ELEKTROTEHNIŠKI VESTNIK 91(5): 229–237, 2024
IZVIRNI ZNANSTVENI ČLANEK

Modulacija in posploševanje kartezičnih trajektorij s kvaternionskimi dinamičnimi generatorji gibov

Aleš Ude

Institut Jožef Stefan, Odsek za avtomatiko, biokibernetiko in robotiko, Jamova 39, 1000 Ljubljana, Slovenija
Univerza v Ljubljani, Fakulteta za elektrotehniko, Tržaška 25, 1000 Ljubljana, Slovenija
E-pošta: ales.ude@ijs.si

Povzetek. Dinamični generatorji gibov (DGG) so učinkovita reprezentacija za zapis poljubnih robotskih trajektorij. Uporabimo jih lahko za predstavitev pozicijskih in orientacijskih trajektorij. Pri zapisu orientacijskih trajektorij se pojavi težava, da ne obstaja minimalna (tridimenzionalna) predstavitev orientacije brez singularnosti. V tem članku obravnavamo orientacijske dinamične generatorje gibov, ki temeljijo na neminimalnem zapisu orientacije s kvaternioni. Pri tem pokažemo, da s kvaternionsko formulacijo ohranimo vse prednosti dinamičnih generatorjev gibov, vključno z enostavnostim učenjem in posploševanjem, modulacijo gibov in robustnim odzivom na perturbacije.

Ključne besede: reprezentacija robotskih gibanj, modulacija trajektorij, statistično učenje

Modulation and generalization of Cartesian trajectories using quaternion-based dynamic movement primitives

Dynamic Movement Primitives (DMPs) are an effective framework for encoding smooth robotic trajectories, capable of representing both positional and orientational motion. However, specifying orientation trajectories can be problematic because minimal (three-dimensional) representations of orientation inherently suffer from singularities. In this paper, we explain quaternion-based DMPs, which offer a singularity-free, non-minimal representation of orientation. We demonstrate that using the quaternion formulation retains all the advantages of standard DMPs, including ease of learning and generalization, motion modulation, and robust response to perturbations.

1 UVOD

Ekplicitna odvisnost robotskih trajektorij od časa je lahko problematična pri generaciji želenih robotskih gibov. Če na primer robotsko roko med izvedbo gibanja prime človek, s katerim robot sodeluje, se bo robotsko gibanje začasno ustavilo. Ker med zaustavitvijo čas teče naprej, se želena robotska trajektorija vse bolj odmika od trenutnega položaja robota. Ko človeški sodelavec spusti robota, bo klasični robotski regulator, na primer PD-regulator, poskusil razdaljo med dejanskim in želenim položajem robota čim hitreje zmanjšati. To lahko pripelje do sunkovitih ali celo nevarnih robotskih gibov. Da bi rešili ta problem, so Ijspeert et al. [1], [2] predlagali reprezentacijo robotskih trajektorij z dinamičnimi generatorji gibov (DGG, angl. dynamic movement primitives). DGG-ji so sestavljeni iz linearnega dinamičnega sistema drugega reda z enolično privlačno točko. Temu sistemu dodamo nelinearni člen, ki omogoča prilagoditev enostavne dinamike drugega reda specifičnim robotskim gibanjem. DGG-je uporabljamo za predstavitev periodičnih in diskretnih robotskih gibov in imajo številne za robotiko ugodne lastnosti. Vsebujejo namreč proste parametre, ki jih lahko uporabimo za robotsko učenje, ne da bi pri tem vplivali na konvergenco in stabilnost dinamičnega sistema. So tudi odporni proti perturbacijam in jih lahko enostavno prilagajamo končnemu cilju in oziroma želeni hitrosti robotskega gibanja.

Načrtovanje orientacijskih trajektorij je v splošnem bolj zapleteno kot načrtovanje pozicijskih trajektorij. Medtem ko lahko položaj robota predstavimo s 3D vektorjem, orientacije ni mogoče predstaviti na ta način, saj množica vseh orientacij $\mathrm{SO}(3)$ ni vektorski prostor. $\mathrm{SO}(3)$ je grupa in realna tridimenzionalna mnogoterost [3]. $\mathrm{SO}(3)$ sicer lahko parametriziramo s tremi parametri, npr. Eulerjevimi koti, vendar pa takšne parametrizacije vedno vsebujejo singularnosti. Izkazaže se, da ne obstaja minimalna (3D) reprezentacija orientacije, ki 1. ne vsebuje singularnosti, tj. vsako zvezno orientacijsko gibanje v kartezičnem prostoru je zvezno tudi v 3D parametričnem prostoru, in 2. je zvezno odvedljiva, tj. parcialni odvodi parametrov glede na vsako diferencialno rotacijo v kateri koli orientaciji so končni.

Za reprezentacijo robotskih gibanj zato uporabljamo predvsem parametrizacije brez nezveznosti, ki ne obstajajo v resničnem svetu. Glede na zgornjo diskusiji so takšne predstavitve orientacij vedno več kot tridimenzionalne in morajo zato izpolnjevati dodatne omejitve v večdimenzionalnih prostorih. Le tako lahko zagotovimo, da želena orientacijska trajektorija pripada tridimenzionalni mnogoterosti v večdimenzionalnem prostoru. Te omejitve pa lahko povzročajo težave pri interpola-

Prejet 15. julij, 2024
Odobren 30. avgust, 2024

ciji v $\mathrm{SO}(3)$, saj standardne interpolacijske metode ne upoštevajo strukture $\mathrm{SO}(3)$ in zato lahko povzročijo, da interpolirani parametri ne ležijo v $\mathrm{SO}(3)$.

V tem prispevku najprej uvedemo dinamične generatorje gibov in nato opišemo predstavitev orientacijskih trajektorij z dinamičnimi generatorji gibov v prostoru enotnih kvaternionov. Pri generaciji trajektorij s kvaternionskimi DGG-ji zagotovimo, da integrirani enotni kvaternioni ostanejo v $\mathrm{SO}(3)$. Nato pokažemo, da orientacijski DGG-ji prav tako kot klasični DGG-ji omogočajo različne prilagoditve robotskih gibanj. Prispevek zaključimo z novim pristopom za posploševanje orientacijskih trajektorij, ki jih opišemo z DGG-ji.

2 DINAMIČNI GENERATORJI GIBOV

Ijspeert et al. [1], [2] so predlagali dinamične generatorje gibov (DGG) kot učinkovito reprezentacijo za zapis želenih robotskih gibanj. DGG-ji temeljijo na sistemu diferencialnih enačb drugega reda, katerega stabilnost je matematično zagotovljena. Trajektorije zapisane z DGG-ji imajo številne dobre lastnosti za robotska gibanja, kot je npr. konvergenca k želeni končni točki. Sistem diferencialnih enačb vsebuje tudi nelinearni člen s prostimi parametri, ki omogočajo zapis poljubnih robotskih gibov. DGG za spremembo položaja v tridimenzionalnem kartezičnem prostoru vzdolž želene trajektorije (premiki od točke do točke) definiramo z naslednjim sistemom nelinearnih diferencialnih enačb [1]

$$\tau\dot{\mathbf{z}} = \alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}) - \mathbf{z}) + \mathbf{f}_p(x), \quad (1)$$

$$\tau\dot{\mathbf{p}} = \mathbf{z}, \quad (2)$$

kjer $\mathbf{p}$, $\mathbf{z} \in \mathbb{R}^3$ označujeta želeno pozicijsko trajektorijo oziroma skalirano hitrost robota, $\mathbf{g}_p \in \mathbb{R}^3$ je želena končna pozicija robota, $\mathbf{f}_p \colon \mathbb{R} \longmapsto \mathbb{R}^3$ pa je nelinearna funkcija s prostimi parametri. $\tau > 0$ je časovna konstanta, ki omogoča spreminjanje hitrosti robota. $x \in \mathbb{R}$ je fazna spremenljivka, ki zagotavlja avtonomost diferencialne enačbe (1), to je njeno posredno odvisnost od časa. Potek faze sledi linearni enačbi prvega reda

