Model učenja robotskog zadatka zasnovan na interakciji s čovjekom

Vidaković, Josip

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

Josip Vidaković

MODEL UČENJA ROBOTSKOG ZADATKA ZASNOVAN NA INTERAKCIJI S ČOVJEKOM

DOKTORSKI RAD

Zagreb, 2020

Sveučilište u Zagrebu

Fakultet strojarstva i brodogradnje

Josip Vidaković

MODEL UČENJA ROBOTSKOG ZADATKA ZASNOVAN NA INTERAKCIJI S ČOVJEKOM

DOKTORSKI RAD

Mentor: prof. dr. sc. Bojan Jerbić

Zagreb, 2020

University of Zagreb

Faculty of Mechanical Engineering and Naval Architecture

Josip Vidaković

MODEL OF ROBOT TASK LEARNING BASED ON HUMAN-ROBOT INTERACTION

DOCTORAL DISSERTATION

Supervisor: prof. dr. sc. Bojan Jerbić

Zagreb, 2020

Podaci za bibliografsku karticu

UDK:	004.8 007.5
Ključne riječi:	robotika, planiranje kretanja, učenje iz demonstracija, optimiranje trajektorije
Znanstveno područje:	Tehničke znanosti
Znanstveno polje:	Strojarstvo
Institucija:	Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje
Mentor:	prof. dr. sc. Bojan Jerbić
Broj stranica:	133
Broj slika:	87
Broj tablica:	3
Broj korištenih bibliografskih jedinica:	148
Datum obrane:	06.10.2020
Povjerenstvo:	dr.sc. Mladen Crneković, red. prof. – predsjednik (Fakultet strojarstva i brodogradnje, Zagreb) dr.sc. Petar Ćurković, docent – član (Fakultet strojarstva i brodogradnje, Zagreb) dr. sc. Zdenko Kovačić, red. prof. – član (Fakultet elektrotehnike i računarstva, Zagreb)
Institucija u kojoj je rad pohranjen:	Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje

Podaci o mentoru

Bojan Jerbić rođen je 13. rujna 1957. godine u Zagrebu. Osnovnu školu i Matematičku gimnaziju pohađao je u Zagrebu. Na Fakultet strojarstva i brodogradnje, Sveučilišta u Zagrebu upisao se 1976 godine. Diplomirao je 25. veljače 1983. godine diplomskim radom: "Kinematička struktura Stanford manipulatora". Poslijediplomski studij pohađao je u razdoblju od 1984. do 1987. godine na FSB-u, smjer Tehnologija u strojarskoj proizvodnji, Projektiranje proizvodnih procesa. Magistarski rad pod naslovom "Istraživanje optimalnog redoslijeda sredstava za proizvodnju", obranio je 1987. godine te stekao naslov magistra tehničkih znanosti. Kao stipendista Florida State University boravi tijekom 1989. godine šest mjeseci na Department of Industrial Engineering, provodeći istraživanja u okviru izrade disertacije i znanstveno-stručnog usavršavanja. Disertaciju pod naslovom: "Interpretacija geometrije CAD modela u projektiranju automatske montaže ekspertnim sustavom", obranio je 1993. godine na Fakultetu strojarstva i brodogradnje, Sveučilišta u Zagrebu, te stekao naslov doktora tehničkih znanosti iz znanstvenog područja Strojarstvo. Od 1984. godine radi na Katedri za projektiranje proizvodnih procesa, Zavoda za tehnologiju, FSB-a, prvo kao pripravnik, a zatim kao stručni suradnik sve do 1986. godine, kada je izabran za asistenta. U zvanje znanstvenog asistenta izabran je 1988. godine. Zvanje docenta stekao je 1995. godine, izvanrednog profesora 2000. godine, a zvanje redovitog profesora 2005. godine. Od 1993. do 2006. godine voditelj je Laboratorija za projektiranje izradbenih i montažnih sustava. Dužnost predstojnika Zavoda za robotiku i automatizaciju proizvodnih sustava obnašao je od 2005. do 2008. godine. Na mjestu voditelja Katedre za projektiranje izradbenih i montažnih sustava je od 2007. godine do danas. Sudjeluje u izvođenju nastave iz predmeta: Automati za montažu, Projektiranje automatskih montažnih sustava, Umjetna inteligencija, Računalne mreže, Virtualno oblikovanje mehatroničkih sustava, Istodobno inženjerstvo, Programiranje automata za montažu, Vizijski sustavi, Inteligentni montažni sustavi i Integrirano inženjerstvo. Uveo je u nastavu brojne nove sadržaje, od primjene inženjerskih računalnih metoda u projektiranju do primijenjene robotike i umjetne inteligencije.

U znanstvenom radu posvetio se projektiranju izradbenih i montažnih sustava, poglavito razvoju računalnih metoda u projektiranju i metoda umjetne inteligencije u robotskoj montaži. U organiziranom znanstvenoistraživačkom radu sudjeluje od 1987. godine i autor je više od stotinu znanstvenih i stručnih radova. Bio je suradnik na sedam znanstvenih projekata te voditelj osam

domaćih i međunarodnih znanstvenih i tehnologijskih projekata. Osim u nastavnim i istraživačkim, aktivno sudjeluje u ostalim aktivnostima Fakulteta doprinoseći ukupnom razvoju znanstveno nastavnih djelatnosti. Od 1990. do 2006. godine član je Odbora za informatizaciju i kompjutorizaciju Fakulteta, a od 1997. godine djelovođa i potom član Povjerenstva za diplomske ispite Proizvodnog smjera, sve do 2010. Od 2009. do 2010. godine član je Odbora za poslijediplomske studije. Funkciju predsjednika Odbora za strategiju Fakulteta obnaša od 2010. do 2012. godine. Tijekom svoje karijere kontinuirano je radio na popularizaciji struke putem brojnih javnih nastupa u medijima te objavljujući mnoge popularizacijske članke. Održao je preko 30 pozvanih predavanja u zemlji i inozemstvu.

Zahvaljujem se mentoru prof.dr.sc. Bojanu Jerbiću na pruženoj prilici za ulazak u fascinantni svijet robotike, te kolegama Filipu, Marku i Bojanu na nesebičnom dijeljenju znanja i odličnoj radnoj atmosferi.

Zahvaljujem se svojim roditeljima na požrtvovnosti i podršci.

Hvala vam Ana i Leopold za strpljenje, ljubav i motivaciju za nastajanje ovog rada.

Sažetak

Robotsko učenje širok je pojam koji se može sagledati sa dva stajališta. Prvo je razina na kojoj se provodi učenje dok je drugo metodologija učenja. Razina učenja odnosi se na to usvaja li se vještina na razini pokreta (niža razina) ili na razini odabira pred-segmetiranih cjelina ponašanja na temelju stanja okoline (viša razina). Metodologija se u osnovi može podijeliti na samostalno učenje i učenje iz primjera (demonstracija). U radu je razvijena metodologija za robotsko učenje zadatka na razini kretanja, temeljeno na usvajanju znanja iz demonstracija potpomognuto kasnijim samostalnim učenjem. Prvo je razvijena metoda za analizu demonstracija dobivenih direktnim prostornim vođenjem robota u obavljanju zadatka. U sklopu ovoga predložena je nova metoda generiranja trajektorija u operacijskom sustavu robota sa mogućnošću aproksimacije prolaznih točaka i izbjegavanja stacionarnih prepreka. Eksperimentalnom validacijom potvrđena je valjanost razvijenog pristupa za generiranje trajektorija za nove slučajeve zadatka (konfiguracije). Nakon toga predložen je sustav za samostalno učenje temeljen na iterativnom optimizacijskom procesu pretraživanja parametarskog prostora trajektorije usmjeren prema ostvarivanju zadatka. Sustav je implementiran u simulacijskom okruženju te je validiran na dva različita zadatka. U zadnjem dijelu rada objedinjene su razvijene metodologije usvajanja znanja iz demonstracija i iterativnog učenja orijentiranog zadatku kako bi se predložio efikasan sustav učenja. Isti je validiran i provedena je usporedba u odnosu na slučaj gdje se koristi samo metoda iterativnog učenja.

Ključne riječi: robotika, planiranje kretanja, učenje iz demonstracija, optimiranje trajektorije.

Extended summary

Chapter 1: *Introduction*. This section is concerned with giving the research context of the thesis. It gives the motivation behind developing learning capabilities for technical systems, especially robotic systems. The one emphasized are the need for higher flexibility and autonomy of such systems in order to remove the barrier of highly specialized knowledge needed during the application of robots in the industry or in the service robotics field. The problem of task-oriented behavior is viewed from the programming and learning perspectives. Three common approaches for accomplishing such high-level behavior have been addressed: learning from demonstration, reinforcement learning and task-oriented motion planning. Current accomplishments in all three fields are referenced and specific advantages and disadvantages of each approach are covered. Based on this, the motivation for the research direction taken in this thesis is pointed out. The research hypothesis and goals are defined. On the end of this section, the structure of the thesis is given.

Chapter 2: *Learning from demonstration*. On the beginning of this section, a broader overview of learning from demonstration (LfD) specific problems is given. The correspondence problem is pointed out together with most frequent demonstration mechanisms. Methods for encoding demonstrated trajectories are covered in both the statistical modeling aspect and encoding based on dynamical systems. The DMP parametrization and its characteristics are covered in detail as it is used for trajectory encoding in other parts of the thesis. Two main approaches for encoding generalizability into learning from demonstration methods are covered – task parametrization and inverse reinforcement learning. A novel methodology for the analysis of demonstrations based on trajectories obtained by kinesthetic teaching is proposed and covered. The method uses a novel classification mechanism in order to determine attracting points, non-attracting points and obstacle points in the working environment of the robot. Experimental results of this methodology are presented and commented on the end of this section.

Chapter 3: *Task-oriented trajectory planning.* Demonstration sampling and analysis by the methodology from the previous section is performed in Cartesian space. In this section, task-oriented reproduction of trajectories in the same domain is performed. Common trajectory representations used in robotics that can be used both for planning in configuration space and operational space and their parametrizations are covered. As the thesis focuses on the application of primitive motions in task-oriented programming, this section gives an overview of the application of primitive motions in task-oriented scenarios. A modified DMP representation is presented which is capable of explicitly using the information obtained by the demonstration analysis. It has the capability of encoding variational information in the low level DMP trajectory definition and achieves this by introducing a modified time function instead of the standard exponential decay function. The methodology is originally presented in the conference paper: *Task Dependent Trajectory Learning from Multiple Demonstrations Using Movement Primitives.* After this, a Cartesian optimization-based path planning model is proposed, based on the following paper: *Learning from Demonstration Based on a Classification of Task Parameters and Trajectory Optimization*. The model is capable of encoding the information from the demonstration analysis by approximating identified via-points and avoiding identified obstacles. The path planning model is transferred into a DMP trajectory using the special DMP state representation presented earlier. The trajectory planning approach is verified on a presented experimental setup.

Chapter 4: *Reinforcement learning in continuous environments.* As models learned from demonstrations often fail to produce completely accurate task solution in the extrapolation phase, the idea of local trajectory improvement through self-exploration has been considered in this section. Reinforcement learning provides the general framework for achieving this. This section therefore covers the theoretical overview of RL, which provides the basis for explaining the methodology seen in the continuous space scenario. Policy search methods are identified as the most suitable when performing improvements on the trajectory level with continuous parametrization. Two main approaches for policy search are covered: critic-based approaches and “black-box” optimization (BBO). Both perform learning directly in the parameters space by observing and evaluating agent’s interactions with the environment. However, BBO approaches simplify the required evaluation mechanism while having comparable performance to critic-based approaches. Possible policy representations in the trajectory domain are covered in a special

subsection. The application of a BBO algorithm together with a DMP policy parametrization is demonstrated at the end of this section.

Chapter 5: *Iterative learning for stochastic tasks*. The BBO policy search methodology presented in the previous section implies the direct interaction of the agent (robot) with the environment. As searching in the parameter space of the trajectory policy in real environments is very dangerous and can lead to physical damages of both the robot and environment, a simulation setup is here introduced, suitable for robot learning. The setup is based on the ROS based physics simulator *Gazebo*. Based on this, a task-oriented iterative learning setup is proposed. At its core, the setup consists of black-box optimization which is given in the form of the evolutionary CMA-ES algorithm. The policy parametrization responsible for the execution of trajectories in the simulation environment is in the DMP form. The CMA-ES algorithm is responsible for updating the policy weight parameters with respect to a task-oriented cost-function. This closes the policy search loop which is performed in an iterative manner in order to achieve task learning convergence. The methodology was tested on two tasks: a peg-in-hole task and a sweeping task. Since the tasks showed high stochasticity with respect to the goal-oriented cost functions, two criteria to evaluate such learning processes where proposed. A best-current solution metric and a current average metric. The first one keeps track of the best solution achieved in the policy search process, while the later gives information about the overall quality of the learning process.

Chapter 6: *LfD as a basis for iterative learning*. The iterative learning algorithm presented in the previous section was initialized by an empirical strategy which used a linear trajectory. Previous research suggested that the search in big parameter spaces is very dependent on the initial conditions and exploration is mostly only locally oriented. In this section, results of the iterative learning algorithm are given, when initialized from demonstrations. The demonstration methodology followed the one presented in section two, which involved kinesthetic guidance for demonstration collection and the coordinate frame classification methodology for extracting useful via-points. The initial cartesian DMP trajectories where constructed using the optimization-based methodology from section three. The obtained results showed that the LfD initialization strategy lead to significantly better results in terms of the quality of the searched solutions as well as faster

convergence to applicable solutions. Findings presented in this section are based on the following paper: *Accelerating Robot Trajectory Learning for Stochastic Tasks*.

Chapter 7: *Conclusion* This section discusses the summary and the main achievements of the doctoral thesis. The main contributions can be viewed as: I) a novel learning from demonstration method for the analysis of trajectory level demonstrations, based on the classification of coordinate frames, II) an optimization-based cartesian trajectory planning algorithm with coordinate frame approximation and obstacle avoiding capabilities, III) a simulation based, iterative learning framework for task-oriented trajectory learning compatible with the LfD methodology. Future research will be focused on finding more efficient algorithms for policy search with sparse evaluation and testing the applicability of different policy representation. The possibilities for automatic estimation of exploration rates will be explored, as well as the automatic extraction of end-result-oriented cost/reward functions in order to remove the need for hand crafted functions.

Sadržaj

Popis slika

Popis tablica

1. Uvod

Suživot ljudi i tehničkih sustava postaje sve češći i kompleksniji, te se očituje u svim sferama društva, gdje isti doprinose značajnom porastu kvalitete života. Primjerice, ljudi u industrijskom okruženju sve manje obavljaju fizički teške ili jednolike poslove. Razvojem tehnologija sučelja i interakcije između čovjeka i strojeva (uređaja), značajni napreci ostvareni su u smjeru intuitivne i jednostavne primjene, što je jedan od razloga velike ekspanzije upotrebe određenih uređaja u svakodnevnom životu. Sukladno tome i očekivanja od tehničkih sustava generalno postaju sve veća. Sljedeći korak u ovom smjeru očituje se u sposobnosti sustava da predviđaju ponašanja i namjere korisnika te sve većoj mogućnosti samostalnog djelovanja (autonomnost). Značajnu ulogu u ovim unaprjeđenjima ima sposobnost učenja tehničkih sustava, koje predstavlja jednu od najkompleksnijih vještina koje stroj ili uređaj može imati i jedna je od osnovnih karakteristika inteligentnog ponašanja.

U biološkom svijetu učenje se definira kao složeni psihički proces promjene ponašanja na osnovu usvojenog znanja i iskustva. Obuhvaća usvajanje navika, informacija, znanja, vještina i sposobnosti s ciljem poboljšavanja djelovanja u određenoj domeni [1]. Kod ljudi je ovaj proces usko povezan sa motivacijom i vrijednostima koje iniciraju i pogone sam proces učenja.

Sa stajališta primjene, dva su temeljna motiva iza ideje učenja kod tehničkih sustava. Prvi je olakšavanje postupka programiranja u zahtjevnim zadacima, a drugi je učenje zbog nedostupnih informacija ili informacija koje se dinamički mijenjaju. Prvi motiv proizlazi iz problema da željeno ponašanje često nije moguće jednostavno prenijeti na tehnički sustav jer je kompleksnost kao i varijantnost situacija jako velika. Primjeri ovdje su autonomna vožnja, prepoznavanje rukopisa, prepoznavanje govora, prepoznavanje lica itd. Drugi motiv proizlazi iz činjenice da sustavi često moraju djelovati u okruženjima koja do tada nisu vidjela. Snalaženje robota za usisavanje u prostorima korisnika, rad servisnih robota u okruženjima korisnika, prilagođavanje djelovanja robotskih ruku u proizvodnji, snalaženje robota na udaljenom planetu itd.

Glavnu ulogu u razvoju metoda učenja kod umjetno stvorenih sustava ima grana strojnog učenja. Ona upotrebom statističkih metoda pokušava pronaći pravilnosti i značajke unutar dostupnih podataka korištenih za učenje, što za cilj ima stvoriti generalizirani model sposoban predvidjeti izlazne podatke za dotad ne viđene ulazne podatke. Upravo ova sposobnost predviđanja,

na osnovu prethodnog procesa učenja, ključna je karakteristika koja je zanimljiva za različite tehničke sustave, pa tako i robote. Područje učenja razvija se od 1980-ih godina, a neki od ranih primjera upotrebe strojnog učenja su: učenje autonomne vožnje ALVINN [2], zadatak obrnutog njihala [3], prepoznavanje rukopisa za potrebe klasifikacije poštanskih brojeva [4].

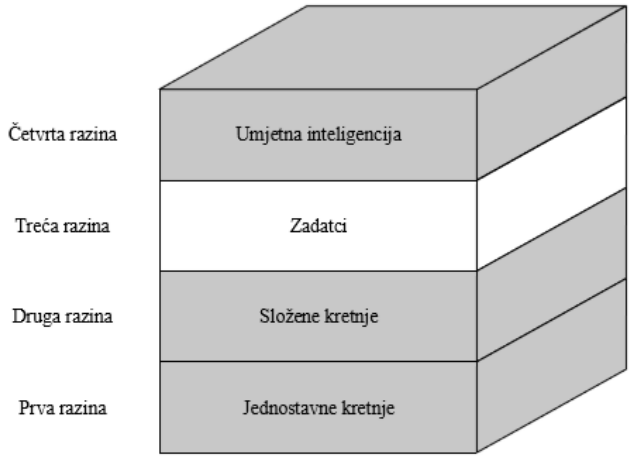
1.1. Učenje u robotici

Dok oba pristupa imaju isti cilj, a to je omogućiti samostalno izvršavanje nekog ponašanja, učenje i programiranje dva su komplementarna pojma u robotici – porast udjela jednog smanjuje potrebu za drugim. Programiranje je u osnovi orijentirano prijenosu informacija sa čovjeka na robota, a učenje ovu potrebu pokušava svesti na minimum. Kako programiranje još uvijek predstavlja temelj za usvajanje znanja u robotici, ovaj se pojam vrlo često koristi kao referentan u ovom području, te se govori o različitim razinama programiranja.

Programiranje se u osnovnom smislu u robotici koristi kako bi se definirale pozicije koje robot treba zauzeti te načine tranzicija između tih pozicija prilikom kretanja. Ovo predstavlja osnovni način programiranja robota. Korištenjem uvjeta unutar robotskih programa moguće je postići određenu varijantnost tj. sposobnost prilagođavanja različitim stanjima okoline. Ipak, porastom broja uvjeta u programu nastaju problemi prevelike kompleksnosti programa kao i nemogućnost predviđanja svih mogućih uvjeta i stanja koja se mogu pojaviti. Drugim riječima, klasičnim diskretnim metodama, procese nije moguće programski opisati tako da bi se postigla potpuna varijantnost i fleksibilnost sustava za neviđene slučajeve - moguće je to učiniti samo za jedan zatvoreni skup uvjeta.

Promatra li se programiranje robota sa četiri razine (slika 1-1.), prve dvije razine predstavljaju standardne industrijske načine programiranja koje ne daju nikakvu autonomiju robotskom sustavu jer se temelje na logici prvog reda. Treća razina (orijentiranost zadatku) zahtjeva primjenu modela koji su u stanju generalizirati znanje na druge situacije kako bi se uspješno riješio zadatak. Ovo za posljedicu ima autonomnost sustava koja ide do razine zadatka. Četvrtom razinom smatra se umjetna inteligencija. Nju odlikuje mogućnost samostalnog izvršavanja većeg broja zadataka u različitim konfiguracijama, ali i mogućnost odabira koji će se zadatak i kada izvršavati. Ovim slijedom, evoluirao je i razvoj robota kroz povijest. U početku su oni smatrani strojevima, a danas se smatraju fizičkom reinkarnacijom računala. Ovo je posljedica mijenjanja informacijske perspektive, gdje roboti više nisu potpuno ovisni o ljudskim

informacijama, nego putem vlastitih računala sposobni su donositi odluke i mijenjati svoje ponašanje. Iz perspektive učenja, treća i četvrta razina ponašanja najčešće zahtijevaju implementaciju ove sposobnosti.



Slika 1-1.Razine programiranja [5].

Razvoj u domeni računala, značajno utječe na robotiku u svim segmentima, a poglavito u području učenja i samostalnog odlučivanja. Sposobnost obrade velike količine podataka u kratkom vremenu ključna je za razvoj ovih pristupa. Ovaj razvoj popraćen je i razvojem senzorskih elemenata koji su nužni za skupljanje podataka iz okoline i kao takvi služe kao informacijski ulazi računala (vizijski sustavi, laserski sustavi, senzori sile i drugi). Tom trendu su značajno doprinijeli i kolaborativni roboti, koji su omogućili da se u djelovanje robota uključi sila kojom robot djeluje na okolinu.

Problematika učenja robota danas je znanstveno područje koje je na sjecištu područja strojnog učenja i robotike. Metode strojnog učenja omogućuju statističku obradu velikog broja informacija, pronalaženje karakterističnih pravilnosti unutar podataka. Robot s druge strane kao izvršni element naučenog znanja prima naredbe na niskoj upravljačkoj razini. Iz tog razloga je sve odluke donesene na visokoj razini uvijek potrebno svesti na nisku razinu upravljanja robotom.

Po uzoru na prirodne mehanizme opažajnog učenja (učenje oponašanjem) [6], [7] i operantnog uvjetovanja [8] razvila su se dva osnovna smjera za učenje u robotici: učenje iz demonstracija i samostalno učenje istraživanjem. Ideja učenja iz demonstracija je robotu predstaviti izvođenje zadatka putem primjera, nakon čega se od njega očekuje generalizacija znanja za različite slučaje (konfiguracije) zadatka. Ukoliko robot ne uspije obaviti zadatak u novim neviđenim situacijama, njegovo znanje se treba nadopuniti dodatnim demonstracijama. S druge

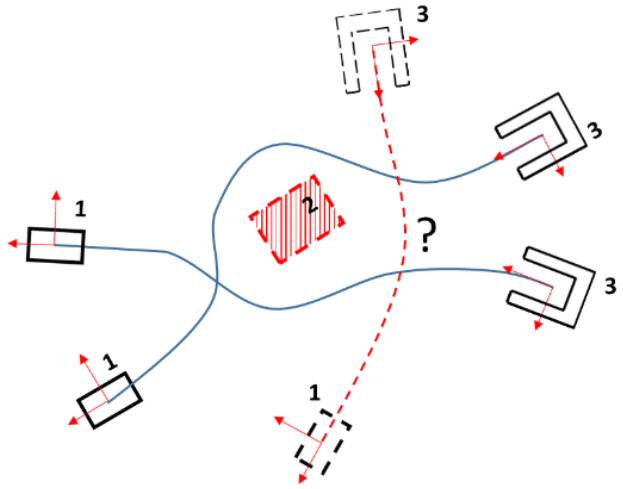
strane, učenje samostalnim istraživanjem ima za cilj kreirati robotsko znanje na temelju samostalnog istraživanja (metodom pokušaja i promašaja). Robot pri tome kombinira dvije strategije, istraživanje novih ponašanja (eng. *exploration*) i korištenje najboljih dosad viđenih ponašanja (eng. *exploitation*). Ukoliko nakon učenja, robot nije sposoban izvršiti zadatak u nekoj situaciji, model znanja treba nadopuniti daljnjim istraživanjem.

1.1.1. Razine djelovanja

Učenje u robotici moguće je ostvariti na dvije glavne razine. Jedna razina obuhvaća učenje kronološkog rasporeda primjene određenih pred definiranih akcija koje dovode do izvršavanja zadatka. Ovo je učenje na visokoj razini apstrakcije gdje se od agenta ne zahtjeva učenje parametara gibanja nego učenje plana djelovanja. Druga razina obuhvaća učenje izvođenja gibanja. Ovaj pristup obuhvaća učenje parametara gibanja na razini trajektorije gibanja. Tu spadaju parametri kao što su pozicija i brzina te se učenje obavlja u kartezijskom ili konfiguracijskom prostoru robota.

1.1.2. Generalizacija

Dva su osnovna načina logičkog zaključivanja kod čovjeka – induktivni i deduktivni. Koncept induktivnog zaključivanja temelji se na prethodno prikupljenim pojedinačnim iskustvima ili zaključcima o drugim temama. Isti je stoga temeljni princip generalizacije znanja, koja predstavlja mogućnost reprodukcije naučenih obrazaca u novim situacijama. Mogućnost generalizacije je kao takva jedna od glavnih odlika naučenog znanja.



Slika 1-2. Problem generalizacije na niskoj razini.

Generalizacija na razini gibanja tj. razini trajektorije u osnovnom obliku obuhvaća mogućnost izvođenja gibanja od proizvoljnih početnih do ciljnih pozicija robota. Proširenjima ovog problema na domenu zadatka, generalizacija je definirana i mogućnostima obavljanja zadatka u različitim konfiguracijama (slika 1-2).

Sa stajališta programiranja robota, ideja generalizacije pokušava se ostvariti uspostavom razvojnih okruženja u kojima su na modularan način dostupni programski elementi potrebni za programiranje različitih vrsta robota. Najpoznatije i najuspješnije razvojno okruženje u robotici je ROS (Robot Operating System) [9]. Ovo je platforma koja je razvijena s ciljem objedinjavanja istraživačkih programskih modula u robotici na jednom mjestu. Koristeći standardizirane komunikacijske kanale omogućena je jednostavna interakcija pojedinih modula međusobno. S druge strane, generalizacija na razini učenja pokušava se omogućiti dijeljenjem znanja između robota. Na globalnoj razini, vrlo važnu ulogu u ovom smjeru ima područje pod nazivom tzv. robotike u oblaku (eng. *cloud robotics*). *RoboEarth* [10] primjer je okruženja koje je namijenjeno za dijeljenje iskustva, informacija, upravljačkih algoritama među robotima. Činjenica da tvrtke kao *Google* i *Microsoft* trenutno ulaze u svijet *cloud* robotike, dovoljno govori o važnosti ovog koncepta.

1.1.3. Učenje oponašanjem

Mehanizam učenja oponašanjem koji ima vrlo značajnu ulogu kod ljudi, u robotici je implementiran u području pod nazivnom: učenje iz demonstracija (eng. *Learning from Demonstration* – LfD ili *Programming by Demonstration* – PbD). Ovaj oblik učenja bazira se na interakciji čovjeka i robota. Robot promatra izvođenje dobrih primjera ponašanja od strane čovjeka i bilježi dostupne senzorske podatke. Kasnijom obradom ovih primjera (demonstracija), najčešće korištenjem metoda strojnog učenja, robot pokušava prilagoditi svoje ponašanje kako bi dobio iste rezultate kao i učitelj.

U [11], definirana su ključni problemi u učenju iz demonstracija u obliku sljedećih pitanja:

- „Što imitirati?“.
- „Kako imitirati?“.
- „Kada imitirati?“.
- „Koga imitirati?“.

Jedan od glavnih koncepata učenja iz demonstracija je segmentacija zadataka na manje dijelove koji se mogu upotrebljavati u fazi analize demonstracija ali i sinteze ponašanja. Ovime se omogućuje cjelokupno učenje zadatka od niske razine učenja segmenata gibanja, do više razine koja je pogodna za učenje raznim statističkim metodama strojnog učenja.

Učenje iz demonstracija našlo je svoju primjenu u različitim vrstama zadataka. U industrijskim primjenama gdje su prisutni manipulatori u obliku robotskih ruku to su uglavnom operacije sklapanja [12], [13]. Neke drugi primjeri u toj domeni manipulatora obuhvaćaju učenje udaraca u stolnom tenisu [14] i valjanje tijesta [15]. U servisnoj robotici gdje je česta upotreba humanoidnih robota primjeri obuhvaćaju hodanje dvonožnog robota [16], točenje tekućina [17], sviranje bubnjeva [18] i kuhanje riže [19]. Druga linija istraživanja obuhvaća učenje suradničkih ponašanja s čovjekom. Primjer su operacije nošenja predmeta [20], sklapanje namještaja [21], pomoć kod oblačenja [22] i različiti oblici suradnje u kirurškim operacijama [23], [24].

Prvi problem ove metode učenja je što je ponašanje učenika (robota) uvjetovano ponašanjem učitelja, pa učenik nije u mogućnosti nadmašiti rezultate učitelja. Kako demonstracije predstavljaju vrijedne izvore informacija za algoritme učenja u ovom scenariju, postavlja se pitanje o broju potrebnih demonstracija, što je ujedno i drugi problem ovog pristupa. Poželjno bi bilo da algoritmi mogu učiti iz što manjeg broja demonstracija kako bi se smanjio potreban angažman ljudi. Treći problem očituje se u tome da je glavni motiv kod učenja iz demonstracija najčešće orijentiran na potpunu imitaciju učitelja, a za ocjenu kvalitete naučenog ponašanja ne postoji jednoznačna metrika i mora se odrediti eksplicitno za svaki slučaj posebno. Zbog svih navedenih razloga, gotovo da nema primjera koji čistim oponašanjem tj. učenjem iz demonstracija postižu izvođenje kompleksnih zadataka.

1.1.4. Učenje istraživanjem

Učenje istraživanjem je koncept koji podrazumijeva samostalno pretraživanje prostora i kontinuiranu evaluaciju ponašanja na osnovu postignutih rezultata. Ovaj mehanizam temelj je metoda podržanog učenja (eng. *Reinforcement Learning*). Za razliku od učenja iz demonstracija, gdje su podaci za učenje uglavnom unaprijed dostupni, ovdje iskustvo (informacije) dolazi iterativno te je ažuriranje ponašanja temeljna ideja. Kao takav, ovaj koncept nema ograničenja koja se javljaju kod demonstracija nego cilj učenja ovisi o samom postavljenom mehanizmu vrednovanja. Jedan od glavnih problema ovog pristupa je upravo u realizaciji sustava nagrađivanja

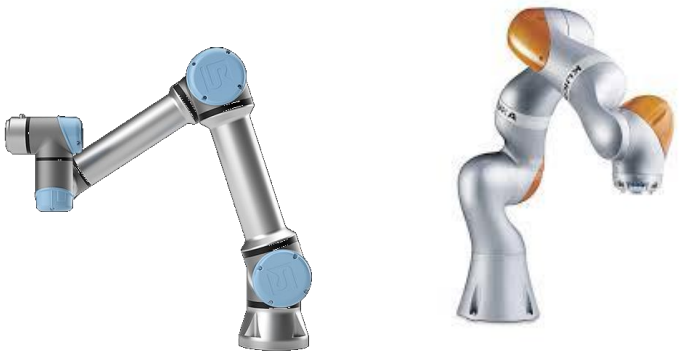
tj. vrednovanja ponašanja agenta s obzirom na postignuto ponašanje. Rješenje za ovaj problem pokušavaju dati metode inverznog učenja istraživanjem, koje funkcije cilja/nagrade pokušavaju automatski definirati na osnovu demonstracija poželjnog ponašanja. Ovi pristupi na neki način predstavljaju kombinaciju učenja iz demonstracija i učenja istraživanjem. Budući da se proces učenja istraživanjem provodi samostalnim pretraživanjem cijelog prostora stanja tj. prostora u kojem se agent (robot) može prilikom zadatka naći, ovaj postupak može biti jako vremenski zahtjevan. Učenje istraživanjem se pokazalo jako korisno i uspješno u računalnom svijetu, gdje je zbog porasta brzine računala moguće simulirati jako velik broj ponašanja agenta u kratkom vremenu. Jedan od najvećih uspjeha ovdje je *AlphaGo Zero* algoritam koji je sposoban naučiti *Go* igru bez ikakvog predznanja, na razini da pobjeđuje najbolje ljudske igrače [25]. Ipak kao i kod drugih „igara na ploči" (eng. *board games*), prostor akcija je ovdje diskretan. S druge strane, roboti djeluju u kontinuiranom prostoru stanja sa kontinuiranim prostorom akcija (zakreti zglobova), što značajno otežava problem učenja istraživanjem tj. podrškom zbog velike dimenzionalnosti problema (eng. *curse of dimensionality*).

Sa stajališta sigurnosti, slobodno učenje pravog robota u realnom okruženju može biti prilično opasno i za okolinu ali i za samog robota. Zbog toga je ono još uvijek ograničeno na strogo kontrolirana okruženja (najčešće laboratorijska). U [26], prikazano je učenje strategija hvatanja na temelju velike količine podataka dobivenim paralelnom upotrebom 14 robota. Manipulacija objektima antropomorfnom hvataljkom dana je u [27].

1.2. Interakcija čovjeka i robota

Mogućnosti robotskog djelovanja postaju sve sličnije ljudskima. Roboti se mogu kretati, pričati, slušati, gledati, osjetiti dodire i sile koje djeluju na njih. Robotika se polako odmiče od pretpostavke da s istima mogu raditi samo vrlo školovani pojedinci koji su specijalizirani u tom području. Područje interakcije čovjeka i robota (eng. *human-robot interaction* - HRI) bavi se istraživanjima intuitivnih načina komunikacije između ove dvije strane. Razvoj senzorskih tehnologija pridonijeo je značajnom razvoju ovog područja. Ovdje su značajni vizijski sustavi koji omogućuju praćenje stanja okoline visokom frekvencijom i razvoj teleoperacijskih uređaja sa povratnom vezom, koji omogućuju prijenos sila sa samog robota do čovjeka, te senzora sila. Ipak, vjerojatno najznačajniju ulogu u razvoju ovog područja uzrokovao je razvoj percepcije dodira. Predloženi su razni taktilni senzori koji prekrivaju površinu robota [28] i omogućuju detekciju dodira bilo koje vrste. S druge

strane, kolaborativni roboti (slika 1-3) integrirali su mogućnost percepcije vanjskih sila koje djeluju na robota na posve novi način [29]. U svakom od zglobova ovakvih robota u realnom vremenu mjere se momenti koji utječu na robotsku ruku. Razdvajanjem ovih momenata od momenata nužnih za gibanje robota, postignuta je mogućnost vrlo točnog mjerenja vanjskih sila koje utječu na robota [30]. Ovo je omogućilo razvoj cijelog niza novih upravljačkih paradigmi u HRI kontekstu.



Slika 1-3. Kolaborativni roboti UR5[31] i Kuka LBR iiwa [32].

Sa stajališta učenja, uloga HRI značajna je u oba predstavljena koncepta učenja. Kod učenja iz demonstracija vrlo je bitan način skupljanja podataka. Tri su osnovna načina: vizualnim promatranjem, kinestetičkim učenjem i teleoperacijom, koji će biti detaljnije obrađeni u sklopu drugog poglavlja. Sve su zanimljivije interaktivne metode učenja, gdje roboti kroz različite intuitivne mehanizme traže informacije koje im nedostaju od ljudi [33], [34]. Neki od primjera su glasovno komuniciranje robota sa čovjekom gdje robot ima priliku postavljati pitanja čovjeku [35]. Učenje istraživanjem samo je po sebi područje koje podrazumijeva interakciju robota s okolinom u kojoj se nalazi. Interaktivni mehanizmi služe ili za inicijalizaciju početnih uvjeta od strane čovjeka ili za prikupljanje podataka iz okoline tijekom učenja, bilo da u njoj sudjeluje čovjek ili ne.

1.3. Planiranje kretanja u kontekstu učenja

U robotici, planiranje kretanja područje je koje obuhvaća probleme kao što su planiranje putanje (eng. *path planning*), planiranje trajektorije (eng. *trajectory planning*), planiranje zadatka (eng. *task planning*). Planiranje putanje bavi problemom pronalaska putanje od neke početne pozicije do cilja („od točke A do točke B“) pri čemu se izbjegavaju prepreke. Planiranje trajektorije problem

je kretanja koji nadograđuje problem planiranja putanje na način da u njega uvodi parametre kao što su vrijeme, brzine, akceleracije, kinematika, kontinuitet, manipulabilnost [36] itd. Kod modeliranja trajektorija ovdje su jako zastupljeni geometrijski pristupi modeliranju koji predstavljaju bazu kretanja zbog svojeg kontinuiteta. Također, optimizacija trajektorija u odnosu na različite parametre tradicionalno je aktivno područje istraživanja [37], [38]. Učenje iz demonstracija i planiranje kretanja, vrlo su kompatibilni pristupi, gdje demonstracije mogu služiti za pronalazak kriterija u odnosu na koje je potrebno optimirati trajektoriju. Ova hipoteza korištena je u nekolicini radova [39], [40], [41], [42]. Ovisno sa kojeg stajališta se problem promatra, ovi kriteriji mogu biti usmjereni na optimiranje putanje s ciljem poboljšavanja kinematskih/dinamičkih performansi ili na optimiranje s ciljem izvršavana zadatka. Tendencija novijih pristupa ide prema tome da se oba cilja kombiniraju za što kvalitetnije i samostalnije izvođenje zadataka.

Prednosti pristupa koji se temelje na planiranju kretanja u odnosu na učenje iz demonstracija je prilagodljivost plana. Ovo je jako bitno u okolnostima u kojima dolazi do dinamičke promjene okoline tijekom izvođenja trajektorije, gdje ovi pristupi često dozvoljavaju ponovno planiranje u realnom vremenu. Općenito, generalizacijske sposobnosti metoda za planiranje kretanja vrlo su napredne u domeni kretanja, jer iste moraju omogućiti kretanje na osnovu proizvoljnih početnih i krajnjih pozicija. S druge strane, nedostatak ovih pristupa je često nemogućnost implementacije različitih kriterija koji su bitni za izvršavanje zadataka, gdje prednost imaju metode koje koriste učenje iz demonstracija.

1.4. Motivacija

U kontekstu generalizacije znanja na razini zadatka, pristupi učenja iz demonstracija često su vrlo ograničeni i ostaju orijentirani vrlo pojednostavljenim primjerima zadataka. S druge strane, isti imaju velik potencijal u smislu generalizacijskih sposobnosti na neviđene slučajeve. Pristupi učenja istraživanjem prirodnije su orijentirani izvršavanju zadatka koji mogu biti i vrlo kompleksni ali im često nedostaje mogućnost generalizacije za nove potpuno drugačije konfiguracije zadatka. Ovdje se otvara mogućnost za kombiniranje ovih dvaju pristupa, s ciljem iskorištavanja pozitivnih karakteristika jednog i drugog pristupa. Istraživanja [43], [44], [45], [46] koja se bave kombinacijom ovih dvaju pristupa potvrđuju potencijal ovakve metodologije ali ostavljaju veliki prostor za napredak i razvoj.

1.5. Hipoteza i doprinosi

Cilj istraživanja je razvoj modela za učenje robota pokazivanjem putem interakcije čovjeka i robota. Model treba biti usmjeren prema generiranju plana gibanja robota usklađenog postavljenom zadatku. Temeljem ograničenog broja primjera potrebno je osigurati generalizaciju znanja koje će omogućiti rješavanje sličnih problema.

Na temelju cilja, postavljene su sljedeće dvije hipoteze istraživanja:

- Učenje robota za obavljanje zadataka u pretpostavljenoj domeni problema moguće je ostvariti putem ograničenog broja primjera prezentiranih interakcijom čovjeka i robota.
- Generalizacija naučenog znanja može se postići statističkom obradom i višekriterijskom optimizacijom specifičnih značajki koji opisuju zadatak.

Očekivani doprinosi:

- Model učenja pokazivanjem izvršavanja robotskog zadatka putem ograničenog broja primjera prezentiranih interakcijom čovjeka i robota, zasnovan na sintezi metoda dinamičkog oblikovanja gibanja pomoću primitiva i inverznog podržanog učenja.
- Algoritam za određivanje i vrednovanje specifičnih značajki koje opisuju zadatak iz pretpostavljene domene problema temeljen na statističkoj analizi i višekriterijskoj optimizaciji u vremenski zavisnom prostoru problema s ciljem generalizacije naučenog znanja.

1.6. Struktura rada

Prvo poglavlje: U uvodnom poglavlju predstavljeno je područje robotskog učenja i motivi vezani za nastanak samog područja. Područje istraživanja samog rada smješta se u širi kontekst programiranja robota orijentiranog zadatku. Opisane su osnovne metode robotskog učenja kao i osnovni problemi. Postavljen je osnovni cilj istraživanja i njegove hipoteze.

Drugo poglavlje: U ovom poglavlju daje se pregled metoda za učenje iz demonstracija. Obrađuje se problem korespondencije kod prijenosa znanja s čovjeka na robota. Na kraju ovog poglavlja daje se opis predložene metode za učenje iz demonstracija te su prikazani rezultati analize radnog prostora realnog robota na temelju ove metodologije.

Treće poglavlje: U trećem poglavlju obrađuje se problematika učenja na razini trajektorije. Predložena je sprega između predložene metode za analizu demonstracija i robotskih trajektorija. U drugom dijelu ovog poglavlja predložena je nova vrsta robotske trajektorije sa svojstvom izbjegavanja prepreka i kompatibilna sa metodologijom učenja iz demonstracija.

Četvrto poglavlje: Ovdje se daje kratak teoretski uvid u metodologiju podržanog učenja sa naglaskom na učenje u kontinuiranim okruženjima kakva su uobičajena u robotici. Opisani su svi koncepti bitni za pretraživanje strategija koje je najznačajniji oblik podržanog učenja. Prikazan je primjer implementacije BB pretraživanja strategija sa dvodimenzionalnim DMP modelom za jednostavniji scenarij prolaska kroz prostornu poziciju. Na kraju poglavlja daje se analiza prednosti i nedostataka svakog od pristupa.

Peto poglavlje: Teoretska osnova i implementacija predstavljena u prethodnom poglavlju, služi kao osnova za iterativni koncept učenja zadatka na izvršnoj razini trajektorije, koji je predstavljen u ovom poglavlju. Predstavljen je koncept koji se bazira na učenju kroz direktnu interakciju sa simuliranom okolinom. Koncept je testiran na dva scenarija od kojih je jedan zadatak umetanja, a drugi zadatak odguravanja, te je pokazao konvergenciju procesa učenja uspješnom obavljanju zadataka.

Šesto poglavlje: Analiza demonstracija u odnosu na prostorne koordinatne sustave koja je predstavljena u drugom poglavlju se ovdje koristi za generiranje inicijalne trajektorije za iterativni proces učenja. Dobiveni rezultati su pokazali da ovakav način inicijalizacije značajno ubrzava proces učenja i povećava kvalitetu pretraživanih rješenja.

Sedmo poglavlje: Na kraju rada dana su zaključna razmatranja, smjernice za buduća istraživanja, kao i znanstveni doprinosi doktorskog rada.

2. Učenje iz demonstracija

Motivacijski faktori koji vode učenju se uglavnom mogu podijeliti na trenutne i vidljive (koji su uzrokovani uvjetovanim nagrađivanjem ili kažnjavanjem (eng. *conditioning*)), te dugoročne i nevidljive, čiji uzrok nije vidljiv na prvi pogled. Ovi faktori mogu bit različitog tipa i značajno ovise o kontekstu u kojem se pojedinac nalazi. Čest način prijenosa znanja prema dugoročno motiviranim pojedincima su demonstracije.

Demonstracije predstavljaju informacijske „pakete“ kojima je moguće prenijeti znanje sa stručne osobe (učitelja) na onoga koji želi ovladati istim znanjem (učenika). Kod ljudi se ovo najčešće događa tako da učitelj fizičkim demonstracijama i/ili govorom izlaže neko znanje ili vještinu, a učenici svojim osjetilima prate izložene informacije. Dodatno, nakon demonstracija, moguće je da učenik samostalno ili uz pomoć učitelja ponavlja demonstrirane vještine. Ovaj model se pokazao kao vrlo učinkovit jer neposredno nakon demonstracija učenik ima najviše informacija iz viđenih demonstracija i sposoban ih je iz kratkoročne memorije odmah upotrijebiti.

Učenje iz demonstracija zanimljiv je koncept i u robotici jer demonstracije teoretski mogu zamijeniti uobičajene načine programiranja ako robot može iz njih izvući dovoljno informacija. Problematika učenja iz demonstracija može se podijeliti na dva dijela. Prikupljanje i strukturiranje podataka za učenje je prvi od njih. Ovaj problem nastaje zbog toga što su demonstracije najčešće informacijski zapisi sa vrlo velikom gustoćom podataka (eng. *information density*) te njihova pohrana u računalno prihvatljivom obliku može biti vrlo kompleksna, ukoliko se želi zadržati ista razina informacija. Drugi problem se odnosi na sami proces učenja koji je najčešće u obliku statističke obrade prikupljenih podataka. Iako demonstracije mogu u sebi sadržavati jako puno podataka po demonstraciji, statističke metode se baziraju na obradi jako puno primjera. Ovo bi značilo da je potreban veliki broj demonstracija, što nije poželjna karakteristika kod ovog načina učenja. Iz tog razloga je ovo područje zanimljivo za razvoj efikasnih metoda učenja.

Ovdje se najčešće javljaju dvije vrste modela kojima se postiže ekstrakcija informacija iz demonstracija i njihovo kodiranje u željeni oblik ponašanja. Prvo su modeli kao funkcije ponašanja gdje se vrlo često koristi modeliranje u ovisnosti o zadanim parametrima zadatka. Drugo su modeli u obliku funkcija nagrada koji se pokušavaju naučiti iz demonstracija. Oba pristupa biti će opisana u nastavku.

2.1. Načini demonstriranja

Kao jedan od preduvjeta učenja javlja se prijenos informacija. Između ljudi obavlja se na tri moguća načina. Govorom, gestama ili fizičkom interakcijom (dodirom). Čovjek je svojim osjetilima sposoban interpretirati svaki od ovih ulaznih podražaja, strukturirati ih i koristiti za stjecanje iskustva. Može se pretpostaviti da je u biološkom svijetu usvajanje i upotreba prenesenih informacija najjednostavnija između jedinki iste vrste tj. sa jednakim fizičkim osobinama. Razlog tome je mogućnost izravnog korištenja informacija, bez potrebe za pretvorbama sa jedne razine na drugu. Ako se demonstrator ili učitelj značajno razlikuje od jedinke koja treba naučiti neki zadatak, nužno mora doći do nekog oblika konverzije informacija iz jedne domene jedinke u drugu. Ovo je posebno vidljivo kod prenošenja informacija za obavljanje zadataka koji uključuju korištenje motoričkih sposobnosti. Ovaj problem se naziva problem korespondencije (eng. *correspondence problem*) [47] i u robotici u području učenja iz demonstracija je on vrlo značajan jer roboti u pravilu raspolažu sa znatno manjim senzorskim i motoričkim sposobnostima nego čovjek (slika 2-1.).

Slika 2-1. Razlike u fizičkim karakteristikama robota s obzirom na broj stupnjeva slobode.

Tri su glavna načina za demonstriranje ponašanja robotu: vanjske demonstracije, teleoperacija i kinestetičko učenje [48].

Kinestetičko učenje: ovo je metoda kod koje čovjek fizički pridržava robota i pokreće ga na način da robot obavlja željeni zadatak. Kako bi se omogućila ova funkcionalnost, roboti moraju često biti opremljeni sa kontrolerom ili mehanizmom za kompenzaciju gravitacije kako čovjek ne bi morao nositi cijelu težinu robota prilikom demonstriranja. Ova funkcionalnost je najčešće dostupna samo za robote manje veličine, a nedostatak ovog pristupa očituje se i kod robota sa velikim brojem stupnjeva slobode, kada je zadatak fizičkog upravljanja svakim zglobom robota tijekom gibanja ovakvog robota kroz prostor vrlo težak. U ovom slučaju se umjesto pokazivanja

dugih pokreta često koristi pokazivanje manjih cjelina zasebno [49]. Problem korespondencije je kod ovog načina učenja minimalan, zbog toga što se demonstracije izvode izravno u operacijskom prostoru robota.

Teleoperacija: izvođenje demonstracija korištenjem teleoperacije provodi se indirektno, preko kontrolera koji je sposoban prenijeti ljudske pokrete na robota. Ovo se obično izvodi putem konzolnog upravljačkog uređaja (eng. *joystick*) ili putem neke druge vrste haptičkog uređaja. Velika prednost ove metode je da onaj koji izvodi demonstracije ne mora biti fizički kraj samog robota, nego može biti na nekoj udaljenoj lokaciji. Nedostatak ovog pristupa očituje se u neizravnom kontaktu čovjeka sa radnom okolinom robota. Drugim riječima, vrlo je teško na pouzdan način prenijeti povratne informacije sa robota na čovjeka. Kamere se koriste kako bi se čovjeku prenijela vizualna informacija, a uređaji sa haptičkom povratnom vezom putem vibracija i varijabilnom krutosti tipki pokušavaju demonstratoru pružiti informacije o silama s kojima robot djeluje na okolinu. Problem korespondencije kod ove metode također nije izražen iz razloga što robot demonstracije izvodi u svom operacijskom prostoru. Osim kod aplikacija učenja, teleoperacija se često koristi u medicinskoj robotici kako bi se vještine kirurga unaprijedile i postiglo lakše izvođenje operacijskih zahvata [50].

Slika 2-2. Upravljanje robotom teleoperacijom [51].

Obzervacijsko učenje: učenje putem obzervacija učitelja najkompleksniji je oblik učenja. Najčešći cilj ovog mehanizma učenja je mapiranje stanja i akcija iz vizualnih demonstracija. Snimanje demonstracija najčešće se provodi upotrebom vizijskih sustava, dok je u usporedbi sa prethodna dva mehanizma demonstriranja problem korespondencije najizraženiji, budući da je kinematiku učitelja potrebno prevesti u kinematski prostor učenika tj. robota.

2.2. Simboličko učenje – učenje na višoj razini

Konačnim ciljem učenja u robotici može se smatrati generalizacija složenih zadataka koji se sastoje od većeg broja jednostavnijih gibanja/djelovanja (eng. *primitives* – "primitivi"). Uobičajeni način učenja ovakvih zadataka je učiti svaki od ovih manjih dijelova zasebno i onda ih kombinirati kako bi se dobilo ponašanje koje rezultira izvršavanjem zadatka.

Usvajanje malih dijelova ponašanja na izvršnoj razini smatra se učenjem na niskoj razini. U slučaju robota ovo se uglavnom odnosi na razinu kretanja dok u kontekstu strojnog učenja to može biti učenje riječi, čitanje slova, prepoznavanje lica itd. S druge strane, učenje koje se odvija na razini odabira naučenih ili programiranih segmenata ponašanja smatra se učenjem na višoj razini djelovanja. Treći i najkompleksniji scenarij je onaj u kojem je cilj iz demonstracija ponašanja niske razine naučiti i modele niske razine kao i modele više razine ponašanja. Strategija koja se ovdje koristi je da se demonstracije pokušavaju automatski segmentirati na manje dijelove, te se onda isti pokušavaju zasebno naučiti na niskoj razini, dok se istovremeno modelira njihov redoslijed pojavljivanja u demonstracijama.

Segmentacija demonstracija je proces dijeljenja demonstracija na manje dijelove koji su definirani na osnovu nekog kriterija npr. prema funkciji, vremenu, obliku kretanja itd. Ako su segmenti ponašanja unaprijed poznati od ranije ili postoji neka vrsta predznanja, proces je okrenut prema klasifikaciji dijelova demonstriranog ponašanja. Ukoliko „primitivi" nisu unaprijed poznati onda postoji potreba za automatskom segmentacijom, gdje sustav treba sam na osnovu nekih kriterija obaviti podjelu demonstracije na određene dijelove. Ovo je vrlo izazovan zadatak jer za jednu vrstu demonstracija može postojati nekoliko strategija za segmentaciju. Pristup za segmentaciju pokreta ljudskog tijela dan je u [52]. Primjer cjelovitog pristupa iz demonstracija, sa automatskom segmentacijom pokreta, učenjem pokreta i višom razinom učenja pomoću HMM modela dan je u [53], gdje je testni scenarij bio valjanje tijesta.

Učenje na višoj razini ne odnosi se samo na učenje redoslijeda izvršavanja primitivnih ponašanja nego i na planiranje korištenja primitivnih dijelova djelovanja (akcija) u ovisnosti o stanju u kojem se okolina i agent nalaze (prostor stanja). Pri tome se uzima u obzir stanje u kojem će agent (robot) završiti i koliko je ono dobro. Proces učenja kod ovakvih modela sastoji se od statističke obrade viđenih akcija poduzetih u određenim stanjima. Pri tome se koriste metode klasifikacije i klasteriranja (grupiranja). Jedan od često korištenih vjerojatnostnih modela su skriveni Markovljevi modeli (eng. *Hidden Markov Models* – HMM) koji su često korišteni za

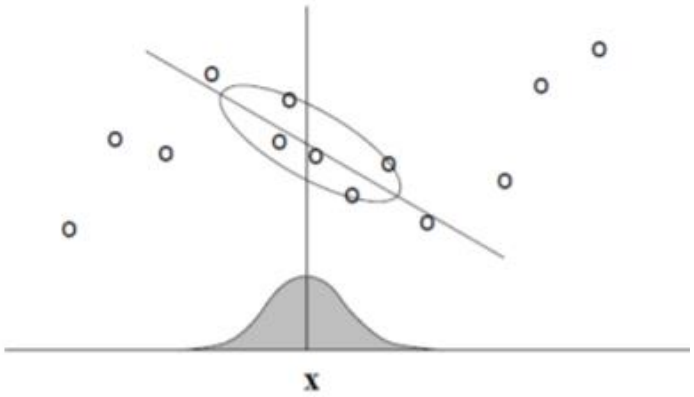
modeliranje sekvence radnji na temelju demonstracija. Još jedan popularni vjerojatnosni model koji se koristi za klasifikaciju akcija za određena stanja su Gaussovi miješani modeli (GMM). U [54] su isti korišteni kako bi se naučila vožnja simuliranog automobila pomoću diskretnih unaprijed definiranih naredbi za skretanje. U sklopu učenja na najvišoj razini izražen je i pristup simboličkog učenja ili planiranja (eng. *symbolic planning*). U [55] je korišten ovaj način zapisa, a učenje je svedeno na ekstrakciju simboličkih pravila za manipulaciju objektima. Drugi primjer je učenje kompleksnih zadataka na visokoj razini poput postavljanja stola za objedovanje ili slaganje kutija. Još jedan primjer simboličkog planiranja dan je [56]. Simbolički opis zadatka izuzimanja i ostavljanja (eng. *pick and place*) zadatka predložen je u [57].

