Elektrotehniški vestnik 77(4): 179-187, 2010
Electrotechnical Review: Ljubljana, Slovenija

Razvoj modelirnega okolja za učinkovito rekonfiguracijo sistemov na podlagi večagentne arhitekture

Janez Sluga[1], Viktor Zaletelj[1], Andrej Žemva[2]

[1] Trimo d.d., Prijateljeva cesta 12, 8210 Trebnje
[2] Univerza v Ljubljani, Fakulteta za elektrotehniko, Tržaška 25, Ljubljana
E-pošta: janez.sluga@trimo.si; viktor.zaletelj@trimo.si; andrej.zemva@fe.uni-lj.si

Povzetek. Večina današnjih sistemov vodenja rešuje specifične probleme namenskih operativnih struktur, ki v svetu dinamičnih sprememb ne omogočajo preproste rekonfiguracije. Rekonfiguracija pomeni spremembo funkcij in vlog, ki se odražajo na logični, informacijski in fizični ravni sistema. To posledično zahteva tudi ustrezno prilagoditev logike delovanja. Faze načrtovanja, optimizacije in izvedbe sprememb največkrat zahtevajo znatna dodatna sredstva in čas.

V ta namen je bil razvit model informacijske arhitekture na podlagi večagentnih sistemov (MAS), ki omogoča distribucijo funkcionalnosti sistema med avtonomne gradnike – agente. Prednosti lokalne avtonomije so možnosti učinkovitejše rekonfiguracije v realnem času v smislu manjšanja oz. širjenja števila komponent in preprostega vnosa redundance brez modifikacije logike obnašanja posameznih komponent.

Predstavljene funkcionalnosti arhitekture so bile izvedene v programski obliki neodvisnih komponent, ki omogočajo povezavo s strojno opremo, distribuirano okolje izvedbe in simulacije ter fleksibilnost konfiguracije sistema brez dodatnega programiranja, kar je predstavljeno na primeru.

Ključne besede: rekonfiguracija, informacijska arhitektura, večagentni sistemi (MAS), avtonomnost, distribuirano okolje

Development of a modeling environment for efficient system reconfiguration based on MAS

Extended abstract. Most of the today's control systems are solving specific problems of dedicated structures, not allowing easy reconfiguration in the world of dynamic changes. Reconfiguration means changing functions and roles reflected in the logical, informational and physical level of the system. This also requires a proper adjustment of the operational logic. Phases of planning, optimization and implementation of the changes often require significant additional resources and time. For this reason an information architecture model based on multi-agent systems (MAS) was developed. This allows distribution of functionalities between autonomous system blocks - agents. The advantages of local autonomy are potentials for effective real-time reconfiguration in terms of reduction or expansion in the number of components, and easy entry of redundancy without modifying the component behavior logic. The presented functionalities of architecture were made in the form of software for independent components allowing connection with hardware, distributed execution environment and simulation, and also flexibility of system configuration without additional programming.

To achieve effective reconfiguration of the system components without any additional modifications, it is necessary to achieve such a level of system distribution which still allows us to support different physical structures. For this reason we defined specific types of agents: "physical agent", "moderator agent" and "demonstration agent" (Figure 1).
Our solution is presented in a case study which is an example of the developed agent modeling environment for dynamic reconfiguration of manipulation structures. It is important that for the purpose of reconfiguration we can describe all components with a general mathematical model which represents the use of positions and transformations for objects in the environment (Figure 2). A description of the structure in the environment for one component is shown in equations (1-12). The movement of one component, which also affects all the after connected components, is described in equations (13-19).

We performed simulations of different manipulation structures, where we used translational and rotational actuators for active components. Results of the movement to a new required point at different solving algorithms of 1-DOF and 3-DOF one axial agent system are shown in Figure 3. Figure 4 shows how the multi-agent system performs a movement of a circular shape for various configurations of manipulation structures without any change in the agent logic.

The represented solution of the agent information architecture was proven to be effective, since the information model turns out to be sufficient for making rapid reconfigurations in a real world or for adapting to new changes in the environment.

Prejet 21. maj, 2010
Odobren 7. julij, 2010

Keywords: reconfiguration, information architecture, multi-agent systems (MAS), autonomy, distributed environment

1 Uvod

V današnjem dinamičnem svetu nastajajo potrebe po hitrih spremembah funkcionalnosti in/ali strukturiranosti sistemov, ki v fazi snovanja še niso bile zaznane oz. predvidene. S takimi problemi se srečujejo v različnih proizvodnih, gradbenih, skladiščnih, montažnih, informacijskih in drugih okoljih. V proizvodnji nastane potreba po prilagajanju linije novim izdelkom, v skladišču lahko pride do spremembe namembnosti in posledično do spremembe strukture skladišča zaradi drugega skladiščnega materiala ali izdelka. Montažni sistem se lahko znajde v situaciji, kjer za izvedbo določenih nalog nima zadostnih zmožnosti glede nosilnosti ali dolžine komponent, zato mu je treba dodati nove aktivne komponente, ki mu te zmogljivosti oz. funkcionalnosti povečajo.

Ko govorimo o vodenju takšnih sistemov, klasične informacijske arhitekture, ki temeljijo na hierarhično organiziranih komponentah, takim zahtevam brez korenitih posegov na logični, podatkovni in povezovalni ravni niso sposobne slediti. Hkrati je količina potrebnih sredstev, znanja in časa največkrat prevelika za učinkovito, ekonomsko sprejemljivo in hitro spremembo.

Na področju rekonfigurabilnosti so bile v preteklosti opravljene nekatere raziskave o primernosti različnih arhitektur za dosego modularnosti in razširljivosti sistemov. Raziskave so se v preteklosti osredotočale na rekonfigurabilnost mehanskih in robotskih sklopov ter predvsem doseganju hitre povezljivosti (standardizacija) in razpoznavanju komponent (hot plug&play), pri čemer pa so bili uporabljeni klasični hierarhično vodeni pristopi upravljanja sistemov. Ta težnja je v zadnjem času preusmerjena na raziskave iskanja alternativnih arhitektur sistemov in orodij za njihovo načrtovanje (agentni sistemi, nevronske mreže, genetski algoritmi, itd.). V prispevku je opisan predlog, ki omogoča preprosto rekonfiguracijo strukture sistema za dosego želene funkcionalnosti, pri tem pa ni potreben poseg v informacijsko arhitekturo in logiko delovanja avtonomnih enot. Opisani so arhitektura, sistem vnosa sprememb funkcij in vlog ter prikazana primernost omenjenega pristopa na primeru.