$$\tau\dot{x} = -\alpha_x x, \quad (3)$$

kjer je $\alpha_x > 0$ in x(0) = 1. Funkcija $\mathbf{f}_p(x)$ je definirana kot linearna vsota N nelinearnih radialnih baznih funkcij, s čimer omogočimo aproksimacijo poljubnih gladkih trajektorij od začetne točke $\mathbf{p}_0$ do končne točke $\mathbf{g}_p$

$$\mathbf{f}_p(x) = \mathbf{D}_p \frac{\sum_{i=1}^{N} \mathbf{w}_i^p \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x, \quad (4)$$

$$\Psi_i(x) = \exp\left(-h_i\,(x - c_i)^2\right). \quad (5)$$

$\mathbf{D}_p = \mathrm{diag}\,(\mathbf{g}_p - \mathbf{p}_0) \in \mathbb{R}^{3\times 3}$ je diagonalna matrika, ki omogoča skaliranje amplitude želenega giba, če se spremeni končni položaj $\mathbf{g}_p$. V zgornjih enačbah so $c_i \in \mathbb{R}$ središča radialnih baznih funkcij vzdolž faze giba x, parametri $h_i \in \mathbb{R}$ pa definirajo širino baznih funkcij. Pri znanem številu baznih funkcij N in če je časovna konstanta τ enaka trajanju želenega giba, lahko definiramo konstante $c_i = \exp\left(-\alpha_x \frac{i-1}{N-1}\right)$, $h_i = \dfrac{1}{(c_{i+1} - c_i)^2}$, $h_N = h_{N-1}$, $i = 1, \ldots, N$. Skalarji $\alpha_z, \beta_z, \alpha_x > 0$ so konstante, ki določajo togost sistema diferencialnih enačb (1) – (3). Njihove vrednosti običajno določi operater. Z izbiro $\alpha_z = 4\beta_z$ postane linearni del sistema diferencialnih enačb (1) – (3), to je sistem brez nelinearnega člena $\mathbf{f}_p$, kritično dušen, pri čemer $\mathbf{p}$ in $\mathbf{z}$ monotono konvergirata proti končni točki $\mathbf{p} = \mathbf{g}_p$, $\mathbf{z} = 0$ [1] in sicer ne glede na začetno točko.

Uteži $\mathbf{w}_i \in \mathbb{R}^3$ izračunamo tako, da integrirana pozicijska trajektorija $\mathbf{p}$ generira želene položaje robota. Če je na primer dano zaporedje želenih položajev, hitrosti in pospeškov $\{\mathbf{p}_j,\, \dot{\mathbf{p}}_j,\, \ddot{\mathbf{p}}_j\}_{j=0}^{T}$ ob časih $\{t_j\}_{j=0}^{T}$, lahko izračunamo uteži s pomočjo linearnega sistema enačb. V ta namen najprej prepišemo sistem dveh diferencialnih enačb prvega reda (1), (2) v eno enačbo drugega reda

$$\tau^2\ddot{\mathbf{p}} = \alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}) - \tau\dot{\mathbf{p}}) + \mathbf{f}_p(x) \quad (6)$$

in vanjo vstavimo želene vrednosti $\mathbf{p}_j$, $\dot{\mathbf{p}}_j$, $\ddot{\mathbf{p}}_j$ ter izpišemo $\mathbf{f}_p$

$$\frac{\sum_{i=1}^{N} \mathbf{w}_i^p \Psi_i(x_j)}{\sum_{i=1}^{N} \Psi_i(x_j)} x_j = \mathbf{D}_p^{-1}\left(\tau^2\,\ddot{\mathbf{p}}_j + \alpha_z\tau\dot{\mathbf{p}}_j - \alpha_z\beta_z\,(\mathbf{g}_p - \mathbf{p}_j)\right), \quad (7)$$

kjer uporabimo za izračun faz x_j analitično rešitev sistema (3)

$$x_j = \exp(-\alpha_x/\tau\, t_j). \quad (8)$$

Da dobimo želeno trajektorijo za vodenje robota, moramo integrirati sistem diferencialnih enačb (1) – (3) pri danih začetnih vrednostih $\mathbf{p}(0) = \mathbf{p}_0$, $\mathbf{z}(0) = \tau\mathbf{v}_0$ in $x(0) = 1$, kjer $\mathbf{p}_0$ in $\mathbf{v}_0$ označujeta začetni položaj oziroma hitrost robota. V ta namen običajno uporabimo Eulerjevo integracijo

$$\mathbf{p}(t + \Delta t) = \mathbf{p}(t) + \frac{1}{\tau}\mathbf{z}(t)\Delta t, \quad (9)$$

$$\mathbf{z}(t + \Delta t) = \mathbf{z}(t) + \frac{1}{\tau}(\alpha_z(\beta_z(\mathbf{g}_p - \mathbf{p}(t)) - \mathbf{z}(t)) + \mathbf{f}_p(x(t)))\Delta t, \quad (10)$$

$$x(t + \Delta t) = x(t) - \frac{1}{\tau}\alpha_x x(t)\Delta t. \quad (11)$$

2.1 *Kvaternionski dinamični generatorji gibov*

Kot že omenjeno minimalna (tridimenzionalna) reprezentacija orientacij brez singularosti ne obstaja [3]. Zato za zapis gladkih orientacijskih trajektorij ponavadi uporabljamo neminimalne reprezentacije, ki ne vsebujejo singularnosti in zagotavljajo zveznost, če je le orientacijska trajektorija v resničnem svetu zvezna. Tovrstno reprezentacijo s štirimi parametri nudijo enotni kvaternioni $\mathbf{q} = v + \mathbf{u} \in \mathrm{S}^3$, kjer z S^3 označimo enotno sfero v $\mathbb{R}^4$, $v \in \mathbb{R}$, $\mathbf{u} \in \mathbb{R}^3$. Enotni kvaternioni so

bili uporabljeni za reprezentacijo orientacij v različnih kontekstih, npr. za robotsko vodenje [4], modeliranje trajektorij [5], [6] in zasledovanje objektov v robotskem vidu [7], [8]. Ta reprezentacija ni enolična saj enotna kvaterniona $\mathbf{q}$ in $-\mathbf{q}$ predstavljata isto orientacijo. Pozicijske dinamične generatorje gibov (1), (2) lahko transformiramo v kvaternionske na naslednji način

$$\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z 2\log\left(\mathbf{g}_o * \overline{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_o(x), \quad (12)$$

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\eta} * \mathbf{q}, \quad (13)$$

kjer z $\mathbf{g}_o \in \mathrm{S}^3$ označimo ciljno orientacijo, strešica označuje konjugirani kvaternion $\overline{\mathbf{q}} = \overline{v + \mathbf{u}} = v - \mathbf{u}$, z $*$ zvezdico pa označimo kvaternionski produkt

$$\begin{aligned}\mathbf{q}_1 * \mathbf{q}_2 &= (v_1 + \mathbf{u}_1) * (v_2 + \mathbf{u}_2) \quad (14)\\ &= (v_1 v_2 - \mathbf{u}_1^{\mathrm{T}}\mathbf{u}_2) + (v_1\mathbf{u}_2 + v_2\mathbf{u}_1 + \mathbf{u}_1 \times \mathbf{u}_2).\end{aligned}$$

Vektor $\boldsymbol{\eta} \in \mathbb{R}^3$ v enačbi (13) obravnavamo kot kvaternion z ničelnim skalarnim delom. Kvaternionski logaritem $\log\colon \mathrm{S}^3 \longmapsto \mathbb{R}^3$ definiramo z naslednjo enačbo [7]

$$\log(\mathbf{q}) = \log(v + \mathbf{u}) = \begin{cases} \arccos(v)\dfrac{\mathbf{u}}{\|\mathbf{u}\|}, & \mathbf{u} \neq 0 \\ [0,0,0]^{\mathrm{T}}, & \text{sicer} \end{cases}. \quad (15)$$