2.3. Statističko modeliranje trajektorija

Statističko modeliranje trajektorija kod učenja iz demonstracija podrazumijeva predstavljanje određenog broja demonstriranih trajektorija nekim generalnim modelom. U osnovnoj varijanti ovo se odnosi na učenje na niskoj razini gdje je cilj modeliranje robotskih upravljačkih varijabli u odnosu na neku ulaznu varijablu. Ova ulazna varijabla obično je vrijeme ili neka druga varijabla koja je posredno ovisna o vremenu. Naučeni modeli trajektorija najčešće se mogu izvršavati direktno na robotu, obično korištenjem kontrolera pozicije, brzine ili momenta. U sljedećem poglavlju će se opisati osnovni regresijski i klasifikacijski algoritmi za modeliranje (učenje) trajektorija.

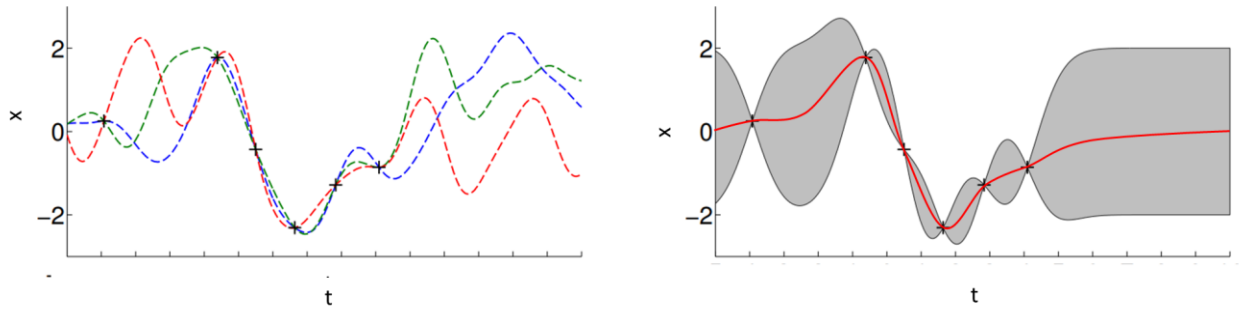
2.3.1. Regresijske metode

Regresijske metode općenito služe za aproksimaciju funkcija za prethodno neviđena stanja. Najpopularniji regresijski algoritmi za modeliranje trajektorija su iz grupe „lokalno otežanog učenja“ (*eng. Locally weighted learning* – LWL) [58], [59]. Standardne regresijske metode pokušavaju prilagoditi parametrizirane funkcije nekom cijelom setu podataka na temelju kriterija poput najmanjeg kvadrata. Ovo je obično jako zahtjevno sa stajališta računalnih resursa. LWL s druge strane pokušava prilagoditi (naučiti) manje parametrizirane modele na manje regije podataka, odnosno „lokalne“ regije podataka. Ovo za posljedicu ima efikasnije i točnije učenje. Iz ove familije algoritama najpoznatiji su LWR (*eng. Locally weighted regression*) i LWPR (eng. *Locally Weighted Projection Regression*) [60]. Za razliku od LWR-a, LWPR metoda ne zahtjeva da su podaci za učenje unaprijed dostupni (spremljeni u memoriju).



Slika 2-3. LWL koristi kernel funkciju kako bi otežao podatke za učenje ovisno o udaljenosti od trenutne lokalne ulazne vrijednosti X [60].

GPR (eng. *Gaussian process regression*) je često korišteni regresijski algoritam za modeliranje trajektorija. Temelj ovog pristupa su Gaussovi procesi (GP) koji predstavljaju učene modele. GP su u osnovi setovi random varijabli, gdje svaka skupina istih za sebe predstavlja Gaussovu distribuciju [61]. GP je sastavljen od srednjih vrijednosti i funkcija kovarijance. Učenje ulazno-izlaznih korelacija ovakih modela odvija se putem GPR algoritma. Gaussovi procesi su zanimljivi jer u model uključuju vjerojatnost (sigurnost) određenog rješenja. Ovo se očituje tako da je vjerojatnost viđenih ulazno-izlaznih parova jako velika (varijanca je jako mala), dok je vjerojatnost neviđenih ulazno-izlaznih parova manja, i smanjuje s udaljenosti od viđenog para (varijanca raste). Prirodno, veći broj viđenih stanja rezultira kvalitetnijim modelom.



Slika 2-4. Demonstrirane trajektorije (lijevo) i model dobiven GPR-om (desno). Crvena linija u desnom grafu je srednja vrijednost funkcije, dok siva regija predstavlja varijancu.

U usporedbi s LWL pristupima, računalna kompleksnost GPR algoritma je veća. U [62], je predložen lokalniji pristup GPR-a kako bi se ponudilo rješenje za ovaj problem. Ova metoda

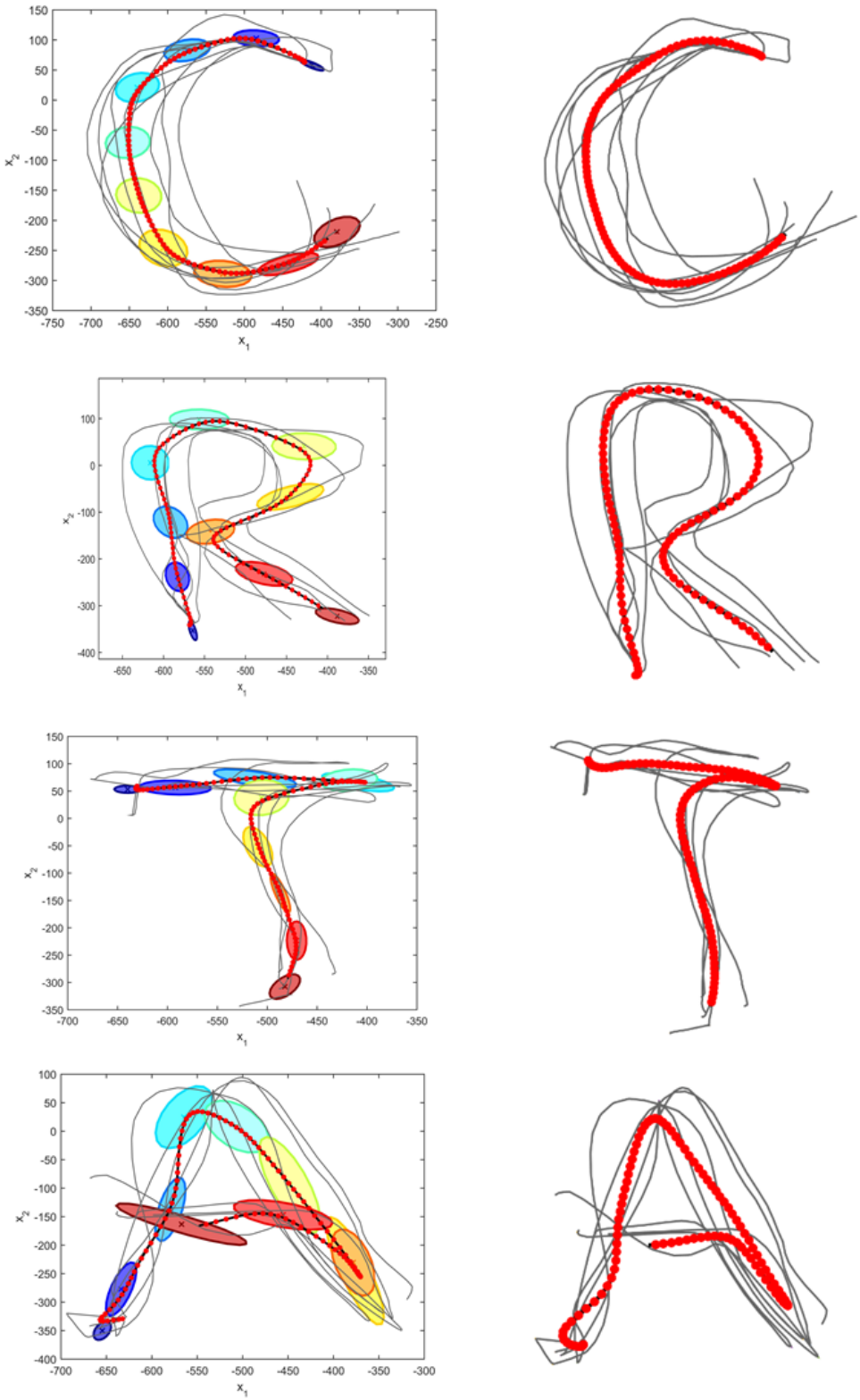
modeliranja osim za učenje trajektorija, primjenjiva je u mnogim drugim aplikacijama. U [63] je prikazana aplikacija GP modela za planiranje gibanja.

Neuronske mreže (eng. *Neural Networks* – NN) su isto tako često korišteni regresijski modeli kod učenja iz demonstracija. Za razliku od LWL i GP pristupa, NN ne trebaju imati unaprijed dostupne sve podatke za učenje. Svojstva neuronskih mreža omogućuju da se učenje provodi svakom pojavom novih podataka, tako da nije potrebna memorija u kojoj će se čuvati svi podaci. Primjer korištenja NN za zadatak učenja zadatka umetanja (eng. *peg-in-hole*) na osnovu demonstracija prikazan je u [64]. Ipak, neuronske mreže se više koriste za opisivanje strategija kod podržanog učenja, nego za direktno vremenski zavisno modeliranje trajektorija.

2.3.2. Klasifikacijske metode

Klasifikacijski modeli kategoriziraju ulazne podatke u diskretne unaprijed definirane klase. Najučestaliji ovakav model kod učenja trajektorija su Gaussovi miješani modeli (GMM). GMM je model koji se sastoji od diskretnog broja normalnih distribucija, koje se „fitaju“ na podatke za učenje. Proces učenja GMM-a sastoji se od iterativnog mijenjanja parametara normalnih distribucija (srednje vrijednosti i varijanca) kako bi se dobila što bolja prilagodba modela na podatke. Nedostatak ovog pristupa je taj što je broj distribucija podatak tzv. otvoreni parametar koji se mora postaviti prije učenja. Ovo se može učiniti ili ručno ili korištenjem Bayesovog informacijskog kriterija (*engl. Bayesian information criterion* – BIC) [65]. Učenje modela se obično provodi EM algoritmom (eng. *Expectation-maximization*) [66]. Nakon učenja, GMM modeli se često kombiniraju sa GMR (eng. *Gaussian Mixture Regression*) regresijskom metodom kako bi se dobile izlazne vrijednosti za dani ulaz. U tom slučaju se izlaz računa kao linearna kombinacija uvjetovanih vjerojatnosti svakog od pojedinačnih Gaussovih modela. Primjer korištenja GMR pristupa za modeliranje trajektorije dan je u [67].

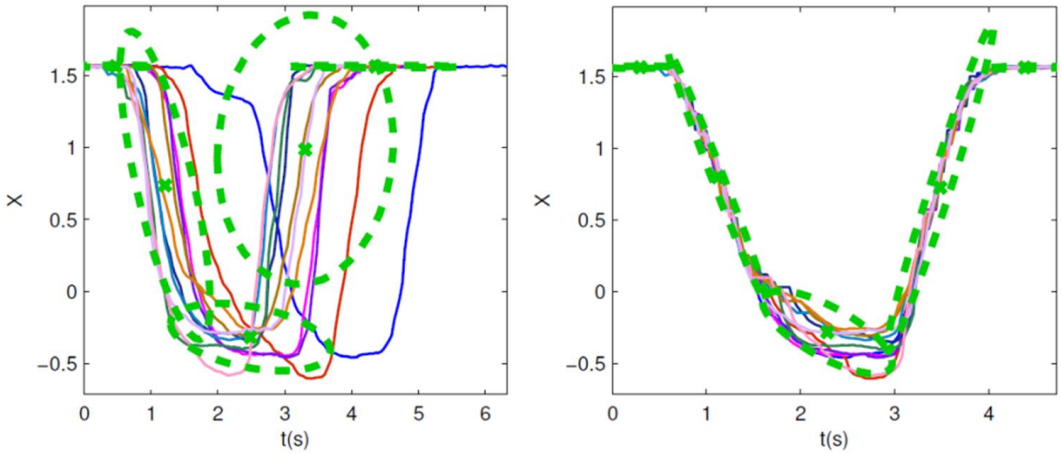
Na slici 2-5. prikazan je primjer upotrebe GMM-a za opisivanje dvodimenzijskih prostornih krivulja. Korišten je model sa 10 normalnih distribucija. Vidljivo je da se svaka od distribucija može poistovjetiti sa poljima koja privlače samu trajektoriju. Generalizirana krivulja je u ovom slučaju dobivena svojevrsnom aproksimacijom stanja koja su dana u obliku položaja centara normalnih distribucija.



Slika 2-5. Generalizacija krivulje pomoću GMM-a.

2.3.3. Poravnavanje trajektorija

Demonstrirane trajektorije često mogu biti demonstrirane u različitim vremenskim okvirima, tj. sa različitim vremenom trajanja i u različitim vremenskim trenutcima. Jedan od osnovnih zahtjeva za statističko učenje trajektorija je da podaci iz kojih se uči budu sortirani na pravi način. U slučaju trajektorija ovo znači da se trajektorije moraju vremenski uskladiti, tj. svesti na istu razinu trajanja (normalizirati). Ovo za posljedicu ima i to da svaka trajektorija sadrži jednaku količinu podataka. Najčešće korištena metoda je DTW (eng. *Dynamic time warping*) koja je izvorno predstavljena u [68] na primjeru vremenskog poravnavanja ljudskog pokreta. Ovaj algoritam temelji se na izračunima udaljenosti između segmenata pokreta i združivanju najsličnijih (*matching*). Primjena DTW vremenskog poravnavanja kao priprema za statističko modeliranje trajektorija prikazana je u [67], [69] i [66].



Slika 2-6. DTW vremensko poravnavanje trajektorije sa jednim stupnjem slobode, te njihovo modeliranje GMM modelom [66].

2.4. DMP

Osim statističkih metoda koje se koriste za modeliranje demonstriranih trajektorija, postoje i drugi pristupi. Proučavanjem bio-mehaničkih sustava došlo se do zaključaka da mnoga gibanja u životinjskom svijetu mogu sagledati kao jednostavna gibanja do određene konačne točke (eng. *point attracting movement*) te da se i na prvi pogled kompleksna gibanja mogu rastaviti na ovakve cjeline. Temeljeno na ovim osnovama, razvijen je model za oponašanje kretanja bioloških sustava pod nazivom DMP (*eng. Dynamic movement primitives*). Ovaj model prvi puta je predstavljen u

[70] i nakon toga proširen u [71]. Budući da je ovaj model osnova velikog dijela metodologije u nastavku rada, ovdje će se dati detaljan uvid u DMP metodu.

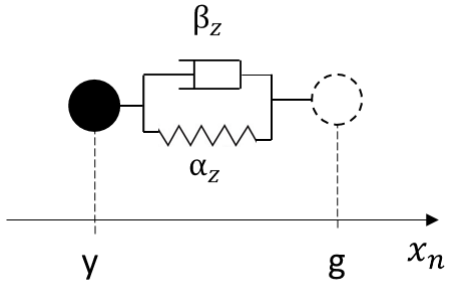
Jednodimenzijsko gibanje iz neke početne pozicije do neke konačne točke ovdje je predstavljeno diferencijalnom jednadžbom mehaničkog sustava opruge i prigušenja.

$$\tau^2 \ddot{y} = \alpha_z(\beta_z(g - y) - \tau\dot{y}). \tag{2.1}$$

gdje je τ vrijeme trajanja gibanja, α_z i β_z su konstante koje se koriste za podešavanje stabilnosti sustava (prigušenje i opruga), g je željena krajnja pozicija sustava, a $y, \dot{y}, \ddot{y}$ su redom varijable pozicije, brzine i ubrzanja sustava. Jednadžba ovog sustava se može zapisati u obliku dvije jednadžbe prvog reda:

$$\begin{aligned} \tau\dot{z} &= \alpha_z(\beta_z(g - y) - z) \\ \tau\dot{y} &= z. \end{aligned} \tag{2.2}$$

Opisani dinamički sustav opruge i prigušenja, je sustav koji uvijek teži (konvergira) nekom konačnom položaju u kojem je sustav stabilan. Tranzicija iz početne pozicije do konačnog stanja ovisna je o parametrima sustava, koji definiraju brzinu odziva sustava i prebačaj. Karakteristike ovog sustava odgovaraju opisanim osnovnim gibanjima, koja su identificirana u prirodnim bio-mehaničkim sustavima.



Slika 2-7. Ilustracija dinamičkog sustava opruge i prigušenja koji se koristi za opisivanje diskretnog jednodimenzijskog gibanja u DMP zapisu (eng. point-attractor system). Parametri modela α_z i β_z analogni su konstanti opruge i konstanti prigušenja, dok su y i g zadana početna i krajnja pozicija.

Ograničenje ovako definiranog sustava je u tome što oblik njegovog odziva nije moguće značajno mijenjati tj. moguće ga je samo ubrzati, prigušiti i time relativno malo utjecati na njegov oblik. Sa stajališta oblikovanja trajektorija, ovo nije dovoljno, jer teoretski, model mora omogućiti bilo koji oblik trajektorije. Kako bi se ovo postiglo, na opisani dinamički model dodaje se tzv. funkcija oblika (eng. *forcing term*). To je u osnovi nelinearna funkcija koja može poprimiti bilo koji oblik,

te kao takva utječe na (linearni) odziv dinamičko sustava tj. u konačnici na oblik trajektorije. Jednadžbe u obliku sustava prvog reda sa dodanom funkcijom oblika f, sada izgledaju ovako:

$$\begin{aligned} \tau\dot{z} &= \alpha_z(\beta_z(g-y) - z + f \\ \tau\dot{y} &= z. \end{aligned} \tag{2.3}$$

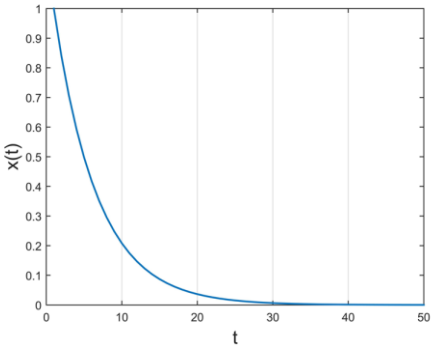
U osnovi, funkcija oblika može biti bilo koja vrsta funkcije. Ipak, zbog prilagodljivosti i modularnosti je izvorno predložen model funkcije u obliku niza N Gaussovih normalnih raspodjela ψ_i (RBF). Ovo je parametarska funkcija čije su vrijednosti definirane kao linearna kombinacija otežanih vrijednosti Gaussovih raspodjela u svakom trenutku funkcije (vremena). Funkcija se može zapisati u vremenski zavisnom obliku kao:

$$f(t) = \frac{\sum_{i=1}^{N} \psi_i(t) w_i}{\sum_{i=1}^{N} \psi_i(t)}. \tag{2.4}$$

Ovakva nelinearna funkcija direktno zavisna o linearnom vremenu još uvijek nije pogodna za modeliranje oblika funkcije. Stoga se kao posredna varijabla vremena koristi tzv. fazna varijabla x koja je ovisna o standardnom linearnom obliku vremena. Fazna varijabla dana je u obliku diferencijalne jednadžbe prvog reda:

$$\tau\dot{x}(t) = -\alpha_x x(t), \tag{2.5}$$

gdje je α_x parametar pojačanja koji definira nagib nove funkcije fazne varijable. Iz funkcije je vidljivo da je fazna varijabla ustvari eksponencijalno padajuća funkcija čiji je oblik vidljiv na slici 2-8.



Slika 2-8. Fazna varijabla u obliku eksponencijalno padajuće funkcije.

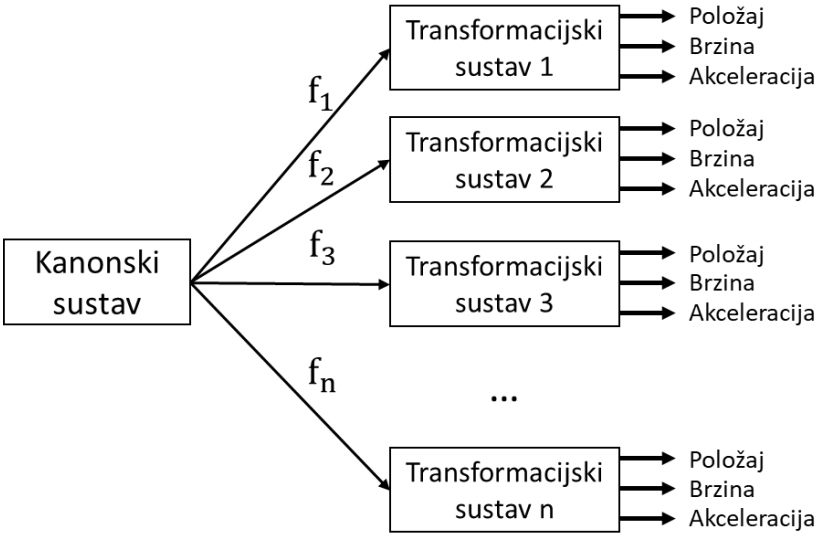
Funkcija oblika ovisna o faznoj varijabli vremena može se zapisati kao u (2.6).

$$f(x) = \frac{\sum_{i=1}^{N} \psi_i(x) w_i}{\sum_{i=1}^{N} \psi_i(x)} x(g - y_0) \tag{2.6}$$

Ovdje je g ciljno stanje, y_0 počento stanje, a eksponencijalne bazne funkcije su definirane kao:

$$\psi_i(x) = exp\left(-\frac{1}{2\sigma_i^2}(x - c_i)^2\right). \tag{2.7}$$

U ovakvom konceptu, gdje sustav ustvari nije direktno vremenski ovisan, nego je ovisan o faznoj varijabli, faznu varijablu se još naziva kanonski sustav (eng. *canonical system*). Taj sustav pokreće dinamički sustav koji se još naziva transformacijski sustav (eng. *transformation system*). Ovdje dolazimo do bitne karakteristike DMP modela, a to je modularnost. Naime, DMP podržava proizvoljan broj transformacijskih sustava gdje svaki predstavlja jedan stupanj slobode. To znači da DMP trajektorija može imati neograničen broj stupnjeva slobode. Kako bi se ovi transformacijski sustavi vremenski sinkronizirali, pokreće ih isti kanonski sustav čija uloga se može opisati kao uloga središnjeg vremenskog mehanizma. Ovakva modularna struktura idealna je za robotske sustave sa velikim brojem stupnjeva slobode, gdje se gibanje svakog zgloba (stupnja slobode) može opisati svojim dinamičkim transformacijskim sustavom. Modularna struktura DMP sustava dana je na slici 2-9.



Slika 2-9. Struktura DMP sustava sa n stupnjeva slobode [71].

2.4.1. Učenje sa DMP modelima

Jako bitna karakteristika DMP modela je mogućnost učenja trajektorije iz demonstracije. Za razliku od prethodno viđenih statističkih metoda, DMP trajektorija se može naučiti iz samo jedne demonstracije. Kako je ranije predstavljena funkcija oblika f u osnovi parametrizirana nelinearna funkcija, čiji parametri mogu biti podešeni kako bi se postigao bilo koji oblik trajektorije, učenje

trajektorije se svodi na učenje ovih parametara. U nastavku će biti pokazano da se pretvaranje neke demonstrirane trajektorije u DMP zapis u biti svodi se na klasični problem nadziranog učenja koji se može riješiti standardnim regresijskim metodama.

Iz jednadžbi transformacijskih sustava, funkcija oblika se može eksplicitno izraziti kao:

$$f = \tau\dot{z} - \alpha_z(\beta_z(g - y) - z. \tag{2.8}$$

Pretpostavi li poznavanje trajektorije koja se želi naučiti (poznate akceleracije, brzine i pozicije), funkcija oblika se može iz ovakve trajektorije izraziti pomoću jednakosti:

$$f_{target} = \tau^2\ddot{y}_{demo} - \alpha_z(\beta_z(g - y_{demo}) - \tau\dot{y}_{demo}. \tag{2.9}$$

Cilj samog procesa učenja kod DMP modela je pronaći težinske parametre nelinearne funkcije oblika f tako da ova funkcija što bolje odgovara funkciji oblika trajektorije koja se želi naučiti f_{target}. Ovaj problem može se riješiti različitim regresijskim algoritmima, među kojima je i već spomenuti LWR algoritam. Cilj regresijskog algoritma je minimizirati otežanu grešku kvadrata

$$J_i = \sum_{t=1}^{T} \psi_i(t)\big(f_{target}(t) - w_i\varepsilon(t)\big)^2, \tag{2.10}$$

gdje $\varepsilon(t) = x(t)(g - y_0)$ za diskretna gibanja.

Rješenje regresijskog problema je pronaći vrijednosti težinskih parametara w_i za svaku baznu funkciju ψ_i. Korištenjem LWR metode težine se računaju na sljedeći način:

$$w_i = \frac{S^T\Gamma_i f_{target}}{S^T\Gamma_i S}, \tag{2.11}$$

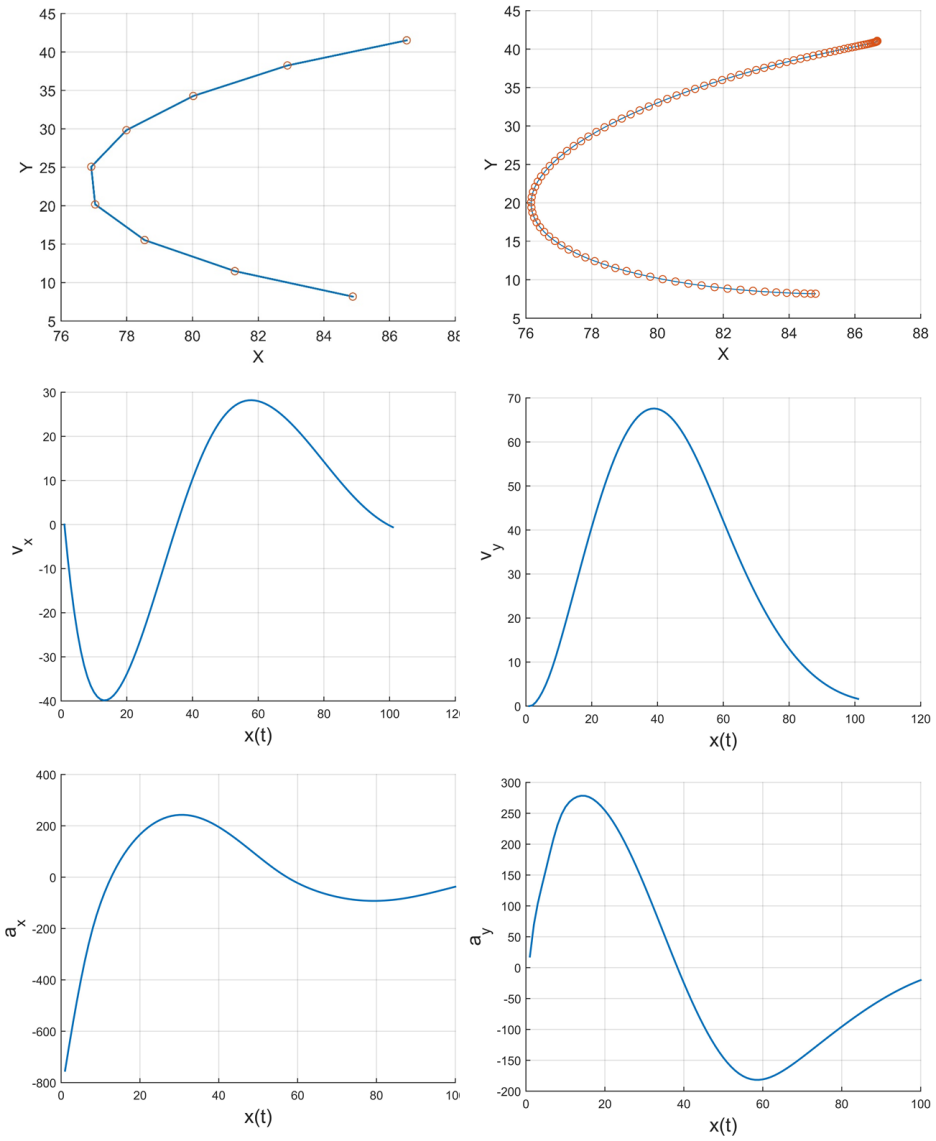
gdje

$$S = \begin{pmatrix} \varepsilon(t_0) \\ \varepsilon(t_1) \\ \vdots \\ \varepsilon(t_N) \end{pmatrix}, \Gamma_i = \begin{pmatrix} \psi_i(t_0) & 0 & \cdots & 0 \\ 0 & \psi_i(t_1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \psi_i(t_N) \end{pmatrix},$$
$$f_{target} = \begin{pmatrix} f_{target}(t_0) \\ f_{target}(t_1) \\ \vdots \\ f_{target}(t_N) \end{pmatrix}. \tag{2.12}$$

Naučene težine mogu imati pozitivne i negativne vrijednosti, te se one koriste od strane transformacijskih sustava kako bi se reproducirala trajektorija. Trajektorija naučena na temelju pokreta koji obavlja neki zadatak, može biti jako slična demonstriranoj trajektoriji. Ipak, ovakva reprodukcija ne može garantirati uspješno obavljanje demonstriranog zadatka od strane robota koji koristi DMP parametrizaciju pokreta. Zato se parametri težina uvijek mogu fino podesiti drugim pristupima kao što su pristupi podržanog učenja.

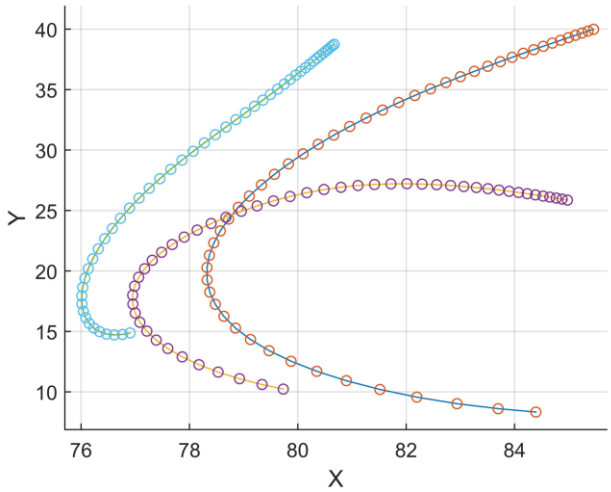
2.4.2. DMP karakteristike

Opisani DMP model kretanja osim učenja, modularnosti i stabilne konvergencije cilju posjeduje i druge karakteristike koje su poželjne za modeliranje gibanja. Prva od njih je kontinuiran i uglađen profil brzina i akceleracija tijekom izvođenja gibanja. Ovo je karakteristika koja je jako bitna kod izvođenja pokreta na bilo kojoj razini gibanja i na bilo kojem robotu, jer u suprotnom lako može doći do mehaničkih oštećenja robota. Primjer dvodimenzionalne DMP trajektorija u kartezijskom koordinatnom sustavu naučenu iz proizvoljne krivulje vidljiva je na slici 2-10., te su prikazani i njezini profili brzine.



Slika 2-10. Dvodimenzionalna DMP trajektorija i pripadajući profili brzina i akceleracija za svaki stupanj slobode.

Druga vrlo bitna karakteristika DMP zapisa je mogućnost prostornog i vremenskog skaliranja kretnji. DMP trajektorija je osim prethodno naučenim težinama definirana početnom i krajnjom pozicijom - y_0 i g. Promjenom parametara starta i cilja, trajektorija ostaje stabilna u vidu konvergencije cilju dok istovremeno zadržava osnovne karakteristike oblika koji su zadani težinama funkcije oblika (slika 2-11.). Ovo svojstvo je vrlo značajno kod primjene trajektorije u različitim konfiguracijama radnog prostora tj. kod generalizacije pokreta. Isto tako, DMP trajektoriju moguće je ubrzavati i usporavati putem parametra τ (odabira razine na kojoj će se tijekom reprodukcije DMP uzorkovati).



Slika 2-11. Prostorno skaliranje DMP modela na različite početne i krajnje pozicije.

2.5. Generalizacije na razini trajektorija

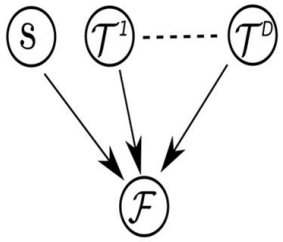
Do sada prikazane statističke metode i DMP omogućuju matematički zapis demonstriranih trajektorija u konfiguracijskom ili kartezijskom sustavu, s ciljem reprodukcije istih na robotima. One kao takve ne pružaju mogućnost za programiranje i učenje orijentirano zadatku ili generalizaciju gibanja, ali predstavljaju dobar okvir za ostvarivanje istog. U nastavku će se prikazati dva osnovna pristupa kojima se postižu generalizacijske sposobnosti u učenju iz demonstracija.

2.5.1. TP modeli

Svaki zadatak posjeduje određene parametre koji ga čine specifičnim. Korištenjem ove hipoteze nastali su modeli koji oblikuju robotske akcije/kretnje u odnosu na ove parametre. Parametri

zadatka (eng. *Task parameters* - TP) mogu biti primjerice položaj objekata u prostoru, visina prepreke, iznos očekivane sile, položaj prepreka u prostoru. Modele učenja robotskog djelovanja u odnosu na parametre zadatka karakterizira kompaktnost, mali broj demonstracija mogućnost učenja na niskoj razini izvršavanja.

TP modeli koriste različite načine zapisa trajektorija kako bi opisali trajektoriju i stvorili vezu između trajektorije i parametara zadatka. Uz vrijeme, ovi modeli kao ulaze uvode parametar zadatka τ. Najčešće korišteni modeli su GMM (eng. *Gaussian mixture model*), a GMR (eng. *Gaussian mixture regression*) se koristi kako bi se dobili izlazni podaci iz modela na osnovu novih vrijednosti parametara zadatka [72]. DMP modeli su isto tako korišteni za modeliranje u odnosu na parametre zadatka. U [73] i [74] pokazano je da korištenjem DMP modela trajektorija zajedno sa statističkim metodama strojnog učenja, je moguće ostvariti generalizaciju na nove vrijednosti parametara.



Slika 2-12. Princip modeliranja parametara u odnosu na DMP [73].

Bitno je naglasiti da parametri zadatka kod ovakvog modeliranja moraju biti definirani od strane čovjeka, a modeli uče isključivo vezu između parametara zadatka i ponašanja. Neki fleksibilniji pristupi prikazuju da se utjecaj pojedinih parametara zadatka može skalirati i na druge zadatke [75], [53].

2.5.2. IRL modeli

Na drugoj strani robotskog učenja nalazi se podržano učenje kod kojeg robot istražuje prostor stanja i biva evaluiran na osnovu funkcije cilja/nagrade koja mora biti unaprijed poznata. Često ovu funkciju nije jednostavno odrediti, jer ona mora opisati bitne dijelove ponašanja za svaki zadatak posebno. Inverzno podržano učenje (eng. *Inverse reinforcement learning* – IRL) predstavlja pristup koji pokušava dati rješenje ovom problemu. IRL pristupi pokušavaju na osnovu demonstracija modelirati funkciju nagrade. U osnovi se ovdje demonstracije koriste kako bi se naučile vrijednosti težina značajki u funkciji cilja/nagrade [76], [77], [78]. Ovo je optimizacijski proces gdje su težine

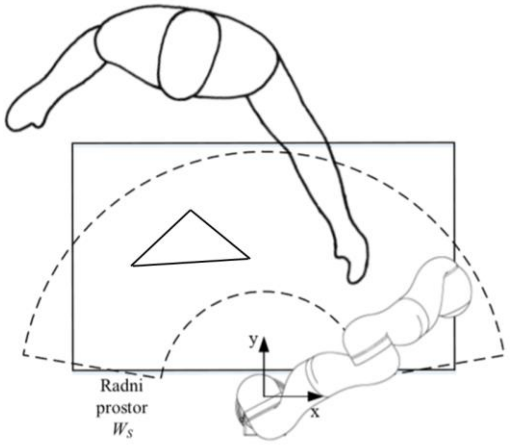
$\boldsymbol{\alpha}$ unaprijed definiranih značajki Φ (eng. *features*) dobiveni s obzirom na optimalno ponašanje. Optimalno ponašanje je ovdje zadano u obliku ljudskih demonstracija. Ovaj proces učenja trebao bi rezultirati funkcijom cilja/nagrade koja opisuje demonstrirano optimalno ponašanje.

$$R(s) = \alpha_1\Phi_1(s) + \alpha_1\Phi_1(s) + \cdots + \alpha_d\Phi_d(s) \tag{2.13}$$

Ovako dobivene funkcije nagrada mogu se koristiti za generalizaciju ponašanja trajektorija.

2.6. Učenje klasifikacijom prostornih koordinatnih sustava

Glavne ideje učenja iz demonstracija na razini trajektorija mogu se sagledati u dvije stavke. Prva je zapis pokreta u neki parametarski model kako bi se sačuvale sve bitne informacije oblika trajektorije i njezine dinamike, a druga je i stvaranje veze između tih parametara i cilja zadatka (TP ili IRL pristup). Kako je ranije pokazano, postoje tri glavne domene u kojima se bilježe informacije iz demonstracija: konfiguracijski prostor robota, kartezijski prostor i vizijska domena (tj. slike i vrijednosti pripadajućih piksela). Vizijska domena predstavlja najvišu razinu apstrakcije koja je na prirodan način prilagođena i razumljiva čovjeku. Njezin nedostatak je vrlo velika kompleksnost u računalnom smislu. Kartezijski prostor parametarski je prostor opisan translacijskim i rotacijskim komponentama koje čine informacije o međusobnom odnosu koordinatnih sustava. Isti je vrlo lako razumljiv ljudima i često se koristi u svim segmentima života, a iz ovog prostora je pretvorba u konfiguracijski prostor putem inverzne kinematike vrlo jednostavna. Konfiguracijski prostor robota je prostor koji je sa stajališta čovjeka najteže sagledati i razumjeti jer je on specifičan za područje robotike. Kartezijski se sustav zato može smatrati kao domena u kojoj je najjednostavnije provesti analizu demonstracija ali i reprodukciju novih rješenja.



Slika 2-13. Interakcija čovjeka i robota u kartezijskom prostoru [79].

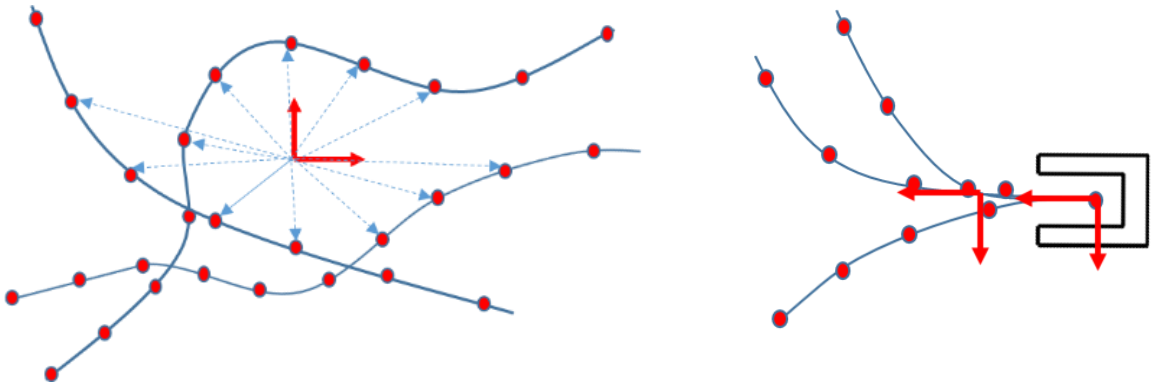
U ovom poglavlju predstaviti će se stoga pristup koji je baziran na demonstracijama i analizi u kartezijskom prostoru. Metoda je namijenjena za učenje zadatka iz demonstracija robotskih trajektorija temeljen na analizi njihovog prostornog odnosa sa karakterističnim koordinatnim sustavima u radnom prostoru robota.

2.6.1. Metodologija klasifikacije prostornih koordinatnih sustava

Demonstrirane trajektorije robota promatraju se kao T_m podatkovnih točaka u svakoj demonstraciji od $m = 1 \dots M$. Parametri zadatka predstavljeni su kao koordinatni sustavi P_n, ovdje nazvani karakteristični koordinatni sustavi, gdje se svaki kooordinatni sustav $n = 1 \dots N$ pojavljuje u svakoj demonstraciji m, ali u različitim prostornim konfiguracijama. Transformiraju li se demonstrirane trajektorije u ovaj koordinatni sustav, moguće je promatrati demonstracije iz perspektive ovog karakterističnog koordinatnog sustava P_n. Pretpostavkom da P_n može biti vezan za neki objekt, brid ili neku drugu značajku u demonstracijskom prostoru trajektorija, ovako se može uspostaviti veza između demonstriranih trajektorija i bilo koje prostorne značajke izražene Kartezijevim koordinatnim sustavom.

Ovdje je korištena pretpostavka da se utjecaj karakterističnih koordinatnih sustava na demonstraciju, a samim time i na robotski zadatak može klasificirati prema aktivnosti u tri kategorije:

- aktivni koordinatni sustavi – koordinatni sustavi koji privlače trajektoriju
- neaktivni koordinatni sustavi – koordinatni sustavi koji nemaju utjecaja na trajektoriju
- prepreke – koordinatni sustavi koji odbijaju trajektoriju



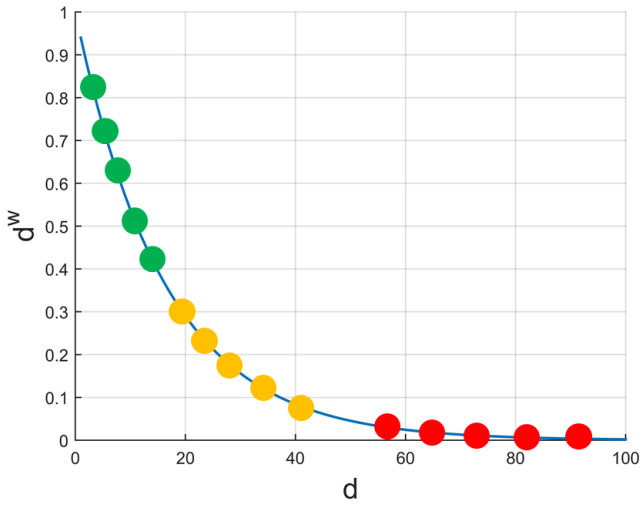
Slika 2-14. Položaj koordinatnog sustava u odnosu na demonstrirane trajektorije.

Demonstracijski setovi sastoje se od demonstriranih trajektorija i karakterističnih koordinatnih sustava iz robotske okoline, čiji je položaj dobiven upotrebom vizijskog/ih sustava.

Za klasifikaciju koordinatnog sustava prema njegovoj aktivnosti u demonstracijama, koristi se Euklidska udaljenost od svake podatkovne točke trajektorije koje su prije toga prostorno uzorkovane. Kao preduvjet za učenje ovom metodom potrebno je trajektorije dobivene kinestetičkim ili teleoperacijskim učenjem obraditi na odgovarajući način. Ta procedura opisana je u poglavlju 2.6.2. Prethodno spomenute Euklidske udaljenosti tvore *M* x *N* matricu udaljenosti $\{\{d_{n,m}\}_{n=1}^{N}\}_{m=1}^{M}$ za svaki karakteristični koordinatni sustav P_n.

Kako bi se iz matrice udaljenosti dobila korisna informacija o aktivnosti koordinatnog sustava, elementi ove matrice udaljenosti se otežavaju pomoću eksponencijalne funkcije $\{\{d_{n,m}{}^{w}\}_{n=1}^{N}\}_{m=1}^{M} = x^{d}$ gdje je $x \in \{0.1 \dots 1\}$. Ovo ima za posljedicu da točke trajektorija koje imaju manje udaljenosti dobivaju veće skalarne vrijednosti, a točke sa većom udaljenosti dobivaju minimalne vrijednosti. Suma ovako otežanih skalarnih vrijednosti matrice daje informaciju o aktivnosti koordinatnog sustava σ (2.14).

$$\sigma = \sum_{m=1}^{M} \sum_{n=1}^{N} d_{n,m}^{w} \tag{2.14}$$



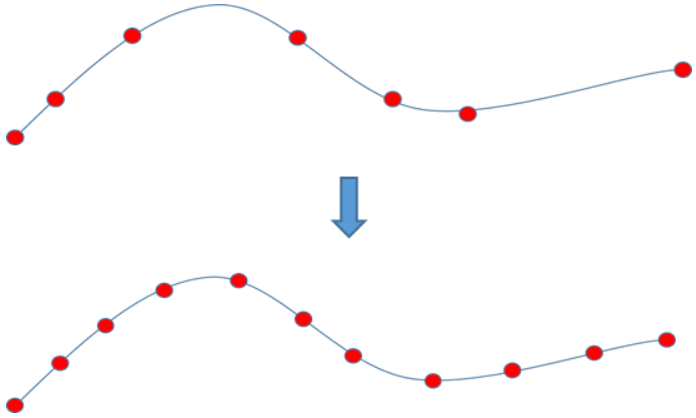
Slika 2-15. Primjer otežanih vrijednosti za tri koordinatna sustava u odnosu na pet prostornih točaka u idealnom slučaju. Upotrebom jednadžbe (2.14), zeleni koordinatni sustav klasificira se kao aktivni koordinatni sustav, žuti kao neaktivni, a crveni kao prepreka.

Velika vrijednost sume može se protumačiti kao slučaj gdje je karakteristični koordinatni sustav u svim demonstracijama dio trajektorije, tj. trajektorija prolazi vrlo blizu njega (slika 2-14. desno).

2.6.2. Prostorno uzorkovanje trajektorije

Robotske trajektorije demonstrirane teleoperacijom ili kinestetičkim učenjem najčešće su uzorkovane konstantnim vremenom uzorkovanja. Takva trajektorija osim informacija o poziciji sadrži i informacije o brzini i akceleraciji demonstrirane trajektorije u svakom uzorkovanom vremenu, ali sadrži prostorno neravnomjerno raspoređene točke uzorkovanja.

U slučaju prostorne analize podataka kakvu smo predstavili u ovom poglavlju, preduvjet je da su točke uzorkovanja trajektorije u kartezijskom prostoru ravnomjerno raspoređene na trajektoriji. Iz tog razloga se prije klasifikacije prostornih koordinatnih sustava demonstrirana trajektorija prvo prostorno jednolično uzorkuje prema euklidskoj udaljenosti.



Slika 2-16. Izvorno uzorkovana trajektorija (gore). Jednolično prostorno uzorkovana trajektorija (dolje).

Algoritam za jednolično uzorkovanje uzima u obzir cjelokupnu duljinu trajektorije, dijeli tu udaljenost na željeni broj uzorkovanih točaka, te formira nove točke prema interpolaciji udaljenosti u odnosu na postojeće točke trajektorije. Ista procedura se provodi za sve stupnjeve slobode gibanja trajektorije. Za dvodimenzijski slučaj, primjer izgleda trajektorije prikazan je na slici 2-16.

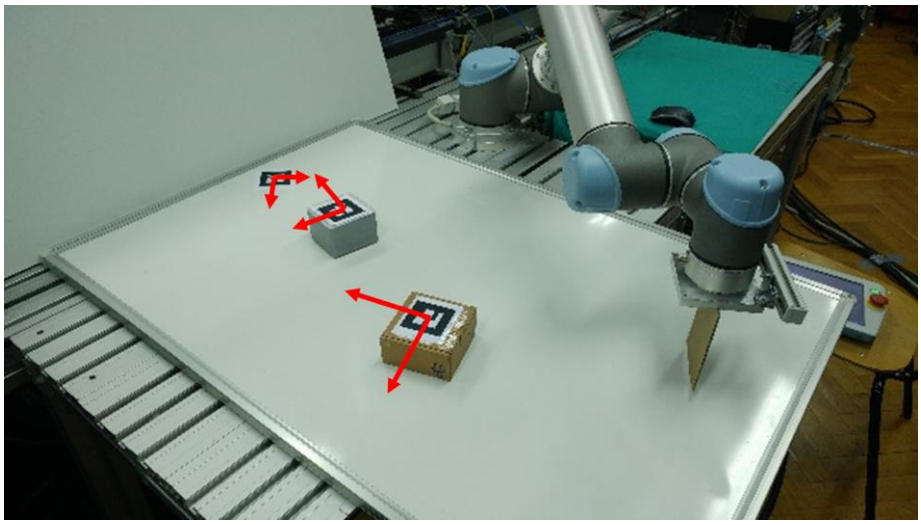
2.6.3. Klasifikacija ciljne točke

Robotska trajektorija u kartezijskom prostoru predstavlja promjenu položaja vrha alata iz nekog početnog stanja u neko krajnje stanje. Demonstracije trajektorija u konačnici opisuju upravo ovu tranziciju. Početno stanje robota u nekoj novoj prostornoj situaciji lako je očitati na temelju trenutnog konfiguracijskog ili Kartezijeva položaja robota. S druge strane, krajnja pozicija trajektorije najčešće nije tako kruto definirana, nego ovisi o cilju zadatka. Predstavljena

klasifikacija prostornih koordinatnih sustava može se koristiti za klasifikaciju ciljne točke trajektorije iz aktivnih koordinatnih sustava. Ovo znači da se iz skupa klasificiranih aktivnih koordinatnih sustava, još jednom dodatnom klasifikacijom može raditi određivanje ciljne pozicije. Dodatni klasifikacijski algoritam radi na istom principu kao i prvi klasifikacijski algoritam ali kao ulazni skup uzima samo prethodno klasificirane aktivne koordinatne sustave, a kao prostorno relevantne točke trajektorije uzima samo krajnje točke svih trajektorija. Aktivni koordinatni sustav koji ima najveću vrijednost s obzirom na udaljenost od krajnjih točaka trajektorija, klasificira se kao ciljna točka.

2.6.4. Eksperimentalna validacija analize prostornih točaka

Eksperimentalni postav za validaciju predložene analize prostornih točaka biti će predstavljen u ovom poglavlju. Postav se sastoji od UR5 robota, osobnog računala i Kinect 2.0 vizijskog sustava.

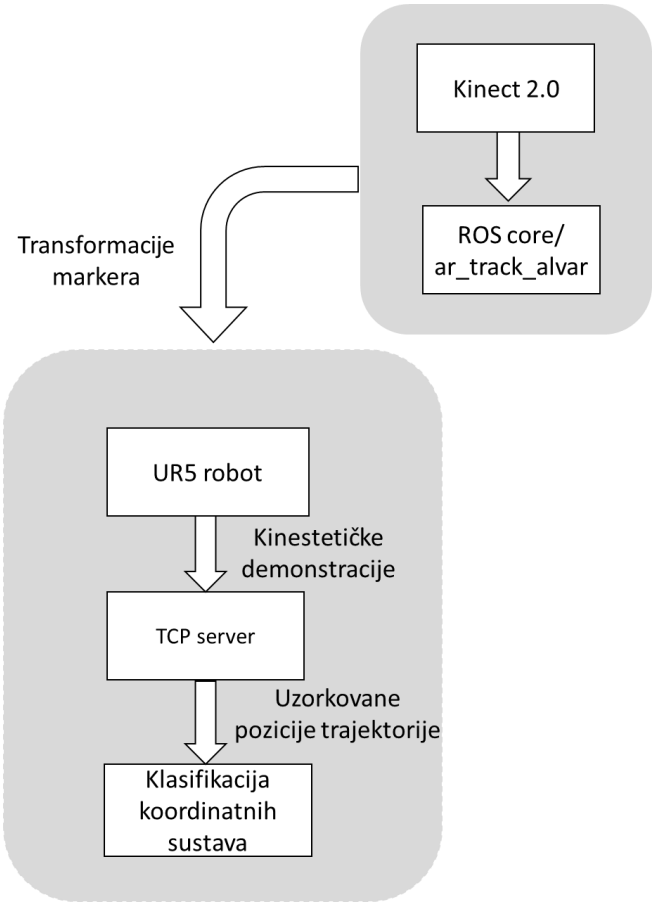


Slika 2-17. Eksperimentalni postav.

UR5 je kolaborativna robotska ruka sa šest stupnjeva slobode koja ima mogućnost ručnog prostornog navođenja sa kompenzacijom gravitacije. Ovaj način rada pogodan je za kinestetičko učenje. Tijekom demonstracija, prostorne koordinate vrha alata uzorkovane su konstantnom frekvencijom od 100 Hz i slane osobnom računalu preko TCP/IP servera gdje su ovi podaci spremani u datoteku. Osim pozicijskih i orijentacijskih podataka o vrhu alata, ovako spremljena trajektorija u sebi sadrži i podatke o brzinama i akceleracijama koje su se javljale u trajektorijama u svakom vremenskom trenutku. Za potrebe klasifikacijskog algoritma, ove trajektorije su

naknadno ponovno uzorkovane kako bi se dobio prostorno jednolik raspored točaka na trajektoriji (prema metodologiji u poglavlju 2.6.2).

Korištenjem Kinect vizijskog sustava spojenog na osobno računalo, omogućeno je snimanje radnog prostora robota neposredno prije samih demonstracija. Radi lakše segmentacije i analize okoline, na predmete od interesa u okolini robota stavljeni su markeri u obliku sličnom QR kodovima. Na osobnom računalu se u ROS okruženju koristi knjižnica pod nazivom *ar_track_alvar* za praćenje spomenutih markera. Koordinatni sustavi markera su se ovim sustavom jednostavno transformirali u k.s. robota. Snimljeni koordinatni sustavi su nakon toga klasificirani prema metodologiji predloženoj u 2.6.1.



