Posnemanje človeških gibov s humanoidnim robotom HOAP

Roman Hribar, Rok Vuga

Laboratorij za humanoidno in kognitivno robotiko, Odsek za avtomatiko, biokibernetiko in robotiko
Institut "Jožef Stefan"
Jamova 39, 1000 Ljubljana, Slovenija
E-pošta: roman.hribar@ijs.si

Abstract

We have developed on-line full body imitation with a humanoid robot, based on prioritized task control. The method allows for real-time simultaneous control of balance and transfer of motion from a human demonstrator to the robot. Furthermore, self-collision avoidance is included in the control loop. In the paper we give a detailed description of all steps of the algorithm. The method was implemented in SL simulation software as well as on humanoid robot HOAP, while human motion was capture using Kinect camera.

1 Uvod

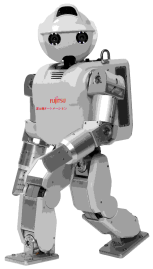
Prenos človeškega gibanja na humanoidnega robota je lahko izveden na različne načine. Človeško gibanje lahko zajemamo z najrazličnejšimi senzorji, ki jih pritrdimo na človeka, ali pa s pomočjo najrazličnejših kamer. Zaradi različnih kinematičnih in dinamičnih lastnosti človeka in robotskega mehanizma, ne moremo gibov prenesti neposredno iz enega na drugega. Če bi te posnete gibe s človeka poslali direktno na robota, ta ne bi obstal na nogah.

Eden izmed najpomembnejših kriterijev pri robotski stabilnosti je točka ničelnega navora oziroma ZMP [7]. Dvonožni robot je dinamično stabilen, če ZMP leži znotraj podpornega poligona, ki je določen s površino obeh podplatov nog. Ob predpostavki, da robot ne izvaja sunkovitih premikov težišča, lahko ZMP aproksimiramo kot projekcijo težišča na tla.

Nekaj metod, ki ohranjajo stabilnost robota je bilo že razvitih, a je večina teh primernejših za posnemanje človeške hoje. Metodo za načrtovanje trajektorij, ki minimizira napako med željenim in dejanskim ZMP-jom, je uporabil Kajita [8]. Aproksimiranje gibanja točke ZMP pri določenem gibanju težišča z modelom inverznega nihala je predstavil Sugihara [11]. Montecillo-Puente in sod. so uporabili metodo prioritete nalog za nadzor stabilnosti pri posnemanju človeških gibov, pri čemer so držo človeka modelirali s pomočjo normaliziranega modela [3].

Namen članka je predstaviti ohranjanje stabilnosti v povezavi s sistemom za zajemanje gibanja, ki bi generiral reprodukcijo človeških gibov v realnem času. Nove generacije humanoidnih robotov so pogosto kinematično redundantne in imajo ponavadi več kot dvajset prostostnih stopenj, ki jih lahko učinkovito uporabimo za ohranjanje stabilnosti robota, medtem ko robot opravlja kakšno drugo nalogo. Za primer nadzorovanja stabilnosti robota je naloga razdeljena na primarno in sekundarno [4]. Primarna je v tem primeru gibanje projekcije težišča znotraj podpornega poligona, medtem ko je sekundarna, ki je vključena v ničelni prostor primarne naloge, posnemanje gibanja človeka. V primarni del je vključena tudi naloga izogibanja, ki je enakovredna ohranjanju stabilnosti in nadrejena posnemanju gibanja.

Algoritem lahko uporabimo pri imitiranju humanoidnega robota v realnem času, ki hkrati stoji na dveh nogah. Za izvedbo sledenja človeških gibov smo uporabili RGB-D senzor Kinect. Aplikacija je narejena tako, da lahko gibanje človeka prenašamo samo v simulacijo ali pa sočasno v simulacijo in na realnega robota.



Slika 1: Hunanoidni robot HOAP, ki ga je razvilo podjetje Fujitsu z Japonske.

Krmiljenje težišča robota se izvaja na podlagi kinematičnega in dinamičnega modela. V poglavju 2 je predstavljen algoritem za izračun kinematike robota, ki je kasneje potrebna za določitev težišča. Glavna vsebina tretjega poglavja je izračun središa mase celotnega sistema in izogibanje rok robota. Poglavje se zaključi z opisom algoritma nadrejene in podrejene naloge. V zadnjem poglavju je podan opis humanoidnega robota HOAP in simulacije SL.