Enačbi (12) in (13) temeljita na naslednjih ugotovitvah: 1. da je rotacija, ki preslika enotni kvaternion $\mathbf{q}_1$ v $\mathbf{q}_2$ določena z enačbo

$$\Delta\mathbf{q} = \mathbf{q}_2 * \overline{\mathbf{q}}_1, \quad (16)$$

2. da lahko kotno hitrost $\tilde{\boldsymbol{\omega}}$, s katero zarotiramo enotni kvaternion $\mathbf{q}_1$ v $\mathbf{q}_2$ v eni časovni enoti, izračunamo s pomočjo enčbe

$$\tilde{\boldsymbol{\omega}} = 2\,\log\left(\mathbf{q}_2 * \overline{\mathbf{q}}_1\right) = 2\,\log\left(\Delta\mathbf{q}\right), \quad (17)$$

in 3. da odvod enotnega kvaterniona in pripadajočo kotno hitrost rotacijskega gibanja $\boldsymbol{\omega}$ povezuje enačba

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\omega} * \mathbf{q}. \quad (18)$$

Kvaternionski dinamični generator gibov torej definiramo z enačbami (12), (13) in (3).

Nelinearni člen

$$\mathbf{f}_o = \mathbf{D}_o \frac{\sum_{i=1}^{N}\mathbf{w}_i^o\Psi_i(x)}{\sum_{i=1}^{N}\Psi_i(x)}x, \quad (19)$$

definiramo na enak način kot v (4). Vsebuje proste parametre $\mathbf{w}_i^o \in \mathbb{R}^3$, ki jih lahko uporabimo za gladko aproksimacijo poljubne izmerjene orientacijske trajektorije $\{\mathbf{q}_j, \boldsymbol{\omega}_j, \dot{\boldsymbol{\omega}}_j, t_j\}_{j=0}^{T}$. $\mathbf{D}_o = \mathrm{diag}\left(2\log(\mathbf{g}_o * \overline{\mathbf{q}}_0)\right) \in \mathbb{R}^{3\times 3}$ je diagonalna matrika, ki omogoča skaliranje amplitude želenega giba, če se spremeni končni položaj $\mathbf{g}_o \in \mathbb{R}^3$. Parametre $\mathbf{w}_i^o$ lahko torej izračunamo tako, da rešimo naslednji sistem linearnih enačb

$$\begin{aligned}&\frac{\sum_{i=1}^{N}\mathbf{w}_i^o\Psi_i(x_j)}{\sum_{i=1}^{N}\Psi_i(x_j)}x_j = \\ &\mathbf{D}_o^{-1}\left(\tau^2\dot{\boldsymbol{\omega}}_j - \alpha_z\left(\beta_z 2\log\left(\mathbf{g}_o * \overline{\mathbf{q}}_j\right) - \tau\boldsymbol{\omega}_j\right)\right),\end{aligned} \quad (20)$$

pri čemer faze x_j izračunamo na enak način kot v enačbi (8). Iz enačb (13) in (18) pa dobimo $\boldsymbol{\eta}_j = \tau\boldsymbol{\omega}_j$, $\dot{\boldsymbol{\eta}}_j = \tau\dot{\boldsymbol{\omega}}_j$.

Reprezentacija orientacij s kvaternioni vsebuje le eno singularnost in sicer pri enotnem kvaternionu $\mathbf{q} = -1 + [0,0,0]^{\mathrm{T}}$, kjer rotacijska os ni določena. Temu kvaternionu se lahko v praksi vedno izognemo in ne povzroča težav pri aplikacijah DGG-jev. Z logaritemsko preslikavo lahko definiramo metriko na S^3 [7] in sicer

$$\mathrm{d}(\mathbf{q}_1, \mathbf{q}_2) = \begin{cases} 2\pi, & \mathbf{q}_1 * \overline{\mathbf{q}}_2 = -1 + [0,0,0]^{\mathrm{T}} \\ 2\|\log(\mathbf{q}_1 * \overline{\mathbf{q}}_2)\|, & \text{sicer} \end{cases}. \quad (21)$$

Pri tem je treba omeniti, da d ni metrika na $\mathrm{SO}(3)$ saj $\mathrm{d}(\mathbf{q}, -\mathbf{q}) = 2\pi$ čeprav $\mathbf{q}$ in $-\mathbf{q}$ predstavljata isto orientacijo.

V nasprotju z našim pristopom so Pastor et al. [9] v enačbi (12) uporabili samo vektorski del kvaternionskega produkta $v + \mathbf{u} = \mathbf{g}_o * \overline{\mathbf{q}}$, to je $\mathbf{u} \in \mathbb{R}^3$ namesto $2\log\left(\mathbf{g}_o * \overline{\mathbf{q}}\right)$. Na podoben način so razliko med dvema orientacijama obravnavali tudi v [10]. Če definiramo $\mathbf{u} = \mathrm{vec}\left(\mathbf{g}_o * \overline{\mathbf{q}}\right)$, lahko njihov sistem zapišemo na naslednji način

$$\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z \mathrm{vec}\left(\mathbf{g}_o * \overline{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_o(x). \quad (22)$$

Zgornja enačba je sicer ekvivalentna enačbi (12), kar zadeva smer spremembe orientacije, vendar pa enačba (22) ne upošteva geometrije prostora $\mathrm{SO}(3)$. Kotno hitrost moramo namreč izračunati po enačbi (17). To pomeni, da lahko kvaternionski produkt $\mathbf{g}_o * \overline{\mathbf{q}}$ preslikamo v kotno hitrost le z logaritmom, pomnoženim z 2. Razliko med obema pristopoma bolj podrobno analiziramo v razdelku 5.

Za integracijo enačbe (13) lahko uporabimo formulo

$$\mathbf{q}(t + \Delta t) = \exp\left(\frac{\Delta t}{2}\frac{\boldsymbol{\eta}(t)}{\tau}\right) * \mathbf{q}(t), \quad (23)$$

kjer je kvaternionska eksponentna funkcija definirana kot inverz logaritma in jo izračunamo po enačbi [7]

$$\exp(\mathbf{r}) = \begin{cases} \cos\left(\|\mathbf{r}\|\right) + \sin\left(\|\mathbf{r}\|\right)\dfrac{\mathbf{r}}{\|\mathbf{r}\|}, & \mathbf{r} \neq 0 \\ 1 + [0,0,0]^{\mathrm{T}}, & \text{otherwise} \end{cases} \quad (24)$$

Če omejimo definicijsko območje eksponentne preslikave $\exp\colon \mathbb{R}^3 \longmapsto \mathrm{S}^3$ na $\{\mathbf{r}, \|\mathbf{r}\| < \pi\}$ in logaritemsko preslikavo na $\mathrm{S}^3/(-1 + [0,0,0]^{\mathrm{T}})$, potem sta obe preslikavi bijektivni, zvezno odvedljivi in inverzni druga drugi. Parametra $\boldsymbol{\eta}$ in x integriramo na enak način kot njuna ekvivalenta v pozicijskih DGG-jih.

3 MODULACIJA KARTEZIČNIH DGG-JEV

Ena izmed glavnih prednosti DGG-jev je, da lahko z enostavnimi spremembami v sistemu diferencialnih enačb hitro prilagajamo želene orientacijske trajektorije. V nadaljevanju tega razdelka obravnavamo dva tipa

prilagoditev, in sicer zaustavitev faze in spremembo ciljne orientacije med izvedbo kvaternionskih DGG-jev. Modificirane enačbe uporabljamo le med izvedbo gibanja. Pri specifikaciji DGG-jev, to je pri računanju uteži $\mathbf{w}_i^o$, moramo uporabiti standardne DGG enačbe.