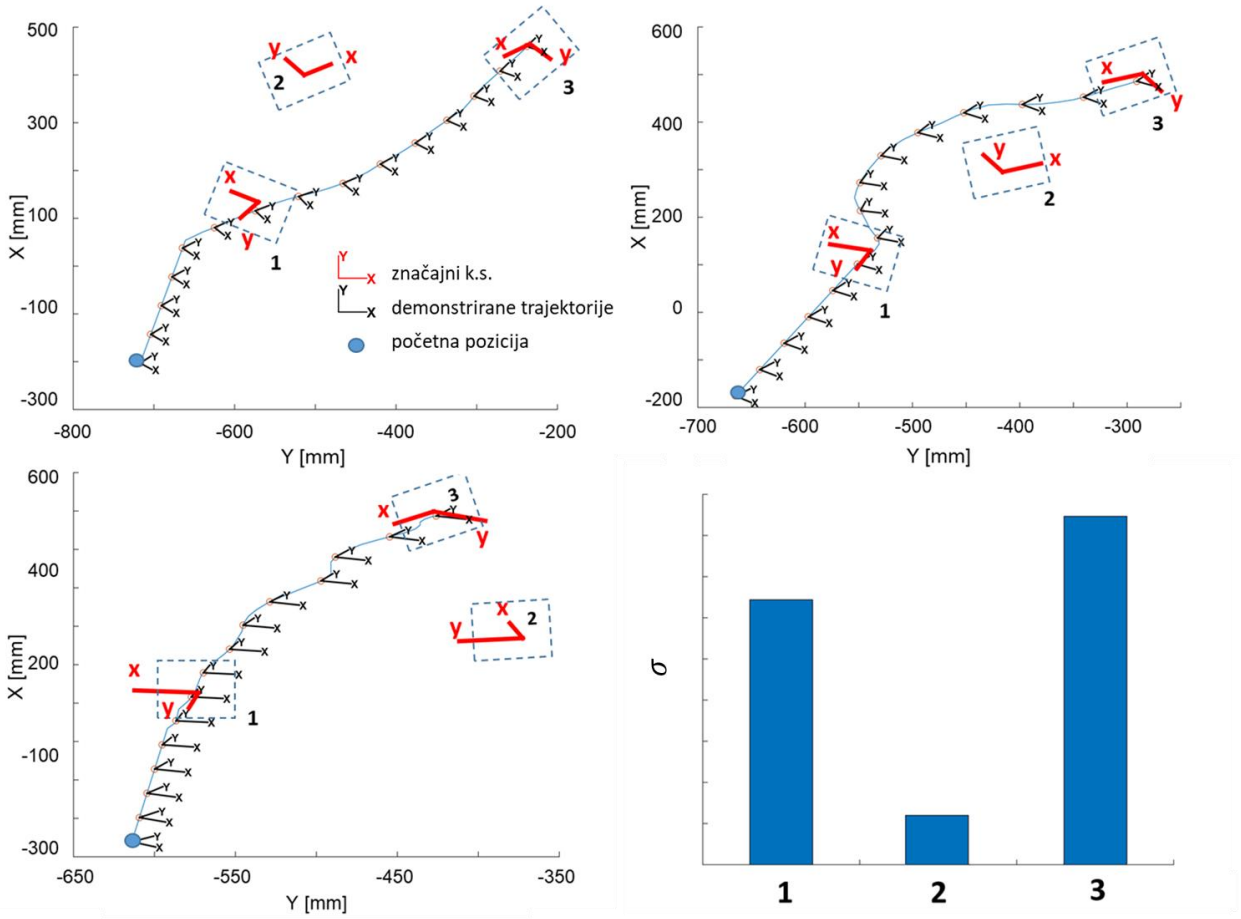
Slika 2-18. Blok dijagram razvijenog eksperimentalno postava.

2.6.5. Rezultati klasifikacijskog algoritma

Geometrijska analiza demonstracija predstavljena u ovom poglavlju alat je za ekstrakciju informacija iz demonstracija. Ista je bazirana na klasifikaciji koordinatnih sustava prema njihovoj

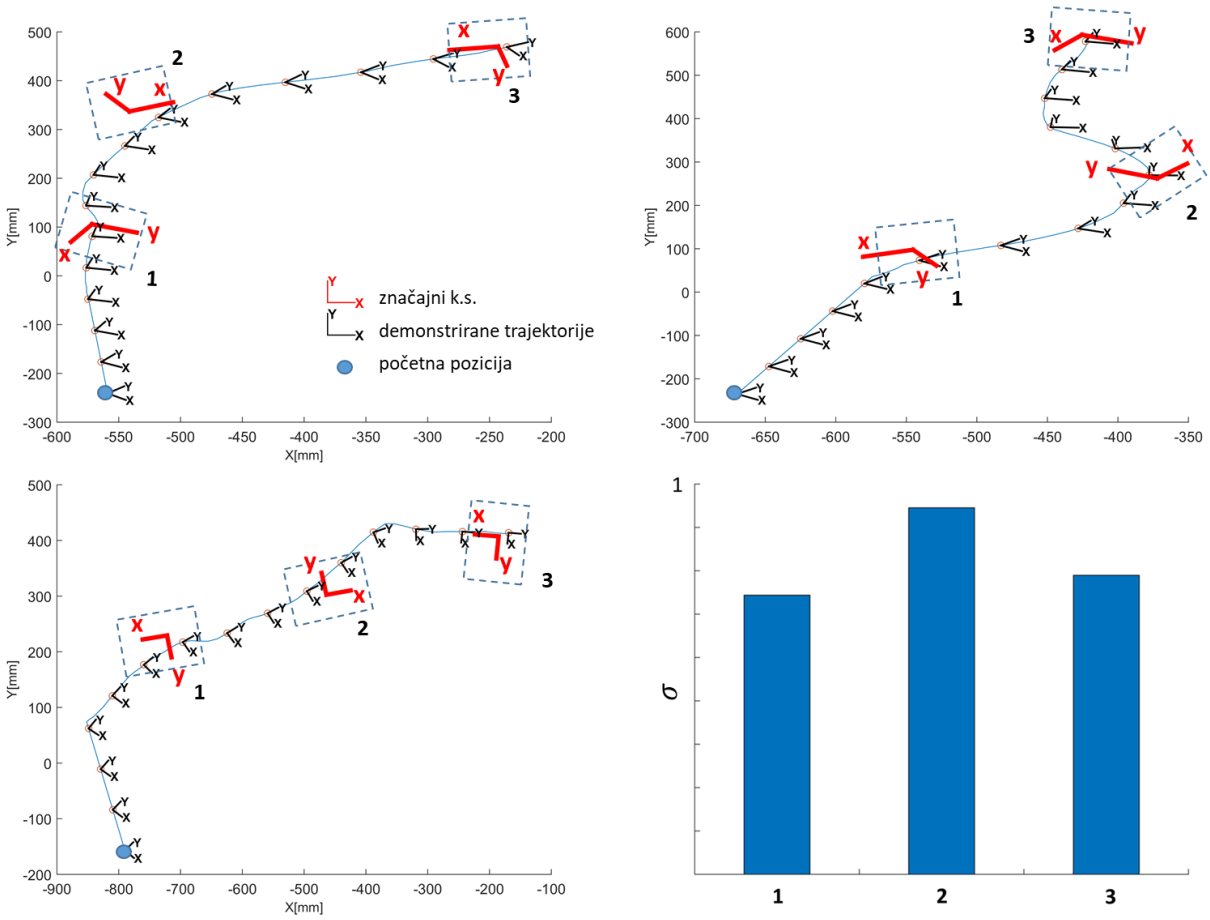
udaljenosti od demonstriranih trajektorija. Ovakva klasifikacija govori o tome na koji je način određeni koordinatni sustav bio sastavni dio demonstracija. Ako je u svakoj demonstraciji bio sastavni dio trajektorije, tada se koordinatni sustav klasificira kao aktivni koordinatni sustav. U obrnutom slučaju gdje promatrani koordinatni sustav nije bio dio trajektorije niti u jednoj demonstraciji, tada se koordinatni sustav klasificira kao prepreka.

Na slici 2-19. prikazane su tri demonstracije trajektrija koje prolaze kroz koordinatni sustav predmeta *1* i završavaju u koordinatnom sustavu *3*, pri čemu se zaobilazi koordinatni sustav predmeta *2*. Predloženom analizom koordinatnih sustava izračunava se aktivnost pojedinog koordinatnog sustava. Vidljivo je da se koordinatni sustavi *1* i *3* mogu jednostavno klasificirati kao aktivni koordinatni sustavi, dok se k.s. *2* može klasificirati kao prepreka.



Slika 2-19. Demonstrirane trajektorije kinestetičkim učenjem u odnosu na promatrane koordinatne sustave 1,2 i 3, za tri različite konfiguracije zadatka. Grafički su prikazane skalarne vrijednosti aktivnosti svakog od koordinatnih sustava.

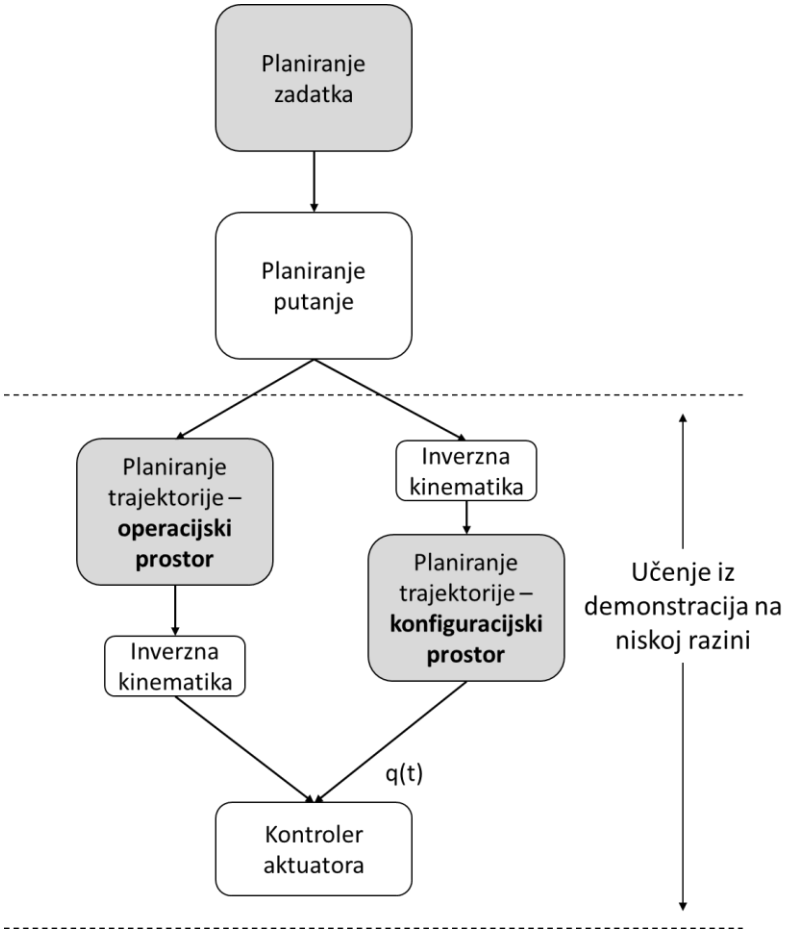
Drugi primjer (slika 2-20.) sastoji se od tri demonstracije kod kojih trajektorije prolaze kroz tri dostupna koordinatna sustava. Analizom aktivnosti svakog od njih, moguće je vidjeti da ne postoji značajna razlika između njihovih vrijednosti što omogućava njihovu klasifikaciju kao aktivnih koordinatnih sustava.



Slika 2-20. Demonstrirane trajektorije kinestetičkim učenjem u odnosu na promatrane koordinatne sustave 1,2 i 3, za tri različite konfiguracije zadatka. Grafički su prikazane skalarne vrijednosti aktivnosti svakog od koordinatnih sustava.

3. Planiranje kretanja orijentirano zadatku

Kako je do sada pokazano, problematika učenja na niskoj razini u robotici usko je vezana za problematiku planiranja kretanja. Metode prikazane u prošlom poglavlju omogućuju statističko modeliranje više demonstracija trajektorije tj. svojevrsno oponašanje koje ostaje na razini planiranja putanje tj. generalizacije planiranja trajektorije. Tek proširenja ove metodologije (IRL, TP) omogućuju svojstva poput generalizacije orijentirane zadatku.



Slika 3-1. Hijerarhija kod izvršavanja robotskog kretanja.

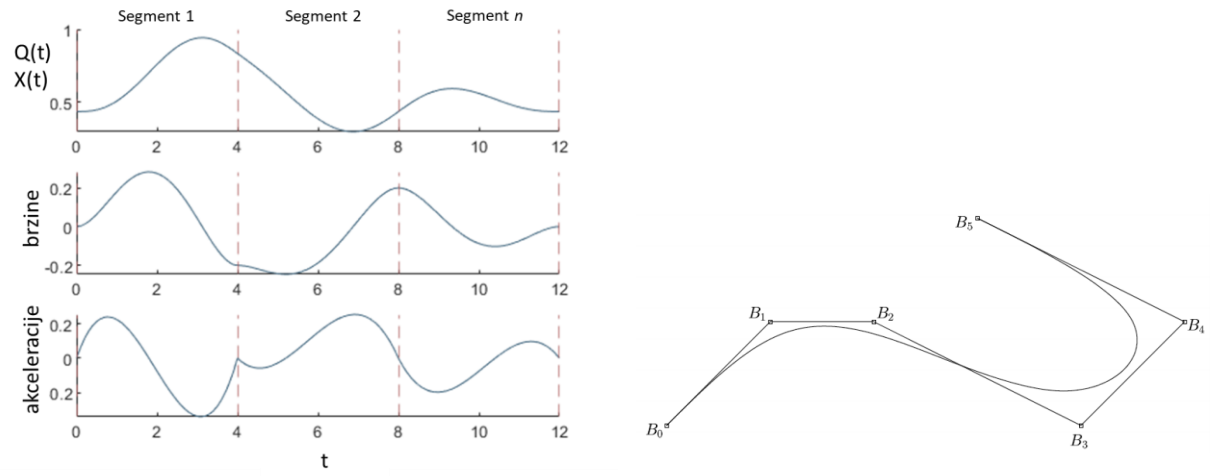
U ovom poglavlju će se predstaviti metodologija za planiranje kretanja u operacijskom prostoru robota, koja je obuhvaća planiranje putanje koje je orijentirano zadatku. Pod pojmom operacijskog prostora podrazumijevamo prostor u kojem je definirano ponašanje (u ovom slučaju trajektorija) ili konačni ishod nekog ponašanja. Gledano sa stajališta robota, operacijski prostor je najčešće kartezijski koordinatni sustav, iz tog razloga što je to definicija prostora bliska čovjeku,

kako je opisano i u prethodno u poglavlju 2.6.. Predložena metodologija oslanja se na već predstavljenu analizu demonstracija temeljenu na klasifikaciji prostornih koordinatnih sustava u kartezijskom prostoru.

3.1. Modeli trajektorija

Kod robotskih manipulatora, trajektorije su prostorne putanje parametrizirane vremenom. U [80], dana je detaljna klasifikacija trajektorija s obzirom na prostor u kojem se planiraju, vrstu zadatka, broj stupnjeva slobode, vremenske profile brzina i akceleracija. Osnovna podjela je na trajektorije u kartezijskom prostoru i trajektorije u konfiguracijskom prostoru robota.

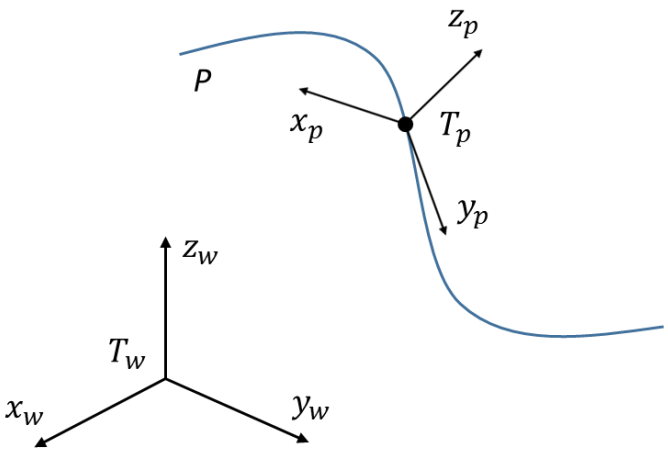
Trajektorije koje se planiraju u konfiguracijskom prostoru mogu se izvršavati direktno na upravljačkoj razini robota, uz korištenje odgovarajućih upravljačkih/regulacijskih uređaja (kontroler motora). S druge strane, trajektorije koje se planiraju u operacijskom prostoru, moraju se putem inverzne kinematike prevesti u konfiguracijski prostor robota. Prednosti planiranja u konfiguracijskom prostoru su uz mogućnost direktnog izvođenja na robotu je ta što se ograničenja manipulatora u obliku limita zglobova i ograničenja brzine mogu direktno uzeti u obzir kod planiranja. S druge strane, u operacijskom (kartezijskom) prostoru lakše je uzeti u obzir izbjegavanje prepreka koje se javljaju na putanji trajektorije, jer je njihov položaj opisan upravo u ovom prostoru. Isto tako, parametre zadatka koje robot izvršava puno je lakše opisati u kartezijskom prostoru, jer su oni predočljivi čovjeku.



Slika 3-2. Lijevo – B - Spline krivulja. Baza ovih krivulja je poligon kontrolnih točaka B_n [81]. Desno – trajektorija sastavljena od više segmenata opisanih polinomima [82].

Trajektorije se standardno u jednom i u drugom prostoru za potrebe planiranja trajektorija u robotici opisuju interpolacijskim metodama kao što su polinomi ili spline krivulje (slika 3-2.). Polinomi omogućuju spajanje više kontinuiranih segmenata u jednu cjelinu gdje je ovisno o stupnju polinoma eksplicitno zadavati iznose brzina i akceleracija na kraju svakog segmenta dok prijelazi brzina i akceleracija između segmenata ostaju kontinuirani. Spline zapis odlikuje kontinuitet i veća robusnost prilikom interpolacije točaka u odnosu na polinome višeg reda.

Trajektorije se u kartezijskom prostoru standardno predstavljaju pravokutnim (desnokretnim) koordinatnim sustavima koji se sastoje od translacijskog i rotacijskog dijela.



Slika 3-3. Transformacija u kartezijskom koordinatnom sustavu.

Translacijski dio se gotovo uvijek predstavlja kao tri linearne kartezijske koordinate, dok za rotacijski dio postoji više opcija. Neke od njih su:

- Eulerovi kutovi: rotaciju u tro-dimenzijskom koordinatnom sustavu (SO3) moguće je opisati sa minimalno tri parametra koji predstavljaju rotacije oko tri translacijske osi. Ovo su Eulerovi kutovi koji su dani u obliku vektora sa tri vrijednosti. Rotacije mogu biti dane različitim redoslijedom (ukupno postoji 27 konvencija), a u robotici su najčešće korištene ZYX i ZYZ konvencije. Nedostatak Euler zapisa je taj što za neke orijentacije ne može ponuditi jedinstvena rješenja (Gimbal lock fenomen, singularnost).
- Rotacijske matrice: dana je u obliku 3x3 matrice što daje ukupno 9 elemenata. Potpuni opis transformacije jednog koordinatnog sustava u drugi dan je u obliku homogene matrice transformacija:

$$T = \begin{bmatrix} & R_{3x3} & & x \\ & & & y \\ & & & z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{3.1}$$

Ovaj zapis pruža mogućnost jedinstvenog definiranja orijentacija, ali je broj parametara vrlo visok (9) u usporedbi sa Euler zapisom (3). Rotacijske matrice omogućuju izvođenje transformacija putem operacija jednostavnog matričnog množenja.

- Axis-angle zapis: dan je u obliku jediničnog vektora u koji određuje smjer rotacije i skalarne vrijednosti θ koja određuje magnitudu tj. iznos rotacije (ukupno četiri vrijednosti). Komponente vektora u i skalarna vrijednost θ međusobno su vezani jednakošću: $r_x^2 + r_y^2 + r_z^2 = \theta^2$. Problem ovog zapisa je postojanje orijentacija za koje postoje istodobno više rješenja (singularnosti).
- Kvaternioni: ovo je zapis koji omogućuje jednoznačno opisivanje orijentacije, pa time rješava probleme singularnosti koji se pojavljuju kod Axis-angle zapisa. Isto kao i Axis-angle, dani su u obliku vektora sa četiri vrijednosti $q = [w\ x\ y\ z]$. Kvaternioni posjeduju niz karakteristika koje su poželjne u robotici, kao što su mogućnost interpolacije kod prelaska iz jedne u drugu orijentaciju (primjerice *SLERP* interpolacija).

Od ova četiri predstavljena zapisa koji su najčešće korišteni za opisivanje orijentacija u robotici, samo rotacijske matrice i kvaternioni pružaju mogućnost jednoznačnog definiranja orijentacije. Isto tako, ovi zapisi pružaju mogućnost kontinuirane interpolacije kod prelaska iz jedne orijentacije u drugu, što je vrlo bitna karakteristika u robotici. Promatrajući kontinuitet kao mogućnost interpolacije svakog parametra zapisa pojedinačno, jedino rotacijske matrice posjeduju ovo svojstvo.

3.2. Jednostavno kretanje u robotskim zadacima

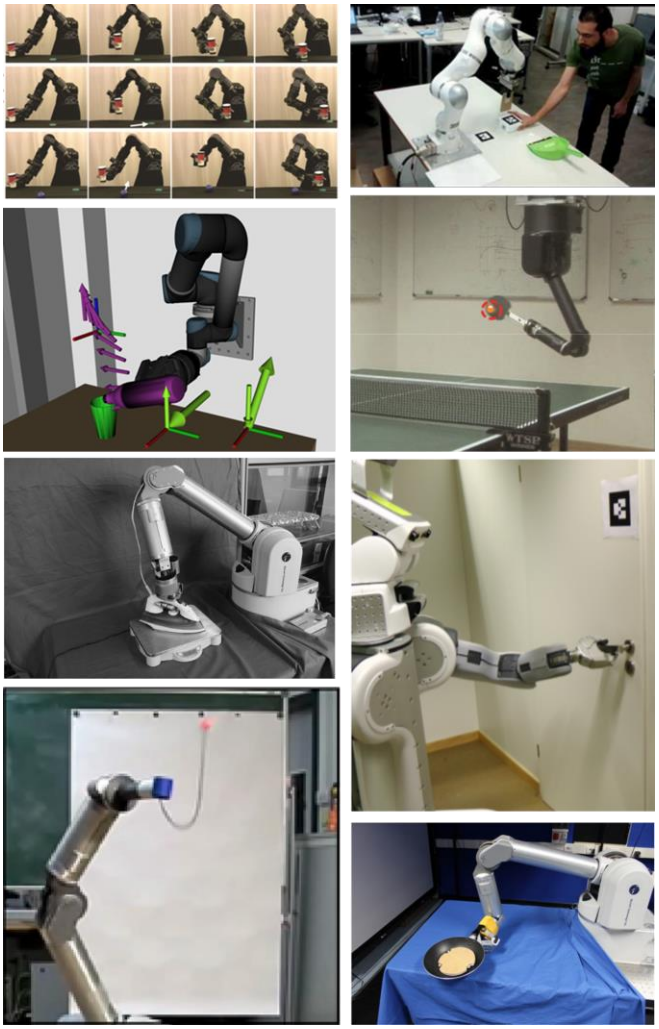
Kretanje je definirano početnom i krajnjom pozicijom, oblikom putanje i profilima brzine i akceleracije. Jednostavnim („primitivnim“) kretanjem u kontekstu robotike, smatra se onim koje se sastoji od samo jednog perioda ubrzavanja i jednog perioda usporavanja. Kao takvi, ovi pokreti su osnovna jedinica gibanja bilo kojeg mehanizma od točke A do točke B. Od njih se mogu sačinjavati razne druge složenije sekvence gibanja.

Na razini kretanja orijentiranog zadatku, primitivna kretanja mogu se koristiti za širok raspon zadataka. Neki od zadataka koji su često uzimani kao referentni prilikom testiranja raznih modela učenja su:

- "pick and place" zadatak [83],
- odguravanje predmeta (metenje) [73],
- zadatak točenja iz boce [84],
- stolni tenis, udaranje loptice [43], [85],
- glačanje [86],
- otvaranje vrata [87],
- "ball in cup" zadatak[88],
- prevrtanje palačinke [89].

Ostali primjeri:

- zadaci umetanja [90],
- pisanje simbola [91],
- hodanje humanoidnih robota [16].



Slika 3-4. Redom prikazani zadaci, od gore prema dolje, s lijeva prema desna.

3.3. DMP stanja

Budući da se prethodno predstavljeni model za učenje iz demonstracija temelji na analizi koordinatnih sustava u operacijskom (kartezijskom) sustavu robota, u ovom poglavlju će se predstaviti model trajektorije, koji se koristi za aproksimaciju kartezijskih stanja, te će isti nadograditi modifikacijom koja će ovaj model spregnuti sa predloženom analizom aktivnosti koordinatnih sustava. Baza ovog modela trajektorije je model primitivnog gibanja – DMP. S time da je korištena posebna varijacija DMP modela koja je sposobna aproksimirati stanja definirana u kartezijskom koordinatnom sustavu.

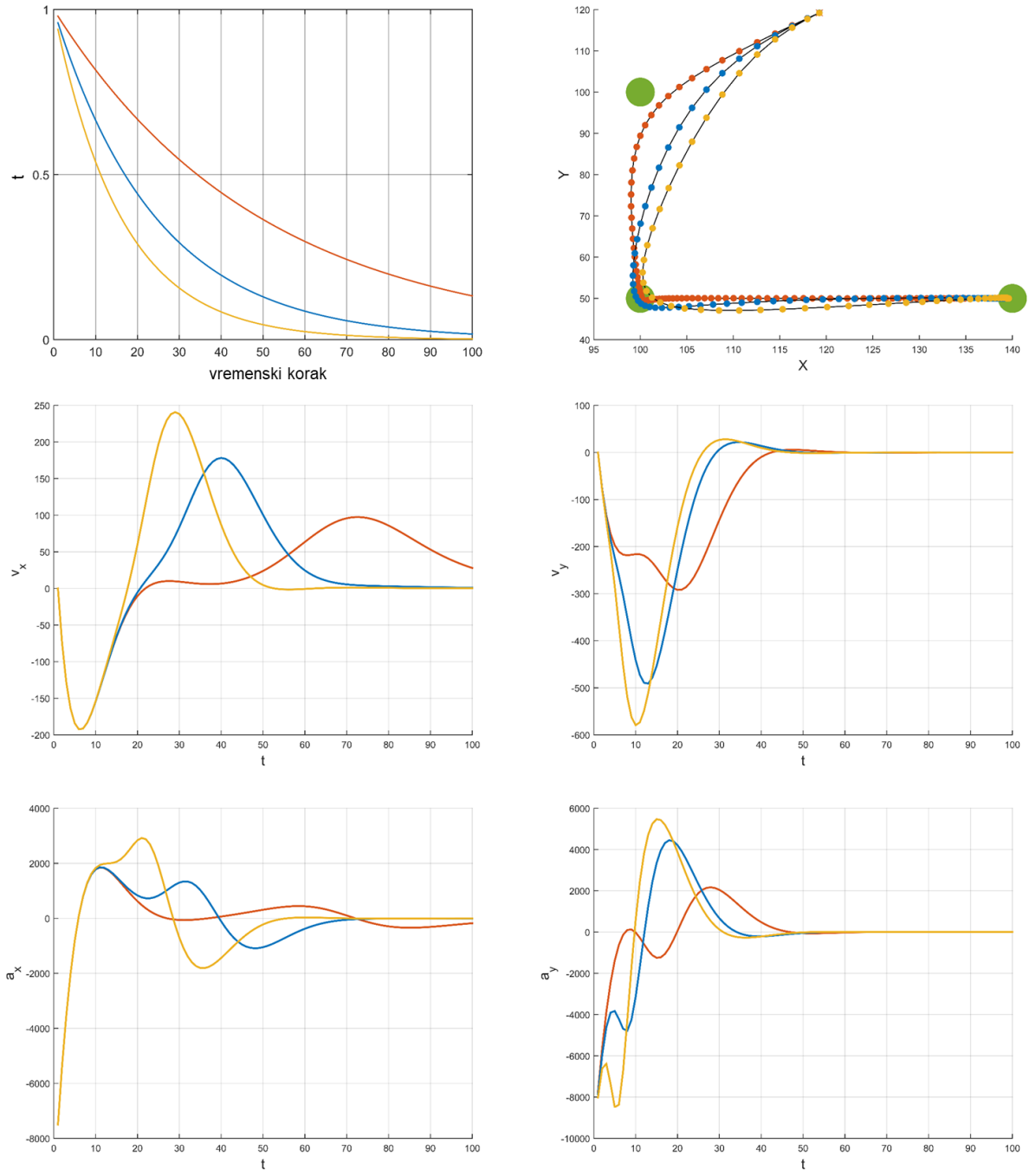
Funkcija oblika (eng. *forcing term*) zamijenjena je stvaranjem međusobne ovisnosti stanja i vremena. Ovo je postignuto tako da se trenutni cilj dinamičkog sustava DMP-a u svakom vremenskom trenutku preračunava prema vremenski zavisnoj vjerojatnosnoj funkciji unaprijed definiranih stanja. Ovaj pristup originalno je prezentiran u [86], kako bi se u trajektoriju kodirala informacija o krutosti za robote koji su upravljani prema zadanom momentu. Za potrebe ovoga pristupa se ne koristi spomenuta matrica krutosti. Akceleracija sustava se računa kao suma *N* proporcionalnih diferencijalnih sustava, gdje *N* označava broj stanja, y i $\dot{y}$ su trenutna pozicija i brzina sustava, a a_y i b_y su parametri pojačanja.

$$\tau^2 \hat{\ddot{y}} = \sum_{i=1}^{K} h_i(t) \cdot a_y\left[b_y(\mu_i - y) - \tau \dot{y}\right] \tag{3.2}$$

Već spomenuta ovisnost stanja o vremenu je ostvarena korištenjem eksponencijalno padajuće funkcije $\dot{t} = -\alpha_x \cdot t$ kao funkcije vremena, gdje je varijabla t inicijalizirana sa $t = 1$ na početku svakog gibanja i postupno konvergira prema nuli u svakom vremenskom koraku. Brzina pada tj. konvergencije ove funkcije prema nuli definirana je parametrom pojačanja α_x. Ova vremenska funkcija vezana je za dinamiku sustava pomoću seta od *N* normalnih distribucija $\mathcal{N}(t;\ {\mu_i}^t, \Sigma_i)$ sa jednoliko raspoređenim srednjim vrijednostima distribucija od nula do jedan, gdje svaka pojedinačna distribucija predstavlja jedno stanje iz vektora stanja μ. Vremenski ovisna vjerojatnost da se sustav nalazi u određenom stanju dana je izrazom (3.3).

$$h_i(t) = \frac{\mathcal{N}(t;\ {\mu_i}^t, \Sigma_i)}{\sum_{k=1}^{K} \mathcal{N}(t;\ {\mu_k}^t, \Sigma_k)} \tag{3.3}$$

Primjer korištenja ovog modela kod aproksimacije stanja dan je na slici 3-5., gdje je vidljivo zadržavanje kontinuiranih profila brzina i akceleracija svojstvenih za DMP model.



Slika 3-5. Primjer trajektorija dobivenih DMP-om stanja – aproksimacija triju stanja (zelene točke) u dvodimenzionalnom prostoru.

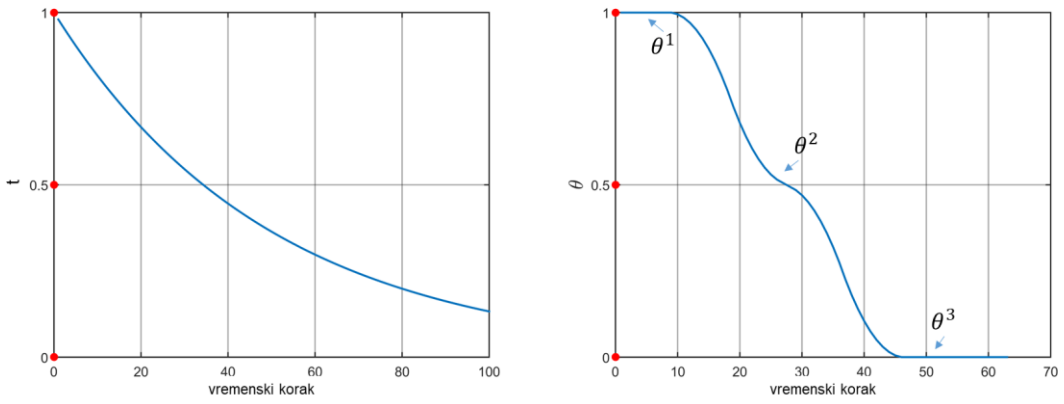
Korištenjem ovog modela može se postići sekvencijalno praćenje proizvoljnih stanja definiranih ili u operacijskom (kartezijskom) ili konfiguracijskom sustavu robota. Nadalje, može se primijetiti kako oblik eksponencijalno padajuće funkcije vremena određuje koliko će se vremena „dodijeliti“ određenom stanju unutar vektora stanja. Korištenjem standardnog oblika ove funkcije,

vidljivo je da funkcija ima veći prirast pada na početku funkcije (nakon inicijalizacije sa vrijednosti 1), što znači da stanja koja se pojavljuju u ovom vremenskom razdoblju imaju manju važnost nego stanja koja se pojavljuju na kraju funkcije, gdje je prirast pada vremena puno manji kako funkcija konvergira vrijednosti 0.

3.3.1. Modifikacija vremenske funkcije

Kako bi se postigla aproksimacija ovisna o eksplicitno definiranoj važnosti stanja, ovdje će se predložiti modifikacija standardne eksponencijalno padajuće funkcije tako da se varijabilna „količina“ vremena može dodijeliti svakom stanju pojedinačno. Metodologija je izvorno predstavljena u radu „*Task Dependent Trajectory Learning from Multiple Demonstrations Using Movement Primitives*“, [92].

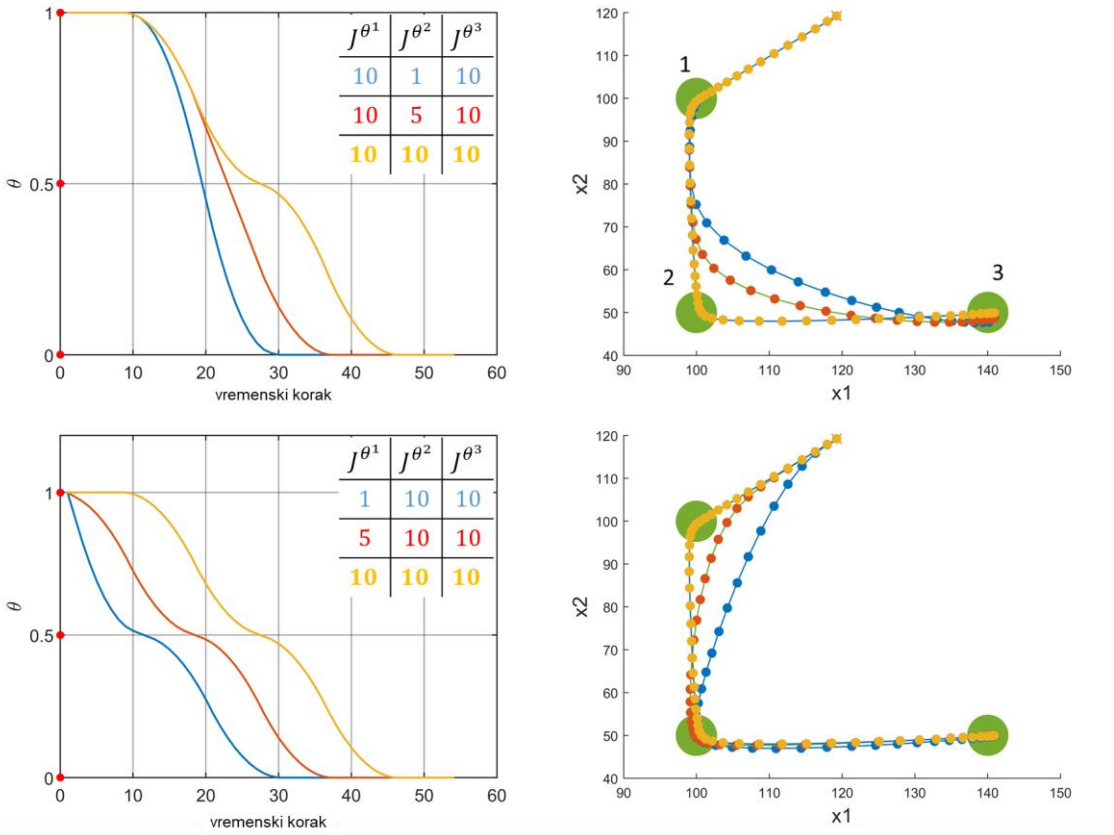
Spomenuta modifikacija vremenske funkcije ostvarena je na sljedeći način. Svakom stanju μ_i kojem je prethodno opisanim pristupom dodijeljena vremenski pozicionirana normalna distribucija sa središtem u μ_i^t, sada je umjesto jedne skalarne vrijednosti vremena, dodijeljen vektor istih vrijednosti vremena. Umjesto vremenske eksponencijalne funkcije koja daje vrijednosti vremena, formiran je vektor vremena za svako stanje N, $\theta^n = \{x_j\}_{j=1}^{J}$, $x_j \epsilon\ \mu_i^t$, gdje broj stanja u vektoru – J, ovisi o eksplicitno zadanoj vrijednosti. Puni vektor vremena je onda dobiven ulančavanjem svih pojedinačnih vremenskih vektora $\theta = \{\theta^1, \theta^2, \dots, \theta^N\}$. Rezultat ovoga je vremenska funkcija koja je ovisna o vremenu eksplicitno dodijeljenom svakom stanju. Na slici 3-6. prikazana je razlika između standardne eksponencijalne i modificirane vremenske funkcije za tri stanja. Duljine vektora θ^1, θ^2i θ^3 zadane su proizvoljno.



Slika 3-6. Modificirana vremenska funkcija.

Na slici 3-7. prikazano je kako ova modifikacija omogućuje veću kontrolu nad oblikom trajektorije. Otvoreni parametri u modificiranoj vremenskoj funkciji mogu se iskoristiti za definiranje stupnja

aproksimacije svakog pojedinog stanja. Ovo pruža znatno veću mogućnost varijantnost i modularnost DMP zapisa.



Slika 3-7. Promjena parametara modificirane vremenske funkcije.

3.3.2. Modeliranje trajektorije na osnovu bitnosti

Kod učenja trajektorije, postoje istraživanja koja pokušavaju iz trajektorija izvući najbitnije dijelove s ciljem ostvarivanja različitih korisnih svojstava reproduciranih trajektorija. Primjerice, u područjima trajektorije gdje je uočena jako mala razlika između svih demonstriranih slučaja, pretpostavlja se potreba za striktnijim poštivanjem demonstriranih prostornih ili orijentacijskih ograničenja. Ako se ovo promatra kroz prizmu robotskog djelovanja, dijelovima trajektorije koji su potpuno jasni (velika podudarnost u demonstracijama) može biti dodijeljena velika sigurnost. Tako je u [86] prikazan model trajektorije koji sigurnost modelira u obliku krutosti robotskog manipulatora. Veću krutost robotu se daje u bitnim dijelovima trajektorije, dok je robot podatniji i manje krut u dijelovima trajektorije gdje je sigurnost manja. Ovo može bit korisno kod kombiniranja naučenih vještina sa interakcijom-čovjeka i robota, gdje čovjek može više intervenirati u dijelovima koji nisu jako definirani samim zadatkom.

U ovom poglavlju predložiti će se jedan ovakav model ponašanja. Biti će demonstrirano *M* trajektorija u istom koordinatnom sustavu, koje u nekim dijelovima gibanja imaju zajedničke dijelove, a u drugima se značajno razlikuju jedna od druge. Na osnovu analize ovih trajektorija, ideja je reproducirati novu trajektoriju započinjući kretanje iz neke proizvoljne točke, a pri tome zadržavajući najbitnije dijelove trajektorije. U tu svrhu će se predloženi modificirani model DMP trajektorije spregnuti sa demonstracijama i analizom aktivnosti koordinatnih sustava prikazanu u poglavlju 2.6.1.

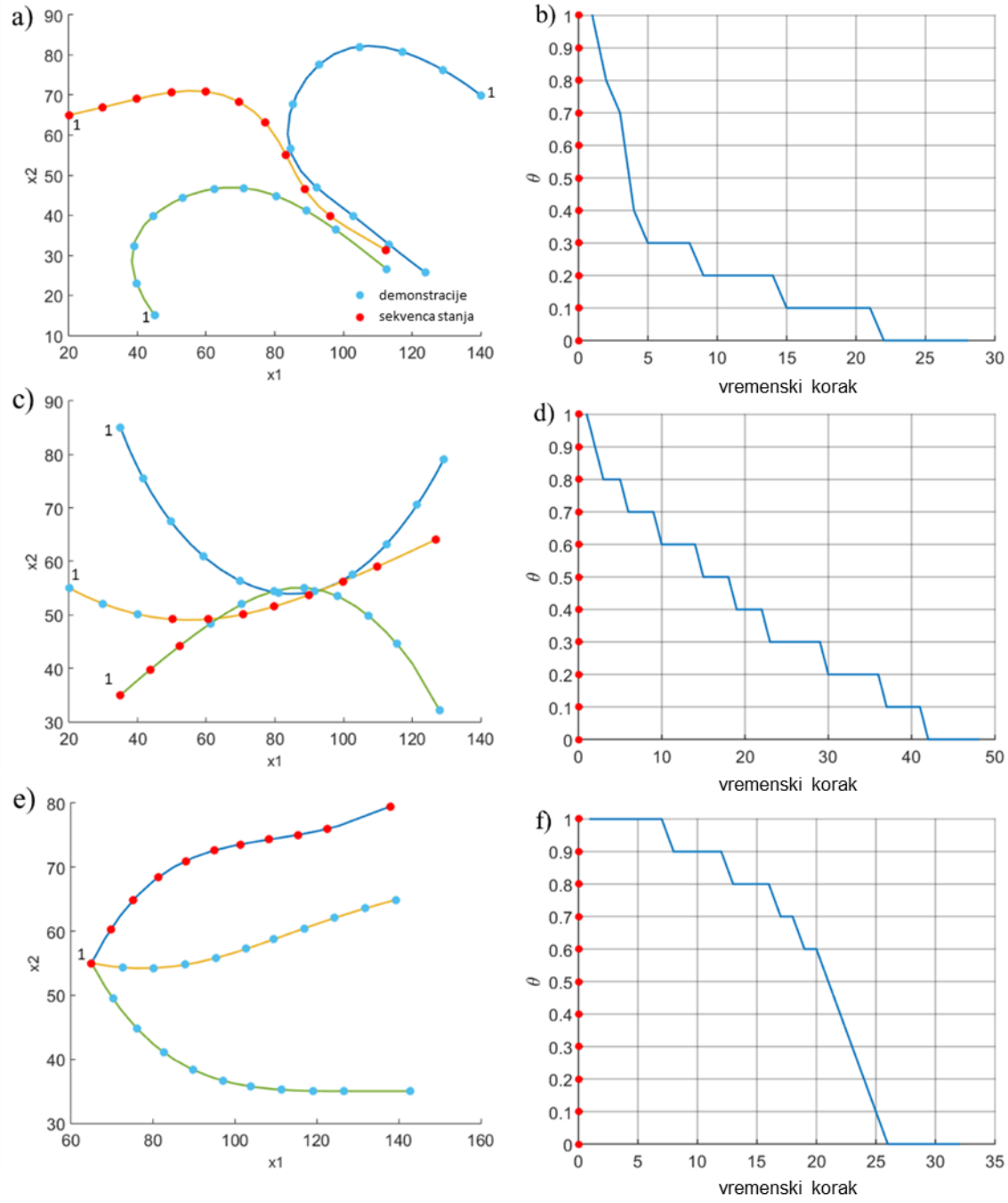
Svaka demonstrirana trajektorija sačinjena je od demonstriranih parametara trajektorije *x* za svaki stupanj slobode. Ovdje su uzimane u obzir ravninske trajektorije sa samo dva stupnja slobode. Demonstrirane trajektorije su ponovno uzorkovane proizvoljno definiranim brojem stanja N kako bi se dobila jednolika raspodjela stanja po trajektorijama prema postupku opisanom u 2.6.2.. Jedini uvjet kod izbora broja stanja je da on mora biti dovoljno velik kako bi se iz uzorkovane trajektorije mogle vidjeti sve bitne tranzicije izvorno demonstriranih trajektorija. Ovime je broj točaka trajektorije značajno smanjen i definirana je matrica $T_{i,j} = \{\{x_j\}_{j=1}^{N}\}_{i=1}^{M}$, koja sadrži stanja svih demonstriranih trajektorija.

Prije reprodukcije svake trajektorije, ovisno o početnoj poziciji trajektorije, odabiru se kandidati stanja po kojima će se trajektorija gibati μ_i. Ista se odabiru prema kriteriju minimalne udaljenosti trenutnog stanja do sljedećeg stanja tj. uvijek se kao sljedeća stanja odabiru ona koja su najbliža. Ovakva sekvenca inicijalnih stanja rezultira vektorom stanja μ, čiji je izraz dan u (3.4) gdje je *z* trenutno stanje trajektorije, a *y* je vektor svih sljedećih mogućih stanja.

$$\mu_i = \{y \rightarrow \min_{\substack{y \in T_{i+1} \\ z \in T_i}} ||y - z||\}_{i=1}^{N} \tag{3.4}$$

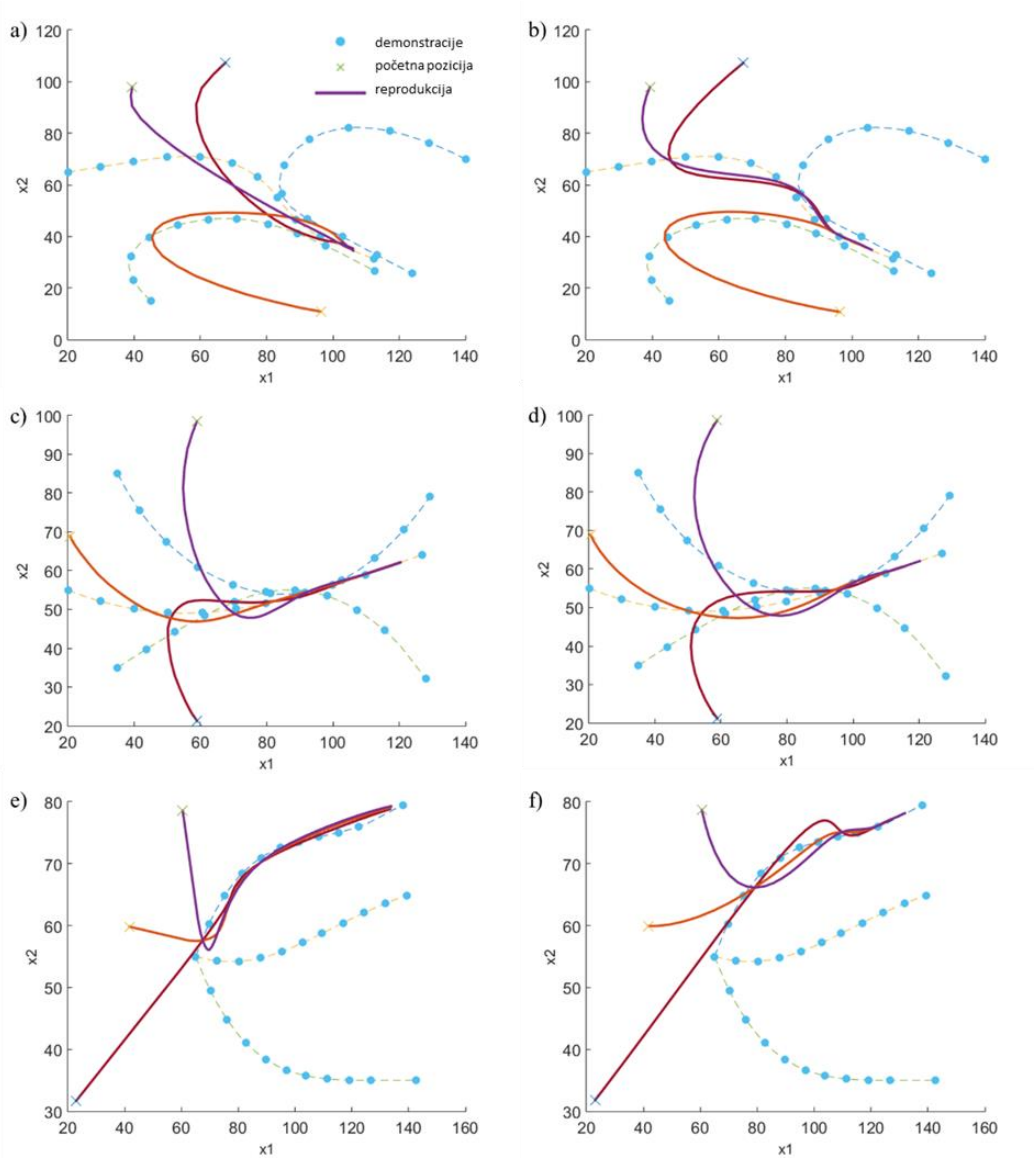
Važnost svakog stanja određena je izračunom aktivnosti (varijabilnosti) u odnosu na druga stanja trajektorije. Varijabilnost σ, dana je kao otežana Euklidska udaljenost zbroja udaljenosti trenutnog stanja do svih mogućih stanja kako je opisano u poglavlju za analizu demonstracija (2.6.1.). Ovo rezultira vektorom koji sadrži po jednu skalarnu vrijednost koja opisuje varijabilnost (aktivnost/važnosti) za svako stanje u inicijalnoj sekvenci gibanja. Ovaj vektor se onda koristi za definiranje modificirane funkcije vremena prema metodologiji prikazanoj u 3.3.1. - $J^{\theta^N} = f(\sigma)$. Na slici 3-8. prikazane su po tri demonstrirane trajektorije za tri različite konfiguracije. Prva konfiguracija pokazuje trajektorije koje imaju zajedničke krajnje dijelove, druga konfiguracija ima zajedničke srednje dijelove trajektorija, dok treća konfiguracija prikazuje trajektorije koje imaju

zajednički početak. Uz svaku konfiguraciju, prikazana je pripadajuća vremenska funkcija slučajno odabranu sekvencu stanja.



Slika 3-8. Dvodimenzionalne trajektorije u kartezijskom prostoru (a, c, e) sa odabranom sekvencom trajektorije koja se sastoji od 11 stanja označenih crvenim točkama. Pripadajuće modificirane vremenske funkcije dane su u (b, d, f) gdje crvene točke označuju vremensku raspodjelu stanja. Početne pozicije označene su indeksom 1. Vidljivo je da stanja sa manjom varijabilnosti (većom bitnosti) imaju dodijeljeno više vremenskih koraka u vremenskoj funkciji.

Na slici 3-9. prikazani su rezultati reprodukcije trajektorija iz različitih početnih stanja. Dana je usporedba reprodukcije sa standardnom eksponencijalno padajućom vremenskom funkcijom i reprodukcije sa modificiranom vremenskom funkcijom. Vidljivo je da trajektorije koje koriste predloženu modifikaciju, bolje prate stanja (dijelove trajektorija) koja su zajednička u svim demonstracijama. Ovo uzrokuje kraće i izravnije trajektorije te trajektorije koje bolje aproksimiraju bitne dijelove trajektorija.

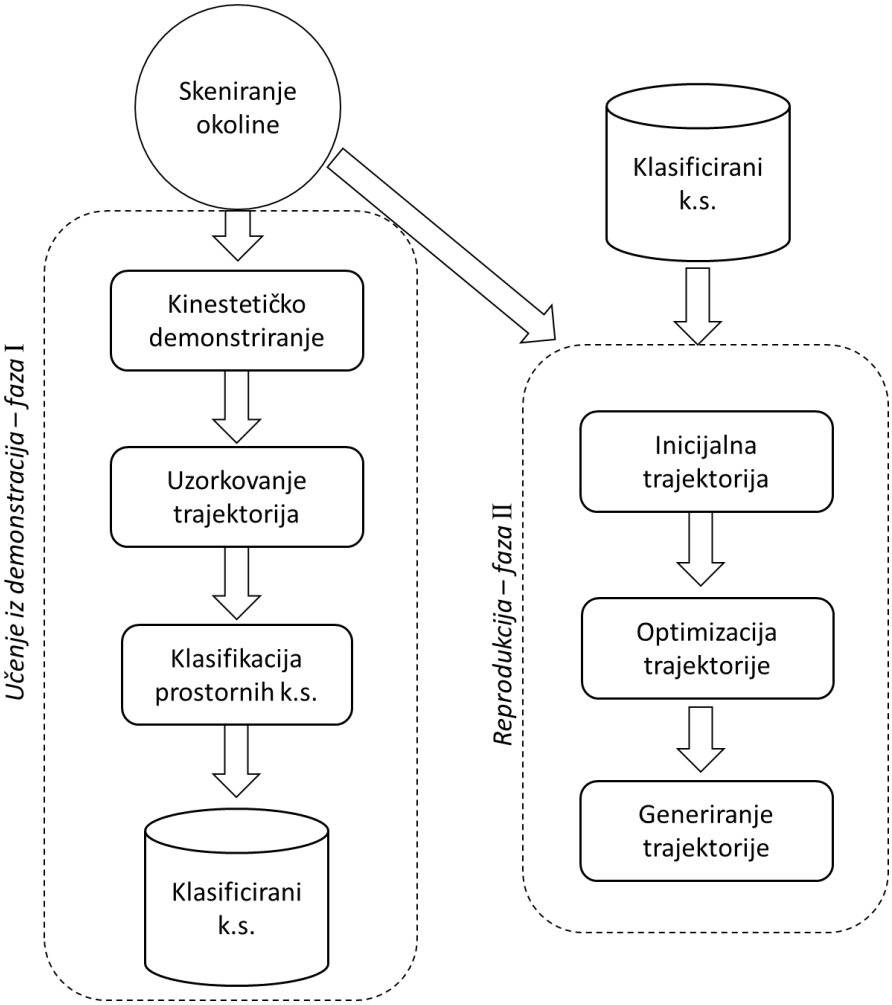


Slika 3-9. Reprodukcija trajektorija za tri različita scenarija. Svaki se sastoji od tri demonstrirane trajektorije (prikazane točkastim linijama), a reproducirane trajektorije su pokazane za tri različita početna položaja prikazane punim linijama. Reprodukcije sa modificiranom vremenskom funkcijom dane su u a), c), e), dok su one sa standardnom eksponencijalnom prikazane u b), d), f).

3.4. Model trajektorije zasnovan na analizi točaka i optimizaciji

Osim mogućnosti analize demonstracija, modeli za učenje iz demonstracija moraju imati mogućnost ekstrapolacije naučenog znanja i primjene u realnoj okolini. Ukoliko je predmet demonstracija bila trajektorija djelovanja robota u prostoru, tada naučeni model mora isto tako omogućiti generiranje trajektorija ali u do sad neviđenim situacijama. Što je mogućnost modela da generalizira znanje u novim situacijama veća, to se model smatra kvalitetnijim.

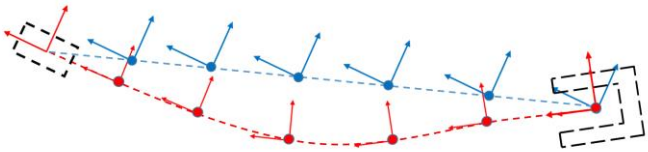
U ovom poglavlju predložiti će se model trajektorije koji je sposoban iskoristiti informacije dobivene iz analize demonstracija predstavljene u 2.6. – aproksimirati aktivne koordinatne sustave (k.s.), dok zaobilazi koordinatne sustave klasificirane kao prepreke. Metodologija je izvorno predstavljena u znanstvenom radu „*Learning from Demonstration Based on a Classification of Task Parameters and Trajectory Optimization*“, [93].



Slika 3-10. Dijagram toka predložene metodologije. Vizijski sustav služi kao ulaz kod faze učenja (I) kao i kod faze reprodukcije (II).