Članek je razdeljen na šest poglavij. Daljši uvod v problematiko je opisan v drugem poglavju, v tretjem pa je prikazana predlagana rešitev za opisano problematiko. Rešitev smo tudi testirali, rezultate podali v četrtem poglavju, v petem pa smo navedli sklepne ugotovitve in zaključke.

2 Opis problematike

Nerešeni izzivi v proizvodnih, montažnih in drugih realnih okoljih so učinkovite spremembe funkcionalnosti sistema, zmožnosti in strukturiranosti glede na nove zahteve. Pri tem si prizadevamo, da so stroški in čas, porabljeni za modifikacije, minimalni. Nekatere rešitve ponujajo sodobne metode realno-časovne rekonfiguracije, pravilno izbiro informacijske arhitekture in povezanost zadnje z realnimi komponentami fizičnega sveta.

2.1 Rekonfiguracija

Rekonfigurabilnost sistema je zmožnost le-tega spreminjati funkcionalnosti, njihov nabor oz. obseg ter vloge in relacije med gradniki strukture. Dinamična rekonfiguracija pomeni zmožnost sistema, da med operativnim delovanjem prilagaja svoje obnašanje glede na spremembe in zahteve okolja.

Za pravilno izvajanje želene funkcionalnosti moramo v strukturo sistema smiselno umestiti primerne gradnike. To storimo tako, da za vsako specifično nalogo in okolje zasnujemo strukturo, ki jo nato podpremo z logiko delovanja. Za kakršnokoli spremembo operativnih zahtev ali strukture je treba sistem ponovno modelno zasnovati, optimizirati in po potrebi rekonfigurirati na realnih komponentah.

Za spremembe strukture mehanskega sistema je bil predlagan pristop agenta, ki temelji na znanju osnovnih pravil okolja v katerem se nahaja. Omenjeni agent rekonfigurira proizvodni sistem, ko zazna spremembe v zahtevah ali v proizvodnem okolju. V večini primerov pa agent ne pozna celotnega okolja in komponent, ki jih ima na voljo, zato ni sposoben upoštevati vseh dejavnikov pri rekonfiguraciji. Ravno tako ta pristop ni bil izdelan tako daleč, da bi bila mogoča integracija na ravni krmilnikov. [1]

Potrebe po rekonfiguraciji velikokrat nastanejo tudi v modularni ali celični robotiki, kjer se morajo avtonomni moduli postaviti v različne konfiguracije za različne namene. Raziskave so bile izvedene na verižnih in mrežastih tipih konfiguracij sistemov. Verižni sistemi so v glavnem zgrajeni iz enodimenzionalne ogrodne strukture, ki se lahko premika in dograjuje v eni smeri, medtem ko mrežasti sistemi težijo k zapolnjevanju prostorov modulov, ki se izoblikujejo relativno tesno v dve ali tri smeri. Glavna izziva sta vodenje sodelovanja in koordinacija modulov. [2]

Za realnočasovno rekonfiguracijo se najpogosteje uporablja pristop s porazdeljeno umetno inteligenco pri planiranju in na ravni kontrole za dosego hitrejšega odziva pri spremembah. Pristop temelji na agentnih metodah in je bil pripeljan do razvoja konceptualne arhitekture. [3]

2.2 Vnos sprememb v sistem

V klasičnem pristopu pri zasnovi in izvedbi sistema vodenja je treba za kakršnokoli modifikacijo strukture gradnikov ponoviti postopek njegove zasnove in izvedbe, kar na splošno pomeni zamudno in drago prilagajanje logike delovanja sistema. Avtomatskega prilagajanja logike delovanja za opravljanje funkcije v drugačni strukturi pa obstoječe metode ne rešujejo.

2.3 Informacijska arhitektura

Obstoječi pristopi, ki temeljijo na hierarhično organiziranih informacijskih arhitekturah, potrebujejo zmogljive centralnoprocesne enote, ki lahko ob vnosu sprememb v informacijsko ali fizično strukturo zaradi nepredvidenosti takih situacij hitro odpovedo. Ravno tako nimajo zmožnosti samodejnega prilagajanja novim zahtevam in konfiguracijam sistemskih komponent. Za prilagajanje na spremembe je nujno, da se lahko spreminjajo tudi relacije med posameznimi informacijskimi komponentami, ter prilagajanje logike delovanja.

Za premostitev omenjene problematike in klasičnega pristopa s hierarhično arhitekturo je predlagana distribuirana ali porazdeljena arhitektura večagentnih sistemov. Glavna lastnost agentov je, da delujejo avtonomno. Agent deluje v dinamičnem okolju, katerega lahko zaznava in v katerem s sodelovanjem izvaja akcije za dosego skupnega cilja, pri čemer so njegove odločitve avtonomne. Iz literature poznamo reaktivne, preudarne in hibridne agente. [4-6]

Reaktivni agenti se osredotočajo samo na trenutni položaj in zato težje dosegajo časovno daljše oz. globalnejše cilje. Glavna pomanjkljivost take arhitekture je težka implementacija proaktivnosti in k cilju usmerjenega obnašanja. Preudarna agentna arhitektura ali arhitektura BDI (angl. believe-desire-intention) eksplicitno opiše cilje in oblikuje načrt, kako se bo agent obnašal v prihodnosti, da bi dosegel te cilje. Pomanjkljivost agentov BDI je ta, da se lahko odzovejo šele potem, ko gre novo stanje na senzorskem vhodu skozi vse potrebne korake za oblikovanje namena. Arhitektura hibridnih agentov pa vključuje mehanizme reaktivnih in preudarnih agentov. Glavna pomanjkljivost je težko načrtovanje koordinacije med nivoji v sami arhitekturi. [4-6]

Porazdeljena večagentna arhitektura je bila uporabljena tudi pri humanoidnem robotu, kjer so razdelili celoten sistem na osnovne gradnike – aktuatorje. Vsak aktuator posebej predstavlja agenta in ima podano neko stopnjo avtonomnosti. Takšna arhitektura je odporna na motnje ali odpovedi, saj ob izgubi enega ali več agentov sistem te nadomesti s prilagajanjem. Problem je pri takem pristopu vodenje hitrosti in pravočasne izvedbe aktivnosti. [7]

Izkaže se, da je pristop večagentnih sistemov primeren tudi za uporabo pri premostitvi problematike rekonfiguracije strukture komponent sistema na ravni informacijske arhitekture.