3.1 Ustavljanje faze

Razlog za definicijo faze x z diferencialno enačbo (3) je, da lahko potek faze spreminjamo z modifikacijo enačbe (3). Izkaže se, da je takšen način precej bolj enostaven kot spreminjanje eksplicitnih časovnih reprezentacij [2]. Za ustavljanje faze [11] zamenjamo originalno enčbo (3) z

$$\tau\dot{x} = -\frac{\alpha_x x}{1+\alpha_{px}\left(\|\tilde{\mathbf{p}}-\mathbf{p}\|+\gamma\,\mathrm{d}(\tilde{\mathbf{q}},\mathbf{q})\right)}, \qquad (25)$$

kjer s $\tilde{\mathbf{p}}$ in $\tilde{\mathbf{q}}$ označimo dejanski položaj in orientacijo vrha robota, medtem ko $\mathbf{p}$ in $\mathbf{q}$ predstavljata želeno trajektorijo izračunano z integracijo DGG-ja. Napako pri sledenju želene trajektorije lahko potem izračunamo z izrazom $\|\tilde{\mathbf{p}}-\mathbf{p}\|+\gamma\,\mathrm{d}(\tilde{\mathbf{q}},\mathbf{q})$. Če postane napaka pri sledenju želene trajektorije velika, tj. $\|\tilde{\mathbf{p}}-\mathbf{p}\|+\gamma\,\mathrm{d}(\tilde{\mathbf{q}},\mathbf{q})$ ima visoko vrednost, pride do zmanjšane hitrosti $\dot{x}$ in počasnega spreminjanja faze. S tem zaustavimo napredovanje faze, dokler robot ne zmanjša napake pri sledenju. Poleg tega je koristno modificirati tudi enačbo (2) in s tem zagotoviti hitrejše zmanjševanje napake [1]. Pri kartezičnih DGG-jih moramo poleg enačbe (2) spremeniti tudi (13). Obe enačbi spremenimo v

$$\tau\dot{\mathbf{p}} = \mathbf{z}+\alpha_{pp}(\mathbf{p}-\tilde{\mathbf{p}}), \qquad (26)$$

$$\tau\dot{\mathbf{q}} = \frac{1}{2}\left(\boldsymbol{\eta}+\alpha_{pr}2\log\left(\mathbf{q}*\overline{\tilde{\mathbf{q}}}\right)\right)*\mathbf{q}. \qquad (27)$$

Namesto originalnih enačb DGG moramo torej med izvedbo robotskega gibanja integrirati enačbi (26) in (27), pri čemer za integracijo enačbe (27) uporabimo podobno formulo kot pri integraciji enačbe (23).

3.2 Spreminjanje končnega cilja gibanja

Če med izvedbo robotskega giba spremenimo ciljno orientacijo $\mathbf{g}_o$ v neko drugo orientacijo, ta sprememba povzroči nezveznost v členu $\log(\mathbf{g}_o*\overline{\mathbf{q}})$ iz enačbe (12) in posledično nezveznost v kotni hitrostii $\dot{\boldsymbol{\omega}}$. Da se nezveznosti izognemo, je za klasične DGG-je predlagana uvedba dodatnega člena v sistemu diferencialnih enačb (1), (2) [1]. Njegov namen je zagotoviti zvezno spreminjanje cilja trajektorije $\mathbf{g}_p$ proti novemu cilju $\hat{\mathbf{g}}_p$

$$\tau\dot{\mathbf{g}}_p = \alpha_{gp}(\hat{\mathbf{g}}_p-\mathbf{g}_p). \qquad (28)$$

Pri tem smo nekdanjo konstanto $\mathbf{g}_p$ spremenili v zvezno spremenljivko in dodali diferencialno enačbo, saj moramo zdaj integrirati (1) – (3) in (28). Tudi če se $\hat{\mathbf{g}}_p$ nenadoma spremeni, npr. zaradi novih senzoričnih informacij, $\mathbf{g}_p$ ostane zvezen in postopoma konvergira proti novemu cilju $\hat{\mathbf{g}}_p$. Zato v želeni trajektoriji kljub spremembi ne pride do nezveznosti. V kontekstu kvaternionskih DGG-jev se enačba (28) spremeni v

$$\tau\dot{\mathbf{g}}_o = \alpha_{go}2\log(\hat{\mathbf{g}}_o*\overline{\mathbf{g}}_o)*\mathbf{g}_o. \qquad (29)$$

Podobno kot $\mathbf{g}_p$ v (28) je tudi $\mathbf{g}_o$ zdaj spremenljivka, ki ob nenadni spremembi cilja orientacije $\hat{\mathbf{g}}_o$ zvezno preide v novo ciljno orientacijo. Enačbo (29) moramo integrirati skupaj z enačbama (12) in (13), pri čemer izhajamo iz formule (23).

4 POSPLOŠEVANJE KARTEZIČNIH DGG-JEV

V nadaljevanju predpostavimo, da na voljo nimamo samo ene, temveč več vzorčnih kartezičnih trajektorij

$$\mathcal{B} = \{\{\mathbf{p}_{j,k},\,\dot{\mathbf{p}}_{j,k},\,\ddot{\mathbf{p}}_{j,k},\,\mathbf{q}_{j,k},\,\boldsymbol{\omega}_{j,k},\,\dot{\boldsymbol{\omega}}_{j,k},\,t_{j,k}\}_{j=0}^{T_k};\,\mathbf{b}_k\}_{k=1}^{M} \qquad (30)$$

kjer so $\mathbf{p}_{j,k}$, $\dot{\mathbf{p}}_{j,k}$, $\ddot{\mathbf{p}}_{j,k}$ in $\mathbf{q}_{j,k}$, $\boldsymbol{\omega}_{j,k}$, $\dot{\boldsymbol{\omega}}_{j,k}$ pozicije, hitrosti in pospeški oziroma orientacije, kotne hitrosti in kotni pospeški na k-ti trajektoriji ob časih $t_{j,k}$. T_k je število meritev na k-ti trajektoriji in M število vzorčnih kartezičnih trajektorij. $\mathbf{b}_k \in \mathbb{R}^n$ so parametri, ki karakterizirajo želeno nalogo v neki dani situaciji. Ti parametri so običajno povezani z namenom želene naloge. Pri našem pristopu jih uporabljamo kot poizvedbeno točko za iskanje podatkov v bazi vzorčnih trajektorij. Podobno kot v primerih iz razdelka 2, lahko tovrstne podatke pridobimo s pomočjo človeških demonstracij različnih variant želene naloge.

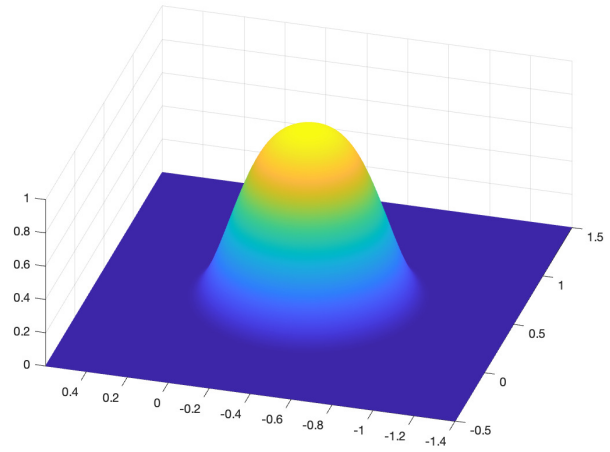
Naš cilj v tem razdelku je, kako pri dani bazi vzorčnih trajektorij izračunati DGG, ki specificira robotsko gibanje za poljubno poizvedbeno točko $\mathbf{b}$, ki v splošnem ni enaka kateri od poizvedbenih točk $\mathbf{b}_k$ iz baze $\mathcal{B}$. Kartezični DGG je enolično določen s prostimi parametri $\mathbf{w}_p = [\mathbf{w}_1^p,\ldots,\mathbf{w}_N^p] \in \mathbb{R}^{3\times N}$, $\mathbf{w}_o = [\mathbf{w}_1^o,\ldots,\mathbf{w}_N^o] \in \mathbb{R}^{3\times N}$, τ, $\mathbf{g}_p$ in $\mathbf{g}_o$. Reševanje zgoraj opisanega problema lahko torej formuliramo kot učenje funkcije

$$\mathbf{G}\,(\mathcal{B})\colon \mathbf{b} \longmapsto \{\mathbf{w}_p,\mathbf{w}_o,\tau,\mathbf{g}_p,\mathbf{g}_o\}, \qquad (31)$$

kjer vzorčne trajektorije s pripadajočimi poizvedbenimi točkami $\mathcal{B}$ iz (30) predstavljajo podatke, ki jih lahko uporabimo za učenje.