2 Kinematika humanoidnega robota

Pri računanju kinematike humanoidnega robota moramo upoštevati, da robot ni pritrjen, kot so pritrjeni običajni

industrijski manipulatorji. V našem primeru ima humanoidni robot omejeno gibanje podplatov, saj smo predpostavili, da je njihova hitrost glede na tla nič. Iz tega lahko sklepamo, da je robot pritrjen na podporno ploskev.

Humanoidnega robota lahko modeliramo kot kombinacijo petih kinematičnih verig. Po dve roki in nogi predstavljata štiri kinematične verige in glava eno (slika 1). Vse verige izhajajo iz enake začetne točke in jo imenujemo baza, ki se v našem primeru nahaja v trebuhu robota.

Vsaka veriga predstavlja en serijski mehanizem. Za opis kinematike serijskih mehanizmov pa lahko uporabimo vektorske ali DenavitHartenbergove parametre [10].

Da bi robotski mehanizem opisali z vektorskimi parametri, smo najprej robotski mehanizem postavili v neko smiselno začetno lego. Lokalne koordinatne sisteme smo postavili v center sklepov in jih orientirali enako kot referenčni koordinatni sistem. Sklepni vektor $\vec{e}_i$ je enotski vektor, s katerim je podana os rotacije in translacije sklepa *i* . Segmentni vektor $\vec{b}_{i-1}$ predstavlja velikost in smer od sklepa *i-1* do sklepa *i*. Segmentni vektor $\vec{b}_n$ (to je zadnji vektor v verigi) povezuje zadnji sklep s končno točko. Za razliko od DH parametrov, kjer moramo za vsak sklep določiti 4 parametre, moramo v tem primeru definirati 6 parametrov (tri za *e* in tri za *b*). Transformacijska matrika iz predhodnega λ v *i*-ti koordinatni sistem je definirana kot

$$^{\lambda(i)}H_i = \begin{bmatrix} rotvec(\vec{e}_i, \varphi) & \vec{b}_{i-1} \\ 0 & 1 \end{bmatrix}, \tag{1}$$

kjer rotvec($\vec{e}_i$,φ) predstavlja rotacijsko matriko okoli vektorja $\vec{e}_i$ za kot φ. Direktno kinematiko vsake od petih verig humanoidnega robota izračunamo

$$^0H_n = ^{\lambda(1)}H_1^{\lambda(2)}H_2 ... ^{\lambda(n)}H_n, \tag{2}$$

kjer n ponazarja število za ena večje kot je število sklepov (ena moramo dodati, da dosežemo vrh verige). Homogena matrika ^0H_n predstavlja transformacijo iz baznega koordinatnega sistema v vrh verige.

2.1 Jacobijeva matrika

Jacobijeva matrika je uporabljena v zaprti zanki programa in jo pri vsaki novi iteraciji na novo izračunamo. Matrika predstavlja relacijo med hitrostmi zunanjih in notranjih koordinat (enačba (3))

$$\dot{x} = J(q)\dot{q}. \tag{3}$$

Matrika zajema odvode zunanjih koordinat na notranje koordinate.

V enačbah (1) in (2) smo definirali homogene matrike, ki predstavljajo transformacijo iz baznega koordinatnega sistema v koordinatni sistem vrha verige. Za vsako izmed verig lahko zapišemo Jacobijevo matriko:

$$J = \begin{bmatrix} J_p \\ J_\omega \end{bmatrix}. \tag{4}$$

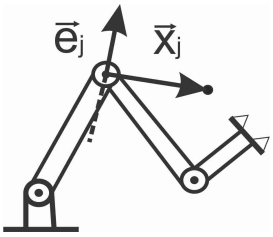
J_p predstavlja pozicijski del Jacobijeve matrike, J_ω pa orientacijski del Jacobijeve matrike. Pozicijski del matrike lahko dobimo iz naslednje enačbe

$$J_p^j = R_j(\vec{e}_j \times \vec{x}_j), \tag{5}$$

pri čemer J_p^j predstavlja *j*-ti stolpec pozicijskega dela Jacobijeve matrike, R_j rotacijski del homogene matrike ^0H_j, $\vec{e}_j$ vektor rotacije okoli sklepa *j* in $\vec{x}_j$ vektor med sklepom in točko, katere Jacobijevo matriko želimo izračunati. Orientacijski del matrike dobimo iz naslednje enačbe