3 MAS arhitektura za rekonfiguracijo komponent

V tem poglavju je predstavljen predlog modela informacijske arhitekture, ki bo omogočala učinkovito rekonfiguracijo strukture sistema, predstavitev strukture sistema v prostoru in njen opis, ter opis povezave med modelirnim in fizičnim okoljem pri tovrstnem pristopu.

3.1 Model informacijske arhitekture

Za doseganje učinkovite rekonfiguracije komponent sistema brez dodatnih modifikacij je treba z informacijsko arhitekturo doseči raven razdelitve sistema, ki nam še omogoča podporo poljubne fizične strukture. Ker hierarhična informacijska arhitektura ne ustreza zahtevam hitrega vnašanja sprememb v strukturo brez dodatnega spreminjanja logike delovanja, moramo strukturo informacijskega sistema razdeliti na osnovne gradnike in jim podati avtonomnost. Tako imamo možnost znova sestaviti mrežo gradnikov v želeno strukturo sistema. Trenutno je najprimernejša informacijska arhitektura za tovrstno uporabo arhitektura večagentnih sistemov.

Predlagana informacijska arhitektura je sestavljena iz več sodelujočih inteligentnih agentov. Ti si prizadevajo najti rešitev za nastalo problematiko oz. prejeto operativno nalogo z medsebojnim sodelovanjem brez intervencije subjekta.

Model predlagane informacijske arhitekture sestavljajo množica avtonomnih agentov, ki informacijsko povezujejo posamezne komponente oz. zaključene funkcionalnosti sistema. Z namenom poenostavljanja in razširljivosti so definirani nekateri specifični tipi agentov.

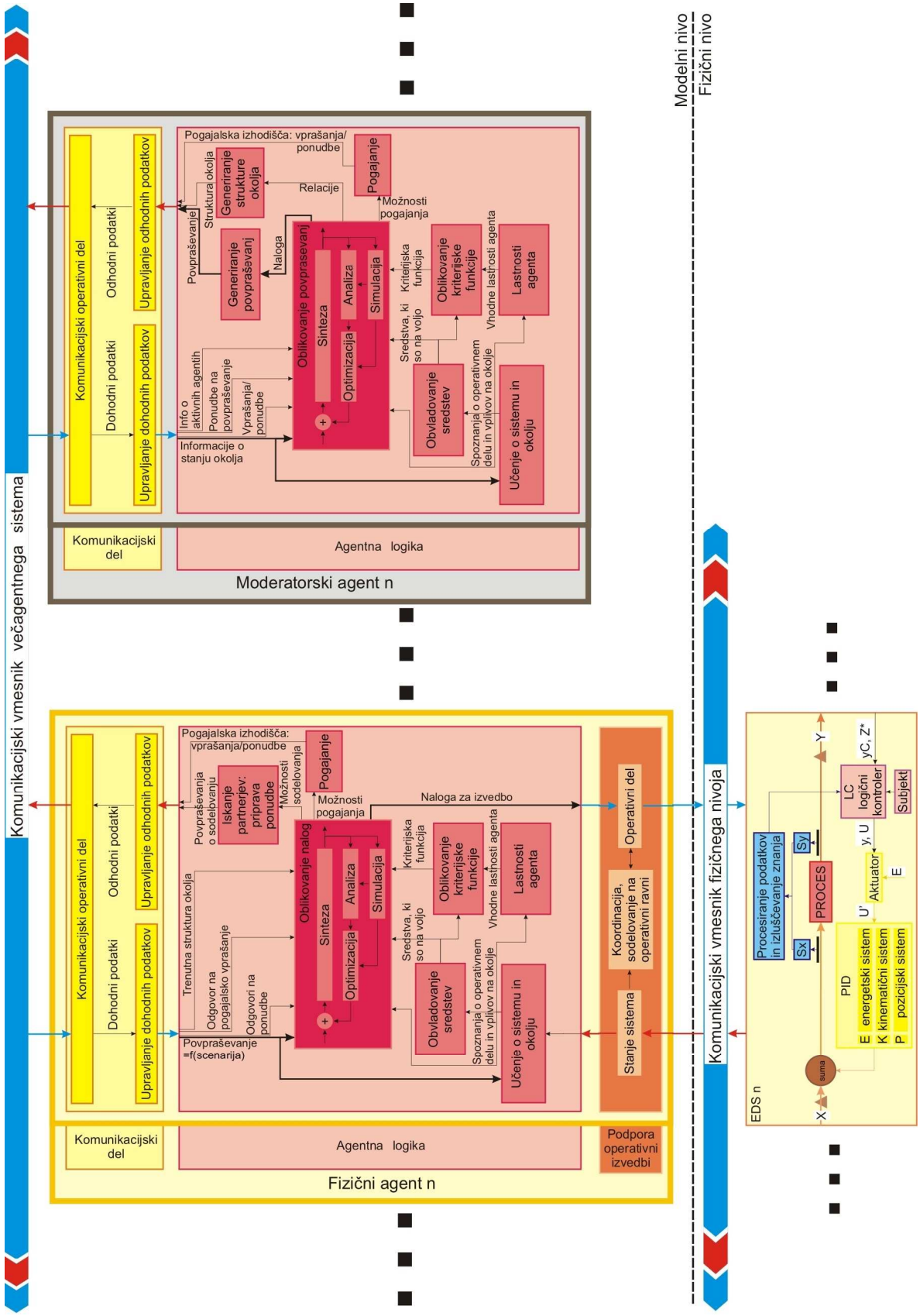
Osnovni tip je »fizični agent«, ki povezuje in zastopa realni svet s svojo informacijsko odslikavo. Agent, ki prevzema naloge, oblikuje cilje, povezuje in strukturira nove forme ter usklajuje sodelovanje fizičnih agentov na poti k skupnemu cilju, je »moderatorski agent«. Za interakcijo s subjekti, analizo stanja sistema in njegovo razumljivo predstavitev, skrbi »predstavitveni agent«. Zaradi povezljivosti in nadgradenj vsi tipi agentov ohranjajo podobno strukturno zasnovo, predstavljeno na sliki 1.

Fizični agent je sestavljen iz treh ključnih funkcionalnih sklopov:

- komunikacijskega,
- odločitvenega z agentno logiko,
- operativnega za podporo in interakcijo s fizičnim svetom.

Komunikacijski del skrbi za pravilen potek in upravljanje dohodnih ter odhodnih podatkov pri izmenjavi informacij med agenti. Njegova informacijska arhitektura je neodvisna od tipa komunikacijskega protokola. Zasnova z jasno opredeljenimi vmesniki omogoča uporabo kateregakoli želenega komunikacijskega protokola oz. njihovo zamenjavo tudi v realnem času.

