. An application for a robot tablet has been developed, which provides an additional form of communication between a robot and a human. A behavior tree model is described and implemented as a method used to connect all of the above required parts for a functional Pepper as a welcome manager.

Keywords: behavior tree, 3D mapping, welcome manager, Pepper robot, navigation