Kot primer si poglejmo nalogo metanja žoge na koš. V tem primeru so vzorčne trajektorije primeri metov na koš, poizvedbene točke pa so podatki o položaju koša v prostoru. Funkcija $\mathbf{G}\,(\mathcal{B})$ torej v tem primeru preslika poljubne položaje koša v prostoru v parametre DGG, ki določajo robotski met, s katerim robot žogo vrže na koš pri danem položaju koša. Kot lahko vidimo na tem primeru, izhajajo poizvedbene točke iz intuitivne karakterizacije naloge, vendar pa funkcionalno razmerje med poizvedbenimi točkami in parametri DGG po navadi ni enostavno.

Pri učenju iz večjih baz vzorčnih trajektorij niso vsi podatki enako relevantni za posploševanje. Na primer, pri metanju žoge so za generacijo novega meta na koš bolj relevantne tiste trajektorije, ki so povezane s položaji koša bližjimi trenutnemu košu, kot pa trajektorije, povezane z meti na koše, ki so bolj oddaljeni od trenutnega koša.



Slika 1: Trikubično jedro za lokalno uteženo regresijo z dvodimenionalno poizvedovalno točko $\mathbf{b} = [0.5, -0.4]^T$ in $h = 0.6$.

Računanje parametrov DGG s pomočjo funkcije (31) temelji na potencialno velikem naboru podatkov (30). Zato je pri računanju funkcije $\mathbf{G}(\mathcal{B})$ smiselno uporabiti lokalne metode z relativno nizko računsko zahtevnostjo [12], [13]. Med takšne metode spada na primer lokalno utežena regresija [14], ki da večji pomen podatkom blizu poizvedbene točke in pri računanju ne upošteva bolj oddaljenih poizvedbenih točk. Za izračun uteži $\mathbf{w}_p$ in $\mathbf{w}_o$ pri dani poizvedbeni točki $\mathbf{b}$ z lokalno uteženo regresijo moramo rešiti naslednji problem najmanjših kvadratov

$$\min_{\mathbf{w}_p, \mathbf{w}_o} = \sum_{k=1}^{M} \left\| \mathbf{\Psi}_k [\mathbf{w}_p^T, \mathbf{w}_o^T] - [\mathbf{P}_k, \mathbf{Q}_k] \right\|^2 \mathrm{K}(\mathbf{b}, \mathbf{b}_k), \tag{32}$$

kjer je

$$\begin{aligned}
\mathbf{P}_k &= [\tilde{\mathbf{p}}_{1,k}, \ldots, \tilde{\mathbf{p}}_{T_k,k}]^T \in \mathbb{R}^{T_k \times 3}, \\
\tilde{\mathbf{p}}_{j,k} &= \mathbf{D}_{p,k}^{-1} \left(\tau_k^2 \ddot{\mathbf{p}}_{j,k} - \alpha_z (\beta_z (\mathbf{g}_{p,k} - \mathbf{p}_{j,k}) - \tau_k \dot{\mathbf{p}}_{j,k}) \right), \\
\mathbf{Q}_k &= [\tilde{\mathbf{q}}_{1,k}, \ldots, \tilde{\mathbf{q}}_{T_k,k}]^T \in \mathbb{R}^{T_k \times 3}, \\
\tilde{\mathbf{q}}_{j,k} &= \mathbf{D}_{o,k}^{-1} \left(\tau_k^2 \dot{\boldsymbol{\omega}}_{j,k} - \alpha_z (\beta_z 2 \log(\mathbf{g}_{o,k} * \overline{\mathbf{q}}_{j,k}) - \tau_k \boldsymbol{\omega}_{j,k}) \right)
\end{aligned}$$

in

$$\mathbf{\Psi}_k = \begin{bmatrix} \frac{\Psi_1(x_{1,k}) x_{1,k}}{\sum_{i=1}^N \Psi_i(x_{1,k})} & \cdots & \frac{\Psi_N(x_{1,k}) x_{1,k}}{\sum_{i=1}^N \Psi_i(x_{1,k})} \\ \vdots & \ddots & \vdots \\ \frac{\Psi_1(x_{T_k,k}) x_{T_k,k}}{\sum_{i=1}^N \Psi_i(x_{T_k,k})} & \cdots & \frac{\Psi_N(x_{T_K,k}) x_{T_k,k}}{\sum_{i=1}^N \Psi_i(x_{T_k,k})} \end{bmatrix}, \tag{33}$$

$\mathbf{\Psi}_k \in \mathbb{R}^{T_k \times N}$, $x_{j,k} = \exp(-\alpha_x / \tau_k \, t_{j,k})$. Utežitveno funkcijo K lahko definiramo na več načinov. Ena od možnosti je trikubično jedro [14] (glej sliko 1), ki je do drugega reda zvezno odvedljivo in katerega zaloga vrednosti je enaka nič zunaj intervala $[-h, h]$

$$\mathrm{K}(\mathbf{b}, \mathbf{b}_k) = \begin{cases} (1 - (\|\mathbf{b} - \mathbf{b}_k\|/h)^3)^3, \|\mathbf{b} - \mathbf{b}_k\|/h \leq 1 \\ 0, \text{ sicer} \end{cases}. \tag{34}$$

Ker so časovne konstante τ_k in ciljni položaji $\mathbf{g}_k^p$ ter ciljne orientacije $\mathbf{g}_k^o$ merjeni neposredno, tj. $\tau_k = T_{t,k}$, $\mathbf{g}_k^p = \mathbf{p}_{T_k,k}$ in $\mathbf{g}_k^o = \mathbf{q}_{T_k,k}$, je njihovo posploševanje z lokalno uteženo regresijo bistveno lažje

$$\tau = \frac{\sum_{k=1}^N \mathrm{K}(\mathbf{b}, \mathbf{b}_k) \tau_k}{\sum_{k=1}^N \mathrm{K}(\mathbf{b}, \mathbf{b}_k)}, \tag{35}$$

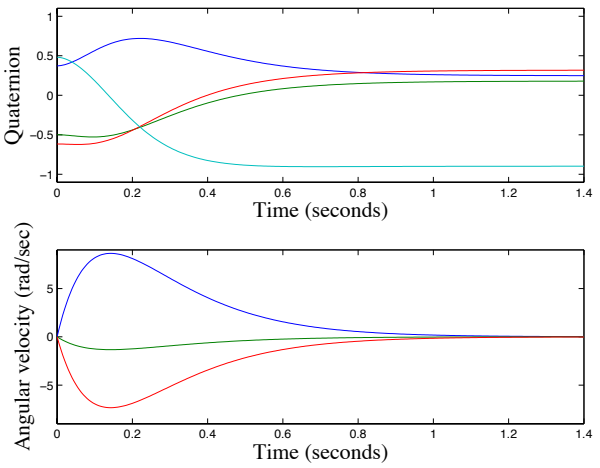
$$\mathbf{g}_p = \frac{\sum_{k=1}^N \mathrm{K}(\mathbf{b}, \mathbf{b}_k) \mathbf{g}_{p,k}}{\sum_{k=1}^N \mathrm{K}(\mathbf{b}, \mathbf{b}_k)}, \tag{36}$$

$$\mathbf{g}_o = \frac{\tilde{\mathbf{g}}_o}{\|\tilde{\mathbf{g}}_o\|}, \; \tilde{\mathbf{g}}_o = \frac{\sum_{k=1}^N \mathrm{K}(\mathbf{b}, \mathbf{b}_k) \mathbf{g}_{o,k}}{\sum_{k=1}^N \mathrm{K}(\mathbf{b}, \mathbf{b}_k)}. \tag{37}$$