Osnovna naloga agentne logike je odločanje o aktivnostih agenta in s tem priprava nalog za izvedbo na fizični ravni. Naloge se izoblikujejo na podlagi povpraševanja za definirano strukturo okolja, na podlagi



Slika 1: Model informacijske arhitekture na podlagi MAS v povezavi s fizičnim nivojem
Figure 1: Information architecture model based on MAS in relation with the physical level

sredstev, ki jih ima agent na voljo, na podlagi kriterijske funkcije in naučenih zakonih o okolju in operativnem delu. Povpraševanja so funkcije scenarija, ki je sestavljen iz stanja v okolju, omejitev v njem, od želene strukture sistema in iz skupnih ali posamičnih ciljev. Kriterijska funkcija odločanja se lahko razlikuje za vsakega posameznega agenta, saj je odvisna od sredstev, ki jih ima na voljo, in trenutnih lastnosti agenta, kot so cilji, prepričanja, želje in načini obnašanja. Na oblikovanje kriterijske funkcije vplivajo tudi naučena spoznanja o operativnem delu in vplivov zadnjega na okolje. Ta spoznanja so pridobljena z zaznavanjem omejitev in zmožnosti elementov strukture ter okolja, na podlagi česar se oblikujejo nova pravila oz. lastnosti obnašanja agenta.

Sinteza naloge v agentni logiki poteka na podlagi povpraševanja, sredstev, spoznanj, kriterijske funkcije in stanja v strukturi. Hkrati se preverijo možnosti sodelovanja z drugimi agenti in če obstajajo potencialni partnerji, se jim pošljejo ponudbe. Po prejetju odgovora se opravi analiza in simulacije za izbrano nalogo. Na podlagi dobljenih rezultatov se lahko izvaja optimizacija z nadaljnjimi pogajanji in izboljšanji ponudb ali pa se odloči za izvedbo trenutno oblikovane naloge. V prvem primeru pošlje vsem ponudnikom nova povpraševanja oz. ponudbe ter ponavlja celoten postopek analize, simulacije, optimizacije ter sinteze, dokler ne dobi optimalno oblikovane naloge glede na njegove želje in cilje.

Oblikovane naloge za izvedbo se prenesejo v agentni podsklop podpore operativni izvedbi. Tu se:

- pretvarjajo sklenjeni dogovori glede izvedbe nalog v operativne ukaze krmilnikom na fizičnih ravneh,
- prejemajo informacije o trenutnem stanju fizičnega sistema,
- vzpostavlja direktna komunikacija prek hitrih komunikacijskih vodil med agenti sistemov, ki morajo izvajati visoko usklajene (časovno, hitrostno) operacije.

Ta del zagotavlja vez med informacijskim (virtualnim) in fizičnim (operativnim) nivojem sodobnih sistemov.

Poslanstvo moderatorskega agenta je, da subjekt prevede naloge v oblikovanje primerne informacijsko-fizične strukture, ki jih bo sposobna izvesti pod danimi omejitvami.

Moderatorski agent je v informacijski arhitekturi enak fizičnemu agentu brez podsklopa podpore operativni izvedbi. Ključne razlike nastanejo v posameznih delih agentne logike, v tipih vhodov, mehanizmov, kontrol in izhodov. Moderatorski agent na podoben način oblikuje povpraševanja, kot fizični agent oblikuje naloge. Sintezo povpraševanja opravi na podlagi zahtevane naloge, ki jo dobi od subjekta prek predstavitvenega agenta. Pri tem upošteva stanje v okolju, seznam aktivnih agentov v tem okolju, globalno kriterijsko funkcijo, sredstva, ki jih ima na voljo, ter naučena spoznanja o okolju. Po sintezi povpraševanja pošlje slednje in generirano strukturo vsem agentom, ki bi jih lahko opravljanje naloge zanimalo. Glede na pridobljene ponudbe in vprašanja izvede simulacijo in analizo obnašanja celotnega sistema. Rezultate uporabi za optimizacijo, pri tem pa se odloči, ali bo sprejel katero izmed ponudb oz. ali bo nadaljeval optimiziranje povpraševanja s pogajanji z izbranimi ponudniki. Nazadnje se odloči za najboljšo varianto ter o tem obvesti izbrane izvajalce nalog.

Predstavljena arhitekturna zasnova sistema omogoča:

- vključevanje poljubnega števila posameznih tipov agentov in s tem a) razširljivost, b) redundantnost ter c) segmentacijo in specializacijo,
- skupno komunikacijsko hrbtenico in s tem vzpostavljanje direktne komunikacije med poljubnimi agenti,
- prevzemanje vlog operativnega in simulacijskega izvajalca, ter tako a) pridobljeno distribuirano simulacijsko okolje, ki se širi s številom vključenih agentov in b) možnost izvedbe redundantnih izračunov oz. scenarijev v hkratnem času s sodelovanjem agenta v več simulacijskih scenarijih (vzporedno izvajanje),
- podobnost zasnove agenta ne glede na tip oz. njegovo vlogo in s tem a) reuporabnost komponent ne glede na cilj oz. namen delovanja ter b) izmenljivost,
- prilagodljivost trenutnemu stanju okolja oz. spremembah v ciljih.

Vse avtonomne agentne enote, ki so del okolja in se vključujejo v strukturo sistema, oznanijo moderatorskemu agentu svojo prisotnost in svoje zmožnosti. Slednji definira želeno konfiguracijo komponent in prek komunikacijskega vmesnika pošlje agentom informacijo o umeščenosti v okolju in vlogi v njem. Po posredovanem globalnem cilju opisovanega sistema posamezni agenti prilagodijo logiko delovanja novim razmeram. Nato poskušajo z avtonomno analizo stanja v okolju in zmožnosti lastnega prispevka h globalnemu cilju poiskati optimalne akcije ter jih tudi pravočasno izvesti.

Izvedbo na fizični ravni omogočajo realne aktuatorske komponente oz. EDSi (glej sliko 1), ki komunicirajo z recipročnim informacijskim modelnim svetom (fizični agent) prek agentnega podsklopa »podpora operativni izvedbi«.

3.2 Rekonfiguracija v realnem času

Izvedbo rekonfiguracije v realnem času nam na logični ravni omogoča informacijska arhitektura večagentnih sistemov. Prek moderatorskega agenta podajamo kompozicijo strukture v realnem času. To pomeni, da lahko med samim delovanjem spremenimo strukturo

sistema ter agentom sporočimo nove podatke o umeščenosti in stanja v okolju ter omejitve fizičnega nivoja. Agent posredovane spremembe sprejme ter ustrezno prilagodi svoje obnašanje.