3.4.1. Model trajektorije zasnovan na optimizaciji

Optimizacija trajektorije u odnosu na neke parametre cilja tradicionalno je područje u robotici. Ti parametri su najčešće duljina trajektorije, vrijeme izvršavanja, potrošnja energije te razni kinematski parametri bitni za izvršavanje trajektorije na robotu. Karakteristika ovog pristupa je da se inicijalna trajektorija iterativno mijenja i oblikuje tako da se zadovolje postavljeni parametri cilja. Zbog ograničenja optimizacijskih metoda i računalnih mogućnosti nakon optimizacije se najčešće mogu dobiti samo lokalno optimalna rješenja. Stoga ovi pristupi značajno ovise o načinu inicijalizacije. Postoje dva osnovna načina optimizacije trajektorija. Moguće je optimizirati same geometrijske parametre trajektorije ili unositi šum u trajektoriju tako da se postigne njezina promjena.

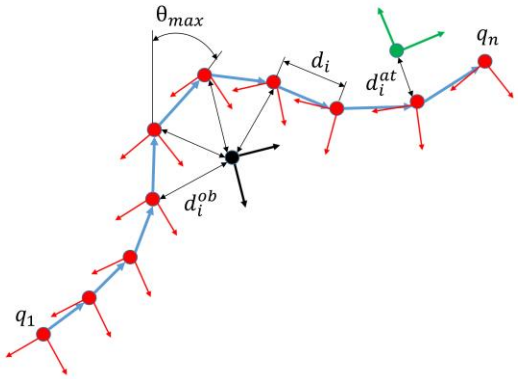


Slika 3-11. Primjer inicijalne trajektorije (plavo) i trajektorije nakon optimizacije (crveno).

Kod generiranja trajektorije za novu situaciju u robotskom okruženju, pretpostavlja se da su na raspolaganju podaci o položajima koordinatnih sustava objekata u radnom prostoru robota i početnoj poziciji robota. Temeljeno na analizi aktivnosti koordinatnih sustava u odnosu na trajektorije iz demonstracija, iz poglavlja 2.6.1, ovi k.s. su klasificirani u privlačne i odbojne. Uz ovo, jedan k.s. je klasificiran kao ciljni. Pomoću ovih informacija generira se vektor inicijalne trajektorije Q veličine q x N. Ovdje q predstavlja broj stupnjeva slobode trajektorije (primjerice, $q = 3$ za ravninsku trajektoriju koju opisuje jedan kut i dvije translacijske koordinate), a N je broj koordinatnih sustava u optimiranoj trajektoriji. Inicijalna trajektorija je oblikovana u obliku N jednoliko raspoređenih koordinatnih sustava od početne pozicije robota do klasificirane ciljne pozicije trajektorije (slika 3-11.). Oblik inicijalne trajektorije se onda u sljedećoj fazi mijenja kako bi se dobile zadovoljavajuće karakteristike opisane predstavljenim modelom, pomoću optimizacijskog procesa koji će biti opisan u nastavku.

Za optimizacijski proces, razvijena je višekriterijska funkcija cilja koja je sposobna optimizirati inicijalni vektor trajektorije Q, kako bi se stvorila osnova za novu trajektoriju. Funkcija cilja je sastavljena od više kriterija (C_1 – C_6) koji omogućuju stvaranje sekvence koordinatnih sustava koji poštuju model ponašanja dobiven analizom demonstracija. Uz ovo, funkcija cilja u

sebi sadrži kriterije koji omogućuju generiranje valjanih robotskih trajektorija. Ovi kriteriji biti će predstavljeni u nastavku ovog poglavlja.



Slika 3-12. Geometrijski parametri korišteni kod optimizacije prolaznih koordinatnih sustava

C_1 – Jednolik raspored koordinatnih sustava

Raspored koordinatnih točaka trajektorije trebao bi biti jednolik gledamo li Euklidsku udaljenost di između točaka trajektorije (slika 3-12.). Ovaj kriterij sprječava nastanak dijelova trajektorije sa gustim rasporedom točaka što se očituje manjim brojem točaka u drugim dijelovima trajektorije gdje je onda aproksimacija trajektorije vrlo loša. Ovaj kriterij je opisan kao standardna devijacija vektora duljina d:

$$C_1 = std(d(Q)). \tag{3.2}$$

C_2 – Maksimalni kut između vektora

Privlačne točke tvore segmente trajektorija. Jedan segment trajektorije čine dvije točke, a dva segmenta trajektorije su definirana sa tri uzastopne točke. Segmenti trajektorije mogu biti promatrani kao vektori (dvodimenzionalni ako je riječ o ravninskoj trajektoriji ili trodimenzionalni u slučaju trodimenzionalne trajektorije) između kojih se mogu odrediti kutovi (slika 3-12.). Kutovi između susjednih segmenata trajektorije ne smiju biti veliki, kako bi se spriječile nagle promjene smjera prilikom gibanja. Iz tog razloga ovdje je predložen kriterij prema kojem se minimizira najveći kut između dva segmenta, identificiran u trajektoriji θ_{max}. Kriterij je opisan kao

$$C_2 = max\ (atan2(\|\overrightarrow{q_i} \times \overrightarrow{q_{i+1}}\|, \overrightarrow{q_i} \cdot \overrightarrow{q_{i+1}})). \tag{3.3}$$

C_3 – Duljina trajektorije

Treći kriterij se odnosi na ukupnu duljinu trajektorije. Ona bi se trebala držati minimalnom kako bi se spriječile nepotrebno duga gibanja

$$C_3 = \sum d(Q). \tag{3.4}$$

C_4 – Parametar zadatka

Prema predloženom modelu ponašanja, trajektorija bi trebala prolaziti kroz prethodno identificirane privlačne točke. Ovo je postignuto minimiziranjem udaljenosti Euklidske udaljenosti d_i^{at} između najbliže točke trajektorije i privlačne točke

$$C_4 = min(d_i^{at}). \tag{3.5}$$

C_5 – Orijentacijski kriterij

Relativna orijentacija trajektorije u odnosu na privlačne trajektorije bi trebala biti jednaka kao i u slučaju demonstracija. Ovo je jako bitno kako bi se očuvale orijentacijske značajke pokazane u demonstracijama. Razlika između prosječne demonstrirane orijentacije u odnosu na privlačne točke se ovdje računa kao norma udaljenosti između rotacijskih dijelova privlačnih koordinatnih sustava i koordinatnih sustava trajektorije. Ova razlika se minimizira preko sljedećeg kriterija:

$$C_5 = \|rot(q_i) - rot(d_i^{at})\|. \tag{3.6}$$

C_6 – Kriterij kolizije

Niti jedna točka trajektorije ne bi trebala proći kroz klasificiranu prepreku. Euklidska udaljenost između prepreke i svih točaka trajektorije d_i^{ob} se računa i zapisuje u jedan vektor. Elementi ovog vektora duljina su onda otežani eksponencijalnom funkcijom koja veće vrijednosti daje točkama koje prolaze jako blizu identificiranih prepreka. Kako bi se ostvarila funkcija izbjegavanja prepreka, minimizira se suma zbroja svih elemenata ovako otežanog vektora udaljenosti, $C_6 = \sum x^{d_i^{ob}}$, gdje $x \in \{0.1 \dots 1\}$, ovisno o tome s kolikom marginom se želi prolaziti kraj prepreka.

Finalna funkcija cilja predložena za optimizaciju linearnih trajektorija kao višekriterijska funkcija opisanih kriterija dana je izrazom:

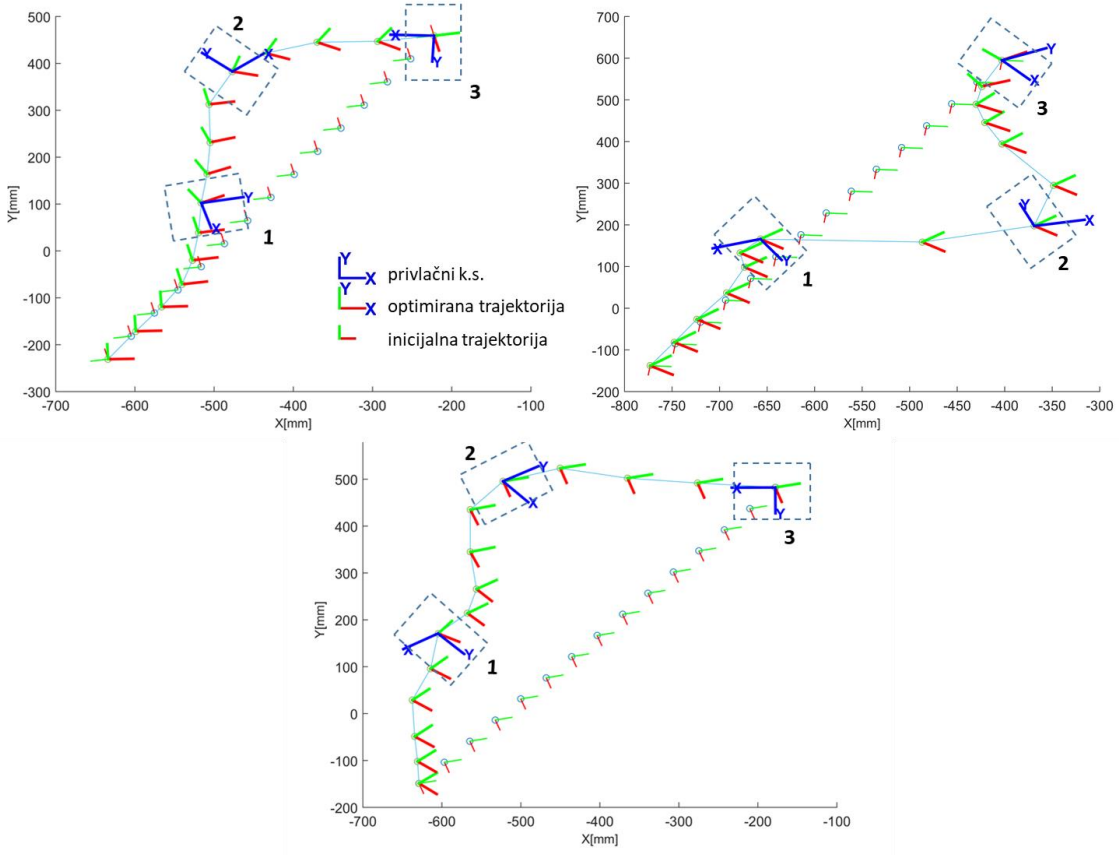
$$C = \min_Q \sum_{i=1}^{6} w_i C_i(Q). \tag{3.7}$$

Ova funkcija cilja minimizira se CMA-ES algoritmom (eng. *Covariance matrix adaptation evolution strategy*) [94],[84]. To je robusna metoda optimizacije koja ne koristi derivacije funkcije cilja i sposobna je optimizirati stohastičke funkcije cilja. Podešavanje CMA-ES algoritma se ostvaruje putem slobodnog parametra σ koji predstavlja inicijalni korak pretrage u obliku širine distribucije pretrage. Pravilnim podešavanjem ovog parametra, može se postići vrlo brza konvergencija uz izbjegavanje lokalnih minimuma. Rezultati i način rada ove metode biti će opisani u sljedećem poglavlju.

3.4.2. Simulacijski rezultati

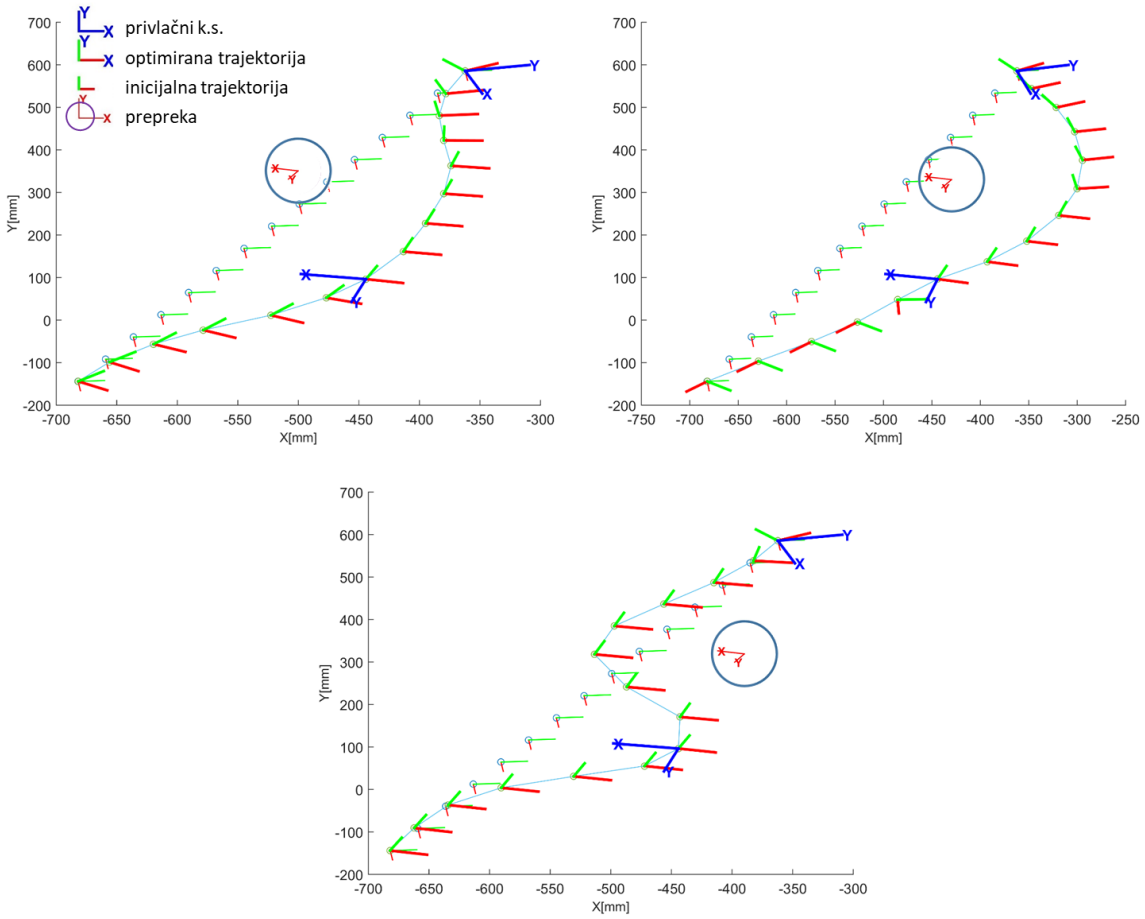
Kao što je opisano u prethodnom poglavlju prvi korak kod generiranja nove trajektorije je stvaranje linearne inicijalne trajektorije sa N jednoliko raspoređenih prostornih koordinatnih točaka (sustava) od početne pozicije do ciljne pozicije. Za to je potrebno prethodno klasificirati ciljnu točku na osnovu demonstracija, što omogućuje klasifikacijska metoda opisana u 2.6.1. Nakon toga, na temelju ostalih podataka iz klasifikacijskog modela, ova trajektorija se optimizira tako da se CMA-ES algoritmom mijenjaju pozicije i orijentacije koordinatnih sustava kako bi se minimizirala predstavljena funkcija cilja C.

Prvi primjer koji će se validirati je slučaj kada postoje tri privlačna objekta (koordinatna sustava) klasificirana iz demonstracija. Na slici 3-13. predstavljene su tri konfiguracije ovih objekata sa različitim polaznim točkama i formiranim inicijalnim trajektorijama. Optimizacijom inicijalnog rasporeda točaka prema funkciji cilja vidljivo je da su dobivene nove trajektorije, koje prolaze kroz privlačne objekte.



Slika 3-13. Generirane sekvence koordinatnih sustava za tri klasificirana prolazna koordinatna sustava. Prikazane su tri različite konfiguracije. Korištene su sekvence od 15 koordinatnih sustava.

Drugi primjer (slika 3-14.) obuhvaća dva koordinatna sustava koja su klasificirana kao privlačna i jedan kao prepreka. U tri različite konfiguracije pokazane su inicijalne trajektorije i one dobivene poslije optimizacije prema funkciji cilja. Vidljivo je da optimirana trajektorija uspješno zaobilazi prepreke, dok aproksimira privlačne koordinatne sustave.



Slika 3-14. Generirane sekvence za dva prolazna k.s. i jedan koji je prepreka. Prikazane su tri različite konfiguracije zadatka.

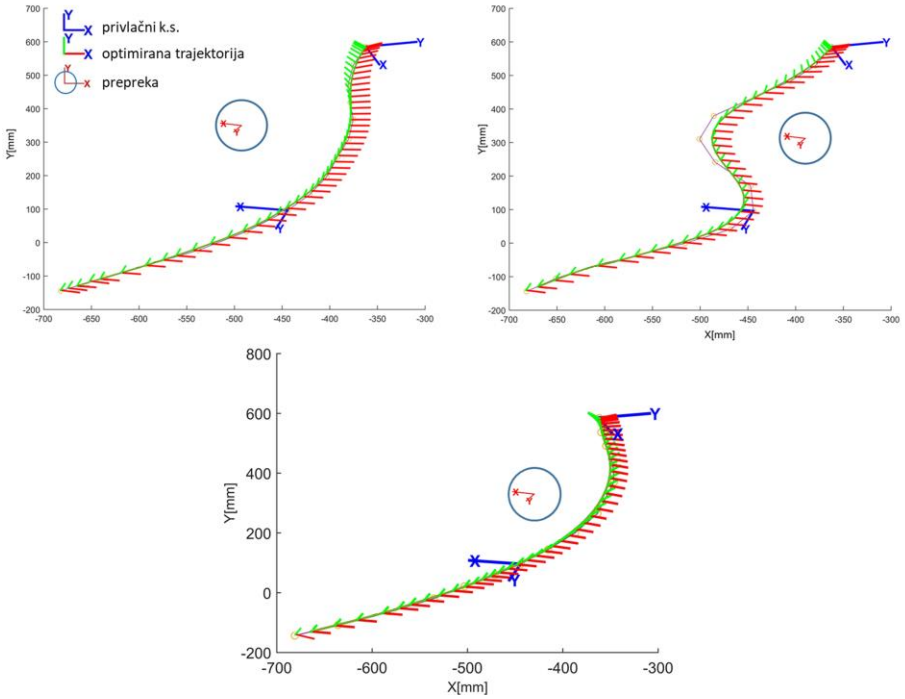
Vrijeme trajanja optimizacije trajektorija za ova dva pokazana primjera na standardnom računalu sa i7-6700HQ procesorom i 16 GB RAM iznosilo je od 20 do 60 sekundi.

3.4.3. Aproksimacija state-DMP modelom

Sekvenca točaka trajektorije dobivena opisanim optimizacijskim postupkom predstavlja grubu aproksimaciju trajektorije prema ustanovljenom modelu ponašanja. Ona u biti predstavlja niz pozicija i orijentacija koordinatnih sustava robotskog vrha alata u odnosu na karakteristične prostorne koordinatne sustave i nije prikladna za izvršnu primjenu na pravom robotskom sustavu.

Kako bi se ovaj opisni oblik trajektorije pretvorio u trajektoriju koja je prikladna za primjenu na pravom manipulatoru, predložiti će se još jedan korak koji će pronađenu trajektoriju pretvoriti u trajektoriju prikladnu za reprodukciju na robotu. Za ovo će se koristiti prethodno opisani model DMP stanja.

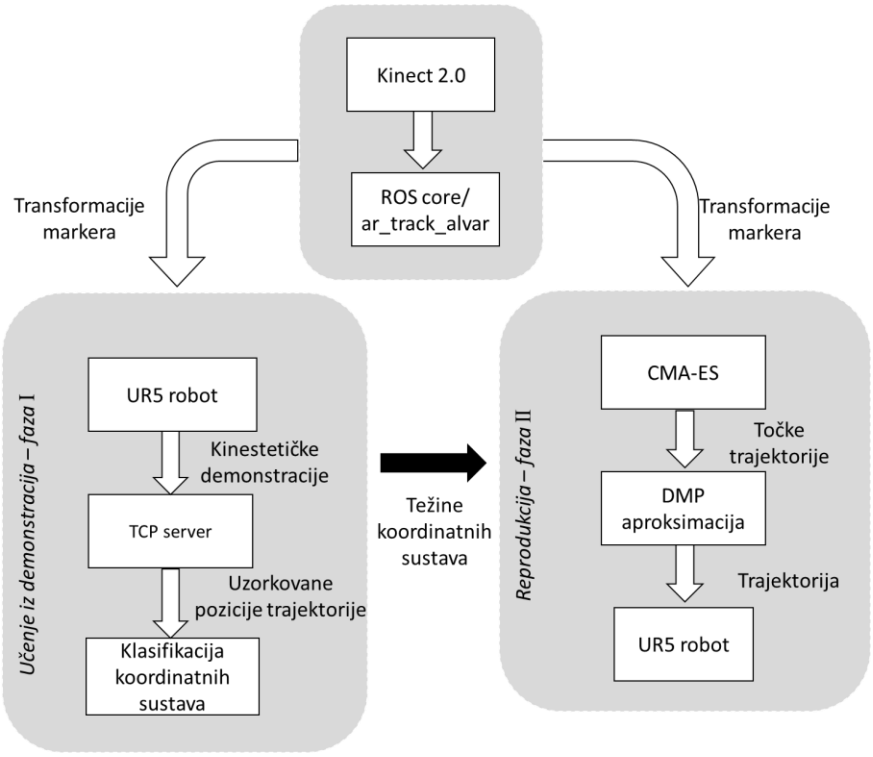
Na slici 3-15. vidljiva je ovakva aproksimacija optimiranih točaka trajektorije za prethodno prikazane konfiguracije. Vidljivo je da trajektorija ima puno veću frekvenciju točaka i odlikuje ju kontinuitet.



Slika 3-15. Sekvenca trajektorije aproksimirana DMP modelom koja predstavlja kontinuiranu trajektoriju.

3.4.4. Eksperimentalna provjera

Eksperimentalni sustav za provjeru ove metodologije (slika 3-16.) naslanja se na onaj razvijen za potrebe učenja iz demonstracija. U demonstracijskoj fazi, UR5 robot korišten je na isti način za kinestetičko pokazivanje, a Kinect vizijski sustav za dobivanje prostornog razmještaja predmeta u prostoru u odnosu na robota. U reprodukcijskoj fazi su podaci dobiveni iz učenja iz demonstracije zajedno sa novim pozicijama objekata korišteni kao ulaz prilikom generiranja nove trajektorije. Generirana trajektorija je potom slana kao kompletno generirani program robotskoj ruci.



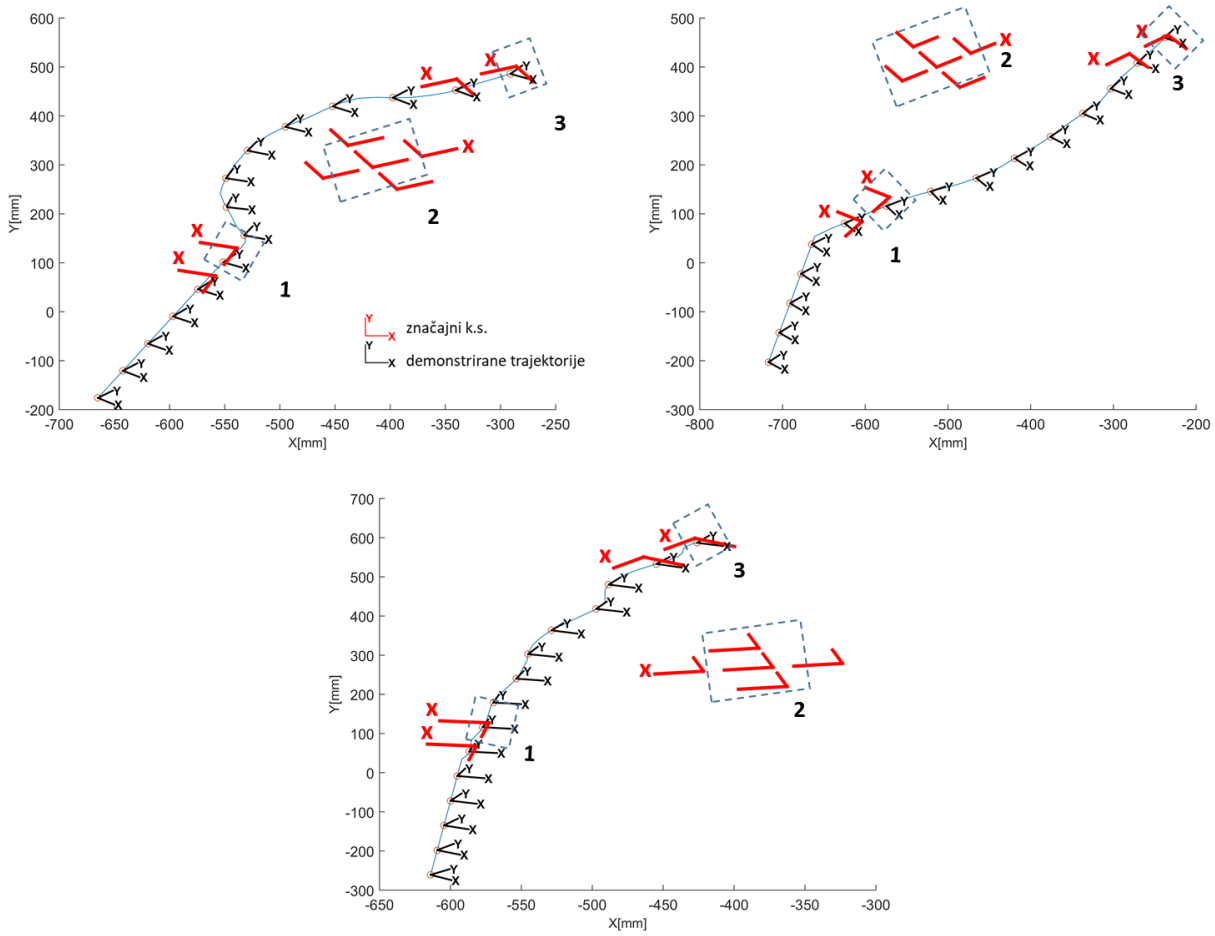
Slika 3-16. Blok dijagram eksperimentalnog postava.

Dobivena trajektorija u sebi implicitno sadrži informacije o brzinama i akceleracijama, što je karakteristika DMP modela. Za izvršavanje ovakve trajektorije na pravom robotu postoje različite opcije. Jedna je eksplicitno izražavanje brzine u određenim interpoliranim pozicijama trajektorije, te korištenje standardnog pozicijskog kontrolera za prolazak po listi unaprijed poznatih točaka, koji je dostupan na gotovo svim industrijskim robotskim manipulatorima. Kod kolaborativnih robotskih ruku često su dostupna sučelja za direktnu komunikaciju upravljačke jedinice robota sa eksternim uređajima na razini realnog vremena. Putem istih, moguće je slati upravljačke naredbe (položaj, brzina ili akceleracija zglobova) robotskoj ruci, pri čemu se zaobilazi standardno dostupni tvornički ugrađeni algoritam za planiranje kretanja. U osnovi, kod korištenja ovih sučelja se robotskoj ruci mora u svakom vremenskom trenutku zadati odgovarajuća naredba kojom se ostvaruje kretanje. Ovu opciju za slučaj pozicijskog upravljanja ima i robotska ruka *Universal robot* UR5. Budući da je generirana trajektorija iz već spomenutih razloga potpuno prilagođena ovako direktnom izvođenju, za verifikaciju trajektorije korištena je ova opcija. Robotu je slana „*servoj"* naredba koja u osnovi koristi pozicijske kontrolere za svaki od zglobova robota koje sinkronizira putem eksplicitno zadanog vremena izvršavanja *t*.

servoj(*q*, *a*, *v*, *t*=0.008, *lookahead_time*=0.1, *gain*=300) [95].

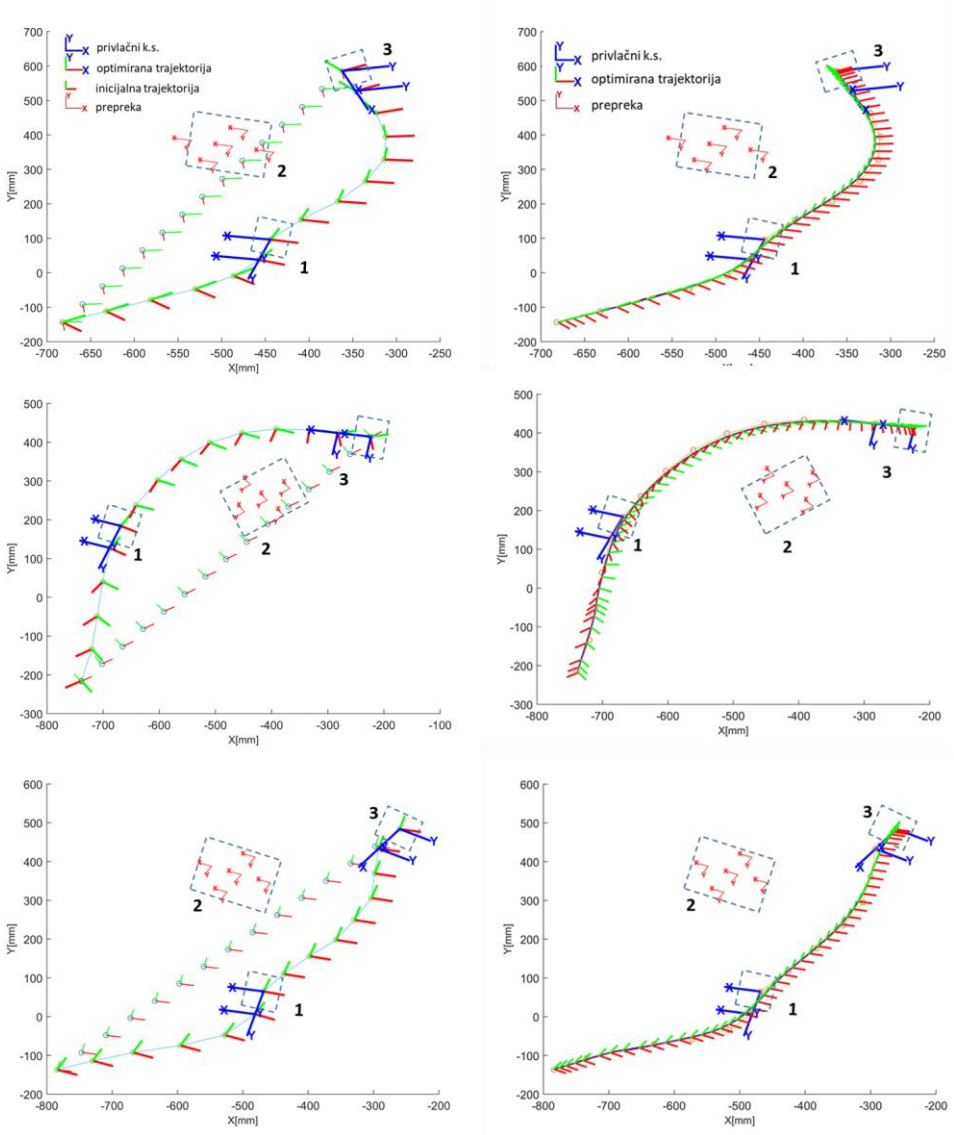
Budući da je trajektorija unaprijed poznata, korišten je *batch* način slanja, svih naredbi odjedanput. Kako je DMP trajektorija u kartezijskom sustavu, jedini dodatni međukorak bio je pretvaranje svih pozicija trajektorije putem inverzne kinematike u konfiguracijski prostor robota.

Za evaluaciju pristupa korišten je zadatak odguravanja predmeta na stolu sa neke početne lokacije na neku ciljnu pred-definiranu poziciju. U zadatku su korištena tri objekta, od kojih je jedan kutija kojom je robot manipulirao (označena sa brojem 1). Drugi predmet (2) je prepreka koju je robot izbjegavao u svim demonstracijama, treći predmet (3) je ciljna pozicija, koji je zapravo samo papirnati marker na podlozi. Demonstrirane trajektorije kao i prvotni položaji objekata u radnom prostoru robota za tri demonstracije vidljivi su na slici 3-17.



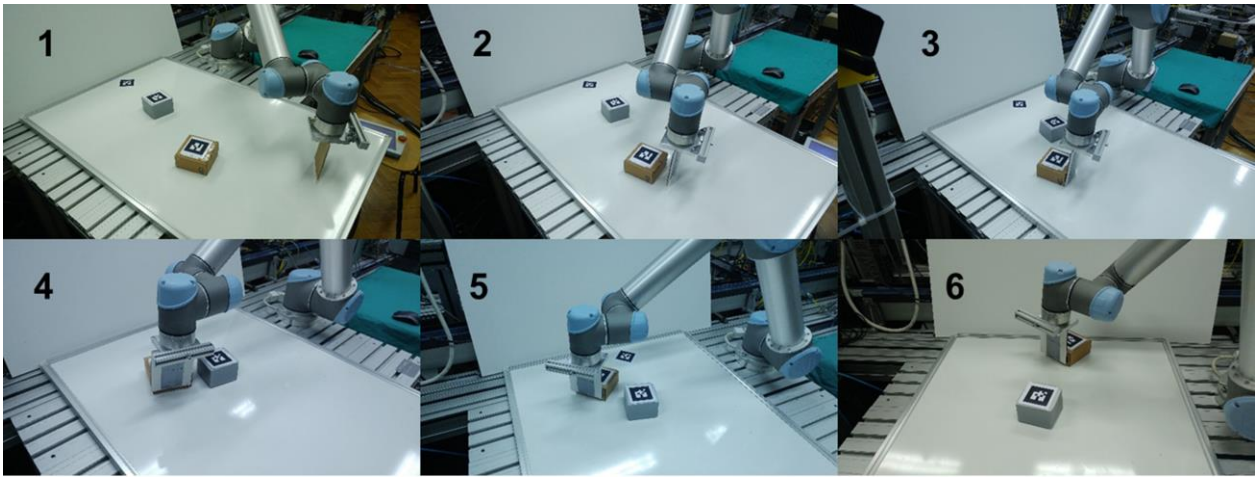
Slika 3-17. Demonstracije zadatka odguravanja. Objekt 1 predstavlja koordinatni sustav predmeta koji se odgurava. Objekt 2 je predmet koji predstavlja prepreku, a objekt 3 je ciljna lokacija na koju se predmet 1 želi odgurati.

Na slici 3-18. prikazane su generirane trajektorije za tri nove konfiguracije ovog zadatka. Vidljivo je da trajektorije poštuju pozicijske kao i orijentacijske zahtjeve vidljive u demonstracijama. Trajektorije prolaze objektima 1 i 3 te njihovim prilaznim točkama dok istovremeno pasivno zaobilaze koordinatne sustave predmeta 2. Orijentacijski zahtjevi se odnose na orijentaciju robotskog alata u blizini koordinatnih sustava 1 i 3. U novo generiranim trajektorijama sačuvan je relativni odnos između ovih orijentacija definiran u demonstracijama. Za ovo ponašanje zaslužan je *orijentacijski kriterij* (C_5), dok je za aproksimaciju privlačnih točaka zaslužan kriterij *parametra zadatka* (C_4).



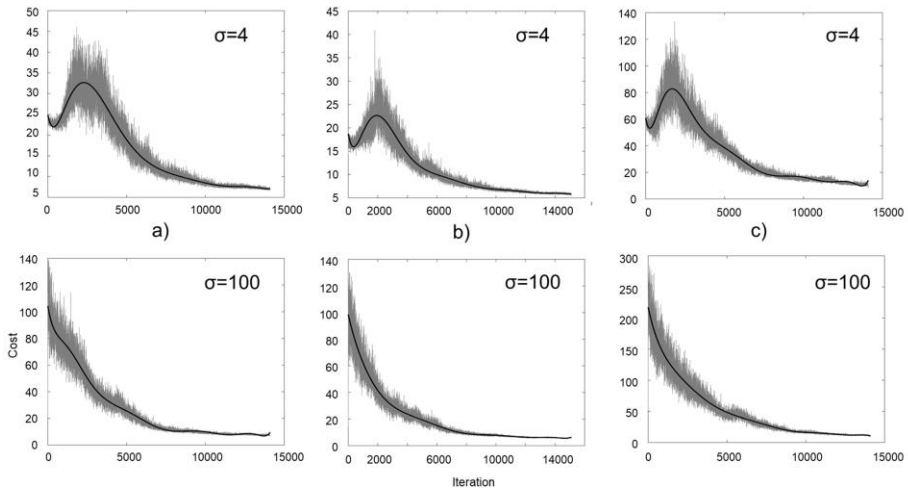
Slika 3-18. Generirana sekvenca trajektorija za rješavanje zadatka odguravanja (lijevi stupac). Aproksimacija sekvenci k.s. DMP modelom (desni stupac). Model je testiran za tri različite konfiguracije.

Kako je prikazano na slici 3-18., optimirane trajektorije su isto tako aproksimirane DMP trajektorijama kako bi bile izvedive na UR5 robotskoj ruci. Primjer sekvence robotskog kretanja dan je na slici 3-19, gdje je vidljivo da robot uspješno obavlja ovaj zadatak.



Slika 3-19. Primjer izvršavanja trajektorije za odguravanje predmeta na UR5 robotu.

Optimizacijski procesi za generiranje grube trajektorije, za sva tri predstavljena problema u ovom poglavlju dani su na slici 3-20. Prikazana je evolucija funkcije cilja kroz iterativni proces. Testirana su dva različita inicijalna koraka pretrage σ, jedan mali i jedan daleko veći. Vidljivo je da je proces optimizacije robustan i konvergira za sva tri slučaja nakon oko 6000 iteracija. Pri tome se robusnijim pokazao veći iznos inicijalnog koraka pretrage ($\sigma = 100$), budući da u tom slučaju pretraga ne zapinje u lokalnom minimumu.

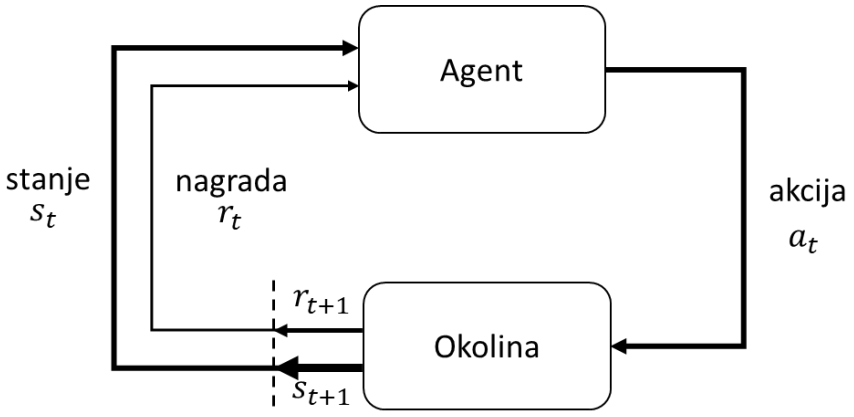


Slika 3-20. Funkcije nagrade različite scenarije. a) trajektorija sa tri prolazna k.s., b) trajektorija sa dva prolazna k.s. i jednom preprekom, c) trajektorija sa četiri prolazna k.s. i četiri prepreke (zadatak odguravanja).

4. Podržano učenje u kontinuiranim okruženjima

U nadziranim načinima učenja kao što je učenje iz demonstracija svi podaci za učenje moraju biti unaprijed spremni i dostupni prije učenja. Ovo je vrlo ograničavajuće kada robot treba naučiti složene zadatke, jer učitelj treba generirati dovoljan broj demonstracija. Podržano učenje (eng. *Reinforcement learning* - RL) se s druge strane temelji na samostalnom skupljanju podataka agenta (robota) putem vlastitog istraživanja i evaluacijama ponašanja od strane nekog ekspertnog sustava koji može biti čovjek ili u obliku nekog umjetnog sustava za ocjenjivanje. Najbolje ponašanje za određeni zadatak dano je u obliku optimalne strategije. Njegova značajna prednost ispred učenja iz demonstracija je ta što robot sva ponašanja izvodi samostalno i direktno svojim upravljačkim sustavom, tako da se kod učenja eliminira problem korespondencije. Isto tako, ljudsko ponašanje ne mora uvijek biti optimalno, pa učenje iz demonstracija može rezultirati sub-optimalnim rješenjima. Kod podržanog učenja ovaj problem ne postoji, nego je kvaliteta finalnog ponašanja najčešće određena samo rezultatom.

Klasični scenarij podržanog učenja opisan je agentom koji se nalazi u nekom trenutnom stanju s i vremenskom trenutku t, te djeluje u okolini svojom akcijom a. Agent svojim djelovanjem završava u novom stanju s_{t+1}, a njegovo se ponašanje ocjenjuje nagradom r_t.



Slika 4-1. Proces podržanog učenja.

Većina RL problema modelira se kao Markovljevi procesi odlučivanja (eng. *Markov decision process* – MDP). Ovo je matematički koncept koji se koristi za modeliranje strategija ponašanja gdje je svijet predstavljen u obliku stanja i akcija agenta. Ovaj koncept se koristi

pretpostavkom da svaka buduća akcija ovisi samo o stanju u kojem se agent trenutno nalazi, a ne o stanjima prije toga. MDP model čini pet glavnih elemenata

$$M = [S, A, R, T] \tag{4.1}$$

- S je stanje nekog okruženja.
- A je set akcija koje su agentu na raspolaganju.
- $R(s, a)$ je funkcija koja daje nagradu za poduzimanje akcije a u stanju s.
- $T\ (s'|s, a)$ je vjerojatnosna tranzicijska funkcija, koja definira vjerojanost da će agent preći u stanje s' ukoliko poduzme akciju a u stanju s.

Strategija je agentova poduzeta akcija a u trenutnom stanju s. Ovo je stoga veza između stanja i akcija. Najjednostavniji oblik strategija su deterministčke strategije. Njihovo ponašanje potpuno predvidivo i za svako stanje se potpuno sigurno može predvidjeti sljedeće stanje – ovo je obično nagrada koja donosi najveću nagradu.

$$\pi(s) = a \tag{4.2}$$

Drugi oblik strategija su stohastičke, koje umjesto jednoznačnih akcija za svako stanje daju vjerojatnosnu raspodjelu akcija. Drugim riječima, za određeno stanje, akcija nije deterministički određena, nego je dana vjerojatnost odabira određene akcije.

$$\pi(a|s) = P(a|s) \tag{4.3}$$

Sa stajališta optimalnog upravljanja, strategija je jednaka kontroleru, jer za svako ulazno stanje u kojem se nalazi sustav, daje izlaz u obliku akcije (upravljačke varijable).

Cilj agenta kod podržanog učenja je u svakom stanju u kojem se agent nađe, primijeniti akciju koja vodi do najveće kumulativne nagrade. Drugim riječima, maksimizirati očekivanu nagradu za svako stanje u kojem se agent trenutno nalazi. Ovakav model na koncu vodi do optimalnog ponašanja tj. strategije π^*.

Modeli optimalnog ponašanja kod podržanog učenja definiraju na koji način će se evaluirati agentovo ponašanje u okolini te posljedično i kako će agent učiti svijet oko sebe. Stoga je jako bitno odabrati odgovarajuću model optimalnog ponašanja za zadatak koji je potrebno naučiti. Postoje tri standardna modela kojima se opisuju ciljevi zadataka.

Model konačnog horizonta (eng. *finite horizon model*) – ovo je najjednostavniji model nagrađivanja. On u osnovi podrazumijeva da agent ima cilj u svakom trenutku (stanju) odabrati akciju sa najvećom nagradom, što kumulativno rezultira najvećom nagradom (4.4.).

$$J = \mathbb{E}\left\{\sum_{h=0}^{H} R_h\right\} \tag{4.4}$$

Model snižavanja nagrada (eng. *discounted reward model (infinite horizon)*) – kod ovog modela se nagrade očekivane u budućim koracima množe sa faktorom snižavanja γ, dajući im manju vrijednost što je očekivana nagrada dalje u nizu akcija (4.5.).

$$J = \mathbb{E}\left\{\sum_{h=0}^{\infty} \gamma^h R_h\right\} \tag{4.5}$$

U kontinuiranim zadacima, agent izvodi akcije u beskonačnom vremenu tj. na izvršavanje ovakvih zadataka ne postoji nikakvo vremensko ograničenje. Ovo znači da bi korištenje modela optimalnog ponašanja koji zbraja maksimalne buduće nagrade (konačni horizont) bilo nepovoljno iz razloga što je u dalekoj budućnosti nemoguće predvidjeti sve nagrade i ponašanja. Bilo zbog stohastičkih strategija ponašanja i/ili stohastičkih tranzicijskih modela svijeta u kojem agent djeluje. Zbog ovoga se u ovakvim problemima koristi model optimalnog ponašanja sa snižavanjem nagrada. Na ovaj način se smanjuje utjecaj budućih nesigurnih nagrada na model učenja.

Model prosječnih nagrada (eng. *average reward model*) – **k**od ovog modela se nagrade računaju tako da se uzima prosjek očekivanih nagrada po vremenskom trenutku (4.6.).

$$J = \lim_{H\to\infty} \mathbb{E}\left\{\frac{1}{H}\sum_{h=0}^{H} R_h\right\} \tag{4.6}$$

Ovakav model optimalnog ponašanja se često može vidjeti u primjerni kod epizodnih zadataka. Epizodni zadaci su oni kod kojih agent iz nekog početnog provodi zadatak sve do neke ciljne točke, koja je definirana vremenom (H). Budući da su ovi zadaci ograničeni u trajanju, ograničen je i broj akcija i stanja koja agent može posjetiti pa je opravdano koristiti sve moguće nagrade do samog kraja djelovanja (epizode).

Jedan od pristupa rješavanju generalnog problema podržanog učenja su metode vrijednosnih funkcija (eng. *value functions*). koje se modeliraju na osnovu agentovog samostalnog istraživanja i mehanizma ocjene ponašanja tj. optimalnog ponašanja. U osnovi postoje dvije vrste vrijednosnih pristupa: metode koje se baziraju na vrijednosti stanja i one koje se baziraju na vrijednosti parova stanja-akcija. Kod vrijednosnih funkcija stanja, svakom stanju cilj je odrediti njegovu vrijednost $V(s)$, dok je kod vrijednosnih parova to analogno za parove stanja-akcija $Q(s, a)$. Vrijednosti $V(s)$

i $Q(s,a)$ se ovdje odnose kumulativne vrijednosti nagrada određene primijenjenim modelom optimalnog ponašanja J. *Bellmanova* jednadžba omogućuje rastavljanje vrijednosne funkcije u zbroj trenutne nagrade i očekivanih budućih nagrada. Uz dane tranzicijske vjerojatnosti na osnovu ove jednadžbe se vrijednosti strategije π za epizodne scenarije mogu izraziti na sljedeće načine:

$$V^{\pi}(s) = \sum_a \pi(a|s)\left(R(s,a) - \bar{R} + \sum_{s'} V^{\pi}(s')T(s'|s,a)\right) \text{ i} \tag{4.7}$$

$$Q^{\pi}(s,a) = R(s,a) - \bar{R} + \sum_{s'} V^{\pi}(s')T(s'|s,a). \tag{4.8}$$

Poznavajući optimalnu vrijednosnu funkciju, optimalne strategije ponašanja dane su kao:

$$\pi^*(s) = arg\max_{\pi} V^{\pi}(s) \text{ i} \tag{4.9}$$

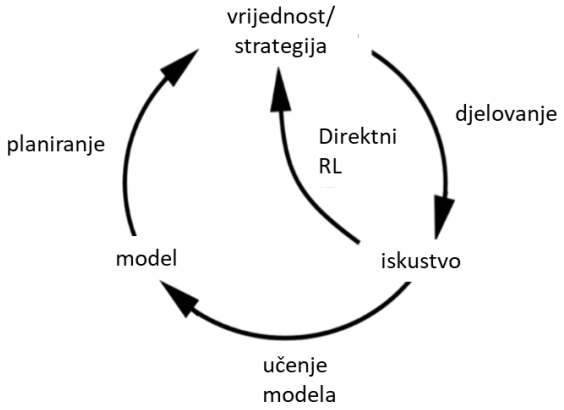
$$\pi^*(s) = arg\max_{a}\left(Q^*(s,a)\right). \tag{4.10}$$

Kombiniranjem istraživanja sa ažuriranjem vrijednosnih funkcija putem različitih metoda kao što su dinamičko programiranje, iteriranje vrijednosti (eng. *value iteration*), učenje vremenskih razlika i dr. ostvaruje se proces učenja vrijednosnih funkcija. Ovakva naučena mreža funkcija tvori mapu između stanja i vrijednosti (eng. *state-value map*). Za diskretne prostore stanja i akcija koji imaju konačni broj stanja i akcija, vrijednosne funkcije mogu biti predočene kao pregledne tablice (eng. *look-up table*), koje omogućuju da se očekivane vrijednosti nagrade vrlo lako izračunaju. U kontinuiranim prostorima, stanje ili akcije mogu imati više parametara i svaki od njih teoretski mogu zauzeti bilo koju vrijednost. Ovo zahtjeva da se funkcije dobrote predoče u obliku aproksimiranih funkcija na temelju kojih se očekivane nagrade izračunavaju najčešće pomoću dodatnog optimizacijskog procesa.

Jedan od osnovnih mehanizama podržanog učenja je samostalno pretraživanje prostora stanja i akcija s ciljem stvaranja slike o okolini i zadatku. Za ovakvo ponašanje nije dovoljno da agent neprestano slijedi optimalnu strategiju, nego je potrebno da istražuje prostor u kojem se nalazi. Ovaj istraživački mehanizam se agentu ugrađuje putem dodavanja signala perturbacija najčešće na vektor akcija. Generalno postoje metode kod kojih se strategija koja se poboljšava direktno primjenjuje u okolini i ocjenjuje (eng. *on-policy*) i metode kod kojih se strategija korištena za istraživanje može razlikovati od strategije koja se poboljšava (eng. *off-policy*).

Sa stajališta modeliranja, u podržanom učenju moguće je koristi modele za opisivanje dinamike okoline u kojoj se agent nalazi, tj. tranzicijske funkcije $T(s'|s,a)$ i modele za estimaciju nagrada R. Ovime se najčešće na probabilistički način može estimirati u kojem stanju će agent završiti ako u trenutnom stanju poduzme neku akciju i koju će akciju agent dobiti. Modeli se

tijekom interakcije sa okolinom mogu učiti, a za cilj imaju svesti potrebu za interakciju sa pravom okolinom agenta na minimum. Ovakvi pristupi se u literaturi nazivaju pristupi podržanog učenja bazirani na modelima (eng. *model-based RL*) i oni uključuju korake učenje modela i planiranje na osnovu modela (slika 4-2.). S druge strane postoje direktni pristupi podržanom učenju (slika 4-2.) koji se umjesto na model okoline potpuno oslanjaju na interakciju agenta sa okolinom (eng. *model-free RL*). Prednost ovog pristupa vidi se u istraživanju kompleksnih prostora stanja, čiju je dinamiku vrlo teško modelirati. Ovaj pristup ima značajnu primjenu u računalnim okruženjima dok kod realnih fizičkih sustava s kakvima se radi u robotici problem predstavljaju sigurnosni uvjeti koje treba osigurati kod ovakvog načina istraživanja.



Slika 4-2. Grafički prikaz toka podržanog učenja s obzirom na upotrebu modela.

U dosadašnjim istraživanjima na području robotike pokazana je upotreba podržanog učenja za učenje zadataka na razini odabira pred-definiranih akcija u različitim situacijama [44], [45],[46], [96], [43]. Isto tako, podržano učenje moguće je koristiti i za učenje na niskoj razini kao što je to učenje trajektorija ili kontrolera za izvođenje nekog zadatka [97], [98], [86], [99], [100], [88], [89], čime se bavi i ovaj rad.

4.1. Pretraživanje strategija

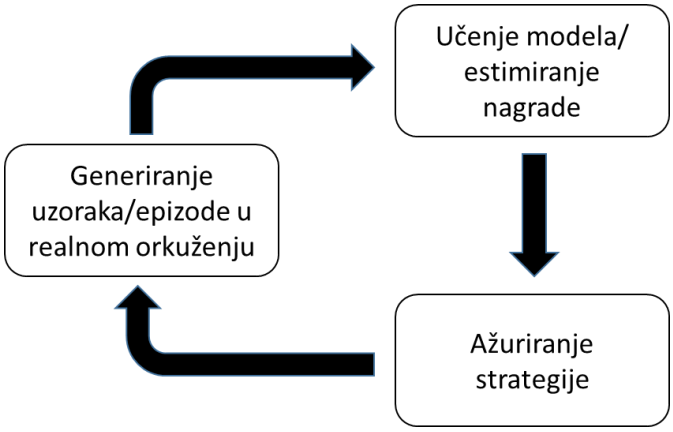
Kod vrijednosnih metoda u podržanom učenju cilj je stvoriti model vrijednosti stanja ili vrijednosti akcija na osnovu istraživanja prostora, te na temelju tog modela odrediti optimalnu strategiju ponašanja. Uz pretpostavku deterministčkog okruženja, ovaj princip teoretski je dokazan i pruža konvergenciju ka rješenju zadatka nakon dovoljno opsežnog istraživanja. U realnosti, roboti ali i drugi autonomni uređaji djeluju u kontinuiranom okruženju, koje nije moguće podijeliti u manji broj determiniranih konačnih stanja. S druge strane, porastom broja stanja, ovakve metode učenja

postaju neodržive zbog eksponencijalnog porasta dimenzionalnosti okruženja, gdje postaje nemoguće efikasno provesti istraživanje svih stanja i akcija i provesti njihovo mapiranje. Ovaj problem naziva se još i „prokletstvo dimenzionalnosti“ (eng. *curse of dimensionality*) [101].

Podržano učenje se može promatrati kao optimizacijski problem, gdje je cilj pronaći ponašanje robota s obzirom na funkciju nagrade, s time da se funkcija nagrade može definirati tako da u sebi sadrži neki od modela optimalnog ponašanja, koji su prethodno opisani. Kao i drugim područjima, optimizacija se može vršiti:

- algoritmi nultog reda, bez upotrebe gradijenta,
- algoritmi prvog reda, sa upotrebom gradijenta,
- algoritmi drugog reda, sa upotrebom „hessiana“.

Na osnovu koncepta optimizacije i prethodno spomenutog direktnog pristupa podržanom učenju (slika 4-2.) proizašao je koncept pretraživanja strategija koje imaju direktniji pristup učenju od klasičnog vrijednosnog podržanog učenja. Strategije su umjesto ovisnosti diskretiziranih akcija o stanjima ovdje definirane kao vektori parametara θ. Ovakav koncept dozvoljava parametrizaciju strategija ponašanja robota različitim modelima. Nadalje, kako u kompleksnim zadacima model okoline nije moguće točno opisati tranzicijskim funkcijama, pretraživanje strategija se oslanja na direktnu interakciju agenta (robota) sa okolinom. Ovo najčešće podrazumijeva interakciju u stvarnom fizičkom ili simulacijskom okruženju. Optimalna strategija je i dalje ona koja maksimizira očekivanu kumulativnu nagradu, što znači da osnove koncepta podržanog učenja ostaju nepromijenjene. Drugim riječima, pretraživanje strategije temelji se na ocjeni ponašanja te direktnom ažuriranju parametara strategije θ kako bi se maksimizirala očekivana nagrada. Iterativnim ponavljanjem ovog postupka ostvaruje se proces učenja.