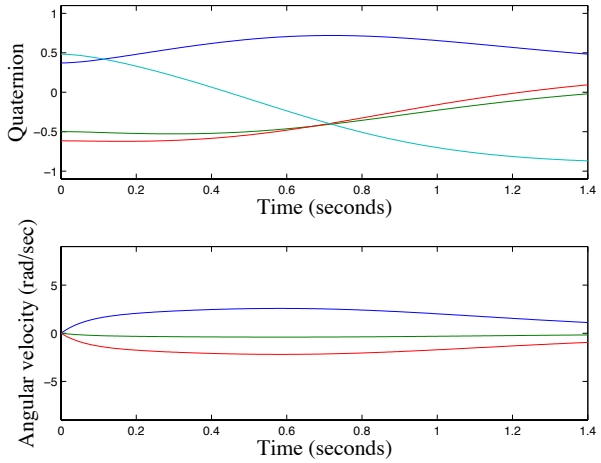
Prednost predlagane metodologije posploševanja z lokalno uteženo regresijo je, da za izračun funkcije (31) ni treba rešiti nobenega globalnega optimizacijskega problema. Učenje je sestavljeno iz preprostega dodajanja vzorčnih trajektorij v bazo (30). Do računanja pride šele, ko dobimo novo poizvedbeno točko $\mathbf{b}$, rešimo problem najmanjših kvadratov (32) ter izračunamo (35), (36) in (37). Tako izračunamo lokalni model za funkcijo (31), ki je veljaven le za poizvedbeno točko $\mathbf{b}$. Metode, ki izračunajo globalno veljavne diferencialne enačbe za vodenje robotov iz množice vzorčnih trajektorij [15], temeljijo na reševanju nelinearnih optimizacijskih problemov in so računsko bistveno bolj zahtevne.

5 EKSPERIMENTALNI REZULTATI

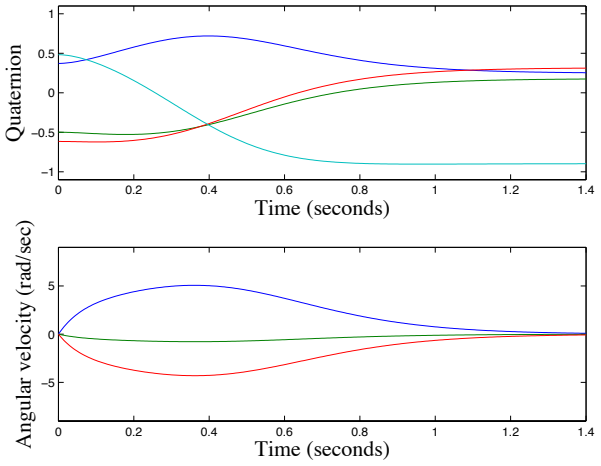
Analizo začenjamo s primerjavo predlaganih DGG-jev s pristopom iz [9], kjer so avtorji kot v enačbi (22) za razliko dveh kvaternionov uporabili vektor $\mathrm{vec}(\mathbf{g}_o * \overline{\mathbf{q}})$ namesto izraza $2 \log(\mathbf{g}_o * \overline{\mathbf{q}})$ iz enačbe (12). Slika 2 dokazuje, da predlagani pristop konvergira k privlačni točki $\mathbf{g}_o$ bistveno hitreje kot pristop iz [9] (glej sliko 3). To je zaželena lastnost za dinamične sisteme. Delno je



Slika 2: Odziv DGG-ja (12), (13) brez nelinearnega člena $\mathbf{f}_o$. Zgornji graf kaže časovni potek štirih trajektorij kvaternionskih komponent, medtem ko spodnji graf prikazuje časovne trajektorije treh komponent kotne hitrosti.



Slika 3: Odziv DGG-ja (22), (13), pri katerem smo uporabili kvaternionsko razliko namesto logaritmične preslikave. Pri tem nismo uporabili nelinearnega člena $\mathbf{f}_o$.



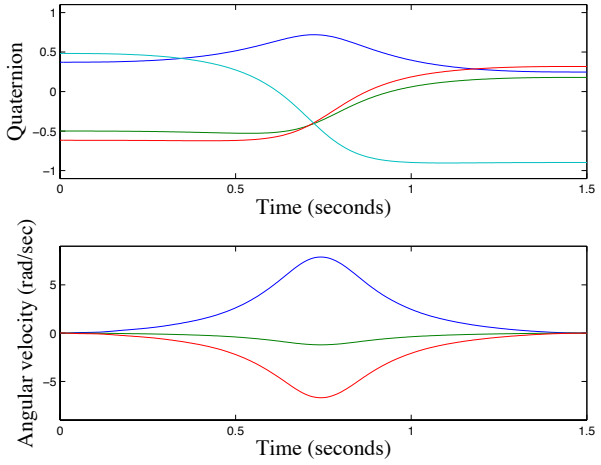
Slika 4: Odziv DGG-ja (38), (13) z dvakratno vektorsko kvaternionsko razdaljo namesto logaritmične preslikave. Pri tem nismo uporabili nelinearnega člena $\mathbf{f}_o$.

razlog za hitrejšo konvergenco pomanjkanje množenja s faktorjem 2 v enačbi (22). Če spremenimo (22) v

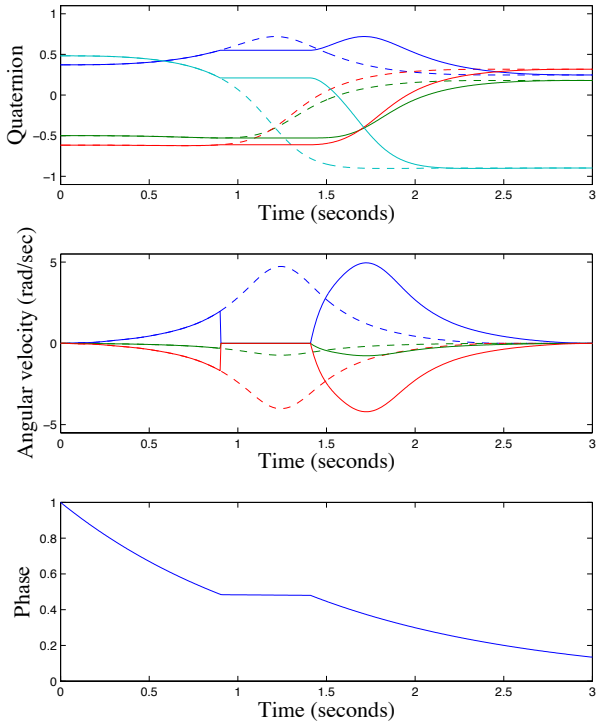
$$\tau\dot{\boldsymbol{\eta}} = \alpha_z\left(\beta_z 2\mathrm{vec}\left(\mathbf{g}_o * \overline{\mathbf{q}}\right) - \boldsymbol{\eta}\right) + \mathbf{f}_o(x), \qquad (38)$$

ugotovimo, da modificrani sistem (12), (13) še vedno generira bistveno hitrejši odziv kot (38), (13). Vendar pa je razlika v tem primeru manjša, kot je razvidno s slik 2 in 4. Pri tem je treba omeniti, da je 2 pravilni skalirni faktor za definicijo kritično dušenega sistema (12), (13). Če ta faktor še povečamo, sistem ni več kritično dušen in začne nihati proti ciljni orientaciji. Takšno obnašanje ni optimalno za robotsko vodenje.