3.3 Povezava fizičnega in modelnega okolja

Pristop na podlagi večagentne informacijske arhitekture, kjer je vsak aktuator zastopan z agentom, omogoča direktno povezavo med modelnim in realnim svetom.

Vsak agent, ki zastopa realni svet, lahko vzporedno z operativnim delom opravlja tudi simulacije na želeni strukturi sistema. Tako se preverja pravilno delovanje ter sproti izvajajo potrebne modifikacije na strukturi za dosego zahtevanih operativnih specifikacij. Modelni svet postaja hkrati tudi simulacijsko okolje in je kot tako lahko orodje za napovedovanje obnašanja sistema v fizičnem okolju.

4 Primer

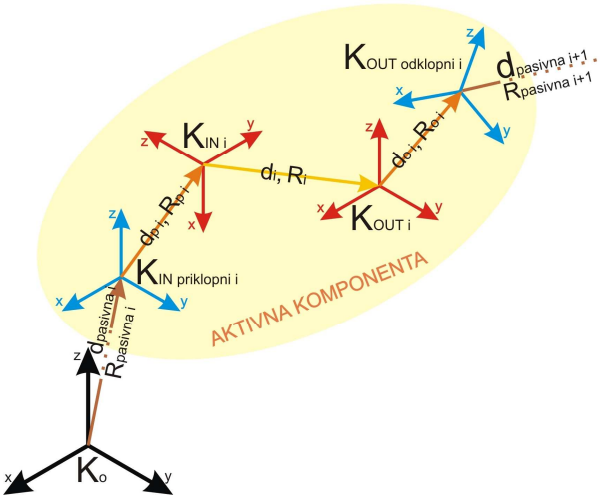
V nadaljevanju je predstavljen primer uporabe razvitega agentnega modelirnega okolja za potrebe dinamičnih rekonfiguracij manipulacijskih struktur. Cilj razvoja takšnega sistema mora omogočiti izvedbo zahtevane kinematike gibanja manipulatorja v prostoru za različne dinamično nastavljive strukture samega manipulatorja, ne da bi med rekonfiguracijo modificirali logiko aktivnih komponent sistema (ki jih zastopajo agenti).

Za potrebe rekonfiguracije je pomembno, da lahko vse komponente sistema opišemo s splošnim matematičnim modelom, ki v pričujočem primeru pomeni obvladovanje lege in transformacij objektov v prostoru.

Ker govorimo o želji po učinkoviti rekonfiguraciji med seboj različnih fizičnih komponent, smo za njihov lažji opis v modelu nadgradili standarden opis robotskega manipulatorja. Ta temelji na opisu ključnih aktivnih komponent sistema (translacijski oz. rotacijski aktuatorji) z ustreznimi homogenimi transformacijami, ki pa ne upošteva dejanskih mest medsebojnega priklopa, kar sicer za standardiziran produkt (robot) zadošča.

V tridimenzionalnem prostoru je struktura sistema na splošno določena z aktivnim in pasivnim delom. Pasivni deli so povezava med komponentami ali povezava med komponento in okoljem. Pasivno komponento določimo z vektorjem pozicije ($d_{pasivna}$) in matriko orientacije ($R_{pasivna}$), ki določata lego »priklopnega« koordinatnega sistema glede na predhodni koordinatni sistem ($K_{IN\ priklopni}$, $K_{OUT\ odklopni}$), kar pa v primerih aktivnih komponent še ne določa njihovih aktuatorskih sposobnosti.

Aktivna komponenta je sestavljena iz štirih koordinatnih sistemov, ki so prav tako medsebojno povezani z vektorji pozicije in matrike orientacij glede na predhodne koordinatne sisteme. Začetni del komponente predstavimo s koordinatnim sistemom priklopa ($K_{IN\ priklopni}$) in vektorjem priklopa, končni del pa s koordinatnim sistemom odklopa ($K_{OUT\ odklopni}$) in vektorjem odklopa. Vmesni del, ki vnaša spremembe v aktivno komponento in v celoten sistem, je sestavljen iz koordinatnih sistemov vhoda (K_{IN}) in izhoda (K_{OUT}), povezuje pa ju vektor aktivne komponente. Vsi premiki ali rotacije se odražajo na spremembi pozicije in usmerjenosti odklopnega koordinatnega sistema ($K_{OUT\ odklopni}$) in vseh drugih koordinatnih sistemov, ki so povezani z njim (slika 2).



Slika 2: Splošni model aktivne komponente v prostoru
Figure 2: General model of the active component in space

Spremembe koordinatnih sistemov v prostoru opišemo v matrični obliki homogene transformacije:

$$\underline{\underline{H}} = \begin{bmatrix} \underline{\underline{I}} & \underline{d} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \underline{\underline{R}} & \underline{0} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \underline{\underline{R}} & \underline{d} \\ 0 & 1 \end{bmatrix} \quad (1)$$

Premik vektorja, točke ali lika lahko zapišemo kot preslikavo:

$$\underline{x'} = \underline{\underline{H}} \cdot \underline{x} \quad (2)$$

Skupna rotacija je sestavljena iz učinkov posameznih rotacij po oseh x, y, z za kote θ, ψ, ϕ:

$$\underline{\underline{R}} = \underline{\underline{Rot}}(x,\theta) \cdot \underline{\underline{Rot}}(y,\psi) \cdot \underline{\underline{Rot}}(z,\phi) \quad (3)$$

$$\underline{\underline{Rot}}(x,\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \quad (4)$$

$$\underline{\underline{Rot}}(y,\psi) = \begin{bmatrix} \cos\psi & 0 & \sin\psi \\ 0 & 1 & 0 \\ -\sin\psi & 0 & \cos\psi \end{bmatrix} \quad (5)$$

$$\underline{\underline{Rot}}(z,\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6)$$

Translacija je enaka spremembi premika po oseh:

$$\underline{d} = \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} \quad (7)$$

Opis strukture v prostoru za i-to komponento:

$$\begin{bmatrix} K_{IN\ priklopni\ i} \\ 1 \end{bmatrix} = \begin{bmatrix} \underline{\underline{R_{pasivna\ i}}} & d_{pasivna\ i} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{OUT\ odklopni\ i-1} \\ 1 \end{bmatrix} \quad (8)$$

$$\begin{bmatrix} K_{IN\ i} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{p\ i} & d_{p\ i} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{IN\ priklopni\ i} \\ 1 \end{bmatrix} \quad (9)$$

$$\begin{bmatrix} K_{OUT\ i} \\ 1 \end{bmatrix} = \begin{bmatrix} R_i & d_i \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{IN\ i} \\ 1 \end{bmatrix} \quad (10)$$

$$\begin{bmatrix} K_{OUT\ odklopni\ i} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{o\ i} & d_{o\ i} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{OUT\ i} \\ 1 \end{bmatrix} \quad (11)$$

In za i+1:

$$\begin{bmatrix} K_{IN\ priklopni\ i+1} \\ 1 \end{bmatrix} = \begin{bmatrix} R_{pasivna\ i+1} & d_{pasivna\ i+1} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{OUT\ odklopni\ i} \\ 1 \end{bmatrix} \quad (12)$$

…, pri čemer je i=1,2,…,n; n = število komponent.