$$J_\omega^j = R_j\vec{e}_j, \tag{6}$$

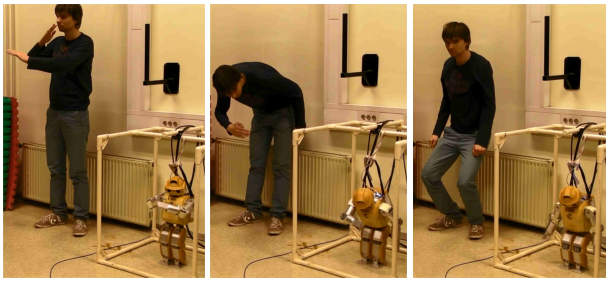
kjer J_ω^j označuje *j*-ti stolpec orientacijskega dela Jacobijeve matrike.



Slika 2: Skica za lažjo predstavo kako izračunati posamezne stoplce Jacobijeve matrike.

3 Posnemanje gibanja s stabilnostjo in izogibanjem

Naloga algoritma je omogočiti imitiranje človeka in hkrati preprečiti, da bi humanoidni robot padel. Človek podzavestno pri opravljanju različnih nalog na prvo mesto postavi stabilnost. Zato smo tudi mi za primarno nalogo izbrali ohranjanje stabilne lege in kot sekundarno imitiranje gibov demonstratorja, ki jih posnamemo s pomočjo Kinect senzorja. Povezava med hitrostjo težišča v koordinatnem sistemu baze in kotnimi hitrosti v sklepih $\dot{q}$ je podana z Jacobijevo matriko težišča J_{CoM}.



Slika 3: Zaporedje slik, ki prikazujejo imitiranje človeških gibov v realnem času. Demonstrator izvaja različne gibe, kot so mahanje z rokami, sklanjanje in počepanje.

3.1 Jacobijeva matrika težišča

Jacobijeva matrika središča mase celotnega sistema J_{CoM} lahko dobimo iz enačb

$$x_{CoM} = \frac{\sum_{i=1}^{n} m_i x_i}{\sum_{i=1}^{n} m_i} \tag{7}$$

in

$$ {}^{b}x_{CoM} = \frac{\sum_{i=1}^{n} m_i \, {}^{b}J_i \dot{q}}{\sum_{i=1}^{n} m_i} = \frac{\sum_{i=1}^{n} m_i \, {}^{b}J_i}{\sum_{i=1}^{n} m_i} \dot{q} = {}^{b} J_{CoM} \dot{q}, \tag{8} $$

pri katerih ${}^{b}J_i$ označuje Jacobijevo matriko središča mase *i*-tega dela robotskega segmenta v baznih koordinatah.

Če nimamo na voljo dinamičnega modela robota, lahko težišče humanoidnega robota računamo tako, da preverimo, za koliko rotacija vsakega sklepa premakne težišče posameznega segmenta. Ker prvi sklep v verigi premika celotno maso verige, zadnji pa le maso zadnjega segmenta, izračun težišča za posamezno verigo poteka od j-tega do prvega segmenta. Gre za rekurzivni način izračuna težišča od konca verige do začetka verige. Ko izračunamo posamezna težišča verig, lahko celotno težišče robota izračunamo po enačbi (7).

Enačba (8) označuje Jacobijevo matriko težišča i-tega dela telesa v baznih koordinatah. Če robot stoji na dveh nogah in so stopala vseskozi v stiku s tlemi velja, da so kotne in obodne hitrosti levega in desnega stopala enake nič, to so $\dot{x}_R = \dot{\omega}_R = 0$, $\dot{x}_L = \dot{\omega}_L = 0$. Ker so te hitrosti zapisane v zunanjih koordinatah, lahko ${}^{b}J_{CoM}$ transformiramo iz baznega v koordinatni sistem desne ali leve noge [11].