Slika 4-3. Koncept pretraživanja strategija.

U sklopu pretraživanja strategija u kontinuiranim prostorima koje se koristi u robotici koristiti će se drugačiji tip notacija gdje

- stanje $S \rightarrow x$,
- akcija $A \rightarrow u$,
- nagrada $R \rightarrow r$,
- tranzicijska funkcija $T\ (s'|s,a) \rightarrow p(x_{t+1}|x_t,u_t)$.

Pristup pretraživanja strategija odmaknuo je koncept podržanog učenja od pretrage prostora stanja i akcija prema pretrazi parametara strategije na osnovu uzorkovanih nagrada. Konačni cilj ovog pristupa je pronaći parametre strategije θ koji daju najveću vrijednost očekivane nagrade J kroz interakciju s okolinom. Očekivana nagrada može se u ovom slučaju zapisati kao estimirana vrijednost ovisna o evaluiranom djelovanju robota u interakciji s okolinom $R(\tau)$ koje je dobiveno temeljem odabira parametara θ:

$$J(\theta) = \mathbb{E}[R(\tau)|\theta]. \tag{4.11}$$

U realnim okruženjima može se pretpostaviti probabilistički odnos djelovanja robota tj. dobivene trajektorije τ sa parametrima strategije $P(\tau|\theta)$. Za slučaj podržanog učenja, dinamika robota se može definirati u vremenski zavisnom obliku, gdje sljedeće stanje uvijek ovisi o trenutnom stanju x_t i upravljačkoj varijabli u_t.

$$p(x_{t+1}|x_t,u_t). \tag{4.12}$$

Za slučaj deterministčkih strategija, $P(\tau|\theta)$ se definira isključivo kroz vjerojatnosni utjecaj dinamike okoline tj. okoline i robota:

$$P(\tau|\theta) = p(x_0)\prod_t p(x_{t+1}|x_t,\pi(x_t,t)). \tag{4.13}$$

Za stohastičke strategije ovdje je uključena i vjerojatnost nastanka akcija (upravljačkih varijabli) u danom stanju:

$$P(\tau|\theta) = p(x_0)\prod_t p(x_{t+1}|x_t,u_t)\pi_\theta(u_t|x_t,t). \tag{4.14}$$

Uzimajući ovo u obzir ove probabilističke odnose, očekivana nagrada $J(\theta)$ dana je kao suma svih mogućih trajektorija s vjerojatnošću nastanka trajektorije za određeni θ - $P(\tau|\theta)$ otežanih s nagradom te trajektorije $R(\tau)$ (4.13).

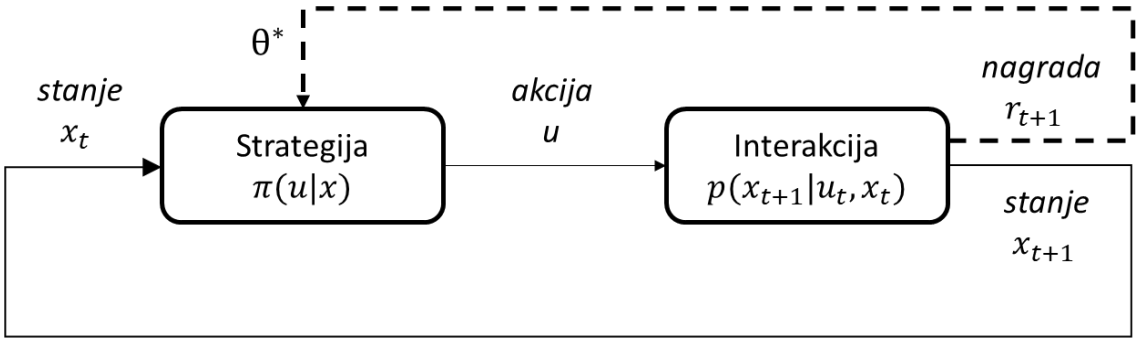
$$J(\theta) = \int R(\tau)P\ (\tau|\theta) \tag{4.15}$$

Budući da je tema ovog rada učenje robota na razini gibanja (trajektorije), može se pretpostaviti da robot djeluje na okolinu epizodno tj. od nekog početnog stanja $t = 0$ do konačnog

stanja $t = T$. Po uzoru na prethodno opisane optimalne modele ponašanja, nagradu je u tom slučaju moguće definirati kao:

$$R(\tau) = \sum_{i=1}^{T} r_{ti}. \tag{4.16}$$

Za ovakav epizodni scenarij, robot izvodi ponašanje (trajektoriju) na osnovu strategije definirane sa unaprijed određenim inicijalnim parametrima θ i promatra dobivenu nagradu. U slijedećoj epizodi agent ažurira parametre na taj način da pokuša dobiti veću očekivanu nagradu.



Slika 4-4. Pretraživanje strategija u kontekstu podržanog učenja

Kako bi se osiguralo istraživačko ponašanje sustava prilikom uzorkovanja trajektorija, perturbacije ϵ_t mogu se dodavati na dva načina. Moguće je dodati direktno na odabranu akciju od strane strategije:

$$u_t = \pi_\theta(x) + \epsilon_t. \tag{4.17}$$

Budući da prostor stanja i akcija na stohastički način ovisi o parametrima strategije $P(\tau|\theta)$, perturbacije se mogu dodati i u parametarskom prostoru:

$$u_t = \pi_{[\theta+\epsilon_t]}(x). \tag{4.18}$$

Problem pretraživanja parametara strategije može se formulirati kao traženje novih parametara strategije θ_{i+1}, kako bi se dobile bolje performanse strategije u sljedećoj iteraciji učenja, tj. terminologijom podržanog učenja – kako bi se maksimizirala očekivana nagrada $J(\theta)$:

$$\theta_{i+1} = \theta_i + \Delta\theta_i. \tag{4.19}$$

Dva su glavna pristupa koja su se razvila za rješavanje problema pretraživanja strategija u kontinuiranim domenama. Metode koje se oslanjaju na RL strukturu i one koje ju ne koriste. Važan koncept podržanog učenja je taj da se u trenutnom stanju x, poduzme akcija u i dalje nastavlja koristiti strategija koja je zadana trenutnim parametrima ili parametrima optimalne strategije koja donosi određenu konačnu nagradu $J(\theta)$. Za ovo je potrebno poznavati model vrijednosti stanja-

akcija $x - u$ koji se može aproksimirati modelom koji se često naziva model kritičara (eng. c*ritic*). Optimizacija parametara strategije θ u odnosu na model kritičara naziva se pristup akter-kritičara, gdje je akter model strategije π_θ. Metode korištene za optimizaciju parametara aktera su iz porodice gradijentnih metoda i EM metoda.

Drugi način ažuriranja parametara aktera je direktnom pretragom parametara aktera u odnosu na vrijednosti nagrada, pri čemu se ne stvara model vrijednosti stanja-akcija nego se direktno pretražuje prostor nagrada $J(\theta)$.

4.2. Ažuriranje parametara kod pretraživanja strategija (metode kritičara)

Ako se problem pretraživanja strategija gleda u kontekstu učenja podrškom, prostor vrijednosti stanja i akcija može se modelirati $U(x, u)$. Spremanjem svih viđenih (x, u) parova i viđenih nagrada $R(\tau)$, moguće je regresijskim metodama aproksimirati ovaj model, koji se naziva i model kritičara. Za ovo je potreban veći broj viđenih stanja koja su dobivena uzorkovanjem na pravom sustavu. Ovo je tipičan scenarij na kojem se mogu primijeniti pristupi temeljeni na Monte-Carlo načinu optimizacije temeljem uzorkovanja. Dva su osnovna pristupa korištena za ažuriranje parametara strategija. Prvi su gradijentni algoritmi, a drugi su algoritmi zasnovani na EM metodama koji koriste težinsko uprosječavanje nagrade (eng. *reward weighted averaging*). Ovdje će se dati uvid u osnovne karakteristike obaju pristupa kako bi se dobila bolja slika o osnovama pretraživanja strategija.

4.2.1. Gradijentne metode

Gradijentne metode su generalni pristupi za pronalaženje minimuma/maksimuma neke funkcije s obzirom na neke ulazne podatke. Gradijentno spuštanje za minimiziranje neke funkcije, a gradijentni uspon za maksimiziranje. Ove dvije metode razlikuju se samo u predznaku. One rade na način da u prostoru ulaznih parametara pretražuju vrijednosti na način da vrijednosti idu u smjeru smanjenja/porasta funkcije. Korak promjene parametara pretrage ovisan je o nagibu (derivaciji/gradijentu) funkcije s trenutnim parametrima. Ukoliko je gradijent velik, korak promjene biti će veći, kako bi algoritam što brže konvergirao rješenju. Približavanjem optimumu, gradijent se smanjuje, a proporcionalno i korak pretrage, kako bi se omogućilo točniji pronalazak

rješenja. Inicijalni korak pretrage obično je otvoreni parametar i zadaje se od strane korisnika, a smatra se i stopom učenja (eng. *learning rate*).

Kod primjene ovih metoda na pretraživanje strategija, set ulaznih parametara je sačinjen od parametara strategije θ, a vrijednosti funkcije dane su preko očekivane nagrade $J(\theta)$, koja se dobiva putem funkcije nagrade ponašanja, nakon svake epizode ponašanja. Ažuriranje parametara putem gradijentnih metoda dano je izrazom:

$$\theta_{i+1} = \theta_i + \alpha \nabla_\theta J, \tag{4.20}$$

gdje je α stopa učenja. Gradijent parametara strategije dan je kao:

$$\nabla_\theta J_\theta = \int \nabla_\theta \, p_\theta(\tau) R(\tau) d\tau. \tag{4.21}$$

Problem estimacije gradijenta funkcije J_θ nastaje uslijed toga što kod metoda pretraživanja strategija ne postoji model koji potpuno opisuje ponašanje agenta u okolini, nego se njegovo ponašanje očituje u direktnoj interakciji s okolinom $p_\theta(\tau)$. Zbog toga se uz poznate faktore na koje se može utjecati direktno kao što su parametri strategije, javljaju i dodatni utjecaji, kao što su dinamika okoline. Utjecaj ovih faktora nisu poznati, pa se ovo u svrhu estimacije gradijenta funkcije očekivane nagrade mora kompenzirati na osnovu uzorkovanja ponašanja.

Postoje različite metode za estimaciju gradijenta $\nabla_\theta J$. Neke od njih su: metoda konačnih razlika (eng. *finite difference methods*), omjer vjerojatnosti (eng. *likelihood ratio*) i prirodni gradijenti (eng. *natural gradients*) [102]. Jedan od najpoznatijih gradijentnih algoritama je REINFORCE (*REward Increment = Nonnegative Factor times Offset Reinforcement times Characteristic Eligibility*) [103], [104]. On koristi princip omjera vjerojatnosti za ažuriranje parametara. Pri tome od nagrade koja služi za otežavanje oduzima baznu vrijednost nagrada b. Na ovaj način se postiže robusniji izračun samog gradijenta (4.22).

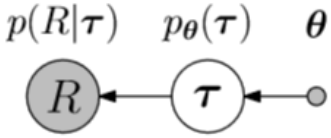
$$\nabla_\theta J_\theta = \mathbb{E}_{p_\theta(\tau)} \left[\sum_{t=0}^{T-1} \nabla_\theta \, log \, \pi_\theta(u_t | x_t, t) \, (R(\tau) - b) \right] \tag{4.22}$$

Upotreba prirodnih gradijenata u robotici prikazana je prvi put u [105]. Prirodni gradijenti su na epizodne scenarije primijenjeni pomoću eNAC algoritma. eNAC je korišten za učenje upravljanja antropomorfnom robotskom rukom za zadatak udaranja loptice palicom u [106], [107]. Modificirana verzija eNAC-a korištena je za učenje hodanja za dvo-segmentnu robotsku ruku [108]. Učenje robotskog hodanja prikazano je i u [109].

4.2.2. Metoda otežane maksimalne vjerojatnosti

Pretraživanje strategije može se modelirati kao problem zaključivanja sa skrivenim varijablama i korištenjem EM algoritama (slika 4-5.). EM algoritmi ažuriraju parametre na temelju otežanih maksimalnih vjerojatnosti (eng. *maximum likelihood estimate* – MLE). Generalno, cilj MLE metode je maksimizirati vjerojatnost da je izmjerena ili viđena neka vrijednost podatka. Ovo se može koristiti kao mjera vjerojatnosti da je trenutni model proizveo neku vrijednost. Ukoliko trenutni model ima malu vjerojatnost da je proizveo viđenu vrijednost, tada se model znatno podešava u određenom smjeru. Ukoliko je vjerojatnost modela da je proizveo viđenu vrijednost velika, tada se model manje podešava tj. model je već prilično dobar. Ovo je analogno vrijednosti gradijenta kod gradijentnog pretraživanja. Smjer podešavanja (pozitivni/negativni) određen je time s koje strane distribucije se nalazi viđena vrijednost.

Kod pretraživanja strategije, vidljiva varijabla su u ovom slučaju nagrade, a skrivena varijabla su trajektorije koje su uzrokovale određenu nagradu. Ulazne varijable su parametri strategije θ. Mogućnost modeliranja pretraživanja strategije kao MLE problem prvi put je predstavljena u [110].

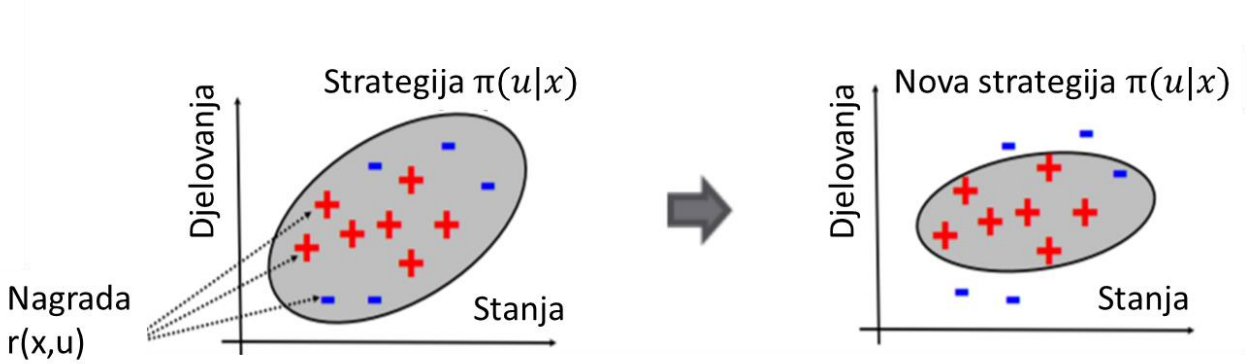


Slika 4-5. Pretraživanje strategija kao problem zaključivanja.

Za problem optimizacije parametara kod pretraživanja strategije koristi se otežana verzije MLE. Ova metoda se razlikuje u tome što je kod nje moguće veće težine dati trajektorijama koje uzrokuju veće nagrade, dok se zanemaruju parametri trajektorija koji daju manje nagrade. Ovo omogućuje da se statistički model usmjeri u modeliranje bitnih vrijednosti u svrhu optimizacije parametara. Ovaj pristup se koristi prefiks uprosječavanja otežanom nagradom (eng. *reward weighted averaging*). Dobro svojstvo istoga je što ovaj način ažuriranja prati prirodni gradijent, što je vrlo korisno u zadacima u robotici.

Iterativnim izvođenjem trajektorija i modeliranjem distribucija, algoritam mijenja svoje parametre prema optimalnim vrijednostima. Primjer ažuriranja parametara modela za prostor od dva parametra dan je na slici 4-6. Smjer ažuriranja određen je oblikom distribucije, te se uvijek kreće u smjeru izduženja elipse. Pri tome se nova distribucija dobiva minimiziranjem KL (*Kullback–Leibler*) divergencije (4.25).

$$D(p_\theta(\tau)||r(\tau)p(\tau)) \rightarrow \min \tag{4.25}$$



Slika 4-6. Ažuriranje parametara strategije metodom otežane maksimalne vjerojatnosti.

Rad EM algoritma se očituje u dva koraka, tzv. E-korak i M-korak.

E-korak – algoritam kreće sa nekom inicijalnom distribucijom i parametrima. Uzorkovanjem N trajektorija parametrima iz te distribucije, dobivaju se vrijednosti nagrada za svaku od trajektorija. U sljedećem koraku estimira se vjerojatnost da je trajektorija prouzročila određenu nagradu $p_\theta(\tau)$ i formira se nova distribucija pomoću vrijednosti koje su otežane nagradom $exp(\beta R(\tau))$ (slika 4-6.). Ova nova distribucija predstavlja skup onih trajektorija koje su se pokazale najboljima (4.26). Sustavom otežavanja, se trajektorije s manjom vrijednosti praktički zanemaruju.

$$p_\theta(\tau|R) = \frac{p_\theta(\tau)\, exp(\beta R(\tau))}{\int p_\theta(\tau)\, exp(\beta R(\tau)) d\tau} \tag{4.26}$$

M korak – estimiraju se novi parametri modela na taj način da se traži maksimalna vjerojatnost MLE metodom (4.27).

$$\theta = arg\, \max_\theta Q(\theta) = \int_\tau p_\theta(\tau|R)\, log\, p(R|\tau) P(\tau, \theta) d\tau \tag{4.27}$$

Postoje dva poznata algoritma za pretraživanje strategije koja koriste ovaj princip. Primjena algoritama koji koriste otežane maksimalne vjerojatnosti započela je kod RWR algoritama (eng. *Reward-weighted regression*) koji ažuriranje parametara ustvari prevodi u regresijski problem. U [111] je ovaj algoritam korišten za kontroler u operacijskom prostoru robota. Ipak, ovaj algoritam se pokazao vrlo osjetljivim i krhkim kod robotskog učenja, što je pripisano nestrukturiranoj strategiji pretraživanja dodavanjem šuma na vektor akcija [88]. Drugi vrlo uspješan algoritam ovdje je PoWER (*Policy learning by Weighting Exploration with the Returns*). On za razliku od RWR algoritma koristi, perturbaciju parametara. U [88] je prikazana upotreba PoWER algoritma na raznim robotskim zadacima kao što su udaranje loptice palicom, podaktuirano inverzno njihalo,

hvatanje loptice („ball-in-cup“ igra). U svim ovim primjerima algoritam je konvergirao naučenom zadatku u oko 100 iteracija. Također, prikazana je primjena ovog algoritma za zadatak bacanja i okretanja palačinke sa robotskom rukom [89].

4.2.3. Druge metode

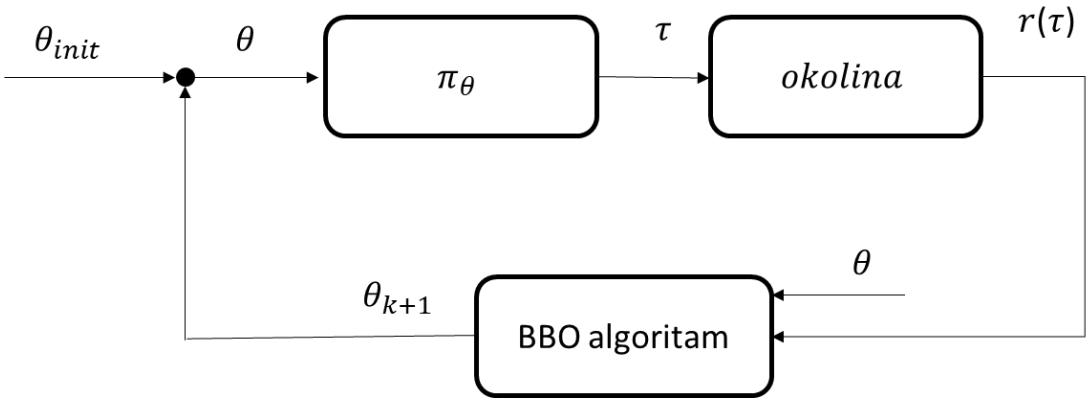
Neke od drugih vrlo bitnih metoda koje uključuju neku vrstu kritičara su PI^2 [100], DDPG [112] , GPS [113], TRPO [114], PPO [115] . U zadnje vrijeme vrlo bitan je razvoj metoda koje modeliraju i dinamiku sustava. Jedan takav primjer je PILCO algoritam [116], koji je pokazao vrlo veliku učinkovitost u učenju zadataka. On koristi GP modele za modeliranje dinamike sustava tj. predviđanje vjerojatnosti budućeg stanja $\hat{p}(x_{t+1})$. PILCO zahtjeva poznavanje funkcije nagrada i to da funkcija nagrada bude derivabilna. PILCO je prvi algoritam ovakav algoritam koji je pokazao veliku efikasnost korištenja podataka na zadacima kao što su inverzno njihalo i upravljanje robotskim manipulatorima [117]. Kasnije je testiran i na drugim dobrim primjerima primjene [118].

4.3. “Black-box” optimizacija (BBO)

Ako se problem nagrađivanja kod podržanog učenja svede na nagrađivanje samo konačnog rezultata nekog ponašanja, bez razmatranja posrednih stanja, a strategije se opišu parametriziranim modelima, pretraživanje strategija se može promatrati kao crna kutija (eng. *black-box*). Ovdje su ulazi parametri strategija, a izlaz skalarna nagrada. U ovom slučaju se pretraživanje parametara strategije može ostvariti standardnim „black-box“ optimizacijskim metodama (eng. *black-box optimization* – BBO). Ovakav način pretrage ne koristi standardnu strukturu RL problema, pa pojednostavljuje postavljanje okoline za učenje. Sustav nagrađivanja ne treba davati nagrade agentu u svakoj vremenskoj jedinici (svakom koraku) nego daje kumulativnu nagradu samo na kraju epizode. Dinamika okoline se ne modelira, nego se rezultati djelovanja parametara direktno primjenjuju za interakciju agenta (robota) s okolinom kroz epizode ponašanja, što predstavlja drugo pojednostavljenje.

Cilj BBO algoritma je ažurirati ulazne parametre strategije θ na način da optimizira vrijednost nagrade/funkcije cilja kroz iterativno poboljšavanje.

$$\theta^* = \text{argmax}\, J(\theta) \tag{4.28}$$



Slika 4-7. Model pretraživanja strategije sa BBO

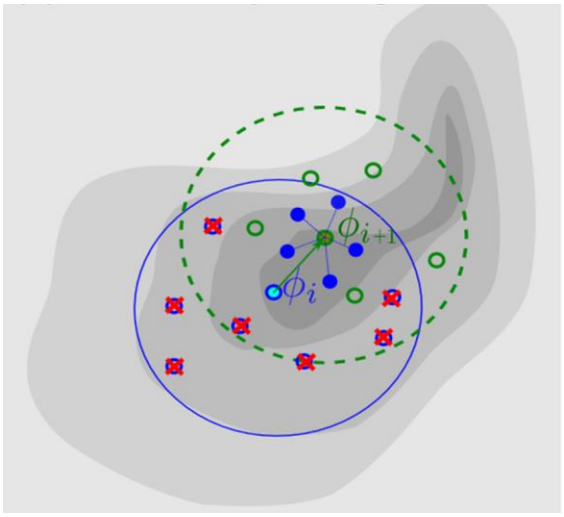
Pretraživanje strategija pomoću BBO metoda u pravilu se izvodi perturbacijama parametara, za razliku od nekih drugih prethodno spomenutih pristupa koji koriste perturbaciju na nivou akcija (REINFORCE, RWR). Ovo se u svim novijim pristupima pokazalo kao robusniji način istraživanja. Svojevrsni nedostatak BBO pretraživanja strategija je to što se izmjena parametara može ostvariti samo prije početka epizode, ne i tijekom izvršavanja trajektorije u epizodi. Usporedba BBO načina pretraživanja parametara sa klasičnim RL metodama dana je u [119].

4.3.1. CMA-ES algoritam

Jedan od primjera BBO algoritama su evolucijske strategije (eng. *evolution strategies* – ES). One koriste operacije mutacije, rekombinacije i selekcije u iterativnom procesu kako bi se pronašle populacije rješenja koje optimiziraju postavljenu funkciju cilja [120]. Neke od primjena ovih algoritama za učenje u robotici su problem inverznog njihala, robusno stajanje robota i hvatanje loptice [121].

CMA-ES (*Covariance Matrix Adaptation Evolution Strategy*) primjer je BBO evolucijskog algoritma koji je pogodan za optimizaciju nelinearnih stohastičkih funkcija sa velikim prostorom ulaznih parametara [94], [122]. Njegova primjena za optimizaciju RL problema prvi put je predstavljena u [123]. CMA-ES koristi matricu kovarijanci za opis distribucije parametara za određene vrijednosti. Kroz evaluaciju rješenja, svaki set parametara matrice kovarijanci ažurira se kako bi se postiglo bolje rješenje zadane funkcije cilja. Algoritam uzorkuje vrijednosti iz parametarskog prostora stanja i na njih dodaje perturbacijski vektor čije su vrijednosti dobivene na

temelju normalne distribucije. Nakon toga se rekombinacijom ažuriraju roditelji rješenja na taj način da se nova distribucija parametara formira na temelju najboljih viđenih vrijednosti parametara. CMA-ES za ovo ažuriranje koristi rješenja koja su otežana nagradom (eng. *reward-weighted averaging*), što je pristup analogan viđenom kod metoda otežane maksimalne vjerojatnosti, a koji se pokazao učinkovitijim i robusnijim od gradijentnih pristupa.



Slika 4-8. Ažuriranje parametara CMA-ES algoritmom. ϕ_i je stara distribucija parametara, dok je ϕ_{i+1} nova distribucija parametara, dobivena iz parametara koji su uzrokovali visoke nagrade [124].

CMA-ES ima jedan otvoreni parametar, a to je tzv. početni korak učenja (eng. *initial step size*). To je u osnovi standardna devijacija distribucije pretrage vrijednosti za svaki parametar. Ovisno o iznosu ove standardne devijacije distribucije se prostor pretrage povećava (veća vrijednost koraka učenja – globalna pretraga) i smanjuje (manja vrijednost koraka učenja – lokalna pretraga). Početni korak učenja stoga ima veliki utjecaj na veličinu prostora rješenja koji se istražuje.

Ovaj algoritam korišten je u nizu problema pretraživanja strategija u robotici. Usporedba ovog pristupa sa gradijentnim pretraživanjem strategije dan je u [125]. U [126], [127] autori su prikazali sistematičnu usporedbu CMA-ES algoritma i gradijentnih pristupa za zadatak inverznog njihala. Isti autori proširili su istraživanje i na dvostruko inverzno njihalo u [128]. Zaključak istraživanja je da je CMA-ES algoritam bolji u usporedbi za gradijentnim metodama, prije svega u pogledu robusnosti za zadane početne parametre. U [129] prikazano je da CMA-ES slijedi prirodni

gradijent kod pretrage. Usporedba CMA-ES algoritma sa DDPG algoritmom dana je u [130]. Preliminarni rezultati pokazali su da je CMA-ES i ovdje znatno robusniji u pogledu izbora početnih parametara strategije i otvorenih parametara za učenje (korak učenja).

U [131] i [132] CMA-ES je korišten u kombinaciji s evolucijskim topologijama kako bi razvili neuronsku mrežu. Korišten je u [133] za učenje zadatka udaranja lopte humanoidnim robotom. Iako je pokazao robusnost, za primjenu u robotici predlažu se nove varijante ovog algoritma. Jedan od primjera je CMA-ES sa regijom povjerenja (eng. *trust region*) koji je predstavljen i validiran na zadatku robotskog učenja stolnog tenisa u [134].

4.4. Parametarski modeli strategija

Strategije su u smislu podržanog učenja kod robota same izvršne varijable ili iz njih mogu proizaći izvršne upravljačke varijable $\pi(u|x,t,\theta)$. Ako strategije ne generiraju direktne upravljačke varijable za robota **u**, tada je to potrebno učiniti posredno. U slučaju strategije koja generira kutove zakreta robotske ruke, ovo se može učiniti korištenjem PID kontrolera ili LQT kontrolera. U pravilu današnje robotske ruke imaju integrirane različite vrste kontrolera (po poziciji, brzini, momentu) koje je moguće koristiti za ove zadatke.

U idealnom scenariju, model strategije je dovoljno općenit i može se primijeniti na različite zadatke učenja. Neki od primjera modela strategija su linearne strategije, radijalne bazne funkcije, neuronske mreže. U robotici se na nižoj razini učenja najčešće koriste modeli strategija u obliku trajektorija u operacijskom ili konfiguracijskom prostoru robota.

Linearne strategije – to su strategije koje se mogu koristiti kada postoji direktna veza između nekog parametra modela strategije i ponašanja robota (agenta).

Prolazne točke i spline – prirodan način definiranja strategije za robota su trajektorije. One se mogu definirati u obliku sekvence jednostavnih prolaznih točaka u kartezijskom ili konfiguracijskom prostoru robota. Nekada prilikom učenja nije potrebno koristiti sve točke trajektorije pa se prilikom učenja mogu koristiti samo manji broj prolaznih točaka, što znatno smanjuje prostor pretrage. Kod korištenja ovakvih strategija potrebno je imati dodatni kontroler koji robotu omogućuje pretvorbu strategije u izvršne naredbe.

U [135] je optimiran položaj prolaznih točaka kako bi se naučio zadatak hvatanja loptice u šalicu (eng. *ball in cup*). Prolazne točke se često kombiniraju sa *spline* interpolacijama koje

generiraju kontinuirane prijelaze između točaka. U [136] je ovaj pristup korišten za upravljanje humanoidnim robotom, a u [137] za parametrizaciju mahanja krilima.

DMP – već opisani DMP modeli predstavljaju često korištene strategije kretanja u robotici. Parametri učenja su težine nelinearne funkcije oblika trajektorije, a izvršni dio strategije su trajektorije u operacijskom ili konfiguracijskom prostoru robota. Jedna od prednosti DMP modela u odnosu na prolazne točke i ostale modele trajektorija je da prilikom učenja parametara težina, DMP osigurava kontinuiranost trajektorija (bez naglih promjena akceleracija).

Radijalne bazne funkcije – parametri učenja kod ovih strategija su srednje vrijednosti i standardne devijacije svih baznih funkcija. Ovaj model strategije predstavlja vrlo općenit način modeliranja koji je generalno primjenjiv na razne vrste problema.

Neuronske mreže – kao još jedan generalni model strategija tu su neuronske mreže. Parametri učenja su otvoreni parametri neuronske mreže, a povratna veza su rezultati ocjene ponašanja u okruženju podržanog učenja.

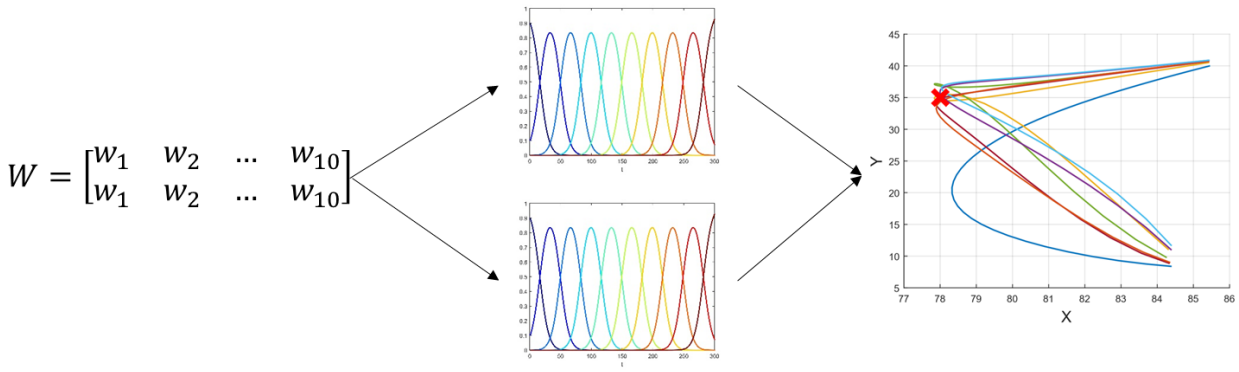
"End-to-end" strategije – ovo su strategije koje integriraju sve razine, od senzorskih ulaza za učenje do izvršnih naredbi za sustav, te između ove dvije razine nije potreban nikakav dodatni sloj za pretvorbu informacija. Primjer ovoga je pristup „from pixels to control", gdje se ponašanje agenta snima vizijskim sustavom i njegovo se ponašanje automatski ocjenjuje. Ovo je zatvorena petlja učenja gdje se u osnovi uči model mapiranja piksela sa vizijskog sustava na naredbe na izvršnoj razini robota. Neki od primjera ovakvih pristupa dani su u [138], [139], [140], [141].

4.5. Učenje u kontinuiranom prostoru s DMP strategijom

U ovom poglavlju biti će opisana upotreba BB optimizacijskog pristupa za proces učenja trajektorije parametrizirane DMP metodom koja je u drugim istraživanjima predstavljena kao parametrizacija koja je pogodna za iterativno učenje. Kao optimizacijski algoritam koristiti će se CMA-ES algoritam.

Zadatak se sastoji od toga da se inicijalna trajektorija pretvori u DMP zapis, te se iterativnom pretragom pokuša naći trajektorija koja prolazi kroz zadanu prostornu poziciju X, koja prethodno nije dio inicijalne trajektorije. Nelinearna funkcija DMP zapisa parametrizirana je sa 10 baznih aktivacijskih funkcija, $n_{bfs} = 10$, za svaki modelirani stupanj slobode. U ovom slučaju radi se o dvodimenzionalnoj trajektoriji, $n_{dmp} = 2$. Težina svake bazne funkcije w_i može služiti kao

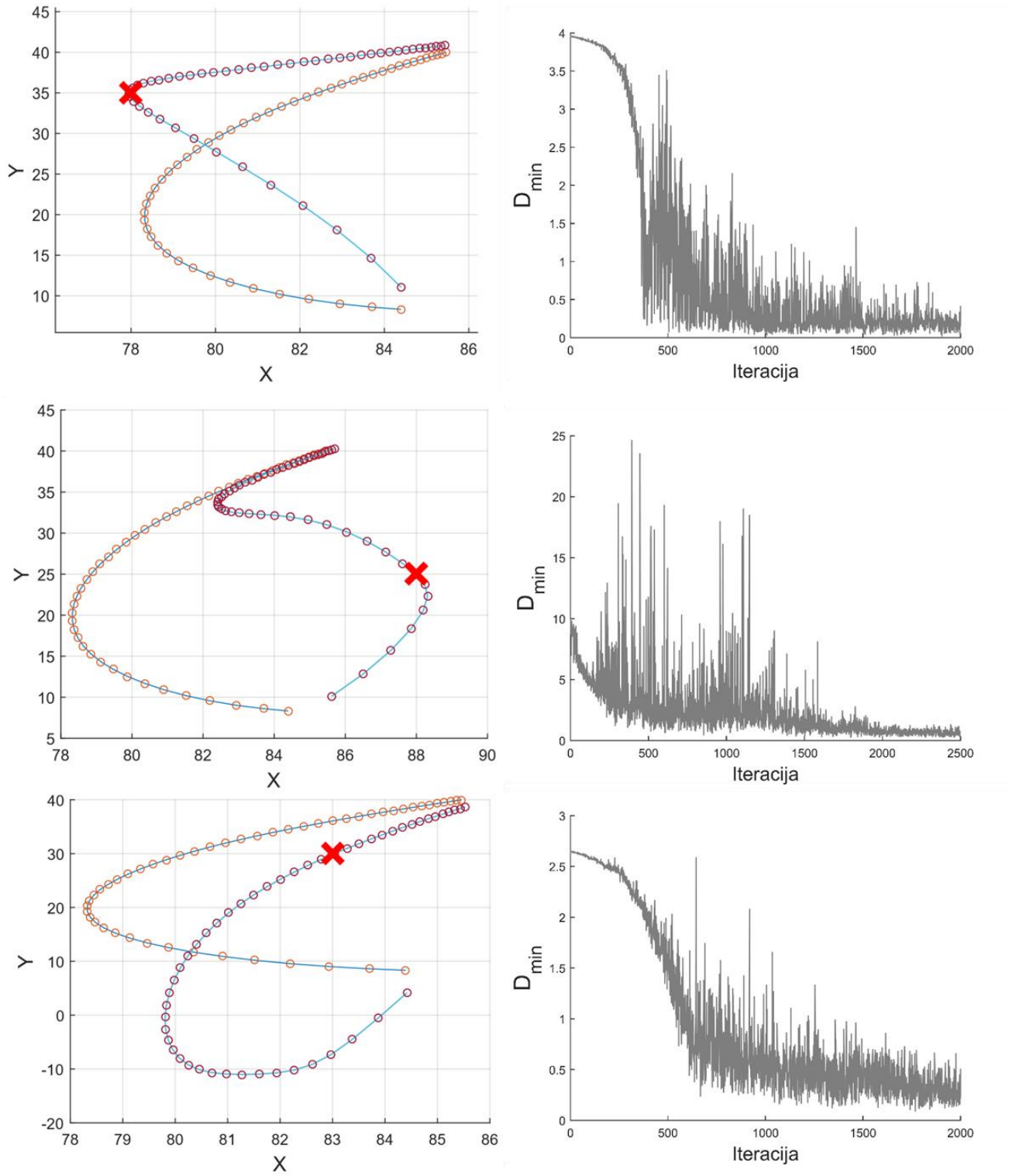
optimizacijski parametar. Parametarski prostor DMP trajektorije opisan je stoga u ovom slučaju sa 20 parametara koji predstavljaju ulaze u BB optimizacijski algoritam (slika 4-9.).



Slika 4-9. Koncept pretraživanja strategije i parametarski prostor problema.

Inicijalni parametri težina dobiveni su standardnom regresijskom metodom učenja DMP-a na osnovu postojeće trajektorije, temeljem metodologije opisane u 2.4.1. Potom su težine inicijalne trajektorije kroz proces pretrage CMA-ES algoritmom transformirane na taj način da trajektorija prolazi kroz zadanu prolaznu točku. Cijeli algoritam ovog postupka dan je u tablici 4-1.

Na slici 4-10. prikazane su tri konfiguracije zadatka i njihova rješenja. Prikazane su i vrijednosti funkcije cilja tijekom samog optimizacijskog procesa. Vidljivo je da se promjenom vrijednosti parametara težina DMP modela, zadržao kontinuitet trajektorije dok je oblik trajektorije ostao isti u dijelovima gdje je to moguće. Ovo je jako bitno svojstvo za primjenu u robotici gdje je kontinuitet u domeni brzina i akceleracija imperativ kod izvršavanja na realnim manipulatorima i sustavima općenito. Uz to, tijekom procesa učenja očuvane su početne i krajnje pozicije trajektorije, dok je samo oblik promijenjen. Ovo je isto tako zanimljivo svojstvo DMP modela, koji jamči stabilnost konvergencije cilju tijekom promjene parametara oblika (težina).



Slika 4-10. Inicijalne trajektorije i trajektorije nakon završenog procesa učenja koji minimizira udaljenost od prostorne prolazne točke X (lijevi stupac). Pripadajuće vrijednosti funkcije cilja (kroz proces učenja) desni stupac.

Tablica 4-1. Iterativno učenje DMP parametrizacijom.

	Ulaz: y_{demo} // demonstrirana trajektorija
1	**inicijalizacija** $n_{dmp} = 2, n_{bfs} = 10,\ d_t = 0.02, \tau = 1,\ \alpha_z,\ \beta_z,\ \alpha_x, c\ , \sigma$
2	**za svaki vremenski trenutak** $t = 1{:}\,\tau/d_t$
3	$x = x(t-1) - \alpha_x \cdot x_t \cdot d_t$
4	$\psi_i(x) = \exp\left(-\frac{1}{2\sigma_i^2}(x - c_i)^2\right)$
5	**kraj**
6	$y_0 = y_{0_{demo}}, g = g_{demo}$
7	**za svaki n_dmp** // učenje težina
8	$f_{target} = \ddot{y}_{demo} - \alpha_z(\beta_z(g - y_{demo}) - \dot{y}_{demo}$
9	$w_i = \frac{S^T \Gamma_i f_{target}}{S^T \Gamma_i S}, S = \begin{pmatrix} x(t_0)(g-y_0) \\ x(t_1)(g-y_0) \\ \vdots \\ x(t_N)(g-y_0) \end{pmatrix}, \Gamma_i = \begin{pmatrix} \psi_i(t_0) & 0 & \cdots & 0 \\ 0 & \psi_i(t_1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \psi_i(t_N) \end{pmatrix}$ //inicijalne težine
10	**kraj**
11	**za svaku iteraciju** w_i
12	**za svaki vremenski trenutak** $x_i \in x$
13	**za svaki** $i = 1{:}\,n_{dmp}$
14	$f(x) = \frac{\sum_{i=1}^{N} \psi_i(x) w_i}{\sum_{i=1}^{N} \psi_i(x)} x(g - y_0)$
15	$\ddot{y}_i = \alpha_z(\beta_z(g - y_i) - z + f$
16	$\dot{y}_i = \dot{y} + \ddot{y} \cdot d_t$
17	$y_i = y + \dot{y} \cdot d_t$
18	**kraj**
19	**kraj**
20	$D_{min} = \min(\boldsymbol{D_{xy}} - \boldsymbol{y})$ //funkcija cilja
21	**kraj**

4.6. Zaključci o poglavlju

U danom pregledu učenja istraživanjem pomoću podržanog učenja (RL) obrađeni su glavni pristupi samom problemu samostalnog učenja robota. Klasični RL pristupi koji koriste vrijednosne funkcije stanja i vrijednosne funkcije stanja-akcija prikladni su za deterministička okruženja sa unaprijed poznatim stanjima i akcijama. Kod simboličkog učenja u robotici ovo ostaje zanimljiv i realan pristup. Budući da su problemi u robotici na niskoj (izvršnoj) razini uglavnom vezani za kontinuirana stanja i akcije, smjer razvoja je ovdje otišao prema parametarskim zapisima strategija, gdje se kristalizirao manji broj parametarskih zapisa koji se mogu koristiti za različite modele ponašanja (poglavlje 4.4.). Tu se uglavnom radi o modelima koji opisuju kretanje. Prednosti parametarskog pristupa su izražene činjenicom da je manjim skupom parametara moguće u potpunosti opisati neko kompleksno ponašanje robota. Kvaliteta parametarske strategije određuje se na temelju univerzalnosti primjene (mogućnost primjene u različitim scenarijima) i prilagođenosti za učenje. Parametarske strategije omogućile su razvoj podržanog učenja u smjeru pretraživanja strategija, gdje su u početku prevladavale gradijentne metode, dok su se kasnije dominantnim pokazale metode koje se oslanjaju na uprosječavanje nagradom otežanih rješenja.

Pretraživanje strategija se umjesto na modeliranje vrijednosnih funkcija, oslanja na direktnu interakciju sa okolinom i sustav nagrađivanja postignuća tijekom te interakcije. Ovaj smjer direktne interakcije doveo je i do primjene BB (black-box) optimizacijskih metoda u učenju robota, gdje je sustav nagrađivanja dodatno pojednostavljen i promatra se samo krajnja akumulirana nagrada ili krajnji ishod zadatka. Radi cjelovitosti pregledanog područja, u ovom poglavlju su izdvojeni i pristupi koji idu u smjer modeliranja. Ovdje su u posljednje vrijeme prisutni pristupi pretraživanja strategija koji se baziraju na stvaranju modela tijekom interakcije, radi boljeg iskorištavanja podataka učenja. Cilj ovog pregleda bio je dati uvid u pristupe koji se mogu koristiti za usavršavanje ponašanja koje je dobiveno na temelju učenja iz demonstracija. Kako pretraživanje strategije optimizacijskim metodama pruža vrlo jasan i prilagodljiv okvir za primjenu u različitim zadacima, zbog direktnog učenja u realnom okruženju robota i jednostavnog sustava nagrađivanja koji može biti orijentiran isključivo obavljanju zadatka, ovaj koncept prikladan je za samostalno učenje u okviru ovog rada.

5. Iterativno učenje za stohastičke zadatke

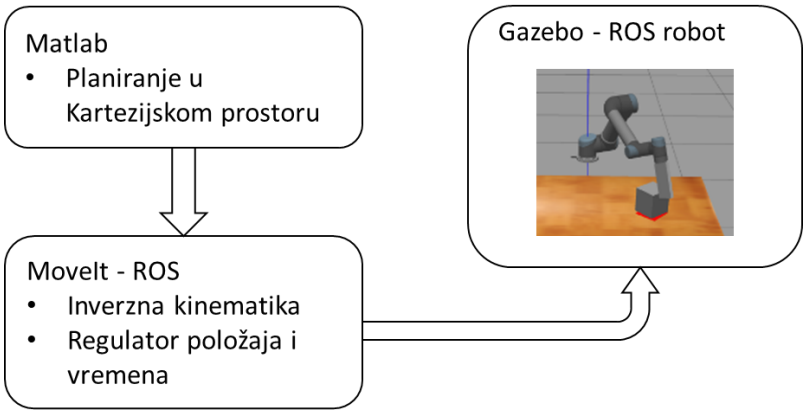
Kako je prikazano u poglavlju o samostalnom učenju robota istraživanjem, cilj robota je naučiti parametre strategije koja ga vodi do uspješnog izvođenja zadatka. Ovo se može ostvariti kroz direktnu interakciju sa stvarnim svijetom ili kroz interakciju sa modeliranom okolinom u kojoj djeluje robot. Sustav nagrađivanja mora na pravi način opisati željeno ponašanje robota ili ishod u određenom zadatku čime u iterativnom procesu učenja utječe na promjenu parametara prema željenom ponašanju robota.

U ovom poglavlju će se predložiti metodologija za iterativno robotsko učenje orijentirano zadatku. Metodologija je bazirana na tzv. BB (eng. *black-box*) optimizacijskom pristupu za pretraživanje parametara strategije opisanom u poglavlju 4.5.1. Istraživanje se provodi kroz direktnu interakciju robota i okoline, s time da je interakcijsko okruženje postavljeno u simulacijskom okruženju. Pristup prikazan u ovom poglavlju temelji se na znanstvenom radu „*Accelerating Robot Trajectory Learning for Stochastic Tasks"* [142].

5.1. Simulacijsko okruženje

Simulacijsko okruženje razvijeno je kao dio ROS sustava. Ono se sastoji od Gazebo simulacijskog paketa [143] i upravljačkog sustava koji je implementiran u MATLAB okruženju. Gazebo je 3D simulacijski paket koji omogućuje simulacije robota u realnim fizičkim okruženjima. On koristi simulator fizike koji može reproducirati pojave iz realnog svijeta u računalnom okruženju, kao što su gravitacijska sila, inercijske sile, trenja, svjetlost itd. On simulira interakcije u realnom vremenu i pruža vrlo dobru platformu za simulaciju robota s obzirom na realne dinamičke uvjete koji se pojavljuju u okolini. Ovo je vrlo korisno, u primjenama gdje realni robot nije dostupan ili se iz sigurnosnih razloga upravljački algoritmi žele testirati u računalnoj domeni. Neki od primjera primjene su: simulacija upravljačkog algoritma drona [144], [145], upravljački algoritam za mobilni robot [146], [147], manipulacijski zadaci [148]. U našem slučaju, ono sadrži UR5 robotsku ruku zajedno sa okolinom u kojoj robot djeluje kako bi ostvario određeni zadatak. Model robotske ruke dan je kao ROS paket (https://github.com/ros-industrial/universal_robot). U Matlab-u implementirano je planiranje trajektorija na operacijskoj razini, a *MoveIt* paket koristi se za

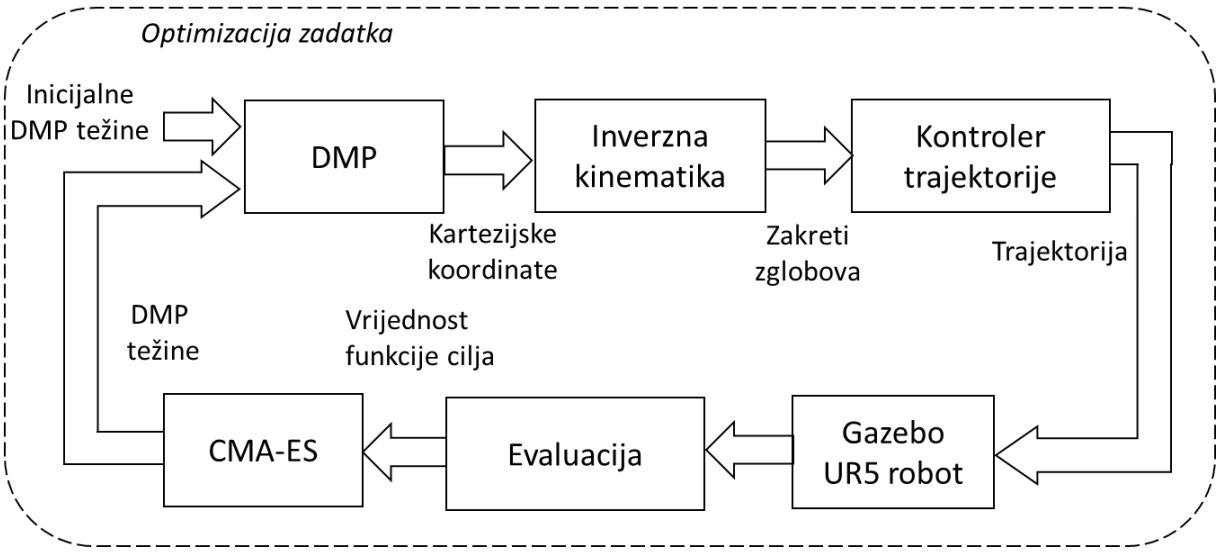
inverznu kinematiku,. Upravljanje simuliranog robota na konfiguracijskom nivou ostvareno je korištenjem pozicijskih kontrolera iz *Ros_control* paketa (slika 5-1.).



Slika 5-1. Razvijeno okruženje za robotsko učenje.

5.2. Optimizacija zadatka

Kao parametrizirani model gibanja odabran je već opisani DMP zapis trajektorije u operacijskom sustavu robota. Sustav je koncipiran tako da inicijalna trajektorija parametrizirana pomoću DMP modela služi kao početna točka potrage za rješenjima. DMP trajektorija se pomoću modela inverzne kinematike robota transformira u konfiguracijski prostor robota, te se trajektorija izvodi na robotu pomoću pozicijskog kontrolera. Ponašanje robota se ocjenjuje na temelju njegovog djelovanja u prostoru na temelju funkcije nagrade. Ova ocjena služi kao ulaz u CMA-ES algoritam koji na temelju istog izračunava nove parametre DMP trajektorije (slika 5-2.).



Slika 5-2. Blok dijagram sustava za iterativno učenje.

Konačni cilj procesa učenja, tj. optimizacije zadatka je minimizacija funkcije cilja. Ista je formulirana kao funkcija za jednokratno ocjenjivanja kvalitete djelovanja robota u njegovom radnom prostoru. Ovaj način vrednovanja robotskog djelovanja je odabran je ispred kontinuiranog načina vrednovanja, kako bi se sačuvala jednostavnost formulacije ove funkcije. Iz istog razloga, kriteriji za ocjenu koji su formulirani unutar funkcije nagrade se odnose isključivo na kvalitetu obavljenog zadatka. Ovdje je moguće uvrstiti i druge kriterije koji se odnose na kvalitetu trajektorije, kao što su minimiziranje akceleracija zglobova, smanjenje potrebne energije prilikom izvođenja, udaljenost od kinematskih ograničenja itd..

5.2.1. Parametarski prostor pretrage

Optimizacija se odvija u kartezijskom sustavu zbog jednostavnijeg i prirodnijeg opisa funkcije cilja zadatka. Isto tako, jedno prethodno istraživanje eksplicitno je pokazalo da je optimizacija problema u domeni u kojoj je postavljena funkcija cilja uvijek brža i robusnija [84]. Generalni cilj prezentiranog pristupa u ovom poglavlju je ostvariti učenje koristeći svih 6 stupnjeva slobode kartezijske trajektorije kako bi se omogućila mogućnost primjene na različite zadatke. Kako je opisano u poglavlju 2.4., DMP trajektoriju karakterizira po jedan transformacijski sustav po stupnju slobode trajektorije, što teoretski znači da bi se trajektorija u potpunosti mogla opisati sa 6 transformacijskih sustava. To ipak nije slučaj jer je dodatni zahtjev da se svaki stupanj slobode može zasebno kontinuirano interpolirati. Iz tog razloga se kod predstavljanja rotacijskog dijela trajektorije ne može odabrati neki od zapisa sa minimalnim brojem parametara (*Euler*, *angle-axis*, *kvaternion* – poglavlje 3.1.), nego je odabran zapis rotacijske matrice čijih devet elemenata posjeduju ovu karakteristiku. Dodavanjem triju translacijskih elemenata, ovo čini trajektoriju sa 12 stupnjeva slobode (transformacijskih sustava DMP-a). Pri tome je svaki stupanj slobode parametriziran sa 10 težina ($n_{bfs} = 10$), što parametarski prostor oblika učene trajektorije opisuje sa 120 parametara.

5.2.2. Inicijalizacija linearnom trajektorijom

Najjednostavniji način kretnja u kartezijskom koordinatnom sustavu je pravocrtno od početne točke do neke krajnje točke, pritom ne mijenjajući orijentaciju. Ovakva trajektorija korištena je za inicijalizaciju sustava za iterativno učenje. Prilikom ovoga startna pozicija robota očitana je za

trenutnu konfiguraciju robota, a krajnja točka je klasificirana iz demonstracija (slično kao u pristupu pokazanom u poglavlju 3.4.1..

5.2.3. Izbor radne konfiguracije

Kod izvršavanja neke trajektorije na robotu, bilo koji zapis trajektorije mora se svesti na razinu upravljačkih naredbi. Za slučaj revolutnog robota kakav se koristi u ovom radu, to su zakreti svakog pojedinog zgloba robotske ruke. Budući da se učenje trajektorija u ovom slučaju provodi u kartezijskom prostoru, prilikom njihovog prevođenja u konfiguracijski prostor putem inverzne kinematike, može doći do različitih problema uzorkovanih kinematskim ograničenjima robota. Neki od njih su prolazak kroz singularne konfiguracije robota ili ne postojanje inverznog kinematskog rješenja. Problem ne postojanja inverznog rješenja se ovdje rješava na način da se trajektorije sa nepostojećim rješenjem bilo koje kartezijske pozicije unutar trajektorije penaliziraju visokom vrijednošću funkcije cilja, te se zbog toga na koncu ne uzimaju u obzir. Problem prolaska kroz singularne pozicije očituje se pojavom velikih akceleracija u konfiguracijskom prostoru tj. u nekom od zglobova tijekom izvršavanja trajektorije. Kako 6-osni robot neku kartezijsku trajektoriju može izvršiti u različitim kinematskim konfiguracijama, ovdje se u obzir uzimaju sva moguća rješenja. Za samo izvršavanje odabire se ona konfiguracija koja uzrokuje najmanje maksimalne akceleracije na svim zglobovima. Dodatno, postavljene su maksimalne prihvatljive vrijednosti akceleracija i brzina za svaki zglob (prag akceleracije: 80 °/s^2, prag brzine: 60 °/s). Ukoliko maksimalne vrijednosti pređu ove vrijednosti, trajektorija se penalizira visokom vrijednosti, kao i u slučaju ne postojanja inverznog rješenja.