V naslednjem eksperimentu smo preizkusili aproksimacijo želene trajektorije s kvaternionskim DGG-jem. Uporabili smo enake parametre kot v eksperimentih s slik 2 – 4, razen nekoliko modificirane časovne kon-

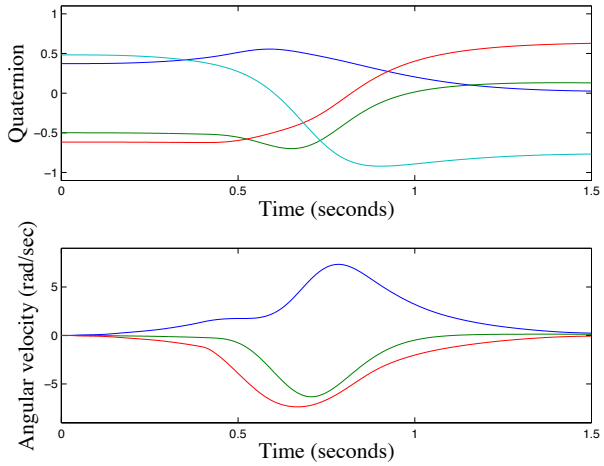


Slika 5: Reprodukcija želenih polinomskih orientacijskih trajektorij z minimalnim trzajem kaže dobro natančnost in s tem dokazuje, da lahko s kvaternionskimi DGG-ji (12), (13) natančno reproduciramo želene orientacijske trajektorije.

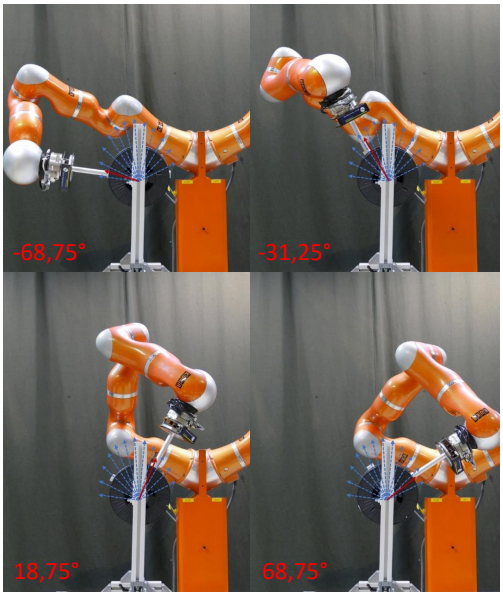


Slika 6: Vpliv zaustavitve poteka faze na potek kvaternionskega DGG-ja, ki ga definiramo z enačbami (25), (26), (27). Originalna trajektorija brez perturbacij je prikazana črtkano, medtem ko neprekinjene črte kažejo perturbiran gib, katerega izvedba je bila zaustavljena med 0.9 do 1.4 sekunde.

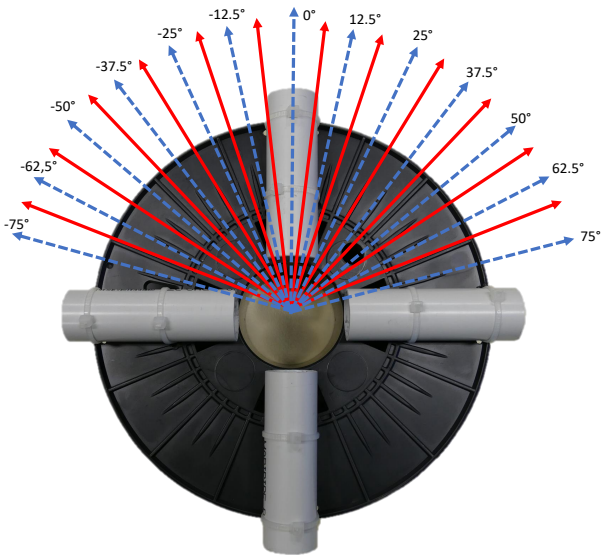
stante τ. Za generacijo vzorčne orientacijske trajektorije smo vzorčili polinom minimalnega trzaja med začetnim in končnim kvaternionom $\mathbf{q}_0$ in $\mathbf{g}_o$. Slika 5 kaže, da lahko s predlaganim pristopom natančno reproduciramo želene trajektorije.



Slika 7: Odziv DGG-ja (12), (13), (29) na nenadno spremembo ciljne orientacije za 30.4° med izvedbo giba. Perturbacija je nastopila pri $0.4\,s$. DGG konvergira proti novi orientaciji. Originalni gib brez perturbacij je prikazan na sliki 5.



Slika 8: Vrtenje ventila z robotom. Na sliki so prikazani robotski položaji pri kotih $\varphi = -68.75^\circ, -31.25^\circ, 18.75^\circ, 68.75^\circ$.



Slika 9: Ventil, ki ga vrti robot. Modra barva označuje kote, pri katerih je uporabnik demonstriral vrtenje ventila, medtem ko rdeča barva označuje poizvedbene kote za posploševanje.

5.1 *Ustavljanje faze in spreminjanje cilja*

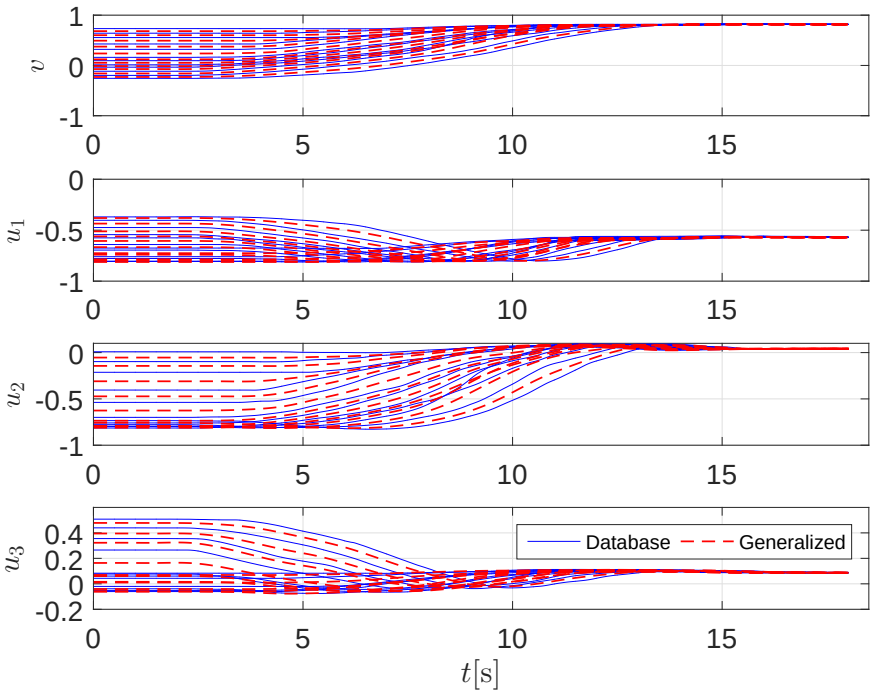
Slika 6 prikazuje postopke ustavljanja faze, ki je opisan v razdelku 3.1. Pri tem smo za krajši čast umetno ustavili sledenje robota želeni trajektoriji. Razen časovne konstante τ, ki smo jo postavili na 2.5, so bili vsi preostali parametri nastavljeni na enake vrednosti kot v prejšnjem eksperimentu. Ta slika kaže, da če gibanje robota ustavimo zaradi zunanje motnje, se v predlaganem sistemu zaustavi napredovanje faze za čas trajanja motnje. Ko je motnja odstranjena, se gibanje robota spet nadaljuje – podprto z dodatnim členom v enačbi (27) – in ostane zvezno. Medtem ko bi podobno trajektorijo lahko dobili tudi z bolj standardnimi pristopi, kot so na primer zlepki, lahko DGG-ji tovrstne spremembe upoštevajo brez dodatnega računanja in vodenja evidenc o poteku časa. Enostavna integracija enačb (25), (26) in (27) zadostuje za generacijo ustreznega odziva na "ustavitev gibanja".