Da bi ohranili drugo stopalo v stiku s tlemi, moramo dodati omejitev, ki preprečuje premikanje levega stopala

$$ J_L \dot{q}_{LW} = 0. \tag{9} $$

V enačbi (9) je J_L Jacobijeva matrika leve noge v zunanjem koordinatnem sistemu in q_{LW} sklepi, ki potekajo po verigi od desnega proti levemu stopalu. Ker so vse Jacobijeve matrike izračunane v baznem koordinatnem sistemu (kinematične verige izhajajo iz trebuha), moramo izdelati Jacobijevo matriko, ki definira relacijo sklepov med vrhom stopala leve noge in vrhom stopala desne noge [1]. Transformacija je podana kot

$$ J_L = \begin{bmatrix} -R_R \Omega (x_L - x_R) J_{\omega R} - R_R^T J_{pR} & R_R^T \\ -R_R^T J_{\omega R} & R_R^T J_{\omega R} \end{bmatrix} \tag{10} $$

$$ q_{LW} = \begin{bmatrix} q_R \\ q_L \end{bmatrix}. \tag{11} $$

V enačbi (10) spremenljivki x_L in x_R označujeta pozicijo obeh stopal, Ω pa poševno simetrično matriko.

Če vzamemo v obzir, da robot stoji na obeh stopalih, lahko hitrost težišča glede na gibanje sklepov robota izrazimo kot

$$ \dot{x}_r = J_r \dot{q}, \tag{12} $$

kjer indeks r predstavlja razširjeno hitrost v zunanjih koordinatah oziroma razširjeno Jacobijevo matriko [2]. Za nalogo ohranjanja težišča in omejitve gibanja obeh stopal lahko zapišemo

$$ \dot{x}_r = \begin{bmatrix} \dot{x}_{CoM} \\ 0 \end{bmatrix}, \tag{13} $$

pri čemer $\dot{x}_{CoM}$ ponazarja kontrolno veličino za hitrost težišča v *x* in *y* smeri in je definirana kot $\dot{x}_{CoM} = k(x_{CoM}^{des} - x_{CoM}^{act})$, pri čemer je x_{CoM}^{act} trenutni položaj težišča, x_{CoM}^{des} pa položaj točno nad središčem podpornega poligona. K je pozitivna konstanta. V enačbi (13) ničla predstavlja kotne in obodne hitrosti levega stopala. Razširjeno Jacobijevo matriko zapišemo:

$$ J_r = \begin{bmatrix} J_{CoM} \\ J_L \end{bmatrix}. \tag{14} $$

V nadaljevanju bomo v razširjeno Jacobijevo matriko J_r in raširjene zunanje koordinate x_r dodali še nalogo izogibanja rok robota.

3.2 Izogibanje trkom

Pri prenosu gibanja s človeka na robota moramo upoštevati nepredvidljivost naloge. Zaradi razlik med kinematičnima strukturama človeka in robota, napak v zaznavanju ali enostavne nepazljivosti lahko gibanje privede do trka med posameznimi robotskimi segmenti. Za izvedbo izogibanja trkom smo uporabili poenostavljeno metodo usmerjene Jacobijeve matrike [5]. Za vsak robotov segment najprej določimo t.i. kritično točko A in vektor izogibanja $\vec{w}$ (slika 4). Kritična točka predstavlja točko na segmentu, ki je najbljižje oviri, t.j. nekemu drugemu segmentu. Vektor izogibanja pa je enotski vektor, ki kaže smer proč od ovire.

Metoda deluje tako, da željeni hitrosti, ki jo določa naloga, prištejemo hitrost izogibanja. To določimo tako, da je nasprotno enaka komponenti hitrosti naloge, ki kaže v smeri proti oviri. Če je hitrost naloge že sama po sebi usmerjena stran od ovire, je hitrost izogibanja enaka nič. Na ta način so blokirani vsi premiki proti oviri, gibanje v vseh ostalih smereh pa je neomejeno.

Formalno hitrost izogibanja zapišemo kot:

$$ \dot{x}_{av} = \vec{w}(\vec{w} \cdot (J_{av} \dot{q_n})) f_v f_p, \tag{15} $$

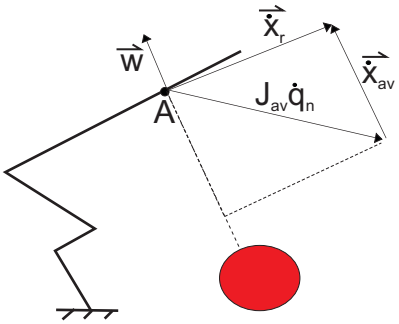
pri čemer je J_{av} Jacobijeva matrika kritične točke, q_n pa hitrost naloge. V našem primeru $\dot{q_n}$ predstavlja primarno nalogo, f_v in f_p sta faktorja, ki določata aktivacijo izogibanja glede na bližino ovire in smer gibanja naloge. Najenostavneje se lahko določita binarno:

$$ f_p = \begin{cases} 1 & \text{če } d < d_{min} \\ 0 & \text{sicer} \end{cases}, \tag{16} $$

$$ f_v = \begin{cases} 1 & \text{če } \vec{w} \cdot (J_{av} \dot{q_n}) < 0 \\ 0 & \text{sicer} \end{cases}. \tag{17} $$

Pri tem je faktor bližine f_p določen tako, da je enak ena samo takrat, ko je obravnavani segment dovolj blizu najbližji oviri. Faktor hitrosti f_v pa je enak ena takrat, ko hitrost naloge vsebuje komponento v obratni smeri vektorja izogibanja; v nasprotnem primeru popravek ni potreben.

Zaradi varnosti mora biti izogibanje in ohranjanje stabilnosti nadrejeno sledenju demonstratorja. Hitrosti izogibanja segmentov dodamo kot enakovredne naloge v



Slika 4: Izogibanje trkom na podlagi modificiranja hitrosti naloge. Rdeča krogla na sliki predstavlja oviro. A označuje kritično točko na robotu, $\vec{w}$ vektor izogibanja, $J_{av}\dot{q}_n$ hitrost kritične točke kot posledico naloge, $\dot{x}_{av}$ pa dobljeno hitrost izogibanja. $\dot{x}_r$ ponazarja rezultirajoči premik kritične točke po modifikaciji naloge.

razširjeno Jacobijevo matriko iz enačb (13) in (14):

$$\dot{x}_r = \begin{bmatrix} \dot{x}_{CoM} \\ 0 \\ \dot{x}^1_{av} \\ \dot{x}^2_{av} \\ \dots \\ \dot{x}^N_{av} \end{bmatrix}, \begin{bmatrix} J_{CoM} \\ J_R \\ J^1_{av} \\ J^2_{av} \\ \dots \\ J^N_{av} \end{bmatrix}, \tag{18}$$

pri čemer je N število segmentov (v našem primeru je N enak 6), ki se aktivno izogibajo trkom.

3.3 Skupna rešitev primarne in sekundarne naloge

Posnemanje gibanja človeka lahko izrazimo tako, da nalogo razdelimo na primarno in sekundarno. V našem primeru smo ohranjanje ravnotežja in izogibanje rok uporabili kot primarno nalogo, medtem ko smo v ničelni prostor (sekundarna naloga) postavili imitiranje gibanja človeka. Takšen problem lahko rešimo z metodo prioritet nalog [4]:

$$\dot{q} = J_r^+ \dot{x}_r + N\dot{q}_{KIN}, \tag{19}$$

$$\dot{q}_{KIN} = k_p(q_{trenutni} - q_{KIN}). \tag{20}$$

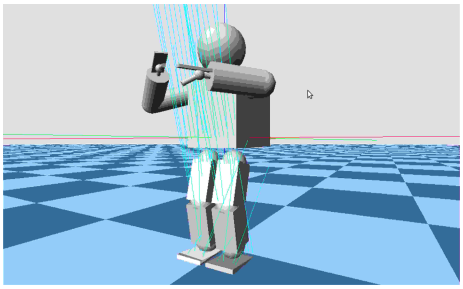
V enačbi (19) z $N = (I - J_r^+ J_r)$ definiramo ničelni prostor od J_r, $\dot{q}_{KIN}$ predstavljajo željene hitrosti v sklepih. S Kinect senzorja sprejemamo željene pozicije sklepov, ki jih s pomočjo enačbe (20) preračunamo v $\dot{q}_{KIN}$. V enačbi (20) $q_{trenutni}$ označujejo trenutne pozicije robota, q_{KIN} pozicije, ki jih prebere Kinect, k_p pa pozitivno ojačanje.

4 Humanoidni robot HOAP in simulacija SL

HOAP je kompakten lahek humanoidni robot. Visok je 600 mm, tehta 8.8 kg in ima 28 prostostnih stopenj, od tega jih ima po 6 v vsaki roki in nogi ter 3 v glavi in eno v trebuhu. Pri aplikaciji smo jih uporabili le 21, med njimi nismo vključili zapestij in prstov obeh rok ter sklep glave. 6 prostostnih stopenj v nogi je razdeljenih po naslednjem vrstem redu: 3 v kolku, 1 v kolenu in 2 v gležnju. V roki si stopnje sledijo: 3 v rami, 1 v komolcu, 1 v zapestju in 1 v prstih.

Poleg zgoraj omenjenih lastnosti robota je robot zmožen tudi razpoznavati in sintetizirati govor ter razpoznavati gibanje. V glavi ima vgrajene CCD kamere, mikrofon in zvočnik v podplatih pa senzorje sile.

SL je dinamični smimulator, cilj uporabe simulacije je realizirati in testirati algoritme, ki so namenjeni premikanju dvonožnih robotov. SL omogoča izdelovanje dinamičnih simulacij togega telesa hitro in enostavno, hkrati pa lahko robotski algoritem testiramo v simulaciji preden ga izvršimo na realnem robotu. Okolje simulacije je narejeno (slika (5)) tako, da lahko algoritem programiramo v enostavni C kodi. Ko programska koda deluje zadovoljivo, zaženemo datoteko, ki omogoča sočasno delovanje simulacije in realnega robota [9].



Slika 5: Simulacijsko okolje SL, ki je namenjeno simulaciji humanoidnih robotov.

5 Zaključek

V članku smo opisali metodo za prenos gibanja s človeka na humanoidnega robota z upoštevanjem stabilnosti in izogibanjem trkom. Algoritem deluje na podlagi metode prioritete nalog in razširjene Jakobijeve matrike. Stabilnost je dosežena na podlagi vodenja hitrosti robotovega težišča tako, da reguliramo njegov položaj znotraj podpornega poligona. Tak način zagotavlja statično stabilnost robota; ob zelo sunkovitih gibih se lahko zgodi, da kljub pravilnemu položaju težišča točka ničelnega navora zapusti podporni poligon in robot pade. Poleg tega referenčni položaj težišča računamo ob predpostavki, da so tla vodoravna. Če niso, algoritem ni zmožen zagotavljati stabilne lege.

V prihodnje želimo v algoritem vključiti tudi gibanje zapestja, prstov in glave. Da bi dosegli željen cilj, moramo uporabiti Kinect V2.0, ki omogoča zajemanje večih prostostnih stopenj.

Literatura

[1] A. Gams, J. van den Kieboom, F. Dzeladini in A.J.Ijspeert: Stable real-time full body imitation on the COMAN humanoid robot, 2013, Portorož, Slovenija

[2] B. Siciliano in O. Khatib: Springer Handbook of Robotics, September 2008

[3] F.J. Montecillo-Puente, M.N. Sreenivasa, J.P. Laumond: On Real-time Whole-body Human to Humanoid Motion

Transfer, ICINCO 2010, 15.-18. Junij 2010, Madeira, Portugalska

[4] J. Lenarčič, T. Bajd: Robotski Mehanizmi, 2003, Ljubljana, Slovenija

[5] L. Žlajpah in B. Nemec: Kinematic Control Algorithms for On-Line Obstacle Avoidance for Redundant manipulators, Intelligent Robots and Systems 2002, 2002, Ljubljana, Slovenij

[6] M. Vukobratovič in B. Borovac: Zero-moment point - thirty five years its life, I.J. Humanoid Robotics, 2004

[7] M. Vukobratovič in D. Jurčič: Contribution to the synthesis of biped gait. Biomedical Engineering, IEEE Transcactions on, 1969

[8] S.Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Biped walking pattern gneration by using preview control of zero-moment point, ICRA 2003, 14.-19. September 2003, Taipei, Taivan

[9] S. Schaal: The SL Simulation and Real-Time Control Software Package, 2006, Los Angeles, ZDA

[10] T. Bajd, M.Mihelj, J. Lenarčič, A. Stanovnik, M. Munih: Robotics, Intelligent Systems, Control and Automation: Science and Engineering, 2010, Ljubljana, Slovenija

[11] T. Sugihara, Y. Nakamura in H. Inoue: Real time humanoid motion generation through ZMP manipulation based on iverted pendulum control, ICRA 2002, Maj 2002, Washington, DC, ZDA