Premik i-te komponente opišemo z matriko homogene transformacije, vpliva pa na vse komponente, ki so priklopljene za njo.

Homogena transformacija premika i-te komponente:

- Rotacija: $H_i = \begin{bmatrix} R_i & 0 \\ 0 & 1 \end{bmatrix}$ (13)

ali

- Translacija: $H_i = \begin{bmatrix} I & d_i \\ 0 & 1 \end{bmatrix}$ (14)

$$\begin{bmatrix} K_{IN\ priklopni\ i}{}' \\ 1 \end{bmatrix} = \begin{bmatrix} K_{IN\ priklopni\ i} \\ 1 \end{bmatrix} \quad (15)$$

$$\begin{bmatrix} K_{IN\ i}{}' \\ 1 \end{bmatrix} = \begin{bmatrix} K_{IN\ i} \\ 1 \end{bmatrix} \quad (16)$$

$$\begin{bmatrix} K_{OUT\ i}{}' \\ 1 \end{bmatrix} = \begin{bmatrix} K_{OUT\ i} \\ 1 \end{bmatrix} \cdot H_i \quad (17)$$

$$\begin{bmatrix} K_{OUT\ odklopni\ i}{}' \\ 1 \end{bmatrix} = \begin{bmatrix} R_{o\ i} & d_{o\ i} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{OUT\ i}{}' \\ 1 \end{bmatrix} \quad (18)$$

In za i+1:

$$\begin{bmatrix} K_{IN\ priklopni\ i+1}{}' \\ 1 \end{bmatrix} = \begin{bmatrix} R_{pasivna\ i+1} & d_{pasivna\ i+1} \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} K_{OUT\ odklopni\ i}{}' \\ 1 \end{bmatrix} \quad (19)$$

…, pri čemer je i=1,2,…,n; n = število komponent.

Na podlagi podanega matematičnega opisa lahko izračunamo lego sistema in posameznih komponent v prostoru in s tem ocenjujemo ustreznost glede na postavljene cilje (hitrost odziva, doseganje toleranc, učinkovitost izvedbe ciljne manipulacije).

Za preverjanje ustreznosti predlagane informacijske arhitekture so bili izdelani programska oprema (z uporabo NI-LabView) agentnega informacijskega okolja in posamezni informacijski gradniki agentov.

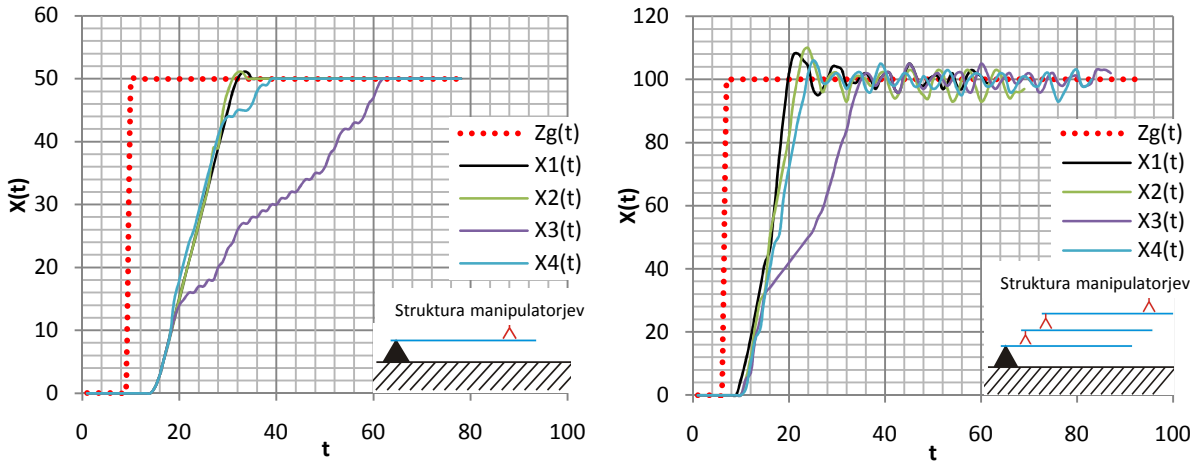
Razvita programska oprema omogoča umeščanje fizičnih, moderatorskih in predstavitvenih agentov v agentno okolje, kjer medsebojno delujejo kot programsko neodvisni procesi.

Glede na zahteve subjekta in trenutno prijavljenih aktivnih fizičnih agentov prek moderatorja formalno opišemo manipulacijsko strukturo. Za opis te je bilo razvito orodje določitve medsebojnega povezovanja ($K_{OUT\ i}$, $K_{IN\ i+1}$) posameznih statičnih in dinamičnih (fizični agenti) komponent. Določita se bazni ($K_{IN\ 1}$)in končni element strukture ($K_{OUT\ končni}$) ter glede na nalogo poda dinamičen globalni cilj (Zg(t)), ki ga mora večagentni sistem doseči.

Iz definirane strukture moderator formulira enačbe direktne kinematike. To informacijo pošlje prek komunikacijskega vmesnika vsem sodelujočim fizičnim agentom.

Vsak fizični agent vsebuje programski algoritem, s katerim preverja potencialne prispevke lastnih odločitev k rešitvi globalnega cilja glede na stanje lastnega aktuatorja in senzorike ter glede na dosegljive informacije o stanju drugih komponent strukture. V opisanih primerih so uporabljeni fizični agenti reaktivnega tipa, ki izvedejo lasten prispevek z najmanjšim pogreškom brez usklajevanja (prostorsko-časovnega) z drugimi avtonomnimi enotami v manipulatorju. Pomembno je, da pri spremembi strukture manipulatorja posamezen fizični agent ne občuti nikakršne spremembe v logiki svojega delovanja, na podlagi česar se zagotavlja visoka stopnja fleksibilnosti celotnega sistema.