Odziv sistema na spremembo ciljne orientacije je prikazan na sliki 7. V primerjavi z originalnim gibanjem na sliki 5, je originalna oblika gibanja dobro ohranjena kljub precejšnji spremembi ciljne orientacije. Pri tem ne preseneča, da spremenjena ciljna orientacija povzroča večje spremembe v kotni hitrosti kot v kvaternionski trajektoriji. V vseh naših eksperimentih je sistem (12), (13), (29) zanesljivo konvergiral k novemu cilju. Oblika gibanja je bila večinoma tudi ohranjena, vendar pa je bila splošna zanesljivost ohranjanja oblike orientacijske trajektorije nekoliko nižja kot pri pozicijskih trajektorijah. Glavni razlog za to je bolj zapletena geometrija $\mathrm{SO}(3)$ v primerjavi z evklidskim prostorom.

5.2 *Posploševanje kvaternionskih DGG-jev*

Eksperimentalno smo preizkusili tudi posploševanje kvaternionskih DGG-jev. V ta namen smo izbrali nalogo vrtenje ventila na sliki 9 od poljubnega začetnega kota do predpisanega končnega kota. Nabor vzorčnih trajektorij za posploševanje $\mathcal{B}$ smo pridobili s pomočjo človeških demonstracij, kjer je demonstrator pokazal 13 izvedb naloge, ki so se začele pri različnih začetnih kotih in končale pri istem končnem kotu. Začetni koti $[-75^\circ, -62,5^\circ, -50^\circ, -37,5^\circ, -25^\circ, -12,5^\circ, 0^\circ, 12,5^\circ, 25^\circ, 37,5^\circ, 50^\circ, 62,5^\circ, 75^\circ]$ so prikazani z modrimi puščicami na sliki 8 in so bili uporabljeni kot poizvedbene točke $b_i = \varphi_i$. Ti podatki so bili prvotno uporabljeni v [16].

Posneta baza vzorčnih trajektorij je na sliki 10 prikazana v modrem. Uporabili smo jo za sintezo



Slika 10: Baza demonstriranih kvaternionskih trajektorij (modra barva) in posplošene trajektorije (rdeča barva).

novih gibanj od izbranega začetnega kota ventila, ki je bil uporabljen kot poizvedbena točka pri posploševanju. Rezultati posploševanja po metodi iz razdelka 4 za vmesne poizvedbe $[-68.75°, -56.25°, -43.75°, -31.25°, -18.75°, -6.25°, 6.25°, 18, 75°, 31.25°, 43.75°, 56.25°, 68.75°]$ so na sliki 10 prikazani z rdečo barvo. S te slike je razvidno, da imajo posplošene kvaternionske trajektorije podobno obliko kot demonstrirane trajektorije in po pričakovanju potekajo med njimi. Učinkovitost metode smo potrdili z uspešnim robotskim vrtenjem ventila od začetne poizvedbene točke b do želenega končnega kota, pri čemer smo kot želene trajektorije uporabili posplošene trajektorije.

6 Zaključek

V tem članku obravnavamo specifikacijo pozicijskih in orientacijskih trajektorij s kartezičnimi dinamičnimi generatorji gibov. Med najpomembnejše zaključke spadajo:

- Kvaternionski dinamični generator gibov (12), (13) precej boljše konvergira proti dani privlačni točki $\mathbf{g}_o$ kot drugi sistemi diferencialnih enačb, ki temeljijo na vektorski kvaternionski razliki $\mathrm{vec}\,(\mathbf{g}_o * \overline{\mathbf{q}})$.
- Prav tako kot pri pozicijskih DGG-jih lahko tudi za orientacijske DGG-je formuliramo različne funkcionalnosti kot je zaustavitev faze, sprememba cilja in skaliranje trajektorij (glej razdelek 3).
- Opis kvaternionskih trajektorij z diferencialnimi enačbami omogoča posploševanje orientacijskih trajektorij ob upoštevanju ukrivljenosti prostora $\mathrm{SO}(3)$.

Dejstvo, da so prosti parametri $\mathbf{w}_i^o$ kvaternionskih DGG-jev iz enačb (12) in (19) tridimenzionalni, bistveno olajša procese aproksimacije, modulacije in posploševanja. V nasprotju s kvaternionsko interpolacijo [17], kjer moramo upoštevati omejitve, kot je norma enotnih kvaternionov, so parametri $\mathbf{w}_i^o$ brez omejitev, zato lahko pri njihovem računanju uporabljamo standardne optimizacijske metode brez omejitev. To v splošnem olajša implementacijo in zmanjša računsko zahtevnost.

Zahvala

Raziskavo je sofinancirala Javna agencija za znanstvenoraziskovalno in inovacijsko dejavnost Republike Slovenije v okviru programa P2-0076 – Avtomatika, robotika in biokibernetika.

Literatura

[1] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computations*, vol. 25, no. 2, pp. 328–373, 2013.

[2] S. Schaal, P. Mohajerian, and A. Ijspeert, "Dynamics systems vs. optimal control – a unifying view," *Progress in Brain Research*, vol. 165, no. 6, pp. 425–445, 2007.

[3] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, New York: CRC Press, 1994.

[4] S. Chiaverini and B. Siciliano, "The unit quaternion: A useful tool for inverse kinematics of robot manipulators," *System Analysis, Modelling and Simulation*, vol. 35, pp. 45–60, 1999.

[5] J. J. Faraway and S. B. Choe, "Modelling orientation trajectories," *Statistical Modeling*, vol. 9, no. 1, pp. 51–68, 2009.

[6] A. Ude, B. Nemec, T. Petrič, and J. Morimoto, "Orientation in Cartesian space dynamic movement primitives," in *IEEE International Conference on Robotics and Automation (ICRA)*, Hong Kong, 2014, pp. 2997–3004.
[7] A. Ude, "Filtering in a unit quaternion space for model-based object tracking," *Robotics and Autonomous Systems*, vol. 28, no. 2-3, pp. 163–172, 1999.
[8] A. M. Sabatini, "Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 7, 2006.
[9] P. Pastor, L. Righetti, M. Kalakrishnan, and S. Schaal, "Online movement adaptation based on previous sensor experiences," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, San Francisco, California, 2011, pp. 365–371.
[10] J. S.-C. Yuan, "Closed-loop manipulator control using quaternion feedback," *IEEE Journal of Robotics and Automation*, vol. 4, no. 4, pp. 434–440, 1988.
[11] A. J. Ijspeert, J. Nakanishi, T. Shibata, and S. Schaal, "Nonlinear dynamical systems for imitation with humanoid robots," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Tokyo, Japan, 2001, pp. 219–226.
[12] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-specific generalization of discrete and periodic dynamic movement primitives," *IEEE Transactions on Robotics*, vol. 26, no. 5, pp. 800–815, 2010.
[13] B. Nemec, M. Simonič, and A. Ude, "Learning of exception strategies in assembly tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 6521–6527.
[14] C. G. A. A. W. Moore and S. Schaal, "Locally weighted learning," *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 11–73, 1997.
[15] S. M. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.
[16] A. Kramberger, A. Gams, B. Nemec, D. Chrysostomou, O. Madsen, and A. Ude, "Generalization of orientation trajectories and force-torque profiles for robotic assembly," *Robotics and Autonomous Systems*, vol. 98, pp. 333–346, 2017.
[17] X. Niu and T. Wang, "C2-continuous orientation trajectory planning for robot based on spline quaternion curve," *Assembly Automation*, vol. 38, no. 3, pp. 282–290, 2018.

Aleš Ude je diplomiral iz uporabne matematike na Fakulteti za naravoslovje in tehnologijo Univerze v Ljubljani ter doktoriral iz inženirskih znanosti na Fakulteti za informatiko Univerze v Karlsruheju v Nemčiji. Je vodja Odseka za avtomatiko, biokibernetiko in robotiko na Institutu Jožef Stefan. Njegove raziskave se nanašajo na različna področja robotike, kot so robotsko učenje, rekonfigurabilne robotske celice, humanoidna robotika in robotske aplikacije v industriji. Na teh področjih je bil vodilni raziskovalec v številnih nacionalnih in mednarodnih projektih, nekatere od njih pa je tudi koordiniral.