5.2.4. Određivanje inicijalnog koraka učenja

Kako je prikazano u poglavlju o pretraživanju strategija, CMA-ES algoritam ima jedan otvoreni parametar, a to je korak učenja σ. Postoji mogućnost zadavanja koraka učenja za svaki parametar pretrage posebno. Za određivanje početnih parametara učenja za zadatke koji su predstavljeni u ovom poglavlju, korišten je empirijski pristup. Parametri su prvo postavljeni na vrlo male vrijednosti te je promatrano ponašanje robota u zadatku. Ukoliko robot nije pokazivao dovoljno istraživačkog djelovanja, parametri koraka učenja su se povećavali sve dok to nije zamijećeno.

5.2.5. Evaluacija procesa učenja

Budući da je proces učenja u stvarnim zadacima vrlo često stohastičan, a primjeri u ovom radu su iste prirode, ovdje se koristio fiksan broj iteracija za učenje koji iznosi 300, kako bi se izbjeglo pre dugo vrijeme izvođenja učenja.

Kako već spomenute stohastičke funkcije nije moguće na prvi pogled jednostavno vrednovati, ovdje će se predložiti dva kriterija koji će ove funkcije transformirati u oblike koji pružaju uvid u kvalitetu procesa učenja. Prvi od njih je trenutna minimalna vrijednost funkcije cilja tijekom izvođenja iterativnog učenja C_{best} (5.1). Ovaj kriterij prati najbolje postignuto rješenje u dosadašnjem procesu učenja do trenutne iteracije n. Ovo daje uvid u brzinu pronalaska najboljeg rješenja (trajektorije).

$$C_{best} = min\,(Cost_{n=1}^{n_{current}}) \tag{5.1}$$

Drugi kriterij je prosječna vrijednost funkcije cilja C_{mean} (5.2) dobivena do trenutne iteracije n. Ovaj kriterij pruža uvid u generalnu kvalitetu samog procesa potrage za rješenjima. Potraga koja rezultira u vrijednostima sa višim prosječnim vrijednostima, vrlo je vjerojatno loše inicijalizirana ili su parametri pretrage loše postavljeni, u odnosu na pretragu koja ima niže vrijednosti ove funkcije.

$$C_{mean} = mean\,(Cost_{n=1}^{n_{current}}) \tag{5.2}$$

Kod stohastičkih zadataka moguće je za iste parametre trajektorije dobiti različite ishode tj. vrijednosti funkcije cilja. Ovo je posebno izraženo kod zadataka kod kojih je izražen utjecaj dinamike okoline. Kako bi se spriječio nastanak lažno pozitivnih rješenja, ovdje je korišten princip uprosječavanja rješenja. Ovo podrazumijeva da se zadatak sa istim parametrima trajektorije izvršavao nekoliko (tri) puta te se kao finalna vrijednost funkcije cilja uzela prosječna vrijednost ocjena svih pojedinačnih pokušaja.

5.3. Umetanje – optimizacija zadatka

Prvi scenarij na kojem je testirana primjena predložene metodologije za iterativno podešavanje trajektorije je zadatak umetanja, koji je klasičan robotski zadatak prilikom operacija robotskog sklapanja, a javlja se i u drugim okruženjima. Prilikom ovog zadatka, krećući iz neke proizvoljne inicijalne pozicije pokušava predmet koji je u zahvatu s robotom umetnuti u drugi element na za to predviđeno mjesto. Tijekom izvršavanja ovog zadatka cilj je zadatak izvršiti upotrebom što manje sile između oba elementa. U standardnim industrijskim aplikacijama ovo se postiže

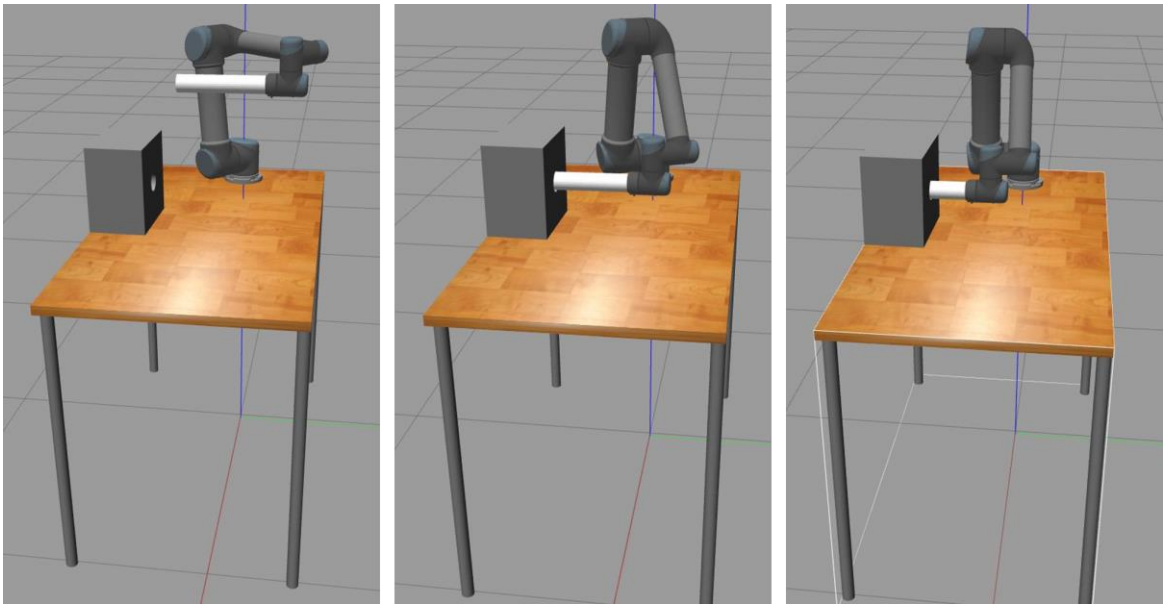
upotrebom mjerenja sa senzora sila i momenata, gdje se preciznim mjerenjima i podešavanjem gibanja robota kompenzira nastanak bilo kakvih prekomjernih sila između robota i elementa na kojem se izvodi umetanje.

Budući da se iterativno učenje robota izvodi u simulacijskom okruženju, gdje nije moguće dobiti pouzdane podatke za evaluaciju zadatka prema izmjerenim silama, ovdje je zadatak umetanja koncipiran na način da se pojava bilo kakve sile očituje u pomaku predmeta na kojem se radi operacija umetanja – umetanje se izvodi u horizontalnoj ravnini.

Funkcija korištena za ocjenu robotskog djelovanja sastoji se od dva kriterija. Prvi se odnosi na točnost pozicioniranja robota na zadano mjesto unutar elementa za umetanje. Drugi se indirektno odnosi na minimizaciju sila koje se javljaju tijekom izvođenja zadatka, a to je pomak drugog elementa (element za umetanje) sa inicijalne pozicije. Oba kriterija su izvedena kao euklidske udaljenosti od željene pozicije do ostvarene pozicije. Funkcija cilja C^{pih} dana je u obliku sljedeće jednadžbe:

$$C^{pih} = \left\| P_{box\ final} - P_{box\ initial} \right\| + \left\| P_{center\ of\ mass} - P_{TCP\ final} \right\|. \qquad (5.3)$$

Simulacijsko okruženje u kojem se izvodi učenje ovog zadatka vidljivo je na slici.5-3.



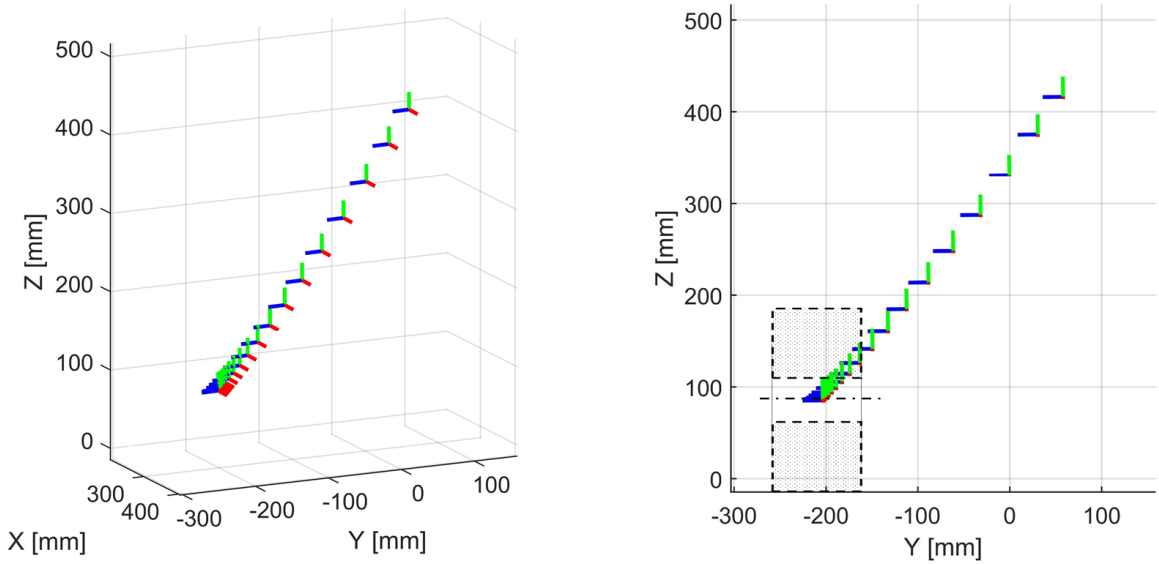
Slika 5-3. Simulacijsko okruženje za zadatak umetanja. Promjer cilindra za umetanje je 40 mm dok je promjer rupe 50 mm (iste dimenzije korištene su i u nastavku rada). Prikazana su tri položaja robota u sekvenci trajektorije koja uspješno obavlja zadatak.

Budući da je ciljna pozicija zadatka umetanja jednoznačno određena centrom kutije, a početna pozicija robota je dostupna kod svake nove konfiguracije, početna i krajnja pozicija trajektorije za rješavanje ovog zadatka ne zahtijevaju učenja. Parametarski prostor ovog zadatka stoga predstavljaju samo parametri oblika trajektorije, tj. težine baznih funkcija nelinearne funkcije oblika DMP modela trajektorije (tablica 5-1.).

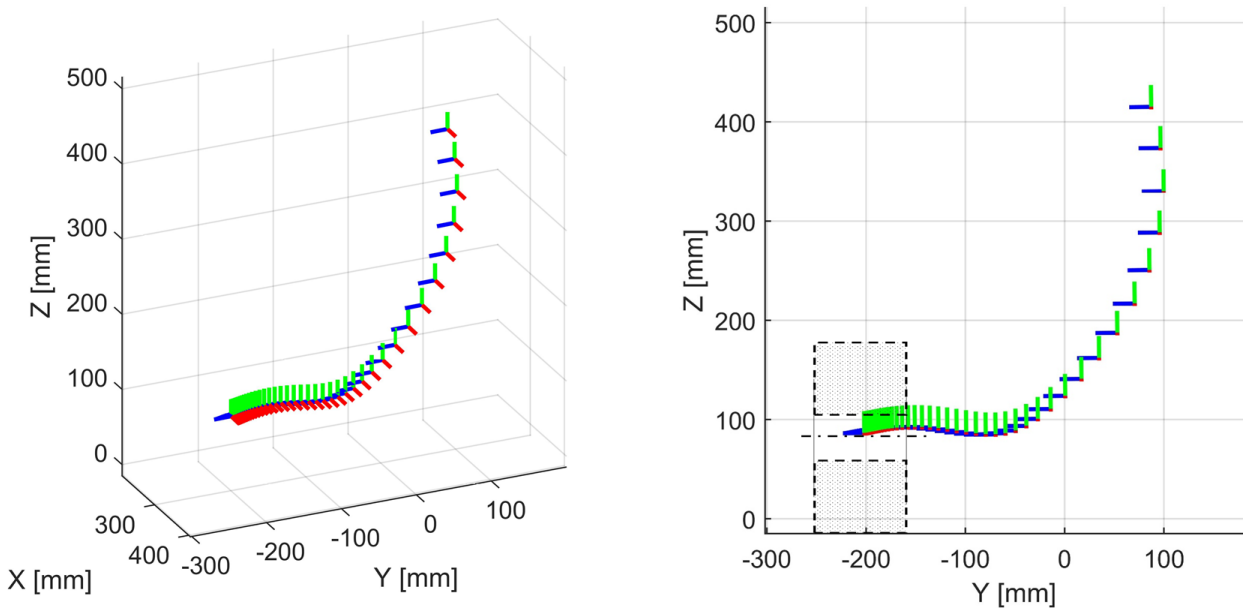
Tablica 5-1. Parametarski prostor za zadatak umetanja.

oblik trajektorije
n_{dmp} x n_{bfs} 12 x 10 $w_1, w_2, \ldots w_{120}$
120 parametara

Linearna trajektorija korištena za inicijalizaciju ovog zadatka vidljiva je na slici 5-4. Ista seže od početne pozicije robota u prostoru do pozicije koja je definirana pomoću centra mase ciljnog elementa. Naučena trajektorija prikazana je na slici (5-5.). Vidljivo je da je trajektorija poprimila oblik trajektorije koji je prikladan za obavljanje zadatka, što se isto tako pokazalo i u simulaciji.

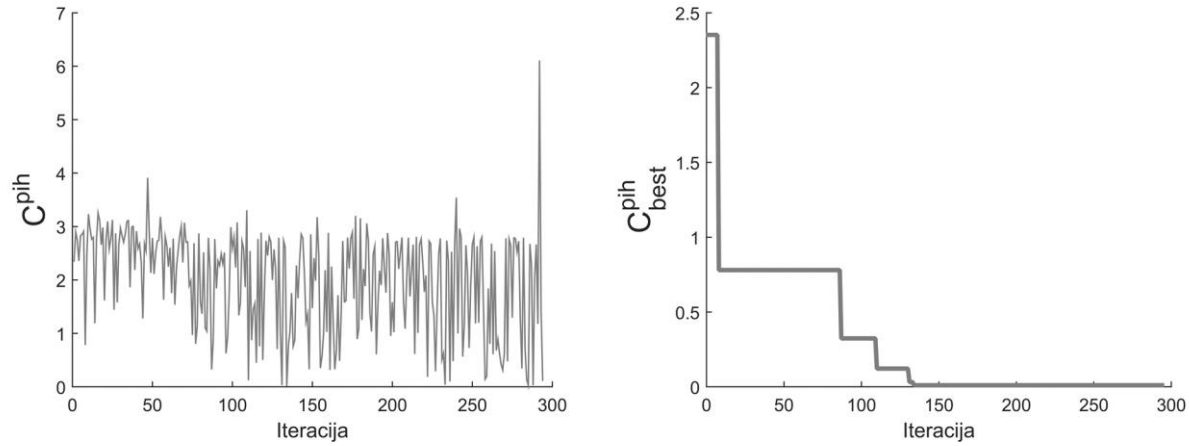


Slika 5-4. Linearna inicijalna trajektorija za zadatak umetanja.



Slika 5-5. Trajektorija nakon procesa učenja za zadatak umetanja.

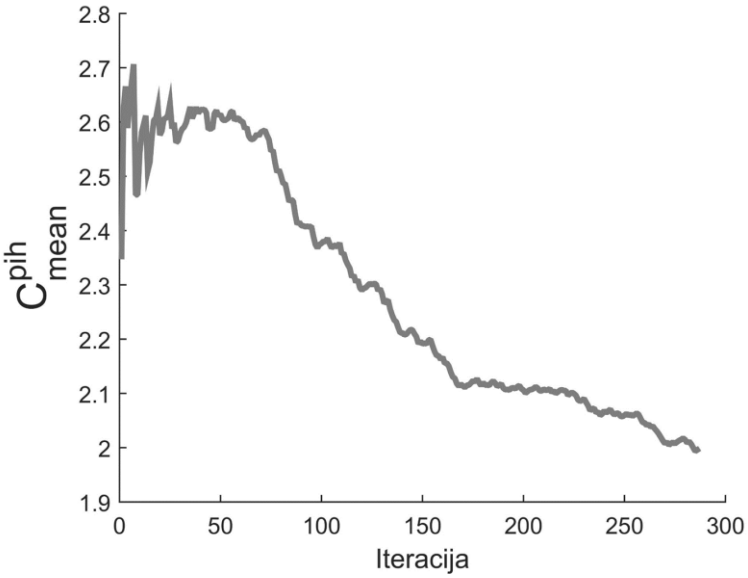
Proces samog učenja ovog zadatka vrlo je stohastičan iz razloga što male promjene u parametrima trajektorije mogu izazvati nepredvidive iznose funkcije cilja. Ovo je zbog činjenice da se pomak kutije prilikom odguravanja ne može predvidjeti tj. dovesti u vezu sa robotskom trajektorijom.



Slika 5-6. Vrijednosti funkcije cilja dobivene tijekom procesa učenja (lijevo). Najbolje trenutno rješenje dobiveno u procesu učenja (desno).

Analizom procesa učenja (pretrage) vidljivo je da se kvalitetno rješenje pronalazi nakon 125 iteracija, gdje je funkcija najboljeg trenutnog rješenja pala na vrijednost 0, što je ujedno i minimalna vrijednost funkcije cilja.

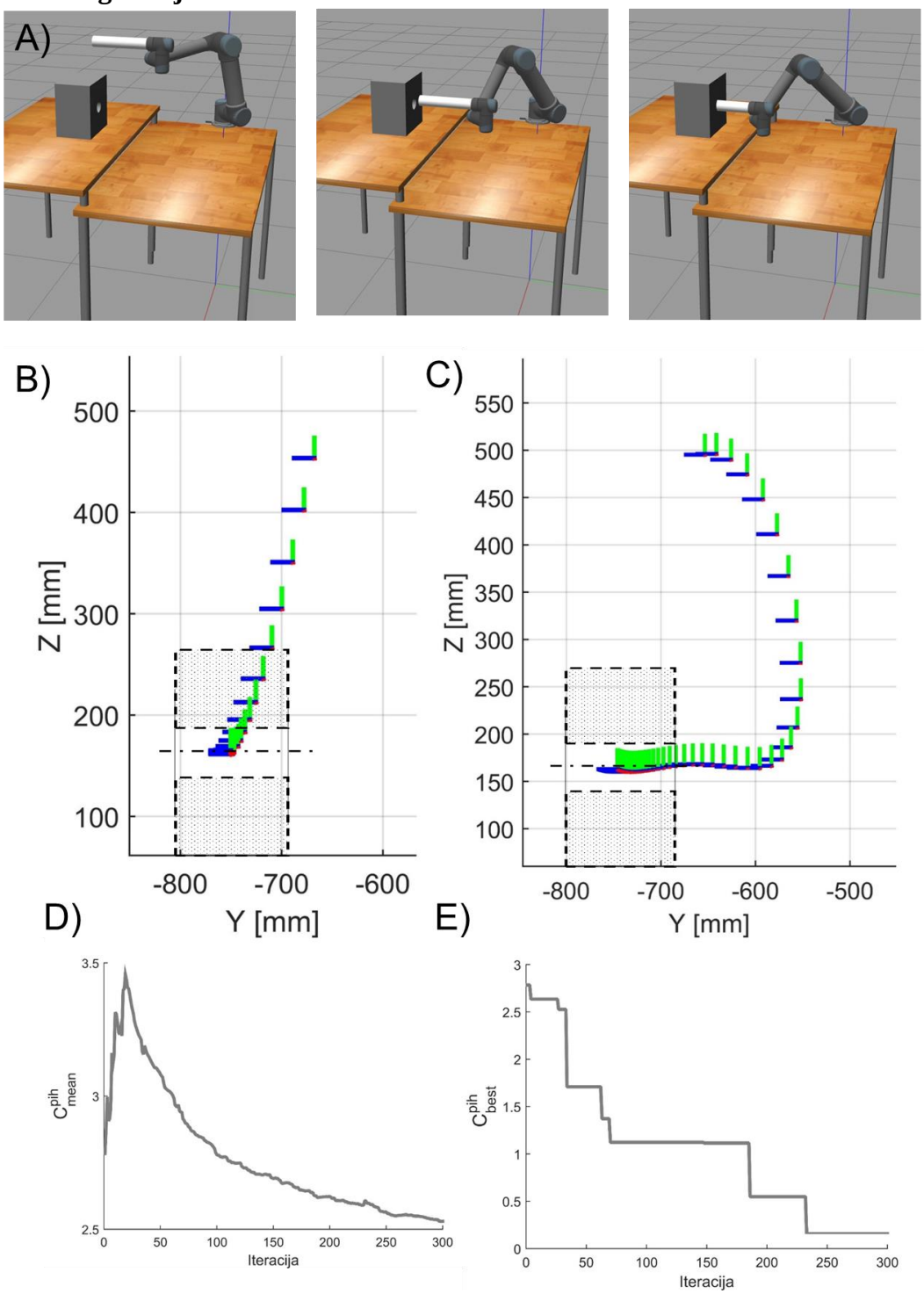
Graf srednjih vrijednosti funkcije cilja dan je na slici 5-7. Iz njega je vidljiva određena konvergencija procesa učenja zadatka, što predstavlja puno korisniju informaciju negoli izvorna funkcija cilja.



Slika 5-7. Srednje vrijednosti dosadašnjih rješenja.

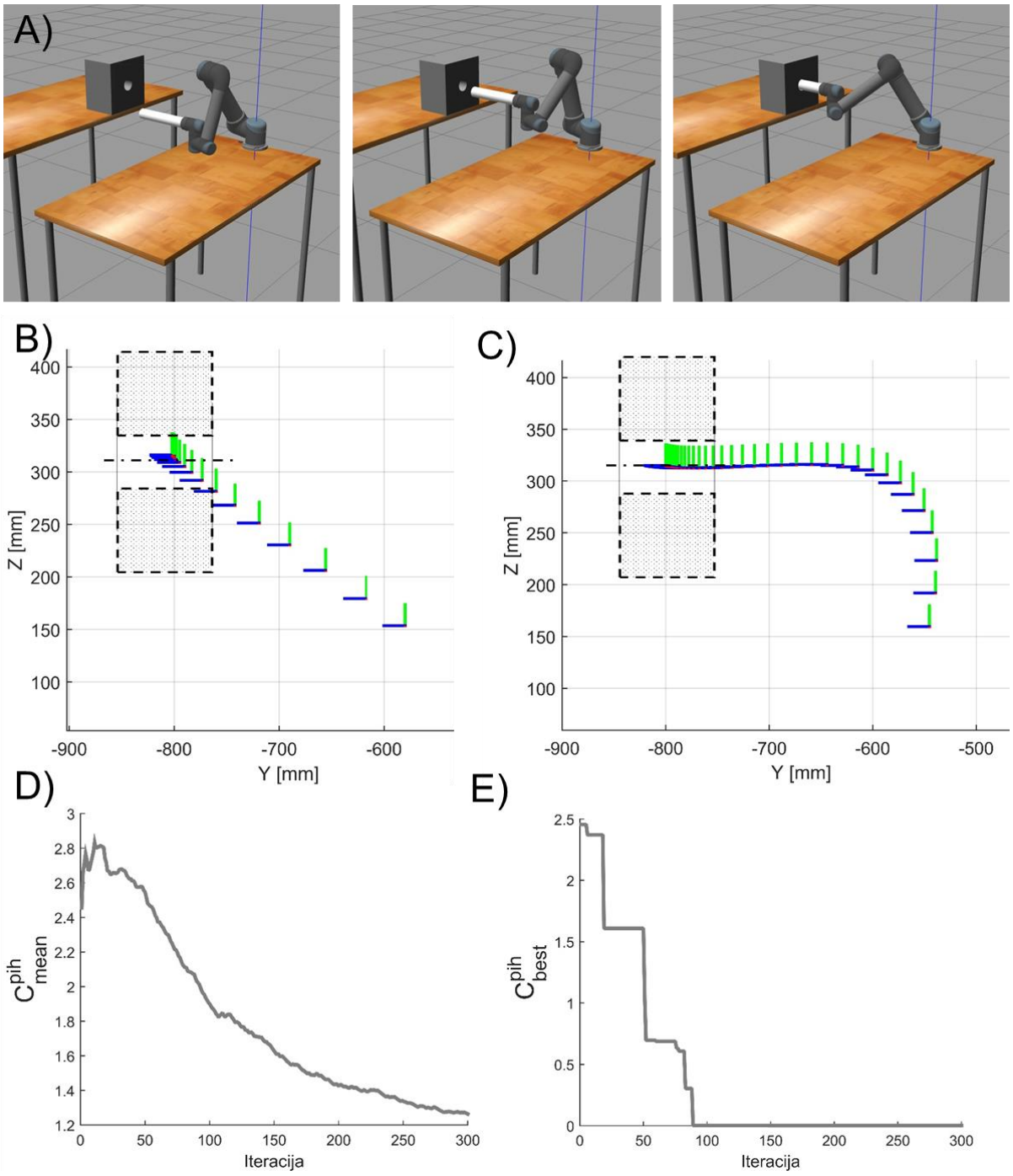
Vrijeme trajanja pretrage od 300 iteracija za ovaj zadatak na standardnom računalu sa i7-6700HQ procesorom i 16 GB RAM memorije i maksimalnom akceleracijom simulacijskog vremena iznosilo je oko 30 minuta. U ovo vrijeme ulazi ažuriranje parametara trajektorije, planiranje trajektorije, rješavanje inverzne kinematike, te samo izvođenje zadatka u simulaciji i evaluacija.

5.3.1. Konfiguracija 2



Slika 5-8. Rezultati dobiveni za drugu konfiguraciju zadatka umetanja. A) sekvenca izvršavanja trajektorije, B) inicijalna trajektorija, C) naučena trajektorija D) proces iterativnog učenja. prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, E) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja.

5.3.2. Konfiguracija 3



Slika 5-9. Rezultati dobiveni za treću konfiguraciju zadatka umetanja*. A) sekvenca izvršavanja trajektorije, B) inicijalna trajektorija, C) naučena trajektorija D) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, E) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja.*

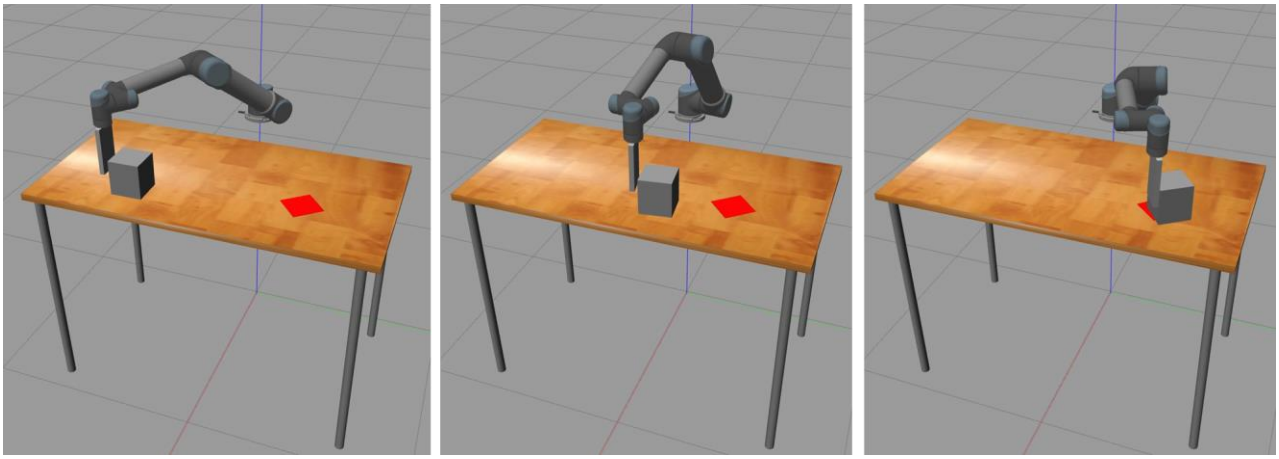
5.4. Odguravanje predmeta – optimizacija zadatka

Drugi scenarij na kojem će se ispitati metodologija iterativnog učenja je zadatak odguravanja predmeta na željeno mjesto. Iz nekog početnog stanja (pozicije), predmet se treba odgurati na željenu poziciju u istoj ravnini. Pri tome se samo koristi kruti ravni alat na robotu koji služi kao lopatica za guranje. Zadatak nije klasičan industrijski ali je često korišten za dokazivanje manipulativnih sposobnosti robota. Ono zašto je zadatak vrlo zanimljiv je to što na proces gibanja kutije po stolu utječu razni parametri kao što su: faktor trenja između robotskog alata i kutije, faktor trenja između kutije i stola, veličina samog alata za guranje. Uz ovo zadatak je vrlo stohastičan, jer male promjene parametara trajektorije mogu izazvati nepredvidive rezultate u funkciji cilja. Iz tog razloga vrlo je izazovno pronaći trajektoriju koja je sposobna obaviti ovaj zadatak za svaki specifični slučaj.

S ciljem zadržavanja jednostavnosti funkcije cilja, ona je ovdje definira kao euklidska udaljenost između željene pozicije na koju je predmet trebao biti odguran i postignute pozicije kutije nakon procesa guranja (5.4).

$$C^{sweeping} = \left\| P_{box\ target} - P_{box\ initial} \right\| \tag{5.4}$$

Simulacijsko okruženje postava za zadatak odguravanja vidljivo je na slici 5-10. Isto tako vidljive su i tri konfiguracije robota odabrane iz trajektorije koja uspješno odgurava kutiju na zadano mjesto.



Slika 5-10. Simulacijsko okruženje za zadatak odguravanja. Dimenzije plohe alata za odguravanje su: 100 x 200 mm, dok je kutija u obliku kocke sa stranicama od 100 mm. Prikazana su tri položaja robota u sekvenci trajektorije koja uspješno obavlja zadatak.

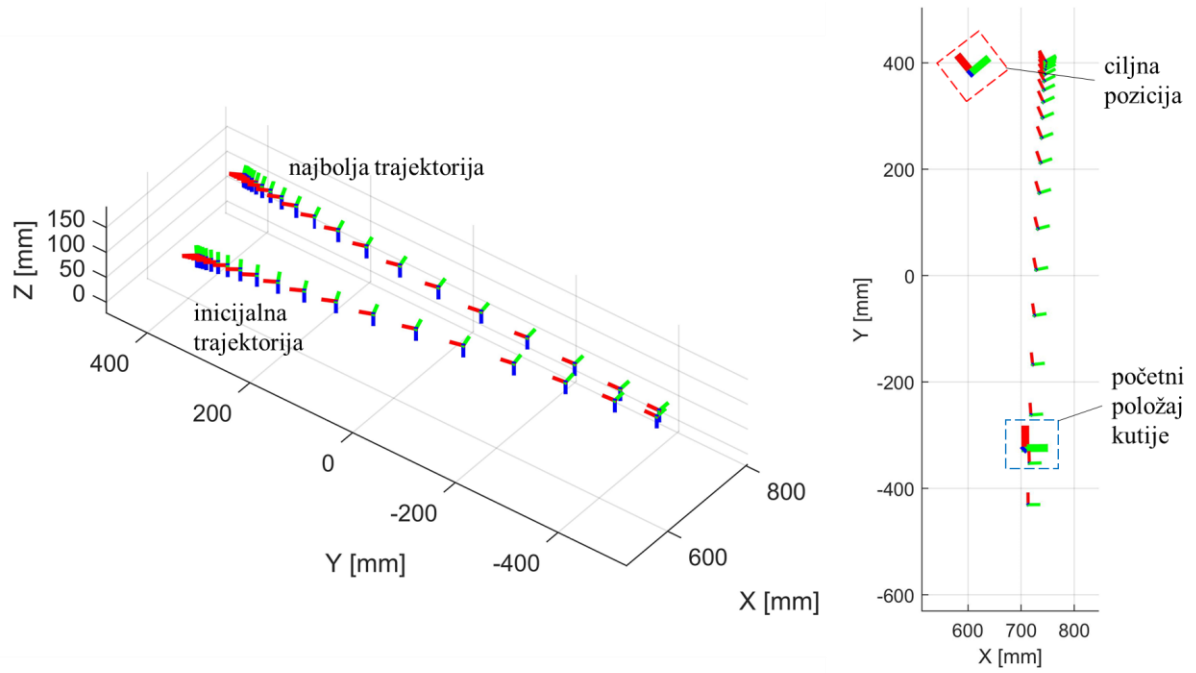
Kako je ovaj zadatak za razliku od zadatka umetanja ovisan o dinamičkim parametrima okoline, parametri oblika trajektorije nisu jedini koji mogu utjecati na uspješno izvršavanje zadatka. Primjerice, vrlo je bitna brzina izvršavanja trajektorije. Kako DMP parametrizacija ima unaprijed definirane kontinuirane profile brzina i ubrzanja, nije moguće mijenjati brzine u svakoj točci trajektorije. Ipak, moguće je skalirati brzinu izvršavanja cjelokupne trajektorije. Ovaj pristup se ovdje koristi kako bi se brzina izvođenja dodala u parametarski prostor pretraživanja kod učenja trajektorije. Drugi osnovni parametri koji opisuju DMP trajektoriju su početna i krajnja pozicija. Kako početna i krajnja pozicija nisu predmet konfiguracije zadatka, nego su to početni i ciljni položaji kutije, isti će se koristiti kao parametri u prostoru pretrage. Unaprijed se može pretpostaviti da je krajnji položaj trajektorije vrlo bitan parametar za uspješno obavljanje zadatka jer ima značajan utjecaj kod završnog pozicioniranja kutije. Kada se obuhvate svi navedeni parametri, prostor pretrage čine ukupno 145 parametara, što je vidljivo u tablici 5-2.

Tablica 5-2. Parametarski prostor za zadatak odguravanja.

početna pozicija robota	oblik trajektorije	krajnja pozicija robota	faktor brzine izvođenja
[T, R] 12 x 1 $x, y, z, r_{11}, \dots r_{33}$	n_{dmp} x n_{bfs} 12 x 10 $w_1, w_2, \dots w_{120}$	[T, R] 12 x 1 $x, y, z, r_{11}, \dots r_{33}$	1 v_{scl}
145 parametara			

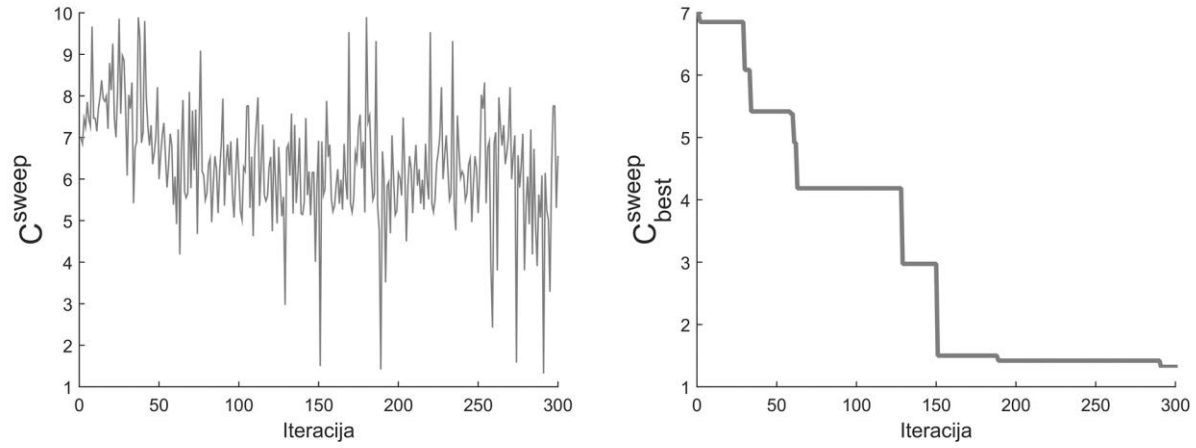
Ovaj zadatak je kao i prethodni inicijaliziran linearnom trajektorijom, koja seže od neke početne pozicije robota do ciljne pozicije, koja je ovdje definirana sa položajem ciljne pozicije (slika 5-11.).

Najbolja pronađena trajektorija nakon procesa optimizacije prikazana je na istoj slici. Može se primijetiti kako je najbolja trajektorija zadržala približno linearan oblik ali kraj ima u posve drugom prostornom položaju. Pronalazak ovakvog rješenja omogućila je opisana proširena parametrizacija prostora pretrage.



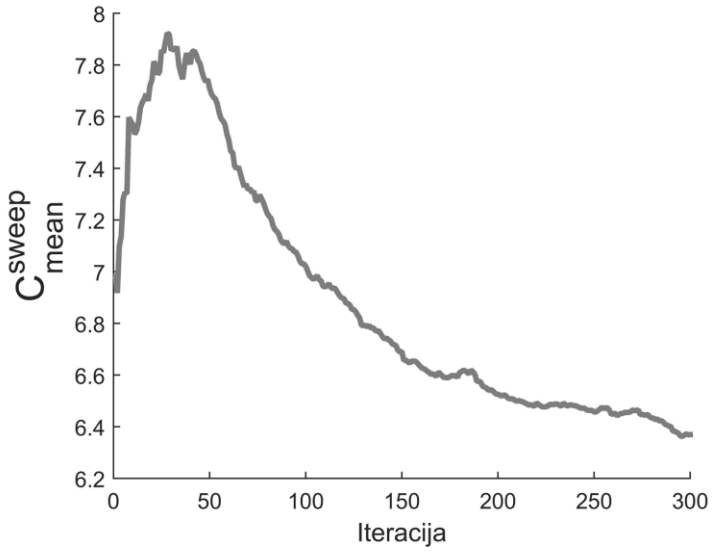
Slika 5-11. Linearna inicijalna trajektorija i naučena trajektorija za zadatak odguravanja (lijevo). Naučena trajektorija u odnosu na početni i ciljni položaj kutije (desno).

Iterativni proces pretrage (učenja) očekivano je rezultirao stohastičkom funkcijom cilja. Prema kriteriju najboljeg trenutnog rješenja, približno dobro rješenje pronađeno je nakon 150 iteracija. Pri tome je vidljivo da pronađena trajektorija u potpunosti ne minimizira funkciju cilja što znači da potpuno dobro rješenje nije pronađeno unutar procesa učenja od 300 iteracija.



Slika 5-12. Vrijednosti funkcije cilja dobivene tijekom procesa učenja (lijevo). Najbolje trenutno rješenje dobiveno u procesu učenja (desno).

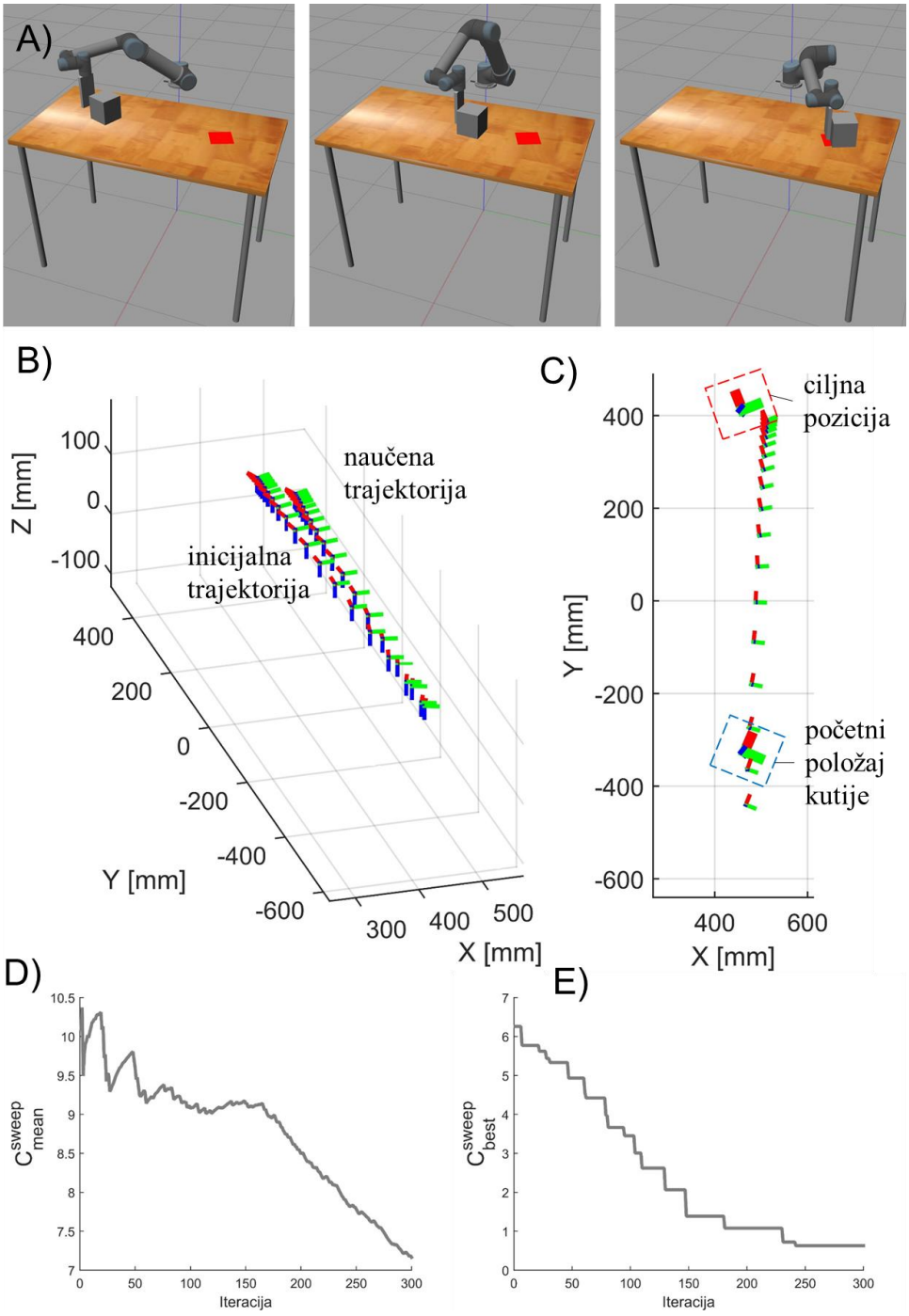
Graf srednjih vrijednosti funkcije cilja dan je na slici 5-13. Kao i u prethodnom zadatku, iz njega je vidljiva određena konvergencija procesa učenja zadatka.



Slika 5-13. Srednje vrijednosti dosadašnjih rješenja.

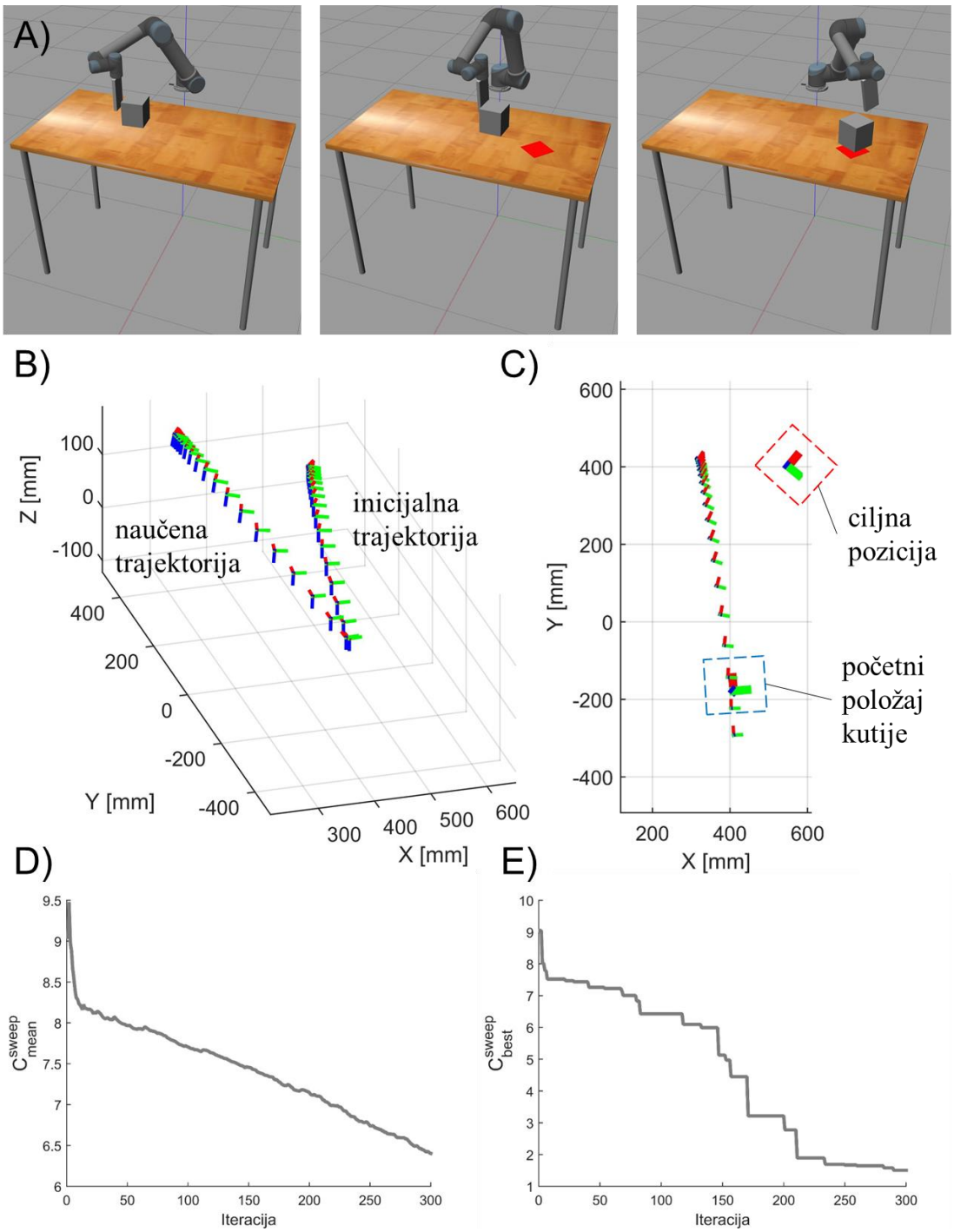
Vrijeme trajanja pretrage od 300 iteracija za ovaj zadatak na istoj računalnoj konfiguraciji kao i prethodni zadatak iznosilo je oko 45 minuta.

5.4.1. Konfiguracija 2



Slika 5-14. Rezultati dobiveni za treću konfiguraciju zadatka odguravanja. A) sekvenca izvršavanja trajektorije, B) usporedba inicijalne i naučene trajektorije, C) prikaz naučene trajektorije u horizontalnoj ravnini u odnosu na početni i ciljni položaj kutije, D) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, E) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja.

5.4.2. Konfiguracija 3



Slika 5-15. Rezultati dobiveni za treću konfiguraciju zadatka odguravanja. A) sekvenca izvršavanja trajektorije, B) usporedba inicijalne i naučene trajektorije, C) prikaz naučene trajektorije u horizontalnoj ravnini u odnosu na početni i ciljni položaj kutije, D) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, E) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja.

5.5. Zaključci o poglavlju

Postupak učenja iterativnim pretraživanjem parametara DMP trajektorije prikazan u ovom poglavlju pokazao je da omogućuje pronalazak zadovoljavajućih rješenja za dva tipa zadataka. Uz ovo moguće je donijeti neke zaključke o karakteristikama ovog pristupa.

Korištenje simuliranog okruženja za robotsko učenje donosi niz prednosti u odnosu na učenje u realnom okruženju. Kod ažuriranja parametara trajektorije moguće je dobiti trajektorije sa neočekivanim putanjama koje rezultiraju kolizijama robota bilo sa samim sobom ili sa drugim predmetima u okolini što dovodi do pojave sila koje mogu biti vrlo opasne po okolinu. U simuliranom okruženju kolizije ne predstavljaju problem, te ih je najčešće moguće detektirati i zanemariti takve trajektorije. Isto tako, u ovom pristupu korištena je mogućnost ubrzanog vremena, gdje je izvođenje simulacija ovisno o snazi računala, moguće ubrzati nekoliko puta u odnosu na realno vrijeme izvršavanja. Ovo značajno skraćuje proces učenja u odnosu na učenje u realnim okruženjima. Jedini nedostatak ovog pristupa je da simulacijska okolina nikada potpuno točno ne modelira sve parametre okoline. Zbog toga se kod puštanja ovako naučene trajektorije na realnom robotu, mogu pojaviti nepredviđena poklapanja sa simulacijskim rezultatima. Ova problematika će biti dodatno razrađena u poglavlju broj 6.

Algoritmi za pretraživanje strategija kao i BB optimizacijski algoritmi najčešće imaju otvorene parametre kojima se može odrediti razina (korak) pretrage. Kod korištenog CMA-ES algoritma to je inicijalni korak pretrage. Kao i kod generiranja trajektorije (u trećem poglavlju) veći opseg pretrage ima za posljedicu mogućnost pronalaska globalnih rješenja, dok se povećava rizik od neispravnih rješenja uslijed kinematskih i kolizijskih ograničenja robota. Manji inicijalni opseg pretrage, s druge strane, uzrokuje lokalniju pretragu za rješenjima, ali smanjuje rizik od pojave potpuno neprihvatljivih ili opasnih rješenja. Kod učenja trajektorija optimalan način učenja je onaj koji je siguran za okolinu i samog robota, što znači da je poželjno držati parametre pretrage što nižima. Ovo ima za posljedicu potrebu za dobrom inicijalizacijom samog postupka pretraživanja. Inicijalizacija linearnom trajektorijom korištena u ovom poglavlju predstavlja samo jednu od mogućih strategija za izbor inicijalnih trajektorija, dok postoje i kvalitetniji načini za inicijalizaciju, kao što su to LfD pristupi. Ovome ide u prilog i činjenica da robot ima i ograničen skup kinematskih rješenja što značajno sužava prostor valjanih rješenja. Ovo je posebno izraženo ukoliko se pretraga vrši u operacijskom prostoru robota, gdje se zbog nelinearne konverzije u konfiguracijski prostor ne može lagano ograničiti prostor valjanih rješenja.

6. Učenje iz demonstracija kao baza za iterativno učenje

Osnovne karakteristike neke savladane ili naučene vještine su: njezina mogućnost primjene u različitim okolnostima (eng. *generalizability*) i točnost primjene u svim situacijama. Pristup za učenje iz demonstracija opisan u drugom poglavlju omogućuje analizu različitih izvedbi istog zadatka i pronalazak generalno bitnih dijelova točaka trajektorije. Time je ovaj pristup prioritetno orijentiran prema pronalasku generalnih rješenja zadatka u obliku trajektorija. Pri tome ova rješenja nisu eksplicitno orijentirana zadatku što ostavlja prostor za neuspješno ili netočno izvršavanje ako promatramo cilj zadatka. Ovo je posebno izraženo kod zadataka koji imaju izraženo međudjelovanje elemenata u okolini, jer ovaj pristup učenju ne uzima u obzir uvjete okoline (trenje, gravitacija). Primjer ovakvog zadatka je zadatak odguravanja u prethodnom poglavlju.

S druge strane, pristup iterativnom učenju iz prošlog poglavlja je zbog svoje unaprijed definirane funkcije cilja primarno orijentiran na ispunjavanje zadatka u njegovoj trenutnoj konfiguraciji. Njegova jedina svrha je prilagoditi parametre trajektorije na način da u jednoj epizodi dođe do njegovog što boljeg izvršavanja. On kao takav ne pruža značajne generalizacijske sposobnosti.

Ljudi znanje i vještine jedni na druge prenose putem fizičkih demonstracija koje su najčešće popraćene glasovnim objašnjenjima. Ovaj oblik interakcije dovoljan je za razvijanje početnog znanja nekog pojedinca o nekoj temi ili vještini. Ipak, ovo početno znanje najčešće nije dovoljno za potpuno samostalno svladavanje određene vještine. U ovoj fazi potreban je samostalni rad pojedinca da bi se direktnim radom u okolini inicijalni set znanja nadopunio specifičnim informacijama iz radne okoline. Primjerice, to mogu biti sile i momenti koji se javljaju prilikom rukovanja nekim alatima, koje je nemoguće prenijeti putem demonstracija i govorom.

Korištenjem do sada pokazane metodologije, moguće je postići mehanizam učenja analogan ovom prirodnom modelu prijenosa znanja. Već je u ranijim istraživanjima prikazano da se LfD metode mogu koristiti za generalizaciju ljudskih demonstracija te na osnovu toga mogu pružati dobre inicijalne trajektorije za nove konfiguracije zadatka [43], [44]. Pretraživanje strategija s druge strane može se koristiti za lokalnu optimizaciju trajektorija kako bi se ispravile eventualne netočnosti naučenog LfD modela.

Kako bi dobili sustav za učenje koji je sadrži obje glavne karakteristike inteligentnog usvajana vještina sa početka poglavlja, ova dva pristupa će se objediniti. U ovom poglavlju će se predstaviti model učenja koji kombinira učenje iz demonstracije i iterativno učenje. Učenje iz demonstracije će se koristiti s ciljem rješavanja problema generalizacije trajektorije, a iterativno učenje s ciljem finog podešavanja ovako generalizirane trajektorije. Predstavit će se pristup koji koristi LfD učenje kako bi se izdvojile informacije iz malog broja ljudskih demonstracija i služi za inicijalizaciju robotske trajektorije. Trajektorija generirana od strane LfD modela parametrizira se i dodatno podešava pretraživanjem strategije s obzirom na funkciju cilja zadatka. Pristup prikazan u ovom poglavlju temelji se na znanstvenom radu „*Accelerating Robot Trajectory Learning for Stochastic Tasks"*, [142].

6.1. Pregled metodologije

Kod iterativnog učenja predstavljenog u prethodnom poglavlju, može se očekivati da pretraga koja je inicijalizirana trajektorijom koja je blizu rješenja bude puno kraća i točnija od loše inicijalizirane pretrage. Ideja predloženog pristupa je da se CMA-ES algoritam kod iterativnog učenja inicijalizira pomoću trajektorije koja je dobivena na temelju metode za učenje iz demonstracije predstavljene u poglavlju broj 2. Pri tome će se kombinirati i karakteristike obaju metoda. Naime, generiranje trajektorije zasnovano na optimizaciji temeljem prolaznih točaka i prepreka (poglavlje 3.4.) generira trajektoriju koja je sposobna pronaći rješenja koja ne sadržavaju koliziju i prenosi informaciju dobivenu iz analize demonstracija. S druge strane, iterativno učenje je isključivo orijentirano optimizaciji trajektorije s obzirom na zadatak. Kombiniranjem ovih metoda predlaže se metodologija za učenje (slika 6-1).

Slika 6-1. Blok dijagram predložene metodologije koja kombinira učenje iz demonstracija i iterativno učenje.

Pristup koristi već obrađenu metodu klasifikacije koordinatnih sustava kako bi analizirali demonstracije, te optimizacijsku metodu generiranja trajektorije predstavljenu u ovom radu kako

bi generirali trajektoriju za svaki pojedini slučaj (konfiguraciju) zadatka. Ovakva trajektorija je inicijalna trajektorija i služi kao ulaz u iterativnu fazu učenja tj. optimizaciju trajektorije prema zadatku.

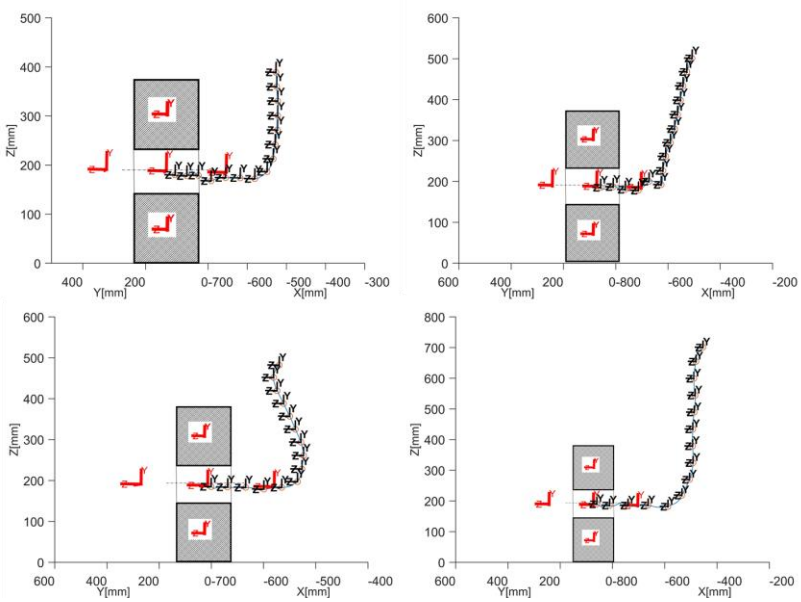
Gledano sa RL stajališta, cilj optimizacije zadatka je ukloniti nedostatke inicijalne trajektorije uzrokovane vanjskim parametrima koji nisu modelirani u sklopu zadatka (dinamika sustava). U osnovi je ovo pristup podržanom učenju bez modeliranja, sa parametriziranim izvršnim djelovanjem. Inicijalizacija koraka učenja za proces iterativnog učenja je kao i u poglavlju 5 provedena empirijski. Budući da se zbog kvalitetnije inicijalizacije očekuju inicijalne trajektorije koje su bliže pravom rješenju nego u slučaju inicijalizacije linearnom trajektorijom, korak pretrage u ovom slučaju je dobiven kao četvrtina onoga korištenog u zadacima u poglavlju 5.

Predloženi hibridni model učenja validiran je na ista dva zadatka koja su obrađena u poglavlju broj 5. s ciljem usporedbe jednog i drugog pristupa tj. načina inicijalizacije. Kako bi omogućili analizu prostornih koordinatnih sustava, svaki zadatak je demonstriran za najmanje tri različite konfiguracije zadatka.

6.2. Zadatak umetanja

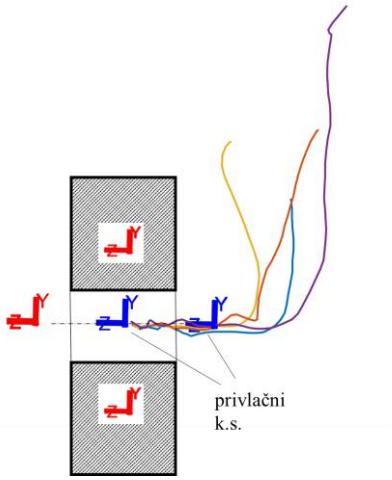
6.2.1. Demonstracije

Korištenjem iste metode kinestetičkog učenja koja je prikazana u drugom poglavlju zadatak umetanja demonstriran je za četiri različita početna prostorna položaja (slika 6-2.). Trajektorije su uzorkovane i transformirane u svaki od koordinatnih sustava od interesa.



Slika 6-2. Demonstracije zadatka umetanja na realnom robotskom sustavu.

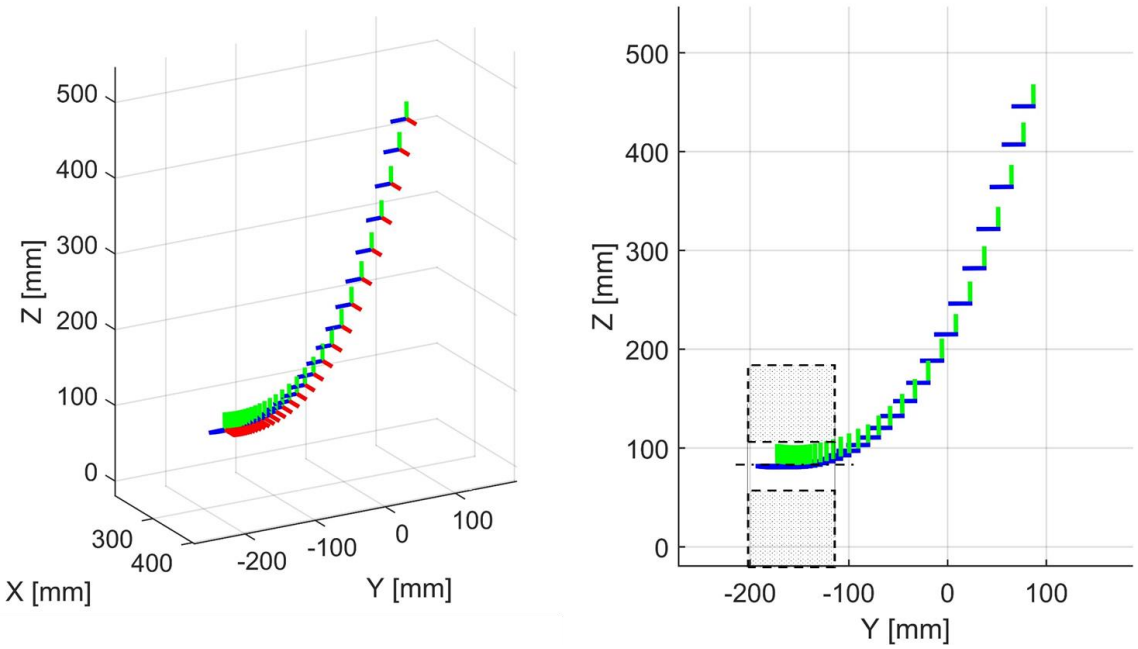
Temeljem analize aktivnosti svakog od ovih koordinatnih sustava, dobivena su dva privlačna koordinatna sustava. Jedan od njih je u središtu objekta sa prolaznom rupom, a drugi je neposredno ispred ulaza u rupu (slika 6-3.).



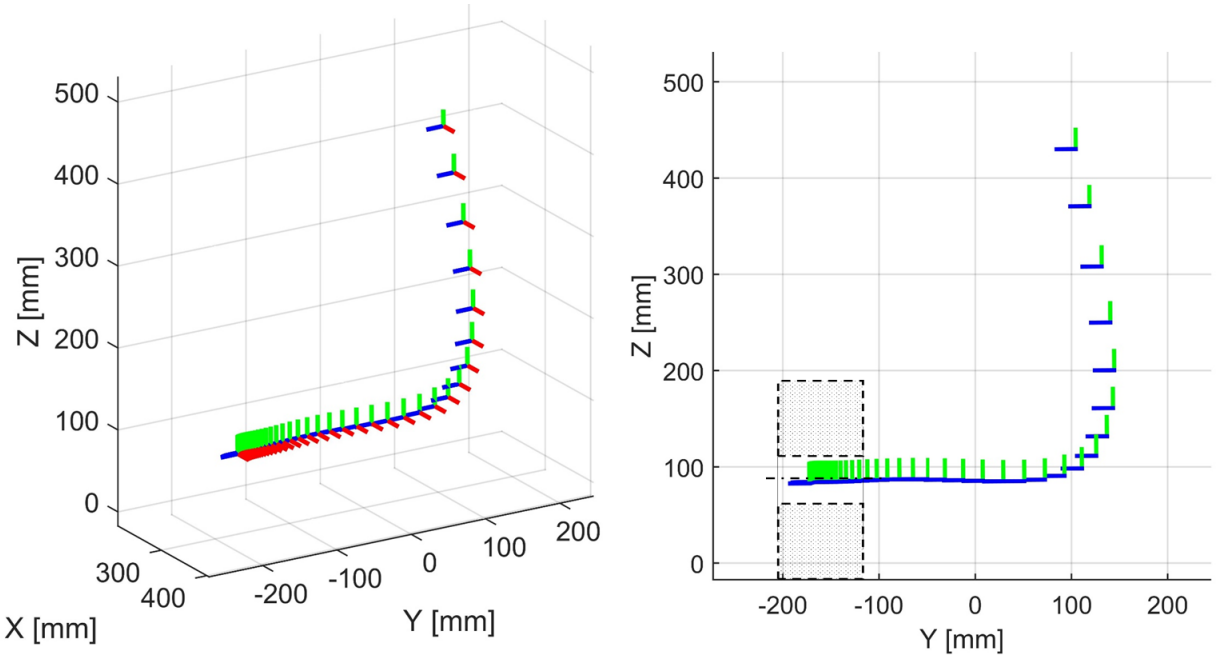
Slika 6-3. Klasificirani privlačni koordinatni sustavi.

6.2.2. Optimizacija zadatka

Na temelju ovih informacija dobivenih iz demonstracija, moguće je za svaku novu konfiguraciju generirati trajektoriju koja prolazi kroz identificirane privlačne točke. Jedan ovakav primjer vidljiv je na slici 6-4.



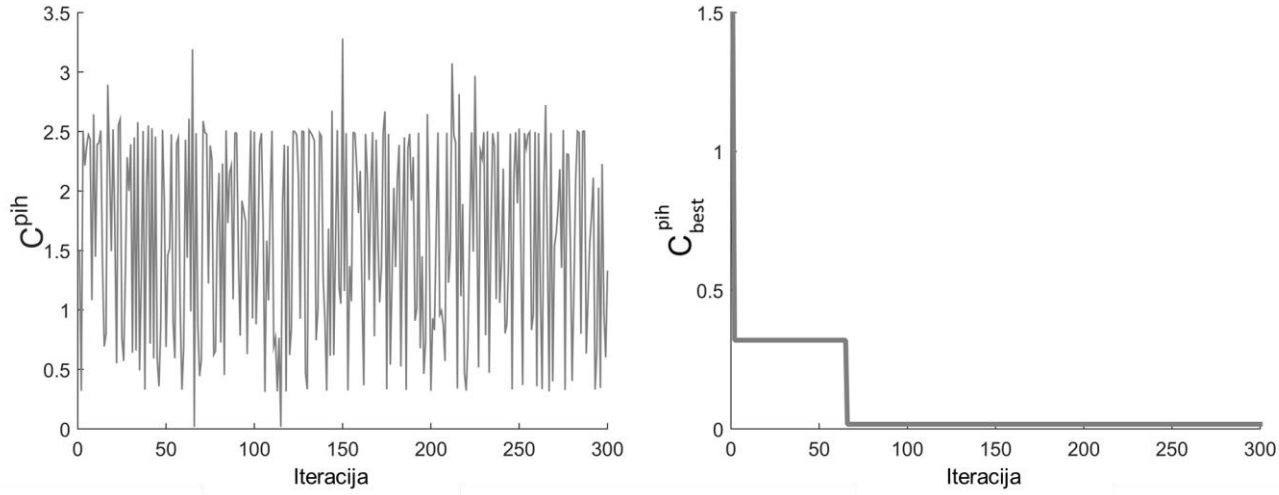
Slika 6-4. Inicijalna trajektorija dobivena aproksimacijom privlačnih koordinatnih sustava.



Slika 6-5.Trajektorija nakon procesa učenja za zadatak umetanja.

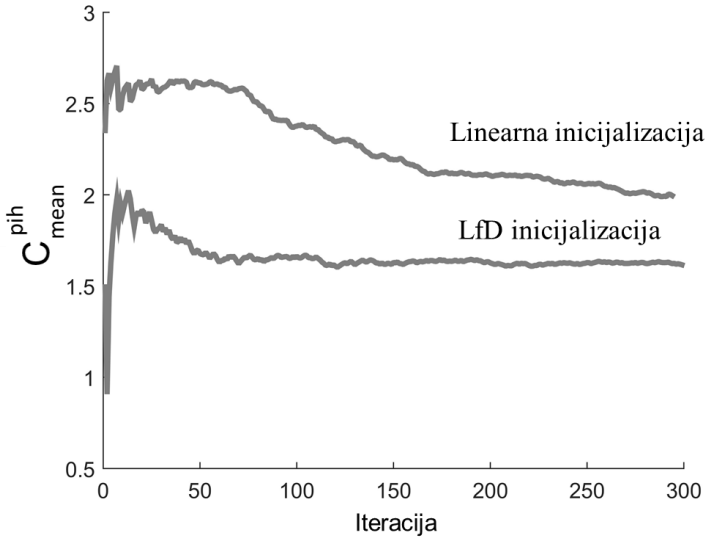
Za ovako inicijaliziranu trajektoriju proces iterativnog podešavanja na temelju zadatka donosi trajektoriju prikazanu na slici 6-5.

Sam proces učenja opet je vrlo stohastičan, ali prema kriteriju najboljeg trenutnog rješenja, zadovoljavajuća trajektorija se pronalazi znatno brže (nakon 60 iteracija) nego kod slučaja sa linearnom inicijalizacijom (125 iteracija).



Slika 6-6. Vrijednosti funkcije cilja dobivene tijekom procesa učenja (lijevo). Najbolje trenutno rješenje dobiveno u procesu učenja (desno).

Usporedba procesa učenja prema kriteriju trenutne srednje vrijednosti funkcije cilja dana je na slici 6-7. Ovdje je vidljivo kako proces učenja inicijaliziran sa trajektorijom proizašlom iz analize demonstracija ima približno 25% nižu vrijednost ovog kriterija. Ovo je jasan pokazatelj da predloženi način inicijalizacije ima značajan utjecaj na kvalitetu procesa učenja.

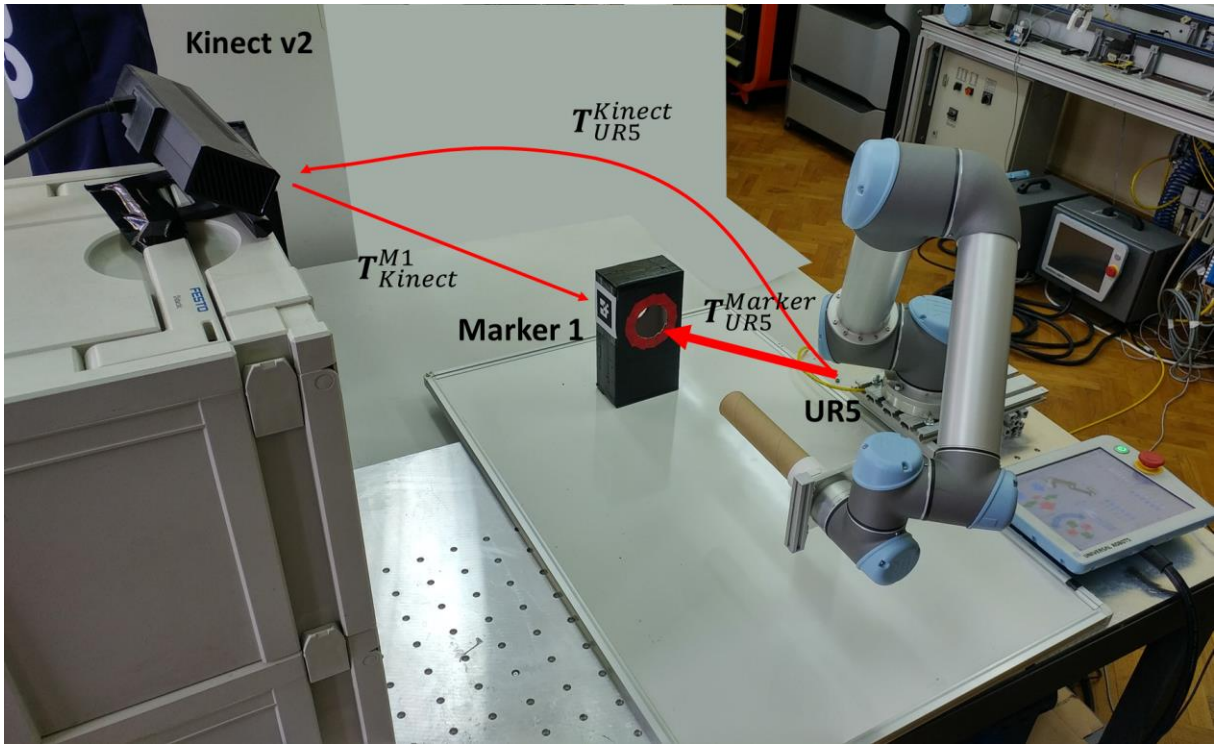


Slika 6-7. Usporedba procesa učenja dvaju načina inicijalizacije prema kriteriju trenutne srednje vrijednosti rješenja.

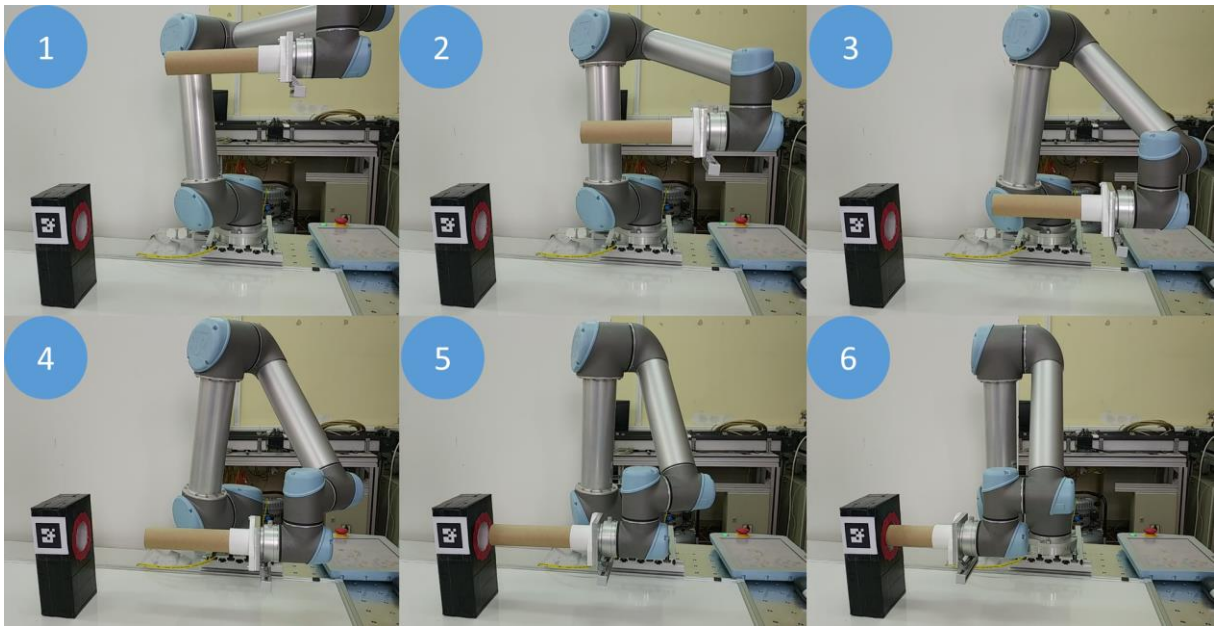
6.2.3. Verifikacija u laboratoriju – zadatak umetanja

Naučene trajektorije iz poglavlja 5 i ovog poglavlja moguće je verificirati na realnoj laboratorijskoj opremi. Pri tome je nužno što vjernije reproducirati uvjete iz simulacije. Za postavljanje predmeta u relativni položaj u odnosu na bazu robota, identičan onome kao u simulaciji, korišten je Kinect vizijski sustav čije su prostorne koordinate kalibracijskim koordinatnim sustavom T_{UR5}^{Kinect} dovedene u odnos sa baznim koordinatnim sustavom robota. Predmet sa rupom za umetanje označen je markerom za čiju je detekciju i određivanje prostornog položaja T_{Kinect}^{M1} korištena ROS implementacija *ar_track_alvar*, koja je ranije bila korištena i u poglavlju 2.6.4. Slanje naučene DMP trajektorije za direktno izvršavanje na robotu vršeno je mehanizmom već opisanim u poglavlju 3.4.4, pri čemu je robot korišten u identičnoj konfiguraciji kao u simulaciji. Cjelokupni laboratorijski postav za zadatak umetanja prikazan je na slici 6-8.

Na slici 6-9 prikazana je sekvenca izvođenja naučene trajektorije za slučaj iz ovog poglavlja.



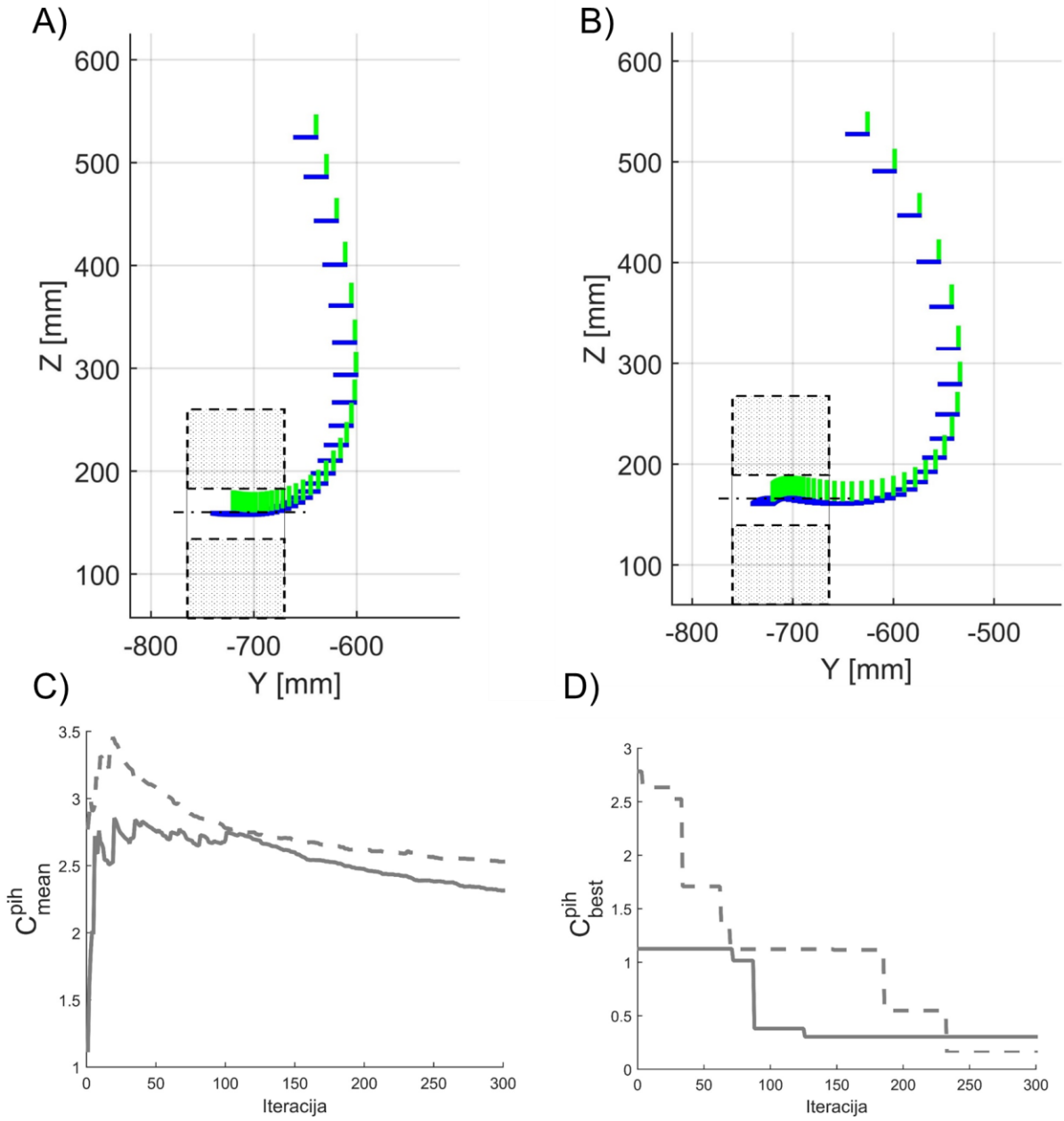
Slika 6-8. Laboratorijski postav za zadatak umetanja.



Slika 6-9. Izvođenje naučene trajektorije na realnom robotu.

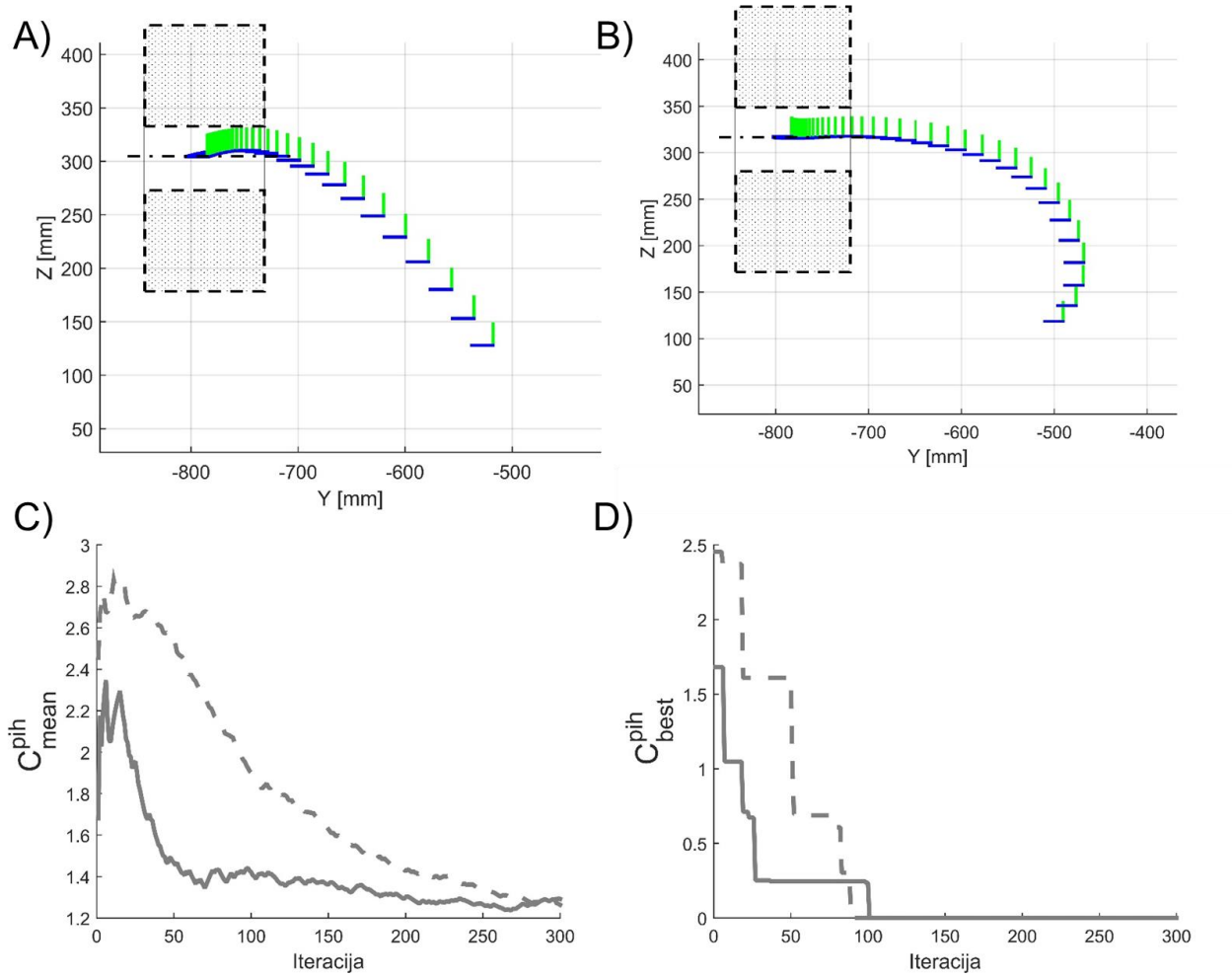
Izvršavanje trajektorija za konfiguracije zadatka iz poglavlja 6.2.4 i 6.2.5 na realnom laboratorijskom postavu prikazano je u video prilogu disertacije.

6.2.4. Konfiguracija 2



Slika 6-10. Rezultati dobiveni za drugu konfiguraciju zadatka umetanja. A) inicijalna trajektorija dobivena aproksimacijom privlačnih koordinatnih sustava, B) naučena trajektorija dobivena iterativnim učenjem zadatka, C) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, usporedba sa procesom koji je inicijaliziran linearnom strategijom. D) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja, usporedba sa procesom koji je inicijaliziran linearnom strategijom.

6.2.5. Konfiguracija 3

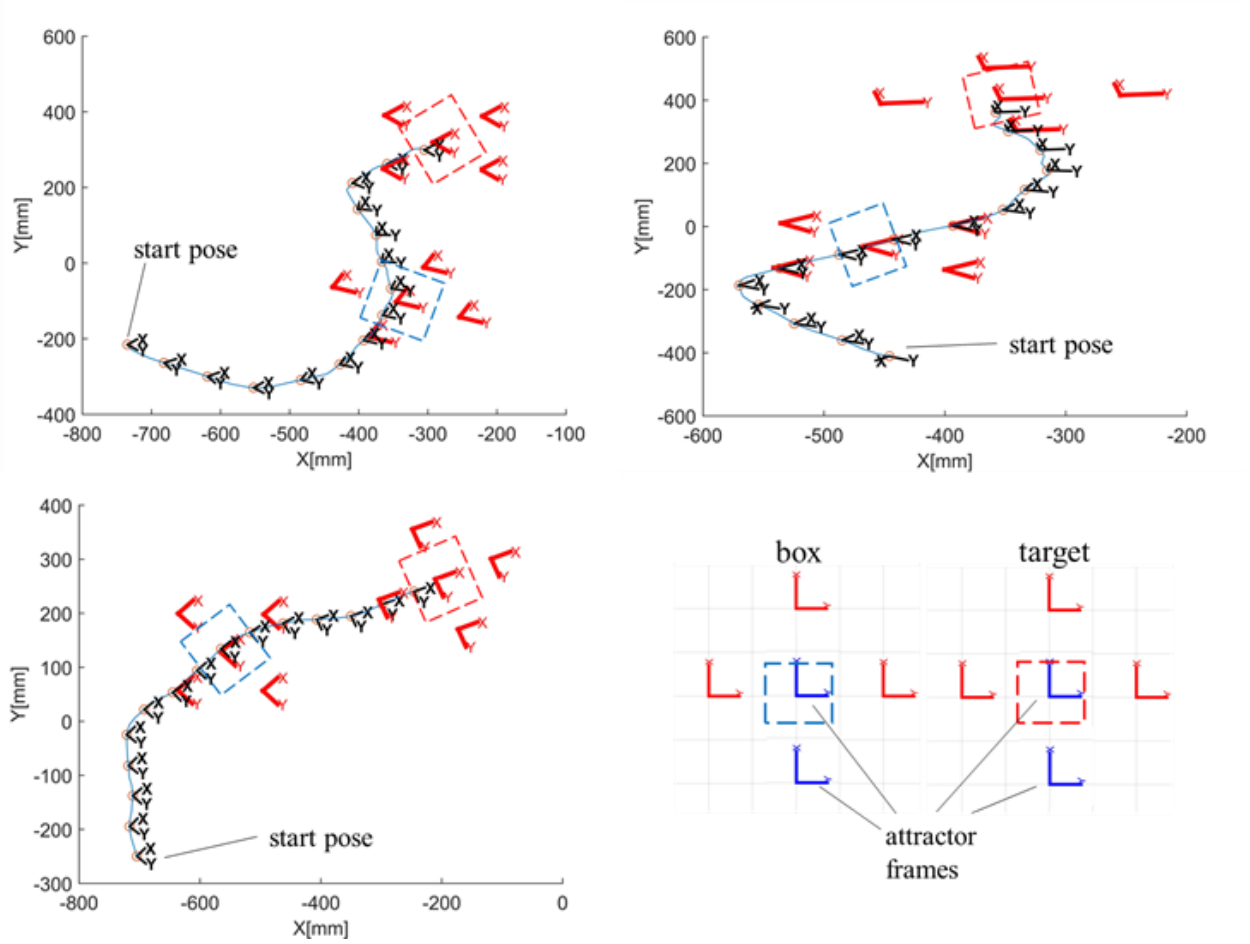


Slika 6-11. Rezultati dobiveni za treću konfiguraciju zadatka umetanja. A) inicijalna trajektorija dobivena aproksimacijom privlačnih koordinatnih sustava, B) naučena trajektorija dobivena iterativnim učenjem zadatka, C) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, usporedba sa procesom koji je inicijaliziran linearnom strategijom. D) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja, , usporedba sa procesom koji je inicijaliziran linearnom strategijom.

6.3. Zadatak odguravanja

6.3.1. Demonstracije

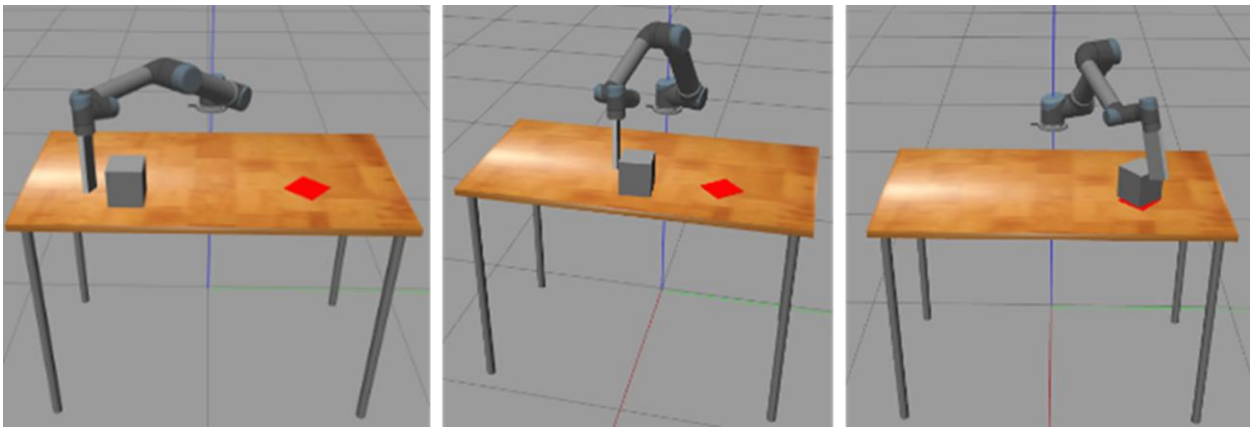
Zadatak odguravanja demonstriran je za tri različite početne konfiguracije (slika 6-11.). Jednu konfiguraciju čini početni prostorni položaj kutije, položaj ciljne pozicije kutije, te početni položaj robota. Koordinatni sustavi od interesa dodijeljeni su početnom položaju kutije (4) i ciljnom položaju kutije (4). Analizom aktivnosti ovih koordinatnih sustava, klasificirana su po dva privlačna koordinatna sustava na svakoj spomenutoj poziciji. To su pozicije koje se odnose na prilazne točke kao i geometrijske centre pojedinih pozicija.

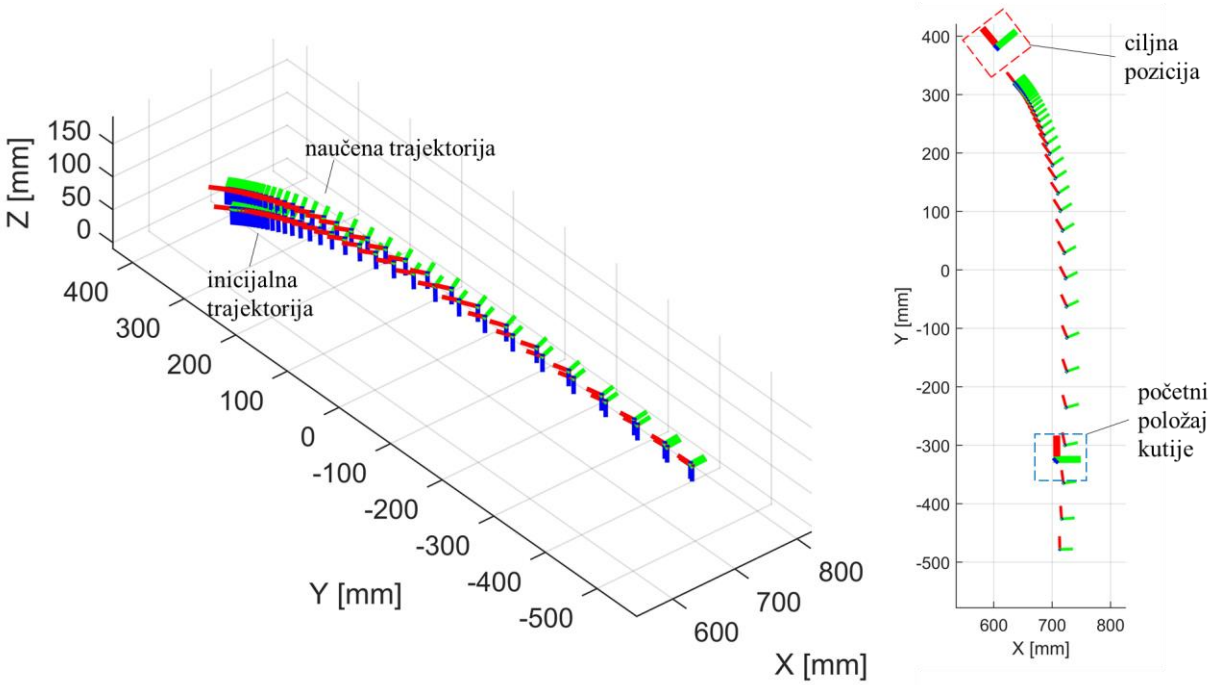


Slika 6-12. Demonstracije zadatka odguravanja, koordinatni sustavi od interesa te klasificirani aktivni koordinatni sustavi.

6.3.2. Optimizacija zadatka

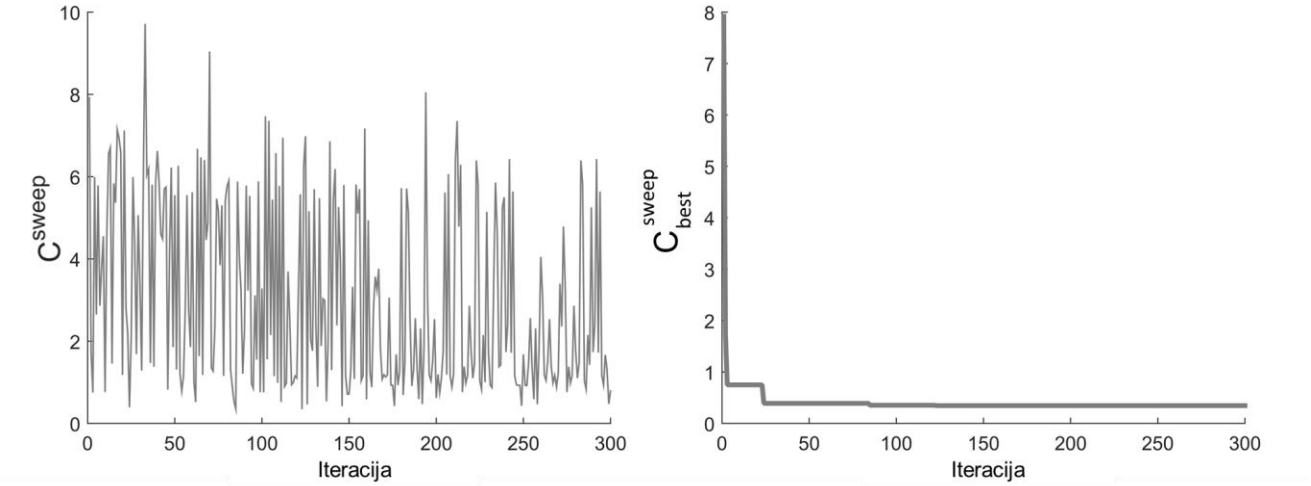
Za novu konfiguraciju zadatka, ovdje će se prikazati novo dobiveno rješenje. Vidljivo je da je za ovu konfiguraciju inicijalno generirana trajektorija vrlo slična konačnoj dobivenoj trajektoriji nakon procesa učenja (slika 6-12.). Ovo je primjer dobro inicijalizirane trajektorije, koja je znatno bliža pravom rješenju negoli predložena linearna trajektorija.





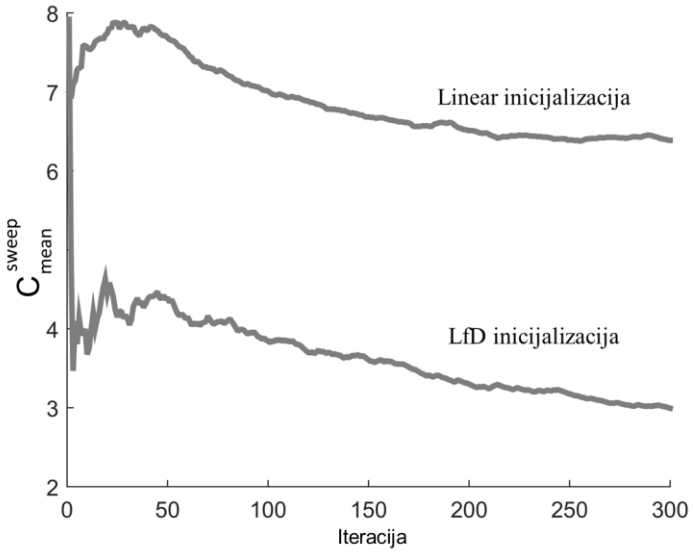
Slika 6-13. Inicijalna trajektorija dobivena aproksimacijom privlačnih koordinatnih sustava i naučena trajektorija za zadatak odguravanja (lijevo). Naučena trajektorija u odnosu na početni i ciljni položaj kutije (desno).

Sami proces učenja je također po oba predložena parametra kvalitetniji. Najbolje rješenje se postiže nakon svega 30 iteracija, dok je za istu konfiguraciju sa linearnom inicijalizacijom bilo potrebno 130.



Slika 6-14. Izvorne vrijednosti funkcije cilja u procesu učenja (lijevo) i transformirana funkcija cilja po kriteriju trenutne najbolje vrijednosti.

Usporedba po kriteriju trenutne srednje vrijednosti također ide u prilog novo predloženom pristupu, te daje u prosjeku približno 45% niže vrijednosti funkcije cilja (slika 6-14.).

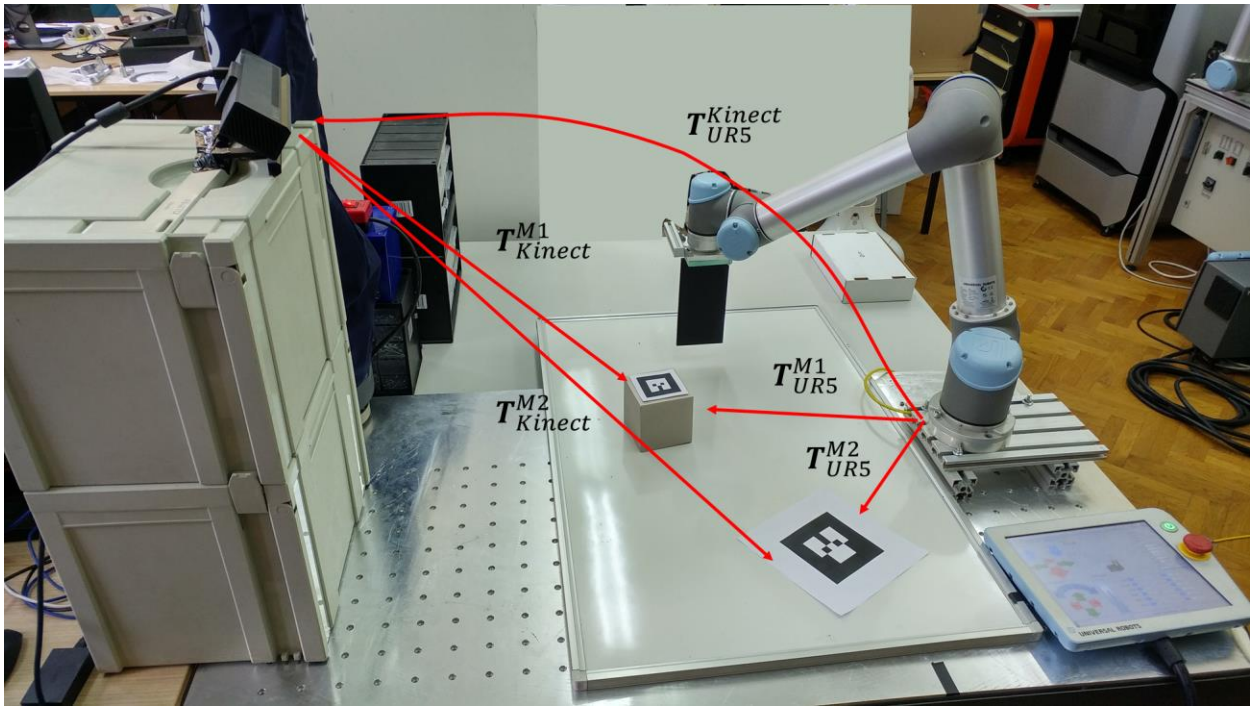


Slika 6-15. Transformirana funkcija cilja prema kriteriju trenutne srednje vrijednosti za slučaj sa linearnom inicijalnom trajektorijom i za slučaj sa trajektorijom generiranom pomoću učenja iz demonstracija.

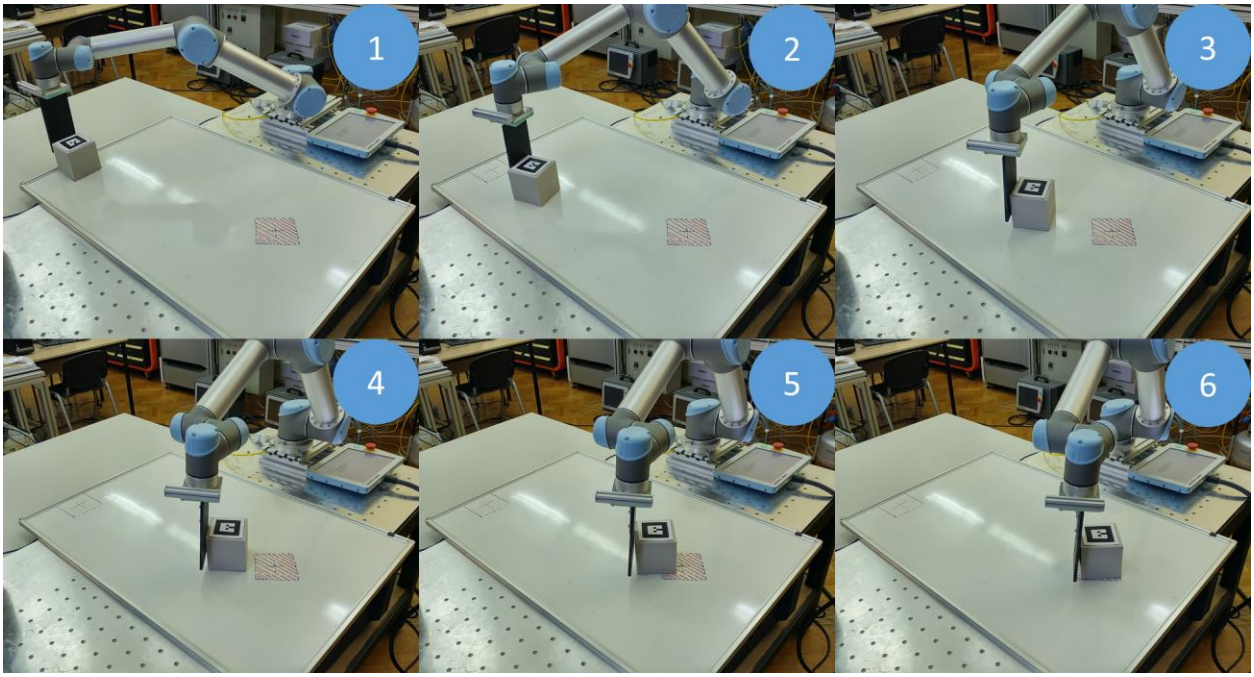
6.3.3. Verifikacija u laboratoriju – odguravanje predmeta

Dok uspješno izvršavanje zadataka umetanja zbog svoje naravi ne ovisi o fizikalnim pojavama, zadatak odguravanja značajno ovisi o dvije. Prva je trenje koje se javlja između predmeta koji se odguruje i podloge, a druga je trenje između lopatice robota i predmeta. Kako je nemoguće u simulaciji replicirati uvjete iz realnog okruženja, ovdje dolazi do mogućeg rascjepa između simulacijskih rezultata i rezultata u realnim uvjetima kod primjene iste trajektorije. Iz tog razloga moguće je kod verifikacije naučenih trajektorija za ovaj zadatak očekivati nešto drugačije rezultate.

Osnova sustava za verifikaciju zadatka odguravanja identična je onoj za zadatak umetanja i sastoji se od Kinect vizijskog sustava i UR5 robota koji se koriste na već opisane načine. Robot je opremljen alatom u obliku ravne plohe koja omogućuje odguravanje predmeta. Predmet odguravanja označen je markerom M1, dok je cilj odguravanja označen markerom M2, kako bi se njihov položaj putem vizijskog sustava mogao točno odrediti u odnosu na bazu robota T_{UR5}^{M1}, T_{UR5}^{M2}. Vizijski sustav korišten je isključivo za postavljanje početnih pozicija predmeta i cilja, dok se isti nije koristio za praćenje putanje predmeta prilikom izvršavanja zadatka.



Slika 6-16. Laboratorijski postav za zadatak odguravanja.

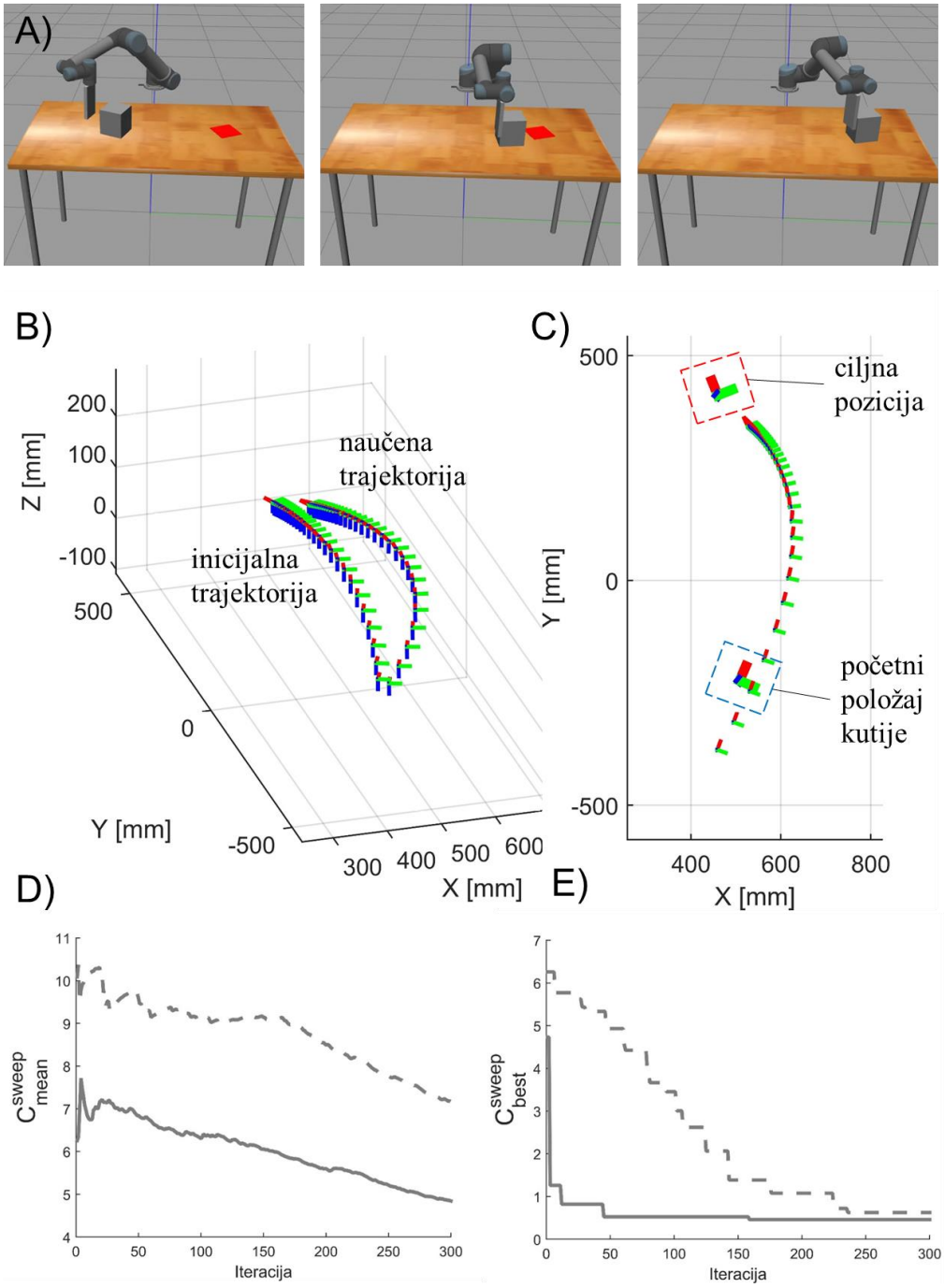


Slika 6-17. Izvođenje naučene trajektorije na realnom robotu. Ciljna pozicija označena je na horizontalnoj plohi crvenom oznakom.

Kod izvršavanja naučene trajektorije korištenjem prikazanog laboratorijskog postava, za konfiguraciju zadatka za naučenu trajektoriju iz poglavlja 6.3.2. prikazanog na slici 6-17., vidljivo je da robot kao i u simulaciji uspješno obavlja zadani zadatak, bez obzira na spomenutu moguću razliku u fizikalnim parametrima okruženja.