Komunikacijski vmesnik, prek katerega usklajujemo



Slika 3: Odzivi 1-DOF in 3-DOF enoosnega agentnega sistema na zahtevan premik v novo točko pri različnih krmilnih algoritmih reševanja nalog

Figure 3: Results of 1-DOF and 3-DOF one-axial agent system movement to a new required point for different solving algorithms

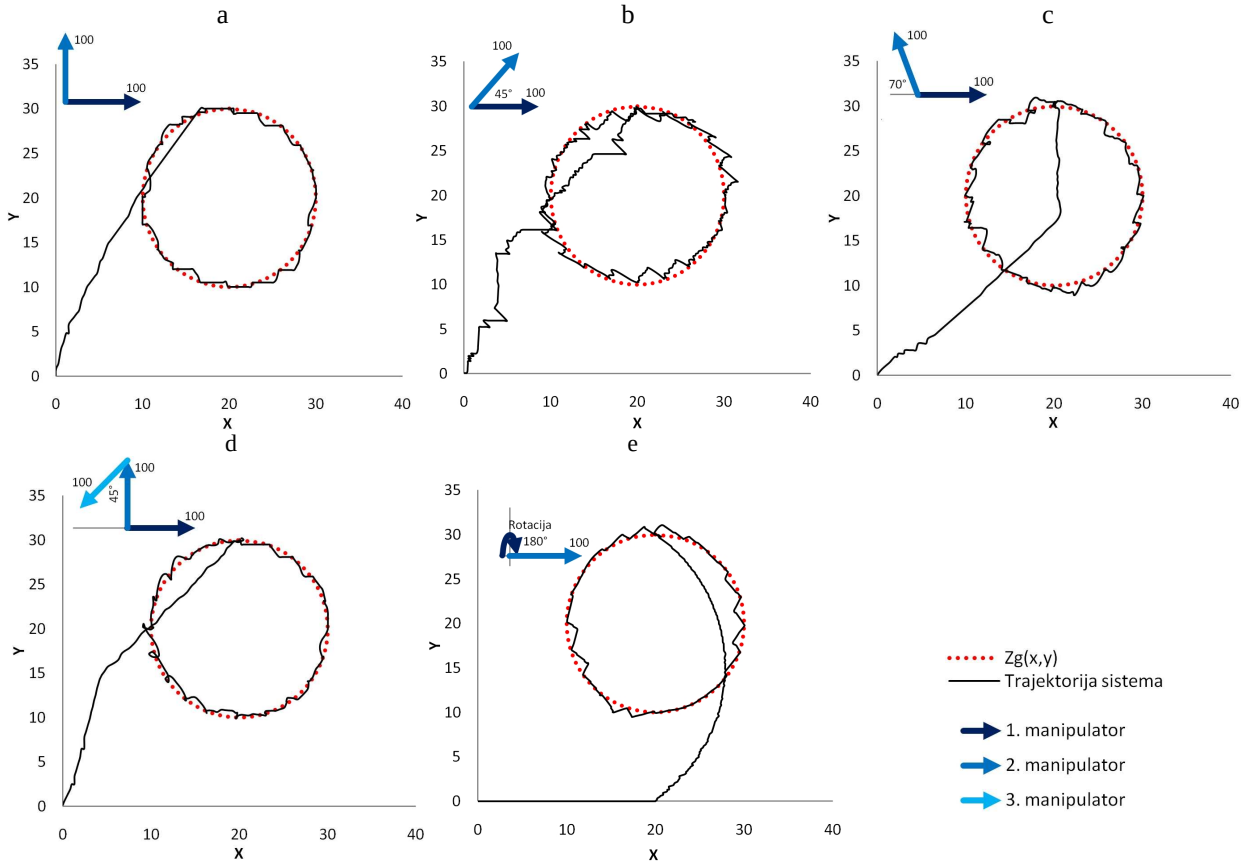
informacije o okolju ter stanja v njem, je izdelan z uporabo funkcionalnih mrežnih spremenljivk (TCP/IP) in je za vse tipe agentov identičen. Mrežne spremenljivke vsebujejo opis strukture manipulatorja, stanja večagentnega sistema, vrednosti aktuatorjev in so javno objavljene na komunikacijski mreži ter dostopne vsem.

V nadaljevanju so prikazani primeri simulacije različnih manipulacijskih struktur, kjer smo za aktivne komponente uporabili translacijske in rotacijske aktuatorje. Vsaki od teh komponent smo podali avtonomnost z dodeljenimi reaktivnimi agenti z različnimi reakcijskimi časi, omejitvami (npr. dolžine gibov) in algoritmi, na podlagi katerih potekajo sinteza, analiza ter optimizacija izvedbe naloge. Primeri si sledijo po zahtevnosti od 1D do 2D manipulacije v prostoru.

V prvem primeru smo izvedli simulacijo 1D gibanja v večagentnem okolju, kjer smo imeli enega moderatorskega in enega fizičnega agenta. Fizičnemu agentu smo podali kontrolo nad 1-DOF aktuatorjem in spremljali odzive na zahtevano manipulacijo (premik v novo točko) pri konstantnem pospešku, omejeni maksimalni hitrosti ter različnih algoritmih reševanja nalog (slika 3 levo). Algoritmi reševanja nalog so se razlikovali po načinu delovanja in izbiri prispevka h globalnemu cilju. Na začetku poskusa smo uporabili osnoven algoritem, ki se odzove na prejeti globalni cilj (zakasnitev odziva je posledica pretoka informacij) ter se s konstantnim pospeškom približuje najvišji možni predpisani hitrosti. S končno hitrostjo doseže cilj, nato pa začne zavirati, kar ima za posledico prevzpon v odzivu (X1(t)). V nadaljevanju eksperimenta smo algoritem dopolnili s predhodnim zaviranjem, ki temelji na predvidevanju dosega cilja glede na hitrostne razmere končne točke manipulatorja. Predstavljeni primeri X2(t), X3(t) in X4(t) se medsebojno razlikujejo po različni varnostni konstanti, ki zagotavlja lokalno neprenihanje sistemskih komponent.