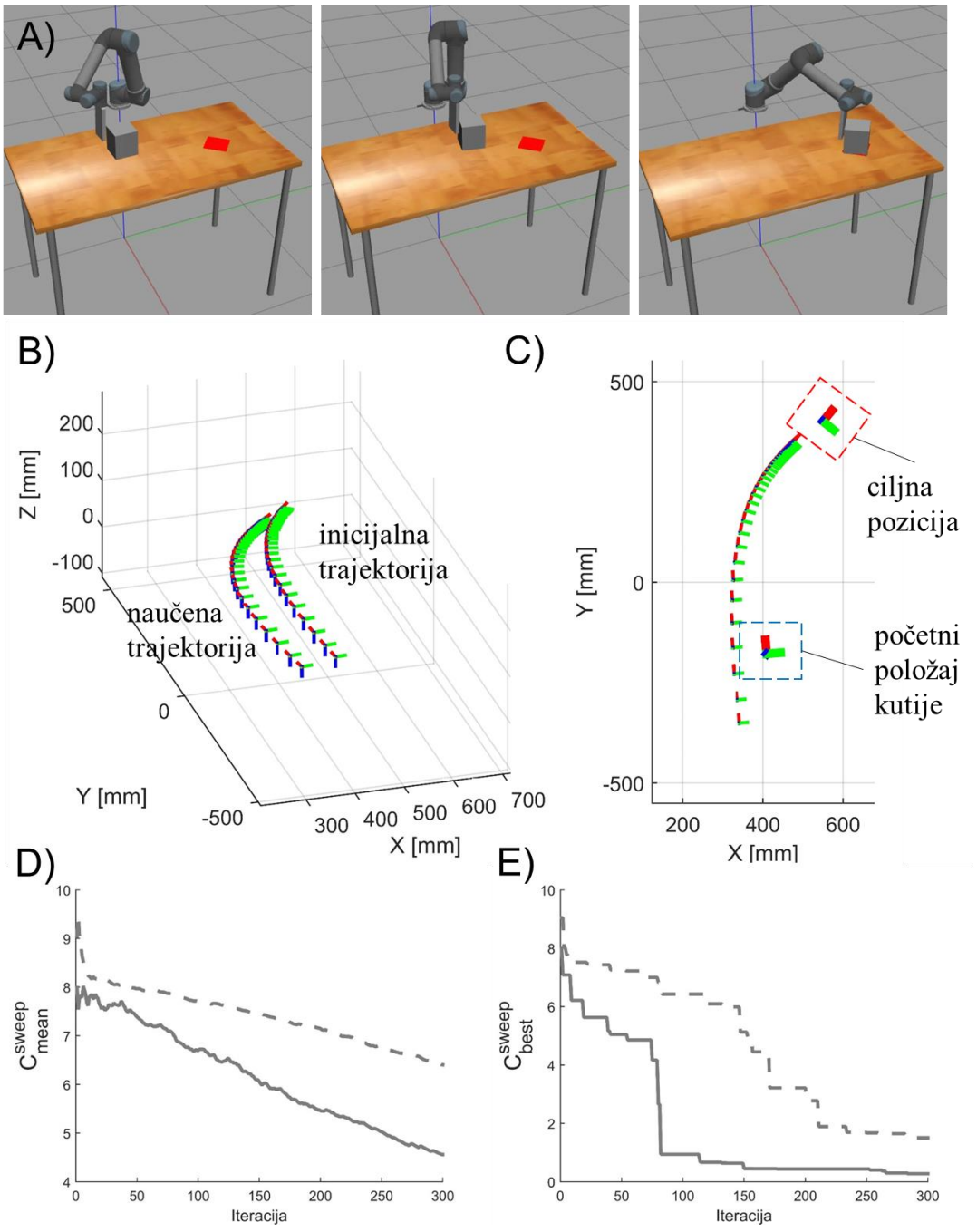
Izvršavanje trajektorija za konfiguracije zadatka iz poglavlja 6.3.4 i 6.3.5 na realnom laboratorijskom postavu prikazano je u video prilogu disertacije.

6.3.4. Konfiguracija 2



Slika 6-18. Rezultati dobiveni za drugu konfiguraciju zadatka odguravanja. A) sekvenca izvršavanja trajektorije, B) usporedba inicijalne i trajektorije dobivene iterativnim učenjem, C) prikaz naučene trajektorije u horizontalnoj ravnini, D) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, E) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja.

6.3.5. Konfiguracija 3



Slika 6-19. Rezultati dobiveni za drugu konfiguraciju zadatka odguravanja. A) sekvenca izvršavanja trajektorije, B) usporedba inicijalne i trajektorije dobivene iterativnim učenjem, C) prikaz naučene trajektorije u horizontalnoj ravnini, D) proces iterativnog učenja prikazan kroz kriterij srednje vrijednosti trenutnih rješenja, E) proces iterativnog učenja prikazan kroz kriterij najboljeg trenutnog rješenja.

7. Zaključak

U radu je predstavljeno istraživanje koje se bavi problematikom učenja robota orijentiranog zadatku. Prvi dio rada (poglavlja dva i prvi dio trećeg poglavlja) bavi se analizom demonstracija i modeliranjem trajektorija na osnovu važnosti pojedinih dijelova trajektorije. Ovdje je predložena nova metoda za analizu demonstracija na temelju malog broja demonstracija robotskih trajektorija u kartezijskom prostoru. Metoda se temelji na otežavanju dijelova trajektorija temeljem analize aktivnosti pojedinih dijelova trajektorija u odnosu na demonstrirane trajektorije. Ovime se postiglo tzv. učenje važnosti svakog određenog segmenta trajektorije koje se kasnije kod generiranja trajektorije upotrijebilo za bližu ili dalju aproksimaciju demonstriranih dijelova trajektorije. Kao model trajektorije se ovdje koristila modificirana verzija DMP trajektorije. Rezultat ovog dijela istraživanja je mogućnost generiranja trajektorija za nove situacije koje dobro opisuju bitne dijelove demonstracija, dok manje bitne dijelove ne uzimaju u obzir.

Drugi dio istraživanja (poglavlje tri) proširuje ovu metodologiju na način da je ranije predstavljena metoda za analizu demonstracija korištena za analizu realnog robotskog zadatka. U ovu svrhu implementiran je eksperimentalni postav koji obuhvaća vizijski sustav za praćenje markera i kolaborativnu robotsku ruku. Predložen je novi model trajektorija temeljen na optimizacijskom postupku koji posjeduje mogućnost aproksimacije stanja i izbjegavanja prepreka. Model trajektorije testiran je zajedno sa metodom za analizu demonstracija za generiranje trajektorija u različitim novo postavljenim slučajevima, a između ostalim i za zadatak odguravanja na realnom robotu.

Rezultati istraživanja provedenih u prva dva dijela rada potvrđuju prvu postavljenu hipotezu, koja tvrdi da je učenje robota za obavljanje zadatka u pretpostavljenoj domeni problema moguće ostvariti putem ograničenog broja primjera prezentiranih interakcijom čovjeka i robota.

Kod reprodukcije trajektorija, za realne robotske zadatke, temeljene isključivo na analizi demonstracija uočena je mogućnost nedovoljno točnog izvršavanja zadatka uslijed raznih utjecaja koji nisu modelirani u sklopu učenja iz demonstracija. U trećem dijelu istraživanja (poglavlje pet) je stoga obrađena problematika samostalnog učenja robota s ciljem omogućavanja samostalnog podešavanja u odnosu na specifične karakteristike radne okoline u kojoj se robot nalazi. Razvijen je model učenja koji je temeljen na iterativnom pretraživanju parametara strategije. Kao parametarska strategija za opis trajektorije koja je predstavljena ranije, odabran je DMP model

trajektorije koji je „naučen“ iz dostupne trajektorije. Kako je iterativno učenje robota u realnoj okolini potencijalno vrlo opasno i dugotrajno, implementirano je simulacijsko okruženje sa emulatorom fizike u ROS okruženju, u kojem je robotu omogućeno testiranje trajektorija u procesu učenja na siguran način. Za podešavanje parametara DMP strategije tijekom učenja korišten je evolucijski optimizacijski algoritam CMA-ES, koji parametre strategije (trajektorije) optimizira u odnosu na funkciju cilja. Korišteni oblik funkcije cilja koncipiran je na način da ocjenjuje samo konačni ishod ponašanja robota, čime je sačuvana jednostavnost implementacije, a omogućena je optimizacija robotskog ponašanja s obzirom na realan zadatak. Sustav iterativnog učenja testiran je na dva različita stohastička robotska zadatka: umetanje (zadatak umetanja) i robotsko metenje (zadatak odguravanja). Za evaluaciju stohastičkog procesa učenja predložena su dva kriterija po kojima je moguće ocijeniti kvalitetu samo procesa. Prvi je kriterij srednje vrijednosti trenutno viđenih rješenja, a drugi je kriterij trenutno najboljeg rješenja. Na oba primjera pokazana mogućnost konvergencije ponašanju koje uspješno obavlja zadatak.

Proces pretraživanja strategija kakvo je ranije predstavljeno značajno ovisi o početnim uvjetima od kojih kreće pretraga. S druge strane, učenje iz demonstracija razvijeno u sklopu ovog rada daje metodologiju za generiranje smislenih trajektorija na osnovu malog broja demonstracija zadatka, prikazanih od strane čovjeka. U četvrtom dijelu rada (poglavlje šest) je stoga predložen model učenja koji obuhvaća razvijenu metodu učenja iz demonstracija i generiranja trajektorije te metodu samostalnog iterativnog učenja. Trajektorija dobivena analizom demonstracija služi kao temeljno polazište (inicijalna trajektorija) za proces samostalnog učenja. Ovaj pristup se evaluirao na oba prethodno spomenuta primjera te je dobivena bolja kvaliteta procesa učenja po oba predstavljena kriterija vrednovanja u svim promatranim slučajevima. Naučene trajektorije su se isto tako testirale u izvršavanju zadatka u realnoj okolini. Ovdje se pokazala dobra podudarnost simulacijskih rezultata sa izvršavanjem u realnoj okolini. Ovo ukazuje na određenu robusnost kod prevođenja trajektorije iz simulacijskog u realno okruženje.

Rezultati dobiveni u trećem i četvrtom dijelu istraživanja sugeriraju da je generalizaciju znanja robota u vidu izvršavanja zadatka za različite konfiguracije, moguće postići statističkom obradom i višekriterijskom optimizacijom specifičnih značajki koje opisuju zadatak. Što je ujedno i potvrda druge hipoteze rada.

Kao izvorni znanstveni doprinosi ovog rada ističu se:

- Metoda za učenje iz demonstracija na razini trajektorija temeljena na klasifikaciji prostornih koordinatnih sustava.
- Model trajektorije za planiranje kretanja u kartezijskom koordinatnom sustavu temeljen na optimizacijskom postupku sa mogućnošću aproksimacije koordinatnih sustava i zaobilaženjem prepreka.
- Metodologija za iterativno učenje trajektorija orijentirano zadatku, temeljena na inicijalizaciji putem učenja iz demonstracija.

Buduća istraživanja biti će fokusirana na pronalaženje efikasnijih algoritama za pretraživanje strategija kao i mogućnosti primjene drugih parametarskih strategija za modeliranje gibanja. Nadalje, značajan napredak jednostavnosti primjene algoritama za pretraživanje moguće je istraživanjem na području automatskog podešavanja parametara učenja kao i automatske ekstrakcije funkcija nagrada/cilja iz malog broja ljudskih demonstracija. Ovo bi značajno povećalo i pojednostavnilo mogućnost primjene ovakvih rješenja u realnim aplikacijama.

Literatura

[1] Passer MW, Smith RE. Psychology: the science of mind and behavior. 4th ed. Boston: McGraw-Hill Higher Education; 2009.

[2] Pomerleau DA. ALVINN: An Autonomous Land Vehicle in a Neural Network. NIPS;p.9; 1998.

[3] Barto A G, Sutton R S, Anderson C W. Neuronlike adaptive elements that can solve difficult learning control problems. IEEE Transactions on Systems, Man, and Cybernetics. vol. SMC-13; pp. 834–846; Sept./Oct. 1983.

[4] LeCun Y. Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation. vol. 1; no. 4; pp. 541-551; Dec. 1989; doi: 10.1162/neco.1989.1.4.541.

[5] Gibilisco S, editor. Concise encyclopedia of robotics. New York: McGraw-Hill; 2003. 365 p.

[6] Bandura A, Ross D, Ross SA. Transmission of aggression through imitation of aggressive models. J Abnorm Soc Psychol. 1961;63(3):575–82.

[7] Bandura A, Ross D, Ross S A. Imitation of film-mediated aggressive models. The Journal of Abnormal and Social Psychology. 66(1); 3–11; 1963; https://doi.org/10.1037/h0048687.

[8] Jones FN, Skinner BF. The Behavior of Organisms: An Experimental Analysis. Am J Psychol. 1939 Oct;52(4):659.

[9] Quigley M, Gerkey B, Conley K, Faust J, Foote T, Leibs J, et al. ROS: an open-source Robot Operating System. ICRA 2009; 2009.

[10] Waibel M, Beetz M, Civera J, D'Andrea R, Elfring J, Gálvez-López D, et al. RoboEarth. IEEE Robot Autom Mag. 2011 Jun;18(2):69–82.

[11] Chrystopher L. Nehaniv, Kerstin Dau. LIKE ME?- MEASURES OF CORRESPONDENCE AND IMITATION. Cybern Syst. 2001 Jan;32(1–2):11–51.

[12] Savarimuthu TR, Buch AG, Schlette C, Wantia N, Robmann J, Martinez D, et al. Teaching a Robot the Semantics of Assembly Tasks. IEEE Trans Syst Man Cybern Syst. 2018 May;48(5):670–92.

[13] Zuyuan Zhu, Huosheng Hu. Robot Learning from Demonstration in Robotic Assembly: A Survey. Robotics. 2018 Apr 16;7(2):17.

[14] Rueckert E, Mundo J, Paraschos A, Peters J, Neumann G. Extracting low-dimensional control variables for movement primitives. In: 2015 IEEE International Conference on

Robotics and Automation (ICRA) [Internet]. Seattle, WA, USA: IEEE; 2015 [cited 2020 Apr 7]. p. 1511–8. Available from: http://ieeexplore.ieee.org/document/7139390/.

[15] Calinon S, Alizadeh T, Caldwell DG. On improving the extrapolation capability of task-parameterized movement models. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems [Internet]. Tokyo: IEEE; 2013 [cited 2019 Jan 18]. p. 610–6. Available from: http://ieeexplore.ieee.org/document/6696414/.

[16] Nakanishi J, Morimoto J, Endo G, Cheng G, Schaal S, Kawato M. Learning from demonstration and adaptation of biped locomotion. Robot Auton Syst. 2004 Jun;47(2–3):79–91.

[17] Mühlig M, Gienger M, Steil JJ. Interactive imitation learning of object movement skills. Auton Robots. 2012 Feb;32(2):97–114.

[18] Ude A, Gams A, Asfour T, Morimoto J. Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. IEEE Trans Robot. 2010 Oct;26(5):800–15.

[19] Lee SH, Suh IH, Calinon S, Johansson R. Learning basis skills by autonomous segmentation of humanoid motion trajectories. In: 2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012) [Internet]. Osaka, Japan: IEEE; 2012 [cited 2020 Apr 7]. p. 112–9. Available from: http://ieeexplore.ieee.org/document/6651507/

[20] Evrard P, Gribovskaya E, Calinon S, Billard A, Kheddar A. Teaching physical collaborative tasks: object-lifting case study with a humanoid. In: 2009 9th IEEE-RAS International Conference on Humanoid Robots [Internet]. Paris, France: IEEE; 2009 [cited 2020 Apr 7]. p. 399–404. Available from: http://ieeexplore.ieee.org/document/5379513/

[21] Maeda GJ, Neumann G, Ewerton M, Lioutikov R, Kroemer O, Peters J. Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks. Auton Robots. 2017 Mar;41(3):593–612.

[22] Pignat E, Calinon S. Learning adaptive dressing assistance from human demonstration. Robot Auton Syst. 2017 Jul;93:61–75.

[23] Bruno D, Calinon S, Caldwell DG. Learning autonomous behaviours for the body of a flexible surgical robot. Auton Robots. 2017 Feb;41(2):333–47.

[24] Krishnan S, Garg A, Patil S, Lea C, Hager G, Abbeel P, et al. Unsupervised Surgical Task Segmentation with Milestone Learning. :16.

[25] Silver D, Schrittwieser J, Simonyan K, Antonoglou I, Huang A, Guez A, et al. Mastering the game of Go without human knowledge. Nature. 2017 Oct;550(7676):354–9.

[26] Levine S, Pastor P, Krizhevsky A, Quillen D. Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. ArXiv160302199 Cs [Internet]. 2016 Aug 28 [cited 2020 Apr 6]; Available from: http://arxiv.org/abs/1603.02199

[27] OpenAI, Andrychowicz M, Baker B, Chociej M, Jozefowicz R, McGrew B, et al. Learning Dexterous In-Hand Manipulation. ArXiv180800177 Cs Stat [Internet]. 2019 Jan 18 [cited 2020 Apr 6]; Available from: http://arxiv.org/abs/1808.00177.

[28] Cirillo A, Ficuciello F, Natale C, Pirozzi S, Villani L. A Conformable Force/Tactile Skin for Physical Human–Robot Interaction. IEEE Robot Autom Lett. 2016 Jan;1(1):41–8.

[29] Schreiber G, Stemmer A, Bischoff R. The Fast Research Interface for the KUKA Lightweight Robot. IEEE Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications How to Modify and Enhance Commercial Controllers, ICRA, 2010.

[30] Colome A, Pardo D, Alenya G, Torras C. External force estimation during compliant robot manipulation. In: 2013 IEEE International Conference on Robotics and Automation [Internet]. Karlsruhe, Germany: IEEE; 2013 [cited 2020 Apr 8]. p. 3535–40. Available from: http://ieeexplore.ieee.org/document/6631072/.

[31] UR5. Accessed on: May. 29, 2020. [Online]. Available: https://www.universal-robots.com/products/ur5-robot/.

[32] Kuka LBR iiwa. Accessed on: May. 29, 2020. [Online]. Available: https://www.kuka.com/en-de/products/robot-systems/industrial-robots/lbr-iiwa.

[33] Cakmak M, DePalma N, Arriaga RI, Thomaz AL. Exploiting social partners in robot learning. Auton Robots. 2010 Nov;29(3–4):309–29.

[34] Breazeal C, Kidd CD, Thomaz AL, Hoffman G, Berlin M. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In: Intelligent Robots and Systems, 2005(IROS 2005) 2005 IEEE/RSJ International Conference on. IEEE; 2005. p. 708–713.

[35] Cakmak M, Thomaz AL. Designing robot learners that ask good questions. In: Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction - HRI '12 [Internet]. Boston, Massachusetts, USA: ACM Press; 2012 [cited 2020 Apr 7]. p. 17. Available from: http://dl.acm.org/citation.cfm?doid=2157689.2157693

[36] Marić F, Limoyo O, Petrović L, Petrović I, Kelly J. Manipulability Maximization Using Continuous-Time Gaussian Processes. ArXiv180309493 Cs [Internet]. 2018 Sep 11 [cited 2020 Apr 6]; Available from: http://arxiv.org/abs/1803.09493

[37] Wigstrom O, Lennartson B, Vergnano A, Breitholtz C. High-Level Scheduling of Energy Optimal Trajectories. IEEE Trans Autom Sci Eng. 2013 Jan;10(1):57–64.

[38] Ratiu M, Adriana Prichici M. Industrial robot trajectory optimization- a review. Grebenisan G, editor. MATEC Web Conf. 2017;126:02005.

[39] Rana MA, Mukadam M, Ahmadzadeh SR, Chernova S, Boots B. Skill Generalization via Inference-based Planning. In Proc. Robotics: Science and Systems (RSS), Workshop on Mathematical Models, Algorithms, and Human-Robot Interaction, Boston, MA, USA, pp. 1--3, 12th-16th Jul. 2017.

[40] Rana MA, Mukadam M, Ahmadzadeh SR, Chernova S, Boots B. Towards Robust Skill Generalization: Unifying Learning from Demonstration and Motion Planning. In Proc. Robotics: Science and Systems (RSS), Workshop on (Empirically) Data-Driven Manipulation, Boston, MA, USA, pp. 1--3, 12th-16th Jul. 2017.

[41] Dong J, Mukadam M, Dellaert F, Boots B. Motion Planning as Probabilistic Inference using Gaussian Processes and Factor Graphs. In: Robotics: Science and Systems XII [Internet]. Robotics: Science and Systems Foundation; 2016 [cited 2020 Apr 6]. Available from: http://www.roboticsproceedings.org/rss12/p01.pdf.

[42] Ostanin M, Popov D, Klimchik A. Programming by Demonstration Using Two-Step Optimization for Industrial Robot. IFAC-Pap. 2018;51(11):72–7.

[43] Mülling K, Kober J, Kroemer O, Peters J. Learning to select and generalize striking movements in robot table tennis. Int J Robot Res. 2013 Mar;32(3):263–79.

[44] Vecerík M, Hester T, Scholz J, Wang F, Pietquin O, Piot B, et al. Leveraging Demonstrations for Deep Reinforcement Learning on Robotics Problems with Sparse Rewards. ArXiv abs/1707.08817, 2017.

[45] Nair A, McGrew B, Andrychowicz M, Zaremba W, Abbeel P. Overcoming Exploration in Reinforcement Learning with Demonstrations. IEEE International Conference On Robotics And Automation (ICRA), 2018.

[46] Brys T, Harutyunyan A, Suay HB, Chernova S, Taylor ME, Nowé A. Reinforcement Learning from Demonstration through Shaping. In: IJCAI. 2015. p. 3352–3358.

[47] Nehaniv CL, Dautenhahn K, editors. Imitation and Social Learning in Robots, Humans and Animals: Behavioural, Social and Communicative Dimensions [Internet]. Cambridge: Cambridge University Press; 2007 [cited 2020 Apr 29]. Available from: http://ebooks.cambridge.org/ref/id/CBO9780511489808.

[48] Kormushev P, Calinon S, Caldwell D. Reinforcement Learning in Robotics: Applications and Real-World Challenges. Robotics. 2013 Jul 5;2(3):122–48.

[49] Akgun B, Cakmak M, Jiang K, Thomaz AL. Keyframe-based Learning from Demonstration: Method and Evaluation. Int J Soc Robot. 2012 Nov;4(4):343–55.

[50] Burgner J, Rucker DC, Gilbert HB, Swaney PJ, Russell PT, Weaver KD, et al. A Telerobotic System for Transnasal Surgery. IEEEASME Trans Mechatron. 2014 Jun;19(3):996–1006.

[51] Vidaković J. Inteligentno vođenje robota pomoću stereo-vizijskog sustava [diplomski rad]. Zagreb: Fakultet strojarstva i brodogradnje; 2014.

[52] Kulić D, Ott C, Lee D, Ishikawa J, Nakamura Y. Incremental learning of full body motion primitives and their sequencing through human motion observation. Int J Robot Res. 2012 Mar;31(3):330–45.

[53] Figueroa N, Pais Ureche AL, Billard A. Learning complex sequential tasks from demonstration: A pizza dough rolling case study. In: The Eleventh ACM/IEEE International Conference on Human Robot Interaction. IEEE Press; 2016. p. 611–612.

[54] Chernova S, Veloso M. Confidence-based policy learning from demonstration using gaussian mixture models. In: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems. ACM; 2007. p. 233.

[55] Ekvall S, Kragic D. Learning Task Models from Multiple Human Demonstrations. In: ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication [Internet]. Univ. of Hertfordshire, Hatfield, UK: IEEE; 2006 [cited 2020 Jan 4]. p. 358–63. Available from: http://ieeexplore.ieee.org/document/4107834/

[56] Ahmadzadeh SR, Paikan A, Mastrogiovanni F, Natale L, Kormushev P, Caldwell DG. Learning symbolic representations of actions from human demonstrations. In: 2015 IEEE International Conference on Robotics and Automation (ICRA) [Internet]. Seattle, WA, USA: IEEE; 2015 [cited 2020 Mar 4]. p. 3801–8. Available from: http://ieeexplore.ieee.org/document/7139728/.

[57] Cubek R, Ertel W, Palm G. High-level learning from demonstration with conceptual spaces and subspace clustering. In IEEE; 2015 [cited 2018 Apr 10]. p. 2592–7. Available from: http://ieeexplore.ieee.org/document/7139548/.

[58] Atkeson CG, Moore AW, Schaal S. Locally Weighted Learning. Artificial Intelligence Review 11, 11–73 (1997). https://doi.org/10.1023/A:1006559212014.

[59] Cleveland WS, Loader C. Smoothing by Local Regression: Principles and Methods. In: Härdle W, Schimek MG, editors. Statistical Theory and Computational Aspects of Smoothing [Internet]. Heidelberg: Physica-Verlag HD; 1996 [cited 2020 Jan 2]. p. 10–49. Available from: http://link.springer.com/10.1007/978-3-642-48425-4_2

[60] Cohn DA, Ghahramani Z, Jordan MI. Active Learning with Statistical Models. J. Artif. Int. Res. 4, 1; 129–145; 1996.

[61] Rasmussen CE, Williams CKI. Gaussian processes for machine learning. Cambridge, Mass: MIT Press; 2006. 248 p. (Adaptive computation and machine learning).

[62] Duy Nguyen-Tuong, Peters J. Local Gaussian process regression for real-time model-based robot control. In: 2008 IEEE/RSJ International Conference on Intelligent Robots and

Systems [Internet]. Nice: IEEE; 2008 [cited 2020 Jan 3]. p. 380–5. Available from: http://ieeexplore.ieee.org/document/4650850/.

[63] Mukadam M, Yan X, Boots B. Gaussian Process Motion planning. In: 2016 IEEE International Conference on Robotics and Automation (ICRA) [Internet]. Stockholm, Sweden: IEEE; 2016 [cited 2020 Jan 3]. p. 9–15. Available from: http://ieeexplore.ieee.org/document/7487091/.

[64] Dillmann R, Kaiser M, Ude A. Acquisition of Elementary Robot Skills from Human Demonstration? International symposium on intelligent robotics systems. Citeseer; pp. 185–192; 1995.

[65] Schwarz G. Estimating the Dimension of a Model. Ann Stat. 1978 Mar;6(2):461–4.

[66] Muhlig M, Gienger M, Hellbach S, Steil JJ, Goerick C. Task-level imitation learning using variance-based movement optimization. In: 2009 IEEE International Conference on Robotics and Automation [Internet]. Kobe: IEEE; 2009 [cited 2020 Jan 4]. p. 1177–84. Available from: http://ieeexplore.ieee.org/document/5152439/.

[67] Calinon S, Guenter F, Billard A. On Learning, Representing, and Generalizing a Task in a Humanoid Robot. IEEE Trans Syst Man Cybern Part B Cybern. 2007 Apr;37(2):286–98.

[68] Chiu C-Y, Chao S-P, Wu M-Y, Yang S-N, Lin H-C. Content-based retrieval for human motion data. J Vis Commun Image Represent. 2004 Sep;15(3):446–66.

[69] Calinon S, Billard A. Statistical Learning by Imitation of Competing Constraints in Joint Space and Task Space. Adv Robot. 2009 Jan;23(15):2059–76.

[70] Ijspeert AJ, Nakanishi J, Schaal S. Trajectory formation for imitation with nonlinear dynamical systems. In: Intelligent Robots and Systems, 2001 Proceedings 2001 IEEE/RSJ International Conference on. IEEE; 2001. p. 752–757.

[71] Ijspeert AJ, Nakanishi J, Hoffmann H, Pastor P, Schaal S. Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors. Neural Comput. 2013 Feb;25(2):328–73.

[72] Calinon S. A tutorial on task-parameterized movement learning and retrieval. Intell Serv Robot. 2016 Jan;9(1):1–29.

[73] Pervez A, Lee D. Learning task-parameterized dynamic movement primitives using mixture of GMMs. Intell Serv Robot. 2018 Jan;11(1):61–78.

[74] Stulp F, Raiola G, Hoarau A, Ivaldi S, Sigaud O. Learning compact parameterized skills with a single regression. In: 2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids) [Internet]. Atlanta, GA: IEEE; 2013 [cited 2019 Jan 8]. p. 417–22. Available from: http://ieeexplore.ieee.org/document/7030008/.

[75] Ureche ALP, Umezawa K, Nakamura Y, Billard A. Task Parameterization Using Continuous Constraints Extracted From Human Demonstrations. IEEE Trans Robot. 2015 Dec;31(6):1458–71.

[76] Kalakrishnan M, Pastor P, Righetti L, Schaal S. Learning objective functions for manipulation. In IEEE; 2013 [cited 2018 Apr 12]. p. 1331–6. Available from: http://ieeexplore.ieee.org/document/6630743/.

[77] Piot B, Geist M, Pietquin O. Bridging the Gap Between Imitation Learning and Inverse Reinforcement Learning. IEEE Trans Neural Netw Learn Syst. 2017 Aug;28(8):1814–26.

[78] Abbeel P, Ng AY. Apprenticeship learning via inverse reinforcement learning. In: Twenty-first international conference on Machine learning - ICML '04 [Internet]. Banff, Alberta, Canada: ACM Press; 2004 [cited 2019 Apr 17]. p. 1. Available from: http://portal.acm.org/citation.cfm?doid=1015330.1015430.

[79] Šekoranja B. Vjerojatnosni model robotskoga djelovanja u fizičkoj interakciji s čovjekom [doktorski rad]. Zagreb: Fakultet strojarstva i brodogradnje; 2015.

[80] Siciliano B, editor. Robotics: modelling, planning and control. London: Springer; 2009. 632 p. (Advanced textbooks in control and signal processing).

[81] Hlaváč V. Robot trajectory generation. Accessed on: January. 20, 2020. [Online]. Available: http://people.ciirc.cvut.cz/~hlavac/TeachPresEn/55AutonomRobotics/090RobotTrajectory GenerationEn.pdf.

[82] Zhao R. Trajectory planning and control for robot manipulations. Robotics [cs.RO]. Université Paul Sabatier - Toulouse III, 2015. English. ffNNT: 2015TOU30240ff. fftel-01285383v2f.

[83] Pastor P, Hoffmann H, Asfour T, Schaal S. Learning and generalization of motor skills by learning from demonstration. In: 2009 IEEE International Conference on Robotics and Automation [Internet]. Kobe: IEEE; 2009 [cited 2020 Apr 4]. p. 763–8. Available from: http://ieeexplore.ieee.org/document/5152385/.

[84] Fabisch A. A Comparison of Policy Search in Joint Space and Cartesian Space for Refinement of Skills. ArXiv190406765 Cs. 2020;980:301–9.

[85] Peters J, Schaal S. Policy Gradient Methods for Robotics. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems [Internet]. Beijing, China: IEEE; 2006 [cited 2019 Dec 14]. p. 2219–25. Available from: http://ieeexplore.ieee.org/document/4058714/.

[86] Kormushev P, Calinon S, Caldwell DG. Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input. Adv Robot. 2011 Jan;25(5):581–603.

[87] Englert P, Toussaint M. Combined Optimization and Reinforcement Learning for Manipulation Skills. In: Robotics: Science and Systems XII [Internet]. Robotics: Science and Systems Foundation; 2016 [cited 2019 Apr 19]. Available from: http://www.roboticsproceedings.org/rss12/p33.pdf.

[88] Kober J, Peters J. Policy search for motor primitives in robotics. Mach Learn. 2011 Jul;84(1–2):171–203.

[89] Kormushev P, Calinon S, Caldwell DG. Robot motor skill coordination with EM-based reinforcement learning. In: Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on. IEEE; 2010. p. 3232–3237.

[90] Zuyuan Zhu, Huosheng Hu. Robot Learning from Demonstration in Robotic Assembly: A Survey. Robotics. 2018 Apr 16;7(2):17.

[91] Hewitt A, Yang C, Li Y, Cui R. DMP and GMR based teaching by demonstration for a KUKA LBR robot. In IEEE; 2017 [cited 2018 Apr 6]. p. 1–6. Available from: http://ieeexplore.ieee.org/document/8081982/.

[92] Vidaković J, Jerbić B, Šekoranja B, Švaco M, Šuligoj F. Task Dependent Trajectory Learning from Multiple Demonstrations Using Movement Primitives. In: Berns K, Görges D, editors. Advances in Service and Industrial Robotics [Internet]. Cham: Springer International Publishing; 2020 [cited 2019 Dec 12]. p. 275–82. Available from: http://link.springer.com/10.1007/978-3-030-19648-6_32.

[93] Vidaković J, Jerbić B, Šekoranja B, Švaco M, Šuligoj F. Learning from Demonstration Based on a Classification of Task Parameters and Trajectory Optimization". J Intell Robot Syst, (2019). https://doi.org/10.1007/s10846-019-01101-2.

[94] Hansen N, Ostermeier A. Completely Derandomized Self-Adaptation in Evolution Strategies. Evol Comput. 2001 Jun;9(2):159–95.

[95] The URScript Programming Language. Accessed on: May. 18, 2019. [Online]. Available: https://www.universal-robots.com/.

[96] Švaco M, Jerbić B, Polančec M, Šuligoj F. A Reinforcement Learning Based Algorithm for Robot Action Planning. Advances in Service and Industrial Robotics. RAAD 2018. Mechanisms and Machine Science, vol 67. Cham: Springer, 2018. str. 493-503 doi:10.1007/978-3-030-00232-9_52.

[97] Chebotar Y, Hausman K, Zhang M, Sukhatme G, Schaal S, Levine S. Combining Model-Based and Model-Free Updates for Trajectory-Centric Reinforcement Learning. ArXiv170303078 Cs [Internet]. 2017 Jun 18 [cited 2019 Nov 19]; Available from: http://arxiv.org/abs/1703.03078.

[98] Vuga R, Nemec B, Ude A. Enhanced Policy Adaptation Through Directed Explorative Learning. Int J Humanoid Robot. 2015 Sep;12(03):1550028.

[99] Levine S, Wagener N, Abbeel P. Learning contact-rich manipulation skills with guided policy search. In: 2015 IEEE International Conference on Robotics and Automation (ICRA) [Internet]. Seattle, WA, USA: IEEE; 2015 [cited 2019 Apr 19]. p. 156–63. Available from: http://ieeexplore.ieee.org/document/7138994/

[100] Theodorou E, Buchli J, Schaal S. Learning Policy Improvements with Path Integrals. J. Mach. Learn. Res., vol. 9, pp. 828–835, Jan. 2010.

[101] Bellman R. Dynamic programming. Princeton, NJ: Princeton Univ. Pr; 1984. 339 p.

[102] Deisenroth MP. A Survey on Policy Search for Robotics. Found Trends Robot. 2011;2(1–2):1–142.

[103] Sutton RS, McAllester DA, Singh SP, Mansour Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation. NIPS; 1999.

[104] Williams RJ. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn., 8, 229–256, 1992. https://doi.org/10.1007/BF00992696.

[105] Kakade SM. A Natural Policy Gradient. In Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic (NIPS'01). MIT Press, Cambridge, MA, USA, 1531–1538.

[106] Peters J, Schaal S. Natural Actor Critic. Neurocomputing, 71, 7–9, March 2008, p. 1180-1190.

[107] Peters J, Schaal S. Reinforcement learning of motor skills with policy gradients. Neural Netw. 2008 May;21(4):682–97.

[108] Park J, Kim J, Kang D. An RLS-Based Natural Actor-Critic Algorithm for Locomotion of a Two-Linked Robot Arm. In: Hao Y, Liu J, Wang Y, Cheung Y, Yin H, Jiao L, et al., editors. Computational Intelligence and Security [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2005 [cited 2019 Dec 11]. p. 65–72. Available from: http://link.springer.com/10.1007/11596448_9.

[109] Mori T, Nakamura Y, Sato M, Ishii S. Reinforcement Learning for a CPG-driven Biped Robot. In Proceedings of the 19th national conference on Artifical intelligence (AAAI'04). AAAI Press, 623–630.

[110] Dayan P, Hinton GE. Using EM for Reinforcement Learning. Neural Computation, 9, 271–278 (1997).

[111] Peters J, Schaal S. Learning to Control in Operational Space. Int J Robot Res. 2008 Feb;27(2):197–212.

[112] Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, et al. Continuous control with deep reinforcement learning. ArXiv150902971 Cs Stat [Internet]. 2019 Jul 5 [cited 2019 Dec 9]; Available from: http://arxiv.org/abs/1509.02971.

[113] Levine S, Koltun V. Guided Policy Search. Proceedings of the 30th International Conference on Machine Learning, PMLR 28(3):1-9, 2013.

[114] Schulman J, Levine S, Moritz P, Jordan MI, Abbeel P. Trust Region Policy Optimization. ArXiv150205477 Cs [Internet]. 2017 Apr 20 [cited 2019 Dec 9]; Available from: http://arxiv.org/abs/1502.05477.

[115] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal Policy Optimization Algorithms. ArXiv170706347 Cs [Internet]. 2017 Aug 28 [cited 2019 Dec 9]; Available from: http://arxiv.org/abs/1707.06347.

[116] Deisenroth MP, Rasmussen CE. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In: Proc.28th Int.Conf. Mach.Learn. (ICML), 2011, pp. 465–472.

[117] Deisenroth MP, Rasmussen CE, Fox D. Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning. Robotics: Science and Systems VII, UCLA, CA, USA, June 27-30, 2011.

[118] Deisenroth MP, Fox D, Rasmussen CE. Gaussian Processes for Data-Efficient Learning in Robotics and Control. IEEE Trans Pattern Anal Mach Intell. 2015 Feb;37(2):408–23.

[119] Stulp F, Sigaud O. Policy Improvement Methods: Between Black-Box Optimization and Episodic Reinforcement Learning. J. Francophones Planification, Décision, Apprentissage Pour la Conduite Syst., pp. 1–15, Jul. 2013.

[120] Beyer H-G, Schwefel H-P. A comprehensive introduction. Natural Computing 1: 3–52, 2002.

[121] Rückstieß T, Sehnke F, Schaul T, Wierstra D, Sun Y, Schmidhuber J. Exploring Parameter Space in Reinforcement Learning. Paladyn J Behav Robot [Internet]. 2010 Jan 1 [cited 2019 Dec 17];1(1). Available from: http://www.degruyter.com/view/j/pjbr.2010.1.issue-1/s13230-010-0002-4/s13230-010-0002-4.xml.

[122] Hansen N, Müller SD, Koumoutsakos P. Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). Evol Comput. 2003 Mar;11(1):1–18.

[123] Igel C. Neuroevolution for reinforcement learning using evolution strategies. In: The 2003 Congress on Evolutionary Computation, 2003 CEC ’03 [Internet]. Canberra, Australia:

IEEE; 2003 [cited 2019 Dec 17]. p. 2588–95. Available from: http://ieeexplore.ieee.org/document/1299414/.

[124] Sigaud O, Stulp F. Policy Search in Continuous Action Domains: an Overview. ArXiv180304706 Cs [Internet]. 2019 Jun 13 [cited 2019 Dec 17]; Available from: http://arxiv.org/abs/1803.04706.

[125] Gomez F, Schmidhuber J, Miikkulainen R. Efficient Non-linear Control Through Neuroevolution. In: Fürnkranz J, Scheffer T, Spiliopoulou M, editors. Machine Learning: ECML 2006 [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2006 [cited 2019 Dec 17]. p. 654–62. Available from: http://link.springer.com/10.1007/11871842_64

[126] Heidrich-Meisner V, Igel C. Similarities and differences between policy gradient methods and evolution strategies. ESANN 2008, 16th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 23-25, 2008, Proceedings.

[127] Heidrich-Meisner V, Igel C. Variable Metric Reinforcement Learning Methods Applied to the Noisy Mountain Car Problem. In: Girgin S, Loth M, Munos R, Preux P, Ryabko D, editors. Recent Advances in Reinforcement Learning [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2008 [cited 2019 Dec 17]. p. 136–50. Available from: http://link.springer.com/10.1007/978-3-540-89722-4_11

[128] Heidrich-Meisner V, Igel C. Evolution Strategies for Direct Policy Search. In: Rudolph G, Jansen T, Beume N, Lucas S, Poloni C, editors. Parallel Problem Solving from Nature – PPSN X [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2008 [cited 2019 Dec 17]. p. 428–37. Available from: http://link.springer.com/10.1007/978-3-540-87700-4_43.

[129] Ollivier Y, Arnold L, Auger A, Hansen N. Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles. ArXiv11063708 Math [Internet]. 2017 Apr 28 [cited 2019 Dec 17]; Available from: http://arxiv.org/abs/1106.3708.

[130] de Broissia A de F, Sigaud O. Actor-critic versus direct policy search: a comparison based on sample complexity. ArXiv160609152 Cs [Internet]. 2016 Aug 22 [cited 2019 Nov 22]; Available from: http://arxiv.org/abs/1606.09152.

[131] Siebel NT, Sommer G. Evolutionary reinforcement learning of artificial neural networks. Köppen M, Weber R, editors. Int J Hybrid Intell Syst. 2007 Oct 17;4(3):171–83.

[132] Kassahun Y, Sommer G. Efficient Reinforcement Learning Through Evolutionary Acquisition of Neural Topologies. ESANN 2005, 13th European Symposium on Artificial Neural Networks, Bruges, Belgium, April 27-29, 2005, Proceedings p. 9, 2005.

[133] Li X, Liang Z, Feng H. Kicking motion planning of Nao robots based on CMA-ES. In: The 27th Chinese Control and Decision Conference (2015 CCDC) [Internet]. Qingdao, China: IEEE; 2015 [cited 2019 Dec 17]. p. 6158–61. Available from: http://ieeexplore.ieee.org/document/7161918/

[134] Abdolmaleki A, Price B, Lau N, Reis LP, Neumann G. Deriving and improving CMA-ES with information geometric trust regions. In: Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO ’17 [Internet]. Berlin, Germany: ACM Press; 2017 [cited 2019 Dec 17]. p. 657–64. Available from: http://dl.acm.org/citation.cfm?doid=3071178.3071252.

[135] Miyamoto H, Schaal S, Gandolfo F, Gomi H, Koike Y, Osu R, et al. A Kendama Learning Robot Based on Bi-directional Theory. Neural Netw. 1996 Nov;9(8):1281–302.

[136] Kuindersma S, Grupen R, Barto A. Learning dynamic arm motions for postural recovery. In: 2011 11th IEEE-RAS International Conference on Humanoid Robots [Internet]. Bled, Slovenia: IEEE; 2011 [cited 2020 Apr 21]. p. 7–12. Available from: http://ieeexplore.ieee.org/document/6100881/.

[137] Roberts JW, Moret L, Zhang J, Tedrake R. Motor Learning at Intermediate Reynolds Number: Experiments with Policy Gradient on the Flapping Flight of a Rigid Wing. In: Sigaud O, Peters J, editors. From Motor Learning to Interaction Learning in Robots [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2010 [cited 2020 Apr 21]. p. 293–309. (Kacprzyk J, editor. Studies in Computational Intelligence; vol. 264). Available from: http://link.springer.com/10.1007/978-3-642-05181-4_13.

[138] Yarats D, Zhang A, Kostrikov I, Amos B, Pineau J, Fergus R. Improving Sample Efficiency in Model-Free Reinforcement Learning from Images. ArXiv191001741 Cs Stat [Internet]. 2019 Oct 6 [cited 2019 Nov 19]; Available from: http://arxiv.org/abs/1910.01741

[139] Hafner D, Lillicrap T, Fischer I, Villegas R, Ha D, Lee H, et al. Learning Latent Dynamics for Planning from Pixels. ArXiv181104551 Cs Stat [Internet]. 2019 Jun 4 [cited 2019 Nov 19]; Available from: http://arxiv.org/abs/1811.04551.

[140] Lee AX, Nagabandi A, Abbeel P, Levine S. Stochastic Latent Actor-Critic: Deep Reinforcement Learning with a Latent Variable Model. ArXiv190700953 Cs Stat [Internet]. 2019 Jul 1 [cited 2019 Nov 19]; Available from: http://arxiv.org/abs/1907.00953.

[141] Nair A, Pong V, Dalal M, Bahl S, Lin S, Levine S. Visual Reinforcement Learning with Imagined Goals. ArXiv180704742 Cs Stat [Internet]. 2018 Dec 4 [cited 2019 Nov 19]; Available from: http://arxiv.org/abs/1807.04742.

[142] Vidakovic J, Jerbic B, Sekoranja B, Svaco M, Suligoj F. Accelerating robot trajectory learning for stochastic tasks. IEEE Access. pp. 1–1; 2020. doi: 10.1109/ACCESS.2020.2986999.

[143] Koenig N, Howard A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat No04CH37566) [Internet]. Sendai, Japan: IEEE; 2004 [cited 2020 Mar 23]. p. 2149–54. Available from: http://ieeexplore.ieee.org/document/1389727/.

[144] Millan-Romera JA, Acevedo JJ, Castano AR, Perez-Leon H, Capitan C, Ollero A. A UTM simulator based on ROS and Gazebo. In: 2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS) [Internet]. Cranfield, United Kingdom: IEEE; 2019 [cited 2020 Mar 23]. p. 132–41. Available from: https://ieeexplore.ieee.org/document/8999705/.

[145] Meyer J, Sendobry A, Kohlbrecher S, Klingauf U, von Stryk O. Comprehensive Simulation of Quadrotor UAVs Using ROS and Gazebo. In: Noda I, Ando N, Brugali D, Kuffner JJ, editors. Simulation, Modeling, and Programming for Autonomous Robots [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2012 [cited 2020 Mar 23]. p. 400–11. (Hutchison D, Kanade T, Kittler J, Kleinberg JM, Mattern F, Mitchell JC, et al., editors. Lecture Notes in Computer Science; vol. 7628). Available from: http://link.springer.com/10.1007/978-3-642-34327-8_36.

[146] Afanasyev I, Sagitov A, Magid E. ROS-Based SLAM for a Gazebo-Simulated Mobile Robot in Image-Based 3D Model of Indoor Environment. In: Battiato S, Blanc-Talon J, Gallo G, Philips W, Popescu D, Scheunders P, editors. Advanced Concepts for Intelligent Vision Systems [Internet]. Cham: Springer International Publishing; 2015 [cited 2020 Mar 23]. p. 273–83. (Lecture Notes in Computer Science; vol. 9386). Available from: http://link.springer.com/10.1007/978-3-319-25903-1_24.

[147] Takaya K, Asai T, Kroumov V, Smarandache F. Simulation environment for mobile robots testing using ROS and Gazebo. In: 2016 20th International Conference on System Theory, Control and Computing (ICSTCC) [Internet]. Sinaia: IEEE; 2016 [cited 2020 Mar 23]. p. 96–101. Available from: https://ieeexplore.ieee.org/document/7790647/.

[148] Qian W, Xia Z, Xiong J, Gan Y, Guo Y, Weng S, et al. Manipulation task simulation using ROS and Gazebo. In: 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014) [Internet]. Bali, Indonesia: IEEE; 2014 [cited 2020 Mar 23]. p. 2594–8. Available from: http://ieeexplore.ieee.org/document/7090732/.

Životopis

Josip Vidaković rođen je 18.03.1991. godine. Godine 2009. završio je Tehničku školu Ruđera Boškovića nakon čega iste godine upisuje Fakultet strojarstva i brodogradnje. Stekao je titulu prvostupnika inženjera strojarstva i magistra inženjera strojarstva redom 2013. i 2014 godine. Po završetku studija iste godine počinje raditi na projektu Robotska Neuronavigacija – RONNA na Fakultetu strojarstva i brodogradnje u Zagrebu i Katedri za projektiranje izradbenih i montažnih sustava, koji traje do travnja 2016. godine. Krajem 2016. godine na istoj katedri zaposlen je na radno mjesto asistenta na projektu u okviru projekta razvoja karijera mladih istraživača Hrvatske zaklade za znanost. Istovremeno upisuje poslijediplomski doktorski studij Mehatronike i robotike na istom Fakultetu.

Njegovi istraživački interesi obuhvaćaju industrijsku i medicinsku robotiku, programiranje robota iz demonstracija, planiranje kretanja i metode strojnog učenja. Autor ili koautor je sedam radova u časopisima od kojih je četiri indeksirano u CC, a tri u SCI bazama, sedam radova u zbornicima konferencija i jednog poglavlja u knjigama.

Popis javno objavljenih radova

1. Vidaković, Josip; Jerbić, Bojan; Šekoranja, Bojan; Švaco, Marko; Šuligoj, Filip
   Accelerating robot trajectory learning for stochastic tasks. // IEEE access, 1 (2020), 1-14 doi:10.1109/access.2020.2986999 (međunarodna recenzija, članak, znanstveni)

2. Vidaković, Josip; Jerbić, Bojan; Šekoranja, Bojan; Švaco, Marko; Šuligoj, Filip
   Learning from Demonstration Based on a Classification of Task Parameters and Trajectory Optimization. // Journal of intelligent & robotic systems, 96 (2019), 1-15 doi:10.1007/s10846-019-01101-2 (međunarodna recenzija, članak, znanstveni)

3. Vidaković, Josip; Jerbić, Bojan; Šekoranja, Bojan; Švaco, Marko; Šuligoj, Filip
   Task Dependent Trajectory Learning from Multiple Demonstrations Using Movement Primitives. // 28th International Conference on Robotics in Alpe- Adria-Danube Region / Springer, Cham (ur.).
   Cham, Njemačka: Springer International Publishing, 2019. str. 275-282 doi:10.1007/978-3-030-19648-6_32 (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

4. Jerbić, Bojan; Švaco, Marko; Chudy, Darko; Šekoranja, Bojan; Šuligoj, Filip; Vidaković, Josip; Dlaka, Domagoj; Vitez, Nikola; Župančić, Ivan; Drobilo, Luka et al.
   RONNA G4—Robotic Neuronavigation: A Novel Robotic Navigation Device for Stereotactic Neurosurgery. // Handbook of Robotic and Image-Guided Surgery / H.

Abedin-Nasab, Mohammad (ur.).
Amsterdam, Netherlands: Elsevier, 2019. str. 599-625 doi:10.1016/B978-0-12-814245-5.00035-9

5. Šuligoj, Filip; Jerbić, Bojan; Šekoranja, Bojan; Vidaković, Josip; Švaco, Marko
Influence of the Localization Strategy on the Accuracy of a Neurosurgical Robot System. // Transactions of FAMENA, 42 (2018), 2; 27-38 doi:10.21278/tof.42203 (međunarodna recenzija, članak, znanstveni)

6. Švaco, Marko; Jerbić, Bojan; Župančić, Ivan; Vitez, Nikola; Šekoranja, Bojan; Šuligoj, Filip; Vidaković, Josip
The Case of Industrial Robotics in Croatia. // Advances in Service and Industrial Robotics. RAAD 2018. Mechanisms and Machine Science, vol 67. / Aspragathos, N. ; Koustoumpardis, P ; , Moulianitis, V (ur.).
Cham: Springer, 2018. str. 607-617 doi:10.1007/978-3-030-00232-9_64 (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

7. Šuligoj, Filip; Švaco, Marko; Jerbić, Bojan; Šekoranja, Bojan; Vidaković, Josip
Automated marker localization in the planning phase of robotic neurosurgery. // IEEE Access, 5 (2017), 12265-12274 doi:10.1109/ACCESS.2017.2718621 (međunarodna recenzija, članak, znanstveni)

8. Švaco, Marko; Šekoranja, Bojan; Šuligoj, Filip; Vidaković, Josip; Jerbić, Bojan; Chudy, Darko; A novel robotic neuronavigation system: RONNA G3. // Strojniški vestnik, 63 (2017), 12; 725-735 doi:10.5545/sv-jme.2017.4649 (međunarodna recenzija, članak, znanstveni)

9. Vidaković, Josip; Jerbić, Bojan; Švaco, Marko; Šuligoj, F ilip; Šekoranja, Bojan; Position planning for collaborating robots and its application in neurosurgery. // Tehnicki vjesnik - Technical Gazette, 24 (2017), 6; 190166, 7 doi:10.17559/tv-20170213110534 (podatak o recenziji nije dostupan, članak, znanstveni)

10. Švaco, Marko; Koren, Petar; Jerbić, Bojan; Vidaković, Josip; Šekoranja, Bojan; Šuligoj, Filip
Validation of Three KUKA Agilus Robots for Application in Neurosurgery. // RAAD 2017: Advances in Service and Industrial Robotics / Ferraresi C., Quaglia G. (ur.).
Torino, Italija: Springer, 2017. str. 996-1006 (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

11. Švaco, Marko; Vitez, Nikola; Jerbić, Bojan; Šuligoj, Filip; Šekoranja, Bojan; Vidaković, Josip
Experimental Evaluation of Parameters for Robotic Contouring Force Feedback Applications. // The International Conference Management of Technology – Step to Sustainable Production (MOTSP 2017) / Predrag Ćosić (ur.).
Zagreb: Croatian Association for PLM, 2017. (poster, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

12. Dlaka, Domagoj; Švaco, Marko; Chudy, Darko; Jerbić, Bojan; Šekoranja, Bojan; Šuligoj, Filip; Vidaković, Josip; Almahariq, Fadi; Romić, Dominik
Brain biopsy performed with the RONNA G3 system: a case study on using a novel robotic navigation device for stereotactic neurosurgery. // International journal of medical robotics and computer assisted surgery, 14 (2017), 1; 10.1002/rcs.1884, 7 doi:10.1002/rcs.1884 (međunarodna recenzija, članak, znanstveni)

13. Vidaković, Josip; Jerbić, Bojan; Šuligoj, Filip; Švaco, Marko; Šekoranja, Bojan
SIMULATION FOR ROBOTIC STEREOTACTIC NEUROSURGERY. // Annals of DAAAM & Proceedings
Mostar, Bosna i Hercegovina, 2016. str. 562-568 (poster, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

14. Švaco, Marko; Jerbić, Bojan; Stiperski, Ivan; Dlaka, Domagoj; Vidaković, Josip; Šekoranja, Bojan; Šuligoj, Filip
T-Phantom: a New Phantom Design for Neurosurgical Robotics. // 27th DAAAM International Symposium On Intelligent Manufacturing And Automation
Beč: DAAAM International Vienna, 2016. str. 266-270 (poster, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

15. Šuligoj, Filip; Jerbić, Bojan; Švaco, Marko; Šekoranja, Bojan; Mihalinec, Dominik; Vidaković, Josip
Medical applicability of a low-cost industrial robot arm guided with an optical tracking system. // Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems
Hamburg, Njemačka, 2015. str. 3785-3790 doi:10.1109/IROS.2015.7353908 (predavanje, međunarodna recenzija, cjeloviti rad (in extenso), znanstveni)

Curriculum vitae

Josip Vidaković was born on March 18th1991. He finished the Technical school of Ruđer Bošković in year 2009, after which he enrolled the Faculty of Mechanical Engineering and Naval Architecture in Zagreb (FSB). He received his B.Sc. and M.Sc. degrees in Mechanical Engineering in 2013 and 2014, respectively. At the end of 2014 he started working on the Robotic Neuronavigation project – RONNA at FSB, at the Department of Robotics and Production System Automation, Chair for Manufacturing and Assembly System Planning, which lasts until April 2016. Since then, he has been employed as a Research Assistant at the same department through the Young Researchers' career development project supported by the Croatian Science Foundation and at the same time he enrolled in the Postgraduate Doctoral study of *Mechatronics and Robotics* at FSB.

His research interests include the fields of industrial and surgical robotics, motion planning, learning from demonstration and machine learning methods in robotics He is author or co-author of seven journal papers of which four indexed in CC and three in SCI databases, and seven papers published in conference proceedings.