V drugem primeru smo spremljali delovanje enoosnega 3DOF sistema, katerega komponente so bile vezane soosno, zaporedno. Vsak agent je imel kontrolo nad svojim 1-DOF manipulatorjem, razlikovali pa so se po reakcijskih časih in omejitvah (npr. pospešek, maksimalna hitrost). Na sliki 3 desno so prikazani rezultati simulacij. Vsi vpleteni agenti dobijo sočasno informacijo o globalnem cilju ter se odzovejo nanj brez medsebojnega usklajevanja. Na ravni posameznega fizičnega agenta so bili uporabljeni isti algoritmi reševanja nalog kot v prvem primeru. Ker se tu prispevki vseh manipulacijskih komponent seštevajo, lahko pričakujemo hitrejše doseganje cilja, a tudi večje prenihanje, kar je bilo ublaženo z višjimi varnostnimi faktorji.

V nadaljevanju smo se osredotočili na problem gibanja v ravnini (krožnica) pri različnih konfiguracijah



Slika 4: Izvedba opisa krožnice z večagentnim sistemom pri različnih konfiguracijah manipulacijskih struktur
Figure 4: Results of multi-agent system movement in a shape of circle for various configurations of manipulation structures

manipulacijskih komponent. Na podlagi predhodno dobljenih rezultatov simulacij odziva večagentnega enoosnega sistema na zahtevan globalni cilj smo izbrali najobetavnejši algoritem reševanja nalog, ki je bil kombinacija hitre odzivnosti na novo prejeti globalni cilj in predvidevanjem dogodkov v bližini cilja. Rezultati za različne konfiguracije manipulatorja so prikazani na sliki 4, kjer lahko opazimo uspešno izvedbo opisa krožnice ob spremembi strukture manipulacijskih komponent, pri tem pa ni bil potreben noben poseg v samo logiko delovanja agentskega sistema. Prva posebnost je prikazana na sliki 4b in c, kjer drugi manipulator (fizični agent) ob premiku povzroča globalni premik v dveh oseh hkrati, kar mora prvi manipulator kompenzirati. Druga posebnost je prikazana na sliki 4d z uporabo redundance v manipulacijski strukturi. Slika 4e prikazuje rezultat gibanja ob uporabi kombinacije translacijskega in rotacijskega fizičnega agenta.

5 Sklep

V delu je predstavljena zasnova agentne informacijske arhitekture, ki omogoča preprosto rekonfiguracijo strukture sistema za dosego želene funkcionalnosti brez posega v logiko delovanja avtonomnih enot. Izdelana je bila programska oprema, s katero smo dokazali ustreznost zasnove sistemov vodenja s predlagano arhitekturo na primeru vodenja večosnih manipulatorskih sistemov.

Iz rezultatov simulacij je razvidno, da je bil izdelan programski algoritem reaktivnih agentov, kar pomeni, da se agenti neposredno odzivajo na trenutno stanje v okolju brez povezave z izkušnjami iz preteklosti. V nadaljevanju se bomo posvetili razvoju algoritma učenja, ki bo agentu omogočal odločanje na podlagi že izvedenih akcij in dogodkov iz preteklosti. V našem navedenem primeru bi to pomenilo, da bi se po večkratnih iteracijah izvajanja opisa krožnice zmanjšala toleranca odstopanja in s tem trajektorija giba približala obliki krožnice. Druga smer, ki jo bomo v nadaljnjem delu upoštevali, je lastnost agenta, da sodeluje ter se z drugimi pogaja o nalogah za dosego skupnega cilja. Na primeru krožnega gibanja bi to pomenilo usklajeno delovanje vseh fizičnih agentov in s tem natančen opis krožnice.

Predstavljena rešitev agentne informacijske arhitekture se je izkazala za učinkovito, saj po prvih testiranjih zadošča potrebam informacijsko modelne podpore pri hitrih rekonfiguracijah realnega sveta oz. prilagajanj spremembam v okolju.

6 Zahvala

Raziskovalno delo, opisano v članku, sta podprla Tehnološka agencija Slovenije (P-MR-08/20) v sodelovanju z Republiko Slovenijo in Evropsko unijo, ter podjetje Trimo, d. d.

7 Literatura

[1] Yazen Al-Safi, Valeriy Vyatkin: An Ontology-Based Reconfiguration Agent for Intelligent Mechatronic Systems, Lecture Notes in Computer Science, Springer Berlin/ Heidelberg, Volume 4659/2007.

[2] B. Siciliano, O. Khatib: Springer Handbook of Robotics, 39. Distributed and Cellular Robots, Springer-Verlag Berlin Heidelberg 2008.

[3] Robert W. Brennan, Martyn Fletcher, Douglas H. Norrie: An Agent-Based Approach to Reconfiguration of Real-Time Distributed Control Systems, IEEE transactions on robotics and automation, Vol. 18, No. 4, August 2002, str. 444–451.

[4] S. Bussmann, N. R. Jennings, M. Wooldridge: Multiagent systems for manufacturing control, Springer-Verlag Berlin Heidelberg 2004.

[5] J. Ferber: Multi-agent sytems – An introduction to distributed artificial intelligence, Pearson Education 2005.

[6] G. Weiss: Multiagent systems – A modern approach to distributed artificial intelligence, MIT Press Cambridge, MA, USA, 1999.

[7] Philippe Lucidarme: Distributed multi-agent architecture for humanoid robots, 3rd National Conference on "Control Architectures of Robots" Bourges, May 29-30, 2008, Presses Universitaires d'Orléans, str.67–75.

Janez Sluga je diplomiral na Fakulteti za elektrotehniko Univerze v Ljubljani v letu 2008 s področja elektronike in nadaljuje podiplomski študij na isti fakulteti. Trenutno opravlja delo mladega raziskovalca iz gospodarstva v podjetju Trimo, d. d. Njegovo razvojno in raziskovalno delo se nanaša na načrtovanje, implementacijo in vodenje večagentnih ter drugih analizno-krmilnih sistemov.

Viktor Zaletelj je diplomiral, magistriral in doktoriral na Fakulteti za strojništvo Univerze v Ljubljani v letih 2000, 2003 in 2008. Njegovo raziskovalno delo vključuje napredne proizvodne teorije in sisteme, modeliranje prilagodljivih proizvodnih sistemov, razvoj porazdeljenih informacijskih sistemov in dizajniranje pripadajočih procesnih sistemov. Je vodja oddelka razvoja sistemov v podjetju Trimo, d. d. in raziskovalec na Fakulteti za strojništvo.

Andrej Žemva je diplomiral, magistriral in doktoriral na Fakulteti za elektrotehniko Univerze v Ljubljani v letih 1989, 1993 in 1996. Njegova raziskovalna in razvojna dejavnost obsega načrtovanje digitalnih elektronskih vezij in sistemov, vgrajene sisteme ter sočasno načrtovanje strojne in programske